

# Model Reduction of Nonlinear Dynamical Systems by System-Theoretic Methods

Maria Cruz Varona

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der  
Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender: Prof. dr. ir. Daniel J. Rixen

Prüfer der Dissertation: 1. Prof. Dr.-Ing. habil. Boris Lohmann  
2. Prof. Dr. Tatjana Stykel

Die Dissertation wurde am 18.06.2020 bei der Technischen Universität München  
eingereicht und durch die Fakultät für Maschinenwesen am 29.10.2020 angenommen.





# Abstract

Numerical simulation is essential for the analysis, design and control of dynamical systems. Due to the ever increasing complexity and required accuracy, models described by a high number of differential equations often result during the modeling process. In many applications nonlinearities also arise, complicating the subsequent numerical analysis even further. In order to reduce the computational effort and make the simulation more efficient, model reduction can be applied to obtain a considerably smaller model approximating the original one.

Widely employed techniques in nonlinear model reduction rely on expensive training simulations of the original model to construct the reduced model. By contrast, the present work discusses simulation-free reduction procedures based on system-theoretic concepts rather than on simulated data. In this sense, the thesis covers the reduction of polynomial, nonlinear state-space and nonlinear second-order systems from a system-theoretic perspective.

The Volterra series representation allows to derive multivariable kernels and transfer functions for polynomial systems (e.g. bilinear, quadratic-bilinear). Based thereof, well-known system-theoretic measures and model reduction approaches can be generalized from linear to polynomial systems. In this thesis, we particularly focus on the interpolation-based reduction of bilinear systems using the transfer functions and the concept of moment matching via Krylov subspaces. We generalize both existing interpolation frameworks to the multiple-input multiple-output case and extend  $\mathcal{H}_2$ -pseudo-optimal reduction to the bilinear setting.

In the context of nonlinear state-space systems model reduction can be accomplished using a linear or nonlinear projection. We discuss the corresponding framework, reduction techniques and properties of both strategies. Then, the focus is laid on the existing generalization of moment matching to nonlinear systems based on signal generators and steady-state considerations. Since the method involves the solution of a Sylvester partial differential equation, we propose certain simplifications to achieve a feasible simulation-free algorithm relying on nonlinear systems of equations. The resulting scheme is extensively discussed and demonstrated via numerical examples. After that, connections between polynomial and nonlinear model reduction by moment matching are established to offer a unifying view.

In the field of nonlinear mechanical systems modal derivatives are very popular for basis augmentation and quadratic manifold reduction. We first revisit the original derivation of modal derivatives based on the perturbation of the linearized quadratic eigenvalue problem. Since this derivation is not system-theoretically substantiated, we present a novel derivation based on the Volterra series representation. The properties of the gained derivatives are discussed, as well as their applications for nonlinear system analysis and model reduction. Finally, we transfer the concept of moment matching to nonlinear second-order systems and provide the corresponding Sylvester partial differential equation. After similar simplifications, the performance of the resulting second-order nonlinear moment matching algorithm is demonstrated on geometrically nonlinear structural models.



# Acknowledgments

This thesis is the result of my research at the Technical University of Munich over the past five years. First of all, I want to thank my supervisor Prof. Boris Lohmann for giving me the opportunity to teach and do research at the Chair of Automatic Control. The support, trust and freedom he gave me during my time as research assistant laid the foundation for my work. I also want to express my sincere gratitude to Prof. Tatjana Stykel for being not only my second examiner, but also for innumerable fruitful discussions during our model reduction workshops. Furthermore, I would like to thank Prof. Daniel Rixen for chairing the examination committee, for his interest in my work and for the great collaboration we had in the realm of nonlinear structural mechanics.

Within the Computational Science and Engineering community I had the privilege to meet many incredible people who enriched me in one way or another. While I am deeply thankful to all of them, I want to especially mention the ones who coined and supported me the most. Many thanks to Ulrike Baur, Tobias Breiten, Pawan Goyal and Jens Saak for sharing their benchmark examples/code, as well as for their friendly mentoring throughout my thesis. I am also much obliged to Serkan Gugercin for giving me the opportunity to visit him at Virginia Tech. I really enjoyed my stay and benefited a lot from the discussions with him, Garret Flagg and Karen Willcox, among others. Moreover, I would like to gratefully thank Martin Pfaller for our amazing collaboration. His expertise in cardiac mechanics made the application of model reduction to a computational human heart model a very enjoyable and successful experience. I also want to thank Johannes Rutzmoser and Christian Meyer for the cooperative writing of our joint grant proposal, the development of AMfe and the inspiring discussions about modal derivatives. All this sparked my interest in model reduction for nonlinear structural dynamics.

Many thanks go also to all my former colleagues at the Chair of Automatic Control, especially to the MORLab members. I am grateful to Heiko Peuscher, Thomas Wolf and Matthias Geuß for introducing me into the research field of model reduction. It was also a pleasure to share the office with Alessandro Castagnotto, with whom I had countless discussions about the “sss” and “sssMOR” toolboxes. Furthermore, I want to thank Tim Moser for the friendly and supportive atmosphere during the last months of my PhD. Outside the MORLab, I would like to thank in particular Nils Pletschen, Klaus Albert and Mikhail Pak for the endless (off-topic) conversations, their collegial advice and friendship.

My deepest gratitude go also to all my students, who contributed valuable insights for my research during their theses. I had the privilege to teach them scientific practices and MOR knowledge, and got in turn enriched by very smart and motivated people.

Last but not least, I would like to thank my family and my boyfriend Jürgen for their unconditional love and emotional support throughout the last years. They always provided a comforting distraction from research and essentially make my life worth living.



# Contents

<b>List of Statements</b>	<b>xi</b>
<b>List of Figures</b>	<b>xiii</b>
<b>List of Algorithms</b>	<b>xiv</b>
<b>I PRELIMINARIES</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Challenges and overview of nonlinear model reduction . . . . .	4
1.2 Scope and objectives of the thesis . . . . .	5
1.3 Scientific contributions . . . . .	6
1.4 Outline of the thesis . . . . .	7
<b>2 Mathematical Fundamentals</b>	<b>9</b>
2.1 Tensor algebra . . . . .	9
2.2 Projection . . . . .	14
2.3 Krylov subspaces . . . . .	15
2.4 Numerical solution of linear matrix equations . . . . .	18
2.5 Numerical solution of linear systems of equations . . . . .	20
2.6 Error measures . . . . .	21
<b>3 Linear Model Order Reduction</b>	<b>25</b>
3.1 Linear time-invariant systems . . . . .	25
3.2 Projective model order reduction . . . . .	30
3.3 Overview of linear reduction methods . . . . .	32
3.3.1 Modal truncation . . . . .	32
3.3.2 Balanced truncation . . . . .	33
3.3.3 Moment matching . . . . .	35
3.3.4 Further approaches . . . . .	37
3.4 Frequency-domain interpretation of linear moment matching . . . . .	37
3.4.1 Frequency-domain notion of linear moments . . . . .	38
3.4.2 Moment matching by Krylov subspaces . . . . .	38
3.4.3 Discussion of different aspects . . . . .	40
3.5 Equivalence of Krylov subspaces and Sylvester equations . . . . .	42
3.6 Time-domain interpretation of linear moment matching . . . . .	45
3.6.1 Time-domain notion of linear moments . . . . .	46
3.6.2 Moment matching by interconnection . . . . .	49
3.6.3 Derivation of linear input Sylvester equation . . . . .	50

3.6.4	Families of reduced models achieving linear moment matching . . . . .	50
3.7	$\mathcal{H}_2$ -optimal reduction of linear systems . . . . .	51
3.8	Adaptive and cumulative rational Krylov subspace method . . . . .	53
II POLYNOMIAL NONLINEAR STATE-SPACE SYSTEMS		55
<b>4</b>	<b>Fundamentals of Polynomial Systems</b>	<b>57</b>
4.1	Obtaining a polynomial system . . . . .	57
4.1.1	Taylor series expansion . . . . .	58
4.1.2	Polynomialization procedure . . . . .	60
4.2	Transformation to special polynomial system classes . . . . .	61
4.2.1	Carleman bilinearization . . . . .	61
4.2.2	Quadratic-bilinearization . . . . .	63
4.3	Volterra series representation . . . . .	64
4.3.1	Picard iteration . . . . .	64
4.3.2	Variational equation approach . . . . .	66
4.4	Generalized transfer functions . . . . .	69
4.4.1	Multidimensional Laplace transform of the kernels and output equations	69
4.4.2	Growing exponential approach . . . . .	73
<b>5</b>	<b>Bilinear Systems</b>	<b>79</b>
5.1	Bilinear systems theory . . . . .	79
5.2	Bilinear model order reduction . . . . .	87
5.2.1	Bilinear balanced truncation . . . . .	89
5.2.2	Bilinear Krylov-based reduction . . . . .	90
5.3	Subsystem interpolation . . . . .	92
5.3.1	MIMO subsystem interpolation . . . . .	94
5.3.2	Flexibility and combinatorial problem . . . . .	97
5.4	Volterra series interpolation . . . . .	98
5.4.1	MIMO Volterra series interpolation . . . . .	102
5.4.2	Multimoment Volterra series interpolation . . . . .	106
5.4.3	Other cases and complete Arnoldi algorithm . . . . .	108
5.5	$\mathcal{H}_2$ -optimal reduction of bilinear systems . . . . .	112
5.5.1	Optimality conditions and bilinear iterative algorithms . . . . .	112
5.5.2	$\mathcal{H}_2$ -pseudo-optimality for bilinear systems . . . . .	114
5.6	Conclusions and other contributions . . . . .	115
III NONLINEAR STATE-SPACE SYSTEMS		117
<b>6</b>	<b>Fundamentals of Nonlinear Model Reduction</b>	<b>119</b>
6.1	Nonlinear time-invariant systems . . . . .	119
6.2	Time integration . . . . .	120
6.3	Nonlinear reduction based on linear projection . . . . .	122
6.3.1	Linear Petrov-Galerkin projection framework . . . . .	122
6.3.2	Time integration . . . . .	122
6.3.3	Linear projection-based reduction approaches . . . . .	123

6.4	Nonlinear reduction based on nonlinear projection . . . . .	127
6.4.1	Nonlinear Petrov-Galerkin projection framework . . . . .	127
6.4.2	Time integration . . . . .	129
6.4.3	Nonlinear projection-based reduction approaches . . . . .	130
6.5	Hyper-reduction . . . . .	132
6.5.1	Polynomial system representation . . . . .	133
6.5.2	Piecewise linear approximation . . . . .	134
6.5.3	Classical hyper-reduction . . . . .	134
6.5.4	Nonlinear manifold-based hyper-reduction . . . . .	137
6.6	Discussion . . . . .	137
<b>7</b>	<b>Model Reduction by Approximated Nonlinear Moment Matching</b>	<b>139</b>
7.1	Steady-state-based nonlinear moment matching . . . . .	140
7.1.1	Time-domain notion of nonlinear moments . . . . .	140
7.1.2	Nonlinear moment matching by interconnection . . . . .	142
7.1.3	Derivation of nonlinear Sylvester-like partial differential equation . . . . .	143
7.1.4	Families of reduced models achieving nonlinear moment matching . . . . .	144
7.2	Approximated nonlinear moment matching . . . . .	144
7.2.1	Simplifications . . . . .	145
7.2.1.1	Nonlinear signal generator . . . . .	145
7.2.1.2	Linear signal generator . . . . .	146
7.2.1.3	Zero signal generator . . . . .	147
7.2.2	Simulation-free nonlinear moment matching algorithm . . . . .	147
7.2.2.1	Proposed algorithm . . . . .	147
7.2.2.2	Computational aspects . . . . .	147
7.2.2.3	Approximated nonlinear moments . . . . .	149
7.2.3	Families of reduced models achieving approximated moment matching . . . . .	150
7.3	Analysis, discussion and limitations . . . . .	150
7.4	Guidelines for practitioners . . . . .	153
7.4.1	Training vs. test phase . . . . .	154
7.4.2	Examples for signal generators . . . . .	154
7.4.3	Selection of shifts and time-snapshots . . . . .	156
7.4.4	Time integration scheme . . . . .	157
7.5	Numerical examples . . . . .	158
7.5.1	Chafee-Infante equation . . . . .	159
7.5.2	FitzHugh-Nagumo model . . . . .	161
7.6	Further remarks . . . . .	165
7.6.1	Applicability to large finite element models . . . . .	165
7.6.2	Simulation-free hyper-reduction . . . . .	165
7.7	Conclusions . . . . .	166
<b>8</b>	<b>Bridging the Gap between Polynomial and Nonlinear Model Reduction</b>	<b>167</b>
8.1	Eigenfunctions of dynamical systems . . . . .	167
8.2	Eigenfunction of bilinear and quadratic-bilinear systems . . . . .	168
8.3	Eigenfunctions for nonlinear moment matching . . . . .	171
8.4	Signal generator interpretation of Volterra series interpolation . . . . .	173

IV	NONLINEAR SECOND-ORDER SYSTEMS	177
<b>9</b>	<b>Fundamentals of Model Reduction for Nonlinear Mechanical Systems</b>	<b>179</b>
9.1	System representation . . . . .	179
9.2	Time integration . . . . .	180
9.3	Nonlinear reduction based on linear projection . . . . .	181
9.4	Nonlinear reduction based on nonlinear projection . . . . .	186
<b>10</b>	<b>Novel Modal Derivatives for Nonlinear Analysis and Model Reduction</b>	<b>191</b>
10.1	Polynomial system representation . . . . .	191
10.2	Variational equations . . . . .	192
10.3	Derivation of modes and modal derivatives . . . . .	193
10.3.1	Modes (First subsystem) . . . . .	193
10.3.2	Modal derivatives (Second subsystem) . . . . .	194
10.4	Impact of the new modal derivatives . . . . .	196
10.4.1	Analytical solution via truncated Volterra series . . . . .	196
10.4.2	Novel quadratic manifold approaches for model reduction . . . . .	197
10.4.3	Connection to the Harmonic Balance method . . . . .	198
10.5	Numerical examination . . . . .	203
10.6	Conclusions . . . . .	209
<b>11</b>	<b>Second-Order Nonlinear Moment Matching</b>	<b>211</b>
11.1	Moment matching for linear mechanical systems . . . . .	211
11.1.1	Notion of moments and Krylov subspaces . . . . .	212
11.1.2	Equivalence of Krylov subspaces and Sylvester equations . . . . .	213
11.1.3	Time-domain interpretation of linear moment matching . . . . .	213
11.2	Steady-state-based second-order nonlinear moment matching . . . . .	215
11.3	Approximated second-order nonlinear moment matching . . . . .	216
11.4	Numerical examples . . . . .	218
11.4.1	Cantilever beam . . . . .	218
11.4.2	S-shaped structure . . . . .	221
11.5	Conclusions . . . . .	223
V	CLOSURE AND APPENDICES	225
<b>12</b>	<b>Concluding Remarks</b>	<b>227</b>
<b>A</b>	<b>Taylor series expansion of non-input-affine nonlinear systems</b>	<b>231</b>
<b>B</b>	<b>Additional material for novel modal derivatives</b>	<b>233</b>
	<b>Bibliography</b>	<b>237</b>
	Selected talks and workshops . . . . .	254
	Supervised student theses . . . . .	255
	Co-supervised student theses . . . . .	256



# List of Statements

2.1	Definition (Kronecker product)	9
2.1	Lemma	9
2.2	Definition (Vectorization)	10
2.2	Lemma	10
2.3	Definition (Tensor)	11
2.4	Definition ( $\mu$ -matricization of a tensor)	11
2.1	Example (Matricizations of a third-order tensor)	11
2.5	Definition ( $\mu$ -mode matrix product)	12
2.1	Corollary (Matricizations of a tensor defined by matrix products)	12
2.6	Definition (Symmetric tensors)	13
2.2	Example (Symmetrization of a quadratic system)	13
2.7	Definition (Projector)	14
2.3	Lemma	14
2.8	Definition (Orthogonal and oblique projection)	15
2.9	Definition (Krylov subspace)	16
2.10	Definition (Block rational Krylov subspace)	16
2.1	Remark (Output Krylov subspaces)	18
3.1	Definition (Laplace transform)	26
3.2	Definition (Controllability)	28
3.3	Definition (Observability)	28
3.4	Definition (Frequency-domain linear moments)	38
3.1	Theorem (Block <i>multimoment</i> matching)	38
3.1	Remark (Matching Markov parameters)	39
3.2	Theorem (Tangential <i>multipoint</i> moment matching)	39
3.1	Example (Making the basis real)	41
3.2	Remark (Solution of linear Sylvester equations)	45
3.5	Definition (Time-domain linear moments)	46
3.1	Lemma (Steady-state notion of linear input moments)	48
3.3	Remark (Global invariant manifold [14])	48
3.4	Remark (Steady-state perception for other cases)	49
3.5	Remark (Steady-state perception of output moments)	49
3.3	Theorem (Steady-state-based linear moment matching)	49
3.1	Corollary (Exact moment matching vs. interpolation)	49
4.1	Example (Polynomialization procedure)	60
4.2	Example (Quadratic-bilinearization procedure)	63
4.1	Definition (Multidimensional Laplace transform)	69
4.3	Example (Kronecker notation for regular transfer functions)	72

---

5.1	Example (Pole-residue formulation for bilinear systems)	80
5.1	Theorem (BIBO stability for bilinear systems)	81
5.2	Theorem (Convergence of bilinear Gramians)	83
5.1	Remark (Solution of bilinear Lyapunov equations vs. existence of Gramians)	83
5.2	Remark (Rescaling of bilinear system)	90
5.1	Definition (Multimoments of bilinear systems)	91
5.3	Theorem ( <i>Multimoment</i> subsystem interpolation (SISO))	92
5.3	Remark (Markov parameters in subsystem interpolation)	92
5.2	Example (Multimoment+multipoint subsystem interpolation (SISO))	93
5.4	Theorem (Tangential <i>multipoint</i> subsystem interpolation (MIMO-1))	94
5.3	Example (Projection matrix for MIMO-1 subsystem interpolation)	95
5.5	Theorem (Tangential <i>multipoint</i> subsystem interpolation (MIMO-2))	96
5.4	Example (Projection matrix for MIMO-2 subsystem interpolation)	97
5.6	Theorem ( <i>Multipoint</i> Volterra series interpolation (SISO))	99
5.5	Example ( <i>Truncated</i> Volterra series interpolation (SISO))	100
5.7	Theorem (Tangential <i>multipoint</i> Volterra series interpolation (MIMO))	102
5.6	Example	104
5.4	Remark (Solution of bilinear Sylvester equations vs. convergence of series)	104
5.7	Example (Multimoment+multipoint Volterra series interpolation)	107
6.1	Remark (Link between nonlinear and linear projection)	128
6.2	Remark (Power series ansatz [146, 124])	129
7.1	Lemma (Steady-state notion of nonlinear input moments)	141
7.1	Remark (Local invariant manifold [14])	142
7.1	Theorem (Steady-state-based nonlinear moment matching)	142
7.1	Corollary (Exact moment matching vs. interpolation)	143
7.2	Lemma (Approximated nonlinear moments)	149
7.1	Example (Complex shifts in NLMM)	155
9.1	Remark	187
11.1	Definition	212
11.1	Lemma (Steady-state notion of linear input moments)	214
11.1	Theorem (Steady-state-based linear moment matching)	214
11.1	Corollary	214
11.2	Lemma (Steady-state notion of nonlinear input moments)	215
11.2	Theorem (Steady-state-based nonlinear moment matching)	215
11.2	Corollary	216
A.1	Example (Quadratic-quadratic system)	232

# List of Figures

2.1	Geometric illustration of a projection . . . . .	14
3.1	Interconnection of linear FOM/ROM with linear signal generator . . . . .	48
4.1	Volterra series representation of a bilinear system . . . . .	67
7.1	Interconnection of nonlinear FOM/ROM with nonlinear signal generator . . . .	142
7.2	Chafee-Infante: Outputs and point-wise relative error for $u_{\text{test},1}(t) = 25 \sin(t)$ . .	160
7.3	Chafee-Infante: Outputs and point-wise relative error for $u_{\text{test},2}(t) = 25 (\sin(t) + 1)$	160
7.4	FHN model, spiking: Phase portrait, outputs and point-wise relative error . . .	163
7.5	FHN model, resting: Outputs and point-wise relative error . . . . .	164
10.1	Auto-MAC of the original/static/tilde/double tilde MDs for a cantilever beam	204
10.2	Cross-MAC of the original/static/tilde/double tilde MDs for a cantilever beam	204
10.3	Singular values of the different MDs for a cantilever beam . . . . .	205
10.4	Original MDs for a plate (scalefactor VM=20, original MDs=20) . . . . .	206
10.5	Static MDs for a plate (scalefactor VM=20, static MDs=100) . . . . .	206
10.6	Tilde MDs for a plate (scalefactor VM=20, tilde MDs=100) . . . . .	207
10.7	Double tilde MDs for a plate (scalefactor VM=20, double tilde MDs=100) . . .	207
10.8	Displacements at the middle node of clamped-clamped beam in QM simulation	208
10.9	Point-wise relative error of the clamped-clamped beam in QM simulation . . .	208
11.1	Snapshot of the cantilever beam simulation ( $r_\phi = 3, r_{\text{aug}} = 9, r = 10$ ) . . . . .	219
11.2	$y$ -displacement of beam's tip for FOM and ROMs ( $r_\phi = 3, r_{\text{aug}} = 9, r = 10$ ) . . .	219
11.3	Point-wise relative error for different reduction methods . . . . .	219
11.4	Snapshot of the cantilever beam simulation ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ ) . . . . .	220
11.5	$y$ -displacement of beam's tip for FOM and ROMs ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ ) . . .	220
11.6	Point-wise relative error for different reduction methods . . . . .	220
11.7	Snapshot of the simulation for the S-shaped structure ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ )	222
11.8	$y$ -displacement of observed node for FOM and ROMs ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ ) .	222
11.9	Point-wise relative error for different reduction methods . . . . .	222

# List of Algorithms

2.1	Construction of the reduced Hessian matrix $\mathbf{H}_r$ . . . . .	12
3.1	Square-Root Balanced Truncation (SR-BT) . . . . .	34
3.2	Iterative rational Krylov algorithm (IRKA) . . . . .	52
5.1	Arnoldi-like algorithm for truncated Volterra series interpolation . . . . .	111
5.2	(Truncated) Bilinear iterative rational Krylov algorithm (BIRKA) . . . . .	113
5.3	Bilinear (input) pseudo-optimal rational Krylov (BPORK-V) . . . . .	114
6.1	Implicit Euler scheme . . . . .	121
6.2	Newton-Raphson method . . . . .	121
7.1	Nonlinear Moment Matching (NLMM) . . . . .	148
9.1	Generalized- $\alpha$ time integration scheme . . . . .	181
9.2	Newmark time integration scheme for manifold-ROM . . . . .	189
11.1	Second-order NLMM (SO-NLMM) . . . . .	217

PART I  
PRELIMINARIES



# Chapter 1

## Introduction

Mathematical models are indispensable for the analysis, design, optimization and control of complex dynamical systems arising in science and engineering. Many real-life applications (e.g. in automotive, aeronautics, biomechanics, circuits, mechatronics, electromagnetism and neuroscience) can be modeled as systems of ordinary differential equations (ODEs), differential-algebraic equations (DAEs) and/or partial differential equations (PDEs). Hereby, the model complexity generally increases with higher demands on the accuracy.

Certain applications (e.g. in control engineering) may lead to rather small models coming from (simplified) lumped-parameter and/or data-driven modeling. Larger models usually originate from technical systems consisting of a high number of individual components/subsystems (e.g. grid/gas networks, integrated circuits) or from the spatial discretization of PDEs describing the underlying physics. Specially in computational mechanics and fluid dynamics a fine discretization over the 2D/3D geometric domain of interest – via e.g. the finite difference (FDM), finite element (FEM) or finite volume (FVM) method – often yields models of thousands or millions of degrees of freedom. The large state dimension ( $n \approx 10^3 \cdots 10^6$ ) – and the associated high storage effort – makes the numerical simulation and computer-aided design computationally demanding. Moreover, it complicates the use of the models for uncertainty quantification, parameter estimation, real-time control or digital twin purposes.

In order to reduce the computational effort and make the numerical analysis more efficient, a reduced-order model (ROM) of lower dimension ( $r \ll n$ ) that accurately approximates the original system with substantially less degrees of freedom is aimed. There are different strategies for reduced-order modeling. The first approach is to construct a lumped-parameter model by simplifying the geometry, neglecting some effects or making certain assumptions. While this technique might be effective in some cases, it usually leads to inaccurate ROMs. A different technique is to employ a coarse mesh, use more sophisticated discretization techniques or exploit certain topological and geometrical properties of the system to reduce the number of discretization points – and consequently of equations. The convergence rate and the achieved model accuracy should be monitored in detail when using this approach. A third option is to employ a fine computational mesh capable of capturing all physical effects of the original system, and then apply model order reduction (MOR) to the full-order model (FOM) to reduce the dimension. In this thesis we focus on this third reduced-order modeling strategy.

The advantages of ROMs (and thus of model order reduction) are obvious. In design optimization for example, where an analysis for different parameters and “what if” scenarios is required, it is essential to speed-up the simulation of the full-order model beforehand. Moreover, the benefit of having multiple cheap online evaluations typically outweighs the upfront offline cost needed for the computation of the reduced model. This is especially the case in real-time applications such as optimization, control and predictive maintenance.

The goals of MOR are threefold. First of all, the aim is to obtain a *good approximation* by accurately capturing the most important dynamics of the full-order model. The approximation quality can be quantified by appropriate error measures. Secondly, important system properties of the original model (e.g. stability, passivity, second-order/Port-Hamiltonian structure, etc.) should be preserved in the reduced model. This requirement is achieved by applying special or adapted reduction techniques tailored to address these demands. Thirdly, the reduction method should be as numerically efficient as possible. Expensive offline, and especially online computations should be avoided. In addition, reduction algorithms should preferably be applicable to large-scale models (i.e. scalable), require less user intervention (i.e. automatable) and provide an upper bound for the error.

Although it is difficult to fulfill all these requirements at once, it is certainly possible to achieve at least some of the goals. For example, for linear time-invariant (LTI) systems a wide variety of model reduction techniques exists (e.g. modal, balanced truncation, moment matching via Krylov subspaces,  $\mathcal{H}_2$ -optimal reduction), all having their purpose, properties, advantages and disadvantages. For *nonlinear* dynamical systems, however, fewer options are available due to the nonlinear nature of the equations and the associated challenges.

## 1.1 Challenges and overview of nonlinear model reduction

Nonlinear effects arise in many technically relevant systems. In circuit and electronic applications, nonlinearities usually originate from nonlinear resistors [64], diodes, transistors, vacuum tubes and magnetic inductors. In fluid dynamics and heat transfer, nonlinear phenomena typically results from convective acceleration (e.g. Navier-Stokes equation), turbulence, nonlinear heat conduction [170] and radiation. In mechanical applications (e.g. in automotive, aeronautics and biomechanics), structures undergoing large deformations show *geometric* nonlinear behavior [224]. Moreover, nonlinear dynamics may also originate from *material* nonlinearities (e.g. plasticity, hyper-elasticity) or nonlinear *boundary conditions* (e.g. contact interactions). Finally, electrostatically actuated microelectromechanical systems (MEMS) can exhibit spring softening and hardening phenomena due to nonlinearities in the restoring force.

Nonlinear dynamical systems pose several challenges for simulation and model reduction. First of all, they can exhibit complex behaviors, such as multiple equilibrium points, several attractors (e.g. stable/unstable/semi-stable limit cycles), bifurcation (e.g. jump phenomenon in nonlinear vibrations [190, 142]), internal resonances and chaotic behavior. Furthermore, general nonlinear systems usually lack of a closed-form solution. Consequently, the input-output behavior cannot be described analytically with transfer functions, the state-transition matrix or convolution integrals. This is only possible for polynomial system classes. Finally, nonlinear MOR comprises not only the reduction of degrees of freedom but also the simplification of the nonlinear terms to gain significant speed-ups (aka. hyper-reduction).

There have been some efforts to extend well-known *system-theoretic* (reduction) concepts to the nonlinear case. Rosenberg [218] generalized the concept of modes to the nonlinear setting, leading to the so-called nonlinear normal modes (NNMs). Based on the center manifold theory [51], Shaw and Pierre [252] later defined them as an invariant manifold in phase-space. Despite their theoretical value, note that NNMs are rather expensive to compute via e.g. shooting techniques and continuation [140]. Next, Scherpen [239] extended the reduction concept of balanced truncation to the nonlinear case exploiting reachability, observability and nonlinear optimal control theory. The approach is (similarly to NNMs) system-theoretically



attractive, but requires the solution of two Hamilton-Jacobi-Bellman PDEs representing the nonlinear counterpart of the linear Lyapunov equations. Finally, Astolfi [13, 14] has transferred the reduction concept of moment matching to the nonlinear setting based on the steady-state response and center manifold theory. The approach requires the solution of a Sylvester-like PDE and is thus not well applicable to large-scale problems.

Due to the lack of analytical solutions and practical system-theoretic measures, nonlinear systems are usually reduced via *simulation-based* approaches. These techniques, e.g. Proper Orthogonal Decomposition (POD) [37, 154], empirical Gramians [165, 164], Trajectory Piecewise Linear (TPWL) [225] and Reduced Basis methods [211, 117], rely on expensive simulations of the FOM for several training input scenarios to construct the reduction basis.

Approaches based on (a single) linearization can only guarantee a good approximation in a small neighborhood around the linearization point. Hence, the idea of enriching a pure linear basis with nonlinear information emerged. A popular way consists in augmenting the basis with so-called modal derivatives [130, 240]. These perturbation derivatives describe the change of the modes w.r.t. the linearization point, hereby capturing some nonlinear behavior. From a computational perspective, this strategy constitutes a *simulation-free* approach since it relies on linearized system matrices rather than on simulated data.

Simulation-free reduction procedures can also be obtained for polynomial nonlinear systems. The Volterra series and variational equation approach [221] allow to represent the solution of a polynomial system as an infinite sum of subresponses described by multivariable convolution integrals and kernels. Moreover, the growing exponential approach enables the derivation of the solution in terms of generalized transfer functions. Based on the kernels and transfer functions, well-known system-theoretic measures (e.g. Gramians,  $\mathcal{H}_2$ -norm) and model reduction approaches (e.g. balanced truncation, moment matching) can then be generalized to this special nonlinear system class [1, 206, 47, 25, 26, 92].

## 1.2 Scope and objectives of the thesis

This thesis focuses on simulation-free (or system-theoretic) model order reduction for polynomial, nonlinear state-space and nonlinear second-order systems.

The first main topic of this thesis involves the system-theoretic model reduction of polynomial systems. Hereby, we will particularly focus on the transfer functions and Krylov-based reduction of (quadratic-)bilinear systems. One of the goals is to extend the subsystem and Volterra series interpolation framework to the multiple-input multiple-output (MIMO) case. Moreover, we aim to develop toolboxes (named (Q)BSSMOR after the sssMOR toolbox [52]) bundling (existing) benchmarks, analysis functions and model reduction algorithms together.

Another central subject concerns the reduction of general nonlinear state-space systems. We will focus on both linear and nonlinear projection-based model reduction and discuss the corresponding existing techniques. Then, we turn our attention to the nonlinear moment matching concept from Astolfi [14] towards the aim of developing a new simulation-free algorithm. Another goal is to establish connections between polynomial and nonlinear MOR.

This thesis also deals with the reduction of nonlinear second-order systems. In this context we focus on modal derivatives as reduction concept. One goal is to apply the Volterra series representation to derive the modal derivatives in a system-theoretical manner. Another aim is to transfer the nonlinear moment matching ideas from the first-order to the second-order case.

### 1.3 Scientific contributions

In the following, the main contributions of this thesis are highlighted:

1. The well-known rational Krylov subspace method (RKSM) [245, 84] is implemented and integrated in the sssMOR toolbox [52]. The function allows both the cumulative reduction of LTI systems and the approximate solution of Lyapunov equations. It further selects the reduction parameters automatically. This is discussed in Section 3.8.
2. In the context of bilinear systems several extensions to the MIMO case are accomplished. Moreover, the Volterra series interpolation is generalized to the multimoment setting and presented from an efficient Arnoldi-like viewpoint. The concept of  $\mathcal{H}_2$ -pseudo-optimality [273] is also extended to the bilinear case. In addition to the theoretical contributions, the BSSMOR toolbox has been developed in parallel. For more details see Chapter 5.
3. For nonlinear state-space systems, a feasible simulation-free model reduction algorithm based on nonlinear moment matching [14] is developed by proposing certain simplifications of the Sylvester-like PDE. Besides computational aspects, practical guidelines are presented to ease the application of the algorithm. Numerical examples demonstrate the efficacy of the method in comparison to POD. This is explained in [72, 71] and Chapter 7.
4. The eigenfunctions of bilinear and quadratic-bilinear systems are derived. Then, we describe how these eigenfunctions could be exploited within the nonlinear moment matching approach from Astolfi. In addition, we try to embed the Volterra series interpolation of bilinear systems in the more general Sylvester-like PDE framework. These connections between polynomial and nonlinear model reduction are elucidated in Chapter 8.
5. A novel derivation of modal derivatives based on the Volterra series representation for nonlinear second-order systems is presented. The gained derivatives are symmetric and allow for a system-theoretic explanation of internal resonances. Moreover, their applications to nonlinear analysis and model reduction are discussed. Preliminary numerical results underline the potential of the new derivatives. This is presented in [73] and Chapter 10.
6. The concept of nonlinear moment matching [14] is transferred to nonlinear second-order systems. After providing the corresponding Sylvester-like PDE, similar simplifications as for state-space systems are proposed to achieve a simulation-free second-order nonlinear moment matching algorithm. The algorithm is implemented and validated within the finite element package AMfe [224] using Python. This is discussed in [75] and Chapter 11.

In addition to these quantitative contributions, Chapters 3, 4, 6 and 9 contain important fundamentals of linear, polynomial, nonlinear state-space and nonlinear second-order systems. These chapters complement the existing literature with valuable explanations and discussions, present partly novel insights (e.g. into nonlinear manifold reduction), and offer a unifying view supported by the consistent notation. Thus, these chapters do not only lay the foundations for the main contributions, but also generate an added value. The goal of the author during her doctoral studies was not only the mere extension of existing MOR methods to other system classes or the development of new simulation-free approaches. The aim was also to understand the conceptual similarities and differences between the various methods, establish connections between them and uncover new perspectives for future research. Focusing on polynomial, state-space and second-order systems helped the author to get inspired by the different communities and acquire a broad MOR knowledge that is presented in this thesis.

Other contributions that are only briefly mentioned in this thesis are:

- The derivation of an expression for the impulse response of bilinear systems [60]. Different from the kernels derived from the Volterra series representation, the impulse response is dependent on the amplitude of the input. Krylov subspaces and an alternative expression for the  $\mathcal{H}_2$ -norm based on the derived impulse response are also given in [Geb17].
- The extension of subsystem interpolation for quadratic-bilinear systems [22] to the MIMO case. Based on the corresponding transfer matrices, we propose different (tangential) Krylov subspaces to achieve both multimoment and Hermite interpolation of the first two subsystems. The algorithms are implemented in our QBSSMOR toolbox. For details, see the talks [Cru16b, CFL17, Cru17b] and the master thesis [Fio16].
- The theoretical development of basis vector (eigenvector, Krylov) derivatives representing the *state-space* counterpart of the perturbation (modal, Krylov) derivatives usually employed in structural dynamics. For details and numerical examples, see [Him18].
- The procedural implementation of the benchmark models “nonlinear heat transfer” [170] and “electrostatic beam” [166]. In contrast to the original versions developed by Lieneemann<sup>1</sup>, our implementation<sup>2</sup> allows to customize the order and the parameters of the model. For more details the reader is referred to the semester thesis [Lep17].

The author has also collaborated with different colleagues and researchers concerning linear and parametric (nonlinear) model order reduction:

- In [203], we apply POD and subspace interpolation (cf. [36]) to reduce a large human heart model. The ROM is then exploited to speed-up the gradient-based inverse analysis.
- In [174], we apply modal- and Krylov-based MOR to a linear thermoacoustic problem. The ROM is evaluated via the approximation quality of intrinsic eigenmodes.
- In [68], we develop an adaptive sampling algorithm based on the concept of subspace angles. This and further parametric MOR methods are integrated in our PSSSMOR toolbox<sup>3</sup>.

## 1.4 Outline of the thesis

This thesis is composed of five parts.

Part I deals with the preliminaries and covers Chapters 1, 2 and 3. After the introduction, Chapter 2 gives an overview of mathematical fundamentals required for the upcoming chapters of this thesis. It includes a review of basic concepts of matrix and tensor algebra, the principles of projection and an introduction to different types of Krylov subspaces. Moreover, the numerical solution of linear matrix equations and linear systems of equations is discussed. Finally, the focus is laid on error measures to quantify the reduction error in time-domain. Chapter 3 is devoted to model order reduction of LTI systems. After the presentation of essential system-theoretic concepts and an overview of linear reduction approaches, we concentrate on the method of moment matching by rational Krylov subspaces. We first review the classical frequency-domain interpretation of this technique and discuss the equivalence between Krylov subspaces and Sylvester equations. Then, we revisit (and partly enrich) the time-domain interpretation of moments as the steady-state response of the system intercon-

<sup>1</sup>Available e.g. at the MOR Wiki under <https://morwiki.mpi-magdeburg.mpg.de>.

<sup>2</sup>Available at <https://doi.org/10.5281/zenodo.3542641>.

<sup>3</sup>Available at <https://www.mw.tum.de/rt/forschung/modellordnungsreduktion/software/psssmor/>.

nected with a signal generator. Lastly, the  $\mathcal{H}_2$ -(pseudo-)optimal reduction problem is briefly explained, and our implementation of CRKSM for the cumulative reduction of LTI systems and the approximate solution of linear Lyapunov equations is described.

Part II deals with polynomial nonlinear state-space systems and covers Chapters 4 and 5. In Chapter 4 we first revisit different ways of obtaining a polynomial system and the transformation to bilinear and quadratic-bilinear form. Afterwards, the variational equation and growing exponential approach are presented to derive the different Volterra kernels and generalized transfer functions of the above mentioned system classes. Chapter 5 then concentrates on the systems theory and MOR of bilinear dynamical systems. Based on the regular kernels and transfer functions, system-theoretic concepts such as the pole-residue formulation, Gramians and  $\mathcal{H}_2$ -norm are discussed. After that, we turn our attention to Krylov-based model reduction. We propose a different strategy for MIMO subsystem interpolation, generalize the Volterra series interpolation to the MIMO and multimoment matching case, and present a new algorithm to obtain bilinear  $\mathcal{H}_2$ -pseudo-optimal ROMs.

Part III deals with nonlinear state-space systems and covers Chapters 6, 7 and 8. Since nonlinear systems can be reduced using either a linear or nonlinear projection, Chapter 6 describes the corresponding framework, time integration of the ROM and reduction techniques of both strategies. It also gives an overview of different hyper-reduction procedures, and discusses the advantages and disadvantages of linear subspaces versus nonlinear manifolds. Chapter 7 first reviews Astolfi's extension of moment matching to nonlinear systems based on the steady-state interpretation of moments. Then, we propose some simplifications to approximate the arising PDE, hereby achieving a feasible algorithm that relies on the solution of nonlinear systems of equations. An extensive discussion, practical guidelines and numerical results of the proposed algorithm are also presented. Chapter 8 establishes connections between polynomial and nonlinear model reduction. We derive the eigenfunctions of bilinear and quadratic-bilinear systems, explain their construction via a signal generator and their use in the nonlinear moment matching framework from Astolfi. Moreover, we provide the signal generator interpretation of the Volterra series interpolation of bilinear systems.

Part IV deals with nonlinear mechanical systems and covers Chapters 9, 10 and 11. Chapter 9 lays the fundamentals of model reduction for nonlinear second-order systems and focuses again on both linear and nonlinear projection. We also revisit the original derivation of modal derivatives and explain their use for basis augmentation and quadratic manifold reduction. In Chapter 10 a novel derivation of modal derivatives using the Volterra series representation is presented. In addition to their properties, we discuss the gained analytical solution, novel quadratic manifold approaches for model reduction, and the connection to the Harmonic Balance method using the new derivatives. Preliminary numerical results are also reported. Chapter 11 transfers the concept of nonlinear moment matching to second-order systems. We first present the time-domain interpretation of moments and provide the corresponding second-order signal generator and Sylvester-like PDE. Then, a similar algorithm as for state-space systems is obtained by applying our proposed simplifications.

Part V completes the thesis with conclusions of the results and an outlook for future research in Chapter 12.

# Chapter 2

## Mathematical Fundamentals

In this chapter we collect important mathematical foundations that are essential for the rest of this thesis. Section 2.1 presents well-known concepts of matrix and tensor theory. In Section 2.2 the fundamentals and properties of projections are revised. These are crucial to later understand the projective framework of model order reduction. Then, in Section 2.3 we deal with different types of Krylov subspaces that were introduced in the literature over the years. Krylov subspaces were originally developed to approximately solve linear matrix and systems of equations. Both direct and iterative methods to solve such kind of equations will be discussed in Sections 2.4 and 2.5. Finally, in Section 2.6 we focus on error measures which will help us to quantify the reduction error in time-domain.

### 2.1 Tensor algebra

In this section we revisit some basic concepts from linear and tensor algebra, which can be found e.g. in [114, 122, 139]. These concepts will play an important role when handling with matrix equations and the Taylor series expansion of nonlinear systems.

#### Kronecker product

We begin by defining the Kronecker product of two matrices.

**Definition 2.1** (Kronecker product). Let  $\mathbf{A} \in \mathbb{R}^{m \times n}$  and  $\mathbf{B} \in \mathbb{R}^{p \times q}$  be arbitrary matrices. Then, the Kronecker product  $\mathbf{A} \otimes \mathbf{B}$  is determined by

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \dots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \dots & a_{mn}\mathbf{B} \end{bmatrix} \in \mathbb{R}^{mp \times nq}. \quad (2.1)$$

The Kronecker product results in a matrix of dimension  $mp \times nq$ . ▲

Throughout the rest of this thesis we will use the following notation for the  $k$ -fold Kronecker product:

$$\mathbf{x}^{(k)} = \underbrace{\mathbf{x} \otimes \dots \otimes \mathbf{x}}_{k\text{-times}} \in \mathbb{R}^{n^k} \quad \text{and} \quad \mathbf{I}_{m^k} = \underbrace{\mathbf{I}_m \otimes \dots \otimes \mathbf{I}_m}_{k\text{-times}} \in \mathbb{R}^{m^k \times m^k}. \quad (2.2)$$

Next, we state some important properties and calculation rules of the Kronecker product.

**Lemma 2.1.** Let  $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$  be matrices of appropriate dimensions. Let  $\mathbf{u}, \mathbf{v}$  be arbitrary vectors. Let  $\mathbf{I}_m, \mathbf{I}_n$  be identity matrices,  $\mathbf{O}$  a zero matrix,  $\mathbf{1}_m$  a vector of ones and  $\alpha$  a scalar. Then the following properties of the Kronecker product hold, assuming conformable dimensions with respect to matrix addition and multiplication.

1. Non-commutative:

$$\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A}, \quad \mathbf{u} \otimes \mathbf{v} \neq \mathbf{v} \otimes \mathbf{u}, \quad (2.3)$$

$$\mathbf{I}_m \otimes \mathbf{A} \neq \mathbf{A} \otimes \mathbf{I}_m, \quad \mathbf{1}_m \otimes \mathbf{A} \neq \mathbf{A} \otimes \mathbf{1}_m, \quad (2.4)$$

$$\mathbf{I}_m \otimes \mathbf{I}_n = \mathbf{I}_n \otimes \mathbf{I}_m = \mathbf{I}_{mn}, \quad \mathbf{A} \otimes \mathbf{O} = \mathbf{O} \otimes \mathbf{A} = \mathbf{O}. \quad (2.5)$$

2. Associative:

$$\mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) = \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C}, \quad (2.6)$$

$$(\mathbf{B} + \mathbf{C}) \otimes \mathbf{A} = \mathbf{B} \otimes \mathbf{A} + \mathbf{C} \otimes \mathbf{A}, \quad (2.7)$$

$$\alpha (\mathbf{A} \otimes \mathbf{B}) = (\alpha \mathbf{A}) \otimes \mathbf{B} = \mathbf{A} \otimes (\alpha \mathbf{B}), \quad (2.8)$$

$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}), \quad (2.9)$$

where the matrix sum  $\mathbf{B} + \mathbf{C}$  can be formed.

3. Mixed-product property:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}), \quad (2.10)$$

where the matrix products  $\mathbf{AC}$  and  $\mathbf{BD}$  can be formed.

## Vectorization

Another important operation is the vectorization of a matrix, which is defined as follows.

**Definition 2.2** (Vectorization). Let  $\mathbf{A} \in \mathbb{R}^{m \times n}$  be an arbitrary matrix. Then, the vectorization  $\text{vec}(\mathbf{A})$  is obtained by stacking the columns of  $\mathbf{A}$  underneath each other, i.e.

$$\text{vec}(\mathbf{A}) = \text{vec} \left( \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \right) = \text{vec} \left( [\mathbf{a}_1, \cdots, \mathbf{a}_n] \right) = \begin{bmatrix} \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} \in \mathbb{R}^{mn \times 1}. \quad (2.11)$$

The vectorization of the matrix results in a  $mn \times 1$  column vector. ▲

The vectorization is frequently used together with the Kronecker product. Some useful properties of the vectorization operator are stated in the following.

**Lemma 2.2.** Let  $\mathbf{A} \in \mathbb{R}^{m \times n}$ ,  $\mathbf{X} \in \mathbb{R}^{n \times r}$  and  $\mathbf{E} \in \mathbb{R}^{r \times s}$  be arbitrary matrices. Then it holds:

$$\text{tr}(\mathbf{AX}) = \text{vec}(\mathbf{A}^\top)^\top \text{vec}(\mathbf{X}), \quad (2.12)$$

$$\text{vec}(\mathbf{AXE}) = (\mathbf{E}^\top \otimes \mathbf{A}) \text{vec}(\mathbf{X}). \quad (2.13)$$

The second relationship (2.13) is particularly useful to express some matrix equations (e.g. Lyapunov and Sylvester equations) as linear system of equations (cf. Eq. (2.43)).

## Tensors

Tensors, which may be conceived as multidimensional arrays, represent the natural extension of the concept of a matrix. For example, vectors and matrices can be interpreted as first- and second-order tensors, respectively. In the following we define a tensor of general order  $d$ .

**Definition 2.3** (Tensor). A  $d$ -th order tensor  $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$  is a  $d$ -dimensional array of entries  $\mathcal{X}_{i_1, \dots, i_d} \in \mathbb{R}$ , where  $i_\mu \in \{1, \dots, n_\mu\}$  for  $\mu = 1, \dots, d$ .  $\blacktriangle$

In what follows, third-order tensors ( $d = 3$ ) will play a particular role in this thesis. We define *fibers* and *slices* of a tensor in the following (cf. [139]).

The  $\mu$ -mode *fibers* of a tensor are defined by fixing all indices except of one. For example, the columns  $\mathcal{X}(:, j, k)$  represent the 1-mode fibers, the rows  $\mathcal{X}(i, :, k)$  represent the 2-mode fibers, and the tubes  $\mathcal{X}(i, j, :)$  represent the 3-mode fibers of a third-order tensor.

*Slices* represent two-dimensional sections of a tensor, defined by fixing all indices except of two. For instance, a third-order tensor possesses horizontal  $\mathcal{X}(i, :, :)$ , lateral  $\mathcal{X}(:, j, :)$  and frontal  $\mathcal{X}(:, :, k)$  slices, respectively. The latter are usually denoted as  $\mathbf{X}_k$ .

Next, we focus on the *unfolding* of a tensor into a matrix. This process is also known as *matricization* and depends on the chosen  $\mu$ -mode fibers used for the unfolding.

**Definition 2.4** ( $\mu$ -matricization of a tensor). Let  $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$  be a third-order tensor. Its  $\mu$ -matricization  $\mathcal{X}^{(\mu)} \in \mathbb{R}^{n_\mu \times \frac{(n_1 n_2 n_3)}{n_\mu}}$  is defined as the arrangement of the tensor's  $\mu$ -mode fibers next to each other as columns of the resulting matrix. For the case of a *cubical* third-order tensor  $\mathcal{X} \in \mathbb{R}^{n \times n \times n}$  with frontal slices  $\mathbf{X}_n \in \mathbb{R}^{n \times n}$ , the  $\mu$ -matricizations are

$$\mathcal{X}^{(1)} = [\mathbf{X}_1, \dots, \mathbf{X}_n], \quad \mathcal{X}^{(2)} = [\mathbf{X}_1^\top, \dots, \mathbf{X}_n^\top], \quad \mathcal{X}^{(3)} = [\text{vec}(\mathbf{X}_1), \dots, \text{vec}(\mathbf{X}_n)]^\top,$$

with respective dimensions  $\mathcal{X}^{(\mu)} \in \mathbb{R}^{n \times n^2}$ ,  $\mu \in \{1, 2, 3\}$ .  $\blacktriangle$

To make this definition more clear, we provide an illustrative example in the following.

*Example 2.1* (Matricizations of a third-order tensor). Let  $\mathcal{X} \in \mathbb{R}^{3 \times 3 \times 2}$  be given with the two frontal slices:

$$\mathcal{X}(:, :, 1) = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad \mathcal{X}(:, :, 2) = \begin{bmatrix} 10 & 11 & 12 \\ 13 & 14 & 15 \\ 16 & 17 & 18 \end{bmatrix}.$$

- 1-matricization:

$$\mathcal{X}^{(1)} = \begin{bmatrix} 1 & 2 & 3 & 10 & 11 & 12 \\ 4 & 5 & 6 & 13 & 14 & 15 \\ 7 & 8 & 9 & 16 & 17 & 18 \end{bmatrix}$$

- 2-matricization:

$$\mathcal{X}^{(2)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 \\ 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \end{bmatrix}$$

- 3-matricization:

$$\mathcal{X}^{(3)} = \begin{bmatrix} 1 & 4 & 7 & 2 & 5 & 8 & 3 & 6 & 9 \\ 10 & 13 & 16 & 11 & 14 & 17 & 12 & 15 & 18 \end{bmatrix}$$

The respective dimensions are  $\mathcal{X}^{(1)} \in \mathbb{R}^{3 \times 3 \times 2}$ ,  $\mathcal{X}^{(2)} \in \mathbb{R}^{3 \times 3 \times 2}$  and  $\mathcal{X}^{(3)} \in \mathbb{R}^{2 \times 3 \times 3}$ .  $\triangle$

Similar to matrix-matrix multiplications, one can also perform tensor-tensor and tensor-matrix multiplications. Particularly important for this thesis are tensor-matrix multiplications, which can be easily performed by means of matricizations. [139]

**Definition 2.5** ( $\mu$ -mode matrix product). Let  $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$  be a third-order tensor and  $\mathbf{A} \in \mathbb{R}^{r \times n_\mu}$  a matrix. The  $\mu$ -mode matrix product of the tensor  $\mathcal{X}$  with  $\mathbf{A}$  is defined as

$$\mathcal{Y} = \mathcal{X} \times_\mu \mathbf{A} \iff \mathcal{Y}^{(\mu)} = \mathbf{A} \mathcal{X}^{(\mu)}. \quad (2.14)$$

I.e., after the calculation of  $\mathcal{Y}^{(\mu)} = \mathbf{A} \mathcal{X}^{(\mu)}$ , the corresponding tensor  $\mathcal{Y}$  can be computed.  $\blacktriangle$

**Corollary 2.1** (Matricizations of a tensor defined by matrix products). Let  $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$  be a third-order tensor and  $\mathbf{A} \in \mathbb{R}^{r_1 \times n_1}$ ,  $\mathbf{B} \in \mathbb{R}^{r_2 \times n_2}$  and  $\mathbf{C} \in \mathbb{R}^{r_3 \times n_3}$  be matrices. If a tensor is given by the matrix product  $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ , then its  $\mu$ -mode matricizations are given by

$$\mathcal{Y}^{(1)} = \mathbf{A} \mathcal{X}^{(1)} (\mathbf{C}^\top \otimes \mathbf{B}^\top), \quad \mathcal{Y}^{(2)} = \mathbf{B} \mathcal{X}^{(2)} (\mathbf{C}^\top \otimes \mathbf{A}^\top), \quad \mathcal{Y}^{(3)} = \mathbf{C} \mathcal{X}^{(3)} (\mathbf{B}^\top \otimes \mathbf{A}^\top). \quad (2.15)$$

I.e., the tensor  $\mathcal{Y}$  can be computed over matrix-matrix products.

In this thesis, tensors arise in the context of Taylor series expansion of a nonlinear system (cf. Section 4.1.1 and Appendix A) or in quadratic-bilinear systems. For example, the Hessian matrix  $\mathbf{A}_2 \hat{=} \mathbf{H} \in \mathbb{R}^{n \times n^2}$  can be interpreted as the 1-mode matricization  $\mathcal{H}^{(1)}$  of the cubical third-order tensor  $\mathcal{H} \in \mathbb{R}^{n \times n \times n}$ . The just reported tensor properties are then exploited to efficiently compute the product  $\mathbf{H}(\mathbf{V} \otimes \mathbf{V})$  or the reduced matrix  $\mathbf{H}_r = \mathbf{W}^\top \mathbf{H}(\mathbf{V} \otimes \mathbf{V})$  without explicitly forming the  $n^2 \times r^2$  matrix  $\mathbf{V} \otimes \mathbf{V}$ . The process is described in the following [22].

---

**Algorithm 2.1** Construction of the reduced Hessian matrix  $\mathbf{H}_r$

---

**Input:** Full-order Hessian matrix  $\mathbf{H} \hat{=} \mathcal{H}^{(1)} \in \mathbb{R}^{n \times n^2}$ , reduction matrices  $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r}$

**Output:** Reduced Hessian matrix  $\mathbf{H}_r \in \mathbb{R}^{r \times r^2}$

- 1: Compute  $\mathcal{Y} \in \mathbb{R}^{r \times n \times n}$  via  $\mathcal{Y}^{(1)} = \mathbf{W}^\top \mathcal{H}^{(1)}$
  - 2: Compute  $\mathcal{Z} \in \mathbb{R}^{r \times r \times n}$  via  $\mathcal{Z}^{(2)} = \mathbf{V}^\top \mathcal{Y}^{(2)}$
  - 3: Compute  $\mathcal{H}_r \in \mathbb{R}^{r \times r \times r}$  via  $\mathcal{H}_r^{(3)} = \mathbf{V}^\top \mathcal{Z}^{(3)}$
  - 4: The reduced Hessian is given by  $\mathbf{H}_r = \mathcal{H}_r^{(1)}$
- 

The calculation of the reduced Hessian becomes computationally feasible by means of this algorithm, since the formation of the large *dense* matrix  $\mathbf{V} \otimes \mathbf{V}$  is completely avoided. Moreover, the matrix  $\mathcal{Y}^{(1)} = \mathbf{W}^\top \mathcal{H}^{(1)} \in \mathbb{R}^{r \times n^2}$  is generally *sparse*, making its storage possible even for large system dimensions  $n$ . Algorithm 2.1 is implemented in our QBSSMOR toolbox (`computeHr`). Note, however, that another efficient way of computing  $\mathbf{H}_r$  is proposed in [35] and [103, Alg. 4.4] by exploiting the Kronecker product structure of the Hessian  $\mathbf{H}$ .



Next, we turn our attention to *symmetry* properties of tensors. In general, a tensor is (partially) symmetric in two or more modes, if its elements remain invariant under permutation of the corresponding indices. [139]

**Definition 2.6** (Symmetric tensors). Let  $\mathcal{H} \in \mathbb{R}^{n \times n \times n}$  be a cubical third-order tensor. Then, the tensor is symmetric in

- a) modes one and two, if  $\mathcal{H}_{ijk} = \mathcal{H}_{jik}$ . This means:  $\mathcal{H}^{(1)} = \mathcal{H}^{(2)}$  or  $\mathbf{H}_k = \mathbf{H}_k^\top$ ,  $k = 1, \dots, n$ .
- b) modes two and three, if  $\mathcal{H}_{ijk} = \mathcal{H}_{ikj}$ . This means:  $\mathcal{H}^{(2)} = \mathcal{H}^{(3)}$ .
- c) modes one and three, if  $\mathcal{H}_{ijk} = \mathcal{H}_{kji}$ . This means:  $\mathcal{H}^{(1)} = \mathcal{H}^{(3)}$ .
- d) all modes (aka. *supersymmetric*), if  $\mathcal{H}_{ijk} = \mathcal{H}_{ikj} = \mathcal{H}_{jik} = \mathcal{H}_{jki} = \mathcal{H}_{kij} = \mathcal{H}_{kji}$ .

▲

In case b) or d), a special commutativity holds for arbitrary vectors  $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ :

$$\mathcal{H}^{(1)}(\mathbf{u} \otimes \mathbf{v}) = \mathcal{H}^{(1)}(\mathbf{v} \otimes \mathbf{u}). \quad (2.16)$$

This property is important, since it will allow us to simplify expressions arising in quadratic-bilinear transfer functions (see Section 4.4.2), as well as in the derivation of novel modal derivatives for nonlinear mechanical systems (cf. Eq. (10.15)).

In this thesis we will assume that the tensor  $\mathcal{H}$  corresponding to the Hessian  $\mathbf{H}$  arising e.g. in quadratic-bilinear systems fulfills the property (2.16), i.e.  $\mathcal{H}^{(2)} = \mathcal{H}^{(3)}$ . However, the Hessian obtained from semidiscretization, Taylor series expansion or quadratic-bilinearization of the underlying system might not always be symmetric. In such case, we can *symmetrize* the tensor  $\mathcal{H}$  w.r.t. modes 2 and 3 by forming the symmetric part as follows:

$$\mathcal{H}_{\text{sym}}^{(2)} := \frac{1}{2}(\mathcal{H}^{(2)} + \mathcal{H}^{(3)}) =: \mathcal{H}_{\text{sym}}^{(3)}. \quad (2.17)$$

Then, it is possible to substitute the original (non-symmetric) Hessian by the 1-mode matricization of the symmetrized tensor, i.e.  $\mathbf{H} \hat{=} \mathcal{H}^{(1)} \rightarrow \mathbf{H}_{\text{sym}} \hat{=} \mathcal{H}_{\text{sym}}^{(1)}$ . This symmetrization process does not change the dynamics of the original quadratic(-bilinear) model. This is illustrated in the following example (leaned on [22], [Fio16]):

*Example 2.2* (Symmetrization of a quadratic system). Let us consider the system

$$\underbrace{\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} a & b & c & d \\ e & f & g & h \end{bmatrix}}_{\mathcal{H}^{(1)} := \mathbf{H}} \underbrace{\begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}}_{\mathbf{x} \otimes \mathbf{x}}, \quad \mathcal{H}(:, :, 1) = \begin{bmatrix} a & b \\ e & f \end{bmatrix}, \quad \mathcal{H}(:, :, 2) = \begin{bmatrix} c & d \\ g & h \end{bmatrix}. \quad (2.18)$$

Since the second and third entries of the vector  $\mathbf{x} \otimes \mathbf{x}$  are equal, the dynamics of the system are not changed as long as the sums  $h_{12} + h_{13} = b + c$  and  $h_{22} + h_{23} = f + g$  hold. The 2- and 3-mode matricizations of  $\mathcal{H}$  can be written as

$$\mathcal{H}^{(2)} = \begin{bmatrix} a & e & c & g \\ b & f & d & h \end{bmatrix}, \quad \mathcal{H}^{(3)} = \begin{bmatrix} a & e & b & f \\ c & g & d & h \end{bmatrix}. \quad (2.19)$$

The symmetrized tensor  $\mathcal{H}_{\text{sym}}$  can then be computed via (2.17):

$$\mathcal{H}_{\text{sym}}^{(2)} = \mathcal{H}_{\text{sym}}^{(3)} = \begin{bmatrix} a & e & \frac{b+c}{2} & \frac{f+g}{2} \\ \frac{b+c}{2} & \frac{f+g}{2} & d & h \end{bmatrix}. \quad (2.20)$$

The 1-mode matricization of the symmetrized tensor finally yields:

$$\mathbf{H}_{\text{sym}} = \mathcal{H}_{\text{sym}}^{(1)} = \begin{bmatrix} a & \frac{b+c}{2} & \frac{b+c}{2} & d \\ e & \frac{f+g}{2} & \frac{f+g}{2} & h \end{bmatrix}. \quad (2.21)$$

It can be observed that the sum of the elements in the second and third columns of the symmetrized Hessian are equal to the original, which means that the dynamics of both systems are equivalent.  $\triangle$

## 2.2 Projection

Model reduction for dynamical systems is generally performed via *projection*. In other words, the reduced-order model is obtained by projecting the full-order model onto a lower dimensional subspace. Thus, in this section we revisit the most important fundamentals and properties of projections, see e.g. [176, 232, 122].

**Definition 2.7** (Projector). A matrix  $\mathbf{\Pi} \in \mathbb{R}^{n \times n}$  is called a *projector*, if  $\mathbf{\Pi}^2 = \mathbf{\Pi}$ .  $\blacktriangle$

**Lemma 2.3.** Let  $\mathbf{\Pi} \in \mathbb{R}^{n \times n}$  be a projector. Then the following assertions hold:

- If  $\text{ran}(\mathbf{\Pi}) = \mathcal{U}$ , then  $\mathbf{\Pi}$  is said to be the projector onto the subspace  $\text{ran}(\mathbf{U}) = \mathcal{U}$ .
- If  $\mathbf{\Pi}$  is a projector onto  $\mathcal{U}$ , then  $\mathbf{\Pi}$  is the identity operator on  $\mathcal{U}$ , i.e.:  $\mathbf{\Pi}\boldsymbol{\xi} = \boldsymbol{\xi}, \forall \boldsymbol{\xi} \in \mathcal{U}$ .
- The matrix  $\mathbf{\Pi}_{\perp} = \mathbf{I} - \mathbf{\Pi}$  is also a projection.  $\mathbf{\Pi}_{\perp}$  is called *complementary projector*.
- $\ker(\mathbf{\Pi}) = \text{ran}(\mathbf{I} - \mathbf{\Pi}) = \mathcal{W}^{\perp}$  is called the *orthogonal complement* of  $\text{ran}(\mathbf{W}) = \mathcal{W}$ .

Next, we briefly derive the projector  $\mathbf{\Pi}$  that is required to achieve the following projection. The vector  $\boldsymbol{\xi} \in \mathbb{R}^n$  shall be projected onto the  $r$ -dimensional subspace  $\mathcal{U}$  spanned by the columns of the matrix  $\mathbf{U} \in \mathbb{R}^{n \times r}$ . The projection shall be performed along the vector  $\boldsymbol{\varepsilon}$ , which is orthogonal to the subspace  $\mathcal{W}$  spanned by the matrix  $\mathbf{W} \in \mathbb{R}^{n \times r}$  (cf. Fig. 2.1).

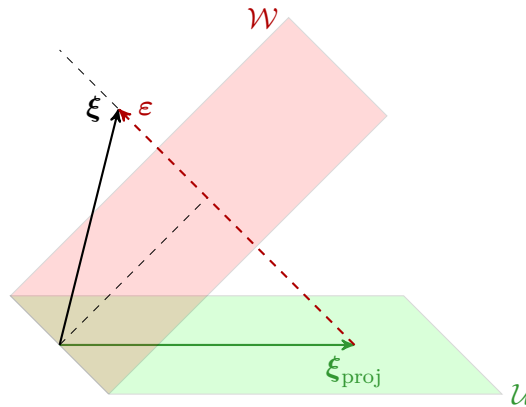


Figure 2.1: Projection of  $\boldsymbol{\xi}$  onto  $\boldsymbol{\xi}_{\text{proj}} \in \mathcal{U}$  along  $\boldsymbol{\varepsilon} \in \mathcal{W}^{\perp}$ .

The projected vector  $\xi_{\text{proj}} \in \mathcal{U} \subset \mathbb{R}^n$  can be given by a linear combination of the basis vectors  $\mathbf{u}_1, \dots, \mathbf{u}_r \in \mathbb{R}^n$  and the coefficients  $c_1, \dots, c_r$  as follows:

$$\xi_{\text{proj}} = \sum_{i=1}^r \mathbf{u}_i c_i = \underbrace{\begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_r \end{bmatrix}}_{\mathbf{U}} \underbrace{\begin{bmatrix} c_1 \\ \vdots \\ c_r \end{bmatrix}}_{\mathbf{c}} = \mathbf{U}\mathbf{c}. \quad (2.22)$$

Here, the coefficient vector  $\mathbf{c} \in \mathbb{R}^r$  is still unknown. Furthermore, the following vector chain is closed (cf. Fig. 2.1):

$$\xi = \xi_{\text{proj}} + \varepsilon \quad \Leftrightarrow \quad \varepsilon = \xi - \xi_{\text{proj}}. \quad (2.23)$$

The columns of  $\mathbf{W}$  are orthogonal to the projection direction  $\varepsilon \in \mathbb{R}^n$ , meaning that

$$\mathbf{W}^T \varepsilon \stackrel{!}{=} \mathbf{0}. \quad (2.24)$$

Inserting Eqs. (2.23) and (2.22) in (2.24) yields

$$\mathbf{W}^T \varepsilon = \mathbf{W}^T \xi - \mathbf{W}^T \xi_{\text{proj}} = \mathbf{W}^T \xi - \mathbf{W}^T \mathbf{U}\mathbf{c} \stackrel{!}{=} \mathbf{0}, \quad (2.25)$$

which can be solved for the coefficient vector  $\mathbf{c}$ , provided that  $\det(\mathbf{W}^T \mathbf{U}) \neq 0$ :

$$\mathbf{c} = (\mathbf{W}^T \mathbf{U})^{-1} \mathbf{W}^T \xi. \quad (2.26)$$

By substituting (2.26) in (2.22), we finally obtain the relation

$$\xi_{\text{proj}} = \mathbf{U}\mathbf{c} = \underbrace{\mathbf{U}(\mathbf{W}^T \mathbf{U})^{-1} \mathbf{W}^T}_{\mathbf{\Pi}} \xi = \mathbf{\Pi}\xi. \quad (2.27)$$

The derived expression for  $\mathbf{\Pi}$  fulfills Definition 2.7, confirming that  $\mathbf{\Pi} = \mathbf{U}(\mathbf{W}^T \mathbf{U})^{-1} \mathbf{W}^T$  is indeed a projector. Furthermore,  $\text{ran}(\mathbf{\Pi}) = \text{ran}(\mathbf{U})$  denotes the subspace onto it is projected, whereas  $\ker(\mathbf{\Pi}) = \text{ran}(\mathbf{W})^\perp$  defines the direction of the projection.

**Definition 2.8** (Orthogonal and oblique projection). A projection  $\mathbf{\Pi}$  is called *orthogonal*, if  $\mathbf{\Pi} = \mathbf{\Pi}^T$ . Otherwise, it is called an *oblique* projection.  $\blacktriangle$

Clearly, the projector  $\mathbf{\Pi} = \mathbf{U}(\mathbf{W}^T \mathbf{U})^{-1} \mathbf{W}^T$  is oblique, since  $\mathbf{\Pi} \neq \mathbf{\Pi}^T$ . The special choice  $\mathbf{W} = \mathbf{U} = \mathbf{V}$  yields the orthogonal projector  $\mathbf{\Pi} = \mathbf{V}(\mathbf{V}^T \mathbf{V})^{-1} \mathbf{V}^T = \mathbf{\Pi}^T$ .

## 2.3 Krylov subspaces

In this thesis, we will mainly employ projections onto so-called *Krylov subspaces*. The concept is named after the Russian mathematician A. N. Krylov, who introduced it in 1931 for the computation of the characteristic polynomial of a matrix, i.e. for eigenvalue problems. In the following, we revisit different types of Krylov subspaces.

### Classical Krylov subspace

**Definition 2.9** (Krylov subspace). Let the matrix  $\mathbf{A} \in \mathbb{R}^{n \times n}$  and the vector  $\mathbf{b} \in \mathbb{R}^n$  be given. The  $q$ -order Krylov subspace  $\mathcal{K}_q(\mathbf{A}, \mathbf{b})$  is defined by the following sequence of vectors:

$$\mathcal{K}_q(\mathbf{A}, \mathbf{b}) = \text{span} \left\{ \mathbf{b}, \mathbf{A}\mathbf{b}, \mathbf{A}^2\mathbf{b}, \dots, \mathbf{A}^{q-1}\mathbf{b} \right\}. \quad (2.28)$$

Are the  $q$  vectors linearly independent, then they form a basis  $\mathcal{V} = \text{ran}(\mathbf{V})$  of the Krylov subspace, i.e.  $\text{ran}(\mathbf{V}) \subseteq \mathcal{K}_q(\mathbf{A}, \mathbf{b})$ . The basis matrix is then  $\mathbf{V} \in \mathbb{R}^{n \times q}$ .  $\blacktriangle$

Due to the powers of  $\mathbf{A}$ , the vectors are usually computed iteratively from the previous ones by  $\mathbf{v}_0 = \mathbf{b}$ ,  $\mathbf{v}_\ell = \mathbf{A}\mathbf{v}_{\ell-1}$ ,  $\ell = 1, 2, \dots, q-1$ , with a subsequent orthonormalization at every iteration via the *modified* Gram-Schmidt process. Such a numerically stable implementation is known under the name of *Lanczos* [155] or *Arnoldi iteration* [8], where the former is a special (and cheaper) case of the Arnoldi iteration for Hermitian matrices. Both algorithms (and numerically improved variants thereof) were applied thereafter to iteratively solve large eigenvalue problems [196, 234, 233] and linear systems of equations [123, 208, 209, 255].

### Rational Krylov subspace

Inspired by the inverse iteration, Ruhe [222] then proposed to apply the Lanczos and Arnoldi iteration to the *shifted and inverted* matrix  $(\sigma\mathbf{E} - \mathbf{A})^{-1}$ . This yields the so-called *rational* Krylov subspace, which is defined here for the case of a matrix  $\mathbf{B} \in \mathbb{R}^{n \times m}$ .

**Definition 2.10** (Block rational Krylov subspace). Let  $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$  and  $\sigma \in \mathbb{C}$  be given. The  $q$ -order *block rational* Krylov subspace  $\mathcal{K}_q(\mathbf{A}_\sigma^{-1}\mathbf{E}, \mathbf{A}_\sigma^{-1}\mathbf{B})$  is defined as

$$\mathcal{K}_q(\mathbf{A}_\sigma^{-1}\mathbf{E}, \mathbf{A}_\sigma^{-1}\mathbf{B}) = \text{span} \left\{ \mathbf{A}_\sigma^{-1}\mathbf{B}, \mathbf{A}_\sigma^{-1}\mathbf{E}\mathbf{A}_\sigma^{-1}\mathbf{B}, \dots, (\mathbf{A}_\sigma^{-1}\mathbf{E})^{q-1}\mathbf{A}_\sigma^{-1}\mathbf{B} \right\}, \quad (2.29)$$

where  $\mathbf{A}_\sigma := (\sigma\mathbf{E} - \mathbf{A})$  and  $\sigma \notin \lambda(\mathbf{E}^{-1}\mathbf{A})$ .  $\blacktriangle$

In this case, the basis takes the form  $\mathbf{V} = [\mathbf{V}_0, \dots, \mathbf{V}_{q-1}] \in \mathbb{C}^{n \times mq}$ , where each block  $\mathbf{V}_\ell$  is computed via the recurrence  $\mathbf{A}_\sigma\mathbf{V}_0 = \mathbf{B}$ ,  $\mathbf{A}_\sigma\mathbf{V}_\ell = \mathbf{E}\mathbf{V}_{\ell-1}$ ,  $\ell = 1, 2, \dots, q-1$ . This means that multiple linear systems of equations (LSEs) with varying right-hand sides have to be solved, wherefore an LU decomposition of  $\mathbf{A}_\sigma$  is recommended. The numerical solution of LSEs will be discussed in more detail in Section 2.5.

**Union of subspaces** It is possible to construct an orthonormal basis  $\mathbf{V}$  spanning the union of several Krylov subspaces. For instance, the union  $\mathcal{K}_q(\mathbf{E}^{-1}\mathbf{A}, \mathbf{E}^{-1}\mathbf{B}) \cup \mathcal{K}_q(\mathbf{A}^{-1}\mathbf{E}, \mathbf{A}^{-1}\mathbf{B})$  combines two rational Krylov subspaces (2.29) for  $\sigma \rightarrow \infty$  and  $\sigma = 0$ . This *extended* Krylov subspace is also referred to as *Krylov-Plus-Inverted-Krylov* (KPIK) and yields

$$\text{span} \left\{ \mathbf{E}^{-1}\mathbf{B}, \dots, (\mathbf{E}^{-1}\mathbf{A})^{q-1}\mathbf{E}^{-1}\mathbf{B}, \mathbf{A}^{-1}\mathbf{B}, \dots, (\mathbf{A}^{-1}\mathbf{E})^{q-1}\mathbf{A}^{-1}\mathbf{B} \right\} \supseteq \text{ran}(\mathbf{V}). \quad (2.30)$$

Alternatively, different shifts  $\sigma_i$ ,  $i = 1, \dots, r$  with respective multiplicities  $q_i$  can be employed. The union  $\mathcal{K}_{q_1}(\mathbf{A}_{\sigma_1}^{-1}\mathbf{E}, \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}) \cup \dots \cup \mathcal{K}_{q_r}(\mathbf{A}_{\sigma_r}^{-1}\mathbf{E}, \mathbf{A}_{\sigma_r}^{-1}\mathbf{B})$  leads to the block *multipoint* rational Krylov subspace

$$\text{span} \left\{ \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \dots, (\mathbf{A}_{\sigma_1}^{-1}\mathbf{E})^{q_1-1}\mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \dots, \mathbf{A}_{\sigma_r}^{-1}\mathbf{B}, \dots, (\mathbf{A}_{\sigma_r}^{-1}\mathbf{E})^{q_r-1}\mathbf{A}_{\sigma_r}^{-1}\mathbf{B} \right\} \supseteq \text{ran}(\mathbf{V}), \quad (2.31)$$

where  $\mathbf{A}_{\sigma_i} := (\sigma_i \mathbf{E} - \mathbf{A})$  and  $\sigma_i \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$ . Note that, for each new shift, an LU decomposition of  $\mathbf{A}_{\sigma_i}$  is required.

**Nestedness property** An important feature of Krylov subspaces is that they can be *nested*. This means that a *cascaded basis* for the same subspace can be constructed by adding new basis vectors that depend on previously computed directions. To illustrate this more, we explain two different ways to construct the subspace (2.30) in the following. One can first compute the Krylov subspaces  $\mathcal{K}_q(\mathbf{E}^{-1}\mathbf{A}, \mathbf{E}^{-1}\mathbf{B})$  and  $\mathcal{K}_q(\mathbf{A}^{-1}\mathbf{E}, \mathbf{A}^{-1}\mathbf{B})$  individually, and then form an orthonormal basis  $\mathbf{V}$  by the union of both spaces. Alternatively, the same subspace can be calculated in a *cascaded manner* via the following iteration

$$\mathbf{V}_1 = \tilde{\mathbf{V}}_1 = [\mathbf{V}_1^\infty, \mathbf{V}_1^0], \quad \mathbf{V}_k = [\mathbf{E}^{-1}\mathbf{A}\mathbf{V}_{k-1}^\infty, \mathbf{A}^{-1}\mathbf{E}\mathbf{V}_{k-1}^0], \quad \tilde{\mathbf{V}}_k = [\tilde{\mathbf{V}}_{k-1}, \mathbf{V}_k], \quad k \leq q, \quad (2.32)$$

where  $\mathbf{V}_1^\infty = \mathbf{E}^{-1}\mathbf{B}$  and  $\mathbf{V}_1^0 = \mathbf{A}^{-1}\mathbf{B}$ . In the former approach, the orthogonalization can only be performed a-posteriori, i.e. after the union of the subspaces. In the latter approach, however, a *modified* Gram-Schmidt procedure yielding orthonormal bases in each iteration can be used, which is numerically much more stable.

The nestedness or cascaded property also holds for the multipoint Krylov subspace (2.31). Exemplarily for multiplicities  $q_1 = \dots = q_r = 1$ , this means that the following standard and cascaded subspaces

$$\text{span} \left\{ \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \mathbf{A}_{\sigma_2}^{-1}\mathbf{B}, \dots, \mathbf{A}_{\sigma_r}^{-1}\mathbf{B} \right\}, \quad (2.33)$$

$$\text{span} \left\{ \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \mathbf{A}_{\sigma_2}^{-1}\mathbf{E}\mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \dots, \mathbf{A}_{\sigma_r}^{-1}\mathbf{E} \dots \mathbf{A}_{\sigma_2}^{-1}\mathbf{E}\mathbf{A}_{\sigma_1}^{-1}\mathbf{B} \right\}, \quad (2.34)$$

are equal, whereby the nested basis  $\tilde{\mathbf{V}}$  is computed iteratively as follows

$$\mathbf{V}_1 = \tilde{\mathbf{V}}_1 = \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}, \quad \mathbf{V}_k = \mathbf{A}_{\sigma_k}^{-1}\mathbf{E}\mathbf{V}_{k-1}, \quad \tilde{\mathbf{V}}_k = [\tilde{\mathbf{V}}_{k-1}, \mathbf{V}_k], \quad k \leq r. \quad (2.35)$$

Note that the nested construction of the subspace is especially exploited in iterative algorithms, such as e.g. the extended Krylov subspace method (EKSM) [245], the rational Krylov subspace method (RKSM) [82, 83] or the alternating direction implicit (ADI) iteration [267, 202, 169]. These techniques are often employed for the iterative solution of linear matrix equations, which is discussed in Section 2.4.

### Tangential Krylov subspace

In case of block Krylov subspaces with  $\mathbf{B} \in \mathbb{R}^{n \times m}$ , the basis grows by  $m$  columns at every new step. To overcome this fast growth of the dimension, one can choose a single tangential direction  $\mathbf{r} \in \mathbb{C}^m$  and apply the vector  $\mathbf{B}\mathbf{r} \in \mathbb{C}^n$  to the subspaces given before. This reduces the number of new columns per iteration to one. Naturally, it is also possible to choose one tangential direction  $\{\mathbf{r}_i\}_{i=1}^r$  for each shift  $\{\sigma_i\}_{i=1}^r$ , leading to the *tangential multipoint* Krylov subspace

$$\text{span} \left\{ \mathbf{A}_{\sigma_1}^{-1}\mathbf{B}\mathbf{r}_1, \mathbf{A}_{\sigma_2}^{-1}\mathbf{B}\mathbf{r}_2, \dots, \mathbf{A}_{\sigma_r}^{-1}\mathbf{B}\mathbf{r}_r \right\} \supseteq \text{ran}(\mathbf{V}). \quad (2.36)$$

Unfortunately, the nestedness property does generally not hold for (2.36) due to the employment of different tangential directions [84, Hei18]. Hence, in such case the basis cannot be

computed in a nested manner like (2.34) anymore. Further note that tangential Krylov subspaces constitute a very general formulation, since the choice  $\mathbf{r}_i \rightarrow \mathbf{I}_m$  leads to a block Krylov subspace, while for  $m=1$  it simplifies according to  $\mathbf{B} \rightarrow \mathbf{b}$  and  $\mathbf{r}_i \rightarrow 1$ .

**Remark 2.1** (Output Krylov subspaces). For all stated input Krylov subspaces, dual counterparts can be considered by replacing  $\mathbf{A} \rightarrow \mathbf{A}^\top$ ,  $\mathbf{E} \rightarrow \mathbf{E}^\top$  and  $\mathbf{B} \rightarrow \mathbf{C}^\top$ . For example, the dual counterpart of (2.29) is given by  $\mathcal{K}_q(\mathbf{A}_\mu^{-\top} \mathbf{E}^\top, \mathbf{A}_\mu^{-\top} \mathbf{C}^\top)$ , where  $\mathbf{C} \in \mathbb{R}^{p \times n}$ ,  $\mathbf{A}_\mu := (\mu \mathbf{E} - \mathbf{A})$  and the output shift  $\mu \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1} \mathbf{A})$  is not necessarily different from the input shift  $\sigma$ . Moreover, in the tangential case (2.36), the right directions  $\mathbf{r}_i \in \mathbb{C}^m$  should be substituted by *left* tangential directions  $\mathbf{l}_i \in \mathbb{C}^p$ . This yields the tangential *output* Krylov subspace

$$\text{span} \left\{ \mathbf{A}_{\mu_1}^{-\top} \mathbf{C}^\top \mathbf{l}_1, \mathbf{A}_{\mu_2}^{-\top} \mathbf{C}^\top \mathbf{l}_2, \dots, \mathbf{A}_{\mu_r}^{-\top} \mathbf{C}^\top \mathbf{l}_r \right\} \supseteq \text{ran}(\mathbf{W}), \quad (2.37)$$

where the basis  $\mathbf{W} \in \mathbb{C}^{n \times r}$ . The choice  $\mathbf{l}_i \rightarrow \mathbf{I}_p$  leads to a block output Krylov subspace, while for  $p=1$  it simplifies according to  $\mathbf{C}^\top \rightarrow \mathbf{c}$  and  $\mathbf{l}_i \rightarrow 1$ .  $\triangle$

## 2.4 Numerical solution of linear matrix equations

Throughout this thesis, we will see that Sylvester and Lyapunov equations play an important role in systems theory and model reduction. For instance, there is a tight connection between rational Krylov subspaces and Sylvester equations [116]. Furthermore, Lyapunov equations arise in the context of controllability and observability Gramians, and are thus relevant for system norms and balancing-based model reduction.

In this section, we discuss the numerical solution of *linear* Sylvester and Lyapunov equations. The aim is to understand the different existing solution methods for the linear setting, as the concepts can be extended to more general cases like e.g. bilinear and quadratic-bilinear matrix equations. Let us consider linear Sylvester equations of the form

$$\mathbf{A} \mathbf{X} \mathbf{F}_1 + \mathbf{E} \mathbf{X} \mathbf{F}_2 + \mathbf{B} \mathbf{F}_3 = \mathbf{0}, \quad (2.38a)$$

$$\mathbf{A}^\top \mathbf{Y} \mathbf{H}_1 + \mathbf{E}^\top \mathbf{Y} \mathbf{H}_2 + \mathbf{C}^\top \mathbf{H}_3 = \mathbf{0}, \quad (2.38b)$$

where  $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$ ,  $\mathbf{C} \in \mathbb{R}^{p \times n}$ ;  $\mathbf{F}_1, \mathbf{F}_2, \mathbf{H}_1, \mathbf{H}_2 \in \mathbb{R}^{r \times r}$  and  $\mathbf{F}_3 \in \mathbb{R}^{m \times r}$ ,  $\mathbf{H}_3 \in \mathbb{R}^{p \times r}$ . We are interested in the solutions  $\mathbf{X} \in \mathbb{R}^{n \times r}$  and  $\mathbf{Y} \in \mathbb{R}^{n \times r}$ . Furthermore, let us consider linear Lyapunov equations of the form

$$\mathbf{A} \mathbf{P} \mathbf{E}^\top + \mathbf{E} \mathbf{P} \mathbf{A}^\top + \mathbf{B} \mathbf{B}^\top = \mathbf{0}, \quad (2.39a)$$

$$\mathbf{A}^\top \mathbf{Q} \mathbf{E} + \mathbf{E}^\top \mathbf{Q} \mathbf{A} + \mathbf{C}^\top \mathbf{C} = \mathbf{0}, \quad (2.39b)$$

which are a special case of the above Sylvester equations with  $\mathbf{P} \in \mathbb{R}^{n \times n}$  and  $\mathbf{Q} \in \mathbb{R}^{n \times n}$ .

In what follows, we will discuss direct and iterative methods for solving large linear *Lyapunov* equations only. Note, however, that the presented techniques can be generalized to solve Sylvester equations as well.

### Direct solution

Medium-sized Lyapunov equations can be solved with *direct methods*, such as the Bartels-Stewart algorithm [49] and Hammarling's method [118]. In the Bartels-Stewart algorithm

(MATLAB: `lyap`), the solutions  $\mathbf{P}, \mathbf{Q}$  of the Lyapunov equations are explicitly computed. On the contrary, Hammarling's method (MATLAB: `lyapchol`) directly solves for the lower triangular Cholesky factors  $\mathbf{S}, \mathbf{R} \in \mathbb{R}^{n \times n}$  of the positive definite solutions  $\mathbf{P} = \mathbf{S}\mathbf{S}^\top$  and  $\mathbf{Q} = \mathbf{R}\mathbf{R}^\top$ , without computing the latter explicitly. Note that Hammarling's method requires the condition  $\lambda(\mathbf{A}, \mathbf{E}) \subset \mathbb{C}_-$  for positive definiteness of  $\mathbf{P}, \mathbf{Q}$ . Further note that this approach is mostly used for square-root balanced truncation (cf. Section 3.3.2), as computations with the explicitly formed solutions are completely avoided.

Both direct methods pose a high computational effort  $\mathcal{O}(n^3)$  and storage complexity  $\mathcal{O}(n^2)$ , since a Hessenberg-Schur decomposition of the pencil  $(\mathbf{A}, \mathbf{E})$  to upper triangular form is employed. Moreover, *dense* forward substitutions are involved.

### Iterative solution

In the large-scale setting, Lyapunov equations are solved with *iterative methods* [235, 246]. The main idea is to approximate the solutions by  $\mathbf{P} \approx \hat{\mathbf{P}} = \mathbf{Z}_c \mathbf{Z}_c^\top$  and  $\mathbf{Q} \approx \hat{\mathbf{Q}} = \mathbf{Z}_o \mathbf{Z}_o^\top$ , using *low-rank Cholesky factors*  $\mathbf{Z}_c \in \mathbb{R}^{n \times q_c}$ ,  $\mathbf{Z}_o \in \mathbb{R}^{n \times q_o}$  of dimension  $q_c, q_o \ll n$ . There exist several approaches to compute low-rank Cholesky factors. In the following, we revisit three different techniques that have proven successful over the last years. The discussion is restricted to the controllability Lyapunov equation (2.39a), but similar considerations hold also for the observability Lyapunov equation (2.39b).

The first type of methods employ a projection onto Krylov subspaces, and then solve the reduced Lyapunov equation by direct solution techniques. This idea was first proposed in [235] and further studied in [133, 135], where *classical* Krylov subspaces are used for projection. Later, in [245], an *extended* subspace given by the union of two Krylov subspaces is employed, which is referred to as *extended Krylov subspace method* (EKSM). Only since recently [82, 83, 84], *rational* Krylov subspaces are also being exploited in order to obtain a low-rank approximation with a smaller subspace. The approach is known as *rational Krylov subspace method* (RKSM), and is usually equipped with an adaptive shift selection strategy. The main steps of these projection-based techniques are the following. At first, orthonormal bases  $\mathbf{V}, \mathbf{W}$  for an input and output Krylov subspace are computed to obtain reduced matrices  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E} \mathbf{V}$ ,  $\mathbf{A}_r = \mathbf{W}^\top \mathbf{A} \mathbf{V}$  and  $\mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$ . Then, the reduced Lyapunov equation

$$\mathbf{A}_r \mathbf{P}_r \mathbf{E}_r^\top + \mathbf{E}_r \mathbf{P}_r \mathbf{A}_r^\top + \mathbf{B}_r \mathbf{B}_r^\top = \mathbf{0} \quad (2.40)$$

is solved for the Cholesky factor  $\mathbf{S}_r$  of the solution  $\mathbf{P}_r = \mathbf{S}_r \mathbf{S}_r^\top$  by Hammarling's method. The low-rank approximation is finally given by  $\hat{\mathbf{P}}_{\text{RKSM}} = \mathbf{V} \mathbf{P}_r \mathbf{V}^\top = \mathbf{Z}_{\text{RKSM}} \mathbf{Z}_{\text{RKSM}}^\top$ , where the low-rank Cholesky factor is  $\mathbf{Z}_{\text{RKSM}} = \mathbf{V} \mathbf{S}_r$ . In general, the orthonormal bases are not computed at once, but are augmented gradually until the desired accuracy for  $\hat{\mathbf{P}}_{\text{RKSM}}$  is achieved. Hereby, the residual for the  $k$ -th iteration

$$\mathbf{Res}_k = \mathbf{A} \mathbf{V}_k \mathbf{P}_{r,k} \mathbf{V}_k^\top \mathbf{E}^\top + \mathbf{E} \mathbf{V}_k \mathbf{P}_{r,k} \mathbf{V}_k^\top \mathbf{A}^\top + \mathbf{B} \mathbf{B}^\top \quad (2.41)$$

is usually employed as stopping criterion. A low-rank formulation of the residual that allows for an efficient computation is given e.g. in [278, 273].

Another existing approach is given by the *alternating direction implicit* (ADI) iteration, which was first employed for the approximate solution of Lyapunov equations in [267]. The low-rank formulation is due to [202, 169], where the method is also denoted as *low-rank Smith* (LR-Smith) or *Cholesky factor ADI* (CF-ADI) iteration. The idea is to cumulatively construct

the low-rank Cholesky factor  $\mathbf{Z}_{\text{ADI}} = [\mathbf{Z}_1, \dots, \mathbf{Z}_q]$  of the approximation  $\hat{\mathbf{P}}_{\text{ADI}} = \mathbf{Z}_{\text{ADI}} \mathbf{Z}_{\text{ADI}}^\top$  using the following iteration

$$\begin{aligned} \mathbf{Z}_1 &= -\sqrt{2\text{Re}(\sigma_1)} (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}, \\ \mathbf{Z}_k &= \sqrt{\frac{\text{Re}(\sigma_k)}{\text{Re}(\sigma_{k-1})}} \left( \mathbf{I} - (\sigma_k + \bar{\sigma}_{k-1}) (\sigma_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right) \mathbf{Z}_{k-1}, \quad k = 2, \dots, q. \end{aligned} \quad (2.42)$$

Over the years, several works have been published concerning the efficient implementation and extension of the ADI to Sylvester and Riccati equations, see e.g. [41, 42]. Moreover, the relation of the ADI to rational Krylov subspace methods has been elucidated in [82, 90, 275]. Indeed, the ADI iteration is equivalent to the RKSM for the special case of  $\mathcal{H}_2$ -pseudo-optimal shifts. In general, the adaptive selection of shift parameters has been topic of extensive research over the last years [39, 84, 151, 279]. Amongst all shift selection procedures, the Wachspress approach [267], the heuristic Penzl strategy [202] and the mirrored Ritz values of the reduced pencil  $(\mathbf{A}_r, \mathbf{E}_r)$ , i.e.  $\sigma_i \leftarrow -\bar{\lambda}_{r,i}$ , are probably the best known techniques.

A third option is to consider the matrix equation as a linear system of equations, and then compute the low-rank approximation by means of an iterative Krylov-based solver. Using the vectorization property (2.13), the Lyapunov equation (2.39a) can be reformulated as a linear system of  $n^2$  equations of the form

$$(\mathbf{E} \otimes \mathbf{A} + \mathbf{A} \otimes \mathbf{E}) \text{vec}(\mathbf{P}) = -\text{vec}(\mathbf{B}\mathbf{B}^\top). \quad (2.43)$$

Due to the exploding dimension, the solution of (2.43) by direct solvers is only feasible for small-scale Lyapunov equations. Instead, *low-rank variants* of iterative solvers like e.g. the conjugate gradient (CG) or the generalized minimal residual method (GMRES) should be employed [150, 28, 21]. The low-rank formulation of the iterative algorithms is basically achieved by applying a low-rank truncation operator  $\mathcal{T}$ , i.e. a column compression, to each intermediate solution and residual. Furthermore, a suitable preconditioner with low-rank structure is essential for a fast convergence of the methods. The low-rank ADI iteration, for instance, can serve as preconditioner.

Finally, note that there exist further low-rank solution techniques for matrix equations, such as e.g. the sign function iteration [44], the data-sparse method [99] and the Riemannian optimization approach [266].

## 2.5 Numerical solution of linear systems of equations

The solution of large sparse linear systems of equations (LSEs) of the form  $\mathbf{A}\mathbf{X} = \mathbf{B}$  is fundamental for different applications within this thesis. For instance, the construction of the rational Krylov subspace (2.29) involves the numerical solution of the following LSEs

$$\mathbf{A}_\sigma \mathbf{V}_0 = \mathbf{B}, \quad \mathbf{A}_\sigma \mathbf{V}_\ell = \mathbf{E} \mathbf{V}_{\ell-1}, \quad \ell = 1, 2, \dots, q-1. \quad (2.44)$$

Moreover, implicit time integration schemes (cf. Section 6.2) or numerical algorithms based on the Newton-Raphson method (cf. Algorithm 7.1) also rely on the solution of LSEs.

In the following, we briefly explain direct and iterative methods for solving LSEs. The presentation is especially focused on the solution of *sparse* LSEs, i.e. the matrices  $\mathbf{A} \in \mathbb{R}^{n \times n}$  and  $\mathbf{B} \in \mathbb{R}^{n \times m}$  are *sparse*.



## Direct solution

Direct solvers are based on a factorization of the matrix  $\mathbf{A}$ , followed by forward/backward substitutions to calculate the solution  $\mathbf{X}$ . Depending on the properties of  $\mathbf{A}$  (e.g. general matrix, symmetric positive definite, etc.), a specific factorization (e.g. LU, Cholesky, LDL) might be particularly suitable. Most commonly, an LU decomposition  $\mathbf{A} = \mathbf{L}\mathbf{U}$  into a lower and upper triangular matrix is employed. The factorization is often performed with pivoting, i.e. row/column permutations, to improve the numerical stability, guarantee its completion and reduce the fill-in of the sparse LU factors. Once the matrices  $\mathbf{L}$  and  $\mathbf{U}$  are available, the solution of the LSE is obtained by simple forward and backward substitutions. Thus, the numerically most expensive part of solving LSEs is the LU decomposition. Note, however, that the LU factors can be reused to solve multiple LSEs with varying right-hand sides. If the left-hand side matrix  $\mathbf{A}$  changes – e.g. due to a new shift or time-step – then a new LU factorization is required. Further note that, depending on the sparsity of  $\mathbf{A}$  and the pivoting, the LU factors may become more or less *dense*, making their storage possibly difficult.

Direct dense/sparse solvers are provided in libraries like BLAS, LAPACK, UMFPACK, Intel MKL PARDISO, MUMPS, etc. They are also available in MATLAB (`mldivide`, `\`) and SciPy (`scipy.linalg.solve`), whose commands basically rely on the routines from the aforementioned libraries.

## Iterative solution

In the large-scale setting, the storage of the LU factors of  $\mathbf{A}$  becomes infeasible due to limited RAM. In such case, iterative methods [232] can be employed to approximately solve LSEs.

The main idea of *Krylov-based* iterative solvers is to find an approximate solution  $\mathbf{X}_*$  for the LSE by generating a sequence of solutions  $\mathbf{X}_k$  that iteratively tries to minimize the residual  $\mathbf{Res}_k = \mathbf{B} - \mathbf{A}\mathbf{X}_k$ . Different iterative solvers exist, such as e.g. the *conjugate gradient* (CG) [123], the nonsymmetric *biconjugate gradient stabilized* (BiCGstab) [262], the *minimum residual* (MinRes) [208] and the nonsymmetric *generalized minimal residual* (GMRES) [255]. These algorithms (and variants thereof) are also available in MATLAB and SciPy. Generally, the convergence of the methods can be significantly accelerated by a suitable preconditioner  $\mathbf{M}$ . A possible choice is to use the factors of an *incomplete* LU decomposition, i.e.  $\mathbf{M} = \mathbf{L}\mathbf{U}$ . But other preconditioning techniques exist. Moreover, the schemes are sometimes restarted after a certain number of iterations to limit the storage and computational effort.

Besides the aforementioned Krylov-based approaches, there exist other iterative methods for solving LSEs. Examples are the Jacobi and Gauss-Seidel iteration – which are based on matrix splitting – or multigrid techniques (e.g. algebraic multigrid (AMG)).

## 2.6 Error measures

The performance of MOR approaches is usually analyzed *qualitatively* via bode plots, output curves or color maps, and *quantitatively* via error measures. In the following we focus on the quantitative analysis of ROMs in *time-domain*, which is mainly employed in the context of nonlinear dynamical systems.

In the linear setting, the error is usually quantified in frequency-domain using the transfer function of full- and reduced-order model (cf. Eqs. (3.31) and (3.32)). Transfer functions and

system norms can also be employed in polynomial MOR to assess the approximation quality of e.g. bilinear and quadratic-bilinear reduced models (cf. Eq. (5.24)). Unfortunately, the input-output behavior of *general* nonlinear systems cannot be characterized by such system-theoretic concepts. Therefore, in the nonlinear case the evaluation is mostly conducted in *time-domain* by analyzing the ROM via simulation runs for different test signals.

Depending on the goal or application, the accuracy can be measured in terms of the whole state vector  $\mathbf{x}(t)$  or certain quantities of interest encoded in  $\mathbf{y}(t)$ . In what follows, we distinguish between point-wise and norm-wise error measures.

### Point-wise error measures

The relative approximation error for different reduction methods “m” can be given *point-wise* in time by

$$e_{x,\text{rel},(\cdot)}^m(t_k) = \frac{\|\mathbf{x}(t_k) - \mathbf{V}\mathbf{x}_r^m(t_k)\|_{(\cdot)}}{\|\mathbf{x}(t_k)\|_{(\cdot)}}, \quad e_{y,\text{rel},(\cdot)}^m(t_k) = \frac{\|\mathbf{y}(t_k) - \mathbf{y}_r^m(t_k)\|_{(\cdot)}}{\|\mathbf{y}(t_k)\|_{(\cdot)}}, \quad (2.45)$$

using a desired vector norm  $(\cdot) \in \{1, 2, \infty, \dots\}$ . Note that the state  $\mathbf{x}(t_k) \in \mathbb{R}^n$ ,  $\mathbf{x}_r^m(t_k) \in \mathbb{R}^r$  and output vectors  $\mathbf{y}(t_k)$ ,  $\mathbf{y}_r^m(t_k) \in \mathbb{R}^p$  are evaluated at every simulated time-step  $t_k$ , meaning that one obtains a *time-series* of error points  $e_{x,\text{rel},(\cdot)}^m(t_k)$  or  $e_{y,\text{rel},(\cdot)}^m(t_k)$ .

### Norm-wise error measures

The error can also be measured *norm-wise* in time. Depending on the type of norm that is employed to compute the error, one can differentiate two cases.

**Matrix norms** The error can be measured norm-wise in time using a desired *matrix* norm  $(\star) \in \{1, 2, \infty, \text{F}\}$  as follows:

$$e_{x,\text{rel},(\star)}^m = \frac{\|\mathbf{x} - \mathbf{V}\mathbf{x}_r^m\|_{(\star)}}{\|\mathbf{x}\|_{(\star)}}, \quad e_{y,\text{rel},(\star)}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{(\star)}}{\|\mathbf{y}\|_{(\star)}}. \quad (2.46)$$

In contrast to the point-wise case, the simulated data is now collected and evaluated as matrices, i.e.  $\mathbf{x} \in \mathbb{R}^{n \times n_s}$ ,  $\mathbf{x}_r^m \in \mathbb{R}^{r \times n_s}$  and  $\mathbf{y}$ ,  $\mathbf{y}_r^m \in \mathbb{R}^{p \times n_s}$ . Consequently, the resulting error measures  $e_{x,\text{rel},(\star)}^m$  and  $e_{y,\text{rel},(\star)}^m$  are not time-series anymore.

**Signal norms** The error can be measured norm-wise in time using a desired *signal* norm  $\mathcal{L}_p \in \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_\infty, \dots\}$  as follows:

$$e_{x,\text{rel},\mathcal{L}_p}^m = \frac{\|\mathbf{x} - \mathbf{V}\mathbf{x}_r^m\|_{\mathcal{L}_p}}{\|\mathbf{x}\|_{\mathcal{L}_p}}, \quad e_{y,\text{rel},\mathcal{L}_p}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\mathcal{L}_p}}{\|\mathbf{y}\|_{\mathcal{L}_p}}. \quad (2.47)$$

In order to understand how the simulated data is evaluated in this case, we explicitly provide the output error measures for the  $\mathcal{L}_1$ -,  $\mathcal{L}_2$ - and  $\mathcal{L}_\infty$ -norm:

$$e_{y,\text{rel},\mathcal{L}_1}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\mathcal{L}_1}}{\|\mathbf{y}\|_{\mathcal{L}_1}} := \frac{\sum_{k=1}^{n_s} \|\mathbf{y}(t_k) - \mathbf{y}_r^m(t_k)\|_2}{\sum_{k=1}^{n_s} \|\mathbf{y}(t_k)\|_2}, \quad (2.48a)$$

$$e_{y,\text{rel},\mathcal{L}_2}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\mathcal{L}_2}}{\|\mathbf{y}\|_{\mathcal{L}_2}} := \frac{\sqrt{\sum_{k=1}^{n_s} \|\mathbf{y}(t_k) - \mathbf{y}_r^m(t_k)\|_2^2}}{\sqrt{\sum_{k=1}^{n_s} \|\mathbf{y}(t_k)\|_2^2}}, \quad (2.48b)$$

$$e_{y,\text{rel},\mathcal{L}_\infty}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\mathcal{L}_\infty}}{\|\mathbf{y}\|_{\mathcal{L}_\infty}} := \frac{\max_k \|\mathbf{y}(t_k) - \mathbf{y}_r^m(t_k)\|_2}{\max_k \|\mathbf{y}(t_k)\|_2}. \quad (2.48c)$$

Note that, by definition, the  $\mathcal{L}_2$  signal error norm  $e_{y,\text{rel},\mathcal{L}_2}^m$  from Eq. (2.48b) is equivalent to the Frobenius matrix error norm  $e_{y,\text{rel},\text{F}}^m$  given by

$$e_{y,\text{rel},\text{F}}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\text{F}}}{\|\mathbf{y}\|_{\text{F}}} := \frac{\sqrt{\sum_{i=1}^p \sum_{k=1}^{n_s} |\mathbf{y}_{ik} - \mathbf{y}_{r,ik}^m|^2}}{\sqrt{\sum_{i=1}^p \sum_{k=1}^{n_s} |\mathbf{y}_{ik}|^2}}. \quad (2.49)$$

This means that  $e_{y,\text{rel},\mathcal{L}_2}^m = e_{y,\text{rel},\text{F}}^m$ .

To conclude this section, we want to remark that the analysis of ROMs could also be attempted in frequency-domain using the concepts of nonlinear frequency response function (NLFRF) or nonlinear normal modes (NNMs) best known from structural dynamics [140]. These concepts represent the nonlinear counterpart of the bode plot and eigenmodes of a linear dynamical system. NLFRFs and NNMs can be computed using either the Harmonic Balance (HB) method (cf. Section 10.4.3) or shooting techniques, both usually combined with numerical path continuation [200, 142]. Since these procedures are fairly expensive, MOR has been lately applied to accelerate their computation (see e.g. [281, 250, 251]). One could evaluate the performance of a reduction approach not only in time-domain via simulation runs, but also in frequency-domain by comparing the NLFRF of FOM and ROM. This would correspond to the classical evaluation with bode plots used in the linear setting. Furthermore, this approach allows to analyze the behavior of a nonlinear system for several excitation frequencies and amplitudes, instead of evaluating it for a single test signal.



# Chapter 3

## Linear Model Order Reduction

In this chapter we deal with *linear* systems theory and model order reduction. After the repetition of crucial system-theoretic concepts and an overview of linear reduction methods, we review the classical frequency-domain interpretation of moment matching. We also discuss the equivalence between Krylov subspaces and Sylvester equations. After that, we focus on the time-domain interpretation of moment matching based on signal generators and steady-state considerations. The chapter also treats  $\mathcal{H}_2$ -optimal MOR, and discusses the automatic selection of both shifts and reduced order in our implemented CRKSM algorithm.

### 3.1 Linear time-invariant systems

A generalized state-space representation of a linear time-invariant (LTI) system is given by

$$\Sigma : \begin{cases} \mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{B} \mathbf{u}(t), & \mathbf{x}(0) = \mathbf{x}_0, \\ \mathbf{y}(t) = \mathbf{C} \mathbf{x}(t), \end{cases} \quad (3.1a)$$

$$(3.1b)$$

where  $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$  and  $\mathbf{C} \in \mathbb{R}^{p \times n}$ .  $\mathbf{x}(t) \in \mathbb{R}^n$  denotes the state vector and  $\mathbf{x}_0$  the initial condition of the system. The vectors  $\mathbf{u}(t) \in \mathbb{R}^m$  and  $\mathbf{y}(t) \in \mathbb{R}^p$  ( $m, p \ll n$ ) contain the inputs and outputs of the system, respectively. In case of  $m, p > 1$ , one speaks of a MIMO system. The special case  $m = p = 1$ , where  $\mathbf{B} \rightarrow \mathbf{b} \in \mathbb{R}^n$ ,  $\mathbf{C} \rightarrow \mathbf{c}^\top \in \mathbb{R}^{1 \times n}$ ,  $\mathbf{u}(t) \rightarrow u(t) \in \mathbb{R}$  and  $\mathbf{y}(t) \rightarrow y(t) \in \mathbb{R}$ , represents a SISO system. For brevity, at times we will use the notation  $\Sigma = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{E})$  to denote the state-space realization (3.1). We do not consider feedthrough terms in this thesis. Thus, the feedthrough matrix  $\mathbf{D} \in \mathbb{R}^{p \times m}$  is assumed to be zero.

**Assumptions** Throughout this thesis the matrix  $\mathbf{E}$  is assumed to be regular, i.e.  $\det(\mathbf{E}) \neq 0$ . Hence, the model is only described by differential equations and does not contain algebraic constraints. Systems with singular descriptor matrix are referred to as *differential-algebraic equations* and have been considered e.g. in [144, 48]. The regularity of  $\mathbf{E}$  theoretically allows us to replace  $\mathbf{A} \rightarrow \mathbf{E}^{-1}\mathbf{A}$  and  $\mathbf{B} \rightarrow \mathbf{E}^{-1}\mathbf{B}$  in order to obtain an *explicit* representation with  $\mathbf{E} = \mathbf{I}$ . Nevertheless, the inverse is usually avoided in practice due to numerical reasons. Thus, the abbreviations  $\tilde{\mathbf{A}} := \mathbf{E}^{-1}\mathbf{A}$  and  $\tilde{\mathbf{B}} := \mathbf{E}^{-1}\mathbf{B}$  will only be used in theoretical statements.

Furthermore, the system  $\Sigma$  is assumed to be *minimal*, i.e. the pair  $(\mathbf{E}^{-1}\mathbf{A}, \mathbf{E}^{-1}\mathbf{B})$  is controllable (cf. Def. 3.2) and the pair  $(\mathbf{C}, \mathbf{E}^{-1}\mathbf{A})$  is observable (cf. Def. 3.3). This way, the state-space model (3.1) represents a *minimal realization* with the smallest possible dimension.

Finally, we focus on asymptotically stable LTI systems, meaning that all eigenvalues are in the open left-half of the complex plane, i.e.  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$ . This restriction is required to ensure that the Gramians and system norms introduced later are defined.

### Time-domain input-output characterization

It is well known that the analytical solution of the state differential equation (3.1a) is composed of two parts: a homogeneous solution describing the response to an initial condition  $\mathbf{x}(0)$ , and a particular solution for given (causal) inputs  $\mathbf{u}(t)$  (cf. Eq. (3.81)). Inserting the overall solution  $\mathbf{x}(t)$  into the output equation (3.1b) yields the input-output characterization in time-domain

$$\mathbf{y}(t) = \int_{\tau=0}^t \underbrace{\mathbf{C} e^{\tilde{\mathbf{A}}(t-\tau)} \tilde{\mathbf{B}}}_{\mathbf{g}(t-\tau)} \mathbf{u}(\tau) d\tau + \mathbf{C} e^{\tilde{\mathbf{A}}t} \mathbf{x}_0, \quad (3.2)$$

with the (causal, i.e.  $\mathbf{g}(t) = \mathbf{0}$  for  $t < 0$ ) impulse response matrix

$$\mathbf{g}(t) = \mathbf{C} e^{\tilde{\mathbf{A}}t} \tilde{\mathbf{B}} \sigma(t) \in \mathbb{R}^{p \times m}. \quad (3.3)$$

Each output response  $y_i(t)$  for  $i = 1, \dots, p$  is given by

$$y_i(t) = \sum_{j=1}^m \int_{\tau=0}^t g_{ij}(t-\tau) u_j(\tau) d\tau + \mathbf{c}_i^\top e^{\tilde{\mathbf{A}}t} \mathbf{x}_0, \quad (3.4)$$

where  $g_{ij}(t) = \mathbf{c}_i^\top e^{\tilde{\mathbf{A}}t} \tilde{\mathbf{b}}_j \sigma(t)$  denotes the  $(i, j)$ -th entry of the impulse response matrix  $\mathbf{g}(t)$ .

### Frequency-domain input-output characterization

The input-output behavior of LTI systems can also be characterized in frequency-domain.

**Definition 3.1** (Laplace transform). The Laplace transform of a function  $f(t) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$  is

$$F(s) := \mathcal{L}\{f(t)\}(s) := \int_{t=0}^{\infty} f(t) e^{-st} dt. \quad (3.5)$$

The integral converges, if the complex variable  $s \in H_\gamma = \{s \in \mathbb{C} \mid \operatorname{Re}(s) > \gamma, \gamma \in \mathbb{R}\}$ . ▲

Applying the Laplace transform either to the state-space representation (3.1) or to the time-domain input-output equation (3.2) yields

$$\mathbf{Y}(s) = \underbrace{\mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}}_{\mathbf{G}(s)} \mathbf{U}(s) + \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1} \mathbf{x}_0, \quad (3.6)$$

with the rational transfer function matrix

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} \in \mathbb{C}^{p \times m}. \quad (3.7)$$

Note that  $\mathbf{G}(s)$  also results from the Laplace transform of the impulse response (3.3), i.e.  $\mathbf{G}(s) = \mathcal{L}\{\mathbf{g}(t)\}(s)$ . Each output response  $Y_i(s)$  for  $i = 1, \dots, p$  is given by

$$Y_i(s) = \sum_{j=1}^m G_{ij}(s) U_j(s) + \mathbf{c}_i^\top (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{x}_0, \quad (3.8)$$

where  $G_{ij}(s) = \mathbf{c}_i^\top (s\mathbf{E} - \mathbf{A})^{-1} \tilde{\mathbf{b}}_j = \mathcal{L}\{g_{ij}(t)\}(s)$  denotes the  $(i, j)$ -th entry of the transfer function matrix  $\mathbf{G}(s)$ .

### Diagonal form and pole-residue formulation

By applying a so-called *state-space transformation* of the form

$$\mathbf{x}(t) = \mathbf{T}^{-1}\mathbf{z}(t) \Leftrightarrow \mathbf{z}(t) = \mathbf{T}\mathbf{x}(t) \quad (3.9)$$

with the new state vector  $\mathbf{z}(t)$  and the regular matrix  $\mathbf{T}$ , we gain the transformed system<sup>1</sup>

$$\hat{\Sigma} : \begin{cases} \mathbf{OET}^{-1}\dot{\mathbf{z}}(t) = \mathbf{OAT}^{-1}\mathbf{z}(t) + \mathbf{OB}\mathbf{u}(t), & \mathbf{z}(0) = \mathbf{T}\mathbf{x}_0, \\ \mathbf{y}(t) = \mathbf{CT}^{-1}\mathbf{z}(t). \end{cases} \quad (3.10a)$$

$$(3.10b)$$

The realization of (3.1) changes from  $\Sigma = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{E}) \rightarrow \hat{\Sigma} = (\mathbf{OAT}^{-1}, \mathbf{OB}, \mathbf{CT}^{-1}, \mathbf{OET}^{-1})$ , but other system properties like stability, controllability, observability and input-output behavior remain invariant under such transformations. Note that the transformed system (3.10) may also be brought into an explicit representation  $\hat{\Sigma} = (\mathbf{TAT}^{-1}, \mathbf{TB}, \mathbf{CT}^{-1}, \mathbf{I})$  by choosing the left transformation matrix as  $\mathbf{O} = \mathbf{TE}^{-1}$ .

Particularly important is the transformation of the system to *diagonal form*. For this, we assume *distinct* eigenvalues  $\lambda_l$  of  $\mathbf{E}^{-1}\mathbf{A}$ , meaning that the matrix is diagonalizable. First, the generalized eigenvalue problem  $\mathbf{AX} = \mathbf{EX}\Lambda$  is solved, where  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  is diagonal and the columns of  $\mathbf{X}$  represent the right eigenvectors of  $\mathbf{E}^{-1}\mathbf{A}$ . Then, we choose  $\mathbf{T} = \mathbf{X}^{-1}$  and  $\mathbf{O} = \mathbf{TE}^{-1} = \mathbf{X}^{-1}\mathbf{E}^{-1}$ . This yields the diagonalized realization  $\hat{\Sigma} = (\Lambda, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \mathbf{I})$  with

$$\Lambda = \mathbf{X}^{-1}\mathbf{E}^{-1}\mathbf{A}\mathbf{X} \Leftrightarrow \mathbf{E}^{-1}\mathbf{A} = \mathbf{X}\Lambda\mathbf{X}^{-1}, \quad \Lambda = \mathbf{D}, \quad [\mathbf{X}, \mathbf{D}] = \text{eig}(\mathbf{A}, \mathbf{E}), \quad (3.11a)$$

$$\hat{\mathbf{B}} = \mathbf{X}^{-1}\mathbf{E}^{-1}\mathbf{B}, \quad \text{Bhat} = \mathbf{X} \setminus (\mathbf{E} \setminus \mathbf{B}), \quad (3.11b)$$

$$\hat{\mathbf{C}} = \mathbf{C}\mathbf{X}, \quad \text{Chat} = \mathbf{C} * \mathbf{X}. \quad (3.11c)$$

A short MATLAB implementation is given on the right to illustrate the diagonalization step.

The pole-residue formulation of the transfer function  $\mathbf{G}(s)$  can be easily obtained by exploiting the diagonal form. Remember that a state transformation does not affect the input-output behavior of a dynamical system. Thus, in the following we make use of the transfer function  $\hat{\mathbf{G}}(s) = \hat{\mathbf{C}}(s\mathbf{I} - \Lambda)^{-1}\hat{\mathbf{B}}$  of the diagonalized system  $\hat{\Sigma} = (\Lambda, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \mathbf{I})$ . Writing the inverse matrix  $(s\mathbf{I} - \Lambda)^{-1}$  explicitly we obtain the partial-fraction decomposition [6]

$$\hat{\mathbf{G}}(s) = \sum_{l=1}^n \frac{\Phi_l}{s - \lambda_l} \quad (3.12)$$

with matrix-residues given by

$$\Phi_l = \hat{\mathbf{c}}_l \cdot \hat{\mathbf{b}}_l^{\top} \in \mathbb{C}^{p \times m}, \quad l = 1, \dots, n, \quad (3.13)$$

where  $\hat{\mathbf{c}}_l \in \mathbb{C}^p$ ,  $\hat{\mathbf{C}} = [\hat{\mathbf{c}}_1, \dots, \hat{\mathbf{c}}_n] \in \mathbb{C}^{p \times n}$  and  $\hat{\mathbf{b}}_l^{\top} \in \mathbb{C}^{1 \times m}$ ,  $\hat{\mathbf{B}} = \begin{bmatrix} \hat{\mathbf{b}}_1^{\top} \\ \vdots \\ \hat{\mathbf{b}}_n^{\top} \end{bmatrix} \in \mathbb{C}^{n \times m}$ .

<sup>1</sup>Note that some textbooks use the transformation  $\mathbf{x}(t) = \mathbf{T}\mathbf{z}(t) \Leftrightarrow \mathbf{z}(t) = \mathbf{T}^{-1}\mathbf{x}(t)$ , yielding the transformed system  $\hat{\Sigma} = (\mathbf{OAT}, \mathbf{OB}, \mathbf{CT}, \mathbf{OET})$ .

### Gramians and Lyapunov equations

The properties of controllability and observability can be analyzed – among others – by the system Gramians. These can, in turn, be characterized by means of Lyapunov equations. [6]

**Definition 3.2** (Controllability). The pair  $(\mathbf{A}, \mathbf{B})$  with  $\mathbf{A} \in \mathbb{R}^{n \times n}$  and  $\mathbf{B} \in \mathbb{R}^{n \times m}$  is called controllable, if the controllability matrix

$$\mathcal{R} = [\mathbf{B}, \mathbf{A}\mathbf{B}, \dots, \mathbf{A}^{n-1}\mathbf{B}] \in \mathbb{R}^{n \times n \cdot m} \quad (3.14)$$

has full row rank, i.e.  $\text{rank}(\mathcal{R}) = n$ . Note the relationship with (2.28):  $\text{ran}(\mathcal{R}) = \mathcal{K}_n(\mathbf{A}, \mathbf{B})$ .  $\blacktriangle$

The linear state-space model (3.1) is controllable, if the pair  $(\mathbf{E}^{-1}\mathbf{A}, \mathbf{E}^{-1}\mathbf{B})$  is controllable.

The *controllability Gramian*  $\mathbf{P} \in \mathbb{R}^{n \times n}$  is defined as

$$\mathbf{P} = \int_{\tau=0}^{\infty} e^{\tilde{\mathbf{A}}\tau} \tilde{\mathbf{B}} \tilde{\mathbf{B}}^{\top} e^{\tilde{\mathbf{A}}^{\top}\tau} d\tau = \int_{\tau=0}^{\infty} \mathbf{p}(\tau) \mathbf{p}(\tau)^{\top} d\tau, \quad (3.15)$$

where  $\mathbf{p}(t) = e^{\tilde{\mathbf{A}}t} \tilde{\mathbf{B}} \in \mathbb{R}^{n \times m}$  represents the input-to-state part  $\mathbf{g}_{i/s}(t)$  of the impulse response. The system (3.1) is controllable, if and only if  $\mathbf{P}$  is positive definite.

**Definition 3.3** (Observability). The pair  $(\mathbf{C}, \mathbf{A})$  with  $\mathbf{C} \in \mathbb{R}^{p \times n}$  and  $\mathbf{A} \in \mathbb{R}^{n \times n}$  is called observable, if the observability matrix

$$\mathcal{O} = [\mathbf{C}^{\top}, \mathbf{A}^{\top}\mathbf{C}^{\top}, \dots, (\mathbf{A}^{\top})^{n-1}\mathbf{C}^{\top}]^{\top} \in \mathbb{R}^{p \cdot n \times n} \quad (3.16)$$

has full column rank, i.e.  $\text{rank}(\mathcal{O}) = n$ . Note the relationship:  $\text{ran}(\mathcal{O}^{\top}) = \mathcal{K}_n(\mathbf{A}^{\top}, \mathbf{C}^{\top})$ .  $\blacktriangle$

The linear state-space model (3.1) is observable, if the pair  $(\mathbf{C}, \mathbf{E}^{-1}\mathbf{A})$  is observable.

The *observability Gramian*  $\tilde{\mathbf{Q}} \in \mathbb{R}^{n \times n}$  is defined as

$$\tilde{\mathbf{Q}} = \int_{\tau=0}^{\infty} e^{\tilde{\mathbf{A}}^{\top}\tau} \mathbf{C}^{\top} \mathbf{C} e^{\tilde{\mathbf{A}}\tau} d\tau = \int_{\tau=0}^{\infty} \mathbf{q}(\tau) \mathbf{q}(\tau)^{\top} d\tau, \quad (3.17)$$

where  $\mathbf{q}(t) = e^{\tilde{\mathbf{A}}^{\top}t} \mathbf{C}^{\top} \in \mathbb{R}^{n \times p}$  represents the state-to-output part  $\mathbf{g}_{s/o}(t)$  of the impulse response. The system (3.1) is observable, if and only if  $\tilde{\mathbf{Q}}$  is positive definite.

By definition, the controllability and observability Gramians are symmetric and positive semi-definite. If the system is asymptotically stable, i.e.  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$ , then the Gramians are positive definite  $\mathbf{P} = \mathbf{P}^{\top} \succ \mathbf{0}$ ,  $\tilde{\mathbf{Q}} = \tilde{\mathbf{Q}}^{\top} \succ \mathbf{0}$  and are related to the unique solutions of the following generalized Lyapunov equations

$$\mathbf{A}\mathbf{P}\mathbf{E}^{\top} + \mathbf{E}\mathbf{P}\mathbf{A}^{\top} + \mathbf{B}\mathbf{B}^{\top} = \mathbf{0}, \quad (3.18a)$$

$$\mathbf{A}^{\top}\mathbf{Q}\mathbf{E} + \mathbf{E}^{\top}\mathbf{Q}\mathbf{A} + \mathbf{C}^{\top}\mathbf{C} = \mathbf{0}. \quad (3.18b)$$

Note that  $\mathbf{Q}$  is the solution of (3.18b), whereas  $\tilde{\mathbf{Q}}$  is given by (3.17) and  $\tilde{\mathbf{Q}} = \mathbf{E}^{\top}\mathbf{Q}\mathbf{E}$ .

### System norms

System norms like the  $\mathcal{H}_2$ - and  $\mathcal{H}_{\infty}$ -norm are important measures to analyze dynamical systems. They will later help us to quantify the difference between two systems.



**$\mathcal{L}_2$ -norm (time-domain)**

The time-domain  $\mathcal{L}_2$ -norm of a MIMO linear system  $\Sigma$  is defined on the  $\mathcal{L}_2[0, \infty)$  space by

$$\|\Sigma\|_{\mathcal{L}_2[0, \infty)}^2 = \int_{\tau=0}^{\infty} \|\mathbf{g}(\tau)\|_{\text{F}}^2 d\tau = \int_{\tau=0}^{\infty} \sum_{i=1}^p \sum_{j=1}^m g_{ij}^2(\tau) d\tau, \quad (3.19)$$

where  $\|\mathbf{g}(\tau)\|_{\text{F}}$  denotes the Frobenius norm of the impulse response (3.3). Using the relation  $\|\mathbf{K}\|_{\text{F}}^2 = \text{tr}(\mathbf{K}\mathbf{K}^{\text{T}}) = \text{tr}(\mathbf{K}^{\text{T}}\mathbf{K})$ , we can express the  $\mathcal{L}_2$ -norm in terms of the controllability and observability Gramians  $\mathbf{P}$  and  $\tilde{\mathbf{Q}}$ :

$$\|\Sigma\|_{\mathcal{L}_2}^2 = \int_{\tau=0}^{\infty} \text{tr}(\mathbf{g}(\tau)\mathbf{g}(\tau)^{\text{T}}) d\tau = \text{tr}\left(\mathbf{C} \int_{\tau=0}^{\infty} e^{\tilde{\mathbf{A}}\tau} \tilde{\mathbf{B}}\tilde{\mathbf{B}}^{\text{T}} e^{\tilde{\mathbf{A}}^{\text{T}}\tau} d\tau \mathbf{C}^{\text{T}}\right) = \text{tr}(\mathbf{C}\mathbf{P}\mathbf{C}^{\text{T}}), \quad (3.20a)$$

$$\|\Sigma\|_{\mathcal{L}_2}^2 = \int_{\tau=0}^{\infty} \text{tr}(\mathbf{g}(\tau)^{\text{T}}\mathbf{g}(\tau)) d\tau = \text{tr}\left(\tilde{\mathbf{B}}^{\text{T}} \int_{\tau=0}^{\infty} e^{\tilde{\mathbf{A}}^{\text{T}}\tau} \mathbf{C}^{\text{T}}\mathbf{C} e^{\tilde{\mathbf{A}}\tau} d\tau \tilde{\mathbf{B}}\right) = \text{tr}(\tilde{\mathbf{B}}^{\text{T}}\tilde{\mathbf{Q}}\tilde{\mathbf{B}}). \quad (3.20b)$$

Note that  $\text{tr}(\tilde{\mathbf{B}}^{\text{T}}\tilde{\mathbf{Q}}\tilde{\mathbf{B}}) = \text{tr}(\mathbf{B}^{\text{T}}\mathbf{E}^{-\text{T}}\tilde{\mathbf{Q}}\mathbf{E}^{-1}\mathbf{B}) = \text{tr}(\mathbf{B}^{\text{T}}\mathbf{Q}\mathbf{B})$ , where  $\mathbf{Q} = \mathbf{E}^{-\text{T}}\tilde{\mathbf{Q}}\mathbf{E}^{-1}$ . The time-domain  $\mathcal{L}_2$ -norm can thus be computed by solving one of the Lyapunov equations (3.18).

 **$\mathcal{H}_2$ -norm (frequency-domain)**

The  $\mathcal{H}_2$ -norm of a linear system is most generally defined on the so-called *Hardy space*  $\mathcal{H}_2$ , using the strictly proper, complex-valued transfer function  $\mathbf{G}(x + iy)$  that is analytic in the open right-half plane  $\mathbb{C}_+$ :

$$\|\Sigma\|_{\mathcal{H}_2}^2 = \sup_{x>0} \int_{y=-\infty}^{\infty} \|\mathbf{G}(x + iy)\|_{\text{F}}^2 dy. \quad (3.21)$$

For a stable system with  $\lambda_l \in \mathbb{C}_-$ ,  $l = 1, \dots, n$ , the  $\mathcal{H}_2$ -norm  $\|\Sigma\|_{\mathcal{H}_2}^2$  is equivalent to the frequency-domain  $\mathcal{L}_2$ -norm  $\|\Sigma\|_{\mathcal{L}_2(i\mathbb{R})}^2$  on the imaginary axis (cf. Phragmén-Lindelöf principle [269]). Thus, in such case the  $\mathcal{H}_2$ -norm is defined on the *Hilbert space*  $\mathcal{L}_2(i\mathbb{R})$  and reduces to

$$\|\Sigma\|_{\mathcal{L}_2(i\mathbb{R})}^2 = \frac{1}{2\pi} \int_{\omega=-\infty}^{\infty} \|\mathbf{G}(i\omega)\|_{\text{F}}^2 d\omega, \quad (3.22)$$

where  $\|\mathbf{G}(i\omega)\|_{\text{F}}^2 = \text{tr}(\mathbf{G}(i\omega)\mathbf{G}(-i\omega)^{\text{T}}) = \text{tr}(\mathbf{G}(-i\omega)^{\text{T}}\mathbf{G}(i\omega))$ . Due to Parseval-Plancherel's theorem, the frequency-domain and time-domain  $\mathcal{L}_2$ -norms are equivalent. To sum up:

$$\|\Sigma\|_{\mathcal{H}_2}^2 = \|\Sigma\|_{\mathcal{L}_2(i\mathbb{R})}^2 = \|\Sigma\|_{\mathcal{L}_2[0, \infty)}^2. \quad (3.23)$$

The  $\mathcal{H}_2$ -norm of a linear system can also be expressed in pole-residue formulation [97]

$$\|\Sigma\|_{\mathcal{H}_2}^2 = \sum_{l=1}^n \hat{\mathbf{c}}_l^{\text{T}} \mathbf{G}(-\bar{\lambda}_l) \hat{\mathbf{b}}_l, \quad (3.24)$$

where  $\hat{\mathbf{c}}_l \in \mathbb{C}^p$ ,  $\hat{\mathbf{b}}_l \in \mathbb{C}^m$  and  $\mathbf{G}(-\bar{\lambda}_l) \in \mathbb{C}^{p \times m}$ . This represents an alternative way to compute the  $\mathcal{H}_2$ -norm besides the approach (3.20).

### $\mathcal{H}_\infty$ -norm

The  $\mathcal{H}_\infty$ -norm of a MIMO linear system  $\Sigma$  is defined as [6]

$$\|\Sigma\|_{\mathcal{H}_\infty} = \sup_{\omega \in \mathbb{R}} \varsigma_{\max}(\mathbf{G}(i\omega)) = \sup_{\omega \in \mathbb{R}} \max_{i=1, \dots, n} \varsigma_i(\mathbf{G}(i\omega)), \quad (3.25)$$

where  $\varsigma_{\max}(\mathbf{G}(i\omega)) = \sqrt{\lambda_{\max}(\mathbf{G}(i\omega) \mathbf{G}(-i\omega)^\top)} = \sqrt{\lambda_{\max}(\mathbf{G}(-i\omega)^\top \mathbf{G}(i\omega))}$  denotes the largest singular value of the transfer matrix. The  $\mathcal{H}_\infty$ -norm corresponds to the peak gain across all SISO input-output channels.

## 3.2 Projective model order reduction

The reduction of dynamical systems is usually performed within a *projective framework*, i.e. it is based on projection (cf. Section 2.2). In the following, we explain the projection-based reduction framework for LTI systems.

The main foundation for model order reduction is to assume that the state trajectory  $\mathbf{x}(t)$  mainly evolves in a  $r$ -dimensional subspace  $\mathcal{V}$  of the state-space  $\mathbb{R}^n$ . Let the full column rank matrix  $\mathbf{V} \in \mathbb{R}^{n \times r}$  be a basis of  $\mathcal{V}$ . Then, the state vector  $\mathbf{x}(t) \in \mathbb{R}^n$  can be expressed in terms of the reduced state vector  $\mathbf{x}_r(t) \in \mathbb{R}^r$  as

$$\mathbf{x}(t) = \mathbf{V} \mathbf{x}_r(t) + \mathbf{e}(t), \quad (3.26)$$

where  $\mathbf{e}(t) \in \mathbb{R}^n$  denotes the approximation error. Inserting the ansatz (3.26) into the state equation (3.1a) yields an overdetermined system ( $n$  equations for  $r$  unknowns in  $\mathbf{x}_r(t)$ )

$$\mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r(t) = \mathbf{A} \mathbf{V} \mathbf{x}_r(t) + \mathbf{B} \mathbf{u}(t) + \boldsymbol{\varepsilon}(t), \quad (3.27)$$

with the residual  $\boldsymbol{\varepsilon}(t) = \mathbf{A} \mathbf{e}(t) - \mathbf{E} \dot{\mathbf{e}}(t) \in \mathbb{R}^n$ . In order to obtain a well-determined (square) reduced-order model, the system (3.27) is *projected* onto the subspace  $\mathcal{U} = \text{ran}(\mathbf{E} \mathbf{V})$ . The projection is performed orthogonally to another subspace  $\mathcal{W} = \text{ran}(\mathbf{W})$ , where the matrix  $\mathbf{W} \in \mathbb{R}^{n \times r}$  has full column rank and is chosen such that  $\mathbf{W}^\top \mathbf{E} \mathbf{V}$  is non-singular. Multiplying (3.27) from the left with the projector  $\mathbf{\Pi} = \mathbf{E} \mathbf{V} (\mathbf{W}^\top \mathbf{E} \mathbf{V})^{-1} \mathbf{W}^\top$  yields

$$\mathbf{\Pi} \underbrace{(\mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r(t) - \mathbf{A} \mathbf{V} \mathbf{x}_r(t) - \mathbf{B} \mathbf{u}(t) - \boldsymbol{\varepsilon}(t))}_{=\boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t))} = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{\Pi} (\boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t)) - \boldsymbol{\varepsilon}(t)) = \mathbf{0}. \quad (3.28)$$

Since all resulting vectors lie now in  $\text{ran}(\mathbf{E} \mathbf{V})$ , the preceding term  $\mathbf{E} \mathbf{V} (\mathbf{W}^\top \mathbf{E} \mathbf{V})^{-1}$  can be omitted in all summands of the equation. By enforcing the so-called *Petrov-Galerkin condition*  $\mathbf{W}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\mathbf{\Pi} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , the residual then vanishes and only the term  $\mathbf{\Pi} \boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t)) = \mathbf{0}$  remains. This finally leads to the reduced-order model

$$\Sigma_r : \begin{cases} \mathbf{E}_r \dot{\mathbf{x}}_r(t) = \mathbf{A}_r \mathbf{x}_r(t) + \mathbf{B}_r \mathbf{u}(t), & \mathbf{x}_r(0) = \mathbf{x}_{r,0}, \\ \mathbf{y}_r(t) = \mathbf{C}_r \mathbf{x}_r(t), \end{cases} \quad (3.29a)$$

$$(3.29b)$$

with reduced matrices  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E} \mathbf{V}$ ,  $\mathbf{A}_r = \mathbf{W}^\top \mathbf{A} \mathbf{V}$ ,  $\mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$ ,  $\mathbf{C}_r = \mathbf{C} \mathbf{V}$  and the initial condition  $\mathbf{x}_r(0) = (\mathbf{W}^\top \mathbf{E} \mathbf{V})^{-1} \mathbf{W}^\top \mathbf{E} \mathbf{x}(0)$ . Similar as for the full-order model (3.1), the abbreviation  $\Sigma_r = (\mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r)$  will denote the reduced linear system (3.29).

In this projective setting, the main task consists in finding suitable reduction bases (also referred to as *projection matrices*)  $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r}$  spanning appropriate subspaces, to ensure a good approximation  $\mathbf{y}(t) \approx \mathbf{y}_r(t)$  and/or preserve structural properties of the FOM.

### Invariance property

Note that only the subspaces  $\mathcal{V} = \text{ran}(\mathbf{V})$  and  $\mathcal{W} = \text{ran}(\mathbf{W})$  are important for model reduction, whereas the specific choice of the bases  $\mathbf{V}, \mathbf{W}$  plays a minor role. In fact, when replacing the projection matrices  $\mathbf{V}, \mathbf{W}$  by  $\tilde{\mathbf{V}} = \mathbf{V}\mathbf{T}_v$  and  $\tilde{\mathbf{W}} = \mathbf{W}\mathbf{T}_w$  for any regular matrices  $\mathbf{T}_v, \mathbf{T}_w \in \mathbb{R}^{r \times r}$ , the realization of the ROM changes, whereas the reduced transfer function remains invariant:  $\tilde{\mathbf{G}}_r(s) = \mathbf{G}_r(s)$ . Nevertheless, specific choices of bases, e.g. real-valued (i.e.  $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r}$ ), orthonormal (i.e.  $\mathbf{V}^\top \mathbf{V} = \mathbf{I}_r, \mathbf{W}^\top \mathbf{W} = \mathbf{I}_r$ ) or biorthonormal (i.e.  $\mathbf{W}^\top \mathbf{V} = \mathbf{I}_r$  or  $\mathbf{W}^\top \mathbf{E}\mathbf{V} = \mathbf{I}_r$ ), are often used due to numerical reasons.

### Galerkin/Petrov-Galerkin projection

Further note that the reduction can be performed by an orthogonal *Galerkin projection* with  $\mathbf{W} = \mathbf{V}$  (or rather  $\text{ran}(\mathbf{W}) = \text{ran}(\mathbf{V})$ ), or by an oblique *Petrov-Galerkin projection* with  $\mathbf{W} \neq \mathbf{V}$  (or rather  $\text{ran}(\mathbf{W}) \neq \text{ran}(\mathbf{V})$ ). The choice of a Galerkin or Petrov-Galerkin projection often depends on the goals and priorities of the reduction. For instance, one can use the degrees of freedom in  $\mathbf{W}$  (or in  $\mathbf{V}$ ) to guarantee stability of the ROM or to preserve other system properties. On the other hand, one can use these degrees of freedom to obtain a better approximation.

### Error system and error system norms

For analysis purposes, it is extremely important to assess the approximation quality of ROMs. One possibility is to measure the accuracy in *time-domain* by means of the output error  $\mathbf{e}_y(t) := \mathbf{y}(t) - \mathbf{y}_r(t)$  or the state error  $\mathbf{e}_x(t) := \mathbf{x}(t) - \mathbf{V}\mathbf{x}_r(t)$  (cf. Section 2.6). Another possibility is to quantify the error in *frequency-domain* via  $\mathbf{G}_e(i\omega) := \mathbf{G}(i\omega) - \mathbf{G}_r(i\omega)$ . This latter approach is specially interesting for LTI systems and is, thus, deepened in the following.

Let us begin with the *error system*  $\Sigma_e := \Sigma - \Sigma_r = (\mathbf{A}_e, \mathbf{B}_e, \mathbf{C}_e, \mathbf{E}_e)$ , where

$$\mathbf{E}_e = \begin{bmatrix} \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_r \end{bmatrix}, \quad \mathbf{A}_e = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_r \end{bmatrix}, \quad \mathbf{B}_e = \begin{bmatrix} \mathbf{B} \\ \mathbf{B}_r \end{bmatrix}, \quad \mathbf{C}_e = \begin{bmatrix} \mathbf{C} & -\mathbf{C}_r \end{bmatrix}. \quad (3.30)$$

The output of the error system is the difference between the output of the FOM and ROM, i.e.  $\mathbf{y}_e(t) = \mathbf{y}(t) - \mathbf{y}_r(t)$ . Moreover, the transfer function of the error system is given by  $\mathbf{G}_e(s) = \mathbf{G}(s) - \mathbf{G}_r(s)$ . Obviously, it is possible to compute the error system for models with a different state dimension, but only feasible for an equal number of inputs  $m$  and outputs  $p$ .

The error between FOM and ROM can be measured *point-wise* in frequency by

$$e_{\text{abs},(*)}(i\omega) := \|\mathbf{G}_e(i\omega)\|_{(*)} = \|\mathbf{G}(i\omega) - \mathbf{G}_r(i\omega)\|_{(*)} \quad (3.31)$$

with a desired matrix norm  $(*) \in \{1, 2, \infty, \text{F}, \dots\}$ . In addition, the error can be measured *norm-wise* in frequency-domain by

$$e_{\text{abs},\mathcal{H}_p} := \|\mathbf{G}_e\|_{\mathcal{H}_p} = \|\mathbf{G} - \mathbf{G}_r\|_{\mathcal{H}_p} \quad (3.32)$$

with a suitable norm  $\mathcal{H}_p \in \{\mathcal{H}_2, \mathcal{H}_\infty, \text{Hankel}, \dots\}$ . The  $\mathcal{H}_2$ - and  $\mathcal{H}_\infty$ -error norms can be calculated by applying the expressions (3.20), (3.24) and (3.25) to the error system  $\Sigma_e$ . For instance, the  $\mathcal{H}_2$ -error norm is given by

$$\|\Sigma_e\|_{\mathcal{H}_2}^2 = \text{tr} \left( C_e P_e C_e^\top \right) = \text{tr} \left( B_e^\top Q_e B_e \right), \quad (3.33)$$

where  $P_e$  and  $Q_e$  are obtained by solving the Lyapunov equations for the error system

$$A_e P_e E_e^\top + E_e P_e A_e^\top + B_e B_e^\top = \mathbf{0}, \quad (3.34a)$$

$$A_e^\top Q_e E_e + E_e^\top Q_e A_e + C_e^\top C_e = \mathbf{0}. \quad (3.34b)$$

It can be shown that the  $\mathcal{H}_2$ - and  $\mathcal{H}_\infty$ -error norms in frequency-domain represent an upper bound for the output error in time-domain, according to [97]

$$\|\mathbf{y} - \mathbf{y}_r\|_{\mathcal{L}_\infty} \leq \|\mathbf{G} - \mathbf{G}_r\|_{\mathcal{H}_2} \|\mathbf{u}\|_{\mathcal{L}_2}, \quad (3.35)$$

$$\|\mathbf{y} - \mathbf{y}_r\|_{\mathcal{L}_2} \leq \|\mathbf{G} - \mathbf{G}_r\|_{\mathcal{H}_\infty} \|\mathbf{u}\|_{\mathcal{L}_2}. \quad (3.36)$$

### 3.3 Overview of linear reduction methods

In this section an overview of established linear reduction approaches is given. The focus lies on the three best known techniques of modal truncation, balanced truncation and moment matching, although further methods are also mentioned at the end of the section. In general, special emphasis is put in the computational aspects, properties, advantages/disadvantages and conceptual similarities/differences of the various reduction techniques.

#### 3.3.1 Modal truncation

*Modal truncation* is one of the oldest techniques for model order reduction. The method was originally developed for second-order systems arising in the context of structural dynamics [212, 125, 113, 53], and then extended to state-space representations [78, 172, 65].

The main idea is to preserve some eigenvalues of the original model in the reduced one. To this end, the projection matrices  $\mathbf{V}$ ,  $\mathbf{W}$  are chosen to span the right and left eigenspaces corresponding to the  $r$  most dominant eigenvalues. This means that  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r]$  and  $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_r]$ , where the right eigenvectors  $\mathbf{v}_i$  and left eigenvectors  $\mathbf{w}_i$ ,  $i = 1, \dots, r$  fulfill the generalized eigenvalue problems

$$(\lambda_i \mathbf{E} - \mathbf{A}) \mathbf{v}_i = \mathbf{0} \quad \iff \quad \mathbf{A} \mathbf{V} = \mathbf{E} \mathbf{V} \Lambda, \quad (3.37a)$$

$$\mathbf{w}_i^\top (\lambda_i \mathbf{E} - \mathbf{A}) = \mathbf{0}^\top \quad \iff \quad \mathbf{W}^\top \mathbf{A} = \Lambda \mathbf{W}^\top \mathbf{E}. \quad (3.37b)$$

Note that, in general, the eigenvalues  $\lambda_i$  and the eigenvectors  $\mathbf{v}_i$  and  $\mathbf{w}_i$  are complex-valued, leading to complex projection matrices  $\mathbf{V}$ ,  $\mathbf{W} \in \mathbb{C}^{n \times r}$ . Nevertheless, if the eigenvalues come in complex conjugate pairs (as it is mostly the case), then a splitting of the eigenvectors in real and imaginary part is possible, in order to obtain real projection matrices  $\tilde{\mathbf{V}}$ ,  $\tilde{\mathbf{W}} \in \mathbb{R}^{n \times r}$  spanning the same subspaces as their complex counterparts. Further note that a QR- or a Gram-Schmidt-type (bi)orthogonalization can be employed to obtain (bi)orthonormal bases.

For large-scale models, the computation of the entire eigendecomposition is unfeasible. Hence, only a few eigenvalues and eigenvectors are computed *iteratively* using *power methods* (e.g. inverse iteration [79, 233]) or *Krylov subspaces* (e.g. implicitly restarted Arnoldi [168]). These algorithms (e.g. MATLAB or SciPy function `eigs`) usually employ different selection criteria to find the eigenvalues. For instance, they can return the eigenvalues with the largest or smallest magnitude ('`lm`', '`sm`'), largest real part ('`lr`') or closest to a given complex shift `sigma`. This allows to retain the desired eigenvalues in the reduced-order model.

Especially in the context of approximating the input-output behavior (3.7), the modal approach can be combined with an *eigenvalue dominance analysis*. The basic idea is to take the aspects of controllability and observability into account, wherefore several dominance measures have been proposed [162, 180, 263, 215]. In this way, the information contained in the input and output matrices  $\hat{\mathbf{B}}, \hat{\mathbf{C}}$  is also exploited to select the eigenvalues. Note again that modal truncation *exactly* preserves some eigenvalues of the FOM. This is particularly advantageous in structural mechanics and acoustics, where the eigendynamics  $(\mathbf{A}, \mathbf{E})$  and the physical interpretation of the reduced state vector play a crucial role. Nevertheless, the restriction to the set of modal coordinates can sometimes be a limitation, especially if other reduced coordinates are better suited to approximate the transfer behavior.

### 3.3.2 Balanced truncation

One of the most prominent system-theoretic methods for model order reduction is given by the so-called *balanced truncation* or *truncated balanced realization*. The idea first originated in the context of digital filter design [183], and was later formalized by Moore in [181].

The method is based on retaining the state variables with highest energy transfer, while truncating those that are less important for the input-output behavior. Hereby, the concepts of controllability and observability are exploited to rank the state variables in terms of their energy contribution. In fact, a final state  $\mathbf{x}_e$  is called *weak controllable* if it requires a lot of input energy  $\mathbf{u}(t)$  to be reached. This energy is measured in terms of the controllability Gramian via  $\mathcal{J}_c(\mathbf{x}_e) = \mathbf{x}_e^\top \mathbf{P}^{-1} \mathbf{x}_e$ . Similarly, an initial state  $\mathbf{x}_0$  is *weak observable* if it provides little energy at the output  $\mathbf{y}(t)$  for  $\mathbf{u}(t) = \mathbf{0}$ . This energy is measured in terms of the observability Gramian by  $\mathcal{J}_o(\mathbf{x}_0) = \mathbf{x}_0^\top \tilde{\mathbf{Q}} \mathbf{x}_0$ . Thus, such weak controllable and observable states do not contribute much to the transfer behavior and can therefore be neglected.

Balanced truncation is composed of two steps. First, the system is transformed into a *balanced realization*  $\Sigma_{\text{bal}} = (\mathbf{T}\mathbf{A}\mathbf{T}^{-1}, \mathbf{T}\mathbf{B}, \mathbf{C}\mathbf{T}^{-1}, \mathbf{T}\mathbf{E}\mathbf{T}^{-1})$ , where each state variable is equally controllable and observable. Mathematically, this means that the controllability Gramian  $\mathbf{P}$  and the observability Gramian  $\tilde{\mathbf{Q}} = \mathbf{E}^\top \mathbf{Q} \mathbf{E}$  are equal and diagonal

$$\mathbf{P} = \tilde{\mathbf{Q}} = \Sigma = \text{diag}(\varsigma_1, \dots, \varsigma_n), \quad \varsigma_1 \geq \varsigma_2 \geq \dots \geq \varsigma_n \geq 0, \quad (3.38)$$

where  $\varsigma_i := \sqrt{\lambda_i(\mathbf{P}\mathbf{E}^\top \mathbf{Q} \mathbf{E})}$  are the so-called *Hankel singular values*. Then, the reduction is performed by *truncating* the state variables associated to the smallest singular values  $\varsigma_j \ll \varsigma_i$ ,  $i = 1, \dots, r$ ,  $j = r + 1, \dots, n$ .

The projection matrices needed for reduction can be computed via, e.g., the *Square-Root Balanced Truncation* (SR-BT) algorithm [101, 157]. The main steps of this procedure are summarized in Algorithm 3.1. From a numerical point of view, the solution of both Lyapunov equations (3.18) is the most expensive, but also the most crucial step of balanced truncation. The numerical solution of large sparse Lyapunov equations is discussed in detail in Section 2.4.

---

**Algorithm 3.1** Square-Root Balanced Truncation (SR-BT)
 

---

1. Solution of Lyapunov equations and Cholesky decomposition

$$\mathbf{P} = \mathbf{S}\mathbf{S}^\top \quad \text{and} \quad \mathbf{Q} = \mathbf{R}\mathbf{R}^\top, \quad (3.39)$$

where  $\mathbf{P}, \mathbf{Q}$  are symmetric positive definite, and  $\mathbf{S}, \mathbf{R}$  are lower triangular.

2. Singular value decomposition (SVD) of  $\mathbf{R}^\top \mathbf{E} \mathbf{S} = \mathbf{M} \boldsymbol{\Sigma} \mathbf{N}^\top$  and partition

$$\mathbf{R}^\top \mathbf{E} \mathbf{S} = \begin{bmatrix} \mathbf{M}_1 & \mathbf{M}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \end{bmatrix} \begin{bmatrix} \mathbf{N}_1^\top \\ \mathbf{N}_2^\top \end{bmatrix}, \quad (3.40)$$

where  $\mathbf{M}_1 \in \mathbb{R}^{n \times r}$ ,  $\mathbf{N}_1 \in \mathbb{R}^{n \times r}$  are orthogonal, and  $\boldsymbol{\Sigma}_1 = \text{diag}(\varsigma_1, \dots, \varsigma_r) \in \mathbb{R}^{r \times r}$ .

3. Computation of the transformation/projection matrices  $\mathbf{T}_1 := \mathbf{W}^\top$  and  $\mathbf{T}_1^{-1} := \mathbf{V}$

$$\mathbf{W}^\top = \boldsymbol{\Sigma}_1^{-1/2} \mathbf{M}_1^\top \mathbf{R}^\top \in \mathbb{R}^{r \times n} \quad \text{and} \quad \mathbf{V} = \mathbf{S} \mathbf{N}_1 \boldsymbol{\Sigma}_1^{-1/2} \in \mathbb{R}^{n \times r}, \quad (3.41)$$

which are inherently biorthogonal w.r.t.  $\mathbf{E}$ , i.e.  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E} \mathbf{V} = \mathbf{I}_r$  and  $\boldsymbol{\Pi} = \mathbf{E} \mathbf{V} \mathbf{W}^\top$ .

---

In the large-scale setting, the direct Hammarling's method cannot be used anymore due to its high storage requirement. Instead, iterative techniques like RKSM and LR-ADI are employed to find *low-rank* approximations of the solutions  $\mathbf{P} \approx \hat{\mathbf{P}} = \mathbf{Z}_c \mathbf{Z}_c^\top$  and  $\mathbf{Q} \approx \mathbf{Z}_o \mathbf{Z}_o^\top$ . By simply replacing the Cholesky factors  $\mathbf{S}$  and  $\mathbf{R}$  in Algorithm 3.1 by the low-rank factors  $\mathbf{Z}_c$  and  $\mathbf{Z}_o$ , the so-called *Low-Rank Square-Root Balanced Truncation* (LR-SRBT) algorithm is obtained.

Balanced truncation has numerous advantages. It preserves asymptotic stability in the reduced-order model, which is also balanced and a minimal realization. Furthermore, a rigorous, a-priori error bound is available [101]:

$$\|\mathbf{G} - \mathbf{G}_r\|_{\mathcal{H}_\infty} \leq 2 \sum_{i=r+1}^n \varsigma_i. \quad (3.42)$$

This allows to estimate the error and to select the reduced order  $r$  according to the desired approximation quality. In addition, balanced truncation is suitable for *automated* or *black-box* model reduction, since the reduced order  $r$  is the only parameter that has to be chosen. Unfortunately, some of these advantages get lost when the Lyapunov equations are solved approximately. For instance, the  $\mathcal{H}_\infty$ -error bound (3.42) does not necessarily hold anymore. Additionally, the iterative Krylov-based techniques RKSM and LR-ADI also require the selection of appropriate shift parameters to compute the low-rank factors. Despite these inconveniences for the application of the method to large-scale models, balanced truncation is still one of the most popular model reduction techniques.

Finally, note that there exist several generalizations and modifications of the method. Examples include balancing for differential-algebraic equations [259, 184, 48], nonlinear [239, 26, 34], time-varying [258, 253, 167], parametric [254] and second-order systems [185, 62, 220, 38], as well as time- and frequency-limited balanced truncation [40, 152].

### 3.3.3 Moment matching

Another approach for model reduction is given by *moment matching* or *rational interpolation*. The main idea is to construct a reduced-order model that interpolates the original transfer function and its derivatives at selected complex shifts  $\sigma_i \in \mathbb{C}$ :

$$\mathbf{G}^{(\ell)}(\sigma_i) = \mathbf{G}_r^{(\ell)}(\sigma_i), \quad i = 1, 2, \dots, \quad \ell = 0, 1, \dots, \quad (3.43)$$

where  $\mathbf{G}^{(\ell)}(s) = \frac{d^\ell \mathbf{G}(s)}{ds^\ell}$  denotes the  $\ell$ -th derivative of  $\mathbf{G}(s)$  with respect to  $s$ . Note that the derivatives  $\mathbf{G}^{(\ell)}(\sigma_i)$  are equal (except for the factor  $\frac{1}{\ell!}$ ) to the so-called *moments*  $\mathbf{m}_\ell(\sigma_i)$ , i.e. the coefficients of the Taylor series expansion of  $\mathbf{G}(s)$  around  $\sigma_i$  (cf. Eq. (3.46)).

The idea of reducing an LTI system by approximating its rational transfer function has been published in the literature under several names: *continued fraction expansion*, *method of moments* [100, 284], (multipoint) *Padé approximation* [242] and *asymptotic waveform evaluation* (AWE) [207], just to mention a few. These methods, however, construct the reduced transfer function  $\mathbf{G}_r(s)$  based on the explicit computation of moments, which is numerically unstable and ill-conditioned. Fortunately, moment matching can also be enforced *implicitly* using projection onto (classical) Krylov subspaces [265]. Later, Lanczos- [89, 98] and Arnoldi-type [106] algorithms were applied to compute bases  $\mathbf{V}, \mathbf{W}$  of *rational* Krylov subspaces in a numerically stable way. The general framework for projective model reduction by Krylov subspaces is described thoroughly in Grimme's PhD thesis [106] as well as in the surveys [18, 94]. These projection-based reduction methods are referred in the literature as *Padé via Lanczos* (PVL), *rational Krylov* (RK), *implicit moment matching* and *rational interpolation*.

For MIMO systems, the moment matching condition (3.43) means that the whole block,  $p \times m$  transfer function matrix (and its derivatives) is interpolated at selected shifts. In many applications, however, it might be desirable to favor relevant input-output channels or customize the reduction according to the frequency ranges of certain SISO paths. To this end, [115] proposed model reduction via *tangential interpolation*

$$\mathbf{G}^{(\ell)}(\sigma_i) \mathbf{r}_i = \mathbf{G}_r^{(\ell)}(\sigma_i) \mathbf{r}_i, \quad i = 1, 2, \dots, \quad \ell = 0, 1, \dots, \quad (3.44a)$$

$$\mathbf{l}_i^\top \mathbf{G}^{(\ell)}(\sigma_i) = \mathbf{l}_i^\top \mathbf{G}_r^{(\ell)}(\sigma_i), \quad i = 1, 2, \dots, \quad \ell = 0, 1, \dots, \quad (3.44b)$$

along selected left and right tangential directions  $\mathbf{l}_i \in \mathbb{C}^p$ ,  $\mathbf{r}_i \in \mathbb{C}^m$ . This kind of interpolation is achieved by constructing the projection matrices  $\mathbf{V}, \mathbf{W}$  as bases of *tangential* Krylov subspaces (cf. Section 2.3). Remember that the use of tangential directions also avoids the increased dimension of the reduction bases from the block case.

Model reduction via Krylov subspaces has both advantages and disadvantages. In the following, we briefly discuss and analyze them.

**Computational effort** The main advantage of Krylov subspace methods is their low numerical effort, since the computation of the reduction bases mainly relies on the solution of sparse linear systems of equations (LSE), which can be solved efficiently using either direct or iterative methods (see Section 2.5). Therefore, Krylov-based reduction is indeed applicable to very large-scale models ( $n \geq 10^6$ ).

**Selection of reduction parameters** As pointed out by Antoulas [6], the choice of the reduction parameters is an advantage, but at the same time a fundamental issue of Krylov methods.

If some knowledge about the system at hand is available, then practitioners can exploit it to customize the reduction by choosing appropriate shifts and tangential directions. On the contrary, if system understanding is missing or the frequency range of interest is not known a-priori, then the selection of reduction parameters for a satisfactory approximation is not a trivial task. Nevertheless, some simple strategies [265, 89, 106] as well as more sophisticated, adaptive techniques [97, 39, 84] exist to select good or even optimal interpolation data.

**Stability preservation** Unfortunately, in contrast to balanced truncation, Krylov subspace methods are not necessarily stability preserving. This means that the ROM may not be asymptotically stable, even if the FOM is stable. Implicitly restarted Lanczos/Arnoldi methods [108, 134], as well as heuristic trials (e.g. changing the interpolation data or the reduced order), are possible remedies to eliminate this unwanted behavior. However, there also exist some systematic techniques and algorithms that deliberately select the reduction degrees of freedom to guarantee stability.

1. The reduction of a *strictly dissipative* system (i.e.  $\mathbf{E} = \mathbf{E}^\top \succ \mathbf{0}$ ,  $\mathbf{A} + \mathbf{A}^\top \prec \mathbf{0}$ ) via orthogonal projection (i.e.  $\text{ran}(\mathbf{W}) = \text{ran}(\mathbf{V})$ ) leads to a strictly dissipative, and consequently asymptotically stable, ROM [244]. The concept of dissipativity (aka. contractivity) plays also a key role for the passivity and structure-preserving reduction of port-Hamiltonian [93, 194] and second-order systems [236].
2. If the FOM is stable, but not necessarily strictly dissipative, then it can be transformed into a strictly dissipative realization by solving a large-scale Lyapunov equation (using e.g. RKSM and LR-ADI). For instance, if  $\mathbf{Q}$  is the solution of the observability Lyapunov equation (3.18b) and  $\mathbf{V}$  spans an input Krylov subspace, then the choice  $\mathbf{W} = \mathbf{QV}(\mathbf{V}^\top \mathbf{QV})^{-1}$  — or less general  $\mathbf{W} = \mathbf{QV}$  — yields a stable ROM (see [95], [283, Ch. 5], [249]). Note that this orthogonal projection-like choice of  $\mathbf{W}$  is also employed in the passivity and structure-preserving model reduction of linear port-Hamiltonian systems [272, 111].
3. Related to the approach before, the Iterative SVD-Rational Krylov (ISRK) algorithm [112] also combines the solution of one large-scale Lyapunov equation with *iterative* rational Krylov steps using  $\sigma_i \leftarrow -\bar{\lambda}_{r,i}$  for the shifts. Upon convergence, ISRK leads to an asymptotically stable  $\mathcal{H}_2$ -pseudo-optimal ROM, whose eigenvalues are the mirrored images of the converged shifts. Note that the ISRK procedure is strongly related to the Iterative Rational Krylov Algorithm for port-Hamiltonian systems (IRKA-PH) [111].
4. An  $\mathcal{H}_2$ -pseudo-optimal ROM that has eigenvalues as mirror images of *user-selected* shifts can be obtained by the *iteration-free* Pseudo-Optimal Rational Krylov (PORK) algorithm [277, 273]. This approach is exploited within the Stability Preserving Adaptive Rational Krylov (SPARK) scheme [197, 198] to guarantee stability.

**Error bounds** Another drawback of Krylov-based model reduction is that – in contrast to balanced truncation – there are no general, rigorous, a-priori, efficiently computable, global error bounds for the approximation quality of the ROM. Nevertheless, there exist error estimates in the literature that at least meet some of the just mentioned requirements. For instance, Bai et al. [17] and Feng, Antoulas and Benner [86] have proposed local (i.e. point-wise in frequency) a-posteriori error bounds, which estimate the accuracy of the ROM within a certain frequency range. These approaches, however, usually require a large and dense set of samples to guarantee rigorously and sharpness of the error bound. On the other hand,



global (i.e. norm-wise) error bounds are also available in the literature, but mostly only for special LTI systems. For example,  $\mathcal{H}_2$  and  $\mathcal{H}_\infty$  upper bounds on the error (3.32) are given in [276] for general systems, and made computationally more efficient in [198, Ch. 5] for strictly dissipative systems. Moreover, there exist error bounds for lossless and passive systems.

### 3.3.4 Further approaches

In addition to the three best known linear techniques explained above, many other reduction methods exist (see e.g. [10, 6, 24]). Some of them will be briefly mentioned in the following.

The method of *Proper Orthogonal Decomposition* (POD) [181, 37] is based on the SVD of a so-called snapshot matrix, which is composed of samples of the FOM state trajectory. This data-driven technique is mostly employed in the context of nonlinear model order reduction, since the simulation-based collection of snapshots is a straightforward remedy to the (general) absence of analytical solutions or system-theoretic concepts for nonlinear dynamical systems.

The *optimal Hankel norm approximation* [101] aims at the construction of a reduced-order model which is optimal in terms of the Hankel norm (i.e. the 2-induced norm of the Hankel operator). Similar to balanced truncation, this approach is stability preserving, provides an a-priori  $\mathcal{H}_\infty$ -error bound and is computationally expensive for large-scale systems.

The *Loewner framework* [171] is a data-driven approach, which constructs a ROM that achieves moment matching using frequency-domain measurements. In that sense, the method aims at the low-order identification of an unknown system from input-output data rather than at the reduction of a known system. The Loewner matrices satisfy certain Sylvester equations, which emphasizes the connection of this technique to Krylov subspaces and rational interpolation (cf. Section 3.5). Another related data-driven approach is given by *vector fitting* [107], which permits to calculate a rational approximation from measured or computed frequency response functions. The recent *AAA algorithm* [192] also falls into this category.

*SVD-Krylov approaches* [10, 95, 112] aim at combining the stability and global error bound features of SVD-based methods, with the numerical efficiency and applicability of Krylov-based techniques. The ISRK algorithm is one popular representative of this type of techniques.

Finally,  $\mathcal{H}_2$ - and  $\mathcal{H}_\infty$ -optimal model reduction techniques aim at finding a ROM that minimizes the approximation error in terms of the  $\mathcal{H}_2$ - or  $\mathcal{H}_\infty$ -norm. The problem of  $\mathcal{H}_2$ -optimal model reduction will be addressed in Section 3.7.

## 3.4 Frequency-domain interpretation of linear moment matching

Throughout this thesis, we focus on the concept of implicit moment matching by rational Krylov subspaces as model reduction technique. In this section we will first review the classical, frequency-domain notion of moments and then recall the Krylov subspaces to achieve implicit moment matching. After that, we will also discuss different important aspects and possibilities that Krylov subspace-based model reduction opens.

### 3.4.1 Frequency-domain notion of linear moments

**Definition 3.4** (Frequency-domain linear moments). The Taylor series expansion of the transfer function  $\mathbf{G}(s)$  at a complex number  $\sigma \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$ , also called *shift* or *expansion/interpolation point*, is given by

$$\mathbf{G}(s) = \sum_{\ell=0}^{\infty} \mathbf{m}_{\ell}(\sigma)(s - \sigma)^{\ell}, \quad (3.45)$$

where  $\mathbf{m}_{\ell}(\sigma)$  is called the  $\ell$ -th *moment* of  $\mathbf{G}(s)$  at  $\sigma$ . The moments represent the Taylor series coefficients and satisfy:

$$\begin{aligned} \mathbf{m}_{\ell}(\sigma) &= \frac{1}{\ell!} \left. \frac{d^{\ell} \mathbf{G}(s)}{ds^{\ell}} \right|_{s=\sigma} = \frac{1}{\ell!} \left[ \frac{d^{\ell}}{ds^{\ell}} \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \right] \Big|_{s=\sigma} \\ &= (-1)^{\ell} \mathbf{C} \left( (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^{\ell} (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \in \mathbb{C}^{p \times m}. \end{aligned} \quad (3.46)$$

If the transfer function  $\mathbf{G}(s)$  is expanded at  $\sigma \rightarrow \infty$ , then the Taylor series is given by

$$\mathbf{G}(s) = \sum_{\ell=0}^{\infty} \mathbf{m}_{\ell, \infty} \frac{1}{s^{\ell+1}} \quad \bullet \text{---} \circ \quad \mathbf{g}(t) = \sum_{\ell=0}^{\infty} \mathbf{m}_{\ell, \infty} \frac{t^{\ell}}{\ell!}, \quad (3.47)$$

where  $\mathbf{m}_{\ell, \infty}$  is called the  $\ell$ -th *Markov parameter* of  $\mathbf{G}(s)$ . The Markov parameters represent the Taylor series coefficients of the impulse response  $\mathbf{g}(t)$  at  $t=0$  and satisfy:

$$\mathbf{m}_{\ell, \infty} = \left. \frac{d^{\ell} \mathbf{g}(t)}{dt^{\ell}} \right|_{t=0} = \left[ \frac{d^{\ell}}{dt^{\ell}} \mathbf{C} e^{\mathbf{E}^{-1} \mathbf{A} t} \mathbf{E}^{-1} \mathbf{B} \right] \Big|_{t=0} = \mathbf{C} \left( \mathbf{E}^{-1} \mathbf{A} \right)^{\ell} \mathbf{E}^{-1} \mathbf{B} \in \mathbb{R}^{p \times m}. \quad (3.48)$$

▲

### 3.4.2 Moment matching by Krylov subspaces

In order to achieve implicit moment matching, the projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  should be chosen as bases of *rational Krylov subspaces* (cf. Section 2.3). The main theorems of Krylov-based model reduction are stated in the following based on [265, 106, 115, 33].

**Theorem 3.1** (Block multimoment matching). *Let the expansion points  $\sigma, \mu \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  be given. Consider a ROM as in (3.29) obtained through projection with  $\mathbf{V}, \mathbf{W}$ . If  $\mathbf{V}$  and  $\mathbf{W}$  are chosen as bases of respective  $q$ -order block input and output Krylov subspaces*

$$\mathcal{K}_q \left( (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \right) \supseteq \text{ran}(\mathbf{V}), \quad (3.49a)$$

$$\mathcal{K}_q \left( (\mu\mathbf{E} - \mathbf{A})^{-\top} \mathbf{E}^{\top}, (\mu\mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^{\top} \right) \supseteq \text{ran}(\mathbf{W}), \quad (3.49b)$$

then the ROM fulfills the following block multimoment matching conditions:

$$\mathbf{G}^{(\ell)}(\sigma) = \mathbf{G}_{\mathbf{r}}^{(\ell)}(\sigma) \quad \Leftrightarrow \quad \mathbf{m}_{\ell}(\sigma) = \mathbf{m}_{\mathbf{r}, \ell}(\sigma), \quad \ell = 0, \dots, q-1, \quad (3.50a)$$

$$\mathbf{G}^{(\ell)}(\mu) = \mathbf{G}_{\mathbf{r}}^{(\ell)}(\mu) \quad \Leftrightarrow \quad \mathbf{m}_{\ell}(\mu) = \mathbf{m}_{\mathbf{r}, \ell}(\mu), \quad \ell = 0, \dots, q-1. \quad (3.50b)$$

Thus, the ROM interpolates the original transfer function matrix and its derivatives up to order  $q-1$  at the shifts  $\sigma$  and  $\mu$ , respectively. If both (3.49a) and (3.49b) are chosen with  $\sigma = \mu$ , then the following interpolation conditions are achieved:  $\mathbf{G}^{(\ell)}(\sigma) = \mathbf{G}_r^{(\ell)}(\sigma)$ ,  $\ell = 0, \dots, 2q-1$ .

**Remark 3.1** (Matching Markov parameters). If  $\mathbf{V}$  or  $\mathbf{W}$  is chosen as basis of

$$\mathcal{K}_q(\mathbf{E}^{-1}\mathbf{A}, \mathbf{E}^{-1}\mathbf{B}) \supseteq \text{ran}(\mathbf{V}), \quad (3.51a)$$

$$\mathcal{K}_q(\mathbf{E}^{-\top}\mathbf{A}^\top, \mathbf{E}^{-\top}\mathbf{C}^\top) \supseteq \text{ran}(\mathbf{W}), \quad (3.51b)$$

then the ROM fulfills  $\mathbf{m}_{\ell,\infty} = \mathbf{m}_{r,\ell,\infty}$ ,  $\ell = 0, \dots, q-1$ . If both (3.51a) and (3.51b) are chosen, then  $\mathbf{m}_{\ell,\infty} = \mathbf{m}_{r,\ell,\infty}$ ,  $\ell = 0, \dots, 2q-1$ .  $\triangle$

Note that, in addition to the afore explained *multimoment* matching strategy, it is also possible to match (high-order) moments at a set of different shifts  $\{\sigma_i\}_{i=1}^r$  and  $\{\mu_i\}_{i=1}^r$  with associated multiplicities  $\{q_i\}_{i=1}^r$ :

$$\mathbf{G}^{(\ell_i)}(\sigma_i) = \mathbf{G}_r^{(\ell_i)}(\sigma_i) \Leftrightarrow \mathbf{m}_{\ell_i}(\sigma_i) = \mathbf{m}_{r,\ell_i}(\sigma_i), \quad \ell_i = 0, \dots, q_i-1, \quad i = 1, \dots, r, \quad (3.52a)$$

$$\mathbf{G}^{(\ell_i)}(\mu_i) = \mathbf{G}_r^{(\ell_i)}(\mu_i) \Leftrightarrow \mathbf{m}_{\ell_i}(\mu_i) = \mathbf{m}_{r,\ell_i}(\mu_i), \quad \ell_i = 0, \dots, q_i-1, \quad i = 1, \dots, r. \quad (3.52b)$$

In this setting, known as *multipoint* moment matching, each subspace  $\text{ran}(\mathbf{V})$  and  $\text{ran}(\mathbf{W})$  is given by the union of all respective rational Krylov subspaces  $\mathcal{K}_{q_i}$  (cf. Eq. (2.31)). Also note that, besides *block* Krylov subspaces, in the MIMO case we alternatively may use *tangential* Krylov subspaces. In the following, we exemplarily show the tangential multipoint case for single multiplicities of the shifts, i.e.  $q_1 = \dots = q_r = 1$ .

**Theorem 3.2** (Tangential multipoint moment matching). *Let  $\{\sigma_i\}_{i=1}^r$  and  $\{\mu_i\}_{i=1}^r$  with  $\sigma_i, \mu_i \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  be different sets of shifts. Let  $\{\mathbf{r}_i\}_{i=1}^r \in \mathbb{C}^m$  and  $\{\mathbf{l}_i\}_{i=1}^r \in \mathbb{C}^p$  be right and left tangential directions, respectively. Consider a ROM as in (3.29) obtained through projection with  $\mathbf{V}, \mathbf{W}$ . If  $\mathbf{V}$  and  $\mathbf{W}$  are chosen as bases of respective 1-order tangential input and output Krylov subspaces*

$$\text{span} \left\{ (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_1, \dots, (\sigma_r \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_r \right\} \supseteq \text{ran}(\mathbf{V}), \quad (3.53a)$$

$$\text{span} \left\{ (\mu_1 \mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^\top \mathbf{l}_1, \dots, (\mu_r \mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^\top \mathbf{l}_r \right\} \supseteq \text{ran}(\mathbf{W}), \quad (3.53b)$$

then the ROM fulfills the following tangential multipoint moment matching conditions:

$$\mathbf{G}(\sigma_i) \mathbf{r}_i = \mathbf{G}_r(\sigma_i) \mathbf{r}_i \quad \Leftrightarrow \quad \mathbf{m}_0(\sigma_i) \mathbf{r}_i = \mathbf{m}_{r,0}(\sigma_i) \mathbf{r}_i, \quad i = 1, \dots, r, \quad (3.54a)$$

$$\mathbf{l}_i^\top \mathbf{G}(\mu_i) = \mathbf{l}_i^\top \mathbf{G}_r(\mu_i) \quad \Leftrightarrow \quad \mathbf{l}_i^\top \mathbf{m}_0(\mu_i) = \mathbf{l}_i^\top \mathbf{m}_{r,0}(\mu_i), \quad i = 1, \dots, r. \quad (3.54b)$$

Thus, the ROM tangentially interpolates the original transfer function matrix at  $\sigma_i$  and  $\mu_i$ , respectively. If both (3.53a) and (3.53b) are chosen with  $\sigma_i = \mu_i$ , then right, left and bitangential Hermite interpolation are achieved:

$$\mathbf{G}(\sigma_i) \mathbf{r}_i = \mathbf{G}_r(\sigma_i) \mathbf{r}_i \quad \Leftrightarrow \quad \mathbf{m}_0(\sigma_i) \mathbf{r}_i = \mathbf{m}_{r,0}(\sigma_i) \mathbf{r}_i, \quad i = 1, \dots, r, \quad (3.55a)$$

$$\mathbf{l}_i^\top \mathbf{G}(\sigma_i) = \mathbf{l}_i^\top \mathbf{G}_r(\sigma_i) \quad \Leftrightarrow \quad \mathbf{l}_i^\top \mathbf{m}_0(\sigma_i) = \mathbf{l}_i^\top \mathbf{m}_{r,0}(\sigma_i), \quad i = 1, \dots, r, \quad (3.55b)$$

$$\mathbf{l}_i^\top \mathbf{G}'(\sigma_i) \mathbf{r}_i = \mathbf{l}_i^\top \mathbf{G}'_r(\sigma_i) \mathbf{r}_i \quad \Leftrightarrow \quad \mathbf{l}_i^\top \mathbf{m}_1(\sigma_i) \mathbf{r}_i = \mathbf{l}_i^\top \mathbf{m}_{r,1}(\sigma_i) \mathbf{r}_i, \quad i = 1, \dots, r. \quad (3.55c)$$

### 3.4.3 Discussion of different aspects

In this section, different important aspects of Krylov-based model reduction are analyzed. The aim is to discuss the various possibilities and trade-offs that Krylov subspace methods offer, as well as to give some valuable remarks concerning the judicious implementation.

**Multimoment vs. Multipoint** As already stated in Theorems 3.1 and 3.2, in Krylov-based reduction one distinguishes between matching high-order moments at a single shift (aka. multimoment), and matching (rather low-order) moments at several shifts (aka. multipoint). While the multimoment strategy allows to achieve a better *local* approximation at a specific shift, the multipoint strategy rather permits to gain a *global* approximation over a broad frequency range. In terms of computational effort, the multipoint strategy is generally more expensive, since for every new expansion point an LU-decomposition of  $\sigma_i \mathbf{E} - \mathbf{A}$  is required. Note again that the two concepts can be combined (multimoment + multipoint), in order to customize the reduction according to the desired accuracy (cf. Eq (3.52)). In fact, the combination of both strategies – together with an application-oriented selection of interpolation data and multiplicities – can be the best choice in terms of approximation quality and numerical efficiency.

**MIMO case** For MIMO systems one can decide to employ either block or tangential Krylov subspaces. While the block interpolation conditions guarantee a full matrix matching, the tangential approach delivers a linearly combined interpolation along selected input and output directions. Thus, tangential directions allow to weight certain SISO paths, while preventing the reduction bases from growing by  $m$  or  $p$  columns per iteration. Another way to overcome this growth is to deflate the block reduction bases, in order to eliminate linearly dependent columns. This procedure, however, might slightly “corrupt” the moment matching conditions, specially if the deflated subspace changes significantly w.r.t. the original one.

**One-sided vs. Two-sided reduction** Krylov-based reduction can be performed by means of a *one-sided* or *two-sided* method. A one-sided reduction is carried out by a Galerkin projection using only one (input *or* output) Krylov subspace. On the contrary, a two-sided reduction is accomplished by a Petrov-Galerkin projection using both input *and* output Krylov subspaces. While a two-sided approach is computationally more expensive, it delivers a better approximation due to the double number of matching moments. In a one-sided approach, however, this better accuracy is often sacrificed in favor of preserving stability or other system properties in the ROM. According to the author’s experience, the selection of an input (**V**-sided) or an output (**W**-sided) Krylov subspace for a Galerkin projection usually depends on the system at hand. If the input matrix  $\mathbf{B}$  is more likely to contain useful information about the system, then a **V**-sided Galerkin reduction is recommended. If, on the other hand, the sensors placement information seems to be crucial, then a one-sided reduction with an output Krylov subspace is preferable [Hei18].

**Complex shifts and real bases** Complex expansion points are fundamental for certain applications, e.g. for vibratory systems. However, using such type of shifts leads to complex-valued reduction bases  $\mathbf{V}, \mathbf{W} \in \mathbb{C}^{n \times r}$ , which result in complex reduced-order matrices. To overcome this drawback, complex conjugate pairs of shifts (and tangential directions) should be em-

ployed, as this allows to split the basis vectors in real and imaginary part to obtain real-valued counterparts  $\tilde{\mathbf{V}}, \tilde{\mathbf{W}} \in \mathbb{R}^{n \times r}$ . For clarification purposes we consider an illustrative example.

*Example 3.1* (Making the basis real). Assume that an arbitrary complex conjugate pair of shifts  $\sigma_{1,2} = \delta \mp i\omega \in \mathbb{C}$  with  $\sigma_1 = \overline{\sigma_2}$ , and a complex conjugate pair of tangential directions  $\mathbf{r}_{1,2} = \mathbf{r}_{\text{real}} \mp i\mathbf{r}_{\text{imag}} \in \mathbb{C}^m$  with  $\mathbf{r}_1 = \overline{\mathbf{r}_2}$  are given. Then, the corresponding Krylov directions are also complex conjugated to each other (provided that  $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{E}$  are real matrices):

$$\mathbf{v}_{1,2} = (\sigma_{1,2}\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}\mathbf{r}_{1,2} = \mathbf{v}_{\text{real}} \mp i\mathbf{v}_{\text{imag}} \in \mathbb{C}^n, \quad \text{with } \mathbf{v}_1 = \overline{\mathbf{v}_2}. \quad (3.56)$$

This property allows to construct real-valued directions  $\tilde{\mathbf{v}}_1$  and  $\tilde{\mathbf{v}}_2$  as linear combinations of the complex-valued directions, such that:

$$\tilde{\mathbf{v}}_1 = \frac{\mathbf{v}_1 + \mathbf{v}_2}{2} = \frac{\mathbf{v}_{\text{real}} - i\mathbf{v}_{\text{imag}} + \mathbf{v}_{\text{real}} + i\mathbf{v}_{\text{imag}}}{2} = \mathbf{v}_{\text{real}}, \quad (3.57)$$

$$\tilde{\mathbf{v}}_2 = i\frac{\mathbf{v}_1 - \mathbf{v}_2}{2} = i\frac{\mathbf{v}_{\text{real}} - i\mathbf{v}_{\text{imag}} - \mathbf{v}_{\text{real}} - i\mathbf{v}_{\text{imag}}}{2} = \mathbf{v}_{\text{imag}}. \quad (3.58)$$

Due to the linear combinations, the complex-valued basis  $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2]$  and the real-valued basis  $\tilde{\mathbf{V}} = [\mathbf{v}_{\text{real}}, \mathbf{v}_{\text{imag}}]$  span the same subspace, i.e.  $\text{ran}(\mathbf{V}) = \text{ran}(\tilde{\mathbf{V}})$ . According to this invariance property, the real basis can be equivalently obtained by  $\tilde{\mathbf{V}} = \mathbf{V}\mathbf{T}_v$  using the regular transformation matrix

$$\mathbf{T}_v = \begin{bmatrix} \frac{1}{2} & \frac{1}{2}i \\ \frac{1}{2} & -\frac{1}{2}i \end{bmatrix}, \quad \text{with} \quad \mathbf{T}_v^{-1} = \begin{bmatrix} 1 & 1 \\ -i & i \end{bmatrix}. \quad (3.59)$$

Note that the complex-valued shift and right tangential directions matrices

$$\mathbf{S}_v = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}, \quad \mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2]. \quad (3.60)$$

also transform to real-valued matrices

$$\tilde{\mathbf{S}}_v = \mathbf{T}_v^{-1}\mathbf{S}_v\mathbf{T}_v = \begin{bmatrix} \delta & \omega \\ -\omega & \delta \end{bmatrix}, \quad \tilde{\mathbf{R}} = \mathbf{R}\mathbf{T}_v = [\mathbf{r}_{\text{real}}, \mathbf{r}_{\text{imag}}]. \quad (3.61)$$

Analogously, for the output Krylov case it holds  $\tilde{\mathbf{W}} = \mathbf{W}\mathbf{T}_w$  for the basis,  $\tilde{\mathbf{S}}_w = \mathbf{T}_w^\top\mathbf{S}_w\mathbf{T}_w^{-\top}$  for the shift matrix and  $\tilde{\mathbf{L}} = \mathbf{L}\mathbf{T}_w$  for the left tangential direction matrix.  $\triangle$

**Deflation and orthogonalization** For a well-conditioned projection, it is essential that the reduction bases  $\mathbf{V}, \mathbf{W}$  have full column rank, and that the matrix  $\mathbf{W}^\top\mathbf{E}\mathbf{V}$  is invertible (cf. Section 3.2). Moreover, for better numerical robustness, it is highly recommended (but not mandatory) that the projection matrices are orthogonal. Hence, deflation or orthogonalization techniques are often employed. If, on the one hand, the projection matrices contain linearly dependent columns, then a rank-revealing QR decomposition (RRQR) or a singular value decomposition (SVD) can be performed to deflate the matrices and obtain full rank, orthogonal bases. On the other hand, full rank bases can be orthogonalized via a QR factorization or Gram-Schmidt process. In the aforementioned approaches, the deflation/orthogonalization is

accomplished a-posteriori, i.e. when all basis vectors have been computed. Nevertheless, a *modified* Gram-Schmidt procedure can also be employed within the Arnoldi iteration, in order to obtain orthonormal bases in each step and improve the numerical stability. Note, however, that modified Gram-Schmidt might not yield a (fully) orthonormal basis, specially if many columns are computed. In such case, a reorthogonalization via the mentioned a-posteriori approaches might be needed. Further note that an orthogonalization only changes the basis from  $\mathbf{V} \rightarrow \tilde{\mathbf{V}}$ , without modifying the subspace (this is not the case for deflation). Therefore, it can be understood as a similarity transformation with corresponding  $\mathbf{T}_v$ , yielding  $\tilde{\mathbf{V}} = \mathbf{V}\mathbf{T}_v$ ,  $\tilde{\mathbf{S}}_v = \mathbf{T}_v^{-1} \mathbf{S}_v \mathbf{T}_v$  and  $\tilde{\mathbf{R}} = \mathbf{R}\mathbf{T}_v$ . The same holds for  $\tilde{\mathbf{W}}$ ,  $\tilde{\mathbf{S}}_w$  and  $\tilde{\mathbf{L}}$ .

### 3.5 Equivalence of Krylov subspaces and Sylvester equations

There exists a strong connection between Krylov subspaces and Sylvester equations. In fact, any basis of an input and output Krylov subspace can be equivalently interpreted as the solution  $\mathbf{V}$  and  $\mathbf{W}$  of the following Sylvester equations [116]:

$$\mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} = \mathbf{B} \mathbf{R}, \quad (3.62a)$$

$$\mathbf{E}^\top \mathbf{W} \mathbf{S}_w^\top - \mathbf{A}^\top \mathbf{W} = \mathbf{C}^\top \mathbf{L}. \quad (3.62b)$$

Hereby, the input interpolation data  $\{\sigma_i, \mathbf{r}_i\}$  is encoded in the matrices  $\mathbf{S}_v \in \mathbb{C}^{r \times r}$  and  $\mathbf{R} \in \mathbb{C}^{m \times r}$ , where the pair  $(\mathbf{R}, \mathbf{S}_v)$  is observable. Similarly, the output interpolation data  $\{\mu_i, \mathbf{l}_i\}$  is specified by the matrices  $\mathbf{S}_w \in \mathbb{C}^{r \times r}$  and  $\mathbf{L} \in \mathbb{C}^{p \times r}$ , where the pair  $(\mathbf{S}_w, \mathbf{L}^\top)$  is controllable. Note that in the multimoment case,  $\mathbf{S}_v, \mathbf{S}_w$  are Jordan matrices, and that in the SISO case,  $\mathbf{R}, \mathbf{L}$  become row vectors with corresponding ones and zeros.

In order to make this equivalence more clear, in the following we explicitly give the Sylvester matrices and the Arnoldi recurrence for four different moment matching cases, and also characterize the respective moments in terms of the corresponding projection matrices  $\mathbf{V}$  and  $\mathbf{W}$ . The goal is to understand the conceptual similarities between Sylvester equations, the Arnoldi process and the matched moments through different examples.

#### Block multimoment case

In the block case, one receives  $m$  or  $p$  columns per shift. Hence, for a single shift  $\sigma, \mu$  with multiplicity  $q$  (cf. Theorem 3.1), the Sylvester matrices have the dimensions  $\mathbf{S}_v \in \mathbb{C}^{mq \times mq}$ ,  $\mathbf{R} \in \mathbb{C}^{m \times mq}$ ,  $\mathbf{S}_w \in \mathbb{C}^{pq \times pq}$ ,  $\mathbf{L} \in \mathbb{C}^{p \times pq}$ . The multiplicity of the shift is represented by *Jordan* matrices  $\mathbf{S}_v, \mathbf{S}_w$  with negative off-diagonal square blocks:

$$\mathbf{S}_v = \begin{bmatrix} \sigma \mathbf{I}_m & -\mathbf{I}_m & & \\ & \ddots & \ddots & \\ & & \ddots & -\mathbf{I}_m \\ & & & \sigma \mathbf{I}_m \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} \mathbf{I}_m & \mathbf{0}_m & \cdots & \mathbf{0}_m \end{bmatrix}, \quad (3.63a)$$

$$\mathbf{S}_w = \begin{bmatrix} \mu \mathbf{I}_p & & & \\ -\mathbf{I}_p & \ddots & & \\ & \ddots & \ddots & \\ & & -\mathbf{I}_p & \mu \mathbf{I}_p \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} \mathbf{I}_p & \mathbf{0}_p & \cdots & \mathbf{0}_p \end{bmatrix}. \quad (3.63b)$$

The matrices  $\mathbf{V} \in \mathbb{C}^{n \times mq}$ ,  $\mathbf{W} \in \mathbb{C}^{n \times pq}$  either represent the unique solution of the Sylvester equations (3.62a),(3.62b) with the respective Sylvester matrices given above, or can be alternatively computed via the Arnoldi process

$$(\sigma \mathbf{E} - \mathbf{A})\mathbf{V}_0 = \mathbf{B}, \quad (\sigma \mathbf{E} - \mathbf{A})\mathbf{V}_\ell = \mathbf{E} \mathbf{V}_{\ell-1}, \quad \ell = 1, \dots, q-1, \quad (3.64a)$$

$$(\mu \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{W}_0 = \mathbf{C}^\top, \quad (\mu \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{W}_\ell = \mathbf{E}^\top \mathbf{W}_{\ell-1}, \quad \ell = 1, \dots, q-1, \quad (3.64b)$$

where  $\mathbf{V} = [\mathbf{V}_0, \dots, \mathbf{V}_{q-1}]$  and  $\mathbf{W} = [\mathbf{W}_0, \dots, \mathbf{W}_{q-1}]$ . The moments  $\mathbf{m}_\ell(\sigma), \mathbf{m}_\ell(\mu) \in \mathbb{C}^{p \times m}$  at  $\sigma, \mu$  are given by

$$\mathbf{m}_\ell(\sigma) = (-1)^\ell \mathbf{C} \mathbf{V}_\ell, \quad (3.65a)$$

$$\mathbf{m}_\ell(\mu) = (-1)^\ell \mathbf{W}_\ell^\top \mathbf{B}, \quad (3.65b)$$

where  $\mathbf{V}_\ell$  and  $\mathbf{W}_\ell$  are calculated according to above.

### Block multipoint case

For different shifts  $\sigma_i, \mu_i, i = 1, \dots, r$  with multiplicity  $q_i = 1$ , the Sylvester matrices have the dimensions  $\mathbf{S}_v \in \mathbb{C}^{mr \times mr}$ ,  $\mathbf{R} \in \mathbb{C}^{m \times mr}$ ,  $\mathbf{S}_w \in \mathbb{C}^{pr \times pr}$ ,  $\mathbf{L} \in \mathbb{C}^{p \times pr}$ :

$$\mathbf{S}_v = \begin{bmatrix} \sigma_1 \mathbf{I}_m & & \\ & \ddots & \\ & & \sigma_r \mathbf{I}_m \end{bmatrix}, \quad \mathbf{R} = [\mathbf{I}_m \quad \mathbf{I}_m \quad \cdots \quad \mathbf{I}_m], \quad (3.66a)$$

$$\mathbf{S}_w = \begin{bmatrix} \mu_1 \mathbf{I}_p & & \\ & \ddots & \\ & & \mu_r \mathbf{I}_p \end{bmatrix}, \quad \mathbf{L} = [\mathbf{I}_p \quad \mathbf{I}_p \quad \cdots \quad \mathbf{I}_p]. \quad (3.66b)$$

The matrices  $\mathbf{V} \in \mathbb{C}^{n \times mr}$ ,  $\mathbf{W} \in \mathbb{C}^{n \times pr}$  either represent the unique solution of the Sylvester equations (3.62a),(3.62b) with the respective Sylvester matrices given above, or can be alternatively computed via the Arnoldi process

$$(\sigma_i \mathbf{E} - \mathbf{A})\mathbf{V}_i = \mathbf{B}, \quad i = 1, \dots, r, \quad (3.67a)$$

$$(\mu_i \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{W}_i = \mathbf{C}^\top, \quad i = 1, \dots, r, \quad (3.67b)$$

where  $\mathbf{V} = [\mathbf{V}_1, \dots, \mathbf{V}_r]$  and  $\mathbf{W} = [\mathbf{W}_1, \dots, \mathbf{W}_r]$ . The 0-th moments  $\mathbf{m}_0(\sigma_i), \mathbf{m}_0(\mu_i) \in \mathbb{C}^{p \times m}$  at  $\sigma_i, \mu_i$  are given by

$$\mathbf{m}_0(\sigma_i) = \mathbf{C} \mathbf{V}_i, \quad (3.68a)$$

$$\mathbf{m}_0(\mu_i) = \mathbf{W}_i^\top \mathbf{B}, \quad (3.68b)$$

where  $\mathbf{V}_i$  and  $\mathbf{W}_i$  are calculated according to above.

### Tangential multimoment case

In order to avoid the increased number of columns from the block case, one could choose a single tangential direction  $\mathbf{r}, \mathbf{l}$  to tangentially match high-order moments at a single shift  $\sigma, \mu$ . In this case, the Sylvester matrices become  $\mathbf{S}_v \in \mathbb{C}^{q \times q}$ ,  $\mathbf{R} \in \mathbb{C}^{m \times q}$ ,  $\mathbf{S}_w \in \mathbb{C}^{q \times q}$ ,  $\mathbf{L} \in \mathbb{C}^{p \times q}$ :

$$\mathbf{S}_v = \begin{bmatrix} \sigma & -1 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \ddots & -1 \\ & & & & \sigma \end{bmatrix}, \quad \mathbf{R} = [\mathbf{r} \quad \mathbf{0} \quad \cdots \quad \mathbf{0}], \quad (3.69a)$$

$$\mathbf{S}_w = \begin{bmatrix} \mu & & & & \\ -1 & \ddots & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & -1 & \mu \end{bmatrix}, \quad \mathbf{L} = [\mathbf{l} \quad \mathbf{0} \quad \cdots \quad \mathbf{0}]. \quad (3.69b)$$

The matrices  $\mathbf{V} \in \mathbb{C}^{n \times q}$ ,  $\mathbf{W} \in \mathbb{C}^{n \times q}$  either represent the unique solution of the Sylvester equations (3.62a),(3.62b) with the respective Sylvester matrices given above, or can be alternatively computed via the Arnoldi process

$$(\sigma \mathbf{E} - \mathbf{A})\mathbf{v}_0 = \mathbf{B} \mathbf{r}, \quad (\sigma \mathbf{E} - \mathbf{A})\mathbf{v}_\ell = \mathbf{E} \mathbf{v}_{\ell-1}, \quad \ell = 1, \dots, q-1, \quad (3.70a)$$

$$(\mu \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{w}_0 = \mathbf{C}^\top \mathbf{l}, \quad (\mu \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{w}_\ell = \mathbf{E}^\top \mathbf{w}_{\ell-1}, \quad \ell = 1, \dots, q-1, \quad (3.70b)$$

where  $\mathbf{V} = [\mathbf{v}_0, \dots, \mathbf{v}_{q-1}]$  and  $\mathbf{W} = [\mathbf{w}_0, \dots, \mathbf{w}_{q-1}]$ . The *tangential* moments  $\mathbf{m}_\ell(\sigma, \mathbf{r}) \in \mathbb{C}^p$  at  $\{\sigma, \mathbf{r}\}$  and  $\mathbf{m}_\ell(\mu, \mathbf{l}) \in \mathbb{C}^m$  at  $\{\mu, \mathbf{l}\}$  are given by

$$\mathbf{m}_\ell(\sigma, \mathbf{r}) := \mathbf{m}_\ell(\sigma) \mathbf{r} = (-1)^\ell \mathbf{C} \mathbf{v}_\ell, \quad (3.71a)$$

$$\mathbf{m}_\ell^\top(\mu, \mathbf{l}) := \mathbf{l}^\top \mathbf{m}_\ell(\mu) = (-1)^\ell \mathbf{w}_\ell^\top \mathbf{B}, \quad (3.71b)$$

where  $\mathbf{v}_\ell$  and  $\mathbf{w}_\ell$  are calculated according to above.

### Tangential multipoint case

For different shifts  $\sigma_i, \mu_i$  with respective tangential directions  $\mathbf{r}_i, \mathbf{l}_i$  (cf. Theorem 3.2), the Sylvester matrices become  $\mathbf{S}_v \in \mathbb{C}^{r \times r}$ ,  $\mathbf{R} \in \mathbb{C}^{m \times r}$ ,  $\mathbf{S}_w \in \mathbb{C}^{r \times r}$ ,  $\mathbf{L} \in \mathbb{C}^{p \times r}$ :

$$\mathbf{S}_v = \text{diag}(\sigma_1, \dots, \sigma_r), \quad \mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \cdots \quad \mathbf{r}_r], \quad (3.72a)$$

$$\mathbf{S}_w = \text{diag}(\mu_1, \dots, \mu_r), \quad \mathbf{L} = [\mathbf{l}_1 \quad \mathbf{l}_2 \quad \cdots \quad \mathbf{l}_r]. \quad (3.72b)$$

The matrices  $\mathbf{V} \in \mathbb{C}^{n \times r}$ ,  $\mathbf{W} \in \mathbb{C}^{n \times r}$  either represent the unique solution of the Sylvester equations (3.62a),(3.62b) with the respective Sylvester matrices given above, or can be alternatively computed via the Arnoldi process

$$(\sigma_i \mathbf{E} - \mathbf{A})\mathbf{v}_i = \mathbf{B} \mathbf{r}_i, \quad i = 1, \dots, r, \quad (3.73a)$$

$$(\mu_i \mathbf{E}^\top - \mathbf{A}^\top)\mathbf{w}_i = \mathbf{C}^\top \mathbf{l}_i, \quad i = 1, \dots, r, \quad (3.73b)$$



where  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r]$  and  $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_r]$ . The 0-th tangential moments  $\mathbf{m}_0(\sigma_i, \mathbf{r}_i) \in \mathbb{C}^p$  at  $\{\sigma_i, \mathbf{r}_i\}$  and  $\mathbf{m}_0(\mu_i, \mathbf{l}_i) \in \mathbb{C}^m$  at  $\{\mu_i, \mathbf{l}_i\}$  are given by

$$\mathbf{m}_0(\sigma_i, \mathbf{r}_i) := \mathbf{m}_0(\sigma_i) \mathbf{r}_i = \mathbf{C} \mathbf{v}_i, \quad (3.74a)$$

$$\mathbf{m}_0^\top(\mu_i, \mathbf{l}_i) := \mathbf{l}_i^\top \mathbf{m}_0(\mu_i) = \mathbf{w}_i^\top \mathbf{B}, \quad (3.74b)$$

where  $\mathbf{v}_i$  and  $\mathbf{w}_i$  are calculated according to above.

Note that the Sylvester matrices for the most general (block or tangential) multimoment+multipoint case (cf. Eq. (3.52)) can be easily given based on the examples above.

To conclude this section, we briefly focus on the numerical solution of the linear Sylvester equations (3.62). Although we will mostly use the equivalence between Krylov subspaces and Sylvester equations for theoretical considerations rather than for computational purposes, it is important to understand how such matrix equations are efficiently solved.

**Remark 3.2** (Solution of linear Sylvester equations). As already mentioned, there are two possible ways to compute the reduction bases  $\mathbf{V}$  and  $\mathbf{W}$  spanning respective input and output Krylov subspaces. On the one hand, it is possible to solve the above shifted linear systems of equations (LSE) arising during the Arnoldi process (cf. Section 2.5). On the other hand, one could also solve the above generalized Sylvester equations, since this is equivalent to the explicit computation of the shifted LSEs. This becomes clear, when the Sylvester equations are reformulated as LSEs using the Kronecker product and the vectorization operator:

$$\mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} = \mathbf{B} \mathbf{R} \quad \iff \quad \left( \mathbf{S}_v^\top \otimes \mathbf{E} - \mathbf{I}_r \otimes \mathbf{A} \right) \text{vec}(\mathbf{V}) = \text{vec}(\mathbf{B} \mathbf{R}), \quad (3.75a)$$

$$\mathbf{E}^\top \mathbf{W} \mathbf{S}_w^\top - \mathbf{A}^\top \mathbf{W} = \mathbf{C}^\top \mathbf{L} \quad \iff \quad \left( \mathbf{S}_w \otimes \mathbf{E}^\top - \mathbf{I}_r \otimes \mathbf{A}^\top \right) \text{vec}(\mathbf{W}) = \text{vec}(\mathbf{C}^\top \mathbf{L}). \quad (3.75b)$$

The direct solution of Sylvester equations via the Bartels-Stewart algorithm (`lyap`) or by solving the big LSEs (3.75) explicitly (`sylvester`, `\`) is only feasible for medium-sized models. Generally, large sparse-dense Sylvester equations are solved iteratively using RKSM, LR-ADI or low-rank variants of Krylov LSE solvers (cf. Section 2.4).  $\triangle$

## 3.6 Time-domain interpretation of linear moment matching

In addition to the classical frequency-domain perception of moment matching as the interpolation of the transfer function at certain shifts, one can also interpret this concept in the time-domain (cf. [12, 14]). Based on the ideas from Astolfi linear moments will be first interpreted as the steady-state response of the linear system excited with a signal generator. Then, we can construe moment matching as the interpolation of the steady-state response of this interconnected system.

For the sake of brevity, from now on we only consider the *input tangential multipoint* moment matching case. The  $\mathbf{W}$ -sided and multimoment setting will be only mentioned in remarks.

### 3.6.1 Time-domain notion of linear moments

**Definition 3.5** (Time-domain linear moments). The moments  $\bar{\mathbf{m}}_\ell(\sigma)$  of the impulse response  $\mathbf{g}(t)$  at  $\sigma$  are given by weighted integrals over the time function (3.3) and satisfy (after repeated partial integration)

$$\begin{aligned}\bar{\mathbf{m}}_\ell(\sigma) &= \int_{\tau=0}^{\infty} \tau^\ell e^{-\sigma\tau} \mathbf{g}(\tau) d\tau = \int_{\tau=0}^{\infty} \tau^\ell \mathbf{C} e^{(\mathbf{E}^{-1}\mathbf{A}-\sigma\mathbf{I})\tau} \mathbf{E}^{-1} \mathbf{B} d\tau \\ &= \ell! \mathbf{C} \left( (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^\ell (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{B}.\end{aligned}\tag{3.76}$$

Hence, the time-domain moments  $\bar{\mathbf{m}}_\ell(\sigma)$  and the frequency-domain moments  $\mathbf{m}_\ell(\sigma)$  from Eq. (3.46) only differ by a factor:

$$\mathbf{m}_\ell(\sigma) = \frac{(-1)^\ell}{\ell!} \bar{\mathbf{m}}_\ell(\sigma).\tag{3.77}$$

The time-domain moments can also be derived using the frequency shifting and frequency derivative properties of the Laplace transform, i.e.  $(-1)^\ell \mathbf{G}^{(\ell)}(s + \sigma) \bullet \longleftarrow t^\ell \mathbf{g}(t) e^{-\sigma t}$ .  $\blacktriangle$

This new definition will help us to understand the interpretation of moments as the steady-state response of the system (3.1) interconnected with a linear signal generator.

#### Notion of linear signal generator

Consider a *linear signal generator* (cf. [12, 14])

$$\dot{\mathbf{x}}_r^v(t) = \mathbf{S}_v \mathbf{x}_r^v(t), \quad \mathbf{x}_r^v(0) = \mathbf{x}_{r,0}^v \neq \mathbf{0},\tag{3.78a}$$

$$\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t),\tag{3.78b}$$

with the following assumptions:

1. The triple  $(\mathbf{S}_v, \mathbf{R}, \mathbf{x}_{r,0}^v)$  is minimal, i.e.
  - $(\mathbf{R}, \mathbf{S}_v)$  is observable (cf. Definition 3.3). This implies that the shift matrix  $\mathbf{S}_v$  is non-derogatory [229, Sec. 2.2.1], meaning that the geometric multiplicity of each eigenvalue  $\sigma_i$  is smaller or equal to the number of inputs  $m$  [273, Sec. 2.3]. This assumption is essential to obtain a *full rank* basis  $\mathbf{V}$ .
  - $(\mathbf{S}_v, \mathbf{x}_{r,0}^v)$  is controllable (cf. Definition 3.2). The controllability of this pair, aka. excitability, implies that the generated input signals  $\mathbf{u}(t)$  are *persistently exciting* [16, Ch. 2], [163, Ch. 13], [231, 230, 229].
2.  $\lambda(\mathbf{S}_v) \subset \mathbb{C}_0$ . This means that only pure imaginary shifts are allowed, meaning that the generated inputs  $\mathbf{u}(t)$  represent permanent oscillations.

As we will see later, these assumptions — in particular the second one — guarantee the existence of a *well-defined* steady-state response of the interconnected system. Nevertheless, less restrictive assumptions can also be imposed on the signal generator:

1. The pair  $(\mathbf{R}, \mathbf{S}_v)$  is observable, and
2.  $\lambda(\mathbf{S}_v) \cap \lambda(\mathbf{E}^{-1}\mathbf{A}) = \emptyset$ . This means that the generated inputs can be decaying ( $\mathbb{C}_-$ ), permanent oscillation ( $\mathbb{C}_0$ ) or growing ( $\mathbb{C}_+$ ) signals.

As already insinuated, interconnecting a system with the linear signal generator (3.78) corresponds to exciting the system with exponential input signals  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$  that are parametrized by the shift  $\mathbf{S}_v$  and tangential matrix  $\mathbf{R}$ . In the following, we give two examples to illustrate how the generated inputs look like depending on the interpolation data.

**Multimoment case** If  $\mathbf{S}_v = [\sigma \ -1 \ 0; 0 \ \sigma \ -1; 0 \ 0 \ \sigma]$  with multiplicity  $q=3$  and  $\mathbf{R} = [\mathbf{r}, \mathbf{0}, \mathbf{0}]$  are employed, then the system is excited by

$$\mathbf{u}(t) = \sum_{\ell=0}^{q-1} \mathbf{u}_\ell(t) = \sum_{\ell=0}^{q-1} \frac{(-1)^\ell}{\ell!} \mathbf{r} t^\ell e^{\sigma t} x_{r,0,\ell}^v. \quad (3.79)$$

**Multipoint case** Alternatively, if  $\mathbf{S}_v = \text{diag}(\sigma_1, \dots, \sigma_r)$  and  $\mathbf{R} = [\mathbf{r}_1, \dots, \mathbf{r}_r]$  are used, then the system is excited by a sum of exponentials

$$\mathbf{u}(t) = \sum_{i=1}^r \mathbf{u}_i(t) = \sum_{i=1}^r \mathbf{r}_i x_{r,i}^v(t) = \sum_{i=1}^r \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v. \quad (3.80)$$

In what follows, we will restrict the analysis to this tangential multipoint case.

### Steady-state response of interconnected system

Consider now the interconnection of system (3.1), where  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$  and  $\mathbf{x}(0) = \mathbf{x}_0$ , with the linear signal generator (3.78), cf. Fig. 3.1. The response of such interconnected system is given by

$$\mathbf{x}(t) = \underbrace{\int_0^t e^{\mathbf{E}^{-1}\mathbf{A}(t-\tau)} \mathbf{E}^{-1} \mathbf{B} \mathbf{u}(\tau) d\tau}_{\mathbf{x}_p(t)} + \underbrace{e^{\mathbf{E}^{-1}\mathbf{A}t} \mathbf{x}_0}_{\mathbf{x}_h(t)}, \quad (3.81)$$

where  $\mathbf{x}_h(t)$  is the *homogeneous solution* and  $\mathbf{x}_p(t)$  is the *particular solution*. Inserting the input  $\mathbf{u}(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v = \sum_{i=1}^r \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v$  yields

$$\begin{aligned} \mathbf{x}(t) &= e^{\mathbf{E}^{-1}\mathbf{A}t} \sum_{i=1}^r \int_0^t e^{(\sigma_i \mathbf{I} - \mathbf{E}^{-1}\mathbf{A})\tau} d\tau \mathbf{E}^{-1} \mathbf{B} \mathbf{r}_i x_{r,0,i}^v + e^{\mathbf{E}^{-1}\mathbf{A}t} \mathbf{x}_0 \\ &= e^{\mathbf{E}^{-1}\mathbf{A}t} \sum_{i=1}^r \left( e^{(\sigma_i \mathbf{I} - \mathbf{E}^{-1}\mathbf{A})t} - \mathbf{I} \right) (\sigma_i \mathbf{I} - \mathbf{E}^{-1}\mathbf{A})^{-1} \mathbf{E}^{-1} \mathbf{B} \mathbf{r}_i x_{r,0,i}^v + e^{\mathbf{E}^{-1}\mathbf{A}t} \mathbf{x}_0 \\ &= e^{\mathbf{E}^{-1}\mathbf{A}t} \left( \mathbf{x}_0 - \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i x_{r,0,i}^v \right) + \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v, \\ &= \underbrace{e^{\mathbf{E}^{-1}\mathbf{A}t} (\mathbf{x}_0 - \mathbf{V} \mathbf{x}_{r,0}^v)}_{\mathbf{x}_i(t)} + \underbrace{\mathbf{V} \mathbf{x}_r^v(t)}_{\mathbf{x}_{ss}(t)}. \end{aligned} \quad (3.82)$$

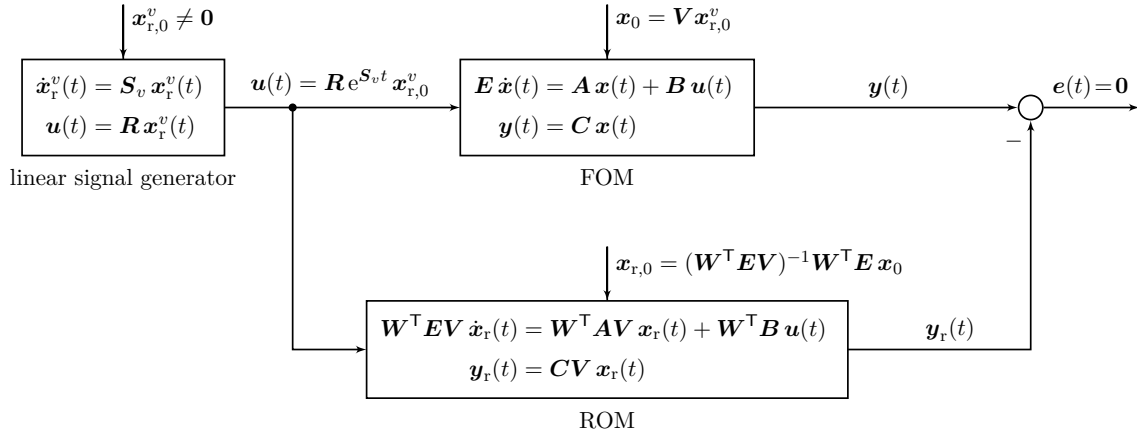


Figure 3.1: Interconnection between the linear FOM/ROM and the linear signal generator to illustrate the time-domain interpretation of moment matching for linear systems.

The *steady-state solution*  $\mathbf{x}_{\text{ss}}(t)$  is given by

$$\mathbf{x}_{\text{ss}}(t) = \sum_{i=1}^r \underbrace{(\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i}_{\mathbf{v}_i} \underbrace{e^{\sigma_i t} x_{r,0,i}^v}_{x_{r,i}^v(t)} = \sum_{i=1}^r \mathbf{v}_i x_{r,i}^v(t) = \mathbf{V} \mathbf{x}_r^v(t). \quad (3.83)$$

Hence,  $\mathbf{y}_{\text{ss}}(t) = \mathbf{C} \mathbf{x}_{\text{ss}}(t) = \mathbf{C} \mathbf{V} \mathbf{x}_r^v(t)$  represents the *steady-state response* of the linear system (3.1) to an input generated by (3.78). By imposing the following condition on the initial state

$$\mathbf{x}_0 \stackrel{!}{=} \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i x_{r,0,i}^v \stackrel{!}{=} \mathbf{V} \mathbf{x}_{r,0}^v \quad (3.84)$$

for  $\mathbf{x}_{r,0}^v \neq \mathbf{0}$  arbitrary, the *transient solution*  $\mathbf{x}_t(t)$  vanishes for all  $t$ . This yields  $\mathbf{x}(t) = \mathbf{x}_{\text{ss}}(t) \forall t$ .

Based on this result, we are ready to present the steady-state interpretation of linear input moments in the next lemma.

**Lemma 3.1** (Steady-state notion of linear input moments). The 0-th tangential moments  $\mathbf{m}_0(\sigma_i, \mathbf{r}_i)$  at  $\{\sigma_i, \mathbf{r}_i\}$  from (3.74a) are related to the (well-defined) steady-state response

$$\begin{aligned} \mathbf{y}_{\text{ss}}(t) &= \sum_{i=1}^r \mathbf{y}_{\text{ss},i}(t) = \sum_{i=1}^r \mathbf{C} (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v \\ &= \sum_{i=1}^r \mathbf{m}_0(\sigma_i, \mathbf{r}_i) e^{\sigma_i t} x_{r,0,i}^v = \mathbf{C} \mathbf{V} \mathbf{x}_r^v(t), \end{aligned} \quad (3.85)$$

of the interconnected system from Fig. 3.1, where  $\mathbf{V}$  is the unique solution of the Sylvester equation (3.62a) with the corresponding Sylvester matrices (3.72a), or  $\mathbf{v}_i = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i$  coming from the Arnoldi process (3.73a).

**Remark 3.3** (Global invariant manifold [14]). The interconnected system has a globally well-defined invariant manifold given by  $\mathcal{M} = \{(\mathbf{x}, \mathbf{x}_r^v) \in \mathbb{R}^{n+r} : \mathbf{x} = \mathbf{V} \mathbf{x}_r^v\}$ , where  $\mathbf{V}$  satisfies (3.62a). Moreover, the dynamics of the interconnected system restricted to the manifold  $\mathbf{x} = \mathbf{V} \mathbf{x}_r^v$  are described by  $\dot{\mathbf{x}}_r^v = \mathbf{S}_v \mathbf{x}_r^v$ , since  $\mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r^v = (\mathbf{A} \mathbf{V} + \mathbf{B} \mathbf{R}) \mathbf{x}_r^v$ .  $\triangle$

**Remark 3.4** (Steady-state perception for other cases). Note that the steady-state interpretation of moments stated above can be generalized for other moment matching cases based on the Sylvester matrices given in Section 3.5.  $\triangle$

**Remark 3.5** (Steady-state perception of output moments). For brevity, the steady-state interpretation of output moments  $\mathbf{m}_0^\top(\mu_i, \mathbf{l}_i)$  is not considered in this thesis. For details concerning *output Krylov moment matching* the reader is referred to [15, 126, 127, 70].  $\triangle$

### 3.6.2 Moment matching by interconnection

Based on Lemma 3.1, the perception of linear moment matching in terms of the interpolation of the steady-state response of an interconnected system follows.

**Theorem 3.3** (Steady-state-based linear moment matching). *Consider the interconnection of system (3.1) with the linear signal generator (3.78), where the triple  $(\mathbf{S}_v, \mathbf{R}, \mathbf{x}_{r,0}^v)$  is minimal and  $\lambda(\mathbf{S}_v) \cap \lambda(\mathbf{E}^{-1}\mathbf{A}) = \emptyset$ . Let  $\mathbf{V}$  be the unique solution of the Sylvester equation (3.62a) and  $\mathbf{W}$  arbitrary such that  $\det(\mathbf{W}^\top \mathbf{E} \mathbf{V}) \neq 0$ . Furthermore, let  $\mathbf{x}_0 = \mathbf{V} \mathbf{x}_{r,0}^v$  with  $\mathbf{x}_{r,0}^v \neq \mathbf{0}$  arbitrary. Then, the (asymptotically stable) ROM (3.29) exactly matches the (well-defined) steady-state response of the output of the FOM (cf. Fig. 3.1), i.e.*

$$\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{y}_r(t) = \mathbf{C} \mathbf{x}(t) - \mathbf{C} \mathbf{V} \mathbf{x}_r(t) = \mathbf{0} \quad \forall t. \quad (3.86)$$

**Corollary 3.1** (Exact moment matching vs. interpolation). *Consequently, moment matching for linear systems can be interpreted as the exact matching of the steady-state response of the FOM and ROM*

$$\mathbf{y}_{\text{ss}}(t) = \sum_{i=1}^r \underbrace{\mathbf{C}(\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i}_{\mathbf{m}_0(\sigma_i, \mathbf{r}_i)} x_{r,i}^v(t) \equiv \sum_{i=1}^r \underbrace{\mathbf{C}_r(\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{B}_r \mathbf{r}_i}_{\mathbf{m}_{r,0}(\sigma_i, \mathbf{r}_i)} x_{r,i}^v(t) = \mathbf{y}_{r,\text{ss}}(t), \quad (3.87)$$

when both are excited by exponential input signals  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$  (see Fig. 3.1) with same  $\mathbf{S}_v, \mathbf{R}$  as the ones used during the reduction. For other arbitrary input signals (applied e.g. in the online phase), the steady-state response is interpolated. Note that the transient response of the FOM is also “matched” or rather vanishes, if the initial condition is chosen like in (3.84).

*Proof.* The output of the FOM for  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t)$  has been derived in (3.82) and Lemma 3.1. Exciting the asymptotically stable ROM, i.e.  $\lambda(\mathbf{E}_r^{-1} \mathbf{A}_r) \subset \mathbb{C}_-$ , with the very same signal  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t) = \sum_{i=1}^r \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v$  yields

$$\mathbf{x}_r(t) = e^{\mathbf{E}_r^{-1} \mathbf{A}_r t} \left( \mathbf{x}_{r,0} - \sum_{i=1}^r (\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{B}_r \mathbf{r}_i x_{r,0,i}^v \right) + \underbrace{\sum_{i=1}^r (\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{B}_r \mathbf{r}_i e^{\sigma_i t} x_{r,0,i}^v}_{\mathbf{x}_{r,\text{ss}}(t)},$$

and thus  $\mathbf{y}_{r,\text{ss}}(t) = \mathbf{C}_r \mathbf{x}_{r,\text{ss}}(t) = \mathbf{C} \mathbf{V} \mathbf{x}_{r,\text{ss}}(t)$ . Alternatively, inserting the input  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t)$  into the ROM state equation (3.29a) with imposed  $\mathbf{x}_r(t) \stackrel{!}{=} \mathbf{x}_r^v(t)$  leads to

$$\mathbf{W}^\top \mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r(t) = \mathbf{W}^\top \underbrace{(\mathbf{A} \mathbf{V} + \mathbf{B} \mathbf{R})}_{\mathbf{E} \mathbf{V} \mathbf{S}_v} \mathbf{x}_r(t),$$

and, consequently,

$$\dot{\mathbf{x}}_r(t) = \mathbf{S}_v \mathbf{x}_r(t), \quad \mathbf{x}_r(0) = (\mathbf{W}^\top \mathbf{E} \mathbf{V})^{-1} \mathbf{W}^\top \mathbf{E} \mathbf{x}_0,$$

whose solution is  $\mathbf{x}_r(t) = e^{\mathbf{S}_v t} \mathbf{x}_{r,0}$ . Hence, the output of the ROM for  $\mathbf{u}(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$  is given by  $\mathbf{y}_r(t) = \mathbf{C} \mathbf{V} \mathbf{x}_r(t) = \mathbf{C} \mathbf{V} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$ . Therefore, we achieve *exact* moment matching for *all*  $t$ , if  $\mathbf{x}_r(0) \stackrel{!}{=} \mathbf{x}_{r,0}^v$ , i.e. if  $\mathbf{x}_0 \stackrel{!}{=} \mathbf{V} \mathbf{x}_{r,0}^v$ .  $\blacksquare$

### 3.6.3 Derivation of linear input Sylvester equation

Interestingly enough, the Sylvester equation (3.62a) can be derived using the notion of signal generators. To this end, first insert the linear approximation ansatz  $\mathbf{x}(t) = \mathbf{V} \mathbf{x}_r(t)$  with  $\mathbf{x}_r(t) \stackrel{!}{=} \mathbf{x}_r^v(t)$  in the state equation (3.1a):

$$\mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r^v(t) = \mathbf{A} \mathbf{V} \mathbf{x}_r^v(t) + \mathbf{B} \mathbf{u}(t). \quad (3.88)$$

Subsequently, the linear signal generator  $\dot{\mathbf{x}}_r^v(t) = \mathbf{S}_v \mathbf{x}_r^v(t)$ ,  $\mathbf{u}(t) = \mathbf{R} \mathbf{x}_r^v(t)$  is plugged into (3.88), yielding

$$\mathbf{0} = (\mathbf{A} \mathbf{V} - \mathbf{E} \mathbf{V} \mathbf{S}_v + \mathbf{B} \mathbf{R}) \cdot \mathbf{x}_r^v(t). \quad (3.89)$$

Since the above equation holds for  $\mathbf{x}_r^v(t) = e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$  and the transition matrix  $e^{\mathbf{S}_v t}$  is invertible, the vector  $\mathbf{x}_r^v(t)$  can be factored out. Consequently, the *constant* (state-independent) linear Sylvester equation (3.62a) of dimension  $n \times r$  is obtained.

### 3.6.4 Families of reduced models achieving linear moment matching

The classical approach to construct a reduced model achieving (input) linear moment matching is based on a two-sided projection, where the ROM is given by Eq. (3.29) with  $\mathbf{V}$  as solution of the Sylvester equation (3.62a) and  $\mathbf{W}$  arbitrary but such that  $\det(\mathbf{W}^\top \mathbf{E} \mathbf{V}) \neq 0$ . Herein, the ROM is parametrized in the projection matrix  $\mathbf{W} \in \mathbb{R}^{n \times r}$ . These remaining degrees of freedom can be exploited to impose certain properties on the reduced model (such as stability, passivity, etc.) or can be used to achieve a better approximation (e.g. by matching more moments using an output Krylov subspace).

Another approach is to parametrize the family of ROMs achieving (input) linear moment matching w.r.t. the reduced input matrix  $\Delta \in \mathbb{R}^{r \times m}$ , yielding

$$\dot{\mathbf{x}}_r(t) = (\mathbf{S}_v - \Delta \mathbf{R}) \mathbf{x}_r(t) + \Delta \mathbf{u}(t), \quad (3.90a)$$

$$\mathbf{y}_r(t) = \mathbf{C} \mathbf{V} \mathbf{x}_r(t), \quad (3.90b)$$

where  $\mathbf{E}_r = \mathbf{I}_r$ ,  $\mathbf{A}_r = \mathbf{S}_v - \Delta \mathbf{R}$ ,  $\mathbf{B}_r = \Delta$  and  $\mathbf{C}_r = \mathbf{C} \mathbf{V}$ , with the input interpolation data given by the observable pair  $(\mathbf{R}, \mathbf{S}_v)$ , and  $\mathbf{V}$  as solution of the Sylvester equation (3.62a). The free parameter  $\Delta$  can then be selected to construct a ROM with desired properties (e.g. minimal reduced order, prescribed eigenvalues/zeros, relative degree, passivity), but should satisfy the constraint  $\lambda(\mathbf{S}_v) \cap \lambda(\mathbf{S}_v - \Delta \mathbf{R}) = \emptyset$ . Procedures to select  $\Delta$  such that the above mentioned properties are imposed on the ROM are described in detail in [14, 128], [229, Sec. 2.3.1]. At this point we only want to remark that the assignment of reduced eigenvalues is closely related to pole placement [7] as well as to  $\mathcal{H}_2$ -pseudo-optimal reduction (cf. PORK in [273]).

Note that the output  $\mathbf{y}_r(t)$  of both families of ROMs is independent of  $\mathbf{W}$  and  $\mathbf{\Delta}$ . Thus, both families achieve (input) linear moment matching with appropriate  $\mathbf{V}$ , where the former family is based on projection, whereas the latter is obtained in a non-projective manner.

### 3.7 $\mathcal{H}_2$ -optimal reduction of linear systems

Moment matching via rational Krylov subspaces allows to reduce an LTI system using desired shifts, multiplicities and tangential directions. However, the quality of the approximation heavily depends on the choice of the reduction parameters. In certain cases, the selection of the interpolation data can be a difficult task. Moreover, the question raises how to find the best possible approximation over the entire frequency range. This problem is addressed by  $\mathcal{H}_2$ -optimal model reduction, where the goal is to find a stable reduced model of desired *fixed* order  $r$  minimizing the  $\mathcal{H}_2$ -error:

$$\mathbf{G}_r(s) = \arg \min_{\deg(\tilde{\mathbf{G}}_r)=r} \|\mathbf{G} - \tilde{\mathbf{G}}_r\|_{\mathcal{H}_2}. \quad (3.91)$$

#### Necessary optimality conditions

Since this optimization problem is non-convex, the aim is usually to find at least a *locally optimal* reduced model. Considering the cost functional  $\mathcal{J} := \|\mathbf{\Sigma} - \mathbf{\Sigma}_r\|_{\mathcal{H}_2}^2$  different first-order necessary optimality conditions have been derived over the years. Meier and Luenberger derived in [179] *interpolation-based* optimality conditions for SISO models by formulating the cost functional in terms of poles/residues as  $\mathcal{J} = f(\phi_i, \lambda_i, \phi_{r,i}, \lambda_{r,i})$  and by setting the gradients  $\frac{\partial \mathcal{J}}{\partial \phi_{r,i}} \stackrel{!}{=} 0$  and  $\frac{\partial \mathcal{J}}{\partial \lambda_{r,i}} \stackrel{!}{=} 0$ . Later, Wilson [271] derived *Gramian-based* optimality conditions by parametrizing the cost functional in terms of the Lyapunov equations (3.34) as  $\mathcal{J} = f(\mathbf{A}_e, \mathbf{B}_e, \mathbf{C}_e, \mathbf{E}_e)$  and by differentiating w.r.t. the reduced matrices  $\mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r$ . Similar Lyapunov-based conditions were derived a few years later by Hyland and Bernstein [119]. It has been shown in [97] that all three formulations of necessary conditions are equivalent to each other. Here we focus on the interpolatory Meier-Luenberger conditions, which can be derived for the MIMO case by differentiating  $\mathcal{J} = f(\mathbf{A}, \mathbf{\Lambda}_r, \mathbf{B}, \hat{\mathbf{B}}_r, \mathbf{C}, \hat{\mathbf{C}}_r)$  w.r.t.  $\hat{\mathbf{C}}_r \in \mathbb{C}^{p \times r}$ ,  $\hat{\mathbf{B}}_r \in \mathbb{C}^{r \times m}$  and  $\mathbf{\Lambda}_r \in \mathbb{C}^{r \times r}$ . This yields the following  $(p + m + 1) \cdot r$  conditions:

$$\mathbf{G}(-\bar{\lambda}_{r,i}) \hat{\mathbf{b}}_{r,i} = \mathbf{G}_r(-\bar{\lambda}_{r,i}) \hat{\mathbf{b}}_{r,i}, \quad i = 1, \dots, r, \quad (3.92a)$$

$$\hat{\mathbf{c}}_{r,i}^\top \mathbf{G}(-\bar{\lambda}_{r,i}) = \hat{\mathbf{c}}_{r,i}^\top \mathbf{G}_r(-\bar{\lambda}_{r,i}), \quad i = 1, \dots, r, \quad (3.92b)$$

$$\hat{\mathbf{c}}_{r,i}^\top \mathbf{G}'(-\bar{\lambda}_{r,i}) \hat{\mathbf{b}}_{r,i} = \hat{\mathbf{c}}_{r,i}^\top \mathbf{G}'_r(-\bar{\lambda}_{r,i}) \hat{\mathbf{b}}_{r,i}, \quad i = 1, \dots, r, \quad (3.92c)$$

where  $\mathbf{E}_r^{-1} \mathbf{A}_r = \mathbf{X}_r \mathbf{\Lambda}_r \mathbf{X}_r^{-1}$ ,  $\hat{\mathbf{B}}_r = \mathbf{X}_r^{-1} \mathbf{E}_r^{-1} \mathbf{B}_r$  and  $\hat{\mathbf{C}}_r = \mathbf{C}_r \mathbf{X}_r$  represent the eigendecomposition of the reduced-order model, i.e.  $[\mathbf{X}_r, \mathbf{\Lambda}_r] = \mathbf{eig}(\mathbf{A}_r, \mathbf{E}_r)$  with  $\mathbf{\Lambda}_r = \text{diag}(\lambda_{r,1}, \dots, \lambda_{r,r})$ .

Comparing the optimality conditions (3.92) with the moment matching conditions (3.55) one can notice that a locally  $\mathcal{H}_2$ -optimal ROM interpolates the FOM at the mirrored reduced eigenvalues  $\sigma_i \leftarrow -\bar{\lambda}_{r,i}$  along the residue directions  $\mathbf{r}_i \leftarrow \hat{\mathbf{b}}_{r,i}$  and  $\mathbf{l}_i^\top \leftarrow \hat{\mathbf{c}}_{r,i}^\top$ . If one knew the reduced poles and residues in advance, then one could choose the shifts and tangential directions accordingly and perform a standard Krylov reduction. Since this is not the case, an Iterative Rational Krylov Algorithm (IRKA) is proposed in [97] to iteratively adapt the interpolation data.





### $\mathcal{H}_2$ -pseudo-optimal reduction of linear systems

The main idea of  $\mathcal{H}_2$ -pseudo-optimality is to find a reduced model that minimizes the  $\mathcal{H}_2$ -error within a certain subset  $\mathcal{G}$  of ROMs of order  $r$ : [32], [273, Sec. 4.3]

$$\mathbf{G}_r(s) = \arg \min_{\tilde{\mathbf{G}}_r \in \mathcal{G}} \|\mathbf{G} - \tilde{\mathbf{G}}_r\|_{\mathcal{H}_2}^2. \quad (3.93)$$

For instance, the subset can be chosen to have ROMs with fixed reduced poles and input residues  $(\lambda_{r,i}, \hat{\mathbf{b}}_{r,i})$ , or fixed reduced poles and output residues  $(\lambda_{r,i}, \hat{\mathbf{c}}_{r,i})$ . Although only the Lagrangian interpolation condition (3.92a) or (3.92b) is being satisfied in this framework (and not the Hermite one (3.92c)),  $\mathcal{H}_2$ -pseudo-optimality yields a *global* minimizer in the respective subspace. Further advantages are the structured orthogonality condition  $\langle \mathbf{G} - \mathbf{G}_r, \mathbf{G}_r \rangle = \mathbf{0}$  simplifying the cost functional  $\mathcal{J}$ , as well as the deliberate construction of stable ROMs by placing the reduced poles at the mirror images of the chosen shifts, i.e.  $\lambda(-\mathbf{E}_r^{-1} \mathbf{A}_r) \leftarrow \lambda(\mathbf{S}_v)$ .

New conditions for  $\mathcal{H}_2$ -pseudo-optimality are presented in [273, Sec. 4.3]. Moreover, an iteration-free algorithm called PORK is proposed for the explicit construction of pseudo-optimal ROMs.  $\mathcal{H}_2$ -pseudo-optimality can also be applied to improve the convergence of IRKA via residue correction in an inner loop, used within the Cumulative Reduction (CuRe) framework and SPARK algorithm [198], or exploited to show the equivalence between the ADI and the RKSM with pseudo-optimal shifts.

In Section 5.5.2 we will briefly mention our extension of the concept of  $\mathcal{H}_2$ -pseudo-optimality to the bilinear setting.

## 3.8 Adaptive and cumulative rational Krylov subspace method

As discussed in the previous section, IRKA is a Krylov-based method that (upon convergence) yields a locally  $\mathcal{H}_2$ -optimal ROM by iteratively adapting the interpolation data. However, the reduced order  $r$  is not adapted during the procedure. This means that the algorithm might need to be restarted for a different fixed order if the approximation quality was not good enough. Thus, it would be also desirable to automatically select the reduced order by gradually augmenting the subspace until the desired accuracy is achieved.

There exist different procedures for an adaptive shift selection, most of them coming from the context of the iterative solution of Lyapunov equations (see e.g. [267, 202, 245, 39, 151, 279]). On the other hand, there exist several methods to cumulatively reduce an LTI system, e.g. the RKSM [83, 84], the Cumulative Reduction (CuRe) framework [198] or the iterative procedure [3]. During this doctoral endeavor we focused on the rational Krylov subspace method with the goal of implementing and integrating it in the sssMOR toolbox. We decided to exploit the algorithm for its two purposes, namely the approximate solution of Lyapunov equations and the iterative reduction of large-scale linear systems. Due to both purposes and the inherent cumulative construction of the Krylov subspaces, we called the algorithm *cumulative* rational Krylov subspace method (CRKSM)<sup>3</sup> in order to distinguish it from the standard RKSM for Lyapunov equations only.

<sup>3</sup>Note that our algorithm differs from the approach proposed in [273, Sec. 5.3], which combines RKSM with the cumulative framework CuRe for the approximate solution of Lyapunov equations.

## Cumulative Rational Krylov Subspace Method (CRKSM)

Our interest in RKSM originally arised from the approximate solution of *bilinear* Lyapunov equations (cf. Eq. (5.15), [257]) during the semester thesis [Hei17]. We then decided to refactor the code, include the MOR purpose and improve the algorithm during the master thesis [Hei18]. In the following we only sketch the most important ingredients of CRKSM. For a detailed discussion and numerical results the reader is referred to the mentioned theses.

Our CRKSM algorithm<sup>4</sup> containing both the Lyapunov and MOR purposes is mainly composed of the following steps:



1. Get new interpolation data (`getShifts`) or recycle initial shifts and tangential directions
2. Compute new directions for an input and/or output Krylov subspace:  $\tilde{\mathbf{V}}_k \leftarrow [\tilde{\mathbf{V}}_{k-1}, \mathbf{V}_k]$
3. Calculate the enlarged reduced matrices  $\mathbf{A}_r$ ,  $\mathbf{B}_r$ ,  $\mathbf{C}_r$ ,  $\mathbf{E}_r$  efficiently by only computing the entries associated to the new Krylov columns
4. *Lyap-purpose*: Solve the reduced Lyapunov equation (2.40) for  $\mathbf{P}_r = \mathbf{S}_r \mathbf{S}_r^\top$  using `lyapchol` (Hammarling). Then calculate the low-rank Cholesky factor  $\mathbf{Z}_{\text{RKSM}} = \mathbf{V} \mathbf{S}_r$
5. Evaluate an appropriate stopping criterion
  - *Lyap-purpose*: Calculate the norm of the residual (2.41) using the low-rank formulation given in [278]. A dual version is also implemented for the  $\mathbf{W}$ -sided case
  - *MOR-purpose*: Computing the true error  $\|\Sigma - \Sigma_{r,k}\|_{\mathcal{H}_2}^2$  via (3.33) is not feasible. Therefore, we employ the difference between the ROMs from the current and previous iteration, i.e.  $\|\Sigma_{r,k} - \Sigma_{r,k-1}\|_{\mathcal{H}_2}^2$
6. If the chosen tolerance is achieved, stop the algorithm. Otherwise go to step 1.

The strength of CRKSM is that it supports many different cases: one can choose between Lyap- or MOR-purpose, a one-sided (more likely stable) or two-sided reduction, and the block or tangential case. It can be called within the sss function `lyapchol` (including also `mess_lradi`) or run standalone. The adaptive selection of shifts is discussed next.

## Adaptive selection of shifts

For the selection of initial interpolation data we employ the function `initializeShifts`. Then, the user can decide to recycle these shifts or generate new “online” data using `getShifts`. In the latter function there are two strategies implemented:

1. Mirrored Ritz values: Similar to the IRKA update in Line 7, we employ the eigenvalues (and the eigenvectors) of the current ROM to obtain new interpolation data.
2. Adaptive scheme [245, 84]: The idea is to complement the above strategy with an additional evaluation criterion to select the most important expansion points. First, the previous shifts and the new mirrored Ritz values are used to build a spectral set as convex hull. Then, the shifts maximizing a residual based on the Skeleton approximation are selected. Tangential directions can be computed via a SVD of the residual.

	sss function(s):	<code>lyapchol (mess_lradi, crksm)</code>
	sssMOR function(s):	<code>crksm, initializeShifts, getShifts</code>

<sup>4</sup>`crksm` is included in the sssMOR toolbox available under <https://github.com/MORLab>. Try both out!

## PART II

### POLYNOMIAL NONLINEAR STATE-SPACE SYSTEMS



## Chapter 4

# Fundamentals of Polynomial Systems

In this chapter, we deal with the mathematical modeling and system-theoretic fundamentals of polynomial systems of the form

$$\Sigma_P : \begin{cases} \mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A}_1 \mathbf{x}(t) + \mathbf{A}_2 (\mathbf{x}(t) \otimes \mathbf{x}(t)) + \mathbf{A}_3 (\mathbf{x}(t) \otimes \mathbf{x}(t) \otimes \mathbf{x}(t)) + \cdots & (4.1a) \\ \quad + \mathbf{B} \mathbf{u}(t) + \sum_{j=1}^m \mathbf{B}_{1,j} \mathbf{x}(t) u_j(t) + \sum_{j=1}^m \mathbf{B}_{2,j} (\mathbf{x}(t) \otimes \mathbf{x}(t)) u_j(t) + \cdots, \\ \mathbf{y}(t) = \mathbf{C}_1 \mathbf{x}(t) + \mathbf{C}_2 (\mathbf{x}(t) \otimes \mathbf{x}(t)) + \cdots, & (4.1b) \end{cases}$$

where  $\mathbf{E} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{A}_k, \mathbf{B}_{k,j} \in \mathbb{R}^{n \times n^k}$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$  and  $\mathbf{C}_k \in \mathbb{R}^{p \times n^k}$ . We will first concentrate on different ways of obtaining such a polynomial system (Sec. 4.1), and then on the transformation to bilinear and quadratic-bilinear form (Sec. 4.2). After that, we will discuss two different approaches to gain a Volterra series representation in time-domain, namely the Picard iteration and the variational equation approach (Sec. 4.3). Finally, the different kernels and generalized transfer functions of (quadratic-)bilinear systems are presented (Sec. 4.4).

Sections 4.1, 4.2 and 4.3 represent a summary of the material discussed in [221, 109, 92], which is complemented here with our own viewpoint. Section 4.4 partly relies on our paper [60] and further explains the growing exponential approach for the *MIMO* case.

### 4.1 Obtaining a polynomial system

A polynomial system may originate in different ways. The most straightforward manner to obtain a polynomial representation (4.1) is by modeling a technical system that exhibits *only* polynomial nonlinearities in  $\mathbf{x}$  (e.g. quadratic, cubic, bilinear, quadratic-bilinear, etc.). Examples for inherent polynomial systems are  $\dot{x} = x^3 + x^2 u$ ,  $\dot{x} = -5x + x^2 + 3xu$ , the Fokker-Planck equation (bilinear), the Burgers equation (quadratic-bilinear), the Chafee-Infante and FitzHugh-Nagumo equations (cubic). However, in many applications other type of nonlinearities (e.g. exponential, rational, logarithmic, trigonometric, root functions, sensor/actuator characteristics, etc.) arise as well (e.g.  $\dot{x} = \sin(x) + e^{-x} + u$ ). In such case, a polynomial representation is foremost obtained from the following *input-affine* nonlinear system

$$\Sigma_{\text{NL,affine}} : \begin{cases} \mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t)) + \sum_{j=1}^m \mathbf{b}_j(\mathbf{x}(t)) u_j(t), & \mathbf{x}(0) = \mathbf{x}_0, & (4.2a) \\ \mathbf{y}(t) = \mathbf{c}(\mathbf{x}(t)), & & (4.2b) \end{cases}$$

where  $\mathbf{a}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ ,  $\mathbf{b}_j(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$  and  $\mathbf{c}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^p$  are *analytic* functions in  $\mathbf{x}$ . At this point there are basically two options. One can employ a Taylor series expansion of the nonlinearities to *approximate* (4.2) by a polynomial system (cf. Section 4.1.1). Another way is to represent (4.2) *exactly* as a polynomial system using the McCormick relaxation (cf. Section 4.1.2).

### 4.1.1 Taylor series expansion

In this section, we aim to approximate the analytic functions in (4.2) by a Taylor series expansion. For better readability we will drop the time argument in the following. We start by expanding the nonlinear function  $\mathbf{a}(\mathbf{x})$  at the equilibrium point  $\mathbf{x}_{\text{eq}}$ :

$$\mathbf{a}(\mathbf{x}) = \mathbf{a}(\mathbf{x}_{\text{eq}}) + \underbrace{\frac{\partial \mathbf{a}(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}}}_{\mathbf{A}_1} (\mathbf{x} - \mathbf{x}_{\text{eq}}) + \underbrace{\frac{1}{2!} \frac{\partial^2 \mathbf{a}(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}^2}}_{\mathbf{A}_2} (\mathbf{x} - \mathbf{x}_{\text{eq}})^{(2)} + \underbrace{\frac{1}{3!} \frac{\partial^3 \mathbf{a}(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}^3}}_{\mathbf{A}_3} (\mathbf{x} - \mathbf{x}_{\text{eq}})^{(3)} + \dots$$

Setting  $\mathbf{x}_{\text{eq}} = \mathbf{0}$  and assuming  $\mathbf{a}(\mathbf{x}_{\text{eq}}) = \mathbf{0}^1$ , we can write

$$\mathbf{a}(\mathbf{x}) = \mathbf{A}_1 \mathbf{x} + \mathbf{A}_2 \overbrace{(\mathbf{x} \otimes \mathbf{x})}^{\mathbf{x}^{(2)}} + \mathbf{A}_3 \overbrace{(\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x})}^{\mathbf{x}^{(3)}} + \dots = \sum_{k=1}^{\infty} \mathbf{A}_k \mathbf{x}^{(k)} \approx \sum_{k=1}^N \mathbf{A}_k \mathbf{x}^{(k)}, \quad (4.3)$$

where  $\mathbf{A}_k = \frac{1}{k!} \frac{\partial^k \mathbf{a}(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}^k} \in \mathbb{R}^{n \times n^k}$  and  $\mathbf{A}_1 \equiv \mathbf{A} \in \mathbb{R}^{n \times n}$  represents the well-known Jacobian. Similarly, we expand each function  $\mathbf{b}_j(\mathbf{x})$  in a Taylor series, now assuming that  $\mathbf{b}_j(\mathbf{x}_{\text{eq}}) \neq \mathbf{0}$ :

$$\mathbf{b}_j(\mathbf{x}) = \mathbf{B}_{0,j} + \mathbf{B}_{1,j} \mathbf{x} + \mathbf{B}_{2,j} (\mathbf{x} \otimes \mathbf{x}) + \dots = \sum_{k=0}^{\infty} \mathbf{B}_{k,j} \mathbf{x}^{(k)} \approx \sum_{k=0}^{N-1} \mathbf{B}_{k,j} \mathbf{x}^{(k)}, \quad (4.4)$$

where  $\mathbf{B}_{k,j} = \frac{1}{k!} \frac{\partial^k \mathbf{b}_j(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}^k} \in \mathbb{R}^{n \times n^k}$ . Note that  $\mathbf{B}_{0,j} \in \mathbb{R}^n$  corresponds to the vector  $\mathbf{b}_j(\mathbf{x}_{\text{eq}})$ . Last but not least, the function  $\mathbf{c}(\mathbf{x})$  can be expanded as follows:

$$\mathbf{c}(\mathbf{x}) = \mathbf{C}_0 + \mathbf{C}_1 \mathbf{x} + \mathbf{C}_2 (\mathbf{x} \otimes \mathbf{x}) + \dots = \sum_{k=0}^{\infty} \mathbf{C}_k \mathbf{x}^{(k)} \approx \sum_{k=0}^{N-1} \mathbf{C}_k \mathbf{x}^{(k)}, \quad (4.5)$$

where  $\mathbf{C}_k = \frac{1}{k!} \frac{\partial^k \mathbf{c}(\mathbf{x}_{\text{eq}})}{\partial \mathbf{x}^k} \in \mathbb{R}^{p \times n^k}$  and  $\mathbf{C}_0 \equiv \mathbf{c}(\mathbf{x}_{\text{eq}}) \in \mathbb{R}^p$ . Taking (4.3), (4.4) and (4.5) into account we can approximate (4.2) as a polynomial system

$$\mathbf{E} \dot{\mathbf{x}} = \sum_{k=1}^N \mathbf{A}_k \mathbf{x}^{(k)} + \sum_{j=1}^m \sum_{k=0}^{N-1} \mathbf{B}_{k,j} \mathbf{x}^{(k)} u_j, \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (4.6a)$$

$$\mathbf{y} = \sum_{k=0}^{N-1} \mathbf{C}_k \mathbf{x}^{(k)}. \quad (4.6b)$$

<sup>1</sup>The assumption that (4.2a) has a zero equilibrium  $\mathbf{x}_{\text{eq}} = \mathbf{0}$ , i.e.  $\mathbf{0} = \mathbf{a}(\mathbf{x}_{\text{eq}})$  holds for zero input  $\mathbf{u}(t) = \mathbf{0}$ , does not entail loss of generality. In fact, if these conditions are not met, then we can set  $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_0(t)$  and rewrite (4.2) such that  $\mathbf{x}_{\text{eq}} = \mathbf{0}$  and  $\mathbf{a}(\mathbf{0}) = \mathbf{0}$  hold. For further details, see [221, Sec. 3.3], [92, Sec. 2.5].

Note that in case of a more general, not input-affine nonlinear system  $\mathbf{E}\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$  (cf. Eq. (6.1)), the Taylor series expansion looks a little different (see Appendix A).

Next, we report different important aspects and properties of the Taylor series expansion. The goal is to discuss the applicability of this method in the large-scale setting [109, 224].

**Truncation of the series** Depending on the dynamics and the desired accuracy, the series should be truncated after one or another term. For instance, the Navier-Stokes equation could reasonably be truncated after the quadratic/cubic term. In other cases, it might be necessary to consider even higher-order terms (e.g. quartic, quintic) to obtain a good approximation. However, the storage limitation usually restricts the choice of a large truncation index  $N$ .

**Computation of the tensors** The polynomial tensors  $\mathbf{A}_k, \mathbf{B}_{k,j}, \mathbf{C}_k$  can be obtained from different strategies. The first approach (i) is to compute them analytically within the FD/FE/FV discretization procedure. This involves the analytical formulation and implementation of the partial derivatives of  $\mathbf{a}(\mathbf{x}), \mathbf{b}(\mathbf{x})$  and  $\mathbf{c}(\mathbf{x})$  at the discretization level. While this procedure can be well applied to rather simple (1D, 2D) problems with *local/jointed* nonlinearities using e.g. FD (c.f. Burgers equation in [25, 22]) or MNA (c.f. RC-Ladder in [47]), it becomes laborious and lengthy for *global/coupled* nonlinearities, especially if the implementation has to be accomplished for many different element types. Thus, the higher-order derivatives are mostly not implemented in (commercial) FE codes, where a nonlinear kinematic and/or material formulation usually leads to global nonlinearities. Moreover, the latter are not always accessible, thus restricting the applicability of this intrusive method.

The second approach (ii) consists in computing the tensors numerically via finite differences. Since the Jacobian matrix  $\mathbf{A}_1 \equiv \mathbf{A}$  is usually available analytically, the idea is to obtain the higher-order derivatives by numerical differentiation, e.g.  $\mathbf{A}_2 = \frac{1}{2!} \frac{\partial \mathbf{A}_1}{\partial \mathbf{x}}, \mathbf{A}_3 = \frac{1}{3!} \frac{\partial^2 \mathbf{A}_1}{\partial \mathbf{x}^2}$ , etc.. While this approach does not require access to the nonlinearities, it is computationally more expensive and less accurate than the analytical procedure described above. An appropriate step width  $h$  is necessary for the selected finite difference scheme.

The third approach (iii) consists in determining the polynomial tensors via identification (cf. [178, 210]). The main idea is to make an ansatz for the Taylor series expansion, evaluate the nonlinearities at different given test vectors  $\mathbf{x}$  and then calculate the coefficients by solving a linear system of equations. This method is fully non-intrusive, since it only requires to evaluate the nonlinear functions which are not needed analytically. However, the evaluation and identification process yields higher offline costs in comparison with the first approach.

**Efficient storage of the tensors** Depending on the type of nonlinearity (local vs. global), the tensors  $\mathbf{A}_k, \mathbf{B}_{k,j} \in \mathbb{R}^{n \times n^k}, \mathbf{C}_k \in \mathbb{R}^{p \times n^k}$  can be more or less sparse. For example, if many cross terms  $\mathbf{x}_i \cdot \mathbf{x}_j$  or  $\mathbf{x}_i \cdot \mathbf{x}_j \cdot \mathbf{x}_k$  vanish, then  $\mathbf{A}_2$  and  $\mathbf{A}_3$  are sparse matrices. However, the tensors scale with the full-order dimension  $n$ . Thus, it is not only essential to exploit sparsity, but also to store them efficiently. For instance, if the tensors are symmetric in modes one and two (i.e.  $\mathcal{A}_{2_{ijk}} = \mathcal{A}_{2_{jik}}$ ), two and three (i.e.  $\mathcal{A}_{2_{ijk}} = \mathcal{A}_{2_{ikj}}$ ) or even in all modes (i.e. supersymmetric), then redundant entries do not have to be stored. Despite these precautions, note that the storage of the tensors may rapidly exceed the available RAM in the large-scale setting ( $n \approx 10^3, \dots, 10^6$ ) and/or for a high truncation index  $N$ . Hence, depending on the case, it might be more reasonable to consider the Taylor series expansion of the nonlinearities at the *reduced-order* level (see Section 6.5.1).

**Local validity** The Taylor series expansion merely provides a local approximation of the nonlinearities around the chosen equilibrium point  $\mathbf{x}_{\text{eq}}$ . Only if the series converges after the truncated term, e.g.  $\mathbf{a}(\mathbf{x}) = \mathcal{O}(\mathbf{x}^3)$  and  $N = 3$ , then the polynomial representation (4.6) is *exact* and not an approximation anymore (cf. Chafee-Infante and FitzHugh-Nagumo).

### 4.1.2 Polynomialization procedure

Another way of obtaining a polynomial system (4.1) from (4.2) is presented in [110]. The procedure is closely related to the transformation via the *McCormick relaxation* [173].

Let us assume that  $\mathbf{a}(\mathbf{x})$  and  $\mathbf{b}(\mathbf{x})$  in (4.2) contain (compositions of) uni-variable functions such as exponential  $e^x$ , logarithmic  $\ln(x)$ , rational  $\frac{1}{x+k}$ , trigonometric  $\arctan(x)$ , root  $\sqrt{x}$ , etc.. The idea is then to transform the initial system into a polynomial form by introducing new state variables for the arising nonlinearities. There are two strategies for this purpose [110]:

1. polynomialization by adding polynomial algebraic equations [109, Sec. 6.3.2]
2. polynomialization by taking Lie derivatives [109, Sec. 6.3.3]

The first approach is only applicable to certain nonlinear functions (such as  $\frac{1}{x+k}$ ), while the latter can deal with a broader set of systems. Since both procedures are well explained in [Fio16], we refrain from giving the mathematical details and instead provide an example with the more general approach 2.

*Example 4.1* (Polynomialization procedure). Similar to the RC-Ladder model from [64], let us consider the parallel connection of a resistor  $R$ , capacitor  $C$  and Shockley diode with the current-voltage (I-V) characteristic  $i_D = e^{\alpha v} - 1$ . Applying Kirchhoff's point rule leads to the scalar nonlinear differential equation

$$\dot{v} = \frac{1}{C} \left( -\frac{v}{R} - e^{\alpha v} + 1 + i \right), \quad (4.7)$$

where  $i$  denotes the input current signal. Introducing the new variable  $w = e^{\alpha v} - 1$  together with its Lie derivative yields (with  $e^{\alpha v} = w + 1$ ):

$$\dot{v} = \frac{1}{C} \left( -\frac{v}{R} - w + i \right), \quad \dot{w} = \alpha e^{\alpha v} \dot{v} = \frac{\alpha}{C} \left( -\frac{vw}{R} - w^2 + wi - \frac{v}{R} - w + i \right). \quad (4.8)$$

This represents a polynomial (quadratic-bilinear) system consisting of 2 ODEs. △

Some important properties of the polynomialization procedure are reported in the following.

**Equivalent representation** The biggest advantage of the polynomialization procedure in comparison to the Taylor series expansion is that the former constitutes an *exact* transformation. This means that the dynamics of the nonlinear input-affine and polynomialized system are equivalent (as long as the initial conditions are also adjusted [109]). No approximation is performed during the whole process.

**Increase of dimension** Unfortunately, the original system dimension  $n$  is increased during the polynomialization procedure due to the introduction of new state variables. Luckily, the growth scales *linearly* with the number  $N_F$  of elementary functions contained in  $\mathbf{a}(\mathbf{x})$  and



$\mathbf{b}(\mathbf{x})$ . This means that the size of the equivalent polynomial system is  $n_P = N_F \cdot n$ , where  $N_F$  depends on the application at hand. For strongly compound functions and/or many different nonlinearities  $N_F$  might become large, but generally  $N_F \approx 2, \dots, 6$ . Note that  $e^x$  and  $\frac{1}{x+k}$  require only the introduction of *one* new variable, while some other elementary functions (e.g.  $\ln(x)$ ,  $\sin(x)$ ,  $\cos(x)$ ,  $\arctan(x)$ ,  $\sqrt{x}$ ) require the introduction of *two* new variables. Thus, the growth in dimension  $n_P$  not only depends on  $N_F$  but also on the *type* of nonlinearities.

**Non-unique transformation** The polynomialization procedure is not unique in the sense that different substitutions may lead to diverse polynomial systems with differently increased dimension. Although it is not trivial, there should exist an optimal transformation leading to the minimum-order (i.e. with least increased dimension) polynomialized system.

**Applicability** The assumption that the nonlinearities  $\mathbf{a}(\mathbf{x})$  and  $\mathbf{b}(\mathbf{x})$  are given by a composition of uni-variable functions is not a limiting factor. Indeed, this form of ODEs arise in many engineering applications, such as circuit simulation or from the discretization of PDEs. Due to the function composition, the method also allows to polynomialize many complicated nonlinear functions, thus covering a wide range of problems. Nevertheless, the procedure can only be applied to analytic nonlinearities. This means that explicit knowledge about the arising elementary functions is required, in order to introduce new state variables. Moreover, symbolic computation tools and/or automatic differentiation might be necessary to calculate the Lie derivatives. To sum up: the polynomialization step can be easily accomplished in certain (rather simple and white-box) examples, while it might be more difficult to apply and automate for models stemming from (commercial) FE codes.

## 4.2 Transformation to special polynomial system classes

In Section 4.1 we have discussed how to obtain a *general* polynomial system (4.1) from the input-affine representation (4.2). Depending on the dynamics, one might directly end up with a special polynomial class, e.g. a *bilinear* system with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}_1 \mathbf{x} + \mathbf{B}\mathbf{u} + \sum_{j=1}^m \mathbf{B}_{1,j} \mathbf{x} u_j$  or a *quadratic-bilinear* system with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}_1 \mathbf{x} + \mathbf{A}_2(\mathbf{x} \otimes \mathbf{x}) + \mathbf{B}\mathbf{u} + \sum_{j=1}^m \mathbf{B}_{1,j} \mathbf{x} u_j$ . If, however, a higher-order polynomial system is obtained, then two ways to proceed are possible. On the one hand, one could continue with steps 4.3 and 4.4 to analyze the general polynomial system in time- and frequency-domain. On the other hand, we can transform the system to an easier bilinear or quadratic-bilinear form. The former approach does not require a (further) transformation, but generally involves more complex system-theoretic analysis due to the more general polynomial structure. In contrast, the transformation to a special system class leads to an increased dimension, but simplifies the subsequent analysis.

In the following, we briefly revisit the Carleman bilinearization (cf. Section 4.2.1) and quadratic-bilinearization (cf. Section 4.2.2) processes to obtain a bilinear and quadratic-bilinear system, respectively.

### 4.2.1 Carleman bilinearization

Some physical phenomena (e.g. biological, chemical and stochastic problems) can be directly described by a bilinear system. In other cases, higher-order terms of the Taylor series expansion (4.6) need to be considered for a good representation. The idea of the Carleman

bilinearization is to transform the polynomial system (4.6) into a bilinear form by defining a new state vector

$$\mathbf{x}^\otimes = \begin{bmatrix} \mathbf{x} \\ \mathbf{x} \otimes \mathbf{x} \\ \vdots \\ \mathbf{x}^{(N)} \end{bmatrix} \in \mathbb{R}^{n+n^2+\dots+n^N}, \quad (4.9)$$

whose dimension depends on the truncation index  $N$ . Developing a differential equation for each  $\mathbf{x}^{(i)}$  and putting them together yields the *bilinearized system* [145, 221]

$$\mathbf{E}^\otimes \dot{\mathbf{x}}^\otimes = \mathbf{A}^\otimes \mathbf{x}^\otimes + \sum_{j=1}^m \mathbf{N}_j^\otimes \mathbf{x}^\otimes u_j + \mathbf{B}^\otimes \mathbf{u}, \quad \mathbf{x}^\otimes(0) = \mathbf{x}_0^\otimes, \quad (4.10a)$$

$$\mathbf{y} = \mathbf{C}^\otimes \mathbf{x}^\otimes, \quad (4.10b)$$

where

$$\mathbf{E}^\otimes = \begin{bmatrix} \mathbf{E} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{E}^{(2)} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{E}^{(N)} \end{bmatrix}, \quad \mathbf{A}^\otimes = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \dots & \mathbf{A}_{1,N} \\ \mathbf{0} & \mathbf{A}_{2,1} & \dots & \mathbf{A}_{2,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{A}_{N,1} \end{bmatrix},$$

$$\mathbf{N}_j^\otimes = \begin{bmatrix} \mathbf{B}_{1,1,j} & \mathbf{B}_{1,2,j} & \dots & \mathbf{B}_{1,N-1,j} & \mathbf{0} \\ \mathbf{B}_{2,0,j} & \mathbf{B}_{2,1,j} & \dots & \mathbf{B}_{2,N-2,j} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{3,0,j} & \dots & \mathbf{B}_{3,N-3,j} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{B}_{N,0,j} & \mathbf{0} \end{bmatrix}, \quad \mathbf{B}^\otimes = \begin{bmatrix} \mathbf{B}_{1,0,1} & \dots & \mathbf{B}_{1,0,m} \\ \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}, \quad \mathbf{x}_0^\otimes = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix},$$

$$\mathbf{C}^\otimes = [\mathbf{C}_1 \quad \mathbf{C}_2 \quad \dots \quad \mathbf{C}_N].$$

Herein,  $\mathbf{E}^{(i)} = \mathbf{E} \otimes \dots \otimes \mathbf{E} \in \mathbb{R}^{n^i \times n^i}$  denotes the  $i$ -fold Kronecker product. For  $i=1$ , it holds  $\mathbf{A}_{1,k} = \mathbf{A}_k$  and  $\mathbf{B}_{1,k,j} = \mathbf{B}_{k,j}$ , while for  $i \geq 2$  it holds

$$\mathbf{A}_{i,k} = \mathbf{A}_k \otimes \mathbf{E} \otimes \dots \otimes \mathbf{E} + \mathbf{E} \otimes \mathbf{A}_k \otimes \mathbf{E} \otimes \dots \otimes \mathbf{E} + \dots + \mathbf{E} \otimes \dots \otimes \mathbf{E} \otimes \mathbf{A}_k,$$

$$\mathbf{B}_{i,k,j} = \mathbf{B}_{k,j} \otimes \mathbf{E} \otimes \dots \otimes \mathbf{E} + \mathbf{E} \otimes \mathbf{B}_{k,j} \otimes \mathbf{E} \otimes \dots \otimes \mathbf{E} + \dots + \mathbf{E} \otimes \dots \otimes \mathbf{E} \otimes \mathbf{B}_{k,j}.$$

Note that the term  $\mathbf{B}^\otimes \mathbf{u}$  in (4.10a) can also be written as  $\mathbf{B}^\otimes \mathbf{u} = \sum_{j=1}^m \mathbf{b}_j^\otimes u_j$ , where the column vector  $\mathbf{b}_j^\otimes$  corresponds to the  $j$ -th column of the matrix  $\mathbf{B}^\otimes$ , i.e.  $\mathbf{b}_j^\otimes = \mathbf{B}^\otimes(:, j)$ .

Some examples for the Carleman bilinearization include: the RC-Ladder ( $n \rightarrow n + n^2$ ) in [47], the Burgers equation ( $n \rightarrow n + n^2$ ) in [25], as well as the FitzHugh-Nagumo ( $2n \rightarrow 2n + (2n)^2 + (2n)^3$ ) and nonlinear heat transfer model ( $n \rightarrow n + n^2$ ) in [92, Sec. 2.5].

**Discussion** With the Carleman bilinearization one can consider higher-order terms of the Taylor series expansion, while obtaining a bilinear system with Kronecker structure. This has the advantage that a better approximation of the nonlinear input-affine system (4.2) can be achieved, while at the same time the subsequent system-theoretic analysis becomes easier. However, the system dimension is tremendously increased from  $n$  to  $n + n^2 + \dots + n^N$ . If the

original order  $n$  is already large and/or the truncation index  $N$  needs to be big (e.g.  $N > 2$ ), then the resulting dimension might cause problems with the memory limitation. Thus, the applicability of this method highly depends on the system at hand.

### 4.2.2 Quadratic-bilinearization

Some nonlinear systems are inherent quadratic-bilinear (e.g. Burgers equation) or can be directly transformed into this special system class after the polynomialization procedure described in Section 4.1.2. In general, however, the polynomialization step yields a polynomial system that needs to be further transformed into a quadratic-bilinear form.

The main idea of the quadratic-bilinearization step is again to introduce new state variables to replace the monomials  $x^M$  (e.g.  $x^3, x^4$ ) with quadratic expressions. According to [110], this can be done in two ways:

1. quadratic-bilinearization by adding *quadratic* algebraic equations [109, Sec. 6.3.5]
2. quadratic-bilinearization by taking Lie derivatives [109, Sec. 6.3.6]

In certain situations, it might be more advantageous to add quadratic algebraic equations than taking Lie derivatives, since the latter approach yields a higher number of ODEs. In most cases, both approaches have to be combined to *iteratively* simplify a polynomial system to a quadratic-bilinear DAE (QBDAE) with least increased dimension and differential index.

We refrain again from the theoretical details and instead provide an example.

*Example 4.2* (Quadratic-bilinearization procedure). Let us consider a similar circuit as in Example 4.1, which is now composed of an additional nonlinear resistor with *cubic* voltage-current (V-I) characteristic (cf. Chua's circuit), leading to:

$$\dot{v} = \frac{1}{C} \left( -\frac{v}{R} - e^{\alpha v} + 1 + v^3 + i \right). \quad (4.11)$$

As before, the new variable  $w = e^{\alpha v} - 1$  is introduced to handle the exponential function. Unfortunately, the introduction of  $z = v^2$  (with  $v^3 \rightarrow vz$ ) cannot lead to a quadratic-bilinear system due to  $\dot{w} = \alpha(w+1)\dot{v}$ , which still depends on the cubic term  $wvz$ . Thus, we instead introduce the new variable  $z = v^3$ , yielding:

$$\dot{v} = \frac{1}{C} \left( -\frac{v}{R} - w + z + i \right), \quad \dot{w} = \frac{\alpha}{C} \left( -\frac{vw}{R} - w^2 + wz + wi - \frac{v}{R} - w + z + i \right), \quad (4.12)$$

with  $\dot{z} = 3v^2\dot{v}$ . We have to introduce another variable  $y = v^2$ , leading to

$$\dot{z} = 3y\dot{v} = \frac{3}{C} \left( -\frac{vy}{R} - wy + zy + yi \right). \quad (4.13)$$

The equation  $y = v^2$  can be handled as algebraic constraint, i.e.  $0 = y - v^2$ , or by taking its Lie derivative, i.e.  $\dot{y} = 2v\dot{v}$ . Using the state vector  $\mathbf{x} = [v, w, z, y]^T$  and the input signal  $u = i$ , the scalar differential equation (4.11) can be rewritten as a quadratic-bilinear system of the form  $\mathbf{E}\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{H}(\mathbf{x} \otimes \mathbf{x}) + \mathbf{N}\mathbf{x}u + \mathbf{b}u$ , with corresponding matrices. In case of  $0 = y - v^2$ , one obtains a QBDAE with 3 differential and 1 algebraic equation, i.e.  $\det(\mathbf{E}) = 0$ . In case of  $\dot{y} = 2v\dot{v}$ , one obtains a QB system consisting of 4 ODEs.  $\triangle$

Other examples for quadratic-bilinearization are given in the literature: the RC-Ladder model ( $n \rightarrow 2n$ ) in [110], as well as the Chafee-Infante ( $n \rightarrow 2n$ ) and FitzHugh-Nagumo equation ( $2n \rightarrow 3n$ ) in [22]. A detailed explanation can also be found in the master thesis [Fio16].

**Discussion** As discussed before, the polynomialization+quadratic-bilinearization procedures allow to transform a nonlinear system into an *equivalent* quadratic-bilinear form, without performing any kind of approximation. Although the dimension is increased during the whole process, the growth is modest in comparison to the Carleman bilinearization (cf.  $n \rightarrow 2n$  vs.  $n \rightarrow n+n^2$  for the RC-Ladder). This is a remarkable advantage. Nevertheless, the requirement of the analytical expressions for the nonlinearities — and possibly of symbolic/automatic differentiation tools — makes the method intrusive and less likely applicable to complex FE models. However, the applicability and suitability of all mentioned methods (Taylor series expansion, polynomialization, Carleman bilinearization, quadratic-bilinearization) has to be investigated for each particular case and system at hand.

### 4.3 Volterra series representation

In this section, we analyze polynomial systems in time-domain using the Volterra series representation. The *Volterra series* represents the solution of a nonlinear dynamical system as an infinite sum  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \mathbf{x}_k(t)$  of subresponses  $\mathbf{x}_k(t)$ . The latter are described by multivariable convolution integrals, but can also be interpreted as the response  $\mathbf{x}_k(t)$  of a  $k$ -th homogeneous subsystem. Consequently, there are two ways of gaining a Volterra series representation:

1. by applying the Picard iteration
2. by applying the variational equation approach

We will revisit both approaches in the following, and primarily apply them to bilinear

$$\mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}(t) u_j(t) + \mathbf{B} \mathbf{u}(t), \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (4.14)$$

and quadratic-bilinear systems

$$\mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t)) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}(t) u_j(t) + \mathbf{B} \mathbf{u}(t), \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (4.15)$$

with  $\mathbf{y}(t) = \mathbf{C} \mathbf{x}(t)$ . Similar considerations hold also for more general polynomial systems. Note that the MIMO terms can be written in a different way:  $\sum_{j=1}^m \mathbf{N}_j \mathbf{x}(t) u_j(t) = \bar{\mathbf{N}}(\mathbf{u}(t) \otimes \mathbf{x}(t))$  with  $\bar{\mathbf{N}} = [\mathbf{N}_1 \cdots \mathbf{N}_m] \in \mathbb{R}^{n \times n \cdot m}$  and  $\mathbf{B} \mathbf{u}(t) = \sum_{j=1}^m \mathbf{b}_j u_j(t)$ .

#### 4.3.1 Picard iteration

The Picard fixed-point iteration allows to approximate the solution of an initial value problem via successive approximations. According to the Picard-Lindelöf theorem [269], the initial value problems (4.14) and (4.15) have a *unique* solution  $\mathbf{x}(t)$  on the time interval  $[0, T]$  for inputs  $u_j(t)$  that are bounded, i.e.  $|u_j(t)| < U$  with  $U > 0$ , and continuous on  $[0, T]$ . Assuming bounded inputs, we apply the Picard iteration to the bilinear system (4.14) in the following. For more details, the reader is referred to [27], [92, Sec. 2.1] and [Röt18].

### Bilinear systems

We assume that  $\mathbf{E}$  is non-singular and define  $\tilde{\mathbf{A}} := \mathbf{E}^{-1}\mathbf{A}$ ,  $\tilde{\mathbf{N}}_j := \mathbf{E}^{-1}\mathbf{N}_j$  and  $\tilde{\mathbf{B}} := \mathbf{E}^{-1}\mathbf{B}$ . Moreover, we consider the MIMO case and employ the sum notation (cf. [27]) rather than the cumbersome and less insightful Kronecker notation (cf. [92]).

To apply the Picard iteration, a change of variable is first employed:

$$\mathbf{x}(t) = e^{\tilde{\mathbf{A}}t}\mathbf{z}(t) \Leftrightarrow \mathbf{z}(t) = e^{-\tilde{\mathbf{A}}t}\mathbf{x}(t) \quad \text{with} \quad \dot{\mathbf{x}}(t) = e^{\tilde{\mathbf{A}}t}\dot{\mathbf{z}}(t) + \tilde{\mathbf{A}}e^{\tilde{\mathbf{A}}t}\mathbf{z}(t). \quad (4.16)$$

Inserting (4.16) into  $\dot{\mathbf{x}}(t) = \tilde{\mathbf{A}}\mathbf{x}(t) + \sum_{j=1}^m \tilde{\mathbf{N}}_j \mathbf{x}(t) u_j(t) + \tilde{\mathbf{B}}\mathbf{u}(t)$  and dissolving for  $\dot{\mathbf{z}}(t)$  yields

$$\dot{\mathbf{z}}(t) = \sum_{j=1}^m \hat{\mathbf{N}}_j(t)\mathbf{z}(t)u_j(t) + \hat{\mathbf{B}}(t)\mathbf{u}(t), \quad \mathbf{z}(0) = \mathbf{x}_0, \quad (4.17)$$

where  $\hat{\mathbf{N}}_j(t) = e^{-\tilde{\mathbf{A}}t}\tilde{\mathbf{N}}_j e^{\tilde{\mathbf{A}}t}$ ,  $\hat{\mathbf{B}}(t) = e^{-\tilde{\mathbf{A}}t}\tilde{\mathbf{B}}$ . Integrating this *time-varying* bilinear system and assuming that  $\mathbf{x}_0 = \mathbf{0}$ , we can write  $\mathbf{z}(t)$  as follows

$$\mathbf{z}(t) = \sum_{j=1}^m \int_{\tau_1=0}^t \hat{\mathbf{N}}_j(\tau_1)\mathbf{z}(\tau_1)u_j(\tau_1)d\tau_1 + \sum_{j=1}^m \int_{\tau_1=0}^t \hat{\mathbf{b}}_j(\tau_1)u_j(\tau_1)d\tau_1. \quad (4.18)$$

Since we need  $\mathbf{z}(\tau_1)$ , let us rewrite the above equation with  $t \rightarrow \tau_k$ ,  $\tau_1 \rightarrow \tau_{k+1}$ ,  $j \rightarrow j_{k+1}$  as:

$$\mathbf{z}(\tau_k) = \sum_{j=1}^m \int_{\tau_{k+1}=0}^{\tau_k} \hat{\mathbf{N}}_j(\tau_{k+1})\mathbf{z}(\tau_{k+1})u_j(\tau_{k+1})d\tau_{k+1} + \sum_{j=1}^m \int_{\tau_{k+1}=0}^{\tau_k} \hat{\mathbf{b}}_j(\tau_{k+1})u_j(\tau_{k+1})d\tau_{k+1}.$$

Now, substituting  $\mathbf{z}(\tau_1)$  in (4.18) leads to (with  $j_1, j_2$ ):

$$\begin{aligned} \mathbf{z}(t) &= \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^t \int_{\tau_2=0}^{\tau_1} \hat{\mathbf{N}}_{j_1}(\tau_1)\hat{\mathbf{N}}_{j_2}(\tau_2)\mathbf{z}(\tau_2)u_{j_2}(\tau_2)u_{j_1}(\tau_1)d\tau_2 d\tau_1 \\ &+ \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^t \int_{\tau_2=0}^{\tau_1} \hat{\mathbf{N}}_{j_1}(\tau_1)\hat{\mathbf{b}}_{j_2}(\tau_2)u_{j_2}(\tau_2)u_{j_1}(\tau_1)d\tau_2 d\tau_1 + \sum_{j_1=1}^m \int_{\tau_1=0}^t \hat{\mathbf{b}}_{j_1}(\tau_1)u_{j_1}(\tau_1)d\tau_1. \end{aligned}$$

The same procedure is repeated with  $\mathbf{z}(\tau_2)$  (with  $j_1, j_2, j_3$ ). After  $N$  steps, we receive

$$\begin{aligned} \mathbf{z}(t) &= \sum_{j_1=1}^m \cdots \sum_{j_N=1}^m \int_{\tau_1=0}^t \cdots \int_{\tau_N=0}^{\tau_{N-1}} \hat{\mathbf{N}}_{j_1}(\tau_1) \cdots \hat{\mathbf{N}}_{j_N}(\tau_N)\mathbf{z}(\tau_N)u_{j_N}(\tau_N) \cdots u_{j_1}(\tau_1)d\tau_N \cdots d\tau_1 \\ &+ \sum_{k=1}^N \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=0}^t \cdots \int_{\tau_k=0}^{\tau_{k-1}} \hat{\mathbf{N}}_{j_1}(\tau_1) \cdots \hat{\mathbf{N}}_{j_{k-1}}(\tau_{k-1}) \\ &\quad \times \hat{\mathbf{b}}_{j_k}(\tau_k)u_{j_k}(\tau_k)u_{j_{k-1}}(\tau_{k-1}) \cdots u_{j_1}(\tau_1)d\tau_k \cdots d\tau_1. \end{aligned} \quad (4.19)$$

The first term in (4.19) still depends on  $\mathbf{z}(\tau_N)$ . Nevertheless, assuming that  $\hat{\mathbf{N}}_j(t)$ ,  $\mathbf{z}(t)$  and  $u_j(t)$  are bounded on  $t \in [0, T]$ , i.e.

$$\max_{1 \leq j \leq m} \sup_{0 \leq t \leq T} \|\hat{\mathbf{N}}_j(t)\| < L, \quad \sup_{0 \leq t \leq T} \|\mathbf{z}(t)\| < Z, \quad \max_{1 \leq j \leq m} \sup_{0 \leq t \leq T} \|u_j(t)\| < U, \quad (4.20)$$

the term can be bounded by  $\frac{(LUT)^N}{N!}Z$  and thus vanishes as  $N \rightarrow \infty$ . Changing back to the

original variables finally yields the uniformly convergent<sup>2</sup> Volterra series representation

$$\begin{aligned} \mathbf{x}(t) = & \sum_{k=1}^{\infty} \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=0}^t \cdots \int_{\tau_k=0}^{\tau_{k-1}} e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{N}}_{j_1} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{N}}_{j_2} \cdots e^{\tilde{\mathbf{A}}(\tau_{k-2}-\tau_{k-1})} \tilde{\mathbf{N}}_{j_{k-1}} \\ & \times e^{\tilde{\mathbf{A}}(\tau_{k-1}-\tau_k)} \tilde{\mathbf{b}}_{j_k} u_{j_k}(\tau_k) u_{j_{k-1}}(\tau_{k-1}) \cdots u_{j_1}(\tau_1) d\tau_k \cdots d\tau_1, \end{aligned} \quad (4.21)$$

where  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \mathbf{x}_k(t)$  with corresponding  $\mathbf{x}_k(t)$ .

### 4.3.2 Variational equation approach

The Picard iteration allows to find a Volterra series representation in terms of convolutional integrals and provides insight into the *convergence* of the series. Another way of gaining a Volterra series representation is given by the variational equation approach, where the nonlinear system is interpreted as an infinite sequence of homogeneous, cascaded subsystems. The solution  $\mathbf{x}(t)$  is then given by the sum over all sub-solutions  $\mathbf{x}_k(t)$  of each subsystem. In order to obtain a state equation for each  $k$ -th degree subsystem, we proceed as follows [221].

First, it is assumed that the response of the system to an input of the form  $\alpha \mathbf{u}(t)$  is given by the power series expansion (aka. *asymptotic/Poincaré/naïve expansion* [187, 264, 136]):

$$\mathbf{x}(t) = \sum_{k=1}^{\infty} \alpha^k \mathbf{x}_k(t) = \alpha \mathbf{x}_1(t) + \alpha^2 \mathbf{x}_2(t) + \alpha^3 \mathbf{x}_3(t) + \dots \quad (4.22)$$

Then, this ansatz is inserted into the system at hand, and coefficients of like powers of  $\alpha$  are equated to obtain the subsystem state equations (aka. *variational equations*). This is demonstrated in the following for bilinear and quadratic-bilinear systems.

#### Bilinear systems

Inserting the assumed input  $\alpha \mathbf{u}(t)$  and ansatz (4.22) in the state equation (4.14) results in

$$\begin{aligned} \mathbf{E}(\alpha \dot{\mathbf{x}}_1(t) + \alpha^2 \dot{\mathbf{x}}_2(t) + \dots) &= \mathbf{A}(\alpha \mathbf{x}_1(t) + \alpha^2 \mathbf{x}_2(t) + \dots) \\ &+ \sum_{j=1}^m \mathbf{N}_j (\alpha \mathbf{x}_1(t) + \alpha^2 \mathbf{x}_2(t) + \dots) \alpha u_j(t) + \mathbf{B} \alpha \mathbf{u}(t). \end{aligned} \quad (4.23)$$

Since this differential equation must hold for all  $\alpha$ , terms of like powers of  $\alpha$  can be equated, yielding a state equation for each subsystem:

$$\begin{aligned} \alpha : \quad & \mathbf{E} \dot{\mathbf{x}}_1(t) = \mathbf{A} \mathbf{x}_1(t) + \mathbf{B} \mathbf{u}(t), & \mathbf{x}_1(0) &= \mathbf{x}_0, \\ \alpha^2 : \quad & \mathbf{E} \dot{\mathbf{x}}_2(t) = \mathbf{A} \mathbf{x}_2(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_1(t) u_j(t), & \mathbf{x}_2(0) &= \mathbf{0}, \\ \alpha^3 : \quad & \mathbf{E} \dot{\mathbf{x}}_3(t) = \mathbf{A} \mathbf{x}_3(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_2(t) u_j(t), & \mathbf{x}_3(0) &= \mathbf{0}, \\ & \vdots & & \end{aligned}$$

<sup>2</sup>The bound  $\frac{(LUT)^N}{N!} Z$  or rather the inequality  $N > LUT$  provides a relation between the input, time interval and amount of considered subsystems to ensure convergence of the series. In other words: if the applied input or time interval gets bigger, then we have to consider more subsystems, i.e. increase  $N$ .

or in generalized form:

$$\mathbf{E} \dot{\mathbf{x}}_1(t) = \mathbf{A} \mathbf{x}_1(t) + \mathbf{B} \mathbf{u}(t), \quad \mathbf{x}_1(0) = \mathbf{x}_0, \quad (4.25a)$$

$$\mathbf{E} \dot{\mathbf{x}}_k(t) = \mathbf{A} \mathbf{x}_k(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_{k-1}(t) u_j(t), \quad \mathbf{x}_k(0) = \mathbf{0}, \quad k \geq 2. \quad (4.25b)$$

Note that each  $k$ -th subsystem state equation is *linear* in  $\mathbf{x}_k(t)$ , but (4.25b) depends *bilinearly* on the solution  $\mathbf{x}_{k-1}(t)$  of the preceding subsystem (cf. Fig. 4.1).

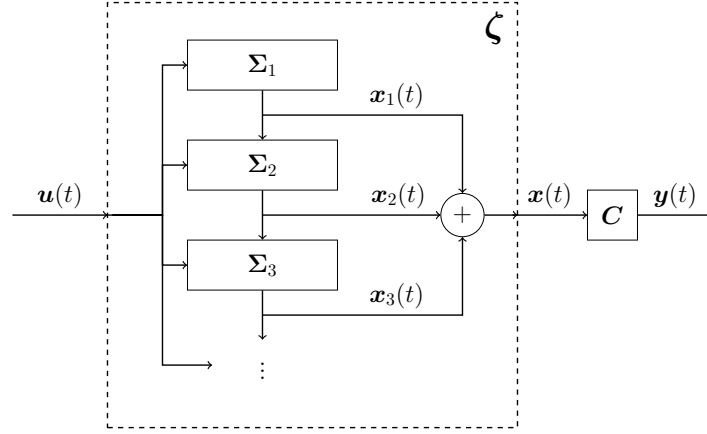


Figure 4.1: Volterra series representation of a bilinear system  $\zeta$ .

The solutions to these differential equations are given (for  $\mathbf{x}_0 = \mathbf{0}$ ) by:

$$\mathbf{x}_1(t) = \int_{\tau=0}^t e^{\tilde{\mathbf{A}}(t-\tau)} \tilde{\mathbf{B}} \mathbf{u}(\tau) d\tau, \quad (4.26a)$$

$$\mathbf{x}_k(t) = \sum_{j=1}^m \int_{\tau=0}^t e^{\tilde{\mathbf{A}}(t-\tau)} \tilde{\mathbf{N}}_j \mathbf{x}_{k-1}(\tau) u_j(\tau) d\tau, \quad k \geq 2. \quad (4.26b)$$

Any  $\mathbf{x}_k(t)$  can then be obtained by iteratively substituting all previous solutions  $\mathbf{x}_{k-1}(t)$  into (4.26b). For instance, substituting  $\mathbf{x}_1(\tau_1)$

$$\mathbf{x}_1(\tau_1) = \sum_{j=1}^m \int_{\tau_2=0}^{\tau_1} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_j u_j(\tau_2) d\tau_2 \quad (4.27)$$

into  $\mathbf{x}_2(t)$  yields<sup>3</sup>

$$\begin{aligned} \mathbf{x}_2(t) &= \sum_{j_1=1}^m \int_{\tau_1=0}^t e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{N}}_{j_1} \mathbf{x}_1(\tau_1) u_{j_1}(\tau_1) d\tau_1 \\ &= \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^t \int_{\tau_2=0}^{\tau_1} e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{N}}_{j_1} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_{j_2} u_{j_2}(\tau_2) u_{j_1}(\tau_1) d\tau_2 d\tau_1. \end{aligned} \quad (4.28)$$

<sup>3</sup>(4.26a) can also be written as  $\mathbf{x}_1(t) = \int_{\tau=0}^t e^{\tilde{\mathbf{A}}\tau} \tilde{\mathbf{B}} \mathbf{u}(t-\tau) d\tau$  with  $\mathbf{x}_1(t-\tau_2) = \sum_{j=1}^m \int_{\tau_1=0}^{t-\tau_2} e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{b}}_j u_j(t-\tau_2-\tau_1) d\tau_1$ . Then,  $\mathbf{x}_1(t-\tau_2)$  can be substituted in  $\mathbf{x}_2(t) = \sum_{j_2=1}^m \int_{\tau_2=0}^t e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{N}}_{j_2} \mathbf{x}_1(t-\tau_2) u_{j_2}(t-\tau_2) d\tau_2$  (cf. Eq. (4.38)).

The same can be repeated with  $\mathbf{x}_3(t)$  by substituting  $\mathbf{x}_2(\tau_1)$  and  $\mathbf{x}_1(\tau_2)$ . After  $k$  substitutions, we end up with the solution  $\mathbf{x}_k(t)$  of the  $k$ -th subsystem that corresponds to the Volterra series representation obtained in (4.21). Note that in case of an arbitrary initial state  $\mathbf{x}_0$ , the solution  $\mathbf{x}_1(t)$  also contains the term  $e^{\tilde{\mathbf{A}}t} \mathbf{x}_0$  (cf. (3.2)), and thus the Volterra series representation (4.21) is also composed of an initial condition term. [60]

### Quadratic-bilinear systems

Inserting the assumed input  $\alpha \mathbf{u}(t)$  and ansatz (4.22) in the state equation (4.15) results in

$$\begin{aligned} \mathbf{E}(\alpha \dot{\mathbf{x}}_1 + \alpha^2 \dot{\mathbf{x}}_2 + \alpha^3 \dot{\mathbf{x}}_3 \dots) &= \mathbf{A}(\alpha \mathbf{x}_1 + \alpha^2 \mathbf{x}_2 + \alpha^3 \mathbf{x}_3 + \dots) \\ &+ \mathbf{H}(\alpha \mathbf{x}_1 + \alpha^2 \mathbf{x}_2 + \alpha^3 \mathbf{x}_3 + \dots) \otimes (\alpha \mathbf{x}_1 + \alpha^2 \mathbf{x}_2 + \alpha^3 \mathbf{x}_3 + \dots) \\ &+ \sum_{j=1}^m \mathbf{N}_j (\alpha \mathbf{x}_1 + \alpha^2 \mathbf{x}_2 + \alpha^3 \mathbf{x}_3 + \dots) \alpha u_j + \mathbf{B} \alpha \mathbf{u}. \end{aligned} \quad (4.29)$$

Comparing terms with equal powers of  $\alpha$  yields a state equation for each subsystem:

$$\begin{aligned} \alpha : \quad \mathbf{E} \dot{\mathbf{x}}_1(t) &= \mathbf{A} \mathbf{x}_1(t) + \mathbf{B} \mathbf{u}(t), & \mathbf{x}_1(0) &= \mathbf{x}_0, \\ \alpha^2 : \quad \mathbf{E} \dot{\mathbf{x}}_2(t) &= \mathbf{A} \mathbf{x}_2(t) + \mathbf{H}(\mathbf{x}_1(t) \otimes \mathbf{x}_1(t)) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_1(t) u_j(t), & \mathbf{x}_2(0) &= \mathbf{0}, \\ \alpha^3 : \quad \mathbf{E} \dot{\mathbf{x}}_3(t) &= \mathbf{A} \mathbf{x}_3(t) + \mathbf{H}(\mathbf{x}_1(t) \otimes \mathbf{x}_2(t) + \mathbf{x}_2(t) \otimes \mathbf{x}_1(t)) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_2(t) u_j(t), & \mathbf{x}_3(0) &= \mathbf{0}, \\ & \vdots \end{aligned}$$

or in generalized form:

$$\mathbf{E} \dot{\mathbf{x}}_1(t) = \mathbf{A} \mathbf{x}_1(t) + \mathbf{B} \mathbf{u}(t), \quad \mathbf{x}_1(0) = \mathbf{x}_0, \quad (4.30a)$$

$$\mathbf{E} \dot{\mathbf{x}}_k(t) = \mathbf{A} \mathbf{x}_k(t) + \sum_{i=1}^{k-1} \mathbf{H}(\mathbf{x}_i(t) \otimes \mathbf{x}_{k-i}(t)) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_{k-1}(t) u_j(t), \quad \mathbf{x}_k(0) = \mathbf{0}. \quad (4.30b)$$

Note that each  $k$ -th subsystem state equation is *linear* in  $\mathbf{x}_k(t)$ , but (4.30b) depends *nonlinearly* on the solution of the preceding subsystems.

The solution to the first subsystem differential equation is given (for  $\mathbf{x}_0 = \mathbf{0}$ ) by (4.26a). However, it is difficult and lengthy to write the solution  $\mathbf{x}_k(t)$  of (4.30b) in general form. Thus, we restrict the analysis to the second subsystem. Defining  $\tilde{\mathbf{H}} := \mathbf{E}^{-1} \mathbf{H}$  and substituting  $\mathbf{x}_1(\tau_1)$  (cf. Eq. (4.27)) in  $\mathbf{x}_2(t)$  yields (cf. [Fio16])

$$\begin{aligned} \mathbf{x}_2(t) &= \int_{\tau_1=0}^t e^{\tilde{\mathbf{A}}(t-\tau_1)} \left( \tilde{\mathbf{H}}(\mathbf{x}_1(\tau_1) \otimes \mathbf{x}_1(\tau_1)) + \sum_{j=1}^m \tilde{\mathbf{N}}_j \mathbf{x}_1(\tau_1) u_j(\tau_1) \right) d\tau_1 \\ &= \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^t \int_{\tau_2=0}^{\tau_1} \int_{\tau_3=0}^{\tau_2} e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{H}}(e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_{j_1} \otimes e^{\tilde{\mathbf{A}}(\tau_1-\tau_3)} \tilde{\mathbf{b}}_{j_2}) u_{j_1}(\tau_2) u_{j_2}(\tau_3) d\tau_1 d\tau_2 d\tau_3 \\ &+ \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^t \int_{\tau_2=0}^{\tau_1} e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{N}}_{j_1} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_{j_2} u_{j_2}(\tau_2) u_{j_1}(\tau_1) d\tau_1 d\tau_2. \end{aligned} \quad (4.31)$$



## 4.4 Generalized transfer functions

In the previous section we have discussed how to obtain a Volterra series representation for bilinear and quadratic-bilinear systems. In this section we want to report different types of Volterra kernels in time-domain and derive generalized transfer functions in frequency-domain. There are two approaches to obtain generalized transfer functions:

1. by multidimensional Laplace transform of the kernels
2. by the growing exponential approach

### 4.4.1 Multidimensional Laplace transform of the kernels and output equations

The first approach to obtain generalized transfer functions consists in transforming the Volterra kernels into frequency-domain. Thus, similar to the single-variable Laplace transform (3.5) required in the linear setting, we revisit the multivariable Laplace transform in the following definition [221].

**Definition 4.1** (Multidimensional Laplace transform). The  $k$ -dimensional Laplace transform of a one-sided, real-valued function  $f(t_1, \dots, t_k) : \mathbb{R}_{\geq 0}^k \rightarrow \mathbb{R}$  is given by

$$\begin{aligned} F(s_1, \dots, s_k) &:= \mathcal{L}_k \{f(t_1, \dots, t_k)\}(s_1, \dots, s_k) \\ &:= \int_{t_1=0}^{\infty} \dots \int_{t_k=0}^{\infty} f(t_1, \dots, t_k) e^{-s_1 t_1} \dots e^{-s_k t_k} dt_k \dots dt_1. \end{aligned} \quad (4.32)$$

The integral converges, if the complex variables  $s_1, \dots, s_k$  are such that

$$\mathbf{s} = (s_1 \ \dots \ s_k)^\top \in H_{\gamma_1, \dots, \gamma_k} := H_\gamma = \left\{ \mathbf{s} \in \mathbb{C}^k \mid \operatorname{Re}(s_i) > \gamma_i, \ i = 1, \dots, k \right\},$$

i.e. for values  $\mathbf{s} \in \mathbb{C}^k$  on the  $k$ -dimensional complex half-space  $H_\gamma$  of  $\mathbb{C}^k$ . ▲

In what follows, we will first discuss different types of Volterra kernels arising from the input-output representation in time-domain obtained in Section 4.3. Then, we will transform the kernels and output equations in frequency-domain to obtain generalized transfer functions.

### Bilinear systems

The Volterra series representation gained in (4.21) from the Picard iteration (or alternatively (4.28) from the variational analysis) is the starting point for the subsequent analysis.

**Triangular kernels** Unlike [221, 92], we are not defining the triangular kernels directly from (4.21) or the corresponding output

$$\begin{aligned} \mathbf{y}_k(t) &= \sum_{j_1=1}^m \dots \sum_{j_k=1}^m \int_{\tau_1=0}^t \dots \int_{\tau_k=0}^{\tau_{k-1}} \mathbf{C} e^{\tilde{\mathbf{A}}(t-\tau_1)} \tilde{\mathbf{N}}_{j_1} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{N}}_{j_2} \dots e^{\tilde{\mathbf{A}}(\tau_{k-2}-\tau_{k-1})} \tilde{\mathbf{N}}_{j_{k-1}} e^{\tilde{\mathbf{A}}(\tau_{k-1}-\tau_k)} \tilde{\mathbf{b}}_{j_k} \\ &\quad \times u_{j_k}(\tau_k) u_{j_{k-1}}(\tau_{k-1}) \dots u_{j_1}(\tau_1) d\tau_k \dots d\tau_1. \end{aligned}$$

Instead, we perform a change of variables to gain an input-output representation without reflected arguments  $-\tau_1, \dots, -\tau_k$  in the exponential functions. To simplify the change of variables, we assume one-sided input signals  $u_j(t) := u_j(t)\sigma(t)$  and one-sided matrix exponentials

$e^{\tilde{\mathbf{A}}t} := e^{\tilde{\mathbf{A}}t} \sigma(t)$ , and use infinite integration limits. Making the change of variables  $\tilde{\tau}_k = t - \tau_1$ ,  $\tilde{\tau}_{k-1} = t - \tau_2, \dots, \tilde{\tau}_1 = t - \tau_k$  with  $\tau_1 - \tau_2 = \tilde{\tau}_{k-1} - \tilde{\tau}_k, \dots, \tau_{k-1} - \tau_k = \tilde{\tau}_1 - \tilde{\tau}_2$  and letting again  $\tilde{\tau}_k \rightarrow \tau_k, \dots, \tilde{\tau}_1 \rightarrow \tau_1$  yields the input-output representation (with interchanged  $j$  indices):

$$\mathbf{y}_k(t) = \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=-\infty}^{\infty} \cdots \int_{\tau_k=-\infty}^{\infty} \mathbf{C} e^{\tilde{\mathbf{A}}\tau_k} \tilde{\mathbf{N}}_{j_k} e^{\tilde{\mathbf{A}}(\tau_{k-1}-\tau_k)} \tilde{\mathbf{N}}_{j_{k-1}} \cdots \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_{j_1} \times u_{j_k}(t - \tau_k) u_{j_{k-1}}(t - \tau_{k-1}) \cdots u_{j_2}(t - \tau_2) u_{j_1}(t - \tau_1) d\tau_k \cdots d\tau_1, \quad (4.33)$$

where the outputs are

$$\mathbf{y}(t) = \sum_{k=1}^{\infty} \mathbf{y}_k(t), \quad \mathbf{y}_k(t) = \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \mathbf{y}_k^{(j_1, \dots, j_k)}(t), \quad (4.34)$$

and the *triangular* MIMO kernels are given by (for  $0 < t_k < \dots < t_1$ ):

$$\mathbf{g}_{k,\Delta}^{(j_1, \dots, j_k)}(t_1, \dots, t_k) = \mathbf{C} e^{\tilde{\mathbf{A}}t_k} \tilde{\mathbf{N}}_{j_k} e^{\tilde{\mathbf{A}}(t_{k-1}-t_k)} \tilde{\mathbf{N}}_{j_{k-1}} \cdots \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}(t_1-t_2)} \tilde{\mathbf{b}}_{j_1}. \quad (4.35)$$

Applying the  $k$ -dimensional Laplace transform to the triangular kernels  $\mathbf{g}_{k,\Delta}^{(j_1, \dots, j_k)}(t_1, \dots, t_k)$  yields the  $(j_1, \dots, j_k)$ -th transfer function  $\mathbf{G}_{k,\Delta}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) \in \mathbb{C}^p$  (cf. derivation in [60]):

$$\mathbf{G}_{k,\Delta}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) = \mathbf{C} ((s_1 + \dots + s_k) \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \cdots \mathbf{N}_{j_3} ((s_1 + s_2) \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}_{j_1}. \quad (4.36)$$

Note that the triangular transfer functions yield factors  $((s_1 + \dots + s_k) \mathbf{E} - \mathbf{A})^{-1}$ , where the frequency variables  $s_1, \dots, s_k$  are summed up. In other words, each singularity term is not expressed by means of a single variable  $s_k$ , but rather by *several* variables.

Instead of merely transforming the kernels, the output equation (4.33) should be rather transformed in order to obtain an input-output representation in frequency-domain similar to the one from the linear case (3.6). To this end, a triangular *auxiliary* output  $\mathbf{y}_k^\Delta(t_1, \dots, t_k)$  depending on the time variables  $t_1, \dots, t_k$  is first introduced and then transformed into frequency-domain using the convolution property of the multidimensional Laplace transform [60]:

$$\mathbf{Y}_k^\Delta(s_1, \dots, s_k) = \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \mathbf{Y}_{k,\Delta}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) \quad (4.37)$$

$$= \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \underbrace{\mathbf{C} ((s_1 + \dots + s_k) \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \cdots \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}_{j_1}}_{\mathbf{G}_{k,\Delta}^{(j_1, \dots, j_k)}(s_1, \dots, s_k)} U_{j_k}(s_k) \cdots U_{j_1}(s_1).$$

This triangular frequency-domain representation is important for the following reason. If the transfer functions  $\mathbf{G}_{k,\Delta}^{(j_1, \dots, j_k)}(s_1, \dots, s_k)$  and the Laplace transform of the inputs are known, then  $\mathbf{Y}_k^\Delta(s_1, \dots, s_k)$  can be calculated *algebraically*. After that,  $\mathbf{y}_k^\Delta(t_1, \dots, t_k)$  can be computed via the multivariable *inverse* Laplace transform of  $\mathbf{Y}_k^\Delta(s_1, \dots, s_k)$ . The actual (single-variable) output of the bilinear system is finally calculated from  $\mathbf{y}_k(t) = \mathbf{y}_k^\Delta(t_1 = t, \dots, t_k = t)$ .

**Regular kernels** To simplify the terms  $e^{\tilde{\mathbf{A}}(\tau_{k-1}-\tau_k)}$  arising in the triangular representation (4.33), we now give the input-output behavior of the bilinear system in terms of the regular kernels. For this purpose, we perform the change of variables  $\tilde{\tau}_k = \tau_k$ ,  $\tilde{\tau}_{k-1} = \tau_{k-1} - \tau_k$ ,  $\dots$ ,  $\tilde{\tau}_1 = \tau_1 - \tau_2$  with  $\tau_{k-1} = \tilde{\tau}_k + \tilde{\tau}_{k-1}$ ,  $\dots$ ,  $\tau_1 = \tilde{\tau}_k + \dots + \tilde{\tau}_1$  and let again  $\tilde{\tau}_k \rightarrow \tau_k, \dots, \tilde{\tau}_1 \rightarrow \tau_1$ . Then, the regular input-output representation is given by [60]:

$$\mathbf{y}_k(t) = \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=-\infty}^{\infty} \cdots \int_{\tau_k=-\infty}^{\infty} \mathbf{C} e^{\tilde{\mathbf{A}}\tau_k} \tilde{\mathbf{N}}_{j_k} e^{\tilde{\mathbf{A}}\tau_{k-1}} \tilde{\mathbf{N}}_{j_{k-1}} \cdots \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{b}}_{j_1} \times u_{j_k}(t - \tau_k) u_{j_{k-1}}(t - \tau_k - \tau_{k-1}) \cdots u_{j_1}(t - \tau_k - \dots - \tau_1) d\tau_k \cdots d\tau_1, \quad (4.38)$$

with the output components (4.34) and the *regular* MIMO kernels (for  $t_1, \dots, t_k > 0$ ):

$$\mathbf{g}_{k,\square}^{(j_1, \dots, j_k)}(t_1, \dots, t_k) = \mathbf{C} e^{\tilde{\mathbf{A}}t_k} \tilde{\mathbf{N}}_{j_k} e^{\tilde{\mathbf{A}}t_{k-1}} \tilde{\mathbf{N}}_{j_{k-1}} \cdots \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}t_1} \tilde{\mathbf{b}}_{j_1}. \quad (4.39)$$

Applying the  $k$ -dimensional Laplace transform to the regular kernels  $\mathbf{g}_{k,\square}^{(j_1, \dots, j_k)}(t_1, \dots, t_k)$  yields the  $(j_1, \dots, j_k)$ -th transfer function  $\mathbf{G}_{k,\square}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) \in \mathbb{C}^p$  (cf. derivation in [60]):

$$\mathbf{G}_{k,\square}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) = \mathbf{C}(s_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \cdots \mathbf{N}_{j_3} (s_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}_{j_1}. \quad (4.40)$$

Note that the regular transfer functions are expressed – in contrast to the triangular ones – by a product of *single-variable* factors  $(s_k \mathbf{E} - \mathbf{A})^{-1}$ . This permits the straightforward application of the residue calculus and the inverse Laplace transform in each variable independently [221, pp. 59-60, 73]. For this reason, the triangular transfer functions (4.36) are rarely used as a starting point for model order reduction, but the regular transfer functions have established instead (cf. Chapter 5).

Similar as before, we want to obtain an input-output representation in frequency-domain in terms of the regular transfer functions. To this end, the regular *auxiliary* output  $\mathbf{y}_k^\square(t_1, \dots, t_k)$  is introduced and then transformed into frequency-domain, yielding [60]:

$$\begin{aligned} \mathbf{Y}_k^\square(s_1, \dots, s_k) &= \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \mathbf{Y}_{k,\square}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) \\ &= \sum_{j_1=1}^m \cdots \sum_{j_k=1}^m \underbrace{\mathbf{C}(s_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \cdots \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}_{j_1}}_{\mathbf{G}_{k,\square}^{(j_1, \dots, j_k)}(s_1, \dots, s_k)} U_{j_k}(s_k - s_{k-1}) \cdots U_{j_2}(s_2 - s_1) U_{j_1}(s_1). \end{aligned} \quad (4.41)$$

Again, this representation is essential to embed the transfer functions in an input-output equation. While (4.40) yields simple, single-variable factors  $(s_k \mathbf{E} - \mathbf{A})^{-1}$ , please note the more complicated, *shifted* input terms  $U_{j_k}(s_k - s_{k-1}) \cdots U_{j_2}(s_2 - s_1) U_{j_1}(s_1)$  (see also (4.38)). This confirms the fact that the triangular and regular representations can be transformed into each other by the change of variables  $\tilde{s}_1 = s_1$ ,  $\tilde{s}_2 = s_1 + s_2$ ,  $\dots$ ,  $\tilde{s}_k = s_1 + \dots + s_k$ : [221, p. 72]

$$\mathbf{Y}_k^\square(s_1, \dots, s_k) = \mathbf{Y}_k^\Delta(s_1, s_2 - s_1, s_3 - s_2, \dots, s_k - s_{k-1}). \quad (4.42)$$

The response  $\mathbf{y}_k(t)$  of the bilinear system to arbitrary inputs can be computed over the frequency-domain as follows. First,  $\mathbf{y}_k^\square(t_1, \dots, t_k)$  is calculated via the  $k$ -dimensional inverse

Laplace transform of  $\mathbf{Y}_k^\square(s_1, \dots, s_k)$ . Then, the actual output is obtained from the auxiliary output via  $\mathbf{y}_k(t) = \mathbf{y}_k^\square(t_1 = 0, \dots, t_{k-1} = 0, t_k = t)$ .

Finally, note that the kernels and transfer functions are usually given in the literature by means of the Kronecker notation, instead of the sum notation employed here. Exemplarily, we report the *regular* transfer functions in Kronecker notation in the following:

$$\begin{aligned} \mathbf{G}_k^\square(s_1, \dots, s_k) &= \mathbf{C}(s_k \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}} (\mathbf{I}_m \otimes (s_{k-1} \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}}) \cdots (\mathbf{I}_{m^{k-2}} \otimes (s_2 \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}}) \\ &\quad \times (\mathbf{I}_{m^{k-1}} \otimes (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}) \in \mathbb{C}^{p \times m^k}, \end{aligned} \quad (4.43)$$

where  $\bar{\mathbf{N}} = [\mathbf{N}_1, \dots, \mathbf{N}_m] \in \mathbb{R}^{n \times nm}$ . Hereby,  $\mathbf{G}_k^\square(s_1, \dots, s_k) \in \mathbb{C}^{p \times m^k}$  contains all  $m^k$  combinations  $\mathbf{G}_{k,\square}^{(j_1, \dots, j_k)}(s_1, \dots, s_k) \in \mathbb{R}^p$  for  $(j_1, \dots, j_k)$  (see (4.40)) or all  $m^{k-1}$  combinations  $\mathbf{G}_{k,\square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k) \in \mathbb{R}^{p \times m}$ . To make this more clear, we provide an insightful example.

*Example 4.3* (Kronecker notation for regular transfer functions). Consider a MIMO bilinear system with  $m = 3$  inputs. The input matrix  $\mathbf{B}$  is then composed of 3 vectors  $\mathbf{B} = [\mathbf{b}_1 \ \mathbf{b}_2 \ \mathbf{b}_3] \in \mathbb{R}^{n \times 3}$  and the bilinear matrix  $\bar{\mathbf{N}}$  is composed of 3 matrices  $\bar{\mathbf{N}} = [\mathbf{N}_1 \ \mathbf{N}_2 \ \mathbf{N}_3] \in \mathbb{R}^{n \times 3n}$ . The transfer matrix of the second subsystem becomes

$$\begin{aligned} \mathbf{G}_2^\square(s_1, s_2) &= \mathbf{C}(s_2 \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}} (\mathbf{I}_m \otimes (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}) \\ &= \mathbf{C}(s_2 \mathbf{E} - \mathbf{A})^{-1} [\mathbf{N}_1 \ \mathbf{N}_2 \ \mathbf{N}_3] \begin{bmatrix} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \end{bmatrix} \\ &= \left[ \mathbf{G}_{2,\square}^{(1)}(s_1, s_2) \ \mathbf{G}_{2,\square}^{(2)}(s_1, s_2) \ \mathbf{G}_{2,\square}^{(3)}(s_1, s_2) \right] \\ &= \left[ \left\{ \mathbf{G}_{2,\square}^{(j_2)}(s_1, s_2) \right\}_{j_2=1}^m \right], \end{aligned}$$

where  $\mathbf{G}_{2,\square}^{(j_2)}(s_1, s_2) = \mathbf{C}(s_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \in \mathbb{C}^{p \times m}$ . Similarly, the transfer matrix of the third subsystem is

$$\begin{aligned} \mathbf{G}_3^\square(s_1, s_2, s_3) &= \mathbf{C}(s_3 \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}} (\mathbf{I}_m \otimes (s_2 \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}}) (\mathbf{I}_m \otimes \mathbf{I}_m \otimes (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}) \\ &= \left[ \left\{ \mathbf{G}_3^{(j_2, j_3)}(s_1, s_2, s_3) \right\}_{j_2, j_3=1}^m \right], \end{aligned}$$

where  $\mathbf{G}_3^{(j_2, j_3)}(s_1, s_2, s_3) = \mathbf{C}(s_3 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_3} (s_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \in \mathbb{C}^{p \times m}$ .

△

### Quadratic-bilinear systems

The derivation of multivariable kernels in time-domain becomes quite cumbersome for quadratic-bilinear systems. Moreover, transfer functions cannot be derived via Laplace transform of the kernels due to the quadratic term. For this reason, we will only state the output response in time-domain in the following. The derivation of transfer functions in frequency-domain will be accomplished by means of the growing exponential approach in Section 4.4.2.

The Volterra series representation for the second subsystem gained in (4.31) is the starting point for the brief subsequent analysis.

**Triangular kernels** Performing change of variables, we can rewrite (4.31) in triangular form as follows (with interchanged  $j$  indices):

$$\begin{aligned} \mathbf{y}_2(t) = & \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=\tau_3}^t \int_{\tau_2=\tau_3}^t \int_{\tau_3=0}^t \mathbf{C} e^{\tilde{\mathbf{A}}\tau_3} \tilde{\mathbf{H}} (e^{\tilde{\mathbf{A}}(\tau_2-\tau_3)} \tilde{\mathbf{b}}_{j_2} \otimes e^{\tilde{\mathbf{A}}(\tau_1-\tau_3)} \tilde{\mathbf{b}}_{j_1}) u_{j_2}(t-\tau_2) u_{j_1}(t-\tau_1) d\tau_1 d\tau_2 d\tau_3 \\ & + \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=\tau_2}^t \int_{\tau_2=0}^t \mathbf{C} e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}(\tau_1-\tau_2)} \tilde{\mathbf{b}}_{j_1} u_{j_2}(t-\tau_2) u_{j_1}(t-\tau_1) d\tau_1 d\tau_2. \end{aligned} \quad (4.44)$$

With the representation above, the second *triangular* kernel reads

$$\mathbf{g}_{2,\Delta}^{(j_1,j_2)}(t_1, t_2) = \mathbf{C} e^{\tilde{\mathbf{A}}t_2} \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}(t_1-t_2)} \tilde{\mathbf{b}}_{j_1} + \int_{\tau_3=0}^t \mathbf{C} e^{\tilde{\mathbf{A}}\tau_3} \tilde{\mathbf{H}} (e^{\tilde{\mathbf{A}}(t_2-\tau_3)} \tilde{\mathbf{b}}_{j_2} \otimes e^{\tilde{\mathbf{A}}(t_1-\tau_3)} \tilde{\mathbf{b}}_{j_1}) d\tau_3.$$

The first term of the kernel corresponds to the bilinear part. Unfortunately, the second term still contains an integral because the quadratic part of (4.44) is composed of three integrals. Note, however, that the quadratic part contains only *two* input terms. Thus, we believe that this part also belongs to the *second* kernel and not to the third one.

**Regular kernels** Performing change of variables, or proceeding in a similar manner as described in Footnote 3 with

$$\mathbf{x}_2(t) = \int_{\tau_3=0}^t e^{\tilde{\mathbf{A}}\tau_3} \tilde{\mathbf{H}} (\mathbf{x}_1(t-\tau_3) \otimes \mathbf{x}_1(t-\tau_3)) d\tau_3 + \sum_{j=1}^m \int_{\tau_2=0}^t e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{N}}_{j_2} \mathbf{x}_1(t-\tau_2) u_{j_2}(t-\tau_2) d\tau_2,$$

yields the regular representation

$$\begin{aligned} \mathbf{y}_2(t) = & \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^{t-\tau_3} \int_{\tau_2=0}^{t-\tau_3} \int_{\tau_3=0}^t \mathbf{C} e^{\tilde{\mathbf{A}}\tau_3} \tilde{\mathbf{H}} (e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{b}}_{j_2} \otimes e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{b}}_{j_1}) u_{j_2}(t-\tau_3-\tau_2) u_{j_1}(t-\tau_3-\tau_1) d\tau_1 d\tau_2 d\tau_3 \\ & + \sum_{j_1=1}^m \sum_{j_2=1}^m \int_{\tau_1=0}^{t-\tau_2} \int_{\tau_2=0}^t \mathbf{C} e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{b}}_{j_1} u_{j_2}(t-\tau_2) u_{j_1}(t-\tau_2-\tau_1) d\tau_1 d\tau_2. \end{aligned} \quad (4.45)$$

From the above equation, regular kernels have been proposed in [34, 35] and [103, Sec. 4.3.1, 5.2.1] to define Gramians and the  $\mathcal{H}_2$ -norm for quadratic-bilinear systems. The main idea is to decouple the quadratic and bilinear terms in order to obtain a second-order kernel which is still purely bilinear, whereas the quadratic term arises only as of the *third* kernel. We do not report the proposed regular kernels here, but refer the reader to the mentioned publications.

#### 4.4.2 Growing exponential approach

The growing exponential approach constitutes another way of finding transfer function descriptions for polynomial systems. The approach is based upon the *eigenfunction property*<sup>4</sup> of linear systems: the response  $\mathbf{y}(t)$  to a growing exponential input  $\mathbf{u}(t) = \mathbf{1}_m e^{\sigma t}$  with  $\mathbf{1}_m \in \mathbb{R}^m$  is given by  $\mathbf{y}(t) = \mathbf{G}(\sigma) \mathbf{1}_m e^{\sigma t}$ , where the transfer function  $\mathbf{G}(\sigma) = \mathbf{C}(\sigma \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}$  represents the scaling factor. This concept can also be applied to polynomial systems described by means of homogeneous subsystems (cf. variational equation approach in Section 4.3.2).

<sup>4</sup>The concept of eigenfunctions will be further discussed in Chapter 8.

The idea is to apply a sum of growing exponentials as input signal (cf. Eq. (3.80))

$$\mathbf{u}(t) = \sum_{l_1=1}^r \mathbf{1}_m u_{l_1} e^{s_{l_1} t}, \quad \text{or equivalently} \quad u_j(t) = \sum_{l_1=1}^r u_{l_1} e^{s_{l_1} t}, \quad j = 1, \dots, m. \quad (4.46)$$

For simplicity, we employ the vector of ones  $\mathbf{1}_m$  as tangential direction for all exponentials. Note, however, that we could also use different directions  $\mathbf{r}_{l_1}$ . The input, together with an expected solution ansatz for  $\mathbf{x}_k(t)$ , is then substituted in the corresponding  $k$ -th variational equation in order to gain an input-output characterization in terms of transfer functions. The procedure will be illustrated for bilinear and quadratic-bilinear systems in the following.

### Bilinear systems

The variational equations (4.25) are the starting point for the application of the growing exponential approach. We will use the sum notation for the MIMO case.

**1-st subsystem** For the first (well-known) subsystem  $\mathbf{E}\dot{\mathbf{x}}_1(t) = \mathbf{A}\mathbf{x}_1(t) + \mathbf{B}\mathbf{u}(t)$ , we expect a solution of the form  $\mathbf{x}_1(t) = \sum_{l_1=1}^r \lambda_1(s_{l_1}) \mathbf{1}_m u_{l_1} e^{s_{l_1} t}$ . Inserting the input (4.46) and the assumed solution into the first variational equation yields

$$\begin{aligned} \mathbf{E} \sum_{l_1=1}^r s_{l_1} \lambda_1(s_{l_1}) \mathbf{1}_m u_{l_1} e^{s_{l_1} t} &= \mathbf{A} \sum_{l_1=1}^r \lambda_1(s_{l_1}) \mathbf{1}_m u_{l_1} e^{s_{l_1} t} + \mathbf{B} \sum_{l_1=1}^r \mathbf{1}_m u_{l_1} e^{s_{l_1} t} \\ \iff \lambda_1(s_{l_1}) &= (s_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}, \quad l_1 = 1, \dots, r. \end{aligned} \quad (4.47)$$

**2-nd subsystem** The second subsystem is given by  $\mathbf{E}\dot{\mathbf{x}}_2(t) = \mathbf{A}\mathbf{x}_2(t) + \sum_{j=1}^m N_j \mathbf{x}_1(t) u_j(t)$ . The ‘‘pseudo-input’’ then reads (for  $r = 2$ ):

$$\begin{aligned} \mathbf{x}_1(t) u_j(t) &= (\lambda_1(s_1) \mathbf{1}_m u_1 e^{s_1 t} + \lambda_1(s_2) \mathbf{1}_m u_2 e^{s_2 t}) (u_1 e^{s_1 t} + u_2 e^{s_2 t}) \\ &= \lambda_1(s_1) \mathbf{1}_m u_1^2 e^{2s_1 t} + \lambda_1(s_2) \mathbf{1}_m u_2^2 e^{2s_2 t} + (\lambda_1(s_1) + \lambda_1(s_2)) \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t}, \end{aligned} \quad (4.48)$$

and is composed of three exponential terms  $e^{2s_1 t}$ ,  $e^{2s_2 t}$  and  $e^{(s_1+s_2)t}$ . Therefore, the solution  $\mathbf{x}_2(t)$  is assumed to be of the form

$$\mathbf{x}_2(t) = \sum_{j_2=1}^m \lambda_{2,a}^{(j_2)} \mathbf{1}_m u_1^2 e^{2s_1 t} + \lambda_{2,b}^{(j_2)} \mathbf{1}_m u_2^2 e^{2s_2 t} + \lambda_{2,c}^{(j_2)} \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t}. \quad (4.49)$$

Substituting the pseudo-input and assumed solution into the second subsystem yields

$$\begin{aligned} \mathbf{E} \sum_{j_2=1}^m &\left( 2s_1 \lambda_{2,a}^{(j_2)} \mathbf{1}_m u_1^2 e^{2s_1 t} + 2s_2 \lambda_{2,b}^{(j_2)} \mathbf{1}_m u_2^2 e^{2s_2 t} + (s_1 + s_2) \lambda_{2,c}^{(j_2)} \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t} \right) \\ &= \mathbf{A} \sum_{j_2=1}^m \left( \lambda_{2,a}^{(j_2)} \mathbf{1}_m u_1^2 e^{2s_1 t} + \lambda_{2,b}^{(j_2)} \mathbf{1}_m u_2^2 e^{2s_2 t} + \lambda_{2,c}^{(j_2)} \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t} \right) \\ &+ \sum_{j_2=1}^m N_{j_2} \left( \lambda_1(s_1) \mathbf{1}_m u_1^2 e^{2s_1 t} + \lambda_1(s_2) \mathbf{1}_m u_2^2 e^{2s_2 t} + (\lambda_1(s_1) + \lambda_1(s_2)) \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t} \right). \end{aligned} \quad (4.50)$$

Comparing the coefficients for terms  $e^{2s_1 t}$ ,  $e^{2s_2 t}$ ,  $e^{(s_1+s_2)t}$  yields the scalings  $\lambda_{2,a}$ ,  $\lambda_{2,b}$  and  $\lambda_{2,c}$ :

$$\begin{aligned}\lambda_{2,a}^{(j_2)} &= \lambda_{2,\text{sym}}^{(j_2)}(s_1, s_1) = \lambda_{2,\Delta}^{(j_2)}(s_1, s_1) = (2s_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_1), \\ \lambda_{2,b}^{(j_2)} &= \lambda_{2,\text{sym}}^{(j_2)}(s_2, s_2) = \lambda_{2,\Delta}^{(j_2)}(s_2, s_2) = (2s_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_2), \\ \lambda_{2,c}^{(j_2)} &= 2\lambda_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \lambda_{2,\Delta}^{(j_2)}(s_1, s_2) + \lambda_{2,\Delta}^{(j_2)}(s_2, s_1) = ((s_1 + s_2)\mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (\boldsymbol{\lambda}_1(s_1) + \boldsymbol{\lambda}_1(s_2)),\end{aligned}$$

with (in triangular form)

$$\begin{aligned}\lambda_{2,\Delta}^{(j_2)}(s_1, s_2) &= ((s_1 + s_2)\mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_1), \\ \lambda_{2,\Delta}^{(j_2)}(s_2, s_1) &= ((s_2 + s_1)\mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_2),\end{aligned}\tag{4.51}$$

and (in *symmetric* form)

$$\lambda_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \frac{1}{2}((s_1 + s_2)\mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (\boldsymbol{\lambda}_1(s_1) + \boldsymbol{\lambda}_1(s_2)).\tag{4.52}$$

Note that  $\lambda_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \frac{1}{2}(\lambda_{2,\Delta}^{(j_2)}(s_1, s_2) + \lambda_{2,\Delta}^{(j_2)}(s_2, s_1))$  and  $\lambda_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \lambda_{2,\text{sym}}^{(j_2)}(s_2, s_1)$ . The symmetric form can thus be obtained by summing over all permutations  $\pi(\cdot)$  of the frequency arguments  $(s_1, s_2)$  of the triangular form. Moreover,  $\lambda_{2,\text{sym}}^{(j_2)}(s_1, s_2)$  is symmetric in each frequency argument.

**$k$ -th subsystem** Based on (4.49) and the obtained results for the second subsystem, the formulas can be generalized for higher-order subsystems as follows:

$$\begin{aligned}\mathbf{x}_k(t) &= \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_k=1}^r \lambda_{k,\Delta}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) \mathbf{1}_m u_{l_1} \cdots u_{l_k} e^{s_{l_1} t} \cdots e^{s_{l_k} t} \\ &= \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{\pi(\cdot)} \lambda_{k,\text{sym}}^{(j_2, \dots, j_k)}(s_{\pi(1)}, \dots, s_{\pi(k)}) \mathbf{1}_m u_{\pi(1)} \cdots u_{\pi(k)} e^{s_{\pi(1)} t} \cdots e^{s_{\pi(k)} t}\end{aligned}\tag{4.53}$$

with the recursion (in triangular form)

$$\begin{aligned}\lambda_{k,\Delta}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) &= ((s_{l_1} + \cdots + s_{l_k})\mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \lambda_{k-1,\Delta}^{(\cdot)}(s_{l_1}, \dots, s_{l_{k-1}}), \quad k \geq 2 \\ \lambda_1(s_{l_1}) &= (s_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B},\end{aligned}\tag{4.54}$$

and (in *symmetric* form)

$$\lambda_{k,\text{sym}}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) = \frac{1}{k!} \sum_{\pi(\cdot)} \lambda_{k,\Delta}^{(j_2, \dots, j_k)}(s_{\pi(1)}, \dots, s_{\pi(k)}).\tag{4.55}$$

For example, in case  $k=3$ , the symmetric form  $\lambda_{3,\text{sym}}^{(j_2, j_3)}(s_{l_1}, s_{l_2}, s_{l_3})$  is obtained by summing over all  $3!$  permutations  $(\pi(1), \pi(2), \pi(3)) \in \{(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)\}$  of the triangular form.

Note that the dimension of the scalings is  $\lambda_{k,\Delta}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) \in \mathbb{R}^{n \times m}$ . An input-output representation can be easily obtained from (4.53) via  $\mathbf{y}_k(t) = \mathbf{C} \mathbf{x}_k(t)$  with the transfer functions  $\mathbf{G}_{k,\star}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) = \mathbf{C} \lambda_{k,\star}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k}) \in \mathbb{C}^{p \times m}$  (cf. Example 4.3).

### Quadratic-bilinear systems

The variational equations (4.30) are the starting point for the application of the growing exponential approach. We again use the sum notation in the MIMO case. The reader is referred to [Fio16, Sec. 4.1.2] for a similar derivation using the Kronecker notation.

**2-nd subsystem** The 2-nd subsystem is  $\mathbf{E}\dot{\mathbf{x}}_2(t) = \mathbf{A}\mathbf{x}_2(t) + \mathbf{H}\mathbf{x}_1(t) \otimes \mathbf{x}_1(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}_1(t) u_j(t)$ . With the assumed solution for  $\mathbf{x}_1(t)$ , the term  $\mathbf{x}_1(t) \otimes \mathbf{x}_1(t)$  reads:

$$\mathbf{x}_1(t) \otimes \mathbf{x}_1(t) = (\boldsymbol{\lambda}_1(s_1) \mathbf{1}_m u_1 e^{s_1 t} + \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m u_2 e^{s_2 t}) \otimes (\boldsymbol{\lambda}_1(s_1) \mathbf{1}_m u_1 e^{s_1 t} + \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m u_2 e^{s_2 t}).$$

The ‘‘pseudo-input’’  $\mathbf{x}_1(t) u_j(t)$  for  $r=2$  is given by Eq. (4.48) and is again composed of three exponential terms. Thus, we make a similar ansatz (4.49) for the assumed solution  $\mathbf{x}_2(t)$ . Inserting  $\mathbf{x}_1 \otimes \mathbf{x}_1$ , the pseudo-input and assumed solution into the 2-nd subsystem yields

$$\begin{aligned} & \mathbf{E} \sum_{j_2=1}^m \left( 2s_1 \boldsymbol{\lambda}_{2,a}^{(j_2)} u_1^2 e^{2s_1 t} + 2s_2 \boldsymbol{\lambda}_{2,b}^{(j_2)} u_2^2 e^{2s_2 t} + (s_1 + s_2) \boldsymbol{\lambda}_{2,c}^{(j_2)} u_1 u_2 e^{(s_1+s_2)t} \right) \\ &= \mathbf{A} \sum_{j_2=1}^m \left( \boldsymbol{\lambda}_{2,a}^{(j_2)} u_1^2 e^{2s_1 t} + \boldsymbol{\lambda}_{2,b}^{(j_2)} u_2^2 e^{2s_2 t} + \boldsymbol{\lambda}_{2,c}^{(j_2)} u_1 u_2 e^{(s_1+s_2)t} \right) \\ & \quad + \mathbf{H} \left( \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m u_1^2 e^{2s_1 t} + \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m u_2^2 e^{2s_2 t} \right. \\ & \quad \left. + (\boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m + \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m) u_1 u_2 e^{(s_1+s_2)t} \right) \\ & \quad + \sum_{j_2=1}^m \mathbf{N}_{j_2} \left( \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m u_1^2 e^{2s_1 t} + \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m u_2^2 e^{2s_2 t} + (\boldsymbol{\lambda}_1(s_1) + \boldsymbol{\lambda}_1(s_2)) \mathbf{1}_m u_1 u_2 e^{(s_1+s_2)t} \right). \end{aligned} \quad (4.56)$$

Comparing the coefficients for terms  $e^{2s_1 t}$ ,  $e^{2s_2 t}$ ,  $e^{(s_1+s_2)t}$  yields the scalings  $\boldsymbol{\lambda}_{2,a}$ ,  $\boldsymbol{\lambda}_{2,b}$  and  $\boldsymbol{\lambda}_{2,c}$ :

$$\begin{aligned} \boldsymbol{\lambda}_{2,a}^{(j_2)} &= \boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_1, s_1) = \boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_1, s_1) = (2s_1 \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m) + \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \right) \\ \boldsymbol{\lambda}_{2,b}^{(j_2)} &= \boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_2, s_2) = \boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_2, s_2) = (2s_2 \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m) + \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \right) \\ \boldsymbol{\lambda}_{2,c}^{(j_2)} &= 2\boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_1, s_2) = ((s_1 + s_2) \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_1) \otimes \boldsymbol{\lambda}_1(s_2) + \boldsymbol{\lambda}_1(s_2) \otimes \boldsymbol{\lambda}_1(s_1)) \mathbf{1}_m^2 \right. \\ & \quad \left. + \mathbf{N}_{j_2} (\boldsymbol{\lambda}_1(s_1) + \boldsymbol{\lambda}_1(s_2)) \mathbf{1}_m \right), \end{aligned}$$

with (in triangular form)

$$\begin{aligned} \boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_1, s_2) &= ((s_1 + s_2) \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m) + \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m \right), \\ \boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_2, s_1) &= ((s_2 + s_1) \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \otimes \boldsymbol{\lambda}_1(s_1) \mathbf{1}_m) + \mathbf{N}_{j_2} \boldsymbol{\lambda}_1(s_2) \mathbf{1}_m \right), \end{aligned} \quad (4.57)$$

and (in *symmetric* form)

$$\begin{aligned} \boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_1, s_2) &= \frac{1}{2} ((s_1 + s_2) \mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H}(\boldsymbol{\lambda}_1(s_1) \otimes \boldsymbol{\lambda}_1(s_2) + \boldsymbol{\lambda}_1(s_2) \otimes \boldsymbol{\lambda}_1(s_1)) \mathbf{1}_m^2 \right. \\ & \quad \left. + \mathbf{N}_{j_2} (\boldsymbol{\lambda}_1(s_1) + \boldsymbol{\lambda}_1(s_2)) \mathbf{1}_m \right). \end{aligned}$$

Again it holds:  $\boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \frac{1}{2} (\boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_1, s_2) + \boldsymbol{\lambda}_{2,\Delta}^{(j_2)}(s_2, s_1))$  and  $\boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_1, s_2) = \boldsymbol{\lambda}_{2,\text{sym}}^{(j_2)}(s_2, s_1)$ .



**$k$ -th subsystem** Based on the obtained results for the second subsystem, the formulas can be generalized for higher-order subsystems as follows:

$$\mathbf{x}_k(t) = \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_k=1}^r \boldsymbol{\lambda}_{k,\Delta}^{(j_2,\dots,j_k)}(s_{l_1}, \dots, s_{l_k}) u_{l_1} \cdots u_{l_k} e^{s_{l_1} t} \cdots e^{s_{l_k} t} \quad (4.58)$$

with the recursion (in triangular form)

$$\begin{aligned} \boldsymbol{\lambda}_{k,\Delta}^{(j_2,\dots,j_k)}(s_{l_1}, \dots, s_{l_k}) &= ((s_{l_1} + \cdots + s_{l_k})\mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H} \sum_{i=1}^{k-1} (\boldsymbol{\lambda}_{i,\Delta}^{(\cdot)} \otimes \boldsymbol{\lambda}_{k-i,\Delta}^{(\cdot)}) + \mathbf{N}_{j_k} \boldsymbol{\lambda}_{k-1,\Delta}^{(\cdot)} \right), \\ \boldsymbol{\lambda}_1(s_{l_1}) &= (s_{l_1}\mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{1}_m. \end{aligned}$$

Note that the dimension of the scalings is now  $\boldsymbol{\lambda}_{k,\Delta}^{(j_2,\dots,j_k)}(s_{l_1}, \dots, s_{l_k}) \in \mathbb{R}^{n \times 1}$ , since they still contain the vectors  $\mathbf{1}_m$  (or the tangential directions) due to the quadratic term. These vectors are omitted in the following to state the transfer functions of the second subsystem.

Using the abbreviation  $\mathbf{A}_s = (s\mathbf{E} - \mathbf{A})$ , the transfer functions  $\mathbf{G}_{2,*}^{(j_1,j_2)}(s_1, s_2) \in \mathbb{C}^p$  of the second subsystem are given in *sum notation* by

$$\begin{aligned} \mathbf{G}_{2,\Delta}^{(j_1,j_2)}(s_1, s_2) &= \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_2} \otimes \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1}) + \mathbf{N}_{j_2} \mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_1} \right), \\ \mathbf{G}_{2,\square}^{(j_1,j_2)}(s_1, s_2) &= \mathbf{C} \mathbf{A}_{s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_2} \otimes \mathbf{A}_{s_2-s_1}^{-1} \mathbf{b}_{j_1}) + \mathbf{N}_{j_2} \mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_1} \right), \\ \mathbf{G}_{2,\text{sym}}^{(j_1,j_2)}(s_1, s_2) &= \frac{1}{2} \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_2} \otimes \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1} + \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1} \otimes \mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_2}) + \mathbf{N}_{j_2} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_1} + \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1}) \right) \\ &= \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_2} \otimes \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1}) + \frac{1}{2} \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \mathbf{N}_{j_2} (\mathbf{A}_{s_1}^{-1} \mathbf{b}_{j_1} + \mathbf{A}_{s_2}^{-1} \mathbf{b}_{j_1}). \end{aligned} \quad (4.59)$$

The quadratic term in  $\mathbf{G}_{2,\text{sym}}^{(j_1,j_2)}(s_1, s_2)$  can be simplified if  $\mathbf{H}$  is symmetric, since the relation  $\mathbf{H}(\mathbf{u} \otimes \mathbf{v}) = \mathbf{H}(\mathbf{v} \otimes \mathbf{u})$  holds for vectors (cf. Eq. (2.16)). The regular transfer function  $\mathbf{G}_{2,\square}^{(j_1,j_2)}(s_1, s_2)$  is obtained from the triangular one by applying the relationship (4.42), i.e.  $\tilde{s}_1 = s_1$  and  $\tilde{s}_2 = s_1 + s_2 \leftrightarrow s_2 = \tilde{s}_2 - s_1$ .

The transfer functions  $\mathbf{G}_2^*(s_1, s_2) \in \mathbb{C}^{p \times m^2}$  of the second subsystem are given in *Kronecker notation* by

$$\begin{aligned} \mathbf{G}_2^\Delta(s_1, s_2) &= \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{B} \otimes \mathbf{A}_{s_2}^{-1} \mathbf{B}) + \bar{\mathbf{N}} (\mathbf{I}_m \otimes \mathbf{A}_{s_1}^{-1} \mathbf{B}) \right), \\ \mathbf{G}_2^\square(s_1, s_2) &= \mathbf{C} \mathbf{A}_{s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{B} \otimes \mathbf{A}_{s_2-s_1}^{-1} \mathbf{B}) + \bar{\mathbf{N}} (\mathbf{I}_m \otimes \mathbf{A}_{s_1}^{-1} \mathbf{B}) \right), \\ \mathbf{G}_2^{\text{sym}}(s_1, s_2) &= \frac{1}{2} \mathbf{C} \mathbf{A}_{s_1+s_2}^{-1} \left( \mathbf{H} (\mathbf{A}_{s_1}^{-1} \mathbf{B} \otimes \mathbf{A}_{s_2}^{-1} \mathbf{B} + \mathbf{A}_{s_2}^{-1} \mathbf{B} \otimes \mathbf{A}_{s_1}^{-1} \mathbf{B}) + \bar{\mathbf{N}} (\mathbf{I}_m \otimes (\mathbf{A}_{s_1}^{-1} \mathbf{B} + \mathbf{A}_{s_2}^{-1} \mathbf{B})) \right). \end{aligned} \quad (4.60)$$

The quadratic term in  $\mathbf{G}_2^{\text{sym}}(s_1, s_2)$  cannot be simplified, since  $\mathbf{H}(\mathbf{U} \otimes \mathbf{V}) \neq \mathbf{H}(\mathbf{V} \otimes \mathbf{U})$  for matrices, even if  $\mathbf{H}$  is symmetric.



# Chapter 5

## Bilinear Systems

In this chapter we focus on the systems theory and model order reduction of bilinear dynamical systems of the form

$$\zeta : \begin{cases} \mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x}(t) u_j(t) + \mathbf{B} \mathbf{u}(t), & \mathbf{x}(0) = \mathbf{x}_0, \\ \mathbf{y}(t) = \mathbf{C} \mathbf{x}(t), \end{cases} \quad (5.1a)$$

$$(5.1b)$$

where  $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{n \times n}$  with  $\mathbf{E}$  regular,  $\mathbf{N}_j \in \mathbb{R}^{n \times n}$  for  $j = 1, \dots, m$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$  and  $\mathbf{C} \in \mathbb{R}^{p \times n}$ . For brevity of presentation, at times we will use the notation  $\zeta = (\mathbf{A}, \mathbf{N}_j, \mathbf{B}, \mathbf{C}, \mathbf{E})$  to denote the bilinear system. All  $m$  bilinear matrices can be concatenated next to each other as  $\bar{\mathbf{N}} := \mathcal{N}^{(1)} = [\mathbf{N}_1 \ \mathbf{N}_2 \ \dots \ \mathbf{N}_m] \in \mathbb{R}^{n \times n \cdot m}$  and  $\bar{\mathbf{N}}^{(2)} := \mathcal{N}^{(2)} = [\mathbf{N}_1^T \ \mathbf{N}_2^T \ \dots \ \mathbf{N}_m^T] \in \mathbb{R}^{n \times n \cdot m}$ . For theoretical purposes, we will sometimes use  $\tilde{\mathbf{A}} := \mathbf{E}^{-1} \mathbf{A}$ ,  $\tilde{\mathbf{N}}_j := \mathbf{E}^{-1} \mathbf{N}_j$  and  $\tilde{\mathbf{B}} := \mathbf{E}^{-1} \mathbf{B}$ .

The Volterra series representation — more specifically the regular kernels and transfer functions — has enabled the generalization of well-known system-theoretic concepts (e.g. Gramians,  $\mathcal{H}_2$ -norm, etc.) to the bilinear setting. In Section 5.1 we will revisit these concepts, hereby focusing on the MIMO case and stating new results for the pole-residue formulation. Then, we will particularly concentrate on *Krylov-based* model reduction of bilinear systems, more specifically on subsystem (Section 5.3) and Volterra series (Section 5.4) interpolation. We will turn our attention to the MIMO case and provide useful insights into the multimoment setting, the choice of the interpolation data and the Arnoldi-like implementation.

Parallel to the theoretical investigation, we have developed the BSSS and BSSSMOR toolboxes including some important analysis and model reduction algorithms. We will highlight these functions at the corresponding place using boxes.

### 5.1 Bilinear systems theory

In this section we revisit important system-theoretic concepts for bilinear systems such as the pole-residue formulation, BIBO stability, Gramians, bilinear Lyapunov equations and the  $\mathcal{H}_2$ -norm. The generalization is based on the *regular* kernels and transfer functions.

#### Diagonal form and pole-residue formulation

Similar to the linear setting, we are able to transform a bilinear system into another state-space realization via the transformation (3.9). This changes the realization from  $\zeta = (\mathbf{A}, \mathbf{N}_j, \mathbf{B}, \mathbf{C}, \mathbf{E})$  to  $\hat{\zeta} = (\mathbf{OAT}^{-1}, \mathbf{ON}_j\mathbf{T}^{-1}, \mathbf{TB}, \mathbf{CT}^{-1}, \mathbf{OET}^{-1})$  or to  $\hat{\zeta} = (\mathbf{T}\tilde{\mathbf{A}}\mathbf{T}^{-1}, \mathbf{T}\tilde{\mathbf{N}}_j\mathbf{T}^{-1}, \mathbf{T}\tilde{\mathbf{B}}, \mathbf{CT}^{-1}, \mathbf{I})$

in the explicit case. Again, important system-theoretic properties (like e.g. stability or input-output behavior) remain invariant under such transformations.

To bring the system into diagonal form, we assume that the matrix  $\tilde{\mathbf{A}}$  is diagonalizable with distinct eigenvalues  $\{\lambda_l\}_{l=1}^n$ . We then employ the right eigenvectors in  $\mathbf{X}$  for the transformation via  $\mathbf{T} = \mathbf{X}^{-1}$ . This yields the diagonalized system  $\hat{\zeta} = (\mathbf{\Lambda}, \hat{\mathbf{N}}_j, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \mathbf{I})$

$$\hat{\zeta} : \begin{cases} \dot{\mathbf{z}}(t) = \mathbf{\Lambda} \mathbf{z}(t) + \sum_{j=1}^m \hat{\mathbf{N}}_j \mathbf{z}(t) u_j(t) + \hat{\mathbf{B}} \mathbf{u}(t), & \mathbf{z}(0) = \mathbf{X}^{-1} \mathbf{x}_0, \\ \mathbf{y}(t) = \hat{\mathbf{C}} \mathbf{z}(t), \end{cases} \quad (5.2a)$$

$$(5.2b)$$

with the matrices

$$\mathbf{\Lambda} = \mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{A} \mathbf{X} \Leftrightarrow \mathbf{E}^{-1} \mathbf{A} = \mathbf{X} \mathbf{\Lambda} \mathbf{X}^{-1}, \quad \mathbf{\Lambda} = \mathbf{D}, \quad [\mathbf{X}, \mathbf{D}] = \mathbf{eig}(\mathbf{A}, \mathbf{E}), \quad (5.3a)$$

$$\hat{\mathbf{N}}_j = \mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{N}_j \mathbf{X}, \quad \mathbf{Njhat} = \mathbf{X} \setminus (\mathbf{E} \setminus (\mathbf{N}_j * \mathbf{X})), \quad (5.3b)$$

$$\hat{\mathbf{B}} = \mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{B}, \quad \mathbf{Bhat} = \mathbf{X} \setminus (\mathbf{E} \setminus \mathbf{B}), \quad (5.3c)$$

$$\hat{\mathbf{C}} = \mathbf{C} \mathbf{X}, \quad \mathbf{Chat} = \mathbf{C} * \mathbf{X}. \quad (5.3d)$$

The regular transfer function of the diagonalized system is given by

$$\hat{\mathbf{G}}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k) = \hat{\mathbf{C}}(s_k \mathbf{I} - \mathbf{\Lambda})^{-1} \hat{\mathbf{N}}_{j_k} \cdots \hat{\mathbf{N}}_{j_3} (s_2 \mathbf{I} - \mathbf{\Lambda})^{-1} \hat{\mathbf{N}}_{j_2} (s_1 \mathbf{I} - \mathbf{\Lambda})^{-1} \hat{\mathbf{B}}.$$

The pole-residue formulation can be easily obtained by exploiting the diagonal form. Writing every term  $(s_k \mathbf{I} - \mathbf{\Lambda})^{-1}$  explicitly using the inverse property yields [92, Röt18]:

$$\hat{\mathbf{G}}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k) = \sum_{l_1=1}^n \cdots \sum_{l_k=1}^n \frac{\Phi_{l_1, \dots, l_k}^{(j_2, \dots, j_k)}}{\prod_{\ell=1}^k (s_\ell - \lambda_{l_\ell})}, \quad j_2, \dots, j_k = 1, \dots, m. \quad (5.4)$$

The matrix-residues are given by

$$\Phi_{l_1, \dots, l_k}^{(j_2, \dots, j_k)} = \hat{\mathbf{c}}_{l_k} \cdot \hat{n}_{j_k l_k, l_{k-1}} \cdots \hat{n}_{j_2 l_2, l_1} \cdot \hat{\mathbf{b}}_{l_1}^\top \in \mathbb{C}^{p \times m}, \quad l_1, \dots, l_k = 1, \dots, n, \quad (5.5)$$

where

$$\hat{\mathbf{C}} = [\hat{\mathbf{c}}_1, \dots, \hat{\mathbf{c}}_n] \in \mathbb{C}^{p \times n}, \quad \hat{\mathbf{B}} = \begin{bmatrix} \hat{\mathbf{b}}_1^\top \\ \vdots \\ \hat{\mathbf{b}}_n^\top \end{bmatrix} \in \mathbb{C}^{n \times m} \quad \text{and} \quad \hat{\mathbf{N}}_j = \begin{bmatrix} \hat{n}_{j_1, 1} & \cdots & \hat{n}_{j_1, n} \\ \vdots & \ddots & \vdots \\ \hat{n}_{j_n, 1} & \cdots & \hat{n}_{j_n, n} \end{bmatrix} \in \mathbb{C}^{n \times n}.$$

To make the above statements more comprehensible, we want to provide some examples.

*Example 5.1* (Pole-residue formulation for bilinear systems). Since the first transfer function is only dependent on one variable  $s_1$  and does not contain the bilinear term, we can write the pole-residue formulation in the SISO case as follows

$$G_1(s_1) = \hat{\mathbf{c}}^\top (s_1 \mathbf{I} - \mathbf{\Lambda})^{-1} \hat{\mathbf{b}} = \sum_{l_1=1}^n \frac{\hat{\mathbf{c}}_{l_1} \cdot \hat{\mathbf{b}}_{l_1}}{s_1 - \lambda_{l_1}}, \quad n^1 \text{ residues } \phi_{l_1} = \hat{\mathbf{c}}_{l_1} \cdot \hat{\mathbf{b}}_{l_1}.$$

Each subsequent transfer function yields an additional factor  $\hat{n}_{l_k, l_{k-1}}$  in the numerator. Thus, we obtain  $n^k$  residues and  $n^k$  pole combinations due to each additional variable  $s_k$ .

$$\begin{aligned} G_2^\square(s_1, s_2) &= \cdots = \sum_{l_1=1}^n \sum_{l_2=1}^n \frac{\hat{c}_{l_2} \cdot \hat{n}_{l_2, l_1} \cdot \hat{b}_{l_1}}{(s_1 - \lambda_{l_1})(s_2 - \lambda_{l_2})}, \quad n^2 \text{ residues } \phi_{l_1, l_2} = \hat{c}_{l_2} \cdot \hat{n}_{l_2, l_1} \cdot \hat{b}_{l_1} \\ G_3^\square(s_1, s_2, s_3) &= \cdots = \sum_{l_1=1}^n \sum_{l_2=1}^n \sum_{l_3=1}^n \frac{\hat{c}_{l_3} \cdot \hat{n}_{l_3, l_2} \cdot \hat{n}_{l_2, l_1} \cdot \hat{b}_{l_1}}{(s_1 - \lambda_{l_1})(s_2 - \lambda_{l_2})(s_3 - \lambda_{l_3})}, \quad n^3 \text{ residues } \phi_{l_1, l_2, l_3} \\ &\vdots \end{aligned}$$

In the MIMO case, the partial-fraction decomposition looks as follows:

$$\hat{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k) = \frac{\Phi_{1, \dots, 1}^{(j_2, \dots, j_k)}}{(s_1 - \lambda_1) \cdots (s_k - \lambda_1)} + \cdots + \frac{\Phi_{n, \dots, n}^{(j_2, \dots, j_k)}}{(s_1 - \lambda_n) \cdots (s_k - \lambda_n)}. \quad (5.6)$$

Note that the denominator of each summand is factored into the product of single-variable polynomials. Despite the  $n^k$  pole combinations, it is important to note that the bilinear system possesses *only*  $n$  eigenvalues  $\lambda_1, \dots, \lambda_n$ .  $\triangle$

### Bounded-input bounded-output (BIBO) stability

In linear systems theory, we can evaluate the asymptotic/BIBO stability by simply investigating whether all eigenvalues/poles of the system lie in the open left-half plane, i.e.  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$ . This criterion follows from the fact that the response (3.2) for zero/bounded input  $\mathbf{u}(t)$  decays for  $t \rightarrow \infty$ /is bounded, if the transition matrix  $e^{\tilde{\mathbf{A}}t}$  contains only decaying exponentials.

Asymptotic stability for *general* nonlinear systems is usually analyzed in terms of an equilibrium point  $\mathbf{x}_{\text{eq}}$  using a Lyapunov function  $V(\mathbf{x}_{\text{eq}})$ . Bounded-input bounded-output (aka. *input-to-state* (ISS)) stability requires to quantify the effect of the input  $\mathbf{u}(t)$  on the stability. For special nonlinear system classes, such as polynomial systems, statements for BIBO stability can be developed by exploiting e.g. the Volterra series representation.

As we have seen in (4.20) and Footnote 2, the Volterra series for bilinear systems (4.21) converges for “small” bounded inputs. Similarly, sufficient conditions for BIBO stability can be derived using the Volterra series representation (see [256], [Röt18]):

**Theorem 5.1** (BIBO stability for bilinear systems). *Let us consider a bilinear system  $\zeta = (\mathbf{A}, \mathbf{N}_j, \mathbf{B}, \mathbf{C}, \mathbf{E})$  with the following assumptions:*

- 1a.  $(\mathbf{A}, \mathbf{E})$  is Hurwitz, i.e.  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$ . The spectral abscissa  $\lambda_{\max} := \max_i \text{Re}(\lambda_i)$  is thus strictly negative.
- 1b. There exist  $\beta > 0$  and numerical abscissa  $\lambda_{\max} \leq \alpha < 0$  satisfying:  $\|e^{\tilde{\mathbf{A}}t}\| \leq \beta e^{\alpha t}$ ,  $\forall t \geq 0$ .
2. The inputs are bounded on  $[0, \infty)$  such that  $\|\mathbf{u}(t)\| = \sqrt{\sum_{j=1}^m |u_j(t)|^2} \leq M$  with  $M > 0$ .
3. We set  $\Gamma := \sum_{j=1}^m \|\mathbf{N}_j\|$ .

Then, the solution  $\mathbf{x}(t)$  or output  $\mathbf{y}(t)$  of the bilinear system is bounded on  $[0, \infty)$ , if  $\frac{M\beta\Gamma}{\alpha} < 1$ .

We can exploit the above theorem in two manners. On the one hand, we can determine whether a bilinear system is BIBO stable for given inputs with bound  $M$  by checking the

inequality  $\Gamma < \frac{\alpha}{M\beta}$ . On the other hand, we can compute an upper bound  $\|\mathbf{u}\|_\infty < \frac{\alpha}{\Gamma\beta}$  for inputs that would guarantee boundedness of the system's outputs.

The conditions for BIBO stability of bilinear systems are implemented in the BSSS function `isbstable`. It first computes the spectral (`isstable`) and numerical abscissa (`issd`), as well as the scalar  $\Gamma$ . Then, the algorithm either checks the stability condition  $\Gamma < \frac{\alpha}{M\beta}$  for given  $M$ , or returns an upper bound for the input.

`s` SSS function(s): `isstable`, `issd`, `eigs`  
`s` BSSS function(s): `isbstable`

### Gramians and bilinear Lyapunov equations

Similar to the linear case, the concepts of reachability and observability can be generalized to bilinear systems. In the following, we revisit these properties and introduce the bilinear Gramians and Lyapunov equations. [81, 221, 1, 26, 92]

A linear system is called controllable, if the controllability matrix (3.14) has full row rank  $n$ . Similarly, we can define the (somewhat stronger) *reachability* of a bilinear system as follows. Let  $\bar{\mathcal{R}}_1 = \tilde{\mathbf{B}}$  and  $\bar{\mathcal{R}}_k = [\tilde{\mathbf{A}}\bar{\mathcal{R}}_{k-1}, \tilde{\mathbf{N}}_1\bar{\mathcal{R}}_{k-1}, \dots, \tilde{\mathbf{N}}_m\bar{\mathcal{R}}_{k-1}]$  for  $k = 2, \dots, n$ . Then, the bilinear system  $\zeta$  is reachable, if and only if  $\bar{\mathcal{R}}_n$  has full row rank, i.e.  $\text{rank}(\bar{\mathcal{R}}_n) = n$ . According to [81, 1], the input-to-state part of the regular kernels is defined as

$$\begin{aligned} \mathbf{p}_1(t_1) &= e^{\tilde{\mathbf{A}}t_1} \tilde{\mathbf{B}}, & \bar{\mathbf{P}}_1(t_1) &= e^{\tilde{\mathbf{A}}t_1} \tilde{\mathbf{B}}, \\ \mathbf{p}_k^{(j_2, \dots, j_k)}(t_1, \dots, t_k) &= e^{\tilde{\mathbf{A}}t_k} \tilde{\mathbf{N}}_{j_k} \dots e^{\tilde{\mathbf{A}}t_2} \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}t_1} \tilde{\mathbf{B}}, & \bar{\mathbf{P}}_k(t_1, \dots, t_k) &= e^{\tilde{\mathbf{A}}t_k} \tilde{\mathbf{N}}(\mathbf{I}_m \otimes \bar{\mathbf{P}}_{k-1}). \end{aligned}$$

Please note that  $\tilde{\mathbf{N}} = [\tilde{\mathbf{N}}_1, \dots, \tilde{\mathbf{N}}_m] \in \mathbb{R}^{n \times n \cdot m}$  and  $\bar{\mathbf{P}}_k(t_1, \dots, t_k) \in \mathbb{R}^{n \times m^k}$ . With this, we can define the reachability Gramian  $\mathbf{P} \in \mathbb{R}^{n \times n}$  as

$$\mathbf{P} = \sum_{k=1}^{\infty} \mathbf{P}_k, \quad \text{with} \quad (5.7)$$

$$\begin{aligned} \mathbf{P}_k &= \int_{\tau_1=0}^{\infty} \dots \int_{\tau_k=0}^{\infty} \bar{\mathbf{P}}_k(\tau_1, \dots, \tau_k) \bar{\mathbf{P}}_k(\tau_1, \dots, \tau_k)^\top d\tau_k \dots d\tau_1 & (5.8) \\ &= \int_{\tau_1=0}^{\infty} \dots \int_{\tau_k=0}^{\infty} \sum_{j_2=1}^m \dots \sum_{j_k=1}^m \mathbf{p}_k^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k) \mathbf{p}_k^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k)^\top d\tau_k \dots d\tau_1. \end{aligned}$$

The bilinear system is reachable, if and only if the reachability Gramian is positive definite.

A linear system is called observable, if the observability matrix (3.14) has full column rank  $n$ . Similarly, we can also define observability for bilinear system as follows. Let  $\bar{\mathcal{O}}_1 = \mathbf{C}$  and  $\bar{\mathcal{O}}_k = [\tilde{\mathbf{A}}^\top \bar{\mathcal{O}}_{k-1}^\top, \tilde{\mathbf{N}}_1^\top \bar{\mathcal{O}}_{k-1}^\top, \dots, \tilde{\mathbf{N}}_m^\top \bar{\mathcal{O}}_{k-1}^\top]^\top$  for  $k = 2, \dots, n$ . The bilinear system  $\zeta$  is observable, if and only if  $\bar{\mathcal{O}}_n$  has full column rank, i.e.  $\text{rank}(\bar{\mathcal{O}}_n) = n$ . According to [81, 1], the state-to-output part of the regular kernels is defined in sum and Kronecker notation as

$$\begin{aligned} \mathbf{q}_1(t_1) &= e^{\tilde{\mathbf{A}}^\top t_1} \mathbf{C}^\top, & \bar{\mathbf{Q}}_1(t_1) &= e^{\tilde{\mathbf{A}}^\top t_1} \mathbf{C}^\top, \\ \mathbf{q}_k^{(j_2, \dots, j_k)}(t_1, \dots, t_k) &= e^{\tilde{\mathbf{A}}^\top t_k} \tilde{\mathbf{N}}_{j_k}^\top \dots e^{\tilde{\mathbf{A}}^\top t_2} \tilde{\mathbf{N}}_{j_2}^\top e^{\tilde{\mathbf{A}}^\top t_1} \mathbf{C}^\top, & \bar{\mathbf{Q}}_k(t_1, \dots, t_k) &= e^{\tilde{\mathbf{A}}^\top t_k} \tilde{\mathbf{N}}^{(2)}(\mathbf{I}_m \otimes \bar{\mathbf{Q}}_{k-1}). \end{aligned}$$

Please note that  $\tilde{\mathbf{N}}^{(2)} = [\tilde{\mathbf{N}}_1^\top, \dots, \tilde{\mathbf{N}}_m^\top] \in \mathbb{R}^{n \times n \cdot m}$  and  $\bar{\mathbf{Q}}_k(t_1, \dots, t_k) \in \mathbb{R}^{n \times m^{k-1} \cdot p}$ . Finally, we define the observability Gramian  $\mathbf{Q} \in \mathbb{R}^{n \times n}$  as

$$\tilde{\mathbf{Q}} = \sum_{k=1}^{\infty} \tilde{\mathbf{Q}}_k, \quad \text{with} \quad (5.9)$$

$$\begin{aligned} \tilde{\mathbf{Q}}_k &= \int_{\tau_1=0}^{\infty} \cdots \int_{\tau_k=0}^{\infty} \bar{\mathbf{Q}}_k(\tau_1, \dots, \tau_k) \bar{\mathbf{Q}}_k(\tau_1, \dots, \tau_k)^\top d\tau_k \cdots d\tau_1 \\ &= \int_{\tau_1=0}^{\infty} \cdots \int_{\tau_k=0}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \mathbf{q}_k^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k) \mathbf{q}_k^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k)^\top d\tau_k \cdots d\tau_1. \end{aligned} \quad (5.10)$$

The bilinear system is observable, if and only if the observability Gramian is positive definite.

By definition, the sub-Gramians  $\mathbf{P}_k, \tilde{\mathbf{Q}}_k$  from (5.8) and (5.10) are symmetric and positive semi-definite. However, the expressions (5.7) and (5.9) involve infinite sums. Thus, the series may diverge, and consequently  $\mathbf{P}$  and  $\tilde{\mathbf{Q}}$  may not exist. Fortunately, the following theorem (due to Zhang and Lam [285]) provides sufficient conditions for the convergence of the infinite series, and hence for the existence of the bilinear Gramians.

**Theorem 5.2** (Convergence of bilinear Gramians). *The reachability and observability Gramians  $\mathbf{P}$  and  $\tilde{\mathbf{Q}}$  given by the series (5.7) and (5.9) exist, if*

1.  $(\mathbf{A}, \mathbf{E})$  is Hurwitz. There exist  $\beta > 0$  and  $\max_i \operatorname{Re}(\lambda_i) \leq \alpha < 0$  with  $\|e^{\tilde{\mathbf{A}}t}\| \leq \beta e^{\alpha t}$ ,  $\forall t \geq 0$ .
2.  $\frac{\beta^2 \Gamma_N^2}{2\alpha} < 1$  or  $\Gamma_N < \frac{\sqrt{2\alpha}}{\beta}$ , where  $\Gamma_N = \sqrt{\|\sum_{j=1}^m \mathbf{N}_j \mathbf{N}_j^\top\|}$ .

Now let us suppose that the above conditions hold and the bilinear Gramians (5.7) and (5.9) exist. Then, it has been proved by [1] that  $\mathbf{P}$  and  $\mathbf{Q}$  (with  $\tilde{\mathbf{Q}} = \mathbf{E}^\top \mathbf{Q} \mathbf{E}$ ) satisfy the following *bilinear* Lyapunov equations

$$\mathbf{A} \mathbf{P} \mathbf{E}^\top + \mathbf{E} \mathbf{P} \mathbf{A}^\top + \sum_{j=1}^m \mathbf{N}_j \mathbf{P} \mathbf{N}_j^\top = -\mathbf{B} \mathbf{B}^\top, \quad (5.11a)$$

$$\mathbf{A}^\top \mathbf{Q} \mathbf{E} + \mathbf{E}^\top \mathbf{Q} \mathbf{A} + \sum_{j=1}^m \mathbf{N}_j^\top \mathbf{Q} \mathbf{N}_j = -\mathbf{C}^\top \mathbf{C}. \quad (5.11b)$$

In addition, every  $\mathbf{P}_k$  and  $\mathbf{Q}_k$  solves the corresponding linear Lyapunov equations

$$\mathbf{A} \mathbf{P}_1 \mathbf{E}^\top + \mathbf{E} \mathbf{P}_1 \mathbf{A}^\top = -\mathbf{B} \mathbf{B}^\top, \quad \mathbf{A} \mathbf{P}_k \mathbf{E}^\top + \mathbf{E} \mathbf{P}_k \mathbf{A}^\top = -\sum_{j=1}^m \mathbf{N}_j \mathbf{P}_{k-1} \mathbf{N}_j^\top, \quad k \geq 2, \quad (5.12a)$$

$$\mathbf{A}^\top \mathbf{Q}_1 \mathbf{E} + \mathbf{E}^\top \mathbf{Q}_1 \mathbf{A} = -\mathbf{C}^\top \mathbf{C}, \quad \mathbf{A}^\top \mathbf{Q}_k \mathbf{E} + \mathbf{E}^\top \mathbf{Q}_k \mathbf{A} = -\sum_{j=1}^m \mathbf{N}_j^\top \mathbf{Q}_{k-1} \mathbf{N}_j, \quad k \geq 2. \quad (5.12b)$$

**Remark 5.1** (Solution of bilinear Lyapunov equations vs. existence of Gramians). As pointed out in [285], it might happen that the Lyapunov equations (5.11) have a unique solution, although the bilinear Gramians (5.7) and (5.9) do not exist/converge. In this case, the solutions  $\mathbf{P}$  and  $\mathbf{Q}$  are not positive semi-definite and do not correspond to the reachability and observability Gramians. In other words: only if the bilinear Lyapunov equations (5.11) have unique, *positive (semi)-definite* solutions  $\mathbf{P} = \mathbf{P}^\top \succeq \mathbf{0}$ ,  $\mathbf{Q} = \mathbf{Q}^\top \succeq \mathbf{0}$ , then (5.7) and (5.9) converge to the reachability and observability Gramians.  $\triangle$

### Solving large-scale bilinear Lyapunov equations

In the context of Lyapunov equations for bilinear systems one has to distinguish between two different strategies. On the one hand, we can solve the bilinear Lyapunov equations (5.11) to compute the *infinite* Gramians  $\mathbf{P}$  and  $\tilde{\mathbf{Q}}$ . On the other hand, one can also solve the cascaded sequence of linear Lyapunov equations (5.12) to calculate the so-called *truncated* Gramians  $\mathbf{P}_{1:N} = \sum_{k=1}^N \mathbf{P}_k$  and  $\tilde{\mathbf{Q}}_{1:N} = \sum_{k=1}^N \tilde{\mathbf{Q}}_k$ . Depending on the convergence of the Volterra series for the bilinear system at hand, it might be necessary to consider the whole series or it can be sufficient to take only the leading  $N$  subsystems into account.

There exist several approaches to solve the bilinear Lyapunov equations (5.11). The most straightforward method consists in reformulating them as linear systems of  $n^2$  equations using the vectorization property (2.13). This yields:

$$\left( \mathbf{E} \otimes \mathbf{A} + \mathbf{A} \otimes \mathbf{E} + \sum_{j=1}^m \mathbf{N}_j \otimes \mathbf{N}_j \right) \text{vec}(\mathbf{P}) = -\text{vec}(\mathbf{B}\mathbf{B}^\top), \quad (5.13a)$$

$$\left( \mathbf{E}^\top \otimes \mathbf{A}^\top + \mathbf{A}^\top \otimes \mathbf{E}^\top + \sum_{j=1}^m \mathbf{N}_j^\top \otimes \mathbf{N}_j^\top \right) \text{vec}(\mathbf{Q}) = -\text{vec}(\mathbf{C}^\top \mathbf{C}). \quad (5.13b)$$

These LSEs can then be solved by direct methods (e.g. \), followed by reshaping  $\text{vec}(\mathbf{P})$  and  $\text{vec}(\mathbf{Q})$  into respective  $n \times n$  matrices. This simple procedure is implemented in the BSSS function `blyap`. However, this approach is only suitable for small-scale systems due to the heavily increasing dimension of the vectorized matrices ( $n^2 \times n^2$ ) and its associated huge storage effort. Hence, tensorized low-rank versions of well-known iterative LSE solvers (e.g. CG, GMRES) should be applied in the large-scale setting (see [150, 21] for more details).

Another approach consists in generalizing low-rank matrix equation solvers like the LR-ADI and the EKSM/RKSM (cf. Section 2.4) to the bilinear case [77, 21], [45, Sec. 4.4]. These methods have not been implemented in the BSSS toolbox so far. Nevertheless, we believe that the Krylov methods implemented for *reduction* purposes in Sections 5.3 and 5.4 can facilitate the implementation of EKSM/RKSM for Lyapunov equation purposes. Crucial is also the low-rank implementation of the bilinear residual  $\mathbf{Res}_k$  (cf. Eq. (2.41)) as stopping criterion

$$\mathbf{Res}_k = \mathbf{A}\mathbf{V}_k\mathbf{P}_{r,k}\mathbf{V}_k^\top\mathbf{E}^\top + \mathbf{E}\mathbf{V}_k\mathbf{P}_{r,k}\mathbf{V}_k^\top\mathbf{A}^\top + \sum_{j=1}^m \mathbf{N}_j\mathbf{V}_k\mathbf{P}_{r,k}\mathbf{V}_k^\top\mathbf{N}_j^\top + \mathbf{B}\mathbf{B}^\top, \quad (5.14)$$

with the approximate solution  $\hat{\mathbf{P}}_k = \mathbf{V}_k\mathbf{P}_{r,k}\mathbf{V}_k^\top$  and the solution  $\mathbf{P}_{r,k}$  of the reduced bilinear Lyapunov equation at step  $k$ .

A further scheme for solving bilinear Lyapunov equations is presented in [257]. The main idea of the proposed method is to calculate an approximate solution by generating a (stationary) sequence of solutions  $\mathbf{P}_k$  via the *fixed-point iteration*



$$\begin{aligned} \mathcal{L}_A(\mathbf{P}_1) &= -\mathbf{B}\mathbf{B}^\top & \iff & \mathbf{A}\mathbf{P}_1\mathbf{E}^\top + \mathbf{E}\mathbf{P}_1\mathbf{A}^\top = -\mathbf{B}\mathbf{B}^\top, \\ \mathcal{L}_A(\mathbf{P}_k) &= -\Pi(\mathbf{P}_{k-1}) - \mathbf{B}\mathbf{B}^\top & \iff & \mathbf{A}\mathbf{P}_k\mathbf{E}^\top + \mathbf{E}\mathbf{P}_k\mathbf{A}^\top = -\mathbf{B}_k\mathbf{B}_k^\top, \quad k = 2, \dots \end{aligned} \quad (5.15)$$

with  $\mathcal{L}_A(\mathbf{P}) := \mathbf{A}\mathbf{P}\mathbf{E}^\top + \mathbf{E}\mathbf{P}\mathbf{A}^\top$  and  $\Pi_N(\mathbf{P}) := \sum_{j=1}^m \mathbf{N}_j\mathbf{P}\mathbf{N}_j^\top$ . The iteration is expected to converge to the positive semi-definite solution  $\mathbf{P}$  of (5.11a), if  $\lambda(\mathcal{L}_A) \subset \mathbb{C}_-$  and the spectral radius  $\rho(\mathcal{L}_A^{-1}\Pi_N) < 1$ . The *linear* Lyapunov equations (5.15) are then solved for the low-



rank factor  $\mathbf{P}_k \approx \mathbf{Z}_k \mathbf{Z}_k^\top$ , where the right-hand side reads  $\mathbf{B}_k = [\mathbf{N}_1 \mathbf{Z}_{k-1}, \dots, \mathbf{N}_m \mathbf{Z}_{k-1}, \mathbf{B}]$ . Consequently, the procedure is composed of two stages. In an inner loop, the linear Lyapunov equations are solved iteratively via conventional LR-ADI or RKSM to obtain the current low-rank factor  $\mathbf{Z}_k$ . In an outer loop, the previous factor  $\mathbf{Z}_{k-1}$  is employed to update the right-hand side  $-\mathbf{B}_k \mathbf{B}_k^\top$  for the next Lyapunov equation, as well as to evaluate the norm  $\|\mathbf{B}_k \mathbf{B}_k^\top\|$  and/or  $\|\mathbf{P}_k - \mathbf{P}_{k-1}\|$  for the stopping criterion. The whole scheme has been implemented in the BSSS function `blyapchol` during the semester thesis [Hei17]. The main framework – with inner/outer loop, matrix truncation  $\mathcal{T}(\Pi_N(\mathbf{Z}_{k-1}) - \mathbf{B}\mathbf{B}^\top)$  and adaptive residual tolerance – is contained in `blyapchol`. The algorithm then calls the SSS function `lyapchol` (i.e. `mess_lradi` or `crksm`) to solve the linear Lyapunov equations within the inner loop. In this regard, our algorithm is more general, modular and flexible than the proposed one [257], since it can exploit all features implemented in `mess_lradi` or `crksm` (e.g. shifts adaption). The reader is referred to [Hei17] for more implementation details and for numerical examples.

Instead of solving the bilinear Lyapunov equations (5.11), one can alternatively solve the cascaded series of linear Lyapunov equations (5.12) until the truncation index  $N$ . The functions `mess_lradi` or `crksm` integrated in `lyapchol` can again be exploited to solve these linear Lyapunov equations. Although this procedure is generally more efficient than solving the bilinear equations, note that it only yields *truncated* Gramians. Both the computational effort and the obtained accuracy basically depend on the number  $N$  of considered subsystems. One way to determine an appropriate truncation index  $N$  consists in substituting the truncated Gramian  $\mathbf{P}_{1:N} = \sum_{k=1}^N \mathbf{P}_k$  into the bilinear Lyapunov equation to compute the norm of  $\mathbf{Res}_{1:N} = \mathbf{A}\mathbf{P}_{1:N}\mathbf{E}^\top + \mathbf{E}\mathbf{P}_{1:N}\mathbf{A}^\top + \sum_{j=1}^m \mathbf{N}_j \mathbf{P}_{1:N} \mathbf{N}_j^\top + \mathbf{B}\mathbf{B}^\top$ . If  $\|\mathbf{Res}_{1:N}\| < \text{tol}$  for small tolerance, then we can stop considering more subsystems. Since the computation of the true norm  $\|\mathbf{Res}_{1:N}\|$  is expensive, one can also try the heuristic measure  $\|\mathbf{P}_N - \mathbf{P}_{N-1}\|$  or the difference between the cumulative sums  $\|\mathbf{P}_{1:N} - \mathbf{P}_{1:N-1}\|$ .

 SSS function(s): `lyapchol` (`mess_lradi`, `crksm`)  
 BSSS function(s): `blyap`, `blyapchol`

## System norms

We now discuss system norms for bilinear systems, which will later allow us to measure the difference between full- and reduced-order model. Since we are particularly interested in  $\mathcal{H}_2$ -optimal model reduction approaches, in the following we only focus on the  $\mathcal{H}_2$ -norm.

### $\mathcal{L}_2$ -norm (time-domain)

Similar to the linear case (cf. Eq. (3.19)), the time-domain  $\mathcal{L}_2$ -norm of a MIMO bilinear system  $\zeta$  is defined as [285, 92]

$$\|\zeta\|_{\mathcal{L}_2}^2 = \sum_{k=1}^{\infty} \int_{\tau_1=0}^{\infty} \cdots \int_{\tau_k=0}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \|\mathbf{g}_{k,\square}^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k)\|_{\text{F}}^2 d\tau_1 \cdots d\tau_k. \quad (5.16)$$

Hereby,  $\|\mathbf{g}_{k,\square}^{(j_2, \dots, j_k)}(\tau_1, \dots, \tau_k)\|_{\text{F}}$  denotes the Frobenius norm of the  $k$ -th regular kernel (4.39). Obviously, the  $\mathcal{L}_2$ -norm exists only if the Volterra series converges, i.e. if the bilinear system  $\zeta$  is BIBO stable and consequently the Gramians exist. In such case, we can use the link

between the Frobenius norm and the trace of a matrix  $\|\mathbf{K}\|_{\text{F}}^2 = \text{tr}(\mathbf{K}\mathbf{K}^{\text{T}}) = \text{tr}(\mathbf{K}^{\text{T}}\mathbf{K})$  to express the  $\mathcal{L}_2$ -norm in terms of the reachability and observability Gramians. This yields:

$$\begin{aligned}
\|\zeta\|_{\mathcal{L}_2}^2 &= \sum_{k=1}^{\infty} \text{tr} \left( \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=0}^{\infty} \cdots \int_{\tau_k=0}^{\infty} \mathbf{g}_{k,\square}^{(j_2,\dots,j_k)}(\tau_1, \dots, \tau_k) \mathbf{g}_{k,\square}^{(j_2,\dots,j_k)}(\tau_1, \dots, \tau_k)^{\text{T}} d\tau_1 \cdots d\tau_k \right) \\
&= \text{tr} \left( \underbrace{\mathbf{C} \int_{\tau_1=0}^{\infty} e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{B}} \tilde{\mathbf{B}}^{\text{T}} e^{\tilde{\mathbf{A}}^{\text{T}}\tau_1} d\tau_1}_{\mathbf{P}_1} \mathbf{C}^{\text{T}} \right) \\
&\quad + \text{tr} \left( \underbrace{\mathbf{C} \int_{\tau_1=0}^{\infty} \int_{\tau_2=0}^{\infty} \sum_{j_2=1}^m e^{\tilde{\mathbf{A}}\tau_2} \tilde{\mathbf{N}}_{j_2} e^{\tilde{\mathbf{A}}\tau_1} \tilde{\mathbf{B}} \tilde{\mathbf{B}}^{\text{T}} e^{\tilde{\mathbf{A}}^{\text{T}}\tau_1} \tilde{\mathbf{N}}_{j_2}^{\text{T}} e^{\tilde{\mathbf{A}}^{\text{T}}\tau_2} d\tau_1 d\tau_2}_{\mathbf{P}_2} \mathbf{C}^{\text{T}} \right) + \dots \\
&= \text{tr}(\mathbf{C}\mathbf{P}\mathbf{C}^{\text{T}}). \tag{5.17}
\end{aligned}$$

Thus, we can compute the  $\mathcal{L}_2$ -norm with the reachability Gramian (5.7). Similarly, it holds

$$\begin{aligned}
\|\zeta\|_{\mathcal{L}_2}^2 &= \sum_{k=1}^{\infty} \text{tr} \left( \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \int_{\tau_1=0}^{\infty} \cdots \int_{\tau_k=0}^{\infty} \mathbf{g}_{k,\square}^{(j_2,\dots,j_k)}(\tau_1, \dots, \tau_k)^{\text{T}} \mathbf{g}_{k,\square}^{(j_2,\dots,j_k)}(\tau_1, \dots, \tau_k) d\tau_1 \cdots d\tau_k \right) \\
&= \text{tr}(\tilde{\mathbf{B}}^{\text{T}} \tilde{\mathbf{Q}} \tilde{\mathbf{B}}) = \text{tr}(\mathbf{B}^{\text{T}} \mathbf{Q} \mathbf{B}), \tag{5.18}
\end{aligned}$$

where  $\mathbf{Q} = \mathbf{E}^{-\text{T}} \tilde{\mathbf{Q}} \mathbf{E}^{-1}$  represents the observability Gramian (5.9). In other words: provided that the Gramians exist, the time-domain  $\mathcal{L}_2$ -norm of a bilinear system  $\zeta$  can be computed by solving one of the bilinear Lyapunov equations (5.11).

### $\mathcal{H}_2$ -norm (frequency-domain)

According to [92], the Hardy space  $\mathcal{H}_2$  can also be generalized to bilinear systems. This yields the  $\mathcal{H}_2$ -norm

$$\|\zeta\|_{\mathcal{H}_2}^2 = \sum_{k=1}^{\infty} \sup_{x_1 > 0, \dots, x_k > 0} \int_{y_1 = -\infty}^{\infty} \cdots \int_{y_k = -\infty}^{\infty} \|\mathbf{G}_k^{\square}(x_1 + iy_1, \dots, x_k + iy_k)\|_{\text{F}}^2 dy_1 \cdots dy_k, \tag{5.19}$$

where  $\mathbf{G}_k^{\square}(s_1, \dots, s_k)$  denotes the  $k$ -th regular transfer function (4.43) that is analytic in  $\mathbb{C}_+^k$ . If  $(\mathbf{A}, \mathbf{E})$  is Hurwitz, i.e.  $\lambda(\mathbf{E}^{-1}\mathbf{A}) \subset \mathbb{C}_-$ , then the Phragmen-Lindelöf principle states that the  $\mathcal{H}_2$ -norm  $\|\zeta\|_{\mathcal{H}_2}^2$  is equivalent to the frequency-domain  $\mathcal{L}_2$ -norm  $\|\zeta\|_{\mathcal{L}_2(i\mathbb{R})}^2$  on the imaginary axis. Applying again the relation  $\|\mathbf{G}_k^{\square}(s_1, \dots, s_k)\|_{\text{F}}^2 = \text{tr}(\mathbf{G}_k^{\square}(s_1, \dots, s_k) \mathbf{G}_k^{\square}(s_1, \dots, s_k)^{\text{T}}) = \text{tr}(\mathbf{G}_k^{\square}(s_1, \dots, s_k)^{\text{T}} \mathbf{G}_k^{\square}(s_1, \dots, s_k))$ , we can rewrite (5.19) in sum notation as

$$\begin{aligned}
\|\zeta\|_{\mathcal{H}_2}^2 &= \sum_{k=1}^{\infty} \frac{1}{(2\pi)^k} \text{tr} \left( \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(i\omega_1, \dots, i\omega_k) \right. \\
&\quad \left. \times \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(-i\omega_1, \dots, -i\omega_k)^{\text{T}} d\omega_1 \cdots d\omega_k \right). \tag{5.20}
\end{aligned}$$

Due to Parseval-Plancherel's theorem, the frequency-domain and time-domain  $\mathcal{L}_2$ -norms are also equivalent. In short: [92]

$$\|\zeta\|_{\mathcal{H}_2}^2 = \|\zeta\|_{\mathcal{L}_2(i\mathbb{R})}^2 = \|\zeta\|_{\mathcal{L}_2(0,\infty)}^2.$$

The  $\mathcal{H}_2$ -norm of a bilinear system can also be expressed in pole-residue formulation. While this has been stated for the SISO case in several publications (e.g. [92], [45]), the MIMO case has not been considered so far. Following the SISO derivation in [92] and making use of the matrix-residues (5.5), we can state our new result:

$$\|\zeta\|_{\mathcal{H}_2}^2 = \sum_{k=1}^{\infty} \sum_{l_1=1}^n \cdots \sum_{l_k=1}^n \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \hat{n}_{j_k l_k, l_{k-1}} \cdots \hat{n}_{j_2 l_2, l_1} \hat{\mathbf{c}}_{l_k}^T \mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{l_1}, \dots, -\bar{\lambda}_{l_k}) \hat{\mathbf{b}}_{l_1}. \quad (5.21)$$

### Truncated $\mathcal{H}_2$ -norm

There are again two different ways to proceed in order to compute the  $\mathcal{H}_2$ -norm of a bilinear system. On the one hand, we can solve one of the *bilinear* Lyapunov equations (5.11) to calculate the *infinite*  $\mathcal{H}_2$ -norm (`bnorm`, `blyap`, `blyapchol`). On the other hand, we can consider only the first  $N$  terms of the Volterra series and define a *truncated*  $\mathcal{H}_2$ -norm  $\|\zeta\|_{\mathcal{H}_2, 1:N}^2 = \text{tr}(\mathbf{C}\mathbf{P}_{1:N}\mathbf{C}^T) = \text{tr}(\mathbf{B}^T\mathbf{Q}_{1:N}\mathbf{B})$  using the truncated Gramians  $\mathbf{P}_{1:N}$  or  $\mathbf{Q}_{1:N}$  from (5.12). The series in (5.21) can also be truncated to calculate the  $\mathcal{H}_2$ -norm. This involves the computation of (a few) eigenvalues  $\lambda_{l_k}$  with their corresponding  $\hat{\mathbf{c}}_{l_k}^T$ ,  $\hat{\mathbf{b}}_{l_k}$  and  $\hat{n}_{j_k l_k, l_{k-1}}$  (`eigs`), together with the evaluation of the transfer functions  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  at several combinations of the mirrored eigenvalues ( $\setminus$ ).

`s` SSS function(s): `norm` (`lyapchol`)  
`s` BSSS function(s): `bnorm` (`blyap`, `blyapchol`)

## 5.2 Bilinear model order reduction

Bilinear systems have been mainly reduced so far using the same *projection-based* framework as for LTI systems (cf. Section 3.2). In this section, we first describe this projective reduction approach for bilinear systems, introduce the error system and then focus on two different strategies to compute the reduction bases needed for projection.

Let us assume that the state vector  $\mathbf{x}(t) \in \mathbb{R}^n$  of the bilinear system (5.1) mainly evolves in a *linear*  $r$ -dimensional subspace  $\mathcal{V} = \text{ran}(\mathbf{V})$  spanned by the full column rank matrix  $\mathbf{V} \in \mathbb{R}^{n \times r}$ . Inserting the ansatz (3.26) with  $\mathbf{x}(t) = \mathbf{V}\mathbf{x}_r(t) + \mathbf{e}(t)$  into (5.1a) yields an overdetermined system

$$\mathbf{E}\mathbf{V}\dot{\mathbf{x}}_r(t) = \mathbf{A}\mathbf{V}\mathbf{x}_r(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{V}\mathbf{x}_r(t)u_j(t) + \mathbf{B}\mathbf{u}(t) + \boldsymbol{\varepsilon}(t), \quad (5.22)$$

with the residual  $\boldsymbol{\varepsilon}(t) = \mathbf{A}\mathbf{e}(t) - \mathbf{E}\dot{\mathbf{e}}(t) + \sum_{j=1}^m \mathbf{N}_j \mathbf{e}(t)u_j(t) \in \mathbb{R}^n$ . To obtain a well-determined ROM, we project the resulting system onto the subspace  $\mathcal{U} = \text{ran}(\mathbf{E}\mathbf{V})$  orthogonally to another subspace  $\mathcal{W} = \text{ran}(\mathbf{W})$  spanned by the full rank matrix  $\mathbf{W} \in \mathbb{R}^{n \times r}$ . This is done by

premultiplying the overdetermined system (5.22) with the projector  $\mathbf{\Pi} = \mathbf{E}\mathbf{V}(\mathbf{W}^\top \mathbf{E}\mathbf{V})^{-1} \mathbf{W}^\top$ , where  $\mathbf{W}^\top \mathbf{E}\mathbf{V}$  is assumed non-singular:

$$\mathbf{\Pi} \left( \underbrace{\mathbf{E}\mathbf{V}\dot{\mathbf{x}}_r(t) - \mathbf{A}\mathbf{V}\mathbf{x}_r(t) - \sum_{j=1}^m \mathbf{N}_j \mathbf{V}\mathbf{x}_r(t) u_j(t) - \mathbf{B}\mathbf{u}(t) - \boldsymbol{\varepsilon}(t)}_{=\boldsymbol{\xi}(\mathbf{v}_{\mathbf{x}_r(t)}, \mathbf{u}(t))} \right) = \mathbf{0} \Leftrightarrow \mathbf{\Pi} \left( \boldsymbol{\xi}(\cdot, \cdot) - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0}.$$

Enforcing the Petrov-Galerkin condition  $\mathbf{W}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\mathbf{\Pi} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , only the term  $\mathbf{\Pi} \boldsymbol{\xi}(\cdot, \cdot) = \mathbf{0}$  remains. Omitting the preceding factor  $\mathbf{E}\mathbf{V}(\mathbf{W}^\top \mathbf{E}\mathbf{V})^{-1}$  finally yields the ROM

$$\zeta_r : \begin{cases} \mathbf{E}_r \dot{\mathbf{x}}_r(t) = \mathbf{A}_r \mathbf{x}_r(t) + \sum_{j=1}^m \mathbf{N}_{j,r} \mathbf{x}_r(t) u_j(t) + \mathbf{B}_r \mathbf{u}(t), & \mathbf{x}_r(0) = \mathbf{x}_{r,0}, \\ \mathbf{y}_r(t) = \mathbf{C}_r \mathbf{x}_r(t), \end{cases} \quad (5.23a)$$

$$(5.23b)$$

with reduced matrices  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E}\mathbf{V}$ ,  $\mathbf{A}_r = \mathbf{W}^\top \mathbf{A}\mathbf{V}$ ,  $\mathbf{N}_{j,r} = \mathbf{W}^\top \mathbf{N}_j \mathbf{V}$  for  $j = 1, \dots, m$ ,  $\mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$  and  $\mathbf{C}_r = \mathbf{C}\mathbf{V}$ , as well as the initial condition  $\mathbf{x}_r(0) = (\mathbf{W}^\top \mathbf{E}\mathbf{V})^{-1} \mathbf{W}^\top \mathbf{E}\mathbf{x}(0)$ . Similar to the full-order model (5.1), the abbreviation  $\zeta_r = (\mathbf{A}_r, \mathbf{N}_{j,r}, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r)$  will denote the reduced bilinear system (5.23) in the following.

The main task in this setting consists in computing suitable reduction bases  $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r}$  to construct the reduced model in a projective manner. As in the linear case, the specific choice of the bases (e.g. real-valued, orthonormal, biorthonormal) can be exploited for numerical reasons, but does not affect the spanned subspaces  $\mathcal{V}, \mathcal{W}$ . Furthermore, we may perform either an orthogonal Galerkin projection with  $\text{ran}(\mathbf{W}) = \text{ran}(\mathbf{V})$  or an oblique Petrov-Galerkin projection with  $\text{ran}(\mathbf{W}) \neq \text{ran}(\mathbf{V})$  depending on the desired accuracy or system properties of the ROM. Since the assessment of the approximation quality of bilinear ROMs is a little different than in the linear case, we will focus on this in the following.

## Error system and error system norms

The most straightforward way to measure the accuracy of a bilinear ROM is by means of the output error  $\mathbf{e}_y(t) := \mathbf{y}(t) - \mathbf{y}_r(t)$  or state error  $\mathbf{e}_x(t) := \mathbf{x}(t) - \mathbf{V}\mathbf{x}_r(t)$  in *time-domain* (cf. Section 2.6). This, however, involves the simulation of both systems for a test input signal  $\mathbf{u}_{\text{test}}(t)$  and only delivers an error measure for that specific input. Another conceivable approach could be to calculate the error between the  $k$ -th full- and reduced transfer functions using the *point-wise* in frequency measure  $\|\mathbf{G}_k^\square(i\omega_1, \dots, i\omega_k) - \mathbf{G}_{k,r}^\square(i\omega_1, \dots, i\omega_k)\|_{(*)}$  (cf. Eq. (3.31)). Nevertheless, the number of frequency point combinations rapidly grows by the power  $k$  and only a finite number  $N$  of subsystems can be considered.

Similar to the linear case, it would be desirable to define a bilinear error system with its system norm such that the *overall* approximation error can be quantified without performing simulation runs for certain inputs. Fortunately, this is possible thanks to the Volterra series representation and the generalized system-theoretic concepts presented in the previous section.

The *bilinear* error system is defined as  $\zeta_e := \zeta - \zeta_r = (\mathbf{A}_e, \mathbf{N}_{j,e}, \mathbf{B}_e, \mathbf{C}_e, \mathbf{E}_e)$ , where

$$\mathbf{E}_e = \begin{bmatrix} \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_r \end{bmatrix}, \quad \mathbf{A}_e = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_r \end{bmatrix}, \quad \mathbf{N}_{j,e} = \begin{bmatrix} \mathbf{N}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_{j,r} \end{bmatrix}, \quad \mathbf{B}_e = \begin{bmatrix} \mathbf{B} \\ \mathbf{B}_r \end{bmatrix}, \quad \mathbf{C}_e = \begin{bmatrix} \mathbf{C} & -\mathbf{C}_r \end{bmatrix}.$$

The error between FOM and ROM can then be measured *norm-wise* in frequency-domain by  $\|\zeta_e\|_{\mathcal{H}_p}^2 = \|\zeta - \zeta_r\|_{\mathcal{H}_p}^2$ , where we consider here only the  $\mathcal{H}_2$ -norm. By applying the definition together with the expressions (5.17) and (5.18), we can write the  $\mathcal{H}_2$ -error as [285]

$$\|\zeta_e\|_{\mathcal{H}_2}^2 = \text{tr} \left( \mathbf{C}_e \mathbf{P}_e \mathbf{C}_e^\top \right) = \text{tr} \left( \mathbf{B}_e^\top \mathbf{Q}_e \mathbf{B}_e \right), \quad (5.24)$$

where  $\mathbf{P}_e$  and  $\mathbf{Q}_e$  are obtained by solving the bilinear Lyapunov equations

$$\mathbf{A}_e \mathbf{P}_e \mathbf{E}_e^\top + \mathbf{E}_e \mathbf{P}_e \mathbf{A}_e^\top + \sum_{j=1}^m \mathbf{N}_{j,e} \mathbf{P}_e \mathbf{N}_{j,e}^\top = -\mathbf{B}_e \mathbf{B}_e^\top, \quad (5.25a)$$

$$\mathbf{A}_e^\top \mathbf{Q}_e \mathbf{E}_e + \mathbf{E}_e^\top \mathbf{Q}_e \mathbf{A}_e + \sum_{j=1}^m \mathbf{N}_{j,e}^\top \mathbf{Q}_e \mathbf{N}_{j,e} = -\mathbf{C}_e^\top \mathbf{C}_e. \quad (5.25b)$$

Again, one can solve either the bilinear Lyapunov equations (5.25) to compute the *infinite*  $\mathcal{H}_2$ -error  $\|\zeta_e\|_{\mathcal{H}_2}^2$  or consider only the first  $N$  subsystems to calculate the *truncated*  $\mathcal{H}_2$ -error  $\|\zeta_e\|_{\mathcal{H}_{2,1:N}}^2 = \text{tr}(\mathbf{C}_e \mathbf{P}_{e,1:N} \mathbf{C}_e^\top) = \text{tr}(\mathbf{B}_e^\top \mathbf{Q}_{e,1:N} \mathbf{B}_e)$ . Note that this latter approach is equivalent to the computation and summation over the norm-wise errors  $\|\mathbf{G}_1 - \mathbf{G}_{1,r}\|_{\mathcal{H}_2} = \text{tr}(\mathbf{C}_e \mathbf{P}_{e,1} \mathbf{C}_e^\top)$ ,  $\|\mathbf{G}_2^\square - \mathbf{G}_{2,r}^\square\|_{\mathcal{H}_2} = \text{tr}(\mathbf{C}_e \mathbf{P}_{e,2} \mathbf{C}_e^\top), \dots, \|\mathbf{G}_N^\square - \mathbf{G}_{N,r}^\square\|_{\mathcal{H}_2} = \text{tr}(\mathbf{C}_e \mathbf{P}_{e,N} \mathbf{C}_e^\top)$ .

To the best of the author's knowledge, a relation between the  $\mathcal{H}_2$ -error norm in frequency-domain and the output error in time-domain similar to (3.35) has not been derived for bilinear systems yet. However, precisely such an inequality would be helpful in order to interpret the measure  $\|\zeta_e\|_{\mathcal{H}_2}$  as an upper bound for the output error.

### 5.2.1 Bilinear balanced truncation

One possible way to compute the reduction bases  $\mathbf{V}, \mathbf{W}$  needed for projection is given by the method of balanced truncation (cf. Section 3.3.2). This approach was first generalized to bilinear systems in [1] based on the algebraic Gramians defined in [81] (cf. Eqs. (5.7), (5.9)).

Luckily, the idea of bringing the bilinear system into a balanced realization  $\zeta_{\text{bal}} = (\mathbf{T} \mathbf{A} \mathbf{T}^{-1}, \mathbf{T} \mathbf{N}_j \mathbf{T}^{-1}, \mathbf{T} \mathbf{B}, \mathbf{C} \mathbf{T}^{-1}, \mathbf{T} \mathbf{E} \mathbf{T}^{-1})$ , where the Gramians  $\mathbf{P} = \tilde{\mathbf{Q}} = \mathbf{\Sigma} = \text{diag}(\varsigma_1, \dots, \varsigma_n)$  are equal and diagonal with  $\varsigma_i := \sqrt{\lambda_i(\mathbf{P} \mathbf{E}^\top \mathbf{Q} \mathbf{E})}$ , remains the same as in the linear case. Unfortunately, the assessment of controllability and observability via energy measures is much more complicated in the nonlinear case [239]. In fact, the energy functions  $\mathcal{J}_c(\mathbf{x})$  and  $\mathcal{J}_o(\mathbf{x})$  for bilinear systems should satisfy certain nonlinear partial differential equations, with gradients  $\frac{\partial \mathcal{J}_c(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{P}(\mathbf{x})^{-1} \mathbf{x}$  and  $\frac{\partial \mathcal{J}_o(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{Q}(\mathbf{x}) \mathbf{x}$  given in terms of *state-dependent* Gramians<sup>1</sup> [102]. The *algebraic* bilinear Gramians  $\mathbf{P}, \tilde{\mathbf{Q}}$  from (5.7), (5.9) cannot exactly determine the energy functions  $\mathcal{J}_c(\mathbf{x})$  and  $\mathcal{J}_o(\mathbf{x})$ . Nonetheless, they can at least provide *bounds* on the energy functions for sufficiently small inputs (cf. [102, 26]). However, as observed in [66], these Gramians are not invariant under time-scaling ( $t \rightarrow \alpha t$ ) and input-scaling ( $\mathbf{u} \rightarrow \gamma \mathbf{u}$ ) transformations. Therefore, [66] proposed an alternative formulation for algebraic Gramians by introducing a parameter  $\xi$  in the bilinear Lyapunov equations (5.11). Despite this reasonable approach to account for important invariant properties and the nonlinear behavior, it should be noted that balanced truncation based on the algebraic Gramians (5.7) and (5.9) has established instead. Only in case of convergence issues, a rescaling of the system is performed (cf. Remark 5.2).

<sup>1</sup>The reader is referred to [Hei17, Sec. 2.2.3] and [Hei18, Ch. 4] for further details on these and related topics.

Bilinear balanced truncation (**btbr**) consists of the *same* three steps described in Algorithm 3.1, where in step 1 the *bilinear* Lyapunov equations (5.11) should be solved instead. For this, the implemented BSSS functions **blyap** or **blyapchol** can be used, whereas the latter provides low-rank factors for the bilinear Gramians. Similar as with the  $\mathcal{H}_2$ -norm, it is also conceivable to perform *truncated* bilinear balanced truncation (**tb\_tbr**) by considering only the truncated Gramians  $\mathbf{P}_{1:N}$  and  $\mathbf{Q}_{1:N}$  from the Lyapunov equations (5.12). These equations can be easily solved with the SSS function **lyapchol**.

In the author's opinion, **btbr** constitutes a more sound extension of balanced truncation to the bilinear setting, and should therefore generally yield better and more global results than **tb\_tbr**. Although **tb\_tbr** is computationally cheaper and has sometimes even outperformed the accuracy of **btbr** in numerical examples [103, Sec. 3.4, 3.5], its performance crucially depends on the exciting input amplitude and the number  $N$  of considered subsystems. This should be investigated in detail with more realistic FE examples in the future, in order to assess the applicability of **tb\_tbr**. The author of this thesis also pleads for a fair comparison between both approaches by using the same low-rank matrix equation solver within **blyapchol** for (5.11) and **lyapchol** for (5.12). This can be easily accomplished with our flexible and modular implementations, which exploit either **mess\_lradi** or **crksm**.

Finally, note that bilinear balanced truncation does not necessarily yield a balanced ROM. Moreover, unlike the linear case, it is not possible to quantify the error in terms of the singular values  $\varsigma_i$  with a similar bound as in (3.42). After all, in the nonlinear case it seems difficult to get a computationally feasible method, which also provides all theoretically useful properties.

**Remark 5.2** (Rescaling of bilinear system). We have discussed sufficient conditions for BIBO stability and convergence of the bilinear Gramians in Theorems 5.1 and 5.2, respectively. In general terms, the convergence depends on the input's amplitude and the norms  $\|\mathbf{N}_j\|$ . If the bilinear system  $\zeta$  at hand does not fulfill these conditions, then the Gramians do not exist and the solutions  $\mathbf{P}$ ,  $\mathbf{Q}$  of the bilinear Lyapunov equations are indefinite. However, we may scale the system as  $\zeta \rightarrow \zeta_\gamma = (\mathbf{A}, \gamma\mathbf{N}_j, \gamma\mathbf{B}, \mathbf{C}, \mathbf{E})$  via the input  $\mathbf{u}(t) \rightarrow \gamma\mathbf{u}_\gamma(t)$ , where  $\gamma > 0$  is chosen sufficiently small to guarantee the existence of the Gramians [26]. Then, the computation of the reduction matrices  $\mathbf{V}$ ,  $\mathbf{W}$  can be carried out based on  $\zeta_\gamma$ , while the original behavior can be recovered by applying the input  $\frac{1}{\gamma}\mathbf{u}(t)$  to the FOM  $\zeta$  and ROM  $\zeta_r$ . This technique has been applied in the context of bilinear balanced truncation, and also in the field of  $\mathcal{H}_2$ -optimal reduction, to enable the analysis of a broader set of bilinear systems. Note, however, that the analysis is then confined to a *small signal behavior*.  $\triangle$

## 5.2.2 Bilinear Krylov-based reduction

Another way of reducing bilinear systems consists in applying Krylov- or interpolation-based techniques. In this context, three different approaches known from the linear case have been successfully extended to the bilinear setting.

1. *Rational Krylov*: Extension of moment matching and Krylov subspaces based on *regular* transfer functions. Two different concepts can be distinguished:
  - *Subsystem interpolation*: Interpolation of some transfer functions [205, 47, 25]
  - *Volterra series interpolation*: Interpolation of the whole Volterra series [91, 2]
2.  *$\mathcal{H}_2$ -optimal reduction*: Extension of optimality conditions and algorithms [285, 20, 91]
3. *Loewner framework*: Extension of low-order identification approach using frequency- or time-domain data [92, Ch. 6], [4, 143]

Since system identification is not topic of this thesis, we will focus only on the methods 1 and 2 in the sequel. In general, all mentioned Krylov approaches rely (in one way or another) on the concepts of transfer functions, moments,  $\mathcal{H}_2$ -norm and Sylvester equations. Especially the idea of subsystem interpolation is heavily based on the moments of the regular transfer functions (4.40). Thus, similar to Definition 3.4, we first revisit the concept of multimoments and Markov parameters of bilinear systems [205, 25].

**Definition 5.1** (Multimoments of bilinear systems). Let  $\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k)$  be the  $k$ -th regular transfer function of a bilinear system  $\zeta$ . Performing a multi-variate series expansion for every variable  $(s_1,\dots,s_k)$  around the expansion points  $(\sigma_1,\dots,\sigma_k)$  yields [25]

$$\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k) = \sum_{\ell_1=0}^{\infty} \cdots \sum_{\ell_k=0}^{\infty} \mathbf{m}_{(\ell_1,\dots,\ell_k)}^{(j_2,\dots,j_k)}(\sigma_1,\dots,\sigma_k) (s_1 - \sigma_1)^{\ell_1} \cdots (s_k - \sigma_k)^{\ell_k},$$

where the *multimoments*  $\mathbf{m}_{(\ell_1,\dots,\ell_k)}^{(j_2,\dots,j_k)}(\sigma_1,\dots,\sigma_k)$  are given (with  $\ell = \ell_1 + \cdots + \ell_k$ ) by

$$\mathbf{m}_{(\ell_1,\dots,\ell_k)}^{(j_2,\dots,j_k)}(\sigma_1,\dots,\sigma_k) = \frac{1}{\ell!} \frac{\partial^{\ell_1+\dots+\ell_k}}{\partial s_1^{\ell_1} \cdots \partial s_k^{\ell_k}} \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k) \Big|_{s_1=\sigma_1,\dots,s_k=\sigma_k} \quad (5.26)$$

$$= (-1)^\ell \mathbf{C} \left( (\sigma_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^{\ell_k} (\sigma_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \cdots \mathbf{N}_{j_2} \left( (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^{\ell_1} (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B}.$$

Alternatively, we may expand the transfer function at  $\sigma_k \rightarrow \infty$  (which is equivalent to expanding the kernels  $\mathbf{g}_{k,\square}^{(j_2,\dots,j_k)}(t_1,\dots,t_k)$  at  $t_k=0$ ) to gain

$$\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k) = \sum_{\ell_1=0}^{\infty} \cdots \sum_{\ell_k=0}^{\infty} \mathbf{m}_{(\ell_1,\dots,\ell_k),\infty}^{(j_2,\dots,j_k)} \frac{1}{s_1^{\ell_1+1}} \cdots \frac{1}{s_k^{\ell_k+1}},$$

where the Markov parameters (aka. *high-frequency moments*) satisfy

$$\mathbf{m}_{(\ell_1,\dots,\ell_k),\infty}^{(j_2,\dots,j_k)} = \mathbf{C} \left( \mathbf{E}^{-1} \mathbf{A} \right)^{\ell_k} \mathbf{E}^{-1} \mathbf{N}_{j_k} \cdots \mathbf{E}^{-1} \mathbf{N}_{j_2} \left( \mathbf{E}^{-1} \mathbf{A} \right)^{\ell_1} \mathbf{E}^{-1} \mathbf{B}. \quad (5.27)$$

▲

Note that this definition corresponds to the frequency-domain interpretation of moments. The corresponding *time-domain* perception of moment matching for bilinear systems will be discussed in Section 8.4.

In the following, we will concentrate on the methods of subsystem interpolation (cf. Section 5.3) and Volterra series interpolation (cf. Section 5.4). In the literature, these two approaches have been primarily discussed for the SISO and multipoint case. Thus, our focus lies on the MIMO setting as well as on generalizing the ideas to the multimoment case. The main motivation for this endeavor is the goal of obtaining the same flexibility and customizability known from linear moment matching. Moreover, we want to provide more insight into the meaning and importance of the interpolation data to raise the awareness and empowerment of practitioners.

### 5.3 Subsystem interpolation

The idea of subsystem interpolation consists in matching the (multi)moments of the leading  $k$ -th subsystem transfer functions at specific expansion points. This problem was first investigated in [205] for SISO bilinear systems, where the multimoments at infinity  $\sigma_k \rightarrow \infty$  (i.e. the Markov parameters) were considered. Then, the methodology was extended to multimoments at zero  $\sigma_k = 0$  (aka. low-frequency moments) in [47]. Finally, the concept was generalized to any given interpolation point  $\sigma_k \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  in [25]. Based on these publications, the main theorems for *multimoment* subsystem interpolation of SISO bilinear systems are stated in the following.

**Theorem 5.3** (*Multimoment subsystem interpolation (SISO)*). *Let  $\zeta = (\mathbf{A}, \mathbf{N}, \mathbf{b}, \mathbf{c}^\top, \mathbf{E})$  be a SISO bilinear system. Let the expansion points  $\{\sigma_k, \mu_k\}_{k=1}^N \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  be given. Let the matrices  $\mathbf{V}$  and  $\mathbf{W}$  be constructed as follows:*

$$\begin{aligned} \text{ran}(\mathbf{V}^{(1)}) &\subseteq \mathcal{K}_q \left( (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \right), & \mathbf{V}^{(1)} &\in \mathbb{R}^{n \times q}, \\ \text{ran}(\mathbf{V}^{(k)}) &\subseteq \mathcal{K}_q \left( (\sigma_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma_k \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} \mathbf{V}^{(k-1)} \right), & \mathbf{V}^{(k)} &\in \mathbb{R}^{n \times q^k}, \quad k \geq 2 \end{aligned} \quad (5.28)$$

and

$$\begin{aligned} \text{ran}(\mathbf{W}^{(1)}) &\subseteq \mathcal{K}_q \left( (\mu_1 \mathbf{E} - \mathbf{A})^{-\top} \mathbf{E}^\top, (\mu_1 \mathbf{E} - \mathbf{A})^{-\top} \mathbf{c} \right), & \mathbf{W}^{(1)} &\in \mathbb{R}^{n \times q}, \\ \text{ran}(\mathbf{W}^{(k)}) &\subseteq \mathcal{K}_q \left( (\mu_k \mathbf{E} - \mathbf{A})^{-\top} \mathbf{E}^\top, (\mu_k \mathbf{E} - \mathbf{A})^{-\top} \mathbf{N}^\top \mathbf{W}^{(k-1)} \right), & \mathbf{W}^{(k)} &\in \mathbb{R}^{n \times q^k}, \quad k \geq 2 \end{aligned} \quad (5.29)$$

with

$$\text{ran}(\mathbf{V}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{V}^{(k)}\}, \quad \text{ran}(\mathbf{W}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{W}^{(k)}\}, \quad \mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r_{\text{tot}}}, \quad (5.30)$$

where the total number of columns is, in case of no deflation, equal to  $r_{\text{tot}} = \sum_{k=1}^N q^k$ . Then, the reduced bilinear model  $\zeta_r = (\mathbf{W}^\top \mathbf{A} \mathbf{V}, \mathbf{W}^\top \mathbf{N} \mathbf{V}, \mathbf{W}^\top \mathbf{b}, \mathbf{c}^\top \mathbf{V}, \mathbf{W}^\top \mathbf{E} \mathbf{V})$  satisfies the following multimoment subsystem interpolation conditions (SISO):

$$m_{(\ell_1, \dots, \ell_k)}(\sigma_1, \dots, \sigma_k) = m_{\mathbf{r}, (\ell_1, \dots, \ell_k)}(\sigma_1, \dots, \sigma_k), \quad (5.31a)$$

$$m_{(\ell_1, \dots, \ell_k)}(\mu_k, \dots, \mu_1) = m_{\mathbf{r}, (\ell_1, \dots, \ell_k)}(\mu_k, \dots, \mu_1), \quad (5.31b)$$

for  $\ell_1, \dots, \ell_k = 0, \dots, q-1$  and  $k = 1, \dots, N$ . In other words, the ROM interpolates the first  $N$  transfer functions and its high-order derivatives at the selected shifts, respectively. If both (5.28) and (5.29) are employed with  $\sigma_1 = \mu_k, \sigma_2 = \mu_{k-1}, \dots, \sigma_k = \mu_1$ , then the number of matched multimoments doubles:  $m_{(\ell_1, \dots, \ell_k)}(\sigma_1, \dots, \sigma_k) = m_{\mathbf{r}, (\ell_1, \dots, \ell_k)}(\sigma_1, \dots, \sigma_k)$  for  $\ell_1, \dots, \ell_k = 0, \dots, 2q-1$  and  $k = 1, \dots, N$ .

**Remark 5.3** (Markov parameters in subsystem interpolation). In order to match the Markov parameters (5.27) of the  $k$ -th transfer functions, the projection matrices should be chosen as

$$\begin{aligned} \text{ran}(\mathbf{V}^{(1)}) &\subseteq \mathcal{K}_q \left( \mathbf{E}^{-1} \mathbf{A}, \mathbf{E}^{-1} \mathbf{b} \right), & \mathbf{V}^{(1)} &\in \mathbb{R}^{n \times q}, \\ \text{ran}(\mathbf{V}^{(k)}) &\subseteq \mathcal{K}_q \left( \mathbf{E}^{-1} \mathbf{A}, \mathbf{E}^{-1} \mathbf{N} \mathbf{V}^{(k-1)} \right), & \mathbf{V}^{(k)} &\in \mathbb{R}^{n \times q^k}, \quad k \geq 2 \end{aligned} \quad (5.32)$$



and

$$\begin{aligned} \text{ran}(\mathbf{W}^{(1)}) &\subseteq \mathcal{K}_q \left( \mathbf{E}^{-\top} \mathbf{A}^\top, \mathbf{E}^{-\top} \mathbf{c} \right), & \mathbf{W}^{(1)} &\in \mathbb{R}^{n \times q}, \\ \text{ran}(\mathbf{W}^{(k)}) &\subseteq \mathcal{K}_q \left( \mathbf{E}^{-\top} \mathbf{A}^\top, \mathbf{E}^{-\top} \mathbf{N}^\top \mathbf{W}^{(k-1)} \right), & \mathbf{W}^{(k)} &\in \mathbb{R}^{n \times q^k}, \quad k \geq 2, \end{aligned} \quad (5.33)$$

If case of a *one-sided* reduction with  $\mathbf{V} = [\mathbf{V}^{(1)}, \dots, \mathbf{V}^{(k)}]$  or  $\mathbf{W} = [\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(k)}]$ , the ROM  $\zeta_r$  fulfills  $m_{(\ell_1, \dots, \ell_k), \infty} = m_{r, (\ell_1, \dots, \ell_k), \infty}$  for  $\ell_1, \dots, \ell_k = 0, \dots, q-1$ . If both (5.32) and (5.33) are chosen, then  $m_{(\ell_1, \dots, \ell_k), \infty} = m_{r, (\ell_1, \dots, \ell_k), \infty}$  for  $\ell_1, \dots, \ell_k = 0, \dots, 2q-1$ .  $\triangle$

As in the linear case, the projection matrix  $\mathbf{W}$  can be used for stability purposes (see [25] for an approach based on 2), or to increase the number of matched multimoments via a *two-sided* reduction (cf. proof in [25]). We will follow the latter approach to gain a better accuracy.

In addition to the afore explained *multimoment* matching strategy, it is again possible to match (high-order) moments at a set of different shifts  $\{\sigma_{l_k}\}_{l_k=1}^r$  and  $\{\mu_{l_k}\}_{l_k=1}^r$  with associated multiplicities  $\{q_{l_k}\}_{l_k=1}^r$ . In the case of the second subsystem, this leads to:

$$\begin{aligned} \frac{\partial^{\ell_1 + \ell_2}}{\partial s_1^{\ell_1} \partial s_2^{\ell_2}} G_{2, \square}(\sigma_{l_1}, \sigma_{l_2}) &= \frac{\partial^{\ell_1 + \ell_2}}{\partial s_1^{\ell_1} \partial s_2^{\ell_2}} G_{2, r, \square}(\sigma_{l_1}, \sigma_{l_2}) \Leftrightarrow m_{(\ell_1, \ell_2)}(\sigma_{l_1}, \sigma_{l_2}) = m_{r, (\ell_1, \ell_2)}(\sigma_{l_1}, \sigma_{l_2}), \\ \frac{\partial^{\ell_1 + \ell_2}}{\partial s_1^{\ell_1} \partial s_2^{\ell_2}} G_{2, \square}(\mu_{l_2}, \mu_{l_1}) &= \frac{\partial^{\ell_1 + \ell_2}}{\partial s_1^{\ell_1} \partial s_2^{\ell_2}} G_{2, r, \square}(\mu_{l_2}, \mu_{l_1}) \Leftrightarrow m_{(\ell_1, \ell_2)}(\mu_{l_2}, \mu_{l_1}) = m_{r, (\ell_1, \ell_2)}(\mu_{l_2}, \mu_{l_1}), \end{aligned}$$

with  $\ell_k = 0, \dots, q_{l_k} - 1$ ,  $l_k = 1, \dots, r$  and  $k = 1, \dots, N$ . Note that the indices  $\ell_k$  represent the multiplicity of the high-order moments, whereas  $l_k$  stand for the shifts selection. The combination of both concepts (multimoment + multipoint) within the subsystem interpolation framework allows to customize the reduction such that the *desired* interpolation conditions and total reduced order are achieved. This is illustrated in the following example.

*Example 5.2* (Multimoment+multipoint subsystem interpolation (SISO)). Let us consider an example, where we want to match the (multi)moments of the first and second subsystem ( $N=2$ ) at two expansion points  $\sigma_1$  and  $\sigma_2$ .

**1st subsystem** We want to match the following moments of the first transfer function:

$$m_{(0)}(\sigma_1) = m_{r, (0)}(\sigma_1), \quad m_{(1)}(\sigma_1) = m_{r, (1)}(\sigma_1), \quad m_{(0)}(\sigma_2) = m_{r, (0)}(\sigma_2).$$

These conditions are fulfilled, if we construct the projection matrix  $\mathbf{V}^{(1)}$  as follows:

$$\mathbf{V}^{(1)} = \left[ \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, \mathbf{A}_{\sigma_1}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, \mathbf{A}_{\sigma_2}^{-1} \mathbf{b} \right].$$

**2nd subsystem** Now let us assume that we want to match the following multimoments of the second transfer function at certain desired shift combinations:

$$\begin{aligned} m_{(0,0)}(\sigma_1, \sigma_1) &= m_{r, (0,0)}(\sigma_1, \sigma_1), & m_{(1,0)}(\sigma_1, \sigma_1) &= m_{r, (1,0)}(\sigma_1, \sigma_1), & m_{(0,0)}(\sigma_2, \sigma_1) &= m_{r, (0,0)}(\sigma_2, \sigma_1), \\ m_{(0,0)}(\sigma_1, \sigma_2) &= m_{r, (0,0)}(\sigma_1, \sigma_2), & m_{(1,0)}(\sigma_1, \sigma_2) &= m_{r, (1,0)}(\sigma_1, \sigma_2), & m_{(0,0)}(\sigma_2, \sigma_2) &= m_{r, (0,0)}(\sigma_2, \sigma_2), \\ m_{(0,1)}(\sigma_1, \sigma_2) &= m_{r, (0,1)}(\sigma_1, \sigma_2), & m_{(1,1)}(\sigma_1, \sigma_2) &= m_{r, (1,1)}(\sigma_1, \sigma_2), & m_{(0,1)}(\sigma_2, \sigma_2) &= m_{r, (0,1)}(\sigma_2, \sigma_2). \end{aligned}$$

These conditions are fulfilled by constructing  $\mathbf{V}^{(2)}$  as

$$\mathbf{V}^{(2)} = \begin{bmatrix} \mathbf{A}_{\sigma_1}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_1}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_1}^{-1} \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{b}, \\ \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{b} \\ \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, & \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_2}^{-1} \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{b} \end{bmatrix}.$$

Concatenating  $\mathbf{V}^{(1)}$  and  $\mathbf{V}^{(2)}$  finally yields the projection matrix

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}^{(1)} \\ \mathbf{V}^{(2)} \end{bmatrix} \in \mathbb{R}^{n \times 12},$$

which ensures the desired moment matching conditions with  $r_{\text{tot}} = 12$ .  $\triangle$

### 5.3.1 MIMO subsystem interpolation

The concept of subsystem interpolation can be extended to MIMO bilinear systems. The key step towards the generalization consists in understanding that the  $k$ -th transfer matrix  $\mathbf{G}_k^\square(s_1, \dots, s_k) \in \mathbb{R}^{p \times m^k}$  from (4.43) contains all  $m^{k-1}$  combinations  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k) \in \mathbb{R}^{p \times m}$  for the indices  $(j_2, \dots, j_k)$  (cf. Example 4.3). Once this has been assimilated, the idea is to match the (multi)moments of these  $p$ -by- $m$  transfer functions.

Subsystem interpolation for MIMO bilinear systems has been investigated in some publications. In [158], *block* Krylov subspaces are employed to match the low-frequency multimoments ( $\sigma_k = 0$ ) of every  $(j_2, \dots, j_k)$  combination  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$ . This has been generalized to the case of *tangential* interpolation at arbitrary shifts in [23] and [92, Sec. 3.3].

In this thesis, we distinguish between two different strategies for MIMO subsystem interpolation. The first strategy (MIMO-1) corresponds to the publications just mentioned, where moments  $\mathbf{m}_{(\ell_1, \dots, \ell_k)}^{(j_2, \dots, j_k)}(\sigma_1, \dots, \sigma_k)$  for every  $(j_2, \dots, j_k)$  transfer function  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  are matched individually. Motivated by the bilinear Sylvester equations (5.51) and Section 5.4.1 we propose a second strategy (MIMO-2), where moments  $\sum_{j_2=1}^m \dots \sum_{j_k=1}^m \mathbf{m}_{(\ell_1, \dots, \ell_k)}^{(j_2, \dots, j_k)}(\sigma_1, \dots, \sigma_k)$  over summed combinations  $(j_2, \dots, j_k)$  are matched instead. In the following, we will further explain both strategies for the tangential multipoint case with  $q_{l_1} = \dots = q_{l_k} = 1$ . Similar considerations hold also for the block and multimoment cases.

#### MIMO-1 subsystem interpolation

The idea is to match the moments  $\mathbf{m}_{(\ell_1, \dots, \ell_k)}^{(j_2, \dots, j_k)}(\sigma_1, \dots, \sigma_k)$  for every  $(j_2, \dots, j_k)$  transfer function  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  individually. This is achieved as follows. [158, 23, 92]

**Theorem 5.4** (Tangential multipoint subsystem interpolation (MIMO-1)). *Let  $\zeta$  be a MIMO bilinear system with  $\bar{\mathbf{N}} = [\mathbf{N}_1, \dots, \mathbf{N}_m] \in \mathbb{R}^{n \times n \cdot m}$  and  $\bar{\mathbf{N}}^{(2)} = [\mathbf{N}_1^\top, \dots, \mathbf{N}_m^\top] \in \mathbb{R}^{n \times n \cdot m}$ . Let the shifts  $\{\sigma_{l_k}, \mu_{l_k}\}_{l_k=1}^r \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1} \mathbf{A})$  be given. Moreover, let  $\{\mathbf{r}_{l_1}\}_{l_1=1}^r \in \mathbb{C}^m$  and  $\{\mathbf{l}_{l_1}\}_{l_1=1}^r \in \mathbb{C}^p$  be right and left tangential directions, respectively. Let the matrices  $\mathbf{V}$  and  $\mathbf{W}$  be constructed as follows:*

$$\begin{aligned} \text{ran}(\mathbf{V}^{(1)}) &\subseteq \mathcal{K}_1 \left( (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_{l_1} \right), & \mathbf{V}^{(1)} &\in \mathbb{R}^{n \times r}, \\ \text{ran}(\mathbf{V}^{(k)}) &\subseteq \mathcal{K}_1 \left( (\sigma_{l_k} \mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma_{l_k} \mathbf{E} - \mathbf{A})^{-1} \bar{\mathbf{N}} (\mathbf{I}_m \otimes \mathbf{V}^{(k-1)}) \right), & \mathbf{V}^{(k)} &\in \mathbb{R}^{n \times r^k \cdot m^{k-1}}, \end{aligned} \quad (5.35)$$

and

$$\text{ran}(\mathbf{W}^{(1)}) \subseteq \mathcal{K}_1 \left( (\mu_{l_1} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{E}^\top, (\mu_{l_1} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^\top \mathbf{l}_{l_1} \right), \quad \mathbf{W}^{(1)} \in \mathbb{R}^{n \times r}, \quad (5.36)$$

$$\text{ran}(\mathbf{W}^{(k)}) \subseteq \mathcal{K}_1 \left( (\mu_{l_k} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{E}^\top, (\mu_{l_k} \mathbf{E} - \mathbf{A})^{-\top} \bar{\mathbf{N}}^{(2)} (\mathbf{I}_m \otimes \mathbf{W}^{(k-1)}) \right), \quad \mathbf{W}^{(k)} \in \mathbb{R}^{n \times r^k \cdot m^{k-1}},$$

with

$$\text{ran}(\mathbf{V}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{V}^{(k)}\}, \quad \text{ran}(\mathbf{W}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{W}^{(k)}\}, \quad \mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r_{\text{tot}}},$$

where the total number of columns is, in case of no deflation, equal to  $r_{\text{tot}} = \sum_{k=1}^N r^k \cdot m^{k-1}$ . Then, the reduced bilinear model  $\zeta_r = (\mathbf{W}^\top \mathbf{A} \mathbf{V}, \mathbf{W}^\top \mathbf{N}_j \mathbf{V}, \mathbf{W}^\top \mathbf{B}, \mathbf{C} \mathbf{V}, \mathbf{W}^\top \mathbf{E} \mathbf{V})$  satisfies the following tangential multipoint subsystem interpolation conditions (MIMO-1):

$$\mathbf{G}_k^\square(\sigma_{l_1}, \dots, \sigma_{l_k}) (\mathbf{I}_{m^{k-1}} \otimes \mathbf{r}_{l_1}) = \mathbf{G}_{k,r}^\square(\sigma_{l_1}, \dots, \sigma_{l_k}) (\mathbf{I}_{m^{k-1}} \otimes \mathbf{r}_{l_1}), \quad (5.37a)$$

$$\mathbf{l}_{l_1}^\top \mathbf{G}_k^\square(\mu_{l_k}, \dots, \mu_{l_1}) = \mathbf{l}_{l_1}^\top \mathbf{G}_{k,r}^\square(\mu_{l_k}, \dots, \mu_{l_1}), \quad (5.37b)$$

for  $l_1, \dots, l_k = 1, \dots, r$  and  $k = 1, \dots, N$ . In  $(j_2, \dots, j_k)$  notation, the above conditions read:

$$\mathbf{G}_{k,\square}^{(j_2, \dots, j_k)}(\sigma_{l_1}, \dots, \sigma_{l_k}) \mathbf{r}_{l_1} = \mathbf{G}_{k,r,\square}^{(j_2, \dots, j_k)}(\sigma_{l_1}, \dots, \sigma_{l_k}) \mathbf{r}_{l_1}, \quad (5.38a)$$

$$\mathbf{l}_{l_1}^\top \mathbf{G}_{k,\square}^{(j_2, \dots, j_k)}(\mu_{l_k}, \dots, \mu_{l_1}) = \mathbf{l}_{l_1}^\top \mathbf{G}_{k,r,\square}^{(j_2, \dots, j_k)}(\mu_{l_k}, \dots, \mu_{l_1}), \quad (5.38b)$$

for  $j_2, \dots, j_k = 1, \dots, m$ ,  $l_1, \dots, l_k = 1, \dots, r$  and  $k = 1, \dots, N$ .

*Example 5.3* (Projection matrix for MIMO-1 subsystem interpolation). Let us consider the first two leading subsystems ( $N=2$ ), three inputs ( $m=3$ ) and the interpolation data  $\sigma_1, \sigma_2, \mathbf{r}_1, \mathbf{r}_2$  with  $r=2$ . In order to fulfill the following tangential multipoint subsystem interpolation conditions ( $l_1 = l_2 = 1, 2$  and  $k = 1, 2$ )

$$\begin{aligned} \mathbf{G}_1(\sigma_1) \mathbf{r}_1 &= \mathbf{G}_{1,r}(\sigma_1) \mathbf{r}_1, & \mathbf{G}_1(\sigma_2) \mathbf{r}_2 &= \mathbf{G}_{1,r}(\sigma_2) \mathbf{r}_2, \\ \mathbf{G}_{2,\square}^{(j_2)}(\sigma_1, \sigma_1) \mathbf{r}_1 &= \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_1, \sigma_1) \mathbf{r}_1, & \mathbf{G}_{2,\square}^{(j_2)}(\sigma_2, \sigma_1) \mathbf{r}_2 &= \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_2, \sigma_1) \mathbf{r}_2, \\ \mathbf{G}_{2,\square}^{(j_2)}(\sigma_1, \sigma_2) \mathbf{r}_1 &= \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_1, \sigma_2) \mathbf{r}_1, & \mathbf{G}_{2,\square}^{(j_2)}(\sigma_2, \sigma_2) \mathbf{r}_2 &= \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_2, \sigma_2) \mathbf{r}_2, \end{aligned}$$

for  $j_2 \in \{1, 2, 3\}$ , each  $\mathbf{V}^{(k)}$  should be constructed as

$$\begin{aligned} \mathbf{V}^{(1)} &= \left[ (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_1, (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_2 \right], \\ \mathbf{V}^{(2)} &= \left[ \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \right. \\ &\quad \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \\ &\quad \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \\ &\quad \left. \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 \right]. \end{aligned}$$

The reduction matrix is given by  $\mathbf{V} = [\mathbf{V}^{(1)}, \mathbf{V}^{(2)}] \in \mathbb{R}^{n \times 14}$ . We tangentially match  $r=2$  moments of the first and  $r^2 \cdot m = 4 \cdot 3 = 12$  moments of the second transfer function.  $\Delta$

### MIMO-2 subsystem interpolation

Instead of matching the moments for every transfer function  $\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k)$  individually, we propose to sum the moments over all  $(j_2,\dots,j_k)$  combinations of the matrices  $\mathbf{N}_{j_2},\dots,\mathbf{N}_{j_k}$ . This is achieved as follows.

**Theorem 5.5** (Tangential multipoint subsystem interpolation (MIMO-2)). *Let  $\zeta$  be a MIMO bilinear system with  $\bar{\mathbf{N}} = [\mathbf{N}_1, \dots, \mathbf{N}_m] \in \mathbb{R}^{n \times n \cdot m}$  and  $\bar{\mathbf{N}}^{(2)} = [\mathbf{N}_1^\top, \dots, \mathbf{N}_m^\top] \in \mathbb{R}^{n \times n \cdot m}$ . Let the shifts  $\{\sigma_{l_k}, \mu_{l_k}\}_{l_k=1}^r \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  be given, together with right  $\{\mathbf{r}_{l_1}\}_{l_1=1}^r \in \mathbb{C}^m$  and left  $\{\mathbf{l}_{l_1}\}_{l_1=1}^r \in \mathbb{C}^p$  tangential directions, respectively. Let  $\mathbf{1}_m \in \mathbb{R}^m$  be a column vector of  $m$  ones. Let the matrices  $\mathbf{V}$  and  $\mathbf{W}$  be constructed as follows:*

$$\begin{aligned} \text{ran}(\mathbf{V}^{(1)}) &\subseteq \mathcal{K}_1\left((\sigma_{l_1}\mathbf{E} - \mathbf{A})^{-1}\mathbf{E}, (\sigma_{l_1}\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}\mathbf{r}_{l_1}\right), & \mathbf{V}^{(1)} &\in \mathbb{R}^{n \times r}, \\ \text{ran}(\mathbf{V}^{(k)}) &\subseteq \mathcal{K}_1\left((\sigma_{l_k}\mathbf{E} - \mathbf{A})^{-1}\mathbf{E}, (\sigma_{l_k}\mathbf{E} - \mathbf{A})^{-1}\bar{\mathbf{N}}(\mathbf{1}_m \otimes \mathbf{V}^{(k-1)})\right), & \mathbf{V}^{(k)} &\in \mathbb{R}^{n \times r^k}, \end{aligned} \quad (5.39)$$

and

$$\begin{aligned} \text{ran}(\mathbf{W}^{(1)}) &\subseteq \mathcal{K}_1\left((\mu_{l_1}\mathbf{E} - \mathbf{A})^{-\top}\mathbf{E}^\top, (\mu_{l_1}\mathbf{E} - \mathbf{A})^{-\top}\mathbf{C}^\top\mathbf{l}_{l_1}\right), & \mathbf{W}^{(1)} &\in \mathbb{R}^{n \times r}, \\ \text{ran}(\mathbf{W}^{(k)}) &\subseteq \mathcal{K}_1\left((\mu_{l_k}\mathbf{E} - \mathbf{A})^{-\top}\mathbf{E}^\top, (\mu_{l_k}\mathbf{E} - \mathbf{A})^{-\top}\bar{\mathbf{N}}^{(2)}(\mathbf{1}_m \otimes \mathbf{W}^{(k-1)})\right), & \mathbf{W}^{(k)} &\in \mathbb{R}^{n \times r^k}, \end{aligned} \quad (5.40)$$

with

$$\text{ran}(\mathbf{V}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{V}^{(k)}\}, \quad \text{ran}(\mathbf{W}) \subseteq \bigcup_{k=1}^N \text{colspan}\{\mathbf{W}^{(k)}\}, \quad \mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r_{\text{tot}}},$$

where the total number of columns is, in case of no deflation, equal to  $r_{\text{tot}} = \sum_{k=1}^N r^k$ . Then, the reduced bilinear model  $\zeta_r = (\mathbf{W}^\top \mathbf{A} \mathbf{V}, \mathbf{W}^\top \mathbf{N}_j \mathbf{V}, \mathbf{W}^\top \mathbf{B}, \mathbf{C} \mathbf{V}, \mathbf{W}^\top \mathbf{E} \mathbf{V})$  satisfies the following tangential multipoint subsystem interpolation conditions (MIMO-2):

$$\mathbf{G}_k^\square(\sigma_{l_1}, \dots, \sigma_{l_k})(\mathbf{1}_{m^{k-1}} \otimes \mathbf{r}_{l_1}) = \mathbf{G}_{k,r}^\square(\sigma_{l_1}, \dots, \sigma_{l_k})(\mathbf{1}_{m^{k-1}} \otimes \mathbf{r}_{l_1}), \quad (5.41a)$$

$$\mathbf{l}_{l_1}^\top \mathbf{G}_k^\square(\mu_{l_k}, \dots, \mu_{l_1})(\mathbf{1}_{m^{k-1}} \otimes \mathbf{I}_m) = \mathbf{l}_{l_1}^\top \mathbf{G}_{k,r}^\square(\mu_{l_k}, \dots, \mu_{l_1})(\mathbf{1}_{m^{k-1}} \otimes \mathbf{I}_m), \quad (5.41b)$$

for  $l_1, \dots, l_k = 1, \dots, r$  and  $k = 1, \dots, N$ . In  $(j_2, \dots, j_k)$  notation, the above conditions read:

$$\sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(\sigma_{l_1}, \dots, \sigma_{l_k}) \mathbf{r}_{l_1} = \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \mathbf{G}_{k,r,\square}^{(j_2,\dots,j_k)}(\sigma_{l_1}, \dots, \sigma_{l_k}) \mathbf{r}_{l_1}, \quad (5.42a)$$

$$\sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \mathbf{l}_{l_1}^\top \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(\mu_{l_k}, \dots, \mu_{l_1}) = \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \mathbf{l}_{l_1}^\top \mathbf{G}_{k,r,\square}^{(j_2,\dots,j_k)}(\mu_{l_k}, \dots, \mu_{l_1}), \quad (5.42b)$$

for  $l_1, \dots, l_k = 1, \dots, r$  and  $k = 1, \dots, N$ .

The summation over all  $m^{k-1}$  combinations  $\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k)$  significantly reduces the number of Krylov vectors contained in  $\mathbf{V}$  and  $\mathbf{W}$  from  $r_{\text{tot}} = \sum_{k=1}^N r^k \cdot m^{k-1}$  to  $r_{\text{tot}} = \sum_{k=1}^N r^k$ . Nevertheless, it is important to note that the interpolation conditions (5.42) are *weaker* than the ones in (5.38), since the latter consider every combination  $(j_2, \dots, j_k)$  independently. However, as we will see later, the strategy MIMO-2 is being implicitly exploited during the Volterra series interpolation and  $\mathcal{H}_2$ -optimal reduction of MIMO bilinear systems.

*Example 5.4* (Projection matrix for MIMO-2 subsystem interpolation). Let us consider the first two leading subsystems ( $N=2$ ), three inputs ( $m=3$ ) and the interpolation data  $\sigma_1, \sigma_2, \mathbf{r}_1, \mathbf{r}_2$  with  $r=2$ . In this case, the interpolation conditions are given by

$$\begin{aligned} \mathbf{G}_1(\sigma_1)\mathbf{r}_1 &= \mathbf{G}_{1,r}(\sigma_1)\mathbf{r}_1, & \mathbf{G}_1(\sigma_2)\mathbf{r}_2 &= \mathbf{G}_{1,r}(\sigma_2)\mathbf{r}_2, \\ \sum_{j_2=1}^3 \mathbf{G}_{2,\square}^{(j_2)}(\sigma_1, \sigma_1)\mathbf{r}_1 &= \sum_{j_2=1}^3 \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_1, \sigma_1)\mathbf{r}_1, & \sum_{j_2=1}^3 \mathbf{G}_{2,\square}^{(j_2)}(\sigma_2, \sigma_1)\mathbf{r}_2 &= \sum_{j_2=1}^3 \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_2, \sigma_1)\mathbf{r}_2, \\ \sum_{j_2=1}^3 \mathbf{G}_{2,\square}^{(j_2)}(\sigma_1, \sigma_2)\mathbf{r}_1 &= \sum_{j_2=1}^3 \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_1, \sigma_2)\mathbf{r}_1, & \sum_{j_2=1}^3 \mathbf{G}_{2,\square}^{(j_2)}(\sigma_2, \sigma_2)\mathbf{r}_2 &= \sum_{j_2=1}^3 \mathbf{G}_{2,r,\square}^{(j_2)}(\sigma_2, \sigma_2)\mathbf{r}_2. \end{aligned}$$

We can fulfill these conditions by constructing each  $\mathbf{V}^{(k)}$  as follows

$$\begin{aligned} \mathbf{V}^{(1)} &= \left[ (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_1, (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_2 \right], \\ \mathbf{V}^{(2)} &= \left[ \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1 + \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1 + \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \right. \\ &\quad \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 + \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 + \mathbf{A}_{\sigma_1}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2, \\ &\quad \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1 + \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1 + \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \\ &\quad \left. \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_1 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 + \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_2 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 + \mathbf{A}_{\sigma_2}^{-1} \mathbf{N}_3 \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 \right]. \end{aligned}$$

The reduction matrix is given by  $\mathbf{V} = \left[ \mathbf{V}^{(1)}, \mathbf{V}^{(2)} \right] \in \mathbb{R}^{n \times 6}$ . Hereby, we *tangentially* match  $r=2$  moments of the first and  $r^2=4$  moments of the second transfer function.  $\triangle$

Finally, note that the Krylov subspaces (5.35), (5.36) for the MIMO-1 case – and (5.39), (5.40) for the MIMO-2 case – are deliberately given in the Kronecker notation for comparison reasons with [158], [23] and [92, Sec. 3.3]. Through the examples the reader should understand how the subspaces are built and note the differences between both MIMO strategies.

### 5.3.2 Flexibility and combinatorial problem

As already mentioned, the subsystem interpolation framework can be customized by deliberately choosing the number of high-order moments (multimoment) and the shift combinations (multipoint). In the next paragraphs, we note two further flexibility aspects of the subsystem interpolation that we have not considered until now for clarity and simplicity purposes.

**Sets of interpolation data** Instead of using the same sets of expansion points  $S_\sigma = \{\sigma_{l_k}\}_{l_k=1}^r$  and  $S_\mu = \{\mu_{l_k}\}_{l_k=1}^r$  for all subsystems, one could also employ individual sets for each  $k$ -th subsystem, i.e.  $S_\sigma^{(k)} = \{\sigma_{l_k}^{(k)}\}_{l_k=1}^r$  and  $S_\mu^{(k)} = \{\mu_{l_k}^{(k)}\}_{l_k=1}^r$ . In the former approach the shifts need to cover the frequency range of all subsystems since they are recycled for  $k \geq 2$ . In the latter case one could consider different frequency ranges for each Laplace variable  $s_k$  independently.

**Different Krylov subspace dimensions** Note that we have assumed equal dimensions of the Krylov subspaces for all subsystems (e.g.  $q$  in Theorem 5.3 or  $q_{l_1} = \dots = q_{l_k} = 1$  in Theorems 5.4 and 5.5). However, the dimensions do not necessarily have to be the same for all  $k$ . In fact, a conceivable approach is to employ different orders  $q_{l_1}, \dots, q_{l_k}$  for each  $k$ -th Krylov subspace

depending on the desired accuracy. Since the Volterra subsystems become less important with increasing  $k$ , it seems reasonable to match less moments for higher subsystems.

The main drawback of the subsystem interpolation approach is that the dimension of the ROM can grow very rapidly due to the rising number of shifts combinations. For instance, in the SISO multipoint case with  $r$  shifts, we have  $r^k$  combinations for each  $k$ -th subsystem and a total reduced order of  $r_{\text{tot}} = \sum_{k=1}^N r^k$ . The order grows even more rapidly in the MIMO-1 case (cf. Example 5.3) or for block Krylov subspaces. This makes difficult to increase the number  $N$  of considered subsystems. However, there are a few workarounds for this combinatorial problem that are listed below:

1. One may consider only  $p_2$  columns of  $\mathbf{V}^{(1)}$  to calculate  $\mathbf{V}^{(2)}$ . This is proposed in [47], where  $\text{ran}(\mathbf{V}^{(2)}) \subseteq \mathcal{K}_q\left((\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E}, (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} \mathbf{V}_{[p_2]}^{(1)}\right)$ . This reduces the number of columns from  $q + q^2$  to  $q + p_2 \cdot q$ , where  $p_2 < q$ .
2. In order to reduce the number of possible shifts combinations, one could interpolate the transfer functions only at points that lie along the line  $s_1 = \dots = s_k = \sigma_i$ . Doing so, one would match  $\mathbf{G}_{k,\square}^{(j_2, \dots, j_k)}(\sigma_i, \dots, \sigma_i) \mathbf{r}_i = \mathbf{G}_{k,r,\square}^{(j_2, \dots, j_k)}(\sigma_i, \dots, \sigma_i) \mathbf{r}_i$  for  $i = 1, \dots, r$ . In this MIMO-1 case the total reduced order becomes  $r_{\text{tot}} = \sum_{k=1}^N r \cdot m^{k-1}$ .
3. Another possibility is to allow different orders  $r_1, r_2, \dots, r_N$ , instead of reducing each subsystem to the same order  $r$ . One reasonable approach could be to reduce the leading subsystems more accurately, i.e.  $r_1 \geq r_2 \geq r_3 \geq \dots \geq r_N$ . In the SISO multipoint case, this leads to  $r_{\text{tot}} = r_1 + r_1 \cdot r_2 + r_1 \cdot r_2 \cdot r_3 + \dots$ , when taking all shifts combinations into account. Neglecting some combinations one could even obtain  $r_{\text{tot}} = r_1 + r_2 + \dots + r_N$ .

In many publications, it has been observed that subsystem interpolation of the first two transfer functions only can yield ROMs that approximate the bilinear dynamics very well. In case that a higher number of subsystems is necessary for the bilinear system at hand, the above workarounds can help to apply the framework in an efficient manner.

Our implementation of subsystem interpolation is contained in the BSSSMOR functions `bilinearRk` and `bilinearArnoldi`, where the workaround 1 is implemented. The functions support only the SISO case so far, but allow to match both Markov parameters and (multi)moments at arbitrary shifts. For more details, the reader is referred to [Olc16].



BSSSMOR function(s): `bilinearRk`, `bilinearArnoldi`, `bilinearMultiMoments`

## 5.4 Volterra series interpolation

The Volterra series interpolation framework was initially proposed in [92, Sec. 3.4] and [91], where its connection to the  $\mathcal{H}_2$ -optimal reduction algorithm B-IRKA [20] was also elucidated. Meanwhile, the framework has established as a powerful alternative to subsystem interpolation for model reduction of bilinear and also quadratic-bilinear systems [103].

The main idea is to find a ROM that interpolates the *whole* Volterra series, instead of interpolating the leading subsystems only. This idea is motivated by the fact that the solution  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \mathbf{x}_k(t)$  (cf. (4.53)) and the  $\mathcal{H}_2$ -norm (cf. (5.21)) of a bilinear system are given by an infinite series of sums over all possible combinations of point evaluations of the transfer

functions. With these summation features, the combinatorial problem and the increasing reduced order of the subsystem interpolation framework get solved.

In the following, we will first explain the concept of Volterra series interpolation in its original form, i.e. for the SISO multipoint case. Then, we will present our results concerning the MIMO case, the multimoment setting and other important aspects like the selection of the interpolation data in Sections 5.4.1, 5.4.2 and 5.4.3, respectively.

### SISO Volterra series interpolation

As already mentioned, the goal is to match the Volterra series along *weighted* sums of the transfer functions evaluated at all possible shifts combinations. The following theorem summarizes the concept of Volterra series interpolation for the multipoint SISO case. [92, 91].

**Theorem 5.6** (*Multipoint Volterra series interpolation (SISO)*). *Let  $\zeta = (\mathbf{A}, \mathbf{N}, \mathbf{b}, \mathbf{c}^\top, \mathbf{E})$  be a BIBO stable bilinear system. Suppose that interpolation points  $\{\sigma_i\}_{i=1}^r \in \mathbb{C}$  and  $\{\mu_i\}_{i=1}^r \in \mathbb{C}$  along with weighting matrices  $\mathbf{U}_v, \mathbf{U}_w \in \mathbb{C}^{r \times r}$  are given. Let the projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  be constructed as*

$$\mathbf{v}_i = \sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1, \dots, l_{k-1}, i} (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} (\sigma_{l_{k-1}} \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} \cdots \mathbf{N} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b},$$

$$\mathbf{w}_i = \sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1, \dots, l_{k-1}, i} (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \mathbf{N}^\top (\mu_{l_{k-1}} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{N}^\top \cdots \mathbf{N}^\top (\mu_{l_1} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{c},$$

where  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r] \in \mathbb{R}^{n \times r}$  and  $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_r] \in \mathbb{R}^{n \times r}$ . Then, the bilinear reduced-order model  $\zeta_r = (\mathbf{A}_r, \mathbf{N}_r, \mathbf{b}_r, \mathbf{c}_r^\top, \mathbf{E}_r)$  computed with the projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  fulfills the following multipoint Volterra series interpolation conditions (SISO):

$$\sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1, \dots, l_{k-1}, i} G_k^\square(\sigma_{l_1}, \dots, \sigma_{l_{k-1}}, \sigma_i) = \sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1, \dots, l_{k-1}, i} G_{k,r}^\square(\sigma_{l_1}, \dots, \sigma_{l_{k-1}}, \sigma_i)$$

$$\sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1, \dots, l_{k-1}, i} G_k^\square(\mu_i, \mu_{l_{k-1}}, \dots, \mu_{l_1}) = \sum_{k=1}^{\infty} \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1, \dots, l_{k-1}, i} G_{k,r}^\square(\mu_i, \mu_{l_{k-1}}, \dots, \mu_{l_1})$$

for each  $i = 1, \dots, r$ . The weights  $\eta_{l_1, \dots, l_{k-1}, i}$  and  $\vartheta_{l_1, \dots, l_{k-1}, i}$  are associated to each shift combination in the transfer functions and are given in terms of the entries of the matrices  $\mathbf{U}_v = \{u_{a,b}^v\}$  and  $\mathbf{U}_w = \{u_{a,b}^w\}$  as follows:

$$\eta_{l_1, \dots, l_{k-1}, i} = u_{i, l_{k-1}}^v u_{l_{k-1}, l_{k-2}}^v \cdots u_{l_2, l_1}^v = \mathbf{U}_v(i, l_{k-1}) \cdots \mathbf{U}_v(l_2, l_1), \quad k \geq 2, \quad \eta_{l_1} = 1,$$

$$\vartheta_{l_1, \dots, l_{k-1}, i} = u_{i, l_{k-1}}^w u_{l_{k-1}, l_{k-2}}^w \cdots u_{l_2, l_1}^w = \mathbf{U}_w(i, l_{k-1}) \cdots \mathbf{U}_w(l_2, l_1), \quad k \geq 2, \quad \vartheta_{l_1} = 1.$$

Note that we fix  $s_k = \sigma_i$  for the *input* interpolation conditions, whereas we fix  $s_1 = \mu_i$  for the *output* ones. Further note that we sum over weighted shifts combinations of the transfer functions determined by the indices  $l_1, \dots, l_{k-1}$ . Moreover, we sum over infinite subsystems, whereby it is assumed that the Volterra series  $\mathbf{v}_i = \sum_{k=1}^{\infty} \mathbf{v}_i^{(k)}$  and  $\mathbf{w}_i = \sum_{k=1}^{\infty} \mathbf{w}_i^{(k)}$  converge.

From a computational perspective, it is clearly not possible to explicitly compute the above infinite series of shifted systems for each  $\mathbf{v}_i$  and  $\mathbf{w}_i$ . This is only possible, if the sum

over  $k$  is truncated and a large but finite number  $N$  of Volterra kernels is taken into account. However, it has been shown in [92, 91] that the *infinite* projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  are solutions of the following generalized Sylvester equations

$$\mathbf{E}\mathbf{V}\mathbf{S}_v - \mathbf{A}\mathbf{V} - \mathbf{N}\mathbf{V}\mathbf{U}_v^\top = \mathbf{b}\mathbf{1}_r^\top, \quad (5.43a)$$

$$\mathbf{E}^\top\mathbf{W}\mathbf{S}_w^\top - \mathbf{A}^\top\mathbf{W} - \mathbf{N}^\top\mathbf{W}\mathbf{U}_w^\top = \mathbf{c}\mathbf{1}_r^\top, \quad (5.43b)$$

where  $\mathbf{S}_v = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{C}^{r \times r}$ ,  $\mathbf{S}_w = \text{diag}(\mu_1, \dots, \mu_r) \in \mathbb{C}^{r \times r}$ ,  $\mathbf{U}_v, \mathbf{U}_w \in \mathbb{C}^{r \times r}$  and  $\mathbf{1}_r \in \mathbb{R}^r$  is a vector of ones. This means that *infinite* Volterra series interpolation can be achieved, if one of the bilinear Sylvester equations (5.43) is solved.

In order to understand the Volterra series interpolation and become confident with the indices, we provide next an insightful example for the truncated case.

*Example 5.5 (Truncated Volterra series interpolation (SISO)).* We want to illustrate how the input Krylov directions are explicitly formed to achieve Volterra series interpolation. Let us consider the *truncated* series with  $N = 2$  and a reduced order of  $r = 2$  with interpolation points  $\{\sigma_i\}_{i=1}^2$ . The Krylov directions corresponding to the first transfer function are

$$\mathbf{v}_1^{(1)} = (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \quad \mathbf{v}_2^{(1)} = (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b},$$

with  $\mathbf{V}^{(1)} = [\mathbf{v}_1^{(1)}, \mathbf{v}_2^{(1)}] \in \mathbb{R}^{n \times 2}$ . The Krylov vectors for the second transfer function are

$$\begin{aligned} \mathbf{v}_1^{(2)} &= (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \sum_{l_1=1}^r \eta_{l_1,1} \mathbf{N}(\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \\ &= \eta_{1,1} (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} + \eta_{2,1} (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \end{aligned}$$

and

$$\begin{aligned} \mathbf{v}_2^{(2)} &= (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \sum_{l_1=1}^r \eta_{l_1,2} \mathbf{N}(\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \\ &= \eta_{1,2} (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} + \eta_{2,2} (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \end{aligned}$$

with  $\mathbf{V}^{(2)} = [\mathbf{v}_1^{(2)}, \mathbf{v}_2^{(2)}] \in \mathbb{R}^{n \times 2}$ . Finally, the reduction matrix is obtained via summation, i.e.  $\mathbf{V}^{(1:2)} = \mathbf{V}^{(1)} + \mathbf{V}^{(2)} \in \mathbb{R}^{n \times 2}$ . With this matrix, we achieve the following *truncated* input Volterra series interpolation conditions:

$$\begin{aligned} &\mathbf{c}^\top (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} + \eta_{1,i} \mathbf{c}^\top (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} + \eta_{2,i} \mathbf{c}^\top (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}(\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \\ &= \\ &\mathbf{c}_r^\top (\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{b}_r + \eta_{1,i} \mathbf{c}_r^\top (\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{N}_r (\sigma_1 \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{b}_r + \eta_{2,i} \mathbf{c}_r^\top (\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{N}_r (\sigma_2 \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{b}_r \end{aligned}$$

for  $i = 1, 2$ . This effectively means that we match the weighted series of transfer functions

$$G_1(\sigma_i) + u_{i,1}^v G_2^\square(\sigma_1, \sigma_i) + u_{i,2}^v G_2^\square(\sigma_2, \sigma_i) = G_{1,r}(\sigma_i) + u_{i,1}^v G_{2,r}^\square(\sigma_1, \sigma_i) + u_{i,2}^v G_{2,r}^\square(\sigma_2, \sigma_i)$$

for  $i = 1, 2$  and with  $\eta_{l_1,i} = u_{i,l_1}^v$ . △



As it can be observed from the example, the directions for the second transfer function  $\mathbf{v}_i^{(2)}$  depend on the vectors for the first transfer function  $\mathbf{v}_i^{(1)}$ . Similarly, the directions  $\mathbf{v}_i^{(3)}$  depend on the vectors  $\mathbf{v}_i^{(2)}$ , etc. This is highlighted next:

$$\begin{aligned}\mathbf{v}_i^{(2)} &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{l_1=1}^r u_{i,l_1}^v \mathbf{N} \mathbf{v}_{l_1}^{(1)} = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{l_1=1}^r u_{i,l_1}^v \mathbf{N} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \\ \mathbf{v}_i^{(3)} &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{l_2=1}^r u_{i,l_2}^v \mathbf{N} \mathbf{v}_{l_2}^{(2)} \\ &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{l_1=1}^r \sum_{l_2=1}^r u_{i,l_2}^v u_{l_2,l_1}^v \mathbf{N} (\sigma_{l_2} \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}.\end{aligned}\quad (5.44)$$

For general  $k = 2, \dots, N$  the input and output Krylov directions are given by [91]

$$\mathbf{v}_i^{(1)} = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \quad \mathbf{v}_i^{(k)} = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{l_{k-1}=1}^r u_{i,l_{k-1}}^v \mathbf{N} \mathbf{v}_{l_{k-1}}^{(k-1)}, \quad (5.45a)$$

$$\mathbf{w}_i^{(1)} = (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \mathbf{c}, \quad \mathbf{w}_i^{(k)} = (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \sum_{l_{k-1}=1}^r u_{i,l_{k-1}}^w \mathbf{N}^{\top} \mathbf{w}_{l_{k-1}}^{(k-1)}. \quad (5.45b)$$

We will further compare and discuss the differences between *infinite* and *truncated* Volterra series interpolation with their underlying Sylvester equations in Section 5.4.1. Before that, we briefly mention the special case of *implicit* Volterra series interpolation.

### Implicit Volterra series interpolation

The so-called *implicit* Volterra series interpolation proposed in [2] represents a special case of the general Volterra series interpolation framework for *diagonal* weighting matrices  $\mathbf{U}_v, \mathbf{U}_w$ . Assume that the weights  $\eta_{l_1, \dots, l_{k-1}, i}$  satisfy

$$\eta_{l_1, \dots, l_{k-1}, i} = \begin{cases} (\eta_{i,i})^{k-1} & \text{if } i = l_{k-1} = \dots = l_1, \\ 0 & \text{otherwise,} \end{cases} \quad (5.46)$$

which corresponds to a diagonal weighting matrix  $\mathbf{U}_v = \text{diag}(\eta_{1,1}, \dots, \eta_{r,r}) = \text{diag}(u_{1,1}^v, \dots, u_{r,r}^v)$ . Then, the Volterra series interpolation conditions from Theorem 5.6 become

$$\sum_{k=1}^{\infty} (\eta_{i,i})^{k-1} G_k^{\square}(\sigma_i, \dots, \sigma_i) = \sum_{k=1}^{\infty} (\eta_{i,i})^{k-1} G_{k,r}^{\square}(\sigma_i, \dots, \sigma_i) \quad (5.47)$$

for  $i = 1, \dots, r$ . Using the Neumann series and assuming that  $\|\eta_{i,i}(\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}\| < 1$  and  $\|\eta_{i,i}(\sigma_i \mathbf{E}_r - \mathbf{A}_r)^{-1} \mathbf{N}_r\| < 1$ , it has been proved in [92, 2] that the implicit Volterra series interpolation conditions (5.47) are equivalent to interpolating the transfer function  $G(s) = \mathbf{c}^{\top} (s \mathbf{E} - (\mathbf{A} + \eta_{i,i} \mathbf{N}))^{-1} \mathbf{b}$  of the linear system  $\Sigma = (\mathbf{A} + \eta_{i,i} \mathbf{N}, \mathbf{b}, \mathbf{c}^{\top}, \mathbf{E})$ , i.e.

$$\mathbf{c}^{\top} (\sigma_i \mathbf{E} - \mathbf{A} - \eta_{i,i} \mathbf{N})^{-1} \mathbf{b} := G(\sigma_i) = G_r(\sigma_i) := \mathbf{c}_r^{\top} (\sigma_i \mathbf{E}_r - \mathbf{A}_r - \eta_{i,i} \mathbf{N}_r)^{-1} \mathbf{b}_r \quad (5.48)$$

for  $i = 1, \dots, r$ . The time-domain interpretation will be given in Section 8.4.

### 5.4.1 MIMO Volterra series interpolation

In this section, we generalize the Volterra series interpolation conditions from Theorem 5.6 to the MIMO case. Interestingly, the bilinear Sylvester equations (5.43) have been already given for the most general MIMO setting in [91]. However, the authors only stated the interpolation conditions for the SISO case. Thus, we fill this gap in the following by providing the corresponding interpolation conditions for the MIMO case as well. Our result was obtained by analyzing the generalized Sylvester equations (5.51) in detail and by observing how the Krylov vectors  $\mathbf{v}_i$  and  $\mathbf{w}_i$  are formed *explicitly*. In general, the key point is again to be aware that we have  $m^{k-1}$  combinations  $\mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(s_1,\dots,s_k) \in \mathbb{C}^{p \times m}$  for the indices  $(j_2,\dots,j_k)$ .

We first state our result in the next theorem and then make it plausible using the MIMO bilinear Sylvester equations.

**Theorem 5.7** (Tangential multipoint Volterra series interpolation (MIMO)). *Let  $\zeta$  be a BIBO stable bilinear system. Suppose that interpolation points  $\{\sigma_i\}_{i=1}^r \in \mathbb{C}$  and  $\{\mu_i\}_{i=1}^r \in \mathbb{C}$  along with weighting matrices  $\mathbf{U}_{v,j}, \mathbf{U}_{w,j} \in \mathbb{C}^{r \times r}$  for  $j=1,\dots,m$  are given. Moreover, let  $\{\mathbf{r}_i\}_{i=1}^r \in \mathbb{C}^m$  and  $\{\mathbf{l}_i\}_{i=1}^r \in \mathbb{C}^p$  be right and left tangential directions, respectively. Let the projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  be constructed as*

$$\mathbf{v}_i = \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} (\sigma_{l_{k-1}} \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_{k-1}} \cdots \mathbf{N}_{j_2} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_{l_1}$$

$$\mathbf{w}_i = \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \mathbf{N}_{j_k}^{\top} (\mu_{l_{k-1}} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{N}_{j_{k-1}}^{\top} \cdots \mathbf{N}_{j_2}^{\top} (\mu_{l_1} \mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^{\top} \mathbf{l}_i$$

where  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r] \in \mathbb{R}^{n \times r}$  and  $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_r] \in \mathbb{R}^{n \times r}$ . Then, the bilinear reduced-order model  $\zeta_r = (\mathbf{A}_r, \mathbf{N}_{j,r}, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r)$  computed using the projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  fulfills the following tangential multipoint Volterra series interpolation conditions (MIMO):

$$\begin{aligned} & \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(\sigma_{l_1}, \dots, \sigma_{l_{k-1}}, \sigma_i) \mathbf{r}_{l_1} \\ &= \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \eta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} \mathbf{G}_{k,r,\square}^{(j_2,\dots,j_k)}(\sigma_{l_1}, \dots, \sigma_{l_{k-1}}, \sigma_i) \mathbf{r}_{l_1} \end{aligned} \quad (5.49)$$

and

$$\begin{aligned} & \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} \mathbf{l}_i^{\top} \mathbf{G}_{k,\square}^{(j_2,\dots,j_k)}(\mu_i, \mu_{l_{k-1}}, \dots, \mu_{l_1}) \\ &= \sum_{k=1}^{\infty} \sum_{j_2=1}^m \cdots \sum_{j_k=1}^m \sum_{l_1=1}^r \cdots \sum_{l_{k-1}=1}^r \vartheta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} \mathbf{l}_i^{\top} \mathbf{G}_{k,r,\square}^{(j_2,\dots,j_k)}(\mu_i, \mu_{l_{k-1}}, \dots, \mu_{l_1}) \end{aligned} \quad (5.50)$$

for each  $i = 1, \dots, r$ . The weights  $\eta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)}$  and  $\vartheta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)}$  are associated to each shift and  $(j_2, \dots, j_k)$  combination of the transfer functions and are defined in terms of the entries of the matrices  $\mathbf{U}_{v,j} = \{u_{j_a,b}^v\}$  and  $\mathbf{U}_{w,j} = \{u_{j_a,b}^w\}$  as follows:

$$\eta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} = u_{j_k,i,l_{k-1}}^v u_{j_{k-1},l_{k-1},l_{k-2}}^v \cdots u_{j_2,l_2,l_1}^v = \mathbf{U}_{v,j_k}(i, l_{k-1}) \cdots \mathbf{U}_{v,j_2}(l_2, l_1), \quad k \geq 2, \quad \eta_{l_1} = 1,$$

$$\vartheta_{l_1,\dots,l_{k-1},i}^{(j_2,\dots,j_k)} = u_{j_k,i,l_{k-1}}^w u_{j_{k-1},l_{k-1},l_{k-2}}^w \cdots u_{j_2,l_2,l_1}^w = \mathbf{U}_{w,j_k}(i, l_{k-1}) \cdots \mathbf{U}_{w,j_2}(l_2, l_1), \quad k \geq 2, \quad \vartheta_{l_1} = 1.$$

Similar as before, we sum over weighted shifts combinations of the transfer functions determined by the indices  $l_1, \dots, l_{k-1}$ . Now, we also sum over all  $(j_2, \dots, j_k)$  combinations  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  arising due to the multiple matrices  $\mathbf{N}_1, \dots, \mathbf{N}_m$ . Finally, we sum over infinite subsystems, whereby it is assumed again that the Volterra series  $\mathbf{v}_i = \sum_{k=1}^{\infty} \mathbf{v}_i^{(k)}$  and  $\mathbf{w}_i = \sum_{k=1}^{\infty} \mathbf{w}_i^{(k)}$  converge.

For the development and proof of the above theorem we exploited certain matrix equations. In fact, based on the bilinear Sylvester equations given for the MIMO  $\mathcal{H}_2$ -optimal case in [92], we state the bilinear Sylvester equations for MIMO *Volterra series interpolation* as

$$\mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} - \sum_{j=1}^m \mathbf{N}_j \mathbf{V} \mathbf{U}_{v,j}^{\top} = \mathbf{B} \mathbf{R}, \quad (5.51a)$$

$$\mathbf{E}^{\top} \mathbf{W} \mathbf{S}_w^{\top} - \mathbf{A}^{\top} \mathbf{W} - \sum_{j=1}^m \mathbf{N}_j^{\top} \mathbf{W} \mathbf{U}_{w,j}^{\top} = \mathbf{C}^{\top} \mathbf{L}, \quad (5.51b)$$

where  $\mathbf{S}_v = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{C}^{r \times r}$ ,  $\mathbf{S}_w = \text{diag}(\mu_1, \dots, \mu_r) \in \mathbb{C}^{r \times r}$ ,  $\mathbf{U}_{v,j} \in \mathbb{C}^{r \times r}$ ,  $\mathbf{U}_{w,j} \in \mathbb{C}^{r \times r}$  for  $j = 1, \dots, m$  and  $\mathbf{R} = [\mathbf{r}_1, \dots, \mathbf{r}_r] \in \mathbb{C}^{m \times r}$ ,  $\mathbf{L} = [\mathbf{l}_1, \dots, \mathbf{l}_r] \in \mathbb{C}^{p \times r}$ . These Sylvester equations were the starting point for Theorem 5.7 and the development of the explicit expressions for  $\mathbf{v}_i$  and  $\mathbf{w}_i$ . Indeed, it can be shown that the *infinite* projection matrices  $\mathbf{V} = \sum_{k=1}^{\infty} \mathbf{V}^{(k)}$  and  $\mathbf{W} = \sum_{k=1}^{\infty} \mathbf{W}^{(k)}$  are solutions of the bilinear Sylvester equations (5.51). In addition, we can show that every  $\mathbf{V}^{(k)}$  and  $\mathbf{W}^{(k)}$  satisfies the corresponding linear Sylvester equations

$$\mathbf{E} \mathbf{V}^{(1)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(1)} = \mathbf{B} \mathbf{R}, \quad \mathbf{E} \mathbf{V}^{(k)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(k)} = \sum_{j=1}^m \mathbf{N}_j \mathbf{V}^{(k-1)} \mathbf{U}_{v,j}^{\top}, \quad (5.52a)$$

$$\mathbf{E}^{\top} \mathbf{W}^{(1)} \mathbf{S}_w^{\top} - \mathbf{A}^{\top} \mathbf{W}^{(1)} = \mathbf{C}^{\top} \mathbf{L}, \quad \mathbf{E}^{\top} \mathbf{W}^{(k)} \mathbf{S}_w^{\top} - \mathbf{A}^{\top} \mathbf{W}^{(k)} = \sum_{j=1}^m \mathbf{N}_j^{\top} \mathbf{W}^{(k-1)} \mathbf{U}_{w,j}^{\top}. \quad (5.52b)$$

We want to sketch the proof and make the given explicit formulas for  $\mathbf{v}_i$  and  $\mathbf{w}_i$  plausible. The proof is conducted by vectorizing the above linear Sylvester equations.

Vectorizing  $\mathbf{E} \mathbf{V}^{(2)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(2)} = \sum_{j=1}^m \mathbf{N}_j \mathbf{V}^{(1)} \mathbf{U}_{v,j}^{\top}$  for the second subsystem yields

$$\begin{bmatrix} \mathbf{v}_1^{(2)} \\ \vdots \\ \mathbf{v}_r^{(2)} \end{bmatrix} = \begin{bmatrix} \sigma_1 \mathbf{E} - \mathbf{A} & & \\ & \ddots & \\ & & \sigma_r \mathbf{E} - \mathbf{A} \end{bmatrix}^{-1} \sum_{j=1}^m \begin{bmatrix} u_{j_1,1}^v \mathbf{N}_j & \cdots & u_{j_1,r}^v \mathbf{N}_j \\ \vdots & \ddots & \vdots \\ u_{j_r,1}^v \mathbf{N}_j & \cdots & u_{j_r,r}^v \mathbf{N}_j \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^{(1)} \\ \vdots \\ \mathbf{v}_r^{(1)} \end{bmatrix}.$$

Replacing the index  $j \rightarrow j_2$ , considering the matrix-vector-product as sum and finally substituting the preceding Krylov directions  $\mathbf{v}_i^{(1)}$ , the above equation can be rewritten as

$$\begin{aligned} \mathbf{v}_i^{(2)} &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{j_2=1}^m \sum_{l_1=1}^r u_{j_2, l_1}^v \mathbf{N}_{j_2} \mathbf{v}_{l_1}^{(1)} \\ &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{j_2=1}^m \sum_{l_1=1}^r u_{j_2, l_1}^v \mathbf{N}_{j_2} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_{l_1} \end{aligned}$$

for  $i = 1, \dots, r$ . Note that  $u_{j_2, l_1}^v \hat{=} \eta_{l_1, i}^{(j_2)}$ .

Vectorizing  $\mathbf{E} \mathbf{V}^{(3)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(3)} = \sum_{j=1}^m \mathbf{N}_j \mathbf{V}^{(2)} \mathbf{U}_{v,j}^\top$  for the third subsystem yields

$$\begin{bmatrix} \mathbf{v}_1^{(3)} \\ \vdots \\ \mathbf{v}_r^{(3)} \end{bmatrix} = \begin{bmatrix} \sigma_1 \mathbf{E} - \mathbf{A} & & \\ & \ddots & \\ & & \sigma_r \mathbf{E} - \mathbf{A} \end{bmatrix}^{-1} \sum_{j=1}^m \begin{bmatrix} u_{j_1,1}^v \mathbf{N}_j & \cdots & u_{j_1,r}^v \mathbf{N}_j \\ \vdots & \ddots & \vdots \\ u_{j_r,1}^v \mathbf{N}_j & \cdots & u_{j_r,r}^v \mathbf{N}_j \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^{(2)} \\ \vdots \\ \mathbf{v}_r^{(2)} \end{bmatrix}.$$

Replacing the index  $j \rightarrow j_3$ , considering the matrix-vector-product as sum and finally substituting the preceding Krylov directions  $\mathbf{v}_i^{(2)}$ , the above equation can be rewritten as

$$\begin{aligned} \mathbf{v}_i^{(3)} &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{j_3=1}^m \sum_{l_2=1}^r u_{j_3,i,l_2}^v \mathbf{N}_{j_3} \mathbf{v}_{l_2}^{(2)} \\ &= (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{j_2=1}^m \sum_{j_3=1}^m \sum_{l_1=1}^r \sum_{l_2=1}^r u_{j_3,i,l_2}^v u_{j_2,l_2,l_1}^v \mathbf{N}_{j_3} (\sigma_{l_2} \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_2} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_{l_1} \end{aligned}$$

for  $i = 1, \dots, r$ . Note that  $u_{j_3,i,l_2}^v u_{j_2,l_2,l_1}^v \hat{=} \eta_{l_1,l_2,i}^{(j_2,j_3)}$ .

If we continue this process until the  $k$ -th subsystem to obtain  $\mathbf{v}_i^{(k)}$ , then it becomes clear that the explicit formula for  $\mathbf{v}_i = \sum_{k=1}^{\infty} \mathbf{v}_i^{(k)}$  from Theorem 5.7 is correct. The proof for the output directions  $\mathbf{w}_i = \sum_{k=1}^{\infty} \mathbf{w}_i^{(k)}$  can be conducted in an analogous way. To sum up, the Krylov directions can be computed recursively as follows:

$$\mathbf{v}_i^{(1)} = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_i, \quad \mathbf{v}_i^{(k)} = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \sum_{j_k}^m \sum_{l_{k-1}}^r u_{j_k,i,l_{k-1}}^v \mathbf{N}_{j_k} \mathbf{v}_{l_{k-1}}^{(k-1)}, \quad (5.53a)$$

$$\mathbf{w}_i^{(1)} = (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \mathbf{C}^\top \mathbf{l}_i, \quad \mathbf{w}_i^{(k)} = (\mu_i \mathbf{E} - \mathbf{A})^{-\top} \sum_{j_k}^m \sum_{l_{k-1}}^r u_{j_k,i,l_{k-1}}^w \mathbf{N}_{j_k}^\top \mathbf{w}_{l_{k-1}}^{(k-1)}. \quad (5.53b)$$

*Example 5.6.* Let us consider the truncated series ( $N=2$ ), three inputs ( $m=3$ ) and a reduced order of  $r=2$  with  $\{\sigma_i\}_{i=1}^r$ . Then, the truncated input Volterra series interpolation conditions are

$$\begin{aligned} \mathbf{C} \mathbf{A}_{\sigma_i}^{-1} \mathbf{B} \mathbf{r}_i + \sum_{j_2=1}^3 u_{j_2,i,1}^v \mathbf{C} \mathbf{A}_{\sigma_i}^{-1} \mathbf{N}_{j_2} \mathbf{A}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1 + u_{j_2,i,2}^v \mathbf{C} \mathbf{A}_{\sigma_i}^{-1} \mathbf{N}_{j_2} \mathbf{A}_{\sigma_2}^{-1} \mathbf{B} \mathbf{r}_2 \\ = \\ \mathbf{C}_r \mathbf{A}_{r,\sigma_i}^{-1} \mathbf{B}_r \mathbf{r}_i + \sum_{j_2=1}^3 u_{j_2,i,1}^v \mathbf{C}_r \mathbf{A}_{r,\sigma_i}^{-1} \mathbf{N}_{j_2,r} \mathbf{A}_{r,\sigma_1}^{-1} \mathbf{B}_r \mathbf{r}_1 + u_{j_2,i,2}^v \mathbf{C}_r \mathbf{A}_{r,\sigma_i}^{-1} \mathbf{N}_{j_2,r} \mathbf{A}_{r,\sigma_2}^{-1} \mathbf{B}_r \mathbf{r}_2 \end{aligned}$$

Thus, we approximate a weighted sum of tangentially interpolated transfer matrices.  $\triangle$

**Remark 5.4** (Solution of bilinear Sylvester equations vs. convergence of series). Similar to the bilinear Lyapunov equations (cf. Remark 5.1), note that the bilinear Sylvester equations (5.51) might yield solutions although the Volterra series does not converge. In this case, the solutions  $\mathbf{V}$  and  $\mathbf{W}$  do not correspond to the sum over the solutions of the linear Sylvester equations (5.52), i.e.  $\mathbf{V} \neq \sum_{k=1}^{\infty} \mathbf{V}^{(k)}$ . Actually, the convergence of the series basically depends on  $\lambda(\mathbf{E}^{-1} \mathbf{A}) \subset \mathbb{C}_-$  and sufficient small norms  $\|\mathbf{N}_j\|$ . Hence, the existence of unique solutions is linked to the BIBO stability conditions discussed in Theorem 5.1.  $\triangle$

### Solving large-scale bilinear Sylvester equations

Similar to bilinear Lyapunov equations and balanced truncation, we also have two options when it comes to Sylvester equations and Volterra series interpolation. On the one hand, we can solve the bilinear Sylvester equations (5.51) to compute the *infinite* reduction matrices  $\mathbf{V}$ ,  $\mathbf{W}$  and enforce interpolation on the *whole* Volterra series. On the other hand, we can solve the cascaded sequence of linear Sylvester equations (5.52) or LSEs (5.53) to compute the *truncated* reduction matrices  $\mathbf{V}^{(1:N)} = \sum_{k=1}^N \mathbf{V}^{(k)}$ ,  $\mathbf{W}^{(1:N)} = \sum_{k=1}^N \mathbf{W}^{(k)}$  and enforce interpolation on *partial* sums. The way to proceed basically depends on the computational resources and the convergence/decay of the Volterra series for  $\zeta$ .

The bilinear Sylvester equations (5.51) can be solved using similar strategies as for the bilinear Lyapunov equations. The most straightforward method consists in vectorizing the former, yielding the linear systems of equation

$$\left( \mathbf{S}_v^\top \otimes \mathbf{E} - \mathbf{I}_r \otimes \mathbf{A} - \sum_{j=1}^m \mathbf{U}_{v,j} \otimes \mathbf{N}_j \right) \text{vec}(\mathbf{V}) = \text{vec}(\mathbf{B}\mathbf{R}), \quad (5.54a)$$

$$\left( \mathbf{S}_w \otimes \mathbf{E}^\top - \mathbf{I}_r \otimes \mathbf{A}^\top - \sum_{j=1}^m \mathbf{U}_{w,j} \otimes \mathbf{N}_j^\top \right) \text{vec}(\mathbf{W}) = \text{vec}(\mathbf{C}^\top \mathbf{L}). \quad (5.54b)$$

Solving these LSEs via direct methods ( $\backslash$ ) and reshaping  $\text{vec}(\mathbf{V}) \in \mathbb{R}^{nr \times 1}$ ,  $\text{vec}(\mathbf{W}) \in \mathbb{R}^{nr \times 1}$  leads to the projection matrices  $\mathbf{V} \in \mathbb{R}^{n \times r}$ ,  $\mathbf{W} \in \mathbb{R}^{n \times r}$ . This approach is implemented in the BSSMOR function `volterraBrk`. However, it is only applicable to medium-sized models due to the dimension  $nr \times nr$ . As in the Lyapunov case, tensorized low-rank variants of iterative Krylov-based solvers (cf. [150, 21]) are also conceivable to solve the above Sylvester-like linear systems of equations in the large-scale setting.

Another option could be to apply the bilinear extensions of LR-ADI and EKSM/RKSM developed in [21], [45, Sec. 4.4] to the bilinear *Sylvester* equations (5.51).

Finally, under similar convergence assumptions as for (5.15), one could also apply the procedure [257] and employ the fixed-point iteration

$$\begin{aligned} \mathbf{E}\mathbf{V}^{(1)}\mathbf{S}_v - \mathbf{A}\mathbf{V}^{(1)} &= \mathbf{B}\mathbf{R}, \\ \mathbf{E}\mathbf{V}^{(k)}\mathbf{S}_v - \mathbf{A}\mathbf{V}^{(k)} &= \sum_{j=1}^m \mathbf{N}_j \mathbf{V}^{(k-1)} \mathbf{U}_{v,j}^\top + \mathbf{B}\mathbf{R}, \quad k = 2, \dots \end{aligned} \quad (5.55)$$

to compute the stationary solution  $\mathbf{V} = \lim_{k \rightarrow \infty} \mathbf{V}^{(k)}$  of the bilinear Sylvester equation (5.51a). The *linear* Sylvester equations (5.55) can then be solved using the usual methods described in Remark 3.2, i.e. `lyap`,  $\backslash$ , low-rank variants of CG/GMRES or LR-ADI/EKSM/RKSM.

Instead of solving the bilinear Sylvester equations (5.51), one can alternatively solve the cascaded series of linear Sylvester equations (5.52) until the truncation index  $N$ . Again, the latter can be solved using the methods described in Remark 3.2. In this thesis, however, we prefer to consider the LSEs (5.53) in an Arnoldi-like manner rather than solving the linear Sylvester equations (5.52). Although both strategies are theoretically equivalent, we favor the *Arnoldi viewpoint* for numerical implementation in `volterraBarnoldi` (cf. Algorithm 5.1). This decision is motivated by the following reason: we prefer to solve the involved LSEs explicitly and one-by-one using the common Krylov subspace workflow (i.e. LU +  $\backslash$  or

iterSolve) instead of implementing the usual matrix equation solvers for the *Sylvester* case (`mess_lyap` only supports Lyapunov equations so far [248]). Note, however, that especially in the multimoment case it may be numerically more robust to consider the Sylvester equations than the Arnoldi-like implementation [273].

### 5.4.2 Multimoment Volterra series interpolation

In this section, we briefly discuss our extension of the Volterra series interpolation framework to the *multimoment* case. Our motivation to look deeper into this case is the fact that matching high-order moments is computationally cheaper than the multipoint strategy. Recall from the discussion in Section 3.4.3 that we need to perform less LU decompositions in the multimoment case due to the multiplicity of shifts. Another reason is the aim of obtaining a bilinear interpolation framework which is flexible, customizable and supports all different cases known from the linear case. Remember that combining multipoint + multimoment represents the best trade-off between approximation quality and computational cost. Thus, we extended the Volterra series interpolation to match both 0-th and high-order moments. Since the extension requires more complicated indices and even longer expressions, we present here only the most important highlights and an insightful example. The reader is referred to the bachelor thesis [Röt18, Sec. 4.3.2] for further details.

The multimoments  $\mathbf{m}_{(\ell_1, \dots, \ell_k)}^{(j_2, \dots, j_k)}(\sigma_1, \dots, \sigma_k)$  of the transfer functions  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  have been introduced in Eq. (5.26). Without giving the formulas explicitly, the multimoment Volterra series interpolation conditions are basically obtained by replacing the transfer functions  $\mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(s_1, \dots, s_k)$  in (5.49) and (5.50) by the multimoments  $\mathbf{m}_{(\ell_1, \dots, \ell_k)}^{(j_2, \dots, j_k)}(\sigma_1, \dots, \sigma_k)$ . Due to the indices  $\ell_1, \dots, \ell_k$  representing the multiplicity of the moments, we have to use more complex notation for the weights and also add sums of the form  $\sum_{\ell_1=0}^{q_{\tilde{\sigma}_1}-1} \dots \sum_{\ell_k=0}^{q_{\tilde{\sigma}_k}-1}$ . Hereby, a tilde shift  $\tilde{\sigma}_i$  denotes the  $i$ -th *unique* shift,  $q_{\tilde{\sigma}_i}$  represents the multiplicity of a certain shift  $\tilde{\sigma}_i$  and  $\tilde{r}$  denotes the number of unique shifts. This notation will become clear in the upcoming example.

The Krylov vectors  $\mathbf{v}_i = \sum_{k=1}^{\infty} \mathbf{v}_i^{(k)}$  and  $\mathbf{w}_i = \sum_{k=1}^{\infty} \mathbf{w}_i^{(k)}$  required to enforce multimoment Volterra series interpolation can again be derived by looking at the explicit structure of the formulas for  $\mathbf{v}_i^{(1)}$ ,  $\mathbf{v}_i^{(2)}$ , etc. Formulating the columns  $\mathbf{v}_i^{(k)}$  using the previous ones  $\mathbf{v}_i^{(k-1)}$  yields the recurrence [Röt18]

$$\begin{aligned} \mathbf{v}_i^{(1)} &= \left( (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^{\ell_1} (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \mathbf{r}_{\tilde{i}}, \\ \mathbf{v}_i^{(k)} &= \left( (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^{\ell_k} \sum_{j_k}^m \sum_{l_{k-1}}^r u_{j_k \tilde{i}, l_{k-1}}^v (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N}_{j_k} \mathbf{v}_{l_{k-1}}^{(k-1)}. \end{aligned} \quad (5.56)$$

Dual expressions hold for the output case as well. The definition for  $\tilde{i}$  is given in [Röt18].

Similar to the linear case, we want to discuss how the shift matrices  $\mathbf{S}_v, \mathbf{S}_w \in \mathbb{C}^{r \times r}$ , the tangential directions matrices  $\mathbf{R} \in \mathbb{C}^{m \times r}$ ,  $\mathbf{L} \in \mathbb{C}^{p \times r}$  and especially the weighting matrices  $\mathbf{U}_{v,j}, \mathbf{U}_{w,j} \in \mathbb{R}^{r \times r}$  have to look like in order to interpret the *multimoment* projection matrices  $\mathbf{V}$  and  $\mathbf{W}$  equivalently as solutions of the Sylvester equations (5.51). This is possible with the following Sylvester matrices:

- $\mathbf{S}_v$  and  $\mathbf{S}_w^\top$  have *Jordan blocks* for equal shifts,

- $\mathbf{R}$  and  $\mathbf{L}$  have *zero-columns* for directions corresponding to high-order moments,
- $\mathbf{U}_{v,j}$  and  $\mathbf{U}_{w,j}$  have *zero-rows* for directions corresponding to high-order moments.

In order to make the above statement plausible, we provide an example for the SISO case.

*Example 5.7* (Multimoment+multipoint Volterra series interpolation). Let us assume that we want to reduce a bilinear system  $\zeta$  with  $r = 3$  shifts  $\sigma_1 = \sigma_2 \neq \sigma_3$ . Since  $\sigma_1 = \sigma_2$ , we have  $\tilde{r} = 2$  unique shifts  $\tilde{\sigma}_1 \hat{=} \sigma_1$  and  $\tilde{\sigma}_2 \hat{=} \sigma_3$ . Hence, the multiplicity of the first shift is  $q_{\tilde{\sigma}_1} = 2$ , whereas  $q_{\tilde{\sigma}_2} = 1$ . Consequently, we have to match the 0-th and 1-st moment for  $\tilde{\sigma}_1$ , and the 0-th moment for  $\tilde{\sigma}_2$ . The Sylvester matrices are accordingly given by

$$\mathbf{S}_v = \begin{bmatrix} \sigma_1 & -1 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}, \quad \mathbf{U}_v = \begin{bmatrix} u_{1,1}^v & u_{1,2}^v & u_{1,3}^v \\ 0 & 0 & 0 \\ u_{3,1}^v & u_{3,2}^v & u_{3,3}^v \end{bmatrix}, \quad [r_1 \ r_2 \ r_3] = [1 \ 0 \ 1].$$

The projection matrix  $\mathbf{V}^{(1)}$  can be obtained by solving

$$\mathbf{E} \begin{bmatrix} \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & \mathbf{v}_3^{(1)} \end{bmatrix} \begin{bmatrix} \sigma_1 & -1 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} - \mathbf{A} \begin{bmatrix} \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & \mathbf{v}_3^{(1)} \end{bmatrix} = \mathbf{b} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}.$$

Considering the equation *column-wise* leads to the Krylov directions

$$\mathbf{v}_1^{(1)} = (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \quad \mathbf{v}_2^{(1)} = (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \mathbf{v}_1^{(1)} = (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b}, \quad \mathbf{v}_3^{(1)} = (\sigma_3 \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}.$$

The projection matrix  $\mathbf{V}^{(2)}$  can be obtained by solving  $\mathbf{E} \mathbf{V}^{(2)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(2)} = \mathbf{N} \mathbf{V}^{(1)} \mathbf{U}_v^T$ :

$$\mathbf{E} \begin{bmatrix} \mathbf{v}_1^{(2)} & \mathbf{v}_2^{(2)} & \mathbf{v}_3^{(2)} \end{bmatrix} \begin{bmatrix} \sigma_1 & -1 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} - \mathbf{A} \begin{bmatrix} \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & \mathbf{v}_3^{(1)} \end{bmatrix} = \mathbf{N} \mathbf{V}^{(1)} \begin{bmatrix} u_{1,1}^v & 0 & u_{3,1}^v \\ u_{1,2}^v & 0 & u_{3,2}^v \\ u_{1,3}^v & 0 & u_{3,3}^v \end{bmatrix}.$$

Considering the equation *column-wise* together with the previous Krylov vectors yields

$$\begin{aligned} \mathbf{v}_1^{(2)} &= (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \left( u_{1,1}^v \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{1,2}^v \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{1,3}^v \mathbf{N} \mathbf{A}_{\sigma_3}^{-1} \mathbf{b} \right), \\ \mathbf{v}_2^{(2)} &= (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \mathbf{v}_1^{(2)} \\ &= (\sigma_2 \mathbf{E} - \mathbf{A})^{-1} \mathbf{E} (\sigma_1 \mathbf{E} - \mathbf{A})^{-1} \left( u_{1,1}^v \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{1,2}^v \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{1,3}^v \mathbf{N} \mathbf{A}_{\sigma_3}^{-1} \mathbf{b} \right), \\ \mathbf{v}_3^{(2)} &= (\sigma_3 \mathbf{E} - \mathbf{A})^{-1} \left( u_{3,1}^v \mathbf{N} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{3,2}^v \mathbf{N} \mathbf{A}_{\sigma_2}^{-1} \mathbf{E} \mathbf{A}_{\sigma_1}^{-1} \mathbf{b} + u_{3,3}^v \mathbf{N} \mathbf{A}_{\sigma_3}^{-1} \mathbf{b} \right). \end{aligned}$$

The truncated reduction matrix is obtained via summation, i.e.  $\mathbf{V}^{(1:2)} = \mathbf{V}^{(1)} + \mathbf{V}^{(2)}$ . Hereby, we achieve the following multimoment + multipoint matching conditions:

$$\begin{aligned} m_{(0)}(\sigma_1) + u_{1,1}^v m_{(0,0)}(\sigma_1, \sigma_1) + u_{1,2}^v m_{(1,0)}(\sigma_1, \sigma_1) + u_{1,3}^v m_{(0,0)}(\sigma_3, \sigma_1) &= m_{r,(0)}(\sigma_1) + \dots \\ m_{(1)}(\sigma_1) + u_{1,1}^v m_{(0,1)}(\sigma_1, \sigma_1) + u_{1,2}^v m_{(1,1)}(\sigma_1, \sigma_1) + u_{1,3}^v m_{(0,1)}(\sigma_3, \sigma_1) &= m_{r,(1)}(\sigma_1) + \dots \\ m_{(0)}(\sigma_3) + u_{3,1}^v m_{(0,0)}(\sigma_1, \sigma_3) + u_{3,2}^v m_{(1,0)}(\sigma_1, \sigma_3) + u_{3,3}^v m_{(0,0)}(\sigma_3, \sigma_3) &= m_{r,(0)}(\sigma_3) + \dots \end{aligned}$$

We refer the reader to [Röt18, Sec. 4.3.2] for the general expressions of the multimoment Volterra series interpolation conditions.  $\triangle$

Despite the difficult notation, note that it is rather easy to embed the multimoment case in Algorithm 5.1. First, the LU factors for *unique* shifts  $\sigma_i$  are computed *once* at the beginning of the algorithm. Then, we distinguish between matching the 0-th moment at a new shift, or matching a high-order moment using the previous shift and direction. Note that the subscript  $\sigma_i$  means that we pick the LU factors corresponding to the current shift  $\sigma_i$  (cf. e.g. line 12). The algorithm can also handle other cases that are discussed in the next section.

### 5.4.3 Other cases and complete Arnoldi algorithm

In this section, we focus on other important aspects that are considered in our Arnoldi-like Volterra series interpolation algorithm.

**Output case and Two-sided reduction** Although we have presented the theory based rather on input Krylov subspaces, note that the *output* case is also implemented in `volterraBarnoldi`. The reduction matrix  $\mathbf{W}^{(1:N)}$  is computed in a dual manner by setting  $\mathbf{A} \rightarrow \mathbf{A}^\top$ ,  $\mathbf{E} \rightarrow \mathbf{E}^\top$ ,  $\mathbf{N}_j \rightarrow \mathbf{N}_j^\top$ ,  $\mathbf{B} \rightarrow \mathbf{C}^\top$  in Algorithm 5.1 using user-defined interpolation data  $(\mathbf{S}_w, \mathbf{L}, \mathbf{U}_{w,j})$ . In `volterraBrk` practitioners can decide to perform a one-sided reduction via input *or* output Krylov subspace, or apply a two-sided reduction with both projection matrices. The latter approach increases the number of matched moments, and hence the approximation quality. However, it is also conceivable to exploit the degrees of freedom in  $\mathbf{W}$  for stability purposes within the Volterra series interpolation framework (using e.g. an approach similar to 2).

**Block case** For MIMO bilinear systems our algorithm `volterraBarnoldi` can support both the tangential and block Krylov case. The latter is basically achieved by replacing the tangential directions by identity matrices, i.e.  $\mathbf{r}_i \rightarrow \mathbf{I}_m$  and  $\mathbf{l}_i \rightarrow \mathbf{I}_p$ . Note that the user-given Sylvester matrices  $(\mathbf{S}_v, \mathbf{R}, \mathbf{U}_{v,j})$  and  $(\mathbf{S}_w, \mathbf{L}, \mathbf{U}_{w,j})$  are enlarged internally by  $m$ - and  $p$ -times per shift, respectively (cf. Eqs. (3.63) and (3.66)). The reader is referred to [Röt18, Sec. 4.3.3] for more details on the block case.

**Markov parameters** To the best of the author's knowledge, Markov parameters have not been considered within the Volterra series interpolation framework so far. However, once the multimoment case has been developed, it is rather easy to extend it to the case of Markov parameters:

$$\mathbf{v}_i^{(1)} = \left(\mathbf{E}^{-1}\mathbf{A}\right)^{\ell_1} \mathbf{E}^{-1}\mathbf{B} \mathbf{r}_i, \quad \mathbf{v}_i^{(k)} = \left(\mathbf{E}^{-1}\mathbf{A}\right)^{\ell_k} \sum_{j_k}^m \sum_{l_{k-1}}^r u_{j_k i, l_{k-1}}^v \mathbf{E}^{-1}\mathbf{N}_{j_k} \mathbf{v}_{l_{k-1}}^{(k-1)}. \quad (5.57)$$

This is implemented in Algorithm 5.1, see e.g. lines 10, 25. Similar expressions hold for the output case as well.

**Hermite case** From the linear setting we know that applying a two-sided reduction with equal shifts  $\sigma_i = \mu_i$  for  $\mathbf{V}$  and  $\mathbf{W}$  is referred to as *Hermite* interpolation. In this case, we match not only the 0-th but also the first moment at the selected shifts (cf. Eq. (3.55)). Furthermore, if  $\sigma_i = \mu_i$  then it is not necessary to perform the LU decompositions of  $(\sigma_i \mathbf{E} - \mathbf{A})$  and  $(\sigma_i \mathbf{E}^\top - \mathbf{A}^\top)$ . Instead, one performs the LU decompositions of  $(\sigma_i \mathbf{E} - \mathbf{A})$  only and uses the *transposed* LU factors to compute the output Krylov directions.



In the bilinear setting both the shifts *and the weights* have to be equal for Hermite Volterra series interpolation, i.e.  $\mathbf{S}_v = \mathbf{S}_w$  and  $\mathbf{U}_{v,j} = \mathbf{U}_{w,j}$ . Then, right (5.49), left (5.50) and *bitangential Hermite* interpolation can be achieved similarly to the  $\mathcal{H}_2$ -optimal case [91]. One again might want to increase the efficiency of the algorithm by reusing and transposing the LU factors for the computation of the output projection matrix. However, practitioners are reminded that we experienced some numerical round-off errors when approaching this strategy. Since the linear systems of equations are computed recursively using the preceding columns, the numerical error might grow significantly.

**Complex shifts** Depending on the bilinear dynamics it might be appropriate to employ complex shifts for the reduction. As in the linear case, we want again to obtain real-valued projection matrices  $\tilde{\mathbf{V}}, \tilde{\mathbf{W}} \in \mathbb{R}^{n \times r}$  to avoid getting a complex-valued reduced model  $\zeta_r$ . The reduction matrices  $\mathbf{V}, \mathbf{W} \in \mathbb{C}^{n \times r}$  are made real by splitting the basis vectors in real and imaginary part, or in other words, via transformation  $\tilde{\mathbf{V}} = \mathbf{V}\mathbf{T}_v$ ,  $\tilde{\mathbf{W}} = \mathbf{W}\mathbf{T}_w$  with regular transformation matrices (cf. Eq. (3.59)). Due to the invariance property, the complex interpolation data also transforms to real-valued matrices via

$$\tilde{\mathbf{S}}_v = \mathbf{T}_v^{-1} \mathbf{S}_v \mathbf{T}_v, \quad \tilde{\mathbf{U}}_{v,j} = \mathbf{T}_v^\top \mathbf{U}_{v,j} \mathbf{T}_v^{-\top}, \quad \tilde{\mathbf{R}} = \mathbf{R} \mathbf{T}_v, \quad (5.58a)$$

$$\tilde{\mathbf{S}}_w = \mathbf{T}_w^\top \mathbf{S}_w \mathbf{T}_w^{-\top}, \quad \tilde{\mathbf{U}}_{w,j} = \mathbf{T}_w^\top \mathbf{U}_{w,j} \mathbf{T}_w^{-\top}, \quad \tilde{\mathbf{L}} = \mathbf{L} \mathbf{T}_w. \quad (5.58b)$$

However, the mentioned transformation is only applicable if the computed Krylov directions are complex conjugated to each other, i.e.  $\mathbf{v}_1 = \overline{\mathbf{v}_2}$ . As shown in Example 3.1, this requires that (i) the shifts and (ii) the tangential directions come in complex conjugate pairs, i.e.  $\sigma_1 = \overline{\sigma_2}$  and  $\mathbf{r}_1 = \overline{\mathbf{r}_2}$ . In the bilinear setting, we further need conditions on (iii) the *weights* to be able to obtain real-valued projection matrices spanning the same subspaces as their complex counterparts. Through analytical and numerical analysis we observed that a *real-valued* matrix  $\tilde{\mathbf{U}}_v$  can be obtained via transformation if the weighting matrix  $\mathbf{U}_v \in \mathbb{C}^{2 \times 2}$  fulfills one of the following constraints:

- all entries are equal and real:  $u_{1,1}^v = u_{1,2}^v = u_{2,1}^v = u_{2,2}^v \in \mathbb{R}$
- opposing entries are equal and real:  $u_{1,1}^v = u_{2,2}^v$  and  $u_{1,2}^v = u_{2,1}^v \in \mathbb{R}$
- opposing entries are complex conjugated:  $u_{1,1}^v = \overline{u_{2,2}^v}$  and  $u_{1,2}^v = \overline{u_{2,1}^v}$

The conditions on  $\mathbf{U}_v$  become much more complicated if both real and complex conjugated shifts are combined. Anyhow: if the interpolation data is chosen such that  $\mathbf{v}_1 = \overline{\mathbf{v}_2}$  can be expected, then we can make the projection and Sylvester matrices real. In such case we only have to solve *one* complex-valued LSE corresponding to  $\sigma_1$ , whereas the other column  $\mathbf{v}_2$  is simply obtained by conjugating  $\mathbf{v}_1$ , i.e.  $\mathbf{v}_2 = \text{conj}(\mathbf{v}_1)$ . Further note that the Sylvester matrices are made real after all Krylov vectors have been computed. This is crucial to keep the diagonal structure of  $\mathbf{S}_v$  during the Arnoldi algorithm to be able to compute the columns individually.

**Orthogonalization** Due to the just mentioned reason with the diagonal structure, the orthogonalization of the projection matrices is also accomplished a-posteriori and not in a modified Gram-Schmidt procedure. The lack of orthogonalization can make the Arnoldi iteration numerically unstable, especially in the multimoment and block case. Nevertheless, we observed that the choice of interpolation data — and more specifically of the weights — rather influences the numerical stability and robustness of the algorithm.

**Selection of interpolation data** The choice of appropriate shifts, tangential directions and weights is crucial to obtain a good approximation. In general, the author of this thesis pleads for a goal-oriented and physically motivated selection of interpolation data using some system knowledge or intuition. For instance, one could spread the shifts over the interesting frequency range  $[\omega_{\min}, \omega_{\max}]$ , and select the tangential directions to favor certain input-output paths. Moreover, one could exploit some knowledge about the amplitude range of relevant input signals  $\mathbf{u}(t)$  to select the weighting matrices. If no system understanding is available a-priori, then one could use e.g.  $\mathbf{s0}=\mathbf{zeros}(1, \mathbf{r})$ ,  $\mathbf{r}_i = \mathbf{1}_m$  and  $\mathbf{U}_v = \alpha \cdot \mathbf{I}_r$  with  $\alpha \approx 10^{-3}, \dots, 10^2$ . A more reasonable strategy is to calculate some eigenvalues of the system using  $[\mathbf{X}, \mathbf{\Lambda}] = \mathbf{eigs}(\mathbf{A}, \mathbf{E})$  and then select the interpolation data as follows:  $\mathbf{S}_v \leftarrow -\overline{\mathbf{\Lambda}}$ ,  $\mathbf{R} \leftarrow (\mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{B})^\top$ ,  $\mathbf{U}_{v,j} \leftarrow \mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{N}_j \mathbf{X}$  and  $\mathbf{S}_w \leftarrow -\overline{\mathbf{\Lambda}}$ ,  $\mathbf{L} \leftarrow \mathbf{C} \mathbf{X}$ ,  $\mathbf{U}_{w,j} \leftarrow \mathbf{X}^{-1} \mathbf{E}^{-1} \mathbf{N}_j \mathbf{X}$  (cf. Eq. (5.3)). This choice is motivated by the  $\mathcal{H}_2$ -optimal procedure BIRKA, which will be discussed in the next section. According to our experience with the typical bilinear benchmarks used in the community (see [45, 92]), the weighting matrices usually contain rather *small* numbers. With the  $\mathbf{eigs}$ -strategy we analyzed how the weighting matrices look like for complex conjugated and large/small eigenvalues ('lm', 'sm'). We observed the tendency that fast decaying/left most eigenvalues (leading to large positive shifts  $\sigma_i \in \mathbb{C}_+$ ) yield very small weights, whereas eigenvalues closer to the origin (leading to smaller positive shifts  $\sigma_i \in \mathbb{C}_+$ ) yield bigger weights.

**Numerical aspects** It is very important to perform the LU decompositions only once at the beginning of the algorithm, instead of embedding the computation within the for-loops in lines 7 or 21. This way we have to store all LU factors (requiring thus more memory), but we do not have to calculate them several times unnecessarily. Crucial for performance is also the evaluation of the sums in 31 before solving the systems of equations in lines 33 and 35. This way we solve the systems of equations  $i$ -times instead of  $ijl$ -times.

The projection matrices  $\mathbf{V}^{(k)}$ ,  $\mathbf{W}^{(k)}$  for the  $k$ -th subsystem are computed using the matrices from the previous subsystems. This recurrent dependency can make the algorithm numerically unstable due to the propagation and amplification of round-off errors. This potentially leads to very big projection matrices, especially if the weights are not chosen appropriately or the Volterra series diverges.

The truncated reduction matrices  $\mathbf{V}^{(1:N)}$ ,  $\mathbf{W}^{(1:N)}$  are obtained by summing over  $N$  subsystems (cf. line 37). Hereby, the user needs to specify the truncation index  $N$  at the beginning of the algorithm. However, an appropriate  $N$  can also be determined on the fly. Similar to the Lyapunov case, one could substitute the current matrix  $\mathbf{V}^{(1:N)} = \sum_{k=1}^N \mathbf{V}^{(k)}$  in the *bilinear* Sylvester equation to compute the norm of the residual

$$\mathbf{Res}^{(1:N)} = \mathbf{E} \mathbf{V}^{(1:N)} \mathbf{S}_v - \mathbf{A} \mathbf{V}^{(1:N)} - \sum_{j=1}^m \mathbf{N}_j \mathbf{V}^{(1:N)} \mathbf{U}_{v,j}^\top - \mathbf{B} \mathbf{R}.$$

Other heuristic — but less expensive — measures are  $\|\mathbf{V}^{(N)} - \mathbf{V}^{(N-1)}\|$  or  $\|\mathbf{V}^{(1:N)} - \mathbf{V}^{(1:N-1)}\|$ .



BSSMOR function(s): `volterraBrk`, `volterraBarnoldi`

**Algorithm 5.1** Arnoldi-like algorithm for truncated Volterra series interpolation

**Input:** Bilinear system  $\zeta = (\mathbf{A}, \mathbf{N}_j, \mathbf{B}, \mathbf{C}, \mathbf{E})$ ,  $r$  sorted shifts  $\{\sigma_i\}_{i=1}^r$ ,  $\tilde{r}$  unique shifts  $\{\tilde{\sigma}_i\}_{i=1}^{\tilde{r}}$ , weights  $U_v$ , tangential directions  $\mathbf{R}$ , truncation index  $N$

**Output:** (orthonormal) truncated projection matrix  $\mathbf{V}^{(1:N)}$

```

1: for  $i = 1 : \tilde{r}$  do
2:   if any( $\tilde{\sigma}_i == \infty$ ) then
3:      $[\mathbf{L}_\infty, \mathbf{U}_\infty] = \text{lu}(\mathbf{E})$            ▶ Compute LU factors for Markov parameters
4:   else
5:      $[\mathbf{L}_{\tilde{\sigma}_i}, \mathbf{U}_{\tilde{\sigma}_i}] = \text{lu}(\tilde{\sigma}_i \mathbf{E} - \mathbf{A})$    ▶ Compute LU factors for unique shifts
6:    $\mathbf{V}_{\text{old}} = \mathbf{0}$ 
7:   for  $i = 1 : r$  do           ▶ Compute  $\mathbf{V}^{(1)}$ 
8:     if  $i > 1$  &&  $\sigma_i == \sigma_{i-1}$  then
9:       if  $\sigma_i == \infty$  then
10:         $\mathbf{V}_{\text{old}}(:, i) = \mathbf{U}_\infty \setminus (\mathbf{L}_\infty \setminus (\mathbf{A} \mathbf{V}_{\text{old}}(:, i-1)))$    ▶ High-order Markov
11:      else
12:         $\mathbf{V}_{\text{old}}(:, i) = \mathbf{U}_{\sigma_i} \setminus (\mathbf{L}_{\sigma_i} \setminus (\mathbf{E} \mathbf{V}_{\text{old}}(:, i-1)))$    ▶ High-order moment
13:      else
14:        if  $\sigma_i == \infty$  then
15:           $\mathbf{V}_{\text{old}}(:, i) = \mathbf{U}_\infty \setminus (\mathbf{L}_\infty \setminus (\mathbf{B} \mathbf{R}(:, i)))$            ▶ Markov parameter
16:        else
17:           $\mathbf{V}_{\text{old}}(:, i) = \mathbf{U}_{\sigma_i} \setminus (\mathbf{L}_{\sigma_i} \setminus (\mathbf{B} \mathbf{R}(:, i)))$            ▶ Moment
18:         $\mathbf{V}_{\text{end}} = \mathbf{V}_{\text{old}}$    ▶  $\mathbf{V}^{(1:N)} \leftarrow \mathbf{V}^{(1)}$ 
19:        for  $k = 2 : N$  do   ▶ Loop over  $N$  subsystems
20:           $\mathbf{V}_{\text{new}} = \mathbf{0}$ 
21:          for  $i = 1 : r$  do   ▶ Compute  $\mathbf{V}^{(\geq 2)}$ 
22:             $\mathbf{v}_{\text{temp}} = \mathbf{0}$ 
23:            if  $i > 1$  &&  $\sigma_i == \sigma_{i-1}$  then
24:              if  $\sigma_i == \infty$  then
25:                 $\mathbf{V}_{\text{new}}(:, i) = \mathbf{U}_\infty \setminus (\mathbf{L}_\infty \setminus (\mathbf{A} \mathbf{V}_{\text{new}}(:, i-1)))$    ▶ High-order Markov
26:              else
27:                 $\mathbf{V}_{\text{new}}(:, i) = \mathbf{U}_{\sigma_i} \setminus (\mathbf{L}_{\sigma_i} \setminus (\mathbf{E} \mathbf{V}_{\text{new}}(:, i-1)))$    ▶ High-order moment
28:            else
29:              for  $l = 1 : r$  do
30:                for  $j = 1 : m$  do
31:                   $\mathbf{v}_{\text{temp}} = \mathbf{v}_{\text{temp}} + \mathbf{U}_{v,j}(i, l) \mathbf{N}_j \mathbf{V}_{\text{old}}(:, l)$    ▶  $\mathbf{v}_{\text{temp}} = \sum_{j=1}^m u_{j,i}^v \mathbf{N}_j \mathbf{v}_i^{(k-1)}$ 
32:                if  $\sigma_i == \infty$  then
33:                   $\mathbf{V}_{\text{new}}(:, i) = \mathbf{U}_\infty \setminus (\mathbf{L}_\infty \setminus \mathbf{v}_{\text{temp}})$            ▶ Markov parameter
34:                else
35:                   $\mathbf{V}_{\text{new}}(:, i) = \mathbf{U}_{\sigma_i} \setminus (\mathbf{L}_{\sigma_i} \setminus \mathbf{v}_{\text{temp}})$            ▶ Moment
36:               $\mathbf{V}_{\text{old}} = \mathbf{V}_{\text{new}}$    ▶ Overwrite:  $\mathbf{V}^{(k-1)} \leftarrow \mathbf{V}^{(k)}$ 
37:               $\mathbf{V}_{\text{end}} = \mathbf{V}_{\text{end}} + \mathbf{V}_{\text{new}}$    ▶ Sum:  $\mathbf{V}^{(1:N)} = \sum_{k=1}^N \mathbf{V}^{(k)}$ 
38:             $\mathbf{V}^{(1:N)} = \text{orth}(\mathbf{V}_{\text{end}})$    ▶ Optional: a-posteriori orthogonalization

```

## 5.5 $\mathcal{H}_2$ -optimal reduction of bilinear systems

Similarly to the linear setting discussed in Section 3.7, the concept of  $\mathcal{H}_2$ -optimal model reduction can be extended to bilinear systems. In this section we give the  $\mathcal{H}_2$ -optimality conditions for the MIMO case in terms of bitangential Hermite Volterra series interpolation. Then, we discuss our implementation of bilinear IRKA, and finally extend the concept of  $\mathcal{H}_2$ -pseudo-optimality to the bilinear setting.

### 5.5.1 Optimality conditions and bilinear iterative algorithms

For the derivation of first-order optimality conditions the  $\mathcal{H}_2$ -norm of the error system is considered, i.e.  $\mathcal{J} := \|\zeta - \zeta_r\|_{\mathcal{H}_2}^2$ . Zhang and Lam [285] extended the Wilson conditions to the bilinear case by considering the cost functional in terms of the Lyapunov equations (5.25) as  $\mathcal{J} = f(\mathbf{A}_e, \mathbf{N}_{j,e}, \mathbf{B}_e, \mathbf{C}_e, \mathbf{E}_e)$  and by differentiating w.r.t.  $\mathbf{A}_r, \mathbf{N}_{j,r}, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r$ . However, their proposed algorithm relies on gradient flow optimization and is thus not well applicable to large-scale models. Therefore, Benner and Breiten [20] derived interpolation-based conditions with the aim of obtaining an IRKA-like algorithm. They derived the MIMO optimality conditions by differentiating the cost functional  $\mathcal{J} = f(\mathbf{A}, \mathbf{\Lambda}_r, \mathbf{N}_j, \hat{\mathbf{N}}_{j,r}, \mathbf{B}, \hat{\mathbf{B}}_r, \mathbf{C}, \hat{\mathbf{C}}_r)$  w.r.t.  $\hat{\mathbf{C}}_r \in \mathbb{C}^{p \times r}$ ,  $\hat{\mathbf{B}}_r \in \mathbb{C}^{r \times m}$ ,  $\hat{\mathbf{N}}_{j,r} \in \mathbb{C}^{r \times r}$  and  $\mathbf{\Lambda}_r \in \mathbb{C}^{r \times r}$ , leading to  $(p + m + m \cdot r + 1) \cdot r$  conditions. Hereby, the eigendecomposition of the ROM  $[\mathbf{X}_r, \mathbf{\Lambda}_r] = \mathbf{eig}(\mathbf{A}_r, \mathbf{E}_r)$  is exploited to obtain  $\mathbf{E}_r^{-1} \mathbf{A}_r = \mathbf{X}_r \mathbf{\Lambda}_r \mathbf{X}_r^{-1}$ ,  $\hat{\mathbf{N}}_{j,r} = \mathbf{X}_r^{-1} \mathbf{E}_r^{-1} \mathbf{N}_{j,r} \mathbf{X}_r$ ,  $\hat{\mathbf{B}}_r = \mathbf{X}_r^{-1} \mathbf{E}_r^{-1} \mathbf{B}_r$  and  $\hat{\mathbf{C}}_r = \mathbf{C}_r \mathbf{X}_r$ . Later, Flagg and Gugercin [92, 91] showed the connection between the  $\mathcal{H}_2$ -optimality and multipoint Volterra series interpolation conditions for the SISO case. Namely, an  $\mathcal{H}_2$ -optimal ROM satisfies multipoint *Hermite* Volterra series interpolation, with shifts given by the mirror images of the reduced poles  $\sigma_i \leftarrow -\bar{\lambda}_{r,i}$  and weights given by the reduced-order residues  $\eta_{l_1, \dots, l_{k-1}, i} \leftarrow \phi_{r, l_1, \dots, l_{k-1}, i}$  with  $\phi_{r, l_1, \dots, l_{k-1}, i} = \hat{\mathbf{c}}_{r,i} \cdot \hat{n}_{r, l_{k-1}} \cdot \dots \cdot \hat{n}_{r, l_2, l_1} \cdot \hat{\mathbf{b}}_{r, l_1}$  (cf. Theorem 5.6).

With our results on the MIMO pole-residue (Eq. (5.5)) and Volterra series interpolation (Theorem 5.7), we can give the  $\mathcal{H}_2$ -optimality conditions in terms of bitangential Hermite interpolation with  $\mathbf{r}_{l_1} \leftarrow \hat{\mathbf{b}}_{r, l_1}$ ,  $\mathbf{l}_i^\top \leftarrow \hat{\mathbf{c}}_{r, i}^\top$  and  $\eta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \vartheta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \leftarrow \hat{n}_{j_k, r, l_{k-1}} \cdot \dots \cdot \hat{n}_{j_2, r, l_2, l_1}$ :

$$\begin{aligned} & \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \eta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, l_1}, \dots, -\bar{\lambda}_{r, l_{k-1}}, -\bar{\lambda}_{r, i}) \hat{\mathbf{b}}_{r, l_1} \\ &= \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \eta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \mathbf{G}_{k, r, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, l_1}, \dots, -\bar{\lambda}_{r, l_{k-1}}, -\bar{\lambda}_{r, i}) \hat{\mathbf{b}}_{r, l_1}, \end{aligned} \quad (5.59a)$$

$$\begin{aligned} & \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \vartheta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \hat{\mathbf{c}}_{r, i}^\top \mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, i}, -\bar{\lambda}_{r, l_{k-1}}, \dots, -\bar{\lambda}_{r, l_1}) \\ &= \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \vartheta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \hat{\mathbf{c}}_{r, i}^\top \mathbf{G}_{k, r, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, i}, -\bar{\lambda}_{r, l_{k-1}}, \dots, -\bar{\lambda}_{r, l_1}), \end{aligned} \quad (5.59b)$$

$$\begin{aligned} & \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \eta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \hat{\mathbf{c}}_{r, i}^\top \left( \sum_{h=1}^k \frac{\partial}{\partial s_h} \mathbf{G}_{k, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, l_1}, \dots, -\bar{\lambda}_{r, i}) \right) \hat{\mathbf{b}}_{r, l_1} \\ &= \sum_{k=1}^{\infty} \sum_{j_2}^m \dots \sum_{j_k}^m \sum_{l_1}^r \dots \sum_{l_{k-1}}^r \eta_{l_1, \dots, l_{k-1}, i}^{(j_2, \dots, j_k)} \hat{\mathbf{c}}_{r, i}^\top \left( \sum_{h=1}^k \frac{\partial}{\partial s_h} \mathbf{G}_{k, r, \square}^{(j_2, \dots, j_k)}(-\bar{\lambda}_{r, l_1}, \dots, -\bar{\lambda}_{r, i}) \right) \hat{\mathbf{b}}_{r, l_1}, \end{aligned} \quad (5.59c)$$

for  $i = 1, \dots, r$ . This result can be obtained by rearranging the optimality conditions in Kronecker notation using the Neumann series and the vectorization operator backwards. The interested reader is referred to [Röt18, Sec. 4.4] for more details.

### (Truncated) Bilinear Iterative Rational Krylov Algorithm (BIRKA)

Using an IRKA-like fixed-point iteration, Benner and Breiten [20] proposed the bilinear iterative rational Krylov algorithm (BIRKA) based on the solution of the bilinear Sylvester equations (5.51). Besides showing the connection of BIRKA to *infinite* Volterra series interpolation, Flagg also proposed a more efficient, *truncated* version of the algorithm (TB-IRKA) based on the solution of the linear Sylvester equations (5.52).

During this doctoral project we have implemented both strategies in the BSSMOR function `b_irka` (see Algorithm 5.2). If the user wants to perform infinite interpolation (and the FOM is not too big), then the bilinear Sylvester equations are solved for the updated interpolation data via (5.54). Otherwise we perform TB-IRKA, whereby we solve the sparse LSEs (5.53) in an Arnoldi-like manner rather than solving (5.52) (using MATLAB's `lyap` or by implementing the LR-ADI for the Sylvester case).

---

#### Algorithm 5.2 (Truncated) Bilinear iterative rational Krylov algorithm (BIRKA)

---

**Input:** Bilinear model  $\zeta = (\mathbf{A}, \mathbf{N}_j, \mathbf{B}, \mathbf{C}, \mathbf{E})$ , reduced order  $r$ , convergence tolerance  $\varepsilon$

**Output:** locally  $\mathcal{H}_2$ -optimal ROM  $\zeta_r^{\text{opt}}$ , optimal interpolation data

- 1: Choose initial shifts  $\mathbf{S}_v = \mathbf{S}_w \leftarrow \{\sigma_i\}_{i=1}^r \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$ , initial tangential directions  $\mathbf{R} \leftarrow \{\mathbf{r}_i\}_{i=1}^r \in \mathbb{C}^m$ ,  $\mathbf{L} \leftarrow \{\mathbf{l}_i\}_{i=1}^r \in \mathbb{C}^p$ , and initial weights  $\{\mathbf{U}_{v,j} = \mathbf{U}_{w,j}\}_{j=1}^m \in \mathbb{C}^{r \times r}$
- 2: **while** relative change in  $\{\sigma_i\} > \varepsilon$  **do**
- 3:   Solve bilinear Sylvester equations   ▶ **compute projection matrices**

$$\mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} - \sum_{j=1}^m \mathbf{N}_j \mathbf{V} \mathbf{U}_{v,j}^\top = \mathbf{B} \mathbf{R}$$

$$\mathbf{E}^\top \mathbf{W} \mathbf{S}_w^\top - \mathbf{A}^\top \mathbf{W} - \sum_{j=1}^m \mathbf{N}_j^\top \mathbf{W} \mathbf{U}_{w,j}^\top = \mathbf{C}^\top \mathbf{L}$$

Alternative: solve linear equations (5.52) or call `volterraBrk` (Alg. 5.1, Eq. (5.53))

- 4:  $\mathbf{A}_r = \mathbf{W}^\top \mathbf{A} \mathbf{V}$ ,  $\mathbf{N}_{j,r} = \mathbf{W}^\top \mathbf{N}_j \mathbf{V}$ ,  $\mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$ ,  $\mathbf{C}_r = \mathbf{C} \mathbf{V}$ ,  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E} \mathbf{V}$
  - 5:  $\mathbf{E}_r^{-1} \mathbf{A}_r = \mathbf{X}_r \mathbf{\Lambda}_r \mathbf{X}_r^{-1}$  with  $\mathbf{\Lambda}_r = \text{diag}(\lambda_{r,1}, \dots, \lambda_{r,r})$    ▶  $[\mathbf{X}_r, \mathbf{\Lambda}_r] = \text{eig}(\mathbf{A}_r, \mathbf{E}_r)$
  - 6:  $\mathbf{S}_v = \mathbf{S}_w \leftarrow -\overline{\mathbf{\Lambda}}_r$ ,  $\mathbf{U}_{v,j} = \mathbf{U}_{w,j} \hat{=} \hat{\mathbf{N}}_{j,r} \leftarrow \mathbf{X}_r^{-1} \mathbf{E}_r^{-1} \mathbf{N}_{j,r} \mathbf{X}_r$
  - 7:  $\mathbf{R} \hat{=} \hat{\mathbf{B}}_r^\top \leftarrow (\mathbf{X}_r^{-1} \mathbf{E}_r^{-1} \mathbf{B}_r)^\top$ ,  $\mathbf{L} \hat{=} \hat{\mathbf{C}}_r \leftarrow \mathbf{C}_r \mathbf{X}_r$
  - 8:  $\Sigma_r^{\text{opt}} \leftarrow (\mathbf{A}_r, \mathbf{N}_{j,r}, \mathbf{B}_r, \mathbf{C}_r, \mathbf{E}_r)$ ,  $\sigma_i^{\text{opt}} \leftarrow \sigma_i$ ,  $\mathbf{r}_i^{\text{opt}} \leftarrow \hat{\mathbf{b}}_{r,i}$ ,  $\mathbf{l}_i^{\text{opt}} \leftarrow \hat{\mathbf{c}}_{r,i}$ ,  $\mathbf{U}_{v,j}^{\text{opt}} = \mathbf{U}_{w,j}^{\text{opt}} \leftarrow \hat{\mathbf{N}}_{j,r}$
- 



BSSMOR function(s): `b_irka` (`volterraBrk`, `volterraBarnoldi`)

**Initialization** In the literature [20, 92, 91], BIRKA is usually initialized with random interpolation data or random reduced-order matrices. We advise against this procedure, since it is not physically motivated and might significantly slow down the convergence. Instead, we recommend to use some system knowledge or apply similar guidelines as discussed in Section 5.4.3. For instance, one can use `s0=zeros(1,r)`,  $\mathbf{r}_i = \mathbf{1}_m$  and  $\mathbf{U}_v = \alpha \cdot \mathbf{I}_r$  with  $\alpha \approx 10^{-3}, \dots, 10^2$ , or exploit some eigenvalues (and eigenvectors) of the FOM calculated via  $[\mathbf{X}, \mathbf{\Lambda}] = \text{eigs}(\mathbf{A}, \mathbf{E})$ .

### 5.5.2 $\mathcal{H}_2$ -pseudo-optimality for bilinear systems

The interest of the author on bilinear systems originally arised with the aim of extending the concept of  $\mathcal{H}_2$ -pseudo-optimality to the bilinear case. The goal is again to find a reduced model that minimizes the  $\mathcal{H}_2$ -error within a certain subset  $\mathcal{G}$  of ROMs of order  $r$ :

$$\zeta_r = \arg \min_{\tilde{\zeta}_r \in \mathcal{G}} \|\zeta - \tilde{\zeta}_r\|_{\mathcal{H}_2}^2. \quad (5.60)$$

The subset can be chosen to have ROMs with fixed reduced poles, weights and input residues  $(\lambda_{r,i}, \hat{N}_{j,r}, \hat{\mathbf{b}}_{r,i})$ , or fixed reduced poles, weights and output residues  $(\lambda_{r,i}, \hat{N}_{j,r}, \hat{\mathbf{c}}_{r,i})$ . The idea is thus to fix three of the four degrees of freedom, in order to simplify the optimization and easily obtain the remaining parameter. Our motivation for the extension was to gain similar advantages and application fields as in the linear case, namely the convexity of the  $\mathcal{H}_2$ -pseudo-optimal problem, the construction of stable bilinear ROMs, a residue correction for a better convergence of BIRKA, the link between the bilinear ADI and RKSM [92, Sec. 5.4], etc.

Following similar steps as in [273, Appendix A], we have derived new conditions for (input and output) pseudo-optimality for bilinear systems. For the sake of brevity the conditions are not explicitly presented here, since this would require introducing a few more concepts and equations. The interested reader is referred to the online available talks [Cru16a, COL16, Cru17a] for more details, as well as to the master thesis [Olc16] where the complete derivation and proof is shown. We only state the resulting iteration-free `bporKV` algorithm for the explicit construction of bilinear (input)  $\mathcal{H}_2$ -pseudo-optimal ROMs (using the notation of this thesis).

---

#### Algorithm 5.3 Bilinear (input) pseudo-optimal rational Krylov (BPORK-V)

---

**Input:**  $V, S_v, U_{v,j}, R$  such that  $EV S_v - AV - \sum_{j=1}^m N_j V U_{v,j}^\top = BR$ , output matrix  $C$

**Output:**  $\mathcal{H}_2$ -pseudo-optimal ROM  $\zeta_r^{\text{opt}}$

- 1:  $P_r^{-1}$    ▶ solution of condition iii):  $P_r^{-1} S_v + S_v^\top P_r^{-1} - \sum_{j=1}^m U_{v,j} P_r^{-1} U_{v,j}^\top - R^\top R = 0$
  - 2:  $N_{j,r} = (P_r^{-1})^{-1} U_{v,j} P_r^{-1}$    ▶ condition ii-2):  $P_r^{-1} U_{v,j}^\top - N_{j,r}^\top E_r^{-\top} P_r^{-1} = 0$
  - 3:  $B_r = (P_r^{-1})^{-1} R^\top$    ▶ condition ii-1):  $E_r^{-1} B_r - P_r R^\top = 0$
  - 4:  $A_r = S_v - B_r R - \sum_{j=1}^m N_{j,r} U_{v,j}^\top$ ,  $E_r = I_r$ ,  $C_r = CV$
  - 5:  $\zeta_r^{\text{opt}} = (A_r, N_{j,r}, B_r, C_r, I_r)$
- 



BSSSMOR function(s): `bporKV`, `bporKW`

Similar to the linear case, the algorithm needs a projection matrix  $V$  spanning an input Krylov subspace for the selected interpolation data  $(S_v, U_{v,j}, R)$ , i.e. which satisfies the Sylvester equation (5.51a). Then, BPORK constructs an  $\mathcal{H}_2$ -pseudo-optimal ROM with low computational effort. The procedure relies on the solution of a bilinear Lyapunov equation (Line 1) and two LSEs (Lines 2, 3), both of reduced order  $r$ . Please note that the ROM constructed via BPORK-V/W satisfies only the Lagrangian condition (5.59a)/(5.59b) and has its poles at the mirror images of the selected shifts. Moreover, the weights are also being chosen by the user, so that – according to Theorem 5.1 – the stability of the ROM can be

enforced by construction. Finally, note that (similarly to PORK) BPORK is actually not a projection-based reduction method, but rather constructs the ROM using the concept of parametrized families of reduced-order models (see [126, 128] and 3.6.4, 7.1.4). This can be seen from the fact that  $\mathbf{W}$  ( $\mathbf{V}$  in BPORK-W) is not explicitly needed to obtain the ROM.

In the talk [Cru17a], we can observe via a simple example that the  $\mathcal{H}_2$ -optimal problem becomes indeed convex if three of the four optimization parameters are fixed. Furthermore, we discuss the already mentioned possible applications of  $\mathcal{H}_2$ -pseudo-optimality of bilinear systems. Due to the sake of time (and other research interests) these topics could not be deepened during this PhD project. The residue correction within BIRKA, the extension of CuRed SPARK to bilinear systems and the application of RKSM+BPORK-Lyap are thus topics of future research.

## 5.6 Conclusions and other contributions

At the end of this chapter we want to recap our *main* contributions regarding bilinear model order reduction:

1. We have generalized the pole-residue formulation of bilinear systems to the MIMO case. This result helps to understand the  $\mathcal{H}_2$ -optimality conditions better.
2. We have presented a different strategy for MIMO subsystem interpolation (MIMO-2). The approach is motivated by the term  $\sum_{j=1}^m \mathbf{N}_j \mathbf{V} \mathbf{U}_{v,j}^T$  in the Sylvester equations.
3. We have generalized the Volterra series interpolation to the MIMO case, extended the framework to the multimoment setting and focused on the Arnoldi-like implementation. This yields the `volterraBarnoldi` algorithm, supporting many different cases.
4. We have extended the concept of  $\mathcal{H}_2$ -pseudo-optimality to bilinear systems, hereby providing new necessary conditions and an iteration-free algorithm called BPORK.
5. We have initiated the development of the BSSMOR toolbox, following similar philosophy and implementation patterns as the sssMOR toolbox. BSSMOR can serve as a starting point for new research endeavors.

In the following we also want to mention other contributions that have not been explicitly addressed within this dissertation:

1. In [60] an expression for the impulse response of bilinear systems has been derived. The derivation is based on applying an input of the form  $\mathbf{u}(t) = \boldsymbol{\mu} \delta(t)$  (with the Dirac function  $\delta(t)$  and the scaling  $\boldsymbol{\mu}$ ) to the *whole* bilinear system, i.e. without employing the Volterra model. As one would expect from nonlinear systems (where the superposition principle does not hold), the impulse response  $\mathbf{g}(t; \boldsymbol{\mu})$  is dependent on the amplitude of the input  $\boldsymbol{\mu}$ . In the thesis [Geb17] the corresponding transfer function  $\mathbf{G}(s; \boldsymbol{\mu})$  is also given, together with Krylov subspaces for model reduction based on this novel function. Expressions for the  $\mathcal{H}_2$ -norm based on the derived impulse response are also stated.
2. In the master thesis [Fio16] and talks [Cru16b, CFL17, Cru17b] we focus on the derivation of transfer matrices and Krylov-based reduction of *MIMO* quadratic-bilinear systems. Due to the more complex structure of the transfer matrices, the generalization of the subsystem interpolation framework [22] to the MIMO case is not trivial. We propose different (tangential) Krylov subspaces to achieve both multimoment and Hermite interpolation of the first two subsystems. We also deal with one-/two-sided reduction as

well as with stability preservation. The algorithms are implemented in the QBSSMOR toolbox. Although the development of QBSSMOR is rather at an early stage, this toolbox can also help to conduct future research.



## PART III

### NONLINEAR STATE-SPACE SYSTEMS



# Chapter 6

## Fundamentals of Nonlinear Model Reduction

In this part of the thesis we focus on the reduction of *general nonlinear* state-space systems. First, the considered system representation and the numerical time integration of the FOM is discussed. Afterwards, we will concentrate on model reduction based on linear and nonlinear projection. The corresponding framework, time integration and reduction techniques will be presented in order to understand the advantages and disadvantages of both strategies.

Sections 6.1, 6.2, 6.3.1 and 6.4.1 rely on the corresponding parts of [72] and [71], whereby they have been extended here. The chapter is also equipped with a section on hyper-reduction.

### 6.1 Nonlinear time-invariant systems

Consider a large-scale, nonlinear time-invariant, exponentially stable, MIMO state-space model of the form

$$\Sigma_{\text{NL}} : \begin{cases} \mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), & \mathbf{x}(0) = \mathbf{x}_0, \\ \mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t)), \end{cases} \quad (6.1a)$$

$$(6.1b)$$

with non-singular descriptor matrix  $\mathbf{E} \in \mathbb{R}^{n \times n}$ , the vectors  $\mathbf{x}(t) \in \mathbb{R}^n$ ,  $\mathbf{u}(t) \in \mathbb{R}^m$ ,  $\mathbf{y}(t) \in \mathbb{R}^p$  and the smooth mappings  $\mathbf{f}(\mathbf{x}, \mathbf{u}) : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$  and  $\mathbf{h}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^p$ , such that  $\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$  and  $\mathbf{h}(\mathbf{0}) = \mathbf{0}$ . The zero equilibrium  $\mathbf{x}_{\text{eq}} = \mathbf{0}$ , calculated from  $\mathbf{0} = \mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{0})$ , is assumed locally exponentially stable.

Note that the above considered equations constitute a very general representation<sup>1</sup> for a nonlinear time-invariant system. For instance, in case of an *input-affine* nonlinear system (4.2), we obtain the mapping  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{a}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{u} = \mathbf{a}(\mathbf{x}) + \sum_{j=1}^m \mathbf{b}_j(\mathbf{x}) u_j$ , with  $\mathbf{a}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$  and  $\mathbf{B}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m}$  or  $\mathbf{b}_j(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ . Moreover, for a linear output mapping one obtains the function  $\mathbf{h}(\mathbf{x}) = \mathbf{C}\mathbf{x}$  with  $\mathbf{C} \in \mathbb{R}^{p \times n}$ .

Unfortunately, unlike the linear case, the input-output behavior of general nonlinear systems such as (6.1) cannot be described analytically with the help of transfer functions, the state-transition matrix or convolution integrals. Such a characterization via the mentioned system-theoretic concepts is only possible for polynomial nonlinear systems (like e.g. bilinear or quadratic-bilinear systems) by exploiting the Volterra series representation [221, 45, 63].

<sup>1</sup>A more general representation for a dynamical system would read

$$\mathbf{0} = \mathbf{f}(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\mu}(t), t), \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (6.2a)$$

$$\mathbf{0} = \mathbf{h}(\mathbf{y}(t), \mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\mu}(t), t), \quad (6.2b)$$

describing a nonlinear, first-order, time-variant, parameter-varying DAE system.

## 6.2 Time integration

Numerical time integration is a fundamental tool in order to solve systems of ordinary (ODEs) or differential-algebraic (DAEs) equations. Especially for nonlinear systems, where the solution  $\mathbf{x}(t)$  can generally not be given analytically, numerical simulation is indispensable. There exist several time integration schemes. Depending on their properties, one can generally distinguish between one/multistep, explicit/implicit and variable/fixed step solvers. [243, 237]

One-step solvers (such as Euler's method) employ only one step in time to determine the next solution. Multi-stage solvers – such as the Crank-Nicolson (second-order approach based on trapezoidal rule) or Runge-Kutta methods – take some intermediate steps, but then discard this information before taking a second step. Multistep solvers take and exploit the information from several time-steps to obtain the next solution.

Explicit solvers (e.g. one-step forward Euler, explicit multi-stage Runge-Kutta methods) calculate the state  $\mathbf{x}_{k+1}$  at the next time-step using explicitly the state  $\mathbf{x}_k$  at the current time-step. For example, the forward Euler method yields the explicit equation

$$\mathbf{E} \mathbf{x}_{k+1} = \mathbf{E} \mathbf{x}_k + h \cdot \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \quad \text{with } t_{k+1} = t_k + h. \quad (6.3)$$

Implicit solvers (e.g. one-step backward Euler, implicit multi-stage Runge-Kutta methods) compute the next state  $\mathbf{x}_{k+1}$  by solving a nonlinear system of equations (NLSE) involving both the current and the next state. For instance, the backward Euler method yields the implicit equation

$$\mathbf{E} \mathbf{x}_{k+1} = \mathbf{E} \mathbf{x}_k + h \cdot \mathbf{f}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \quad \text{with } t_{k+1} = t_k + h. \quad (6.4)$$

This type of solvers are particularly applied to *stiff* problems, for which explicit methods require an impractically small step-size  $h$  to be numerically stable. Thus, for such problems, an implicit solver with larger step-size usually needs less computational time (despite the NLSEs) than an explicit solver with smaller step-size. Note that in some applications a so-called *implicit-explicit* (IMEX) scheme is rather applied, where the stiff (linear) part of the problem is treated implicitly and the non-stiff (nonlinear) one explicitly [9]. Due to this splitting, the solution of NLSEs required in a purely implicit scheme is avoided. Further note that so-called *predictor-corrector* schemes typically employ an explicit method for the predictor step and an implicit formula for the corrector step. A well-known member of this family of methods is the generalized- $\alpha$  scheme, which will be treated in Algorithm 9.1.

Fixed step solvers use the same step-size  $h$  during the whole simulation, whereas variable step solvers (e.g. MATLAB's explicit `ode23`, `ode45` and implicit `ode15s`, `ode23s`) adapt it depending on the dynamics. They reduce the step-size at highly dynamic events, and increase it at slowly changing regions to avoid taking unnecessary steps. Thus, variable step solvers might need less computational time (despite the step-size adaption) than fixed step ones.

### Implicit Euler and Newton-Raphson method

In this thesis, a self-programmed fixed step implicit Euler scheme (cf. Algorithm 6.1) is employed for the numerical simulation of FOM and ROMs. Using the backward Euler formula (6.4) leads to the residual (cf. line 3)

$$\mathit{res}(\mathbf{x}_{k+1}) = \mathbf{E} \mathbf{x}_k - \mathbf{E} \mathbf{x}_{k+1} + h \cdot \mathbf{f}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \stackrel{!}{=} \mathbf{0}. \quad (6.5)$$

A root-finding algorithm such as the Newton-Raphson method (see e.g. [193]) is then employed to solve the NLSE (6.5) in each time-step (cf. line 5).

The Newton-Raphson method (cf. Algorithm 6.2) iteratively searches for the solution  $\mathbf{x}_{k+1}$  of a nonlinear system of equations  $\mathbf{res}(\mathbf{x}_{k+1}) = \mathbf{0}$  by successive approximations. The linearization of the residual equation around the solution  $\mathbf{x}_{k+1}^{\text{iter}}$  at the current iteration  $\text{iter}$  yields the following linear system of equations (LSE)

$$\underbrace{\frac{\partial \mathbf{res}(\mathbf{x}_{k+1})}{\partial \mathbf{x}_{k+1}} \bigg|_{\mathbf{x}_{k+1} = \mathbf{x}_{k+1}^{\text{iter}}}}_{=: \mathbf{J}_{\mathbf{res}}^{\text{iter}}(\mathbf{x}_{k+1}^{\text{iter}})} \Delta \mathbf{x}_{k+1}^{\text{iter}} = \mathbf{res}(\mathbf{x}_{k+1}^{\text{iter}}), \quad (6.6)$$

which can be solved for the correction  $\Delta \mathbf{x}_{k+1}^{\text{iter}}$  using a direct or iterative solver (cf. Section 2.5). The update then reads  $\mathbf{x}_{k+1}^{\text{iter}+1} = \mathbf{x}_{k+1}^{\text{iter}} - \Delta \mathbf{x}_{k+1}^{\text{iter}}$  (cf. line 10).

In case of the implicit Euler scheme, the analytical Jacobian of the residual (6.5) reads

$$\mathbf{J}_{\mathbf{res}}^{\text{iter}}(\mathbf{x}_{k+1}^{\text{iter}}) = -\mathbf{E} + h \cdot \mathbf{A}(\mathbf{x}_{k+1}^{\text{iter}}, \mathbf{u}_{k+1}), \quad (6.7)$$

where  $\mathbf{A}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} \bigg|_{(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})}$  represents the Jacobian of  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  at  $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$ .

---

#### Algorithm 6.1 Implicit Euler scheme

---

**Input:**  $\mathbf{E}$ ,  $\mathbf{f}(\mathbf{x}, \mathbf{u})$ ,  $\mathbf{A}(\mathbf{x}, \mathbf{u})$ ,  $\mathbf{t}_{\text{sim}} = [t_0, \dots, t_{\text{end}}]$ ,  $\mathbf{x}_0$

**Output:** solution  $\mathbf{x}(t)$

- 1:  $h = t_1 - t_0$
  - 2: **for**  $k = 0 : \text{length}(\mathbf{t}_{\text{sim}}) - 2$  **do**
  - 3:      $\text{fun}(\mathbf{x}_{k+1}) = \mathbf{E} \mathbf{x}_k - \mathbf{E} \mathbf{x}_{k+1} + h \cdot \mathbf{f}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1})$      ▶ residual (6.5)
  - 4:      $\mathbf{J}\text{fun}(\mathbf{x}_{k+1}) = -\mathbf{E} + h \cdot \mathbf{A}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1})$      ▶ Jacobian of residual (6.7)
  - 5:      $\mathbf{x}_{k+1} = \text{NewtonRaphson}(\text{fun}, \mathbf{x}_k, \mathbf{J}\text{fun})$      ▶ call Alg. 6.2
  - 6:      $\mathbf{x}_k \leftarrow \mathbf{x}_{k+1}$
- 

---

#### Algorithm 6.2 Newton-Raphson method

---

**Input:**  $\text{fun}(\mathbf{x})$ , initial guess  $\mathbf{x}_0$ ,  $\mathbf{J}\text{fun}$ ,  $\text{Opts}$      ▶ analytical or numerical Jacobian  $\mathbf{J}\text{fun}$

**Output:** root  $\mathbf{x}$

- 1:  $\text{tol} = \text{Opts.AbsTol}$
  - 2:  $\mathbf{x}_{\text{curr}} = \mathbf{x}_0$ ,  $\mathbf{f}_{\text{curr}} = \text{fun}(\mathbf{x}_{\text{curr}})$
  - 3:  $\text{iter} = 0$
  - 4: **while**  $\text{norm}(\mathbf{f}_{\text{curr}}) > \text{tol}$  **do**     ▶ norm of residual as stopping criterion
  - 5:      $\text{iter} = \text{iter} + 1$
  - 6:     **if**  $\text{iter} > \text{Opts.MaxIter}$  **then**     ▶  $\text{Opts.MaxIter}$  as further stopping criterion
  - 7:         **break**
  - 8:      $\mathbf{f}_{\text{curr}} = \text{fun}(\mathbf{x}_{\text{curr}})$
  - 9:      $\mathbf{d}\mathbf{x}_{\text{curr}} = \mathbf{J}\text{fun}(\mathbf{x}_{\text{curr}}) \setminus \mathbf{f}_{\text{curr}}$      ▶ solve LSE (6.6)
  - 10:      $\mathbf{x}_{\text{curr}} = \mathbf{x}_{\text{curr}} - \mathbf{d}\mathbf{x}_{\text{curr}}$
  - 11:      $\text{tol} = \text{Opts.RelTol} * \text{norm}(\mathbf{f}_{\text{curr}}) + \text{Opts.AbsTol}$      ▶ tolerance update
  - 12:  $\mathbf{x} = \mathbf{x}_{\text{curr}}$
-

### 6.3 Nonlinear reduction based on linear projection

The goal of nonlinear model order reduction is to approximate the FOM (6.1) by a ROM of smaller dimension  $r \ll n$ . Similar to the linear case, the reduction of nonlinear systems is also usually performed within a projection framework. In the nonlinear case, however, we distinguish between linear and nonlinear Petrov-Galerkin projection.

In this section, the reduction of nonlinear systems based on *linear projection* is discussed. The reduction based on nonlinear projection will be treated in Section 6.4.

#### 6.3.1 Linear Petrov-Galerkin projection framework

An established and successful way to reduce nonlinear systems is to apply the classical Petrov-Galerkin projection framework, where *linear mappings* given by the matrices  $\mathbf{V}$ ,  $\mathbf{W}$  are used. In the following, we briefly derive the ROM in a similar fashion as described in Section 3.2.

Inserting the linear approximation ansatz (3.26) into the FOM (6.1a) and premultiplying the overdetermined system with the projector  $\mathbf{\Pi}$  leads to

$$\mathbf{\Pi} \left( \underbrace{\mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r(t) - \mathbf{f}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t))}_{=\boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t))} - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0} \Leftrightarrow \mathbf{\Pi} \left( \boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t)) - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0}. \quad (6.8)$$

Enforcing the Petrov-Galerkin condition  $\mathbf{W}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\mathbf{\Pi} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , the residual then vanishes and only the term  $\mathbf{\Pi} \boldsymbol{\xi}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t)) = \mathbf{0}$  remains. This finally yields the ROM

$$\mathbf{E}_r \dot{\mathbf{x}}_r(t) = \mathbf{W}^\top \mathbf{f}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t)), \quad \mathbf{x}_r(0) = \mathbf{x}_{r,0}, \quad (6.9a)$$

$$\mathbf{y}_r(t) = \mathbf{h}(\mathbf{V} \mathbf{x}_r(t)), \quad (6.9b)$$

with  $\mathbf{E}_r = \mathbf{W}^\top \mathbf{E} \mathbf{V}$  and  $\mathbf{x}_r(0) = (\mathbf{W}^\top \mathbf{E} \mathbf{V})^{-1} \mathbf{W}^\top \mathbf{E} \mathbf{x}(0)$  given as in the linear case, and the reduced nonlinear function  $\mathbf{f}_r(\mathbf{x}_r(t), \mathbf{u}(t)) = \mathbf{W}^\top \mathbf{f}(\mathbf{V} \mathbf{x}_r(t), \mathbf{u}(t))$  with  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) : \mathbb{R}^r \times \mathbb{R}^m \rightarrow \mathbb{R}^r$ .

Please note that the use of a linear projection  $\mathbf{x} \approx \mathbf{V} \mathbf{x}_r$  constitutes a special case of the power series ansatz (6.28), and hence of the most general nonlinear projection  $\mathbf{x} \approx \boldsymbol{\nu}(\mathbf{x}_r)$  described in Section 6.4. Further note that the main tasks of model reduction in this setting are twofold: (i) the efficient computation of reduction bases  $\mathbf{V}$ ,  $\mathbf{W}$  for the *dimensional reduction* (cf. Section 6.3.3), and (ii) the application of so-called *hyper-reduction* methods for the effective evaluation of the nonlinear terms (cf. Section 6.5). Finally, note that nonlinear systems are usually reduced by an orthogonal (Galerkin) projection (e.g. with  $\mathbf{W} = \mathbf{V}$ ) rather than by an oblique (Petrov-Galerkin) projection. An alternative *least-squares* Petrov-Galerkin (LSPG) projection framework has also been proposed in [55, 54] for reduction.

#### 6.3.2 Time integration

We now briefly discuss the time integration of the ROM (6.9). For the case of an implicit Euler scheme (cf. Eq. (6.4)), the residual reads

$$\begin{aligned} \text{res}(\mathbf{x}_{r,k+1}) &= \mathbf{E}_r \mathbf{x}_{r,k} - \mathbf{E}_r \mathbf{x}_{r,k+1} + h \cdot \mathbf{f}_r(\mathbf{x}_{r,k+1}, \mathbf{u}_{k+1}) \\ &= \mathbf{W}^\top \left( \mathbf{E} \mathbf{V} \mathbf{x}_{r,k} - \mathbf{E} \mathbf{V} \mathbf{x}_{r,k+1} + h \cdot \mathbf{f}(\mathbf{V} \mathbf{x}_{r,k+1}, \mathbf{u}_{k+1}) \right), \end{aligned} \quad (6.10)$$

with the *constant* reduced matrix  $\mathbf{E}_r$  and reduced nonlinear function  $\mathbf{f}_r(\mathbf{V}\mathbf{x}_{r,k+1}, \mathbf{u}_{k+1})$  with  $\mathbf{x}_{k+1} \approx \mathbf{V}\mathbf{x}_{r,k+1}$ . The analytical Jacobian of the residual is given by

$$\mathbf{J}_{res}^{\text{iter}}(\mathbf{x}_{r,k+1}^{\text{iter}}) = \left. \frac{\partial res(\mathbf{x}_{r,k+1})}{\partial \mathbf{x}_{r,k+1}} \right|_{\mathbf{x}_{r,k+1} = \mathbf{x}_{r,k+1}^{\text{iter}}} = -\mathbf{E}_r + h \cdot \mathbf{W}^\top \mathbf{A}(\mathbf{V}\mathbf{x}_{r,k+1}^{\text{iter}}, \mathbf{u}_{k+1}) \mathbf{V}, \quad (6.11)$$

and depends on the Jacobian  $\mathbf{A}(\mathbf{x}, \mathbf{u})$  of the nonlinear function  $\mathbf{f}(\mathbf{x}, \mathbf{u})$ . Replacing lines 3 and 4 of Algorithm 6.1 by Equations (6.10) and (6.11) leads to the implicit Euler scheme for the ROM (6.9).

### 6.3.3 Linear projection-based reduction approaches

In this section, we give an overview of existing approaches to calculate the bases  $\mathbf{V}, \mathbf{W}$  needed for the linear projection-based reduction of nonlinear systems. The bases should generally comprise enough information about the nonlinear dynamics, in order to yield a good approximation despite the use of a linear projection.

#### Pure linear bases applied to nonlinear system

A very simple approach consists in calculating pure linear bases  $\mathbf{V}, \mathbf{W}$  based on a linearization, and then apply them to the nonlinear system. If this “naive” approach returns accurate results for the application at hand, then the employment of more sophisticated nonlinear MOR techniques might not be necessary at all.

The first step to construct pure linear bases is to linearize the nonlinear system (6.1) around a linearization  $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$  or equilibrium point  $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$ .

**Linearization point** The choice of a good linearization/equilibrium tuple is highly problem-dependent, since nonlinear systems are usually linearized around a physically meaningful (stationary) point of operation. Very often, the steady-state solution  $\mathbf{x}_{\text{eq}}$  is computed for an appropriately selected stationary input  $\mathbf{u}_{\text{eq}}$  by solving the NLSE  $\mathbf{0} = \mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$  (via e.g. MATLAB’s `fsolve` or `NewtonRaphson` from 6.2). Nevertheless, one could also give a stationary state  $\mathbf{x}_{\text{eq}}$  using some system knowledge and then compute the stationary input  $\mathbf{u}_{\text{eq}}$ , or even give both if enough system understanding is available. Another popular approach is to select  $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = (\mathbf{0}, \mathbf{0})$ , which usually implies  $\mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = \mathbf{0}$ . The choice of a linearization point  $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$  that does not represent an equilibrium point is also possible.

Once the equilibrium point is chosen, the linearized matrices

$$\mathbf{A}_{\text{eq}} = \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} \right|_{(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})}, \quad \mathbf{B}_{\text{eq}} = \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}} \right|_{(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})}, \quad \mathbf{C}_{\text{eq}} = \left. \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}_{\text{eq}}} \quad (6.12)$$

can be obtained. These matrices are then employed to run any linear MOR technique, like modal truncation, balanced truncation, standard rational Krylov or the  $\mathcal{H}_2$ -optimal algorithm IRKA. The computed reduction bases (e.g.  $\mathbf{V}_{\text{RK}}, \mathbf{W}_{\text{RK}}$ ) are finally applied to the nonlinear system using a linear projection (cf. Eq. (6.9)).

The accuracy of this approach is expected to be bad for large deviations from the linearization point, since the nonlinear dynamics might then not be captured by the pure linear basis. Nevertheless, this simple procedure might be worth a try and can serve as a reference.

### Augmented basis

Another popular, and more sophisticated, approach consists in enriching a pure linear reduction basis with nonlinear information, e.g. as follows:

$$\mathbf{x}(t) \approx \mathbf{V}_{\text{aug}} \mathbf{x}_{\text{r,aug}}(t), \quad \mathbf{V}_{\text{aug}} = [\mathbf{V}^{(1)}, \mathbf{V}^{(2)}], \quad (6.13)$$

with the reduced coordinates  $\mathbf{x}_{\text{r,aug}}(t) = [x_{\text{r},1}(t), \dots, x_{\text{r},r}(t) \mid x_{\text{r},11}(t), \dots, x_{\text{r},1r}(t), \dots, x_{\text{r},rr}(t)]^T$ . Herein,  $\mathbf{V}^{(1)} \in \mathbb{R}^{n \times r}$  contains information about the linearized system, whereas  $\mathbf{V}^{(2)} \in \mathbb{R}^{n \times r^2}$  comprises some nonlinear behavior. Some possible enrichment methods are:

**Subsystem interpolation** This approach is known in the context of polynomial model reduction using the Volterra series representation [206, 92, 22]. The matrices  $\mathbf{V}^{(1)}$  and  $\mathbf{V}^{(2)}$  are constructed as bases of input Krylov subspaces for the first and second subsystem and then concatenated to obtain  $\mathbf{V}_{\text{aug}}$  (cf. Section 5.3). A similar approach can be employed with output Krylov subspaces to calculate  $\mathbf{W}_{\text{aug}} = [\mathbf{W}^{(1)}, \mathbf{W}^{(2)}]$ .

**Eigenvector/Modal derivatives** The idea is to use the right eigenvectors  $\{\mathbf{v}_{i,\text{eq}}\}_{i=1}^r$  of the *linearized* system  $(\mathbf{A}_{\text{eq}}, \mathbf{E})$  for  $\mathbf{V}^{(1)}$ , and the so-called *eigenvector derivatives*  $\{\boldsymbol{\theta}_{ij}\}_{i,j=1}^r$  for  $\mathbf{V}^{(2)}$ . The latter are defined as  $\boldsymbol{\theta}_{ij} = \partial \mathbf{v}_{i,\text{eq}} / \partial x_{\text{r},j}$  and describe the change of the eigenvector  $\mathbf{v}_{i,\text{eq}}$  with respect to the amplitude  $x_{\text{r},j}$  of  $\mathbf{v}_{j,\text{eq}}$ . Similarly, the *left* eigenvectors  $\{\mathbf{w}_{i,\text{eq}}\}_{i=1}^r$  of the *dual* linearized system  $(\mathbf{A}_{\text{eq}}^T, \mathbf{E}^T)$  can be augmented with the *left* eigenvector derivatives  $\partial \mathbf{w}_{i,\text{eq}} / \partial x_{\text{r},j}^d$ . Note that the eigenvector derivatives<sup>2</sup> represent the state-space counterpart of the so-called *modal derivatives* employed in nonlinear structural dynamics [129, 240, 224]. Modal derivatives will be further treated in Chapters 9 and 10.

**IRKA + POD** The idea is to combine the linear reduction approach IRKA with the nonlinear technique of Proper Orthogonal Decomposition (POD). First, IRKA is applied to the linearized system  $(\mathbf{A}_{\text{eq}}, \mathbf{B}_{\text{eq}}, \mathbf{C}_{\text{eq}}, \mathbf{E})$ , in order to obtain the bases  $\mathbf{V}^{(1)}$  and  $\mathbf{W}^{(1)}$ . Secondly, bases  $\mathbf{V}_{\text{POD}}$  and  $\mathbf{W}_{\text{POD}}$  are obtained from snapshots. Finally, the respective matrices are concatenated yielding augmented bases. [56, 156]

In general, the augmentation approach has the disadvantage that the bases can rapidly grow, thus increasing the reduced order and consequently inhibiting the speed-up gained through model reduction. Nevertheless, the growth of the bases can be alleviated with a deflation technique to capture the most relevant subspaces and reduce the dimension to  $r_{\text{def}}$ .

<sup>2</sup>For more details on eigenvector (and Krylov) derivatives for *state-space systems* the reader is referred to the master thesis [Him18], where numerical results are also presented. A recent related paper is [177].



### Trajectory piecewise linear approximation (TPWL)

The conceptual idea of Trajectory Piecewise Linear (TPWL) [214, 225] is to first linearize the nonlinear system at different operating points, where linear reduction bases are computed. After that, the nonlinear reduced model is obtained by a weighted sum of linearized ROMs.

The linearization points  $\{(\bar{\mathbf{x}}_s, \bar{\mathbf{u}}_s)\}_{s=1}^S$  are selected — according to the distance to each other  $\|\mathbf{x} - \bar{\mathbf{x}}_s\| \geq \delta$  — along one or several FOM state trajectories simulated for certain training input signals  $\mathbf{u}_{\text{train}}(t)$ . Then, reduction bases  $\{\mathbf{V}_s, \mathbf{W}_s\}_{s=1}^S$  are computed for every linearization point using the linearized matrices  $\{\mathbf{A}_s, \mathbf{B}_s, \mathbf{C}_s\}_{s=1}^S$  and any desired linear MOR technique. Generally, these local bases are then concatenated to obtain respective *global bases*  $\mathbf{V}, \mathbf{W}$  for projection. Note, however, that the local bases could also be applied individually, for which state transformations to a common subspace would be required prior to the weighted interpolation (see e.g. [160], [Gri16]). For the global case, the reduced-order model reads

$$\mathbf{W}^\top \mathbf{E} \mathbf{V} \dot{\mathbf{x}}_r(t) = \sum_{s=1}^S \omega_s(\mathbf{x}_r(t)) \mathbf{W}^\top \left( \mathbf{f}(\bar{\mathbf{x}}_s, \bar{\mathbf{u}}_s) + \mathbf{A}_s(\mathbf{V} \mathbf{x}_r(t) - \bar{\mathbf{x}}_s) + \mathbf{B}_s(\mathbf{u}(t) - \bar{\mathbf{u}}_s) \right), \quad (6.14)$$

where the weights are updated in the online phase depending on the current state  $\mathbf{x}_r(t)$  and fulfill  $\sum_{s=1}^S \omega_s(\mathbf{x}_r(t)) = 1$  with  $\omega_s(\mathbf{x}_r(t)) \geq 0$ .

The method of TPWL can handle strong nonlinearities and requires only linear MOR techniques. Moreover, the process of hyper-reduction is already accomplished due to the approximation of the nonlinear function by a weighted combination of linearized ROMs. On the other hand, the quality of TPWL depends on the choice of many degrees of freedom (e.g. number  $S$ , tolerance  $\delta$  of linearization points, weights), which are often selected heuristically. Furthermore, the FOM has to be simulated for one or several training input signals, making the quality of the ROM also dependent on the selected training conditions.

### Proper orthogonal decomposition (POD)

The Proper Orthogonal Decomposition (POD) [181, 37], aka. Karhunen-Loève transform (KLT) or Principal Component Analysis (PCA), aims at finding a basis that optimally approximates the space spanned by an arbitrary set of data points. In POD, the data points are called *snapshots* and represent discrete samples  $\{\mathbf{x}(t_k)\}_{k=1}^{n_s}$  of the FOM state trajectory. The goal is then to find an orthonormal basis  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r] \in \mathbb{R}^{n \times r}$  such that [153, 74]:

$$\min_{\mathbf{V}} \sum_{k=1}^{n_s} \|\mathbf{x}(t_k) - \mathbf{V} \mathbf{x}_r(t_k)\|_2^2 = \min_{\mathbf{v}_i} \sum_{k=1}^{n_s} \|\mathbf{x}(t_k) - \sum_{i=1}^r \mathbf{v}_i \underbrace{\mathbf{v}_i^\top \mathbf{x}(t_k)}_{\mathbf{x}_{r,i}(t_k)}\|_2^2, \quad \text{s.t. } \mathbf{v}_i^\top \mathbf{v}_j = \delta_{ij}. \quad (6.15)$$

Herein, the coefficient vector  $\mathbf{x}_r(t_k) \in \mathbb{R}^r$  is also adapted to minimize the above expression. The solution to this least-squares minimization problem can be obtained by the singular value decomposition (SVD) of the *snapshot matrix*  $\mathbf{X} = [\mathbf{x}(t_1), \dots, \mathbf{x}(t_{n_s})] \in \mathbb{R}^{n \times n_s}$ , in which all  $n_s$  sampled observations  $\mathbf{x}(t_k)$  are gathered, yielding:

$$\mathbf{X} = \mathbf{U} \mathbf{\Sigma} \mathbf{T}^\top. \quad (6.16)$$

The orthogonal matrices  $\mathbf{U} \in \mathbb{R}^{n \times n}$  and  $\mathbf{T} \in \mathbb{R}^{n_s \times n_s}$  contain the left and right singular vectors, respectively. The diagonal matrix  $\mathbf{\Sigma} = \text{diag}(\varsigma_1, \dots, \varsigma_{n_s}) \in \mathbb{R}^{n \times n_s}$ , where  $\varsigma_1 \geq \dots \geq \varsigma_{n_s} \geq 0$ ,

features all singular values sorted in descending order. The reduction basis  $\mathbf{V} \in \mathbb{R}^{n \times r}$  is finally constructed by taking the first  $r$  left singular vectors  $\{\mathbf{u}_i\}_{i=1}^r$  corresponding to the largest singular values  $\varsigma_i$ .

Note that the dimensions for the SVD matrices are given for the most general case with an  $n$ -by- $n_s$  snapshot matrix. If one employs the economy version of the SVD (e.g. `svd(X, 'econ')` in MATLAB), then the dimensions differ depending on the case. If  $n > n_s$ , where the number of snapshots  $n_s$  is smaller than the number of degrees of freedom  $n$ , then  $\mathbf{U} \in \mathbb{R}^{n \times n_s}$ ,  $\mathbf{\Sigma} \in \mathbb{R}^{n_s \times n_s}$ ,  $\mathbf{T} \in \mathbb{R}^{n_s \times n_s}$ . This is the case for most MOR settings, where models with very large dimension are considered and the snapshot matrix  $\mathbf{X}$  is consequently tall-and-skinny. If  $n < n_s$ , the dimensions of the matrices are  $\mathbf{U} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{T} \in \mathbb{R}^{n_s \times n}$ . Further note that in case  $n \gg n_s$ , it is more efficient to perform the eigenvalue decomposition (EVD) of the correlation matrix  $\mathbf{C} = \mathbf{X}^T \mathbf{X} \in \mathbb{R}^{n_s \times n_s}$  than the SVD of  $\mathbf{X} \in \mathbb{R}^{n \times n_s}$  [191, Suk16].

POD is a straightforward data-driven approach that can be applied for the reduction of any nonlinear system. For this reason it is the most employed and well-known nonlinear reduction technique. Furthermore, an appropriate reduced order  $r$  can be chosen according to the decay of the singular values, e.g. such that [154, 191, Suk16]

$$I(r) = \frac{\sum_{i=1}^r \varsigma_i^2}{\sum_{i=1}^{n_s} \varsigma_i^2} \geq 1 - \varepsilon_{\text{POD}}^2 \quad (6.17)$$

for a given tolerance  $\varepsilon_{\text{POD}}$ . Once  $r$  is fixed, the minimum 2-norm error is given by [161]

$$\sum_{k=1}^{n_s} \left\| \mathbf{x}(t_k) - \sum_{i=1}^r \mathbf{v}_i \underbrace{\mathbf{v}_i^T \mathbf{x}(t_k)}_{x_{r,i}(t_k)} \right\|_2^2 = \sum_{i=r+1}^{n_s} \varsigma_i^2. \quad (6.18)$$

Finally, note that the method provides an optimal low-rank approximation  $\mathbf{X}_r = \sum_{i=1}^r \varsigma_i \mathbf{u}_i \mathbf{t}_i^T$  with rank  $r$  of the snapshot matrix  $\mathbf{X}$ .

In addition to all mentioned advantages, POD also has some drawbacks that are discussed in the following. First of all, the method relies on expensive simulations of the full-order model for several training input signals  $\mathbf{u}_{\text{train}}(t)$ , in order to gather representative snapshots. This leads to immense offline costs, that are even higher, if the model also has to be simulated for different parameter sets and/or boundary conditions. The SVD of the gathered snapshot matrix/matrices also contributes to the offline costs, but to a lesser extent than the FOM simulations. Another drawback of POD is that the approximation quality of the ROM crucially depends on the selected snapshots and training scenarios (aka. training input dependency). If the latter do not capture the dynamics for all relevant operating conditions of the system, then the performance might deteriorate for scenarios that were not trained.

To conclude this section, please note that there exist other simulation-based nonlinear reduction techniques that are similar or related to POD. Examples are balanced-POD [274], empirical Gramians [164] and Reduced Basis methods [211, 117]. For transport phenomena, e.g. advection-dominated systems, a very slow decay of the singular values is usually observed with the classical POD. Thus, the *shifted* POD (sPOD) has been proposed [213] to handle such kind of problems. Besides the common (Petrov-)Galerkin projection, POD has widely been employed within the (spatial) least-squares Petrov-Galerkin (LSPG) framework [55, 50, 54], and recently also within the space-time LSPG method [57] using higher-order singular value decomposition (HOSVD). Finally, reduction techniques based on neural networks or deep/manifold learning [199, 260, 159] are also data-driven, and thus related to POD.

## 6.4 Nonlinear reduction based on nonlinear projection

After having focused on the linear projection-based reduction of nonlinear systems in Section 6.3, we now direct our attention to model reduction based on *nonlinear projection*. Similarly as before, we present the corresponding projection framework, time integration and reduction techniques.

### 6.4.1 Nonlinear Petrov-Galerkin projection framework

One promising, and not yet perfectly well developed, way of reducing nonlinear systems is to apply a *nonlinear* Petrov-Galerkin projection, where the approximation ansatz is given by

$$\mathbf{x}(t) = \boldsymbol{\nu}(\mathbf{x}_r(t)) + \mathbf{e}(t), \quad (6.19)$$

with the smooth nonlinear mapping  $\boldsymbol{\nu}(\mathbf{x}_r) : \mathbb{R}^r \rightarrow \mathbb{R}^n$  and the approximation error  $\mathbf{e}(t) \in \mathbb{R}^n$ . In contrast to the linear ansatz (3.26), the state vector  $\mathbf{x}(t) \in \mathbb{R}^n$  is now expressed as a nonlinear function of the reduced state vector  $\mathbf{x}_r(t) \in \mathbb{R}^r$ . The derivative of (6.19) reads

$$\dot{\mathbf{x}}(t) = \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r(t))}{\partial \mathbf{x}_r(t)} \dot{\mathbf{x}}_r(t) + \dot{\mathbf{e}}(t), \quad (6.20)$$

with the Jacobian matrix

$$\widetilde{\mathbf{V}}(\mathbf{x}_r) = \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r)}{\partial \mathbf{x}_r} \in \mathbb{R}^{n \times r}. \quad (6.21)$$

The tangential Jacobian  $\widetilde{\mathbf{V}}_{\mathbf{x}_r}$  spans the  $r$ -dimensional tangent space  $\widetilde{\mathcal{V}} = \mathcal{T}_{\mathbf{x}_r} \mathcal{M} = \text{ran}(\widetilde{\mathbf{V}}_{\mathbf{x}_r})$  of the manifold  $\mathcal{M} = \{\mathbf{x}_r \in \mathbb{R}^r : \mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r)\}$ .

Inserting the ansatz and its derivative in (6.1a) yields an overdetermined system of equations with the residual  $\boldsymbol{\varepsilon}(t) \in \mathbb{R}^n$ . To obtain a square ROM, the resulting system is projected onto the subspace  $\widetilde{\mathcal{U}} = \text{ran}(\mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r})$ . The projection is performed orthogonally to another subspace  $\widetilde{\mathcal{W}} = \text{ran}(\mathbf{W}_{\mathbf{x}_r})$ , or in other words, along the orthogonal complement  $\widetilde{\mathcal{W}}^\perp = \text{ran}(\mathbf{W}_{\mathbf{x}_r})^\perp$  with the Jacobian matrix [109]

$$\widetilde{\mathbf{W}}(\mathbf{x}_r)^\top = \left. \frac{\partial \boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}, \mathbf{u}))}{\partial \mathbf{x}} \right|_{\mathbf{x}=\boldsymbol{\nu}(\mathbf{x}_r)} \in \mathbb{R}^{r \times n}. \quad (6.22)$$

The tangential Jacobian  $\widetilde{\mathbf{W}}_{\mathbf{x}_r} \in \mathbb{R}^{n \times r}$  spans the  $r$ -dimensional subspace  $\widetilde{\mathcal{W}} = \text{ran}(\mathbf{W}_{\mathbf{x}_r})$ . Its orthogonal complement is spanned by the tangent space  $\widetilde{\mathcal{W}}^\perp = \mathcal{T}_{\mathbf{x}_r} \mathcal{N} = \text{ran}(\widetilde{\mathbf{W}}_{\mathbf{x}_r})^\perp$  of the manifold  $\mathcal{N} = \{\boldsymbol{\xi}(\mathbf{x}_r, \mathbf{u}) \in \mathbb{R}^n : \boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}, \mathbf{u}))|_{\mathbf{x}=\boldsymbol{\nu}(\mathbf{x}_r)} = \mathbf{0}\}$  with the mapping  $\boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}_r, \mathbf{u})) : \mathbb{R}^n \rightarrow \mathbb{R}^r$  and the residual function  $\boldsymbol{\xi}(\mathbf{x}, \mathbf{u}) = \mathbf{E}\dot{\mathbf{x}} - \mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{0}$ . Multiplying the overdetermined system from the left with the projector  $\widetilde{\boldsymbol{\Pi}} = \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r} (\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r})^{-1} \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top$ , where  $\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r}$  is non-singular, leads to

$$\widetilde{\boldsymbol{\Pi}} \left( \underbrace{\mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r} \dot{\mathbf{x}}_r(t) - \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t))}_{=\boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t))} - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0} \Leftrightarrow \widetilde{\boldsymbol{\Pi}} \left( \boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t)) - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0}. \quad (6.23)$$

Enforcing the Petrov-Galerkin condition  $\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\widetilde{\boldsymbol{\Pi}} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , the residual then vanishes and only the term  $\widetilde{\boldsymbol{\Pi}} \boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t)) = \mathbf{0}$  remains. This yields the ROM

$$\widetilde{\mathbf{E}}_r \dot{\mathbf{x}}_r(t) = \left. \frac{\partial \boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}(t), \mathbf{u}(t)))}{\partial \mathbf{x}(t)} \right|_{\mathbf{x}(t)=\boldsymbol{\nu}(\mathbf{x}_r(t))} \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t)), \quad \mathbf{x}_r(0) = \mathbf{x}_{r,0}, \quad (6.24a)$$

$$\mathbf{y}_r(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r(t))), \quad (6.24b)$$

with  $\widetilde{\mathbf{E}}_r = \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r}$ , the initial condition  $\mathbf{x}_r(0) = \arg \min_{\mathbf{x}_{r,0}} \|\boldsymbol{\nu}(\mathbf{x}_{r,0}) - \mathbf{x}_0\|_2^2$  and the reduced nonlinear mapping  $\widetilde{\mathbf{f}}_r(\mathbf{x}_r(t), \mathbf{u}(t)) = \widetilde{\mathbf{W}}_{\mathbf{x}_r}(\mathbf{x}_r(t))^\top \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{u}(t))$  with  $\widetilde{\mathbf{f}}_r(\mathbf{x}_r, \mathbf{u}) : \mathbb{R}^r \times \mathbb{R}^m \rightarrow \mathbb{R}^r$ .

Please note that for the computation of the initial condition  $\mathbf{x}_r(0)$  a nonlinear least-squares problem should be solved [138] using, for instance, the gradient descent method, the trust region method, Gauss-Newton or the LevenbergMarquardt algorithm (cf. e.g. `lsqnonlin` command in MATLAB) [193]. Further note that the reduced nonlinear mapping  $\widetilde{\mathbf{f}}_r(\mathbf{x}_r, \mathbf{u})$  has to be simplified, in order to gain further speed-ups. This procedure is known under the name of *hyper-reduction*, which will be treated in Section 6.5.

**Remark 6.1** (Link between nonlinear and linear projection). The afore explained nonlinear projection framework is a generalization from the linear case. Thus, the nonlinear reduction mappings  $\boldsymbol{\nu}(\mathbf{x}_r) : \mathbb{R}^r \rightarrow \mathbb{R}^n$  and  $\boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}_r, \mathbf{u})) : \mathbb{R}^n \rightarrow \mathbb{R}^r$  are linked to the linear case:

$$\mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r) \hat{=} \widetilde{\mathbf{V}}_{\mathbf{x}_r} \mathbf{x}_r, \quad \widetilde{\mathbf{V}}_{\mathbf{x}_r} \hat{=} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r)}{\partial \mathbf{x}_r}, \quad (6.25a)$$

$$\boldsymbol{\omega}(\boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{x}_r), \mathbf{u})) \hat{=} \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{x}_r), \mathbf{u}) = \mathbf{0}, \quad \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \hat{=} \left. \frac{\partial \boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}, \mathbf{u}))}{\partial \mathbf{x}} \right|_{\mathbf{x}=\boldsymbol{\nu}(\mathbf{x}_r)}. \quad (6.25b)$$

Moreover, the projection mappings  $\mathbf{x}_{\text{proj}}(t) = \mathbf{u}(\mathbf{c}(t))$  with  $\mathbf{u}(\mathbf{c}) : \mathbb{R}^r \rightarrow \mathbb{R}^n$ ,  $\mathbf{c}(t) = \mathbf{w}(\mathbf{x}(t))$  with  $\mathbf{w}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^r$  and  $\mathbf{x}_{\text{proj}}(t) = \boldsymbol{\varrho}(\mathbf{x}(t)) = \mathbf{u}(\mathbf{w}(\mathbf{x}(t)))$  with  $\boldsymbol{\varrho}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$  are strongly related to their linear counterparts:

$$\mathbf{x}_{\text{proj}} = \mathbf{u}(\mathbf{c}) \hat{=} \widetilde{\mathbf{U}} \mathbf{c}, \quad \widetilde{\mathbf{U}} \hat{=} \frac{\partial \mathbf{u}(\mathbf{c})}{\partial \mathbf{c}}, \quad (6.26a)$$

$$\mathbf{c} = \mathbf{w}(\mathbf{x}) \hat{=} \underbrace{(\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \widetilde{\mathbf{U}})^{-1} \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top}_{*} \mathbf{x}, \quad * \hat{=} \frac{\partial \mathbf{w}(\mathbf{x})}{\partial \mathbf{x}}, \quad (6.26b)$$

$$\boldsymbol{\varrho}(\mathbf{x}) = \mathbf{u}(\mathbf{w}(\mathbf{x})) \hat{=} \widetilde{\boldsymbol{\Pi}} \mathbf{x}, \quad \widetilde{\boldsymbol{\Pi}} \hat{=} \frac{\partial \mathbf{u}(\mathbf{c})}{\partial \mathbf{c}} \frac{\partial \mathbf{w}(\mathbf{x})}{\partial \mathbf{x}}, \quad (6.26c)$$

where  $\widetilde{\mathbf{U}} = \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r}$  and the projector is given by

$$\widetilde{\boldsymbol{\Pi}} = \mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r)}{\partial \mathbf{x}_r} \left( \left. \frac{\partial \boldsymbol{\omega}(\cdot)}{\partial \mathbf{x}} \right|_{\boldsymbol{\nu}(\mathbf{x}_r)} \mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r)}{\partial \mathbf{x}_r} \right)^{-1} \left. \frac{\partial \boldsymbol{\omega}(\cdot)}{\partial \mathbf{x}} \right|_{\boldsymbol{\nu}(\mathbf{x}_r)} = \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r} (\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r})^{-1} \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top. \quad (6.27)$$

For nonlinear systems, a Galerkin projection ( $\widetilde{\mathbf{W}}_{\mathbf{x}_r} = \widetilde{\mathbf{V}}_{\mathbf{x}_r}$ ) is commonly used. For the special case  $\mathbf{E} = \mathbf{I}$  this choice yields  $* = (\widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{x}_r})^{-1} \widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top$  and  $\widetilde{\boldsymbol{\Pi}} = \widetilde{\mathbf{V}}_{\mathbf{x}_r} (\widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{x}_r})^{-1} \widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top$ , or  $* = \widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top$  and  $\widetilde{\boldsymbol{\Pi}} = \widetilde{\mathbf{V}}_{\mathbf{x}_r} \widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top$  if  $\widetilde{\mathbf{V}}_{\mathbf{x}_r}$  is orthogonal ( $\widetilde{\mathbf{V}}_{\mathbf{x}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{x}_r} = \mathbf{I}_r$ ). Note that the condition  $\mathbf{w}(\mathbf{E} \boldsymbol{\nu}(\mathbf{x}_r)) = \mathbf{x}_r$  corresponds to  $(\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r})^{-1} \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r} \mathbf{x}_r = \mathbf{x}_r$ .  $\triangle$

**Remark 6.2** (Power series ansatz [146, 124]). The power series approximation ansatz

$$\mathbf{x} = \sum_{k=1}^N \mathbf{V}^{(k)} \mathbf{x}_r^{(k)} = \mathbf{V}^{(1)} \mathbf{x}_r^{(1)} + \mathbf{V}^{(2)} \mathbf{x}_r^{(2)} + \dots \quad (6.28)$$

with  $\mathbf{V}^{(k)} \in \mathbb{R}^{n \times r^k}$  and  $\mathbf{x}_r^{(k)} = \overbrace{\mathbf{x}_r \otimes \dots \otimes \mathbf{x}_r}^{k \text{ times}} \in \mathbb{R}^{r^k}$  constitutes a special case of the most general nonlinear projection  $\mathbf{x} \approx \boldsymbol{\nu}(\mathbf{x}_r)$ . Depending on the underlying nonlinearities, the power series ansatz could be customized for the system at hand. For instance, the following *quadratic manifold* projection ansatz

$$\mathbf{x} = \mathbf{V}^{(1)} \mathbf{x}_r + \mathbf{V}^{(2)}(\mathbf{x}_r \otimes \mathbf{x}_r) \hat{=} \boldsymbol{\nu}(\mathbf{x}_r) \quad (6.29a)$$

$$\dot{\mathbf{x}} = \mathbf{V}^{(1)} \dot{\mathbf{x}}_r + \mathbf{V}^{(2)}(\dot{\mathbf{x}}_r \otimes \mathbf{x}_r + \mathbf{x}_r \otimes \dot{\mathbf{x}}_r) \quad (6.29b)$$

could be used for the reduction of a polynomial (e.g. quadratic) system

$$\begin{aligned} \mathbf{E} \dot{\mathbf{x}}(t) &= \mathbf{A}_1 \mathbf{x}(t) + \mathbf{A}_2 (\mathbf{x}(t) \otimes \mathbf{x}(t)) + \mathbf{B} \mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C} \mathbf{x}(t), \end{aligned} \quad (6.30)$$

yielding the ROM (6.24) with the corresponding  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  and  $\mathbf{h}(\mathbf{x})$  from above, together with  $\widetilde{\mathbf{V}}(\mathbf{x}_r) = \partial \boldsymbol{\nu}(\mathbf{x}_r) / \partial \mathbf{x}_r = \mathbf{V}^{(1)} + \mathbf{V}^{(2)}(\mathbf{1}_r \otimes \mathbf{x}_r + \mathbf{x}_r \otimes \mathbf{1}_r)$ , where  $\mathbf{1}_r^\top = [1, \dots, 1] \in \mathbb{R}^{1 \times r}$ . A similar example is considered in Eq. (6.39) and [109, Sec. 4.7.1], where  $\widetilde{\mathbf{W}}(\mathbf{x}_r)^\top = \mathbf{W}^\top$  is used.  $\triangle$

## 6.4.2 Time integration

Before we explain some approaches to construct the reduction manifolds  $\boldsymbol{\nu}(\mathbf{x}_r)$  and  $\boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}_r, \mathbf{u}))$  in Section 6.4.3, we first want to address the time integration of the ROM (6.24).

The application of a nonlinear projection yields a rather complicated reduced model with state-dependent bases  $\widetilde{\mathbf{V}}_{\mathbf{x}_r}$  and  $\widetilde{\mathbf{W}}_{\mathbf{x}_r}$ . Thus, for certain states  $\mathbf{x}_r(t)$  and  $\mathbf{x}(t) \approx \boldsymbol{\nu}(\mathbf{x}_r(t))$  the bases might get rank deficient or the matrix  $\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r}$  become badly conditioned. Furthermore, the time integration scheme becomes more involved due to the changing bases. This is illustrated in the following for the case of an implicit Euler method.

Applying the backward Euler formula (6.4) to the ROM state equation (6.24a) yields

$$\widetilde{\mathbf{E}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k+1} = \widetilde{\mathbf{E}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k} + h \cdot \widetilde{\mathbf{f}}_{\mathbf{x}_r, k+1}(\mathbf{x}_{r, k+1}, \mathbf{u}_{k+1}), \quad \text{with } t_{k+1} = t_k + h. \quad (6.31)$$

The reduced descriptor matrix at the time-step  $k+1$  is given by  $\widetilde{\mathbf{E}}_{\mathbf{x}_r, k+1} = \widetilde{\mathbf{W}}_{\mathbf{x}_r, k+1}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r, k+1}$ . The reduced nonlinear mapping reads  $\widetilde{\mathbf{f}}_{\mathbf{x}_r, k+1}(\mathbf{x}_{r, k+1}, \mathbf{u}_{k+1}) = \widetilde{\mathbf{W}}_{\mathbf{x}_r, k+1}^\top \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_{r, k+1}), \mathbf{u}_{k+1})$  with  $\mathbf{x}_{k+1} \approx \boldsymbol{\nu}(\mathbf{x}_{r, k+1})$ . This finally leads to the residual

$$\begin{aligned} \mathbf{res}(\mathbf{x}_{r, k+1}) &= \widetilde{\mathbf{E}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k} - \widetilde{\mathbf{E}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k+1} + h \cdot \widetilde{\mathbf{f}}_{\mathbf{x}_r, k+1}(\mathbf{x}_{r, k+1}, \mathbf{u}_{k+1}) \\ &= \widetilde{\mathbf{W}}_{\mathbf{x}_r, k+1}^\top \left( \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k} - \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r, k+1} \mathbf{x}_{r, k+1} + h \cdot \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_{r, k+1}), \mathbf{u}_{k+1}) \right) \\ &= \widetilde{\mathbf{W}}_{\mathbf{x}_r, k+1}^\top \cdot \mathbf{res}_{\text{full}}(\mathbf{x}_{r, k+1}). \end{aligned} \quad (6.32)$$

The Jacobian of the residual

$$\mathbf{J}_{res}^{iter}(\mathbf{x}_{r,k+1}^{iter}) = \left. \frac{\partial res(\mathbf{x}_{r,k+1})}{\partial \mathbf{x}_{r,k+1}} \right|_{\mathbf{x}_{r,k+1} = \mathbf{x}_{r,k+1}^{iter}} \quad (6.33)$$

is required for the Newton-Raphson loop. An analytical expression for  $\mathbf{J}_{res}^{iter}(\mathbf{x}_{r,k+1}^{iter})$  can be obtained by applying the product rule to (6.32). This leads to several terms due to the dependence of  $\widetilde{\mathbf{W}}_{\mathbf{x}_{r,k+1}}^T$  and  $res_{full}$  on the reduced state  $\mathbf{x}_{r,k+1}$ . One of the terms reads

$$h \cdot \widetilde{\mathbf{W}}_{\mathbf{x}_{r,k+1}}^T \mathbf{A}(\boldsymbol{\nu}(\mathbf{x}_{r,k+1}), \mathbf{u}_{k+1}) \widetilde{\mathbf{V}}_{\mathbf{x}_{r,k+1}} \quad (6.34)$$

and depends on the Jacobian  $\mathbf{A}(\mathbf{x}, \mathbf{u})$  of the nonlinear function  $\mathbf{f}(\mathbf{x}, \mathbf{u})$ . The analytical Jacobian (6.33) is composed of many other terms, which are not given explicitly here.

Note that the exact analytical Jacobian may be approximated by neglecting some of the difficult terms arising. Although the quadratic convergence of the Newton-Raphson method cannot be ensured anymore, such an approximate Jacobian can result in a more stable and efficient time integration scheme (cf. [132], 22). Another possibility consists in calculating the Jacobian numerically via finite differences, which is however much more time-consuming.

### 6.4.3 Nonlinear projection-based reduction approaches

In this section, a brief overview of several approaches to construct nonlinear reduction mappings  $\boldsymbol{\nu}(\mathbf{x}_r)$  and  $\boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}_r, \mathbf{u}))$  is given. The focus is laid in explaining the main idea of the different methods rather than in giving all mathematical details.

#### Quadratic manifold

As mentioned in Remark 6.2, a power series expansion of the nonlinear projection ansatz  $\mathbf{x}(t) \approx \boldsymbol{\nu}(\mathbf{x}_r(t))$  may be a simpler approach than employing the most general mapping  $\boldsymbol{\nu}(\mathbf{x}_r)$ . For instance, truncating the series after the second-order term yields the quadratic manifold

$$\mathbf{x}(t) \approx \mathbf{V}^{(1)} \mathbf{x}_r(t) + \mathbf{V}^{(2)} (\mathbf{x}_r(t) \otimes \mathbf{x}_r(t)), \quad (6.35)$$

with the reduced coordinates  $\mathbf{x}_r(t) = [x_{r,1}(t), \dots, x_{r,r}(t)]^T$ . The question is now how to compute the reduction matrices  $\mathbf{V}^{(1)} \in \mathbb{R}^{n \times r}$  and  $\mathbf{V}^{(2)} \in \mathbb{R}^{n \times r^2}$  that parametrize this manifold.

One possibility consists in using the right eigenvectors  $\{\mathbf{v}_{i,eq}\}_{i=1}^r$  of the linearized system  $(\mathbf{A}_{eq}, \mathbf{E})$  for  $\mathbf{V}^{(1)}$ , and the eigenvector derivatives  $\{\boldsymbol{\theta}_{ij}\}_{i,j=1}^r$  with  $\boldsymbol{\theta}_{ij} = \partial \mathbf{v}_{i,eq} / \partial x_{r,j}$  for  $\mathbf{V}^{(2)}$ . These *perturbation* derivatives capture the quadratic behavior of the series and are therefore conceivable not only for basis augmentation (cf. Eq. (6.13)), but also for the ansatz (6.35). Note that quadratic manifold reduction based on modes and modal derivatives has been successfully applied in the context of nonlinear structural dynamics [132, 223, 224]. To the best of the author's knowledge such an approach has not been applied for state-space systems so far. Thus, our idea of using eigenvector (or Krylov) derivatives for manifold reduction of nonlinear *first-order* systems seems to be novel and has potential for application.

Another approach is to construct the matrices  $\mathbf{V}^{(1)}$  and  $\mathbf{V}^{(2)}$  as bases of input Krylov subspaces for the first and second subsystem of the Volterra series representation (cf. Sec-

tion 5.3). Our idea is to use these matrices in a quadratic manifold setting (cf. (6.35)), rather than concatenating them to an augmented basis as it has been done so far in polynomial MOR. Note that this approach is only proposed here as a conceptual idea. Its implementation and validation is subject to future research.

Regarding the mapping  $\boldsymbol{\omega}(\cdot)$ , a quadratic manifold ansatz with matrices  $\mathbf{W}^{(1)} \in \mathbb{R}^{n \times r}$  and  $\mathbf{W}^{(2)} \in \mathbb{R}^{n \times r^2}$  — or rather  $\mathbf{W}^{(1)\top} \in \mathbb{R}^{r \times n}$  and  $\mathbf{W}^{(2)\top} \in \mathbb{R}^{r^2 \times n}$  — could be applied as well. These matrices can be computed by employing the aforementioned approaches in a dual manner. This means: the *left* eigenvectors  $\{\mathbf{w}_{i,\text{eq}}\}_{i=1}^r$  of the *dual* linearized system  $(\mathbf{A}_{\text{eq}}^\top, \mathbf{E}^\top)$  are used for  $\mathbf{W}^{(1)}$ , and the *left* eigenvector derivatives  $\partial \mathbf{w}_{i,\text{eq}} / \partial x_{r,j}^d$  are utilized for  $\mathbf{W}^{(2)}$ . Moreover, the bases of *output* Krylov subspaces for the first and second subsystem of the Volterra series can be employed in a quadratic manifold setting.

### ManiMOR approach

One of the methods proposed in [109, Sec. 4.3, 4.6] consists first in computing local Krylov subspaces around one or several equilibrium points  $\{(\mathbf{x}_{\text{eq},s}, \mathbf{u}_{\text{eq},s})\}_{s=1}^S$ , and then in integrating along the Krylov vectors to obtain the manifold  $\boldsymbol{\nu}(\mathbf{x}_r)$ . In the following, we explain (and partly generalize) this method using our own notation.

First, an equilibrium point-dependent basis  $\mathbf{V}(\mathbf{x}_{\text{eq}})$  is computed using the Krylov subspace  $\mathcal{K}_q(\mathbf{A}_{\text{eq},\sigma}^{-1} \mathbf{E}, \mathbf{A}_{\text{eq},\sigma}^{-1} \mathbf{B}_{\text{eq}})$ , where  $\mathbf{A}_{\text{eq},\sigma} := (\sigma \mathbf{E} - \mathbf{A}_{\text{eq}})$ . Naturally, the basis could also be constructed by the union of several Krylov subspaces computed for different equilibrium points (and also shifts and tangential directions), e.g. as

$$\mathbf{V}(\mathbf{x}_{\text{eq},s}) = \left[ \mathbf{A}_{\text{eq},1,\sigma_1}^{-1} \mathbf{B}_{\text{eq},1} \mathbf{r}_1, \mathbf{A}_{\text{eq},1,\sigma_1}^{-1} \mathbf{E} \mathbf{A}_{\text{eq},1,\sigma_1}^{-1} \mathbf{B}_{\text{eq},1} \mathbf{r}_1, \dots, \mathbf{A}_{\text{eq},S,\sigma_S}^{-1} \mathbf{B}_{\text{eq},S} \mathbf{r}_S, \dots \right]. \quad (6.36)$$

Then, the manifold  $\boldsymbol{\nu}(\mathbf{x}_r)$  is computed by integrating

$$\frac{\partial \mathbf{x}_{\text{eq},s}}{\partial \mathbf{x}_r} = \mathbf{V}(\mathbf{x}_{\text{eq},s}), \quad \text{or rather} \quad \frac{\partial \mathbf{x}_{\text{eq},s}}{\partial x_{r,i}} = \mathbf{v}_i(\mathbf{x}_{\text{eq},s}), \quad i = 1, 2, \dots, \quad (6.37)$$

for each Krylov vector. This yields the *exponential* manifold  $\mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r) = \exp(\mathbf{V}(\mathbf{x}_{\text{eq},s}) \mathbf{x}_r)$ . The integral manifold methods from Gu can be combined with a hyper-reduction technique (cf. Section 6.5). In fact, the so-called ManiMOR approach [109, Sec. 5] combines an integral manifold with a PWL approximation of the nonlinear function (cf. Eq. (6.40)).

Note that the Krylov vector  $\mathbf{v}(\mathbf{x}_{\text{eq}}) = -\mathbf{A}_{\text{eq}}^{-1} \mathbf{B}_{\text{eq}}$  for shift  $\sigma = 0$  spans the tangent space of a so-called *DC manifold*, which is named after the close connection of  $\mathbf{v}(\mathbf{x}_{\text{eq}})$  to the computation of steady-state equilibria for nonlinear systems (cf. Section 7.2.1.3). Further note that the integral manifold method involves the solution of a series of differential equations (6.37), which can be expensive.

### Nonlinear balancing

The method of balanced truncation has been carried over to nonlinear systems in [239] based on reachability and observability considerations, as well as on nonlinear optimal control theory. The generalization leads to two Hamilton-Jacobi-Bellman (HJB) partial differential equations, which represent the nonlinear counterpart of the linear Lyapunov equations (3.18). These partial differential equations (PDEs) are difficult and expensive to solve due to the curse of

dimensionality, especially if large models are involved. Therefore, techniques that avoid the HJB-PDEs and instead yield *state-dependent* Lyapunov equations have been developed. These are (i) *dynamical balancing* based on dynamic controllability and observability extension [226, 227], and (ii) *differential balancing* based on contraction theory and variational systems [147, 149]. Both streams are reviewed and well explained in the master thesis [Hei18], where a numerical approach to solve the state-dependent Lyapunov equations is also discussed. Bringing the system into a nonlinear balanced realization involves (in both dynamical and differential balancing) a *state-dependent* transformation matrix, leading to singular value *functions*  $\varsigma_i(\boldsymbol{x})$ . At this point it is not clear to the author how the methods can be efficiently implemented for large-scale systems without making use of symbolic computations. In fact, the development of a numerically feasible — and at the same time system-theoretic/simulation-free — nonlinear balancing method is still topic of ongoing research in the MOR community.

### Nonlinear moment matching

The method of moment matching has also been transferred to the nonlinear case in [13, 14, 127] based on the steady-state interpretation of input and output moments (cf. Section 3.6). The generalization leads to two Sylvester-like partial differential equations, which represent the nonlinear counterpart of the linear Sylvester equations (3.62). These PDEs must be solved, in order to obtain the reduction mappings  $\boldsymbol{\nu}(\boldsymbol{x}_r)$  and  $\boldsymbol{\omega}(\boldsymbol{\xi}(\boldsymbol{x}_r, \boldsymbol{u}))$  needed for projection. More details concerning input nonlinear moment matching are given in Chapter 7, where some simplifications to approximate the underlying PDE are also proposed.

## 6.5 Hyper-reduction

Nonlinear dimensional reduction techniques reduce the number of equations from  $n$  to the much smaller dimension  $r$ , yielding the ROM (6.9) or (6.24). Due to this drastical reduction of degrees of freedom, the computational cost associated with the solution of linear systems of equations in implicit time integration schemes (cf. line 9) is reduced. However, the calculation of the nonlinear term  $\boldsymbol{f}_r(\boldsymbol{x}_r, \boldsymbol{u})$  in (6.9) — and similarly of  $\tilde{\boldsymbol{f}}_r(\boldsymbol{x}_r, \boldsymbol{u})$  in (6.24) — still involves expensive computations. More specifically, it requires (1) the transformation of the reduced coordinates  $\boldsymbol{x}_r$  to the full ones via  $\boldsymbol{x} \approx \boldsymbol{V}\boldsymbol{x}_r$ , (2) the evaluation of the full nonlinear function  $\boldsymbol{f}(\boldsymbol{V}\boldsymbol{x}_r, \boldsymbol{u})$  and (3) the projection with the reduction basis  $\boldsymbol{W}$  to obtain  $\boldsymbol{f}_r(\boldsymbol{x}_r, \boldsymbol{u}) = \boldsymbol{W}^\top \boldsymbol{f}(\boldsymbol{V}\boldsymbol{x}_r, \boldsymbol{u})$ . In case of an implicit scheme, the Jacobian  $\boldsymbol{A}(\boldsymbol{x}, \boldsymbol{u})$  of the nonlinear function is also required for the Newton-Raphson loop (cf. Eqs. (6.11) and (6.34)). Similar as before, the Jacobian  $\boldsymbol{A}(\boldsymbol{V}\boldsymbol{x}_r, \boldsymbol{u})$  is also first evaluated in high dimension and then projected to obtain the reduced Jacobian  $\boldsymbol{A}_r(\boldsymbol{x}_r, \boldsymbol{u}) = \boldsymbol{W}^\top \boldsymbol{A}(\boldsymbol{V}\boldsymbol{x}_r, \boldsymbol{u})\boldsymbol{V}$ . All these additional computations inhibit the speed-up gained through dimensional reduction, and can make, in small-sized cases, the simulation of the ROM even more expensive than the FOM one.

As a consequence thereof, so-called hyper-reduction techniques have been developed. The main idea is to approximate the nonlinear function and Jacobian, in order to reduce the computational cost associated with the evaluation of these terms and further increase the speed-up. Hyper-reduction methods are usually employed “on top” of dimensional reduction approaches. In this regard, different strategies exist to accomplish the hyper-reduction step. The polynomial representation and the PWL approximation are fundamentally different from the “classical” hyper-reduction methods DEIM and ECSW. In the following, we give a brief overview of hyper-reduction and focus on the conceptual differences between the methods.



### 6.5.1 Polynomial system representation

One possibility consists in representing the nonlinear function  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  as a polynomial system (4.1) using a Taylor series expansion, a polynomialization procedure or a bilinear/quadratic-bilinear representation (cf. Sections 4.1, 4.2). This approach has the advantage that the nonlinearity needs not to be evaluated on the element level within the FE code anymore, since it is expressed analytically by means of polynomial tensors. Although the full-order tensors are generally *sparse*, their storage becomes prohibitive for large  $n$ , especially if high-order polynomials (e.g. cubic, quartic) are considered. Hence, depending on the application and degree of nonlinearity, this approach might be more or less suitable.

Once a polynomial system representation is obtained, the dimensional reduction can be accomplished using either a linear or nonlinear projection. This is illustrated in the following.

**Reduction with linear bases** Let us consider a cubic system representation with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}_1 \mathbf{x} + \mathbf{A}_2(\mathbf{x} \otimes \mathbf{x}) + \mathbf{A}_3(\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) + \mathbf{B}\mathbf{u}$ . Using linear bases  $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{n \times r}$  for projection yields a reduced nonlinear function

$$\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{A}_{1r} \mathbf{x}_r + \mathbf{A}_{2r}(\mathbf{x}_r \otimes \mathbf{x}_r) + \mathbf{A}_{3r}(\mathbf{x}_r \otimes \mathbf{x}_r \otimes \mathbf{x}_r) + \mathbf{B}_r \mathbf{u} \quad (6.38)$$

with (dense!) reduced matrices

$$\mathbf{A}_{1r} = \mathbf{W}^\top \mathbf{A}_1 \mathbf{V}, \quad \mathbf{A}_{2r} = \mathbf{W}^\top \mathbf{A}_2 (\mathbf{V} \otimes \mathbf{V}), \quad \mathbf{A}_{3r} = \mathbf{W}^\top \mathbf{A}_3 (\mathbf{V} \otimes \mathbf{V} \otimes \mathbf{V}), \quad \mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$$

of dimension  $\mathbf{A}_{1r} \in \mathbb{R}^{r \times r}$ ,  $\mathbf{A}_{2r} \in \mathbb{R}^{r \times r^2}$ ,  $\mathbf{A}_{3r} \in \mathbb{R}^{r \times r^3}$  and  $\mathbf{B}_r \in \mathbb{R}^{r \times m}$ .

**Reduction with nonlinear manifold** Let us now consider the quadratic system (6.30) with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}_1 \mathbf{x} + \mathbf{A}_2(\mathbf{x} \otimes \mathbf{x}) + \mathbf{B}\mathbf{u}$ . Using a quadratic manifold ansatz with  $\mathbf{V}^{(1)} \in \mathbb{R}^{n \times r}$ ,  $\mathbf{V}^{(2)} \in \mathbb{R}^{n \times r^2}$  and e.g.  $\widehat{\mathbf{W}}(\mathbf{x}_r)^\top = \mathbf{W}^\top \in \mathbb{R}^{r \times n}$  yields the reduced nonlinear function

$$\begin{aligned} \mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) &= \mathbf{W}^\top \mathbf{A}_1 \mathbf{V}^{(1)} \mathbf{x}_r + \left( \mathbf{W}^\top \mathbf{A}_1 \mathbf{V}^{(2)} + \mathbf{W}^\top \mathbf{A}_2 (\mathbf{V}^{(1)} \otimes \mathbf{V}^{(1)}) \right) (\mathbf{x}_r \otimes \mathbf{x}_r) \\ &\quad + \mathbf{W}^\top \mathbf{A}_2 \left( \mathbf{V}^{(1)} \otimes \mathbf{V}^{(2)} + \mathbf{V}^{(2)} \otimes \mathbf{V}^{(1)} \right) (\mathbf{x}_r \otimes \mathbf{x}_r \otimes \mathbf{x}_r) \\ &\quad + \mathbf{W}^\top \mathbf{A}_2 \left( \mathbf{V}^{(2)} \otimes \mathbf{V}^{(2)} \right) (\mathbf{x}_r \otimes \mathbf{x}_r \otimes \mathbf{x}_r \otimes \mathbf{x}_r) + \mathbf{W}^\top \mathbf{B} \mathbf{u} \end{aligned} \quad (6.39)$$

with (dense!) reduced matrices of corresponding dimension (cf. [109, Sec. 4.7.1]).

The computational speed-up of this approach becomes evident through the smaller costs of evaluating the reduced polynomial function  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u})$  in comparison to the evaluation of  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{W}^\top \mathbf{f}(\mathbf{V} \mathbf{x}_r, \mathbf{u})$ . However, special care has still to be taken with the computation of the reduced tensors  $\mathbf{A}_{2r}$ ,  $\mathbf{A}_{3r}$ , etc., in order to efficiently handle the Kronecker products  $(\mathbf{V} \otimes \mathbf{V})$ , etc. (cf. Alg. 2.1 from [22, Sec. 3.2.] or [103, Alg. 4.4]). Furthermore, it is important to note that the reduced tensors are *dense*, meaning that their storage can become unattractive, especially if a large reduced order  $r$  or high-order polynomials are considered.

Finally, note that an alternative approach is to reduce the nonlinear system (6.1) first, and then calculate the *reduced* polynomial tensors  $\mathbf{A}_{2r}$ ,  $\mathbf{A}_{3r}$ , ... based on  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u})$  and  $\mathbf{A}_r(\mathbf{x}_r, \mathbf{u})$ . This can be accomplished by considering the same procedures (i), (ii) or (iii) mentioned in Section 4.1.1, but now applied at the reduced-order level. This involves: (i) an efficient formulation of the products  $\mathbf{W}^\top \mathbf{A}_2 (\mathbf{V} \otimes \mathbf{V})$ , etc., (ii) numerical differentiation of  $\mathbf{A}_{1r}$ , or

(iii) an identification of reduced coefficients using e.g. [182, 120, 210]. These approaches have been applied in the context of nonlinear structural dynamics [178, 224], and could, in the author's opinion, also prove successful in the field of state-space systems and fluid dynamics.

### 6.5.2 Piecewise linear approximation

Another way of accomplishing hyper-reduction is to represent the nonlinear function using a piecewise linear (PWL) approximation, e.g. like in (6.14), or more generally as:

$$\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) \approx \widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \sum_{s=1}^S \omega_s(\mathbf{x}_r) \left( \mathbf{f}(\bar{\mathbf{x}}_s, \bar{\mathbf{u}}_s) + \mathbf{A}_s(\boldsymbol{\nu}(\mathbf{x}_r) - \bar{\mathbf{x}}_s) + \mathbf{B}_s(\mathbf{u} - \bar{\mathbf{u}}_s) \right). \quad (6.40)$$

The nonlinearity is thus constructed by a weighted interpolation of reduced linearized matrices, obtained via linearization at different given points  $\{\mathbf{x}_s\}_{s=1}^S$ . For more details concerning this approach, the reader is referred to [109, Sec. 4.7.2].

### 6.5.3 Classical hyper-reduction

In contrast to the afore explained approaches, “classical” hyper-reduction techniques like the *Discrete Empirical Interpolation Method* (DEIM) [74] or the *Energy-Conserving Sampling and Weighting* (ECSW) [87] achieve the computational speed-up by evaluating the nonlinear function at fewer elements of the full mesh. Both methods subtly differ in the way they choose the reduced set of elements for the approximation, wherefore they are reviewed next.

**Discrete Empirical Interpolation Method** The theoretical foundation for DEIM was initially laid by the gappy POD [85, 270], the missing point estimation [11] and the empirical interpolation method (EIM) [19]. The key idea is to approximate the nonlinear function as

$$\mathbf{f}(\mathbf{V}\mathbf{x}_r) \approx \mathbf{f}_{\text{DEIM}}(\mathbf{V}\mathbf{x}_r) = \mathbf{U}_f \mathbf{c}(\mathbf{V}\mathbf{x}_r), \quad (6.41)$$

with the basis  $\mathbf{U}_f \in \mathbb{R}^{n \times m}$  and the coefficient vector  $\mathbf{c}(\mathbf{V}\mathbf{x}_r) \in \mathbb{R}^m$ . The approximation is then carried out such that the vector  $\mathbf{f}_{\text{DEIM}}(\mathbf{V}\mathbf{x}_r)$  is equal to  $\mathbf{f}(\mathbf{V}\mathbf{x}_r)$  at certain dofs, and interpolated elsewhere. This *collocation* is achieved by the Boolean matrix  $\mathbf{P} \in \mathbb{R}^{n \times m}$ , enforcing  $\mathbf{P}^\top \mathbf{U}_f \mathbf{c}(\mathbf{V}\mathbf{x}_r) \stackrel{!}{=} \mathbf{P}^\top \mathbf{f}(\mathbf{V}\mathbf{x}_r)$ . This latter equation can be solved for the coefficient vector  $\mathbf{c}$  and then inserted in the approximation ansatz (6.41), leading to

$$\mathbf{f}_{\text{DEIM}}(\mathbf{V}\mathbf{x}_r, \mathbf{u}) = \mathbf{U}_f (\mathbf{P}^\top \mathbf{U}_f)^{-1} \mathbf{P}^\top \mathbf{f}(\mathbf{V}\mathbf{x}_r, \mathbf{u}). \quad (6.42)$$

The whole procedure represents a projection of the nonlinear function by the *oblique* projector  $\mathbf{\Pi}_{\text{DEIM}} = \mathbf{U}_f (\mathbf{P}^\top \mathbf{U}_f)^{-1} \mathbf{P}^\top$ , where  $\det(\mathbf{P}^\top \mathbf{U}_f) \neq 0$  is assumed. In FE-notation, the approximation of the reduced nonlinear function  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{W}^\top \mathbf{f}(\mathbf{V}\mathbf{x}_r, \mathbf{u})$  by the hyper-reduced one  $\mathbf{f}_{r,\text{DEIM}}(\mathbf{x}_r, \mathbf{u}) = \mathbf{W}^\top \mathbf{f}_{\text{DEIM}}(\mathbf{V}\mathbf{x}_r, \mathbf{u})$  reads as follows:

$$\begin{aligned} \mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) &= \sum_{e=1}^{n_e} \mathbf{W}^\top \mathbf{L}_e^\top \mathbf{f}_e(\mathbf{L}_e \mathbf{V}\mathbf{x}_r, \mathbf{u}) \\ &\approx \sum_{e \in \tilde{E}} \mathbf{W}^\top \mathbf{U}_f (\mathbf{P}^\top \mathbf{U}_f)^{-1} \mathbf{P}^\top \mathbf{L}_e^\top \mathbf{f}_e(\mathbf{L}_e \mathbf{V}\mathbf{x}_r, \mathbf{u}) = \mathbf{f}_{r,\text{DEIM}}(\mathbf{x}_r, \mathbf{u}). \end{aligned} \quad (6.43)$$

Herein,  $\mathbf{L}_e$  represents the Boolean assembly matrix corresponding to the element  $e$  and  $n_e$  denotes the number of elements of the full mesh [224, Ch. 2, 12]. Concerning the implementation, the matrix  $\mathbf{M} := \mathbf{W}^\top \mathbf{U}_f (\mathbf{P}^\top \mathbf{U}_f)^{-1} \in \mathbb{R}^{r \times m}$  is precomputed, whereas the term  $\mathbf{P}^\top \mathbf{f}(\mathbf{V} \mathbf{x}_r, \mathbf{u})$  only needs to be evaluated at  $m$  entries of  $\mathbf{f} \in \mathbb{R}^n$ . Hereby, the operator  $\mathbf{P}^\top \mathbf{L}_e^\top$  is efficiently realized by index operations, in order to evaluate the nonlinearity only at the dofs corresponding to non-zero rows in  $\mathbf{P}$ . Thus, huge computational savings can be obtained by hyper-reduction due to  $m \ll n$  and the looping over a reduced set of elements  $\tilde{E}$ . Please note that the reduced Jacobian  $\mathbf{A}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{W}^\top \mathbf{A}(\mathbf{V} \mathbf{x}_r, \mathbf{u}) \mathbf{V}$  can also be approximated via DEIM using the hyper-reduced matrix  $\mathbf{A}_{r,\text{DEIM}}(\mathbf{x}_r, \mathbf{u}) = \mathbf{W}^\top \mathbf{U}_f (\mathbf{P}^\top \mathbf{U}_f)^{-1} \mathbf{P}^\top \mathbf{A}(\mathbf{V} \mathbf{x}_r, \mathbf{u}) \mathbf{V}$ .

For DEIM, both the basis  $\mathbf{U}_f$  and the collocation matrix  $\mathbf{P}$  have to be calculated. For the computation of  $\mathbf{U}_f$  some representative information about the nonlinear function  $\mathbf{f}$  is required, which is however difficult to obtain in a system-theoretic manner. Therefore, similarly to  $\mathbf{X} = [\mathbf{x}(t_1), \dots, \mathbf{x}(t_{n_s})] \in \mathbb{R}^{n \times n_s}$  and POD, *nonlinear function snapshots*  $\mathbf{F} = [\mathbf{f}(\mathbf{x}(t_1)), \dots, \mathbf{f}(\mathbf{x}(t_{n_s}))] \in \mathbb{R}^{n \times n_s}$  gathered from a FOM simulation for certain training input signals  $\mathbf{u}_{\text{train}}(t)$  are usually employed to obtain the basis  $\mathbf{U}_f \in \mathbb{R}^{n \times m}$  via SVD. After that, the *greedy* DEIM algorithm [74] is used to select the collocation indices  $p_1, \dots, p_m$ , i.e. the Boolean matrix  $\mathbf{P}$ . Note that a different algorithm named QDEIM [80] selects the indices based on the permutation/pivoting information of the QR decomposition of  $\mathbf{U}_f^\top$ .

Different variants and extensions of the discrete empirical interpolation method are available in the literature. For instance, the unassembled DEIM (UDEIM) [261] operates in the unassembled finite element mesh, making the method more efficient for FE applications. Moreover, the hyper-reduction of nonlinear *matrix-valued* functions or the Jacobian  $\mathbf{A}_r(\mathbf{x}_r, \mathbf{u})$  can be accomplished by the matrix version of DEIM (MDEIM), using either vectorization [280, 191] or a different approach presented in [148]. Finally, due to the oblique projection and consequent loss of symmetry and stability, a symmetrized version of DEIM has been proposed for nonlinear port-Hamiltonian systems in [56].

Another DEIM-related hyper-reduction technique is given by the Gauss-Newton with Approximated Tensors (GNAT) [50]. Recently, a structure-preserving variant of this method has been proposed for structural dynamics [76] and finite-volume models [58]. In the following, a conceptually different structure- and energy-preserving hyper-reduction approach is described.

**Energy-Conserving Sampling and Weighting** The ECSW hyper-reduction method has been proposed by Farhat et al. [87, 88] in the context of structural dynamics, and is based on the principle of *virtual work*. The main idea is to find a reduced set of elements  $\tilde{E}$  with  $|\tilde{E}| \ll n_e$  and positive weights  $\xi_e^* \geq 0$ , such that the virtual work of the *reduced* nonlinear term is preserved. Using a Galerkin projection ( $\mathbf{W} = \mathbf{V}$ ), the reduced nonlinear function  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{V}^\top \mathbf{f}(\mathbf{V} \mathbf{x}_r, \mathbf{u})$  is approximated by the hyper-reduced one  $\mathbf{f}_{r,\text{ECSW}}(\mathbf{x}_r, \mathbf{u})$  as:

$$\mathbf{f}_r(\mathbf{x}_r, \mathbf{u}) = \sum_{e=1}^{n_e} \mathbf{V}^\top \mathbf{L}_e^\top \mathbf{f}_e(\mathbf{L}_e \mathbf{V} \mathbf{x}_r, \mathbf{u}) \approx \sum_{e \in \tilde{E}} \xi_e^* \mathbf{V}^\top \mathbf{L}_e^\top \mathbf{f}_e(\mathbf{L}_e \mathbf{V} \mathbf{x}_r, \mathbf{u}) = \mathbf{f}_{r,\text{ECSW}}(\mathbf{x}_r, \mathbf{u}). \quad (6.44)$$

Similarly, the reduced Jacobian  $\mathbf{A}_r(\mathbf{x}_r, \mathbf{u}) = \mathbf{V}^\top \mathbf{A}(\mathbf{V} \mathbf{x}_r, \mathbf{u}) \mathbf{V}$  can also be approximated by  $\mathbf{A}_{r,\text{ECSW}}(\mathbf{x}_r, \mathbf{u})$ . Again, the computational cost associated with the evaluation of the nonlinearity and Jacobian is drastically reduced due to  $|\tilde{E}| \ll n_e$ . In addition, ECSW can preserve intrinsic properties of the model like e.g. symmetry, energy and stability, making the method particularly suitable for mechanical/conservative systems.

For the computation of the weights  $\xi_e^*$ , some information about the *reduced* nonlinear function  $\mathbf{f}_r$  is required. Usually, state snapshots  $\{\mathbf{x}(t_k)\}_{k=1}^{n_s} \in \mathbb{R}^n$  obtained from a FOM simulation for certain training inputs  $\mathbf{u}_{\text{train}}(t)$  are used to get *reduced* state snapshots  $\{\mathbf{x}_r(t_k)\}_{k=1}^{n_s} \in \mathbb{R}^r$  via projection

$$\mathbf{x}_{r,k} = (\mathbf{V}^\top \mathbf{E} \mathbf{V})^{-1} \mathbf{V}^\top \mathbf{E} \mathbf{x}_k. \quad (6.45)$$

After that, the reduced function contributions  $\mathbf{g}_{ke} \in \mathbb{R}^r$  and  $\mathbf{b}_k \in \mathbb{R}^r$

$$\mathbf{g}_{ke} = \mathbf{V}^\top \mathbf{L}_e^\top \mathbf{f}_e(\mathbf{L}_e \mathbf{V} \mathbf{x}_{r,k}, \mathbf{u}_{\text{train},k}), \quad \mathbf{b}_k = \sum_{e=1}^{n_e} \mathbf{g}_{ke}, \quad (6.46)$$

of all  $n_e$  elements and for all  $n_s$  training snapshots are arranged in the matrix  $\mathbf{G} \in \mathbb{R}^{r n_s \times n_e}$  and the vector  $\mathbf{b} \in \mathbb{R}^{r n_s}$ . The weights are then computed by solving the following inexact sparse non-negative least-squares (S-NNLS) problem:

$$\xi^* \approx \arg \min_{\xi} \|\mathbf{G} \xi - \mathbf{b}\|_2^2 \quad \text{s.t.} \quad \xi \in \mathbb{R}^{n_e}, \quad \xi \geq \mathbf{0}, \quad (6.47)$$

using an active-set greedy algorithm. The non-zero entries of  $\xi^*$  form the reduced mesh  $\tilde{E}$ .

**Simulation-free generation of snapshots** For both DEIM and ECSW representative information about the (reduced) nonlinear function is necessary to compute the matrix  $\mathbf{U}_f$  or the weights  $\xi_e^*$ . If the reduction basis  $\mathbf{V}$  is constructed using a simulation-based approach like POD, then state snapshots  $\{\mathbf{x}_k\}_{k=1}^{n_s}$  are already available from training simulations. In such case, it is straightforward to use them to obtain either the nonlinear function or the reduced state snapshots (6.45) required for the hyper-reduction methods. This combination is called POD-DEIM or POD-ECSW, and has been widely used in numerous applications. If, on the contrary, the basis  $\mathbf{V}$  is calculated in a simulation-free manner (using e.g. basis augmentation or the approximated NLMM Algorithm 7.1), then it is not appealing to run expensive FOM training simulations to gather snapshots for hyper-reduction. In such case, it is more reasonable to employ a *simulation-free* hyper-reduction technique, where the offline costs for the generation of training sets are lower. Two approaches developed in the context of structural dynamics are mentioned in the following.

The quadratic manifold lifting approach [137] uses an augmented basis  $\mathbf{V}_{\text{aug}} = [\mathbf{V}^{(1)}, \mathbf{V}^{(2)}]$  for dimensional reduction, and quadratically lifted linear reduced coordinates as training snapshots for hyper-reduction (using ECSW):

$$\mathbf{x}_{k,\text{lifted}} = \mathbf{V}^{(1)} \mathbf{x}_{r,k}^{\text{lin}} + \mathbf{V}^{(2)} (\mathbf{x}_{r,k}^{\text{lin}} \otimes \mathbf{x}_{r,k}^{\text{lin}}). \quad (6.48)$$

To obtain the linear reduced snapshots  $\mathbf{x}_{r,k}^{\text{lin}}$ , the *linear(ized)* system (cf. Eq. (6.12)) is reduced using only  $\mathbf{V}^{(1)}$ , and then the linear ROM is simulated very cheaply. The solution  $\mathbf{x}_r^{\text{lin}}(t)$  is then lifted on a quadratic manifold, in order to generate snapshots that capture the nonlinear behavior. Note, however, that this approach may perform badly, especially if the system dynamics do not evolve over the selected manifold [223, 219, 224]. In such case, a different manifold ansatz or the following linear subspace-based approach could be attempted.

The lean snapshot generation technique [219] is based on the solution of nonlinear static problems of the form  $\mathbf{f}(\mathbf{x}_\ell) = \mathbf{f}_{\text{rand},\ell}$  for randomly generated vectors  $\mathbf{f}_{\text{rand},\ell}$ . The latter are

calculated by amplifying the basis vectors of the Krylov subspace  $\mathcal{K}_q(-\mathbf{A}_{\text{eq}}^{-1}\mathbf{E}, -\mathbf{A}_{\text{eq}}^{-1}\mathbf{B}_{\text{eq}}\mathbf{r})$  with Gaussian random values. After that, nonlinear systems of equations (NLSEs) must be solved to calculate the snapshots  $\mathbf{x}_\ell$  needed for hyper-reduction. Although the offline costs of this method are higher than for the quadratic manifold lifting approach, its applicability is wider since the snapshots are not restricted to a specific manifold. [224, Sec. 13.3]

#### 6.5.4 Nonlinear manifold-based hyper-reduction

Linear projection-based dimensional reduction approaches — such as POD, augmented basis or approximated NLMM (cf. Algorithm 7.1) — can be combined with (1) a polynomial system representation, (2) a PWL approximation or (3) (simulation-free) hyper-reduction based on DEIM/ECSW. If, on the contrary, a nonlinear projection-based dimensional reduction approach (e.g. a quadratic manifold) is used to obtain the ROM (6.24), then (1) a polynomial expansion or (2) PWL approximation can be employed, whereas “classical” DEIM/ECSW are not possible. To overcome this, the method of ECSW has recently been extended to allow for hyper-reduction using nonlinear (quadratic) manifolds [138]. In the author’s opinion, a similar extension could also be attempted with DEIM or GNAT, in order to enable nonlinear manifold-based hyper-reduction for first-order systems as well.

## 6.6 Discussion

Before concluding this chapter, we want to discuss the properties, advantages and disadvantages of the linear and nonlinear Petrov-Galerkin projection framework.

Nonlinear dynamical systems are usually reduced via projection onto *linear* subspaces. This is due to the fact that classical Petrov-Galerkin projections are simple and well-understood. Moreover, the time integration of a linear subspace ROM (6.9) barely changes w.r.t. the numerical simulation of the nonlinear FOM. This allows the employment of (almost) the same standardized toolchain, without having to intervene or adapt well-established numerical software. In addition, several nonlinear reduction approaches exist to construct the linear bases (e.g. POD, TPWL, basis augmentation), which have proven successful in many applications. However, the linear subspaces employed for projection should comprise enough nonlinear information in order to yield a good approximation. This means that many basis vectors may need to be considered (e.g. in basis augmentation approaches or in case of slowly decaying singular values), thus leading to a high reduced order. Finally, linear projections may impose a limitation in terms of accuracy and the restriction to linear subspaces, especially if the underlying dynamics are expected to evolve in more sophisticated manifolds.

The nonlinear Petrov-Galerkin projection framework represents a very promising way of reducing dynamical systems. For instance, applying a nonlinear manifold or a series expansion ansatz with user-defined basis functions may outperform the classical linear projection, or even be indispensable in certain cases to obtain decent results. Furthermore, this approach allows to reduce a system to a smaller set of coordinates (cf. (6.13) with (6.35)) because the reduction mapping  $\boldsymbol{\nu}(\mathbf{x}_r)$  depends nonlinearly on the reduced state vector. Nevertheless, nonlinear projections are more difficult and less developed than linear ones. The projection onto the tangent space of the manifold results in a more complicated ROM (6.24), whose time integration is also more involved due to the state-dependent bases  $\tilde{\mathbf{V}}_{\mathbf{x}_r}$  and  $\tilde{\mathbf{W}}_{\mathbf{x}_r}$ . Moreover, it might be challenging to select a particular ansatz (6.28) or appropriate basis functions fitting

to the problem under study. To be more precise: since the solution of the ROM is constrained to evolve on the selected manifold, its suitable selection is crucial for a good performance of the method. Some *system-theoretic* approaches to calculate quadratic and general nonlinear manifolds have been explained in Section 6.4.3. Recent *data-driven* trends in the field of MOR employ deep neural networks to learn these nonlinear manifolds from data [199, 159]. Herein, an appropriate network architecture tailored to the system at hand together with representative snapshot data are essential for the approximation quality.

The suitability of a linear or nonlinear projection for model order reduction is strongly problem- and physics-dependent. Thus, a general recommendation cannot be given. Nevertheless, this chapter discusses dimensional *and* hyper-reduction approaches for both categories, offers a unifying view via conceptual comparisons between the approaches, and uncovers new perspectives for future research. Furthermore, we have implemented (from scratch) our own simulation framework with functions such as `implicitEuler` and `NewtonRaphson`, as well as developed fundamental model reduction routines like `basisRed`, `DEIM`, etc.<sup>3</sup> These algorithms lay the foundation for the time integration, simulation-based reduction and hyper-reduction of nonlinear state-space systems. The `NewtonRaphson` scheme can also be employed within the approximated nonlinear moment matching algorithm presented in the next chapter.



MATLAB function(s): `ode45`, `ode23`, `ode15s`, `ode23s`, ...  
POD-DEIM function(s): `implicitEuler`, `NewtonRaphson`, `NumJacobian`, ...  
`POD`, `basisRed`, `DEIM`, `QDEIM`, `funcSnapBasisDEIM`, ...

<sup>3</sup>Most functions of our simulation framework are available under <https://doi.org/10.5281/zenodo.3542641>.

## Chapter 7

# Model Reduction by Approximated Nonlinear Moment Matching

In the previous chapter, we have revisited many approaches to reduce large-scale nonlinear dynamical systems and have categorized them in linear (Section 6.3.3) or nonlinear projection-based techniques (Section 6.4.3). The methods can also be classified in two main branches depending on their fundamental concept. *Simulation-based* dimensional reduction techniques such as Proper Orthogonal Decomposition (POD), balanced-POD, empirical Gramians, Trajectory Piecewise Linear (TPWL) and Reduced Basis methods rely on expensive full training simulations for several input excitations to construct the reduction basis. On the other hand, *simulation-free* reduction procedures try to extract the most dominant nonlinear dynamics by exploiting some system-theoretic concept rather than from simulated data. Examples include basis augmentation or quadratic manifold with eigenvector/modal derivatives, Krylov subspace methods for polynomial systems (subsystem/Volterra series interpolation), nonlinear balanced truncation as well as nonlinear moment matching. From a system-theoretic perspective, this latter method represents a promising approach towards a reduction procedure for *general* nonlinear systems, which does not rely on the numerical simulation of the full model to construct the reduced model. Thus, we will focus on this method in the following.

The transfer of the moment matching concept from linear to nonlinear systems has been initiated by Astolfi [12, 13, 14] based on the center manifold theory [51], the steady-state response of nonlinear systems [131, Ch. 8] and the techniques of nonlinear output regulation [146, 124]. Since then, moment matching for linear and, specially, nonlinear systems has been further developed in several publications. For instance, the equivalence between projection-based and non-projective families of reduced models achieving moment matching is presented in [15]. Therein, the time-domain interpretation of *output Krylov subspace*-based moment matching is also established for linear systems using the dual Sylvester equation. These findings are transferred to the nonlinear case in [126] and further developed in [127] to provide a two-sided, nonlinear moment matching theory. Moreover, the steady-state interpretation of moments is extended to linear and nonlinear time-delay systems in [228]. Although the previously mentioned papers [14, 126, 127, 228] are very appealing from a theoretical point of view, practically they all face the same difficulty, namely the solution of a nonlinear Sylvester-like partial differential equation to compute the reduction mapping  $\nu(\mathbf{x}_r)$ . More recently, the data-driven low-order identification of an unknown nonlinear system by moment matching has been presented in [231]. The proposed algorithm does not involve the solution of a partial differential equation, as it rather aims at estimating the moments of a nonlinear system from *time-domain* input-output data. In this sense, the approach is related to the Loewner framework [171], which is a *frequency-domain* data-driven approach achieving moment matching, as well as to other *time-domain* low-order system identification techniques

(e.g. the time-domain Loewner framework [204], the dynamic mode decomposition [238] and machine/deep/manifold learning [43, 199, 260, 159]).

In this chapter we develop the concept of nonlinear moment matching presented in [14] towards practical application. Inspired by the POD community, which usually employs a linear projection and time-snapshots to reduce nonlinear systems, we propose some simplifications to *approximate* the Sylvester-like PDE and achieve a feasible, numerical algorithm for model reduction. The proposed employment of a *linear projection* renders the PDE into a system of nonlinear *algebraic* equations, which is much easier to solve. Note that this simplification resembles a special case of the power series approximation method (6.28) from [146], [124, Ch. 4], which in turn is related to the asymptotic expansion (aka. Poincaré/naïve expansion [187, 264, 136] or variational equation approach [221]) usually employed for approximating the solution of nonlinear differential equations. Further note that other approximation techniques for invariant manifolds exist, such as e.g. the symbolic iteration scheme [217, 216] as well as numerical path continuation (cf. [140, 142]). Comparisons between invariant manifold approaches and perturbation techniques can be found e.g. in [186, 201].

Our proposed approach is linked to the technique presented in [231] in the sense that both methods *approximately* match nonlinear moments. However, the goals of both techniques are different, since [231] focuses on the data-driven, *low-order identification* of an unknown nonlinear system, whereas we deal with the *reduction* of a known nonlinear system. For this reason, the proposed algorithms are also different. Algorithm 2 in [231] requires the solution of a moving window, recursive least-square estimation problem using input-output measurements and user-defined basis functions for the reduced output mapping, whereas the practical algorithm presented here relies on the solution of nonlinear systems of equations using the explicitly known governing system.

Based on the steady-state interpretation of moment matching for linear systems revisited in Section 3.6, in the following we first explain its extension to nonlinear systems due to [14]. Then, in Section 7.2 our simplifications towards a *feasible, simulation-free* algorithm for *approximated* nonlinear moment matching are proposed and extensively discussed in Section 7.3. This is followed by practical guidelines in Section 7.4, which instruct practitioners with the degrees of freedom and ease the application of the proposed algorithm. In Section 7.5, two benchmark examples are employed to illustrate the efficacy of the presented reduction approach both in terms of approximation quality and computational effort. Finally, Section 7.6 contains further remarks concerning the applicability of the algorithm to finite element models and its combination with simulation-free hyper-reduction.

Many parts of this chapter represent an edited and extended version of the corresponding sections of [72] and [71]. The theses [Suk17] and [Sch18] have also contributed to this chapter.

## 7.1 Steady-state-based nonlinear moment matching

In this section, the extension of moment matching to nonlinear systems presented in [13, 14] is explained in detail.

### 7.1.1 Time-domain notion of nonlinear moments

First, the steady-state-based interpretation of nonlinear moments is described in a similar fashion as in Section 3.6.1.



### Notion of nonlinear signal generator

Consider the following *nonlinear signal generator* (cf. [14])

$$\dot{\mathbf{x}}_r^v(t) = \mathbf{s}_v(\mathbf{x}_r^v(t)), \quad \mathbf{x}_r^v(0) = \mathbf{x}_{r,0}^v \neq \mathbf{0}, \quad (7.1a)$$

$$\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t)), \quad (7.1b)$$

where  $\mathbf{s}_v(\mathbf{x}_r^v) : \mathbb{R}^r \rightarrow \mathbb{R}^r$ ,  $\mathbf{r}(\mathbf{x}_r^v) : \mathbb{R}^r \rightarrow \mathbb{R}^m$  are smooth mappings such that  $\mathbf{s}_v(\mathbf{0}) = \mathbf{0}$  and  $\mathbf{r}(\mathbf{0}) = \mathbf{0}$ . Hereby it is assumed that:

1. The signal generator  $(\mathbf{s}_v, \mathbf{r}, \mathbf{x}_{r,0}^v)$  is *observable*, i.e. for any pair of initial conditions  $\mathbf{x}_{r,a}^v(0) \neq \mathbf{x}_{r,b}^v(0)$ , the corresponding trajectories  $\mathbf{r}(\mathbf{x}_{r,a}^v(t))$  and  $\mathbf{r}(\mathbf{x}_{r,b}^v(t))$  do not coincide:  $\mathbf{r}(\mathbf{x}_{r,a}^v(t)) \neq \mathbf{r}(\mathbf{x}_{r,b}^v(t))$ .
2. The signal generator is *neutrally stable*<sup>1</sup>, i.e. Poisson stable in a neighborhood of its stable equilibrium  $\mathbf{x}_{r,\text{eq}}^v = \mathbf{0}$ .

Although neutral stability is crucial for establishing a meaningful relation between moments and *well-defined* steady-state responses, note that this assumption can be relaxed. The neutral stability of the signal generator will be further discussed in Section 7.3.

Interconnecting a system with the nonlinear signal generator (7.1) corresponds to exciting the system with an input signal  $\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$ , where  $\mathbf{x}_r^v(t)$  is the given or computed solution of the nonlinear ordinary differential equation (7.1a). Therefore, the signal generator should be chosen such that it characterizes and excites the important dynamics of the underlying nonlinear system. This will also be discussed later.

### Steady-state response of interconnected system

Consider the interconnection of system (6.1), where  $\mathbf{x}_{\text{eq}} = \mathbf{0}$  is locally exponentially stable, with the nonlinear signal generator (7.1), cf. Fig. 7.1. The response of such interconnected system is (similar to the linear case) given by

$$\mathbf{x}(t) = \underbrace{\tau(t, \mathbf{x}_0 - \boldsymbol{\nu}(\mathbf{x}_{r,0}^v))}_{\mathbf{x}_t(t)} + \underbrace{\boldsymbol{\nu}(\mathbf{x}_r^v(t))}_{\mathbf{x}_{\text{ss}}(t)}, \quad (7.2)$$

where  $\mathbf{x}_t(t)$  is the *transient solution* with the *nonlinear transition mapping*  $\tau(t, \mathbf{x}_0)$ , and  $\mathbf{x}_{\text{ss}}(t)$  defines the *steady-state solution*. Hence,  $\mathbf{y}_{\text{ss}}(t) = \mathbf{h}(\mathbf{x}_{\text{ss}}(t)) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r^v(t)))$  — or  $\mathbf{y}_{\text{ss}}(t) = \mathbf{C} \mathbf{x}_{\text{ss}}(t) = \mathbf{C} \boldsymbol{\nu}(\mathbf{x}_r^v(t))$  for a linear output mapping — is the *steady-state response* of the nonlinear system (6.1) to an input generated by (7.1). By imposing the condition  $\mathbf{x}_0 \stackrel{!}{=} \boldsymbol{\nu}(\mathbf{x}_{r,0}^v)$ , for  $\mathbf{x}_{r,0}^v \neq \mathbf{0}$  arbitrary, such that  $\tau(t, \mathbf{0}) = \mathbf{0} \forall t$  holds, then the transient solution  $\mathbf{x}_t(t)$  vanishes for all  $t$ . This yields  $\mathbf{x}(t) = \mathbf{x}_{\text{ss}}(t) \forall t$ .

Based on this result, we are ready to present the steady-state interpretation of nonlinear input moments in the next lemma.

<sup>1</sup>Corresponds to  $\lambda(\mathcal{S}_v) \subset \mathbb{C}_0$  in the linear setting, i.e. exciting the system with a permanent oscillation (cf. [131, Ch. 8] and Sec. 3.6.1).

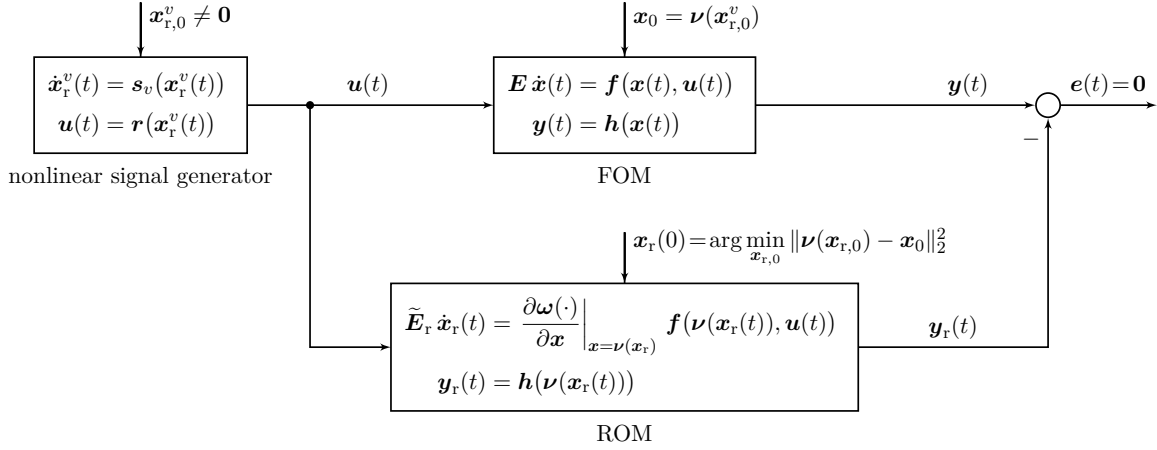


Figure 7.1: Interconnection between the nonlinear FOM/ROM and the nonlinear signal generator to illustrate the time-domain interpretation of moment matching for nonlinear systems.

**Lemma 7.1** (Steady-state notion of nonlinear input moments). The 0-th nonlinear moments  $\mathbf{m}_0(\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v)$  at  $\{\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v\}$  are related to the (locally well-defined) steady-state response

$$\mathbf{y}_{\text{ss}}(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r^v(t))) := \mathbf{m}_0(\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v) \quad (7.3)$$

of the interconnected system from Fig. 7.1, where the mapping  $\boldsymbol{\nu}(\mathbf{x}_r^v)$ , defined in a neighborhood of  $\mathbf{x}_{r,\text{eq}}^v = \mathbf{0}$ , is the unique solution of the following Sylvester-like partial differential equation (PDE)

$$\mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r^v)}{\partial \mathbf{x}_r^v} \mathbf{s}_v(\mathbf{x}_r^v) = \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r^v), \mathbf{r}(\mathbf{x}_r^v)). \quad (7.4)$$

**Remark 7.1** (Local invariant manifold [14]). By the center manifold theory, the interconnected system possesses a locally well-defined invariant manifold at  $(\mathbf{x}_{\text{eq}}, \mathbf{x}_{r,\text{eq}}^v) = (\mathbf{0}, \mathbf{0})$  given by  $\mathcal{M} = \{(\mathbf{x}, \mathbf{x}_r^v) \in \mathbb{R}^{n+r} : \mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r^v)\}$ , where  $\boldsymbol{\nu}(\mathbf{x}_r^v)$  satisfies (7.4). Moreover, the dynamics of the interconnected system restricted to the manifold  $\mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r^v)$  are described by  $\dot{\mathbf{x}}_r^v = \mathbf{s}_v(\mathbf{x}_r^v)$ , since  $\mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r^v)}{\partial \mathbf{x}_r^v} \dot{\mathbf{x}}_r^v = \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r^v), \mathbf{r}(\mathbf{x}_r^v))$ .  $\triangle$

### 7.1.2 Nonlinear moment matching by interconnection

Based on Lemma 7.1, the perception of nonlinear moment matching in terms of the interpolation of the steady-state response of an interconnected system follows.

**Theorem 7.1** (Steady-state-based nonlinear moment matching). *Consider the interconnection of system (6.1) with the nonlinear signal generator (7.1), where the triple  $(\mathbf{s}_v, \mathbf{r}, \mathbf{x}_{r,0}^v)$  is assumed observable and neutrally stable. Let  $\boldsymbol{\nu}(\mathbf{x}_r^v)$  be the unique solution of the Sylvester-like PDE (7.4) and  $\boldsymbol{\omega}(\cdot)$  arbitrary such that  $\det(\widetilde{\mathbf{W}}_{x_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{x_r}) \neq 0$ . Furthermore, let  $\mathbf{x}_0 = \boldsymbol{\nu}(\mathbf{x}_{r,0}^v)$  with*

$\mathbf{x}_{r,0}^v \neq \mathbf{0}$  arbitrary. Then, the (exponentially stable) ROM (6.24) exactly matches the (locally well-defined) steady-state response of the output of the FOM (cf. Fig. 7.1), i.e.

$$\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{y}_r(t) = \mathbf{h}(\mathbf{x}(t)) - \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r(t))) = \mathbf{0} \quad \forall t. \quad (7.5)$$

**Corollary 7.1** (Exact moment matching vs. interpolation). *Thus, moment matching for nonlinear systems can be interpreted as the exact matching of the steady-state response of the FOM and ROM*

$$\begin{aligned} \mathbf{y}_{ss}(t) &= \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r^v(t))) = \mathbf{m}_0(\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v) \\ &\equiv \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_{r,ss}(t))) = \mathbf{m}_{r,0}(\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v) = \mathbf{y}_{r,ss}(t), \end{aligned} \quad (7.6)$$

when both are excited with the signal generator (7.1) (see Fig. 7.1) with same  $\mathbf{s}_v$ ,  $\mathbf{r}$  as the ones used during the reduction. For other arbitrary input signals the steady-state response is interpolated. Note here again that the transient response of the FOM vanishes, if the initial condition is chosen like  $\mathbf{x}_0 = \boldsymbol{\nu}(\mathbf{x}_{r,0}^v)$ . In such case, the matching conditions hold for all  $t$  (transient+steady-state).

*Proof.* The output of the FOM for  $\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$  has been derived in (7.2) and Lemma 7.1. Exciting the exponentially stable ROM, i.e.  $\mathbf{x}_{r,eq} = \mathbf{0}$  is stable, with the very same signal  $\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$  with  $\mathbf{x}_r(t) \stackrel{!}{=} \mathbf{x}_r^v(t)$  yields

$$\tilde{\mathbf{E}}_r \dot{\mathbf{x}}_r(t) = \left. \frac{\partial \boldsymbol{\omega}(\boldsymbol{\xi}(\mathbf{x}(t), \mathbf{u}(t)))}{\partial \mathbf{x}(t)} \right|_{\mathbf{x}=\boldsymbol{\nu}(\mathbf{x}_r)} \underbrace{\mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r(t)), \mathbf{r}(\mathbf{x}_r(t)))}_{\mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r(t))}{\partial \mathbf{x}_r(t)} \mathbf{s}_v(\mathbf{x}_r(t))}$$

and, consequently,

$$\dot{\mathbf{x}}_r(t) = \mathbf{s}_v(\mathbf{x}_r(t)), \quad \mathbf{x}_r(0) = \arg \min_{\mathbf{x}_{r,0}} \|\boldsymbol{\nu}(\mathbf{x}_{r,0}) - \mathbf{x}_0\|_2^2, \quad (7.7)$$

whose solution is  $\mathbf{x}_r(t)$ . Thus, the output of the ROM for  $\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$  is given by  $\mathbf{y}_r(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r(t)))$  with  $\mathbf{x}_r(t)$  satisfying (7.7). Therefore, we achieve *exact* moment matching for all  $t$ , if  $\mathbf{x}_r(0) \stackrel{!}{=} \mathbf{x}_{r,0}^v$ , i.e. if  $\mathbf{x}_0 \stackrel{!}{=} \boldsymbol{\nu}(\mathbf{x}_{r,0}^v)$ .  $\blacksquare$

### 7.1.3 Derivation of nonlinear Sylvester-like partial differential equation

The Sylvester-like PDE (7.4) represents the nonlinear counterpart of the linear equation (3.89) with  $\mathbf{x}_r^v(t) = e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$ :

$$\mathbf{E} \mathbf{V} \mathbf{S}_v e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v = \mathbf{A} \mathbf{V} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v + \mathbf{B} \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v. \quad (7.8)$$

Thus, the PDE can be similarly derived as follows. First, the nonlinear approximation ansatz  $\mathbf{x}(t) = \boldsymbol{\nu}(\mathbf{x}_r(t))$  with  $\mathbf{x}_r(t) \stackrel{!}{=} \mathbf{x}_r^v(t)$  is inserted in the state equation (6.1a):

$$\mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r^v(t))}{\partial \mathbf{x}_r^v(t)} \dot{\mathbf{x}}_r^v(t) = \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r^v(t)), \mathbf{u}(t)). \quad (7.9)$$

Afterwards, the nonlinear signal generator  $\dot{\mathbf{x}}_r^v(t) = \mathbf{s}_v(\mathbf{x}_r^v(t))$ ,  $\mathbf{u}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$  is plugged into (7.9), yielding

$$\mathbf{E} \frac{\partial \boldsymbol{\nu}(\mathbf{x}_r^v(t))}{\partial \mathbf{x}_r^v(t)} \mathbf{s}_v(\mathbf{x}_r^v(t)) = \mathbf{f}(\boldsymbol{\nu}(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t))). \quad (7.10)$$

In contrast to the linear, state-independent Sylvester equation (3.62a) of dimension  $n \times r$ , note that the PDE (7.10) is a *nonlinear, state-dependent* equation of dimension  $n \times 1$ . This follows from the fact that the state vector  $\mathbf{x}_r^v(t)$  cannot be factored out so easily anymore. This shortcoming will be further discussed in Section 7.2.

### 7.1.4 Families of reduced models achieving nonlinear moment matching

The *projection-based* approach to construct a reduced model achieving (input) nonlinear moment matching is given by Eq. (6.24), with  $\boldsymbol{\nu}(\mathbf{x}_r^v(t))$  as solution of the Sylvester-like PDE (7.10) and  $\widetilde{\mathbf{W}}(\mathbf{x}_r)^\top = \partial \boldsymbol{\omega}(\cdot) / \partial \mathbf{x}|_{\mathbf{x}=\boldsymbol{\nu}(\mathbf{x}_r)}$  arbitrary but such that  $\det(\widetilde{\mathbf{W}}_{\mathbf{x}_r}^\top \mathbf{E} \widetilde{\mathbf{V}}_{\mathbf{x}_r}) \neq 0$ . Herein, the ROM is parametrized in  $\widetilde{\mathbf{W}}_{\mathbf{x}_r} \in \mathbb{R}^{n \times r}$ , which can be chosen to impose certain properties on the reduced model (e.g. by a Galerkin projection with  $\widetilde{\mathbf{W}}_{\mathbf{x}_r} = \widetilde{\mathbf{V}}_{\mathbf{x}_r}$ ) or to achieve a better approximation using the concept of output nonlinear moment matching [15, 126, 127].

The *non-projective* approach consists in parametrizing the family of ROMs w.r.t. the reduced input matrix function  $\boldsymbol{\Delta}(\mathbf{x}_r) : \mathbb{R}^r \rightarrow \mathbb{R}^{r \times m}$ , yielding

$$\dot{\mathbf{x}}_r(t) = \mathbf{s}_v(\mathbf{x}_r(t)) - \boldsymbol{\Delta}(\mathbf{x}_r(t)) \mathbf{r}(\mathbf{x}_r(t)) + \boldsymbol{\Delta}(\mathbf{x}_r(t)) \mathbf{u}(t), \quad (7.11a)$$

$$\mathbf{y}_r(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r(t))), \quad (7.11b)$$

where  $\widetilde{\mathbf{E}}_r = \mathbf{I}_r$ ,  $\mathbf{a}_r(\mathbf{x}_r(t)) = \mathbf{s}_v(\mathbf{x}_r(t)) - \boldsymbol{\Delta}(\mathbf{x}_r(t)) \mathbf{r}(\mathbf{x}_r(t))$  and  $\mathbf{B}_r(\mathbf{x}_r(t)) = \boldsymbol{\Delta}(\mathbf{x}_r(t))$ . Note that this represents an *input-affine* nonlinear ROM that has a similar structure as the one in (3.90). The free mapping  $\boldsymbol{\Delta}(\mathbf{x}_r)$  can then be selected to enforce e.g. asymptotic stability, a prescribed relative degree and zero dynamics, or a passivity constraint, but should satisfy a certain partial differential equation [14], [229, Sec. 2.3]. Particularly interesting is the family of ROMs that is obtained, when a linear signal generator with  $\mathbf{s}_v(\mathbf{x}_r(t)) = \mathbf{S}_v \mathbf{x}_r(t)$  and  $\mathbf{r}(\mathbf{x}_r(t)) = \mathbf{R} \mathbf{x}_r(t)$  is employed:

$$\dot{\mathbf{x}}_r(t) = (\mathbf{S}_v - \boldsymbol{\Delta}(\mathbf{x}_r(t)) \mathbf{R}) \mathbf{x}_r(t) + \boldsymbol{\Delta}(\mathbf{x}_r(t)) \mathbf{u}(t), \quad (7.12a)$$

$$\mathbf{y}_r(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{x}_r(t))). \quad (7.12b)$$

If the free mapping is also chosen constant/state-independent, i.e.  $\boldsymbol{\Delta}(\mathbf{x}_r) = \boldsymbol{\Delta}$ , then the family of ROMs achieving nonlinear moment matching is described by a *linear* differential equation with a nonlinear output map. [229, Sec. 2.4.3]

## 7.2 Approximated nonlinear moment matching

The approach for nonlinear moment matching described in Section 7.1 requires the solution  $\boldsymbol{\nu}(\mathbf{x}_r^v(t))$  of the nonlinear, state-dependent PDE (7.10) for a given signal generator, in order to reduce the FOM (6.1). This e.g. involves either symbolic computations, or the numerical solution of a resulting system of ordinary differential equations (ODEs) after reduced state-

space discretization of the PDE (7.10). Since we aim to reduce large-scale nonlinear systems, almost only *numerical* methods come into consideration, which preferably should also avoid an expensive simulation. Hence, some step-by-step simplifications are performed in the following towards a practical, simulation-free method for nonlinear moment matching, which relies on solving *algebraic* nonlinear systems of equations rather than a PDE. Note that the proposed simplifications constitute an *approximation technique* for the solution of the PDE (7.10). Therefore, our practical simplifications yield an *approximated* nonlinear moment matching approach that matches *approximated* nonlinear moments. The simplifications are also given in the papers [72] and [71].

### 7.2.1 Simplifications

The proposed simplifications are threefold: (i) the use of a linear projection, (ii) the column-wise consideration of the equation and (iii) a time discretization with time-snapshots. All simplifications are practically motivated, try to avoid a nonlinear projection and to approximate the PDE (7.10). In the following we explain the simplifications for three different signal generator cases.

#### 7.2.1.1 Nonlinear signal generator

**(i) Linear projection** Motivated by the fact that nonlinear projections are complicated and more involved, whereas linear ones are often successfully employed even in nonlinear MOR, we propose to apply a linear projection  $\mathbf{x}(t) = \boldsymbol{\nu}(\mathbf{x}_r^v(t)) = \mathbf{V}\mathbf{x}_r^v(t)$  instead of the nonlinear projection mapping  $\boldsymbol{\nu}(\mathbf{x}_r^v(t))$ .

By doing so, the PDE (7.10) becomes the following *algebraic* nonlinear system of equations (NLSE)

$$\mathbf{0} = \mathbf{f}(\mathbf{V}\mathbf{x}_r^v(t), \mathbf{r}(\mathbf{x}_r^v(t))) - \mathbf{E}\mathbf{V}\mathbf{s}_v(\mathbf{x}_r^v(t)), \quad (7.13)$$

where the triple  $(\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v)$  is user-defined and the projection matrix  $\mathbf{V} \in \mathbb{R}^{n \times r}$  is the sought solution.

**(ii) Column-wise consideration** System (7.13) consists of  $n$  equations for  $n \cdot r$  unknowns in  $\mathbf{V} \in \mathbb{R}^{n \times r}$ , i.e. it is underdetermined. This shortcoming is a consequence of the usage of a linear projection instead of a nonlinear mapping on a manifold. To overcome this problem, we propose to consider the equation column-wise for each  $\mathbf{v}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, r$

$$\mathbf{0} = \mathbf{f}(\mathbf{v}_i \mathbf{x}_{r,i}^v(t), \mathbf{r}_i(\mathbf{x}_{r,i}^v(t))) - \mathbf{E}\mathbf{v}_i \mathbf{s}_{v_i}(\mathbf{x}_{r,i}^v(t)), \quad (7.14)$$

with  $\mathbf{x}_{r,i}^v(t) \in \mathbb{R}$ ,  $\mathbf{r}_i(\mathbf{x}_{r,i}^v(t)): \mathbb{R} \rightarrow \mathbb{R}^m$ ,  $\mathbf{s}_{v_i}(\mathbf{x}_{r,i}^v(t)): \mathbb{R} \rightarrow \mathbb{R}$  and  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r]$ . Please bear in mind that, in the linear setting, a column-wise construction of the orthogonal basis  $\mathbf{V}$  using the Arnoldi process still fulfills the Sylvester matrix equation (3.62a). In the nonlinear setting, however, this does not hold true anymore, since equation (7.13) is generally not satisfied, even if each column  $\mathbf{v}_i$  fulfills (7.14). This limitation will be further discussed in Section 7.3.

**(iii) Time discretization** In the linear case (cf. (3.89)), the state vector  $\mathbf{x}_r^v(t)$  could be factored out, yielding a constant linear matrix equation. Unfortunately, in the nonlinear

setting (cf. (7.10)),  $\mathbf{x}_r^v(t)$  can generally not be factored out anymore. Consequently, the nonlinear equation (7.14) is still state-dependent. For this reason, we propose to discretize the state-dependent equation with *time-snapshots*  $\{t_k^*\}$ ,  $k = 1, \dots, K$ , similar as in POD.

For a time-discretized nonlinear signal generator  $s_{v_i}(x_{r,i}^v(t_k^*))$ ,  $\mathbf{r}_i(x_{r,i}^v(t_k^*))$  and  $x_{r,0,i}^v$ , the following state-independent equation results

$$\mathbf{0} = \mathbf{f}(\mathbf{v}_{ik} x_{r,i}^v(t_k^*), \mathbf{r}_i(x_{r,i}^v(t_k^*))) - \mathbf{E} \mathbf{v}_{ik} s_{v_i}(x_{r,i}^v(t_k^*)), \quad (7.15)$$

which can be solved for each  $\mathbf{v}_{ik} \in \mathbb{R}^n$ , with  $i = 1, \dots, r$  and  $k = 1, \dots, K$ , if desired. Note that the discrete solution  $x_{r,i}^v(t_k^*)$  of the nonlinear signal generator ODE (7.1a) must be given or computed via simulation before solving equation (7.15).

### 7.2.1.2 Linear signal generator

Motivated from the linear case, one may also come to the idea of interconnecting the nonlinear system (6.1) with the linear signal generator (3.78), where  $\mathbf{s}_v(x_r^v(t)) = \mathbf{S}_v x_r^v(t)$  and  $\mathbf{r}(x_r^v(t)) = \mathbf{R} x_r^v(t)$  with  $\mathbf{S}_v \in \mathbb{C}^{r \times r}$ ,  $\mathbf{R} \in \mathbb{R}^{m \times r}$ .

**(i) Linear projection** By doing so, equation (7.13) becomes

$$\mathbf{0} = \mathbf{f}(\mathbf{V} x_r^v(t), \mathbf{R} x_r^v(t)) - \mathbf{E} \mathbf{V} \mathbf{S}_v x_r^v(t), \quad (7.16)$$

where the triple  $(\mathbf{S}_v, \mathbf{R}, x_{r,0}^v)$  is user-defined. Remember that the usage of a linear signal generator corresponds to exciting the nonlinear system with exponential input signals  $\mathbf{u}(t) = \mathbf{R} x_r^v(t) = \mathbf{R} e^{\mathbf{S}_v t} x_{r,0}^v$ . This choice naturally raises the question whether (growing) exponential inputs are sufficiently valid for characterizing nonlinear systems. Note that the dynamics of the selected signal generator represent the dynamics of the nonlinear system for which the steady-state responses are matched. Therefore, the signal generator should ideally be chosen such that it excites and characterizes the important dynamics of the nonlinear system. It is well known that exponential functions are the characterizing *eigenfunctions* for linear systems. By exciting the nonlinear system with exponential input signals, we therefore hope to describe the nonlinear dynamics adequately as well (cf. Section 8.2).

**(ii) Column-wise consideration** Considering the underdetermined equation again column-wise delivers

$$\mathbf{0} = \mathbf{f}\left(\mathbf{v}_i x_{r,i}^v(t), \underbrace{\mathbf{r}_i x_{r,i}^v(t)}_{\mathbf{r}_i(x_{r,i}^v(t))}\right) - \mathbf{E} \mathbf{v}_i \underbrace{\sigma_i x_{r,i}^v(t)}_{s_{v_i}(x_{r,i}^v(t))}, \quad (7.17)$$

where the signal generator (3.78) becomes  $\dot{x}_{r,i}^v(t) = \sigma_i x_{r,i}^v(t)$  and  $\mathbf{u}_i(t) = \mathbf{r}_i x_{r,i}^v(t)$  with  $x_{r,i}^v(t) = e^{\sigma_i t} x_{r,0,i}^v$  for  $i = 1, \dots, r$ .

**(iii) Time discretization** Using the time-discretized signal generator  $\hat{x}_{r,i}^v(t_k^*) = \sigma_i x_{r,i}^v(t_k^*)$ ,  $\mathbf{u}_i(t_k^*) = \mathbf{r}_i x_{r,i}^v(t_k^*)$  and  $x_{r,0,i}^v$ , equation (7.17) becomes state-independent

$$\mathbf{0} = \mathbf{f}(\mathbf{v}_{ik} x_{r,i}^v(t_k^*), \mathbf{r}_i x_{r,i}^v(t_k^*)) - \mathbf{E} \mathbf{v}_{ik} \sigma_i x_{r,i}^v(t_k^*), \quad (7.18)$$

with  $x_{r,i}^v(t_k^*) = e^{\sigma_i t_k^*} x_{r,0,i}^v$  for  $i = 1, \dots, r$ . Note that in this case, the discrete solution  $x_{r,i}^v(t_k^*)$  of the linear signal generator ODE (3.78a) is analytically given by exponential functions with exponents  $\sigma_i$ , so that no simulation of the signal generator is required.

### 7.2.1.3 Zero signal generator

This special (linear) signal generator is defined as  $\dot{\mathbf{x}}_r^v(t) = \mathbf{s}_v(\mathbf{x}_r^v(t)) = \mathbf{0}$ , which means that  $\mathbf{x}_r^v(t) = \mathbf{x}_{r,0}^v = \text{const}$  and  $\mathbf{u}(t) = \mathbf{R}\mathbf{x}_r^v(t) = \mathbf{R}\mathbf{x}_{r,0}^v = \text{const}$ . Hence, the usage of a zero signal generator is equivalent to exciting the nonlinear system with a constant input signal.

**(i) Linear projection** In this particular case, equation (7.13) becomes

$$\mathbf{0} = \mathbf{f}(\mathbf{V}\mathbf{x}_{r,0}^v, \mathbf{R}\mathbf{x}_{r,0}^v), \quad (7.19)$$

which is a nonlinear, *state-independent* system of equations.

**(ii) Column-wise consideration** A column-wise consideration of the underdetermined equation yields

$$\mathbf{0} = \mathbf{f}\left(\mathbf{v}_i x_{r,0,i}^v, \overbrace{\mathbf{r}_i x_{r,0,i}^v}^{r_i(x_{r,0,i}^v)}\right), \quad (7.20)$$

where  $\dot{x}_{r,i}^v(t) = 0$  with  $\sigma_i = 0$ ,  $\mathbf{u}_i(t) = \mathbf{r}_i x_{r,0,i}^v = \text{const}$  and  $x_{r,i}^v(t) = x_{r,0,i}^v = \text{const}$  hold for  $i = 1, \dots, r$ . In other words, the employment of a zero signal generator corresponds to moment matching at shifts  $\sigma_i = 0$ .

**(iii) Time discretization** For this special case, no time discretization is needed, since (7.20) already represents a state-independent equation. Note that solving the nonlinear system of equations (7.20) is strongly related to computing the *steady-state*  $\mathbf{x}_\infty$ , also called *equilibrium point*  $\mathbf{x}_{\text{eq}}$ , of the nonlinear system (6.1) by means of  $\mathbf{0} = \mathbf{f}(\mathbf{x}_\infty, \mathbf{u}_{\text{const}})$ .

## 7.2.2 Simulation-free nonlinear moment matching algorithm

### 7.2.2.1 Proposed algorithm

After the step-by-step simplifications discussed in the previous section, we are now ready to state our proposed simulation-free nonlinear moment matching (NLMM) algorithm 7.1.

Note that the algorithm is given for the most general case of a nonlinear signal generator (cf. Eq. (7.15)), and where *two* nested **for**-loops are used to compute all possible  $\mathbf{v}_{ik} \in \mathbb{R}^n$ . Nevertheless, other (simpler) strategies are also conceivable. These and further aspects are discussed in the following.

### 7.2.2.2 Computational aspects

**a) Different strategies, degrees of freedom and special cases** In addition to a nonlinear signal generator, one could also apply a linear or a zero signal generator. To this end, line 3 (and correspondingly line 4 also) in Algorithm 7.1 should be replaced by the equations

**Algorithm 7.1** Nonlinear Moment Matching (NLMM)

**Input:**  $\mathbf{E}$ ,  $\mathbf{f}(\mathbf{x}, \mathbf{u})$ ,  $\mathbf{A}(\mathbf{x}, \mathbf{u})$ ,  $x_{r,i}^v(t_k^*)$ ,  $\dot{x}_{r,i}^v(t_k^*) = s_{v_i}(x_{r,i}^v(t_k^*))$ ,  $\mathbf{r}_i(x_{r,i}^v(t_k^*))$ , initial guesses  $\mathbf{v}_{0,ik}$ , deflated order  $r_{\text{def}}$

**Output:** orthogonal basis  $\mathbf{V}$

```

1: for i = 1 : r do      ▶ e.g. r different shifts  $\sigma_i$ 
2:   for k = 1 : K do  ▶ e.g. K samples in each shift
3:     fun=@(v) f(v x_{r,ik}^v, r_i(x_{r,ik}^v)) - E v s_{v_i}(x_{r,ik}^v)      ▶ residual (7.15)
4:     Jfun=@(v) A(v x_{r,ik}^v, r_i(x_{r,ik}^v)) x_{r,ik}^v - E s_{v_i}(x_{r,ik}^v)  ▶ Jacobian of residual
5:     v_{ik} = NewtonRaphson(fun, v_{0,ik}, Jfun)      ▶ call Alg. 6.2
6:     V(:, (i-1)*K+k) ← v_{ik}
7:     V = gramSchmidt(v_{ik}, V)      ▶ optional
8: [U, Sigma, ~] = svd(V, 'econ'); V = U(:, 1:r_{def})  ▶ deflation is optional

```

(7.18) and (7.20). Note again that the latter cases do not require the simulation of the signal generator ODE to compute  $x_{r,i}^v(t_k^*)$ . Moreover, remember the importance of the choice of an adequate signal generator for a suitable characterization and reduction of the nonlinear system at hand.

Besides the depicted most general approach, where basis vectors are computed for different signal generators at several time points ( $i=1, \dots, r$ ,  $k=1, \dots, K$ ), one could also pursue other strategies. For instance, a single signal generator at several time points ( $i=1$ ,  $k=1, \dots, K$ ) is a possible simpler approach. Herein, the choice of appropriate time-snapshots  $t_k^*$  of the selected signal generator is of crucial importance. Another procedure consists in matching moments for different signal generators at only one time-snapshot ( $i=1, \dots, r$ ,  $K=1$ ). This multipoint moment matching strategy implies, exemplarily for a linear signal generator, the choice of different shifts, tangential directions and initial conditions  $\{\sigma_i, \mathbf{r}_i, x_{r,0,i}^v\}$ , which may be selected e.g. logarithmically between  $[\omega_{\min}, \omega_{\max}]$  or via IRKA [97] applied to the linearized system. For a zero signal generator this strategy implies the choice of different initial conditions and tangential directions  $\{x_{r,0,i}^v, \mathbf{r}_i\}$ . The selection of these degrees of freedom will be further discussed in Section 7.4.3.

If the NLMM algorithm is applied to a *linear* first-order system (3.1) with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$  using a linear signal generator, the algorithm boils down to the classical rational Krylov subspace method, since line 3 becomes

$$\mathbf{0} = \mathbf{A} \mathbf{v}_{ik} \cancel{e^{\sigma_i t_k^*} x_{r,0,i}^v} + \mathbf{B} \mathbf{r}_i \cancel{e^{\sigma_i t_k^*} x_{r,0,i}^v} - \mathbf{E} \mathbf{v}_{ik} \sigma_i \cancel{e^{\sigma_i t_k^*} x_{r,0,i}^v} \quad (7.21)$$

$$\Leftrightarrow (\sigma_i \mathbf{E} - \mathbf{A}) \mathbf{v}_i = \mathbf{B} \mathbf{r}_i.$$

In this special case, the reduction parameters condensate to  $\{s_{v_i}, \mathbf{r}_i, \cancel{x_{r,0,i}^v}, t_k^*\}$ , where the initial conditions and time-snapshots no longer play a role, but only the shifts and tangential directions matter. The above fact underlines the generalizability of the proposed algorithm: the linear Krylov case can be retrieved from the NLMM algorithm for a special system class and a specific choice of the reduction parameters.



**b) Computational effort** The presented reduction technique is *simulation-free*, since it does not require the numerical integration of the large-scale nonlinear system (6.1). However, the algorithm should rather be considered *simulation-lean*, since it still involves the solution of (at most  $r \cdot K$ ) nonlinear systems of equations (NLSE) of full order dimension  $n$ . These NLSEs can be solved using either a self-programmed Newton-Raphson scheme (cf. line 5) or the MATLAB's built-in function `fsolve`. For a faster computation of the Newton method, it is highly recommended to supply the analytical Jacobian of the right-hand side `Jfun`, for which the Jacobian  $\mathbf{A}(\mathbf{x}, \mathbf{u})$  of the nonlinearity  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  is needed. If `Jfun` is not provided, then the Jacobian is approximated using finite differences, which can be very time-consuming. One could even think of applying hyper-reduction to the nonlinear function or the Jacobian to further speed-up the Newton-Raphson scheme during NLMM.

Reduction techniques like POD require a forward numerical simulation (one/multistep, explicit/implicit, adaptive/fixed step-size) of the FOM to gather the snapshots. In case of an implicit scheme, the computational effort of POD compared to NLMM is supposed to be higher, since – within an implicit simulation – a NLSE must be solved in *each* time-step with the Newton-Raphson method (cf. Section 6.2). This will be further discussed in Section 7.4.4.

**c) Initial guesses and deflation** A good initial guess for the solution of a NLSE can considerably speed-up the convergence of the Newton method. Towards this aim, initial guesses can be taken (depending on the case) from the solution for a zero signal generator  $\mathbf{v}_{0,i} \leftarrow (7.20)$ , from linearized Krylov vectors  $\mathbf{v}_{0,i} = (\sigma_i \mathbf{E} - \mathbf{A}_{\text{eq}})^{-1} \mathbf{B}_{\text{eq}} \mathbf{r}_i$  computed with linearized matrices (6.12) or from the solutions at neighbouring shifts  $\mathbf{v}_{0,i+1} \leftarrow \mathbf{v}_{iK}$  or time-snapshots  $\mathbf{v}_{0,i,k+1} \leftarrow \mathbf{v}_{ik}$ .

Another important aspect is that the matrix  $\mathbf{V}$  that is used for projection must have full rank, and should preferably be orthogonal for better numerical robustness. Thus, if too many or redundant basis vectors  $\mathbf{v}_{ik}$  are available, a deflation via a rank-revealing QR factorization (RRQR) or a singular value decomposition (SVD) should be performed (cf. line 8) to truncate linearly dependent columns and obtain a full rank, orthogonal matrix. Alternatively, a modified Gram-Schmidt orthogonalization process or QR decomposition can optionally be employed to orthogonalize a full rank basis  $\mathbf{V}$  (cf. line 7).

### 7.2.2.3 Approximated nonlinear moments

After the simplifications, the important question arises what moments are actually being matched when applying Algorithm 7.1. In fact, since the Sylvester-like PDE is not being solved, the “true” nonlinear moments  $\mathbf{m}_0(s_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v)$  at  $\{s_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v\}$  from Lemma 7.1 are not being exactly matched. Instead, we are *approximately* matching these nonlinear moments at the chosen interpolation data  $\{s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*\}$ . To enlighten this more, we discuss the approximated nonlinear moments in the following.

**Lemma 7.2** (Approximated nonlinear moments). The 0-th approximated nonlinear moments  $\mathbf{m}_0(s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*)$  at  $\{s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*\}$  are related to the (locally well-defined) steady-state response

$$\mathbf{y}_{\text{ss},i}(t_k^*) = \mathbf{h}(\mathbf{v}_{ik} x_{r,i}^v(t_k^*)) := \mathbf{m}_0(s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*) \quad (7.22)$$

or rather

$$\mathbf{y}_{\text{ss}}(t_k^*) = \mathbf{h}\left(\sum_{i=1}^r \mathbf{v}_{ik} x_{r,i}^v(t_k^*)\right), \quad \text{or} \quad \mathbf{y}_{\text{ss}}(t) = \mathbf{h}(\mathbf{V} \mathbf{x}_r^v(t)) = \mathbf{h}\left(\sum_{i=1}^r \sum_{k=1}^K \mathbf{v}_{ik} x_{r,i}^v(t_k^*)\right) \quad (7.23)$$

of the interconnection of (6.1) with the signal generator  $\{s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*\}$ . The projection matrix  $\mathbf{V} = [\mathbf{v}_{11}, \dots, \mathbf{v}_{1K}, \dots, \mathbf{v}_{r1}, \dots, \mathbf{v}_{rK}]$  is composed of the vectors  $\mathbf{v}_{ik}$ , i.e. the solutions of (7.15) for  $i=1, \dots, r, k=1, \dots, K$ . The reduced coordinates are given by  $\mathbf{x}_r^v(t) = [x_{r,1}^v(t_1^*), \dots, x_{r,1}^v(t_K^*), \dots, x_{r,r}^v(t_1^*), \dots, x_{r,r}^v(t_K^*)]^\top$ .

Please note again that the ansatz for  $x_{r,i}^v(t_k^*)$  is explicitly given by  $x_{r,i}^v(t_k^*) = e^{\sigma_i t_k^*} x_{r,0,i}^v$  (cf. (3.83)) when matching moments  $\mathbf{m}_0(\sigma_i x_{r,i}^v(t_k^*), \mathbf{r}_i x_{r,i}^v(t_k^*), x_{r,0,i}^v, t_k^*)$  with a linear signal generator, and by  $x_{r,i}^v(t) = x_{r,0,i}^v = \text{const}$  when matching moments  $\mathbf{m}_0(0, \mathbf{r}_i x_{r,0,i}^v, x_{r,0,i}^v)$  with a zero signal generator.

### 7.2.3 Families of reduced models achieving approximated moment matching

The approach that we will mainly employ to construct a family of ROMs achieving approximated (input) nonlinear moment matching is based on the two-sided linear projection given in (6.9), with  $\mathbf{V}$  stemming from the NLMM algorithm and  $\mathbf{W}$  arbitrary but such that  $\det(\mathbf{W}^\top \mathbf{E} \mathbf{V}) \neq 0$ . Herein, the ROM is parametrized in  $\mathbf{W} \in \mathbb{R}^{n \times r}$ . This matrix can be selected to impose stability, passivity, etc. on the reduced model (e.g. by an orthogonal projection with  $\mathbf{W} = \mathbf{V}$ ) or can be used to achieve a better approximation (e.g. by matching more moments using the output NLMM algorithm [70]).

The *non-projective* approach described in Section 7.1.4 is also conceivable to construct ROMs achieving *approximated* nonlinear moment matching in a projection-free manner. The families of ROMs are given by (7.11) and (7.12), with the output  $\mathbf{y}_r(t) = \mathbf{h}(\mathbf{V} \mathbf{x}_r(t))$  and  $\mathbf{V}$  stemming from the NLMM algorithm. Although these families are appealing in terms of their simple structure (especially (7.12) for  $\Delta(\mathbf{x}_r) = \Delta$ ) and consequent avoidance of hyper-reduction, we will not employ them in this thesis. The concrete exploitation of the free parameters  $\mathbf{W}$  or  $\Delta(\mathbf{x}_r)$ , as well as the investigation of the suitability of the projection-free families in comparison to the projection-based family (6.9) is topic of future research.

## 7.3 Analysis, discussion and limitations

In this section, a discussion about the proposed simplifications and the presented simulation-lean algorithm is given. Different important aspects are broken down in respective subsections, which are analyzed in the following.

### Adequate selection of the projection ansatz

As mentioned in Section 6.3.1, a linear projection resembles a special case of the most general nonlinear projection or the polynomial expansion-based ansatz (6.28) proposed in [124, Ch. 4]. In fact, applying a more sophisticated projection like e.g. a quadratic manifold (6.29) (see e.g. [132, 224, 73]) or a series expansion ansatz with basis functions customized for the nonlinear system at hand (cf. [146, 124, 231]) could be superior and even indispensable in

certain cases. This approach should also allow to reduce a nonlinear system to a minimum reduced order due to the state-dependent, adaptive nature of the projection. However, from a numerical perspective, nonlinear projections are much more difficult to handle than linear ones (cf. Section 6.6). For instance, they yield a rather complicated ROM (6.24), with the state-dependent tangent basis  $\widetilde{\mathbf{V}}(\mathbf{x}_r)$ , and also an emerging convective term in the second-order case [132, 224, 75] (see Ch. 9). Consequently, the (implicit) time integration scheme becomes more involved due to the changing  $\widetilde{\mathbf{V}}(\mathbf{x}_r)$ , which might possibly get rank deficient (see Sec. 6.4.2). On the contrary, linear projections are simple and frequently employed even in nonlinear MOR. They have proven to be successful for several applications and acceptable for many types of nonlinearities, as long as the reduction basis  $\mathbf{V}$  contains sufficient nonlinear information (using e.g. an augmented basis). Precisely for that reason this approach normally leads to a higher reduced order in comparison to a nonlinear projection [132, 224].

### Appropriate choice of the signal generator

The choice of the signal generator determines (1) the ansatz for the dynamics  $\mathbf{x}_r^v(t)$ ,  $\dot{\mathbf{x}}_r^v(t)$  and (2) the exciting input of the system. Thus, the signal generator should be chosen such that (1)  $\mathbf{x}_r^v(t)$  constitutes *representative eigen-/ansatz functions* of the underlying nonlinear system and (2)  $\mathbf{u}(t)$  represents a typical operating input which excites the important dynamics.

Although the validity of a linear signal generator for characterizing general nonlinear systems is questionable, note that this type of signal generator (where complex exponentials serve as ansatz functions) is being implicitly used for the reduction of bilinear and quadratic-bilinear systems (see [221, 59]). Complex exponentials are also being employed in the Fourier approximation ansatz of the Harmonic Balance Method [190, 201, 142]. We therefore believe that linear signal generators might be adequate to reduce nonlinear systems in a proper way.

Note that the linear signal generator (3.78) constitutes a special case of the expansion-based ansatz

$$\dot{\mathbf{x}}_r^v = \sum_{k=1}^N \mathbf{S}_v^{(k)} \mathbf{x}_r^{v(k)} = \mathbf{S}_v^{(1)} \mathbf{x}_r^v + \mathbf{S}_v^{(2)} (\mathbf{x}_r^v \otimes \mathbf{x}_r^v) + \dots, \quad (7.24a)$$

$$\mathbf{u} = \sum_{k=1}^N \mathbf{R}^{(k)} \mathbf{x}_r^{v(k)} = \mathbf{R}^{(1)} \mathbf{x}_r^v + \mathbf{R}^{(2)} (\mathbf{x}_r^v \otimes \mathbf{x}_r^v) + \dots, \quad (7.24b)$$

with  $\mathbf{S}_v^{(k)} \in \mathbb{C}^{r \times r^k}$ ,  $\mathbf{R}^{(k)} \in \mathbb{C}^{m \times r^k}$  and  $\mathbf{x}_r^{v(k)} \in \mathbb{R}^{r^k}$ . Indeed, such a signal generator ansatz could be customized for the nonlinear system at hand and, naturally, be combined with the power series ansatz (6.28), in order to approximately solve the PDE (7.10). [124, Ch. 4]

### Obtaining a state-independent matrix equation

In the Sylvester-like PDE (7.10) the state vector  $\mathbf{x}_r^v(t)$  cannot be factored out so easily than in (3.89). In fact, the key to obtain a *constant/state-independent* matrix equation of dimension  $n \times r$  from (7.10) lies on both the choice of an adequate projection ansatz (e.g. (6.28)) and an appropriate signal generator (e.g. (7.24)), tailored for the nonlinear system at hand. Interestingly, the state-independent matrix Sylvester equations used in bilinear/quadratic-bilinear MOR (see [92, 45, 103]) can indeed be derived from (7.10) by using the Volterra series representation with a linear projection and a linear signal generator. This interesting

aspect is further discussed in Section 8.4. First results concerning the connection between Krylov/Sylvester-based moment matching for bilinear systems and the general nonlinear moment matching framework from Astolfi are published in [59].

Note that if the factorization of  $\mathbf{x}_r^v(t)$  works out and a state-independent matrix equation is obtained, then the other two simplifications steps – namely (ii) and (iii) – are no longer needed. Therefore, the choice of an adequate projection and signal generator that allow to factor out  $\mathbf{x}_r^v(t)$  is highly recommended, but is obviously a “true art” in itself and strongly problem-dependent.

### Limitations of the column-wise consideration

If a linear projection is applied and the factorization of  $\mathbf{x}_r^v(t)$  does not succeed, then, lamentably, the *underdetermined* system (7.13) is obtained. The proposed column-wise consideration (7.14) with  $x_{r,i}^v(t) \in \mathbb{R}$ ,  $s_{v_i}(x_{r,i}^v(t)) : \mathbb{R} \rightarrow \mathbb{R}$ ,  $\mathbf{r}_i(x_{r,i}^v(t)) : \mathbb{R} \rightarrow \mathbb{R}^m$  instead of  $\mathbf{x}_r^v(t) \in \mathbb{R}^r$ ,  $\mathbf{s}_v(\mathbf{x}_r^v(t)) : \mathbb{R}^r \rightarrow \mathbb{R}^r$ ,  $\mathbf{r}(\mathbf{x}_r^v(t)) : \mathbb{R}^r \rightarrow \mathbb{R}^m$  has the limitation that (7.13) is generally not fulfilled, since the *couplings* in  $\mathbf{V}\mathbf{x}_r^v(t)$ ,  $\mathbf{V}\mathbf{s}_v(\mathbf{x}_r^v(t))$  and  $\mathbf{r}(\mathbf{x}_r^v(t))$  are not being considered. Hence, the column-wise consideration constitutes a further simplification/approximation step, which is naturally restricting the universality of the original nonlinear moment matching framework from Astolfi.

We are currently working on possible ways to numerically solve the underdetermined systems (7.13), (7.16) and (7.19). One possibility could be to consider

$$\mathbf{0} = \underbrace{\left[ \mathbf{rhs}(\mathbf{V}, \mathbf{x}_r^v(t_1^*)), \dots, \mathbf{rhs}(\mathbf{V}, \mathbf{x}_r^v(t_K^*)) \right]}_{\mathbf{Rhs}(\mathbf{V})}, \quad (7.25)$$

with the discretized equation (7.13)

$$\mathbf{0} = \underbrace{\mathbf{f}(\mathbf{V}\mathbf{x}_r^v(t_k^*), \mathbf{r}(\mathbf{x}_r^v(t_k^*))) - \mathbf{E}\mathbf{V}\mathbf{s}_v(\mathbf{x}_r^v(t_k^*))}_{\mathbf{rhs}(\mathbf{V}, \mathbf{x}_r^v(t_k^*))}, \quad (7.26)$$

where  $\mathbf{Rhs}(\mathbf{V}) : \mathbb{R}^{n \times r} \rightarrow \mathbb{R}^{n \times K}$  and  $\mathbf{rhs}(\mathbf{V}, \mathbf{x}_r^v(t_k^*)) : \mathbb{R}^{n \times r} \times \mathbb{R}^r \rightarrow \mathbb{R}^n$ . The solution of the nonlinear matrix equation (7.25) could e.g. be computed via Newton’s method or a similar fixed-point iteration

$$\mathbf{V}_{n+1} = \mathbf{V}_n - \left( \frac{\partial \mathbf{Rhs}(\mathbf{V})}{\partial \mathbf{V}} \Big|_{\mathbf{V}_n} \right)^+ \mathbf{Rhs}(\mathbf{V}_n), \quad (7.27)$$

for which the non-trivial derivative  $\partial \mathbf{Rhs}(\mathbf{V}) / \partial \mathbf{V}$  is required. The arising linear systems of equations (LSEs) can then be solved using pseudoinverse, least-squares or direct approaches.

### Neutral stability of the signal generator

In [14], the signal generator is assumed to be neutrally stable (see fn. 1), so that the steady-state of the nonlinear system is well-defined. In other words: the input signal should – on the one hand – be persistently exciting (i.e. cannot decay to zero), but should – on the other hand – be bounded, so that the system does not show a diverging steady-state. In linear MOR, however, Assumption 2 can be relaxed meaning that shifts  $\sigma_i \in \mathbb{C} \setminus \lambda(\mathbf{E}^{-1}\mathbf{A})$  lying to the left,

on or to the right of the imaginary axis can be used. Indeed, due to the Meier-Luenberger conditions known from  $\mathcal{H}_2$ -optimal reduction [179, 97], the shifts are often chosen on the right half-plane ( $\sigma_i \in \mathbb{C}_+$ ) in order to hopefully obtain a stable ROM. This choice effectively means that the system is excited by *growing* exponentials, and that we are using basis vectors  $\mathbf{v}_i$  which point into the direction of this growing steady-state solution (cf. (3.83)).

We believe that, within the NLMM algorithm, the nonlinear system can also be excited by a stable, neutrally stable or even an unstable generator, as long as the considered time interval for the signal generator  $t_k^* \in [t_0^{\text{SG}}, t_{\text{end}}^{\text{SG}}]$  is (naturally) well chosen, but *limited*. In this way, the assumed reduced coordinates  $x_{r,i}^v(t_k^*)$  in (7.22) may become large but not unfeasible, the exciting input is bounded and the steady-state response is also limited. Further note that a well-defined solution  $\boldsymbol{\nu}(\mathbf{x}_r^v)$  to the PDE (7.10) may exist even for signal generators not satisfying the neutral stability assumption. Thus, moments can also be determined for generators producing diverging or decaying trajectories, although the relation with the steady-state response cannot be established anymore [229, Sec. 2.2].

### Conceptual comparison to POD

Looking closer at the Sylvester-like PDE (7.10) with  $\{\mathbf{s}_v(\mathbf{x}_r^v(t)), \mathbf{r}(\mathbf{x}_r^v(t)), \mathbf{x}_{r,0}^v\}$  or at the simplified equation (7.15) with  $\{s_{v_i}(x_{r,i}^v(t_k^*)), \mathbf{r}_i(x_{r,i}^v(t_k^*)), x_{r,0,i}^v, t_k^*\}$ , one can detect the conceptual similarities and differences between POD and the nonlinear moment matching approach.

Similar to POD, in the nonlinear moment matching framework training input signals are also being chosen. In fact, the training inputs for NLMM are determined by the choice of the signal generator:  $\mathbf{u}_{\text{train}}^{\text{NLMM}}(t) = \mathbf{r}(\mathbf{x}_r^v(t))$  or rather  $\mathbf{u}_{\text{train},i}^{\text{NLMM}}(t_k^*) = \mathbf{r}_i(x_{r,i}^v(t_k^*))$ . However, while in NLMM the discrete solutions  $x_{r,i}^v(t_k^*)$  are guessed or intuited by the user, in POD they depend on the simulated snapshots according to  $x_{r,i}^v(t_k) = \mathbf{v}_i^\top \mathbf{x}(t_k)$  (cf. Eq. (6.18)). In other words: POD extracts the main characteristics of the *simulated* solution manifold in form of *empirical eigenfunctions*, whereas NLMM employs the ansatz functions  $x_{r,i}^v(t_k^*)$  that result from the selected signal generators.

This fundamental difference serves to characterize both methods in the following way: POD is based on discrete samples of the solution of the FOM for typical operating input signals (aka. *training data*) and does not necessarily require a-priori theoretic knowledge about the system. Thus, it falls under the category of “simulation-based”, “data-driven” or “black-box” methods. As opposed to that, NLMM relies on *assumed* or *guessed* eigenfunctions  $x_{r,i}^v(t_k^*)$  determined by the chosen signal generator. Therefore, it requires more expert knowledge/intuition about the system. Consequently, the algorithm could be categorized in the class of “simulation-lean” (since it relies on the solution of less NLSEs), “system-theoretic” (since it is based on the concepts of steady-state response and center manifold theory) and “grey/white-box” methods (since it requires a-priori knowledge to perform well). Finally note that – as a member of the Krylov family – NLMM does not provide an error bound, and does not guarantee stability. The latter issue, however, can be tackled using a Galerkin projection or the projection-free families of ROMs discussed in Section 7.2.3.

## 7.4 Guidelines for practitioners

In this section, we give some practical guidelines on how to use the NLMM algorithm. The aim is to help practitioners with this new MOR technique and its degrees of freedom, so that they can apply it to their own application.

### 7.4.1 Training vs. test phase

Similar as POD, NLMM can be decomposed in a training and test phase.

During the *training* or *offline* phase, the reduction bases  $\mathbf{V}_{\text{POD}}$  and  $\mathbf{V}_{\text{NLMM}}$  are computed using one or several training input signals. In case of NLMM, the training signals  $\mathbf{u}_{\text{train},i}^{\text{NLMM}}(t_k^*)$  are determined by the chosen signal generators. The training signals for POD  $\mathbf{u}_{\text{train},i}^{\text{POD}}(t_k)$  could be chosen the same as for NLMM (for comparison reasons), or differently.

During the *test* or *online* phase, the FOM and the ROMs are evaluated with a test signal  $\mathbf{u}_{\text{test}}(t)$ . This could be chosen the same as for the training phase. However, it is absolutely mandatory to select  $\mathbf{u}_{\text{train}}(t) \neq \mathbf{u}_{\text{test}}(t)$  as well, in order to assess the approximation quality of the ROMs for signals that were not trained. If the same input signal is applied for both training and test phase, then the error is expected to be smaller than using different signals.

### 7.4.2 Examples for signal generators

In the following, some examples for useful signal generators are given. The list is by no means complete, but should rather serve as a guide and inspiration for interested readers who want to apply NLMM to their own model.

#### Cosinusoids/Sinusoids

To generate a cosine function  $x_r^v(t) = A \cdot \cos(\omega t)$  with desired amplitude  $A$  and angular frequency  $\omega$ , use

$$\dot{x}_r^v(t) = -A \cdot \sin(\omega t) \cdot \omega, \quad x_r^v(0) = A. \quad (7.28)$$

For a sine function  $x_r^v(t) = A \cdot \sin(\omega t)$ , use

$$\dot{x}_r^v(t) = A \cdot \cos(\omega t) \cdot \omega, \quad x_r^v(0) = 0. \quad (7.29)$$

Note that the initial condition  $x_r^v(0)$  influences the signal generator. For instance, the choice

$$\dot{x}_r^v(t) = -A \cdot \sin(\omega t) \cdot \omega, \quad x_r^v(0) = x_{r,0}^v \quad (7.30)$$

yields the modified signal  $x_r^v(t) = A \cdot \cos(\omega t) - A + x_{r,0}^v$ . Similarly, the choice

$$\dot{x}_r^v(t) = A \cdot \cos(\omega t) \cdot \omega, \quad x_r^v(0) = x_{r,0}^v \quad (7.31)$$

leads to the sine  $x_r^v(t) = A \cdot \sin(\omega t) + x_{r,0}^v$ , which is shifted along the  $y$ -axis.

#### Squared cosinusoids/sinusoids

To generate a squared cosine signal  $x_r^v(t) = A \cdot \cos^2(\omega t) = A \frac{1+\cos(2\omega t)}{2}$ , use the following signal generator

$$\dot{x}_r^v(t) = -2A \cdot \cos(\omega t) \cdot \sin(\omega t) \cdot \omega, \quad x_r^v(0) = A. \quad (7.32)$$

Similarly, for a squared sine signal  $x_r^v(t) = A \cdot \sin^2(\omega t) = A \frac{1 - \cos(2\omega t)}{2}$ , Eq. (7.29) should be adapted accordingly. Note again that the initial condition  $x_r^v(0)$  can be used to shift the signals along the  $y$ -axis.

### Exponential function

An exponential signal of the form  $x_r^v(t) = e^{\sigma t} x_{r,0}^v$  can be generated by the signal generator

$$\dot{x}_r^v(t) = \sigma \cdot e^{\sigma t} x_{r,0}^v, \quad x_r^v(0) = x_{r,0}^v. \quad (7.33)$$

The initial condition  $x_{r,0}^v \in \mathbb{R}$  specifies the amplitude scaling, whereas the shift  $\sigma = \delta \mp i\omega \in \mathbb{C}$  determines the characteristic behavior of the exponential function.

**Zero shift** ( $\sigma = 0$ ,  $\delta = 0$ ,  $\omega = 0$ ) As mentioned in Section 7.2.1.3, the choice of  $\sigma = 0$  yields a constant signal  $x_r^v(t) = x_{r,0}^v = \text{const}$  with amplitude  $x_{r,0}^v$ .

**Pure real shift** ( $\sigma = \delta$ ,  $\delta \neq 0$ ,  $\omega = 0$ ) If  $\delta < 0$ , then  $x_r^v(t) = e^{\delta t} x_{r,0}^v$  describes a decaying exponential, whereas for  $\delta > 0$  (e.g.  $\sigma = 10$ )  $x_r^v(t)$  represents a growing exponential.

**Pure imaginary shifts** ( $\sigma_{1/2} = \mp i\omega$ ,  $\delta = 0$ ,  $\omega \neq 0$ ) The signal  $x_{r,1}^v(t) = e^{-i\omega t} x_{r,0,1}^v = (\cos(\omega t) - i \sin(\omega t)) x_{r,0,1}^v$  describes a permanent oscillation. It is highly recommended to also use the complex conjugated pair, i.e. the signal  $x_{r,2}^v(t) = e^{+i\omega t} x_{r,0,2}^v$  with  $x_{r,0,1}^v = x_{r,0,2}^v$ . This allows to split the computed basis vectors  $\mathbf{v}_1 = \overline{\mathbf{v}_2} \in \mathbb{C}^n$  in real and imaginary part, in order to obtain a real projection matrix spanning the same subspace as its complex counterpart (cf. Example 7.1). Note that a real-valued projection matrix  $\mathbf{V}$  can also be obtained by using the cosinusoids/sinusoids generators (7.28) and (7.29), where complex arithmetic is avoided.

**Complex shifts** ( $\sigma_{1/2} = \delta \mp i\omega$ ,  $\delta \neq 0$ ,  $\omega \neq 0$ ) If  $\delta < 0$ , then  $x_{r,1/2}^v(t) = e^{\delta t} (\cos(\omega t) \mp i \sin(\omega t)) x_{r,0,1/2}^v$  describes a decaying oscillation, whereas for  $\delta > 0$  (e.g.  $\sigma_{1/2} = 10 \mp 3i$ )  $x_r^v(t)$  represents a growing oscillation. The use of the complex conjugated pair is again essential to obtain a real-valued projection matrix. This is illustrated in the following.

*Example 7.1* (Complex shifts in NLMM). Assume that NLMM is applied using a complex conjugate pair of shifts  $\sigma_{1,2} \in \mathbb{C}$  with  $\sigma_1 = \overline{\sigma_2}$ , a complex conjugate pair of tangential directions  $\mathbf{r}_{1,2} \in \mathbb{C}^m$  with  $\mathbf{r}_1 = \overline{\mathbf{r}_2}$  and equal initial conditions  $x_{r,0,1}^v = x_{r,0,2}^v \in \mathbb{R}$ . Then, the corresponding directions computed from

$$\mathbf{f}(\mathbf{v}_{1k} e^{\sigma_1 t_k^*} x_{r,0,1}^v, \mathbf{r}_1 e^{\sigma_1 t_k^*} x_{r,0,1}^v) - \mathbf{E} \mathbf{v}_{1k} \sigma_1 e^{\sigma_1 t_k^*} x_{r,0,1}^v = \mathbf{0} \quad (7.34a)$$

$$\mathbf{f}(\mathbf{v}_{2k} e^{\sigma_2 t_k^*} x_{r,0,2}^v, \mathbf{r}_2 e^{\sigma_2 t_k^*} x_{r,0,2}^v) - \mathbf{E} \mathbf{v}_{2k} \sigma_2 e^{\sigma_2 t_k^*} x_{r,0,2}^v = \mathbf{0} \quad (7.34b)$$

are also complex conjugated to each other  $\mathbf{v}_{1k} = \overline{\mathbf{v}_{2k}} \in \mathbb{C}^n$ , provided that  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  and  $\mathbf{E}$  are real-valued and the initial guesses are complex conjugated  $\mathbf{v}_{0,1k} = \overline{\mathbf{v}_{0,2k}} \in \mathbb{C}^n$ . Thus, if the interpolation data is chosen such that  $\mathbf{v}_{1k} = \overline{\mathbf{v}_{2k}}$  can be expected, then *only one* complex NLSE (7.34) needs to be solved, since  $\mathbf{v}_{2k}$  can be determined by conjugating  $\mathbf{v}_{1k}$ .

What is more, this property allows to construct real-valued directions  $\tilde{\mathbf{v}}_{1k} = \text{Re}(\mathbf{v}_{1k})$  and  $\tilde{\mathbf{v}}_{2k} = \text{Im}(\mathbf{v}_{1k})$  in order to obtain a real basis  $\tilde{\mathbf{V}}$  (cf. linear case in Example 3.1).  $\Delta$

### Shifted exponential function

The following signal generator ODE

$$\dot{x}_r^v(t) = \sigma \cdot (x_r^v(t) + \alpha), \quad x_r^v(0) = x_{r,0}^v \quad (7.35)$$

can be analytically solved by the method of *variation of parameters*, leading to the solution

$$x_r^v(t) = e^{\sigma t} x_{r,0}^v + \alpha(e^{\sigma t} - 1) = e^{\sigma t} \underbrace{(x_{r,0}^v + \alpha)}_{\beta} - \alpha. \quad (7.36)$$

This represents an exponential function with amplitude scaling  $\beta$ , which is shifted along the  $y$ -axis by  $\alpha$ . These two parameters can be exploited to generate a signal  $x_r^v(t)$  that covers the known or supposed value range of interest.

### Ramp/Linear function

To generate a linear function  $x_r^v(t) = \alpha \cdot t + x_{r,0}^v$  of desired slope  $\alpha$  and  $y$ -intercept  $x_{r,0}^v$ , the constant signal generator ODE  $\dot{x}_r^v(t) = \alpha$  with  $x_r^v(0) = x_{r,0}^v$  should be used. Similar considerations apply for polynomial functions.

## 7.4.3 Selection of shifts and time-snapshots

### Shifts, tangential directions and $x_{r,0,i}^v$

As a first attempt, we recommend practitioners to apply exponential functions as exciting training signals for NLMM (cf. Section 7.4.2). The question is, however, how to appropriately choose the shifts, tangential directions and initial conditions  $\{\sigma_i, \mathbf{r}_i, x_{r,0,i}^v\}$  that parametrize those signals.

The most straightforward approach is to select the shifts at zero. Then, the user has to choose different initial conditions and tangential directions  $\{x_{r,0,i}^v, \mathbf{r}_i\}$ . The initial conditions  $x_{r,0,i}^v$  should represent physically meaningful amplitudes for (1) the resulting ansatz  $x_{r,i}^v(t) = x_{r,0,i}^v$  and (2) the systems' inputs  $\mathbf{u}_i(t) = \mathbf{r}_i x_{r,0,i}^v$ , covering e.g. the admissible control input range. The tangential directions can be chosen to amplify the initial conditions, to favor certain input-output paths, or simply as  $\mathbf{r}_i = \mathbf{1}_m \in \mathbb{R}^m$  if no prior system understanding is available.

If zero signal generators are not sufficient, then shifts  $\sigma_i \neq 0$  may be taken into account. Preferably, one should exploit some system knowledge, and pre-analyze the value range that the signals  $x_{r,i}^v(t) = e^{\sigma_i t} x_{r,0,i}^v$  and  $\mathbf{u}_i(t) = \mathbf{r}_i x_{r,i}^v(t)$  would cover, when choosing  $\{\sigma_i, \mathbf{r}_i, x_{r,0,i}^v\}$ . Apart from that, one could choose the mirror images of some eigenvalues of the linearized FOM as expansion points, i.e. `s0=-eigs(sys, r).'` or logarithmically spread the shifts over the interesting frequency range  $[\omega_{\min}, \omega_{\max}]$ , e.g. `s0Real=logspace(log10(wmin), log10(wmax), r)` or `s0=cplxpair([1i*s0Real, -1i*s0Real])`. Furthermore, one could also apply IRKA to the linearized FOM and then take the optimal shifts and tangential directions (together with



some  $x_{r,0,i}^v$ ) for the NLMM algorithm. Other shift selection procedures known from linear MOR [39, 151], such as the heuristic Penzl method, the Wachspress approach or exploiting the Ritz values, are also conceivable within NLMM.

Depending on the model, it might be reasonable to combine zero signal generators with pure real and/or complex shifts. In this way, different characteristic behaviors (constant signal, real exponential, periodic oscillation) can be easily assumed and captured.

### Time-snapshots

Except for zero signal generators, time-snapshots  $\{t_k^*\}, k=1, \dots, K$  are needed to obtain and solve the state-independent equations (7.15) and (7.18).

In general, the user should first select an appropriate time interval  $[t_0^{\text{SG}}, t_{\text{end}}^{\text{SG}}]$  for the signal generator (SG). Then, the choice of time-snapshots basically depends on how rapidly the exponential function (or the chosen training signal) changes. One possibility is to exploit the form of the chosen training signal by selectively placing the time points  $t_k^*$  at crucial time instants or deliberately choosing more snapshots at dynamic regions. A second simpler technique is to select an appropriate number of samples  $K_i$  for each signal generator and then equidistantly place the snapshots in the selected time interval, e.g. `ti=linspace(t0,tEnd,Ki)`. Another, rather black-box, approach is to simulate the signal generators with an adaptive step-size solver (e.g. MATLAB's built-in `ode45`), which autonomously selects more points at dynamic regions and less snapshots when the dynamics do not change significantly.

Note that the NLMM algorithm would return (almost) the same basis vectors  $v_{ik}$ , if several time-snapshots are placed at static or similar regions of the training signals. Even though a deflation would help to truncate linearly dependent columns a-posteriori, it is highly recommended – if possible – to *a-priori* avoid the solution of redundant NLSEs in order to keep the basis computation process as efficient as possible.

#### 7.4.4 Time integration scheme

The comparison of NLMM and POD in terms of computational effort highly depends on the time integration scheme selected to train and gather the FOM snapshots. As already discussed in Section 6.2, there exist many time integration schemes (one/multistep, explicit/implicit and variable/fixed step solvers) that are more or less suitable depending on the application.

In this thesis, we employ the fixed step implicit Euler scheme 6.1 for the FOM training simulation within POD and also for the test phase of FOM and ROMs. The `implicitEuler` scheme requires to solve a NLSE in *each* time-step with the Newton-Raphson method (cf. line 5). Hence, the computational effort of POD is expected to be higher than of NLMM, where less NLSEs have to be solved.

In our opinion, the choice of a fixed step backward Euler as time integration scheme is reasonable. Firstly, it is an implicit solver, which is essential to simulate stiff nonlinear ODEs usually arising from the discretization of partial differential equations. Secondly, it is a basic one-step method, which is simple, but still reliable and often already integrated in many simulation tools. Nevertheless, practitioners are reminded that the comparisons between NLMM and POD could also be attempted with an explicit solver (like `ode45`) or a more sophisticated implicit scheme (like `ode15s`). Depending on the model and the chosen parameters, it then could happen that POD requires less computational time than NLMM.

## 7.5 Numerical examples

In this section, the efficiency of the proposed simulation-lean NLMM algorithm is illustrated by means of two numerical examples: the Chafee-Infante equation and the FitzHugh-Nagumo system. Both models result from the semidiscretization of nonlinear PDEs and have often been considered as benchmarks for model reduction. We compare NLMM with two other reduction techniques, namely a pure linear basis computed with rational Krylov (RK) from the sssMOR toolbox<sup>2</sup>, and POD (cf. Section 6.3.3). Some more details concerning the numerical examples are given in the following:

1. All simulations were done on a board with an Intel<sup>®</sup> Core<sup>™</sup> i7-8700 CPU @ 3.20 GHz clock speed, 6 cores, 12 threads and 31.8 GB RAM using MATLAB<sup>™</sup> Version 9.5.0.10-49112 (R2018b) Update 3 on a Microsoft<sup>®</sup> Windows<sup>®</sup> 10 Pro x64-bit operating system.
2. The FOM and ROMs are integrated by the routine `implicitEuler`, using a `NewtonRaphson` scheme with a relative error tolerance of  $10^{-3}$  and an absolute error tolerance of  $10^{-6}$ . Although we only report here the simulation times obtained with backward Euler, note that simulations were also run with MATLAB's `ode45` and `ode15s` solvers with the same tolerances.
3. For a fair comparison, the above tolerances were also employed within the `NewtonRaphson` scheme required for NLMM. Note that, for other models or depending on the initial guesses  $\mathbf{v}_{0,ik}$ , it sometimes might be necessary to increase these tolerances to achieve convergence within a reasonable number of iterations.
4. All ROMs are obtained via orthogonal (Galerkin) projection, i.e. by (6.9) with  $\mathbf{W} = \mathbf{V}$ . Further note that all reduction bases  $\mathbf{V}_{\text{RK}}$ ,  $\mathbf{V}_{\text{POD}}$  and  $\mathbf{V}_{\text{NLMM}}$  are made orthogonal, i.e.  $\mathbf{V}^T \mathbf{V} = \mathbf{I}_r$  holds.
5. From a control engineering perspective, the *input-output behavior* of the FOM is particularly important. Thus, the approximation error is quantified w.r.t. the outputs  $\mathbf{y}(t)$  of interest in time-domain. We employ a *point-wise* in time relative error measure

$$e_{y,\text{rel},(\cdot)}^m(t) = \frac{\|\mathbf{y}(t) - \mathbf{y}_r^m(t)\|_{(\cdot)}}{\|\mathbf{y}(t)\|_{(\cdot)}},$$

calculated for different reduction methods  $m \in \{\text{RK}, \text{POD}, \text{NLMM}, \dots\}$  using a desired vector norm  $(\cdot) \in \{1, 2, \infty, \dots\}$ . Furthermore, we use a *norm-wise* relative error measure

$$e_{y,\text{rel},\mathcal{L}_p}^m = \frac{\|\mathbf{y} - \mathbf{y}_r^m\|_{\mathcal{L}_p}}{\|\mathbf{y}\|_{\mathcal{L}_p}},$$

applying a desired signal norm  $\mathcal{L}_p \in \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_\infty, \dots\}$  (cf. Section 2.6).

The numerical results reported in the following only represent an extract of the wide variety of scenarios that can be studied. To ensure reproducibility, the parameters and results of the selected scenarios are listed in tables. Furthermore, the implemented source code is provided as supplemental material<sup>3</sup> to promote transparency and reusability. We encourage users to study their own scenarios, run simulations for other test signals and reduced orders, and compare the results to `ode15s` and further error measures. The material also includes scripts

<sup>2</sup>The sss and sssMOR toolboxes can be downloaded under <https://github.com/MORLab>.

<sup>3</sup>The MATLAB implementation of NLMM, together with the whole software framework (`implicitEuler`, `NewtonRaphson`, templates, etc.), is available under <https://doi.org/10.5281/zenodo.3542641>.

for other benchmarks, such as the nonlinear RC-Ladder and the viscous Burgers' equation, as well as some signal generator templates that can serve as an inspiration.

### 7.5.1 Chafee-Infante equation

First, we consider the one-dimensional Chafee-Infante (aka. Allen-Cahn) equation [67]. This nonlinear reaction-diffusion equation describes the process of phase separation, thus appearing in multiphase fluid dynamics, chemical processes and biological applications. The governing parabolic PDE has a cubic nonlinearity

$$\frac{\partial v}{\partial t}(x, t) - \frac{\partial^2 v}{\partial x^2}(x, t) = \lambda v(x, t) - \lambda v^3(x, t),$$

and is subject to the initial and boundary conditions

$$\begin{aligned} v(x, 0) &= 0, & x &\in (0, L), \\ v(0, t) &= u(t), & \frac{\partial v}{\partial x}(L, t) &= 0, & t &\in (0, T). \end{aligned}$$

The bifurcation parameter is fixed to  $\lambda = 1$ . For our evaluations, we set the length of the spatial domain to  $L = 1$ . The spatial discretization via finite differences with  $\ell$  grid points yields  $n = \ell$ . The model equation is given by (6.1), with  $\mathbf{E} = \mathbf{I}$ ,  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}\mathbf{x} - \tilde{\mathbf{f}}(\mathbf{x}) + \mathbf{b}\mathbf{u}$ ,  $h(\mathbf{x}) = \mathbf{c}^\top \mathbf{x}$  and  $\mathbf{x} = \mathbf{v}$ . The control input  $u(t)$  is applied to the left boundary ( $x=0$ ), whereas the output is measured at the right boundary ( $x=L$ ), i.e.  $\mathbf{c}^\top = [0, \dots, 0, 1]$ .

The parameters of the selected scenarios are summarized in Tab. 7.1. First, the linearization point  $(u_{\text{eq}}, \mathbf{x}_{\text{eq}}) = (1, \mathbf{1}_n)$  and a vector of shifts are chosen for the application of rational Krylov. For POD, we collect 501 snapshots of the simulated solution for the training input  $u_{\text{train}}^{\text{POD}}(t) = 5 \cos(t)$ . In the first NLMM scenario, zero signal generators with amplitudes between  $[-2, 2]$  are employed. These constant signals are amplified by the factor 25, in order to generate input signals covering the range  $[-50, 50]$ . In NLMM<sub>2</sub>, a combination of a zero signal generator with the two exponential functions  $x_{r,2}(t) = -2e^{-t}$  and  $x_{r,3}(t) = 2e^{-t}$  is used. These real exponentials share some shape similarities with the eigenfunctions of the linearized Chafee-Infante equation [268, Fig. 1] and are therefore selected. For the test phase, two different sine signals covering the ranges  $[-25, 25]$  and  $[0, 50]$  are chosen to assess the performance of the ROMs for different input amplitudes.

The numerical results for both test signals are illustrated in Fig. 7.2 and Fig. 7.3. Moreover, the approximation errors are listed in Tab. 7.2. We can observe that NLMM<sub>2</sub> performs the best in both cases. This can be explained by (1) the deliberate choice of the initial conditions  $x_{r,0,i}^v$  and  $r_i$  to cover the interested input value range and (2) by the curve shape of the selected exponential functions. Remarkably, POD still yields satisfactory results for  $u_{\text{test},1}(t)$ , although it was only trained with input amplitudes in the interval  $[-5, 5]$ . However, for the higher amplitudes of  $u_{\text{test},2}(t)$ , it has problems in approximating the maximum value. RK shares this same difficulty, which is a consequence of the large deviation from the stationary input  $u_{\text{eq}} = 1$ . If a good approximation is desired for a wider range of input amplitudes, then several linearization points and RK-reductions would be needed (cf. TPWL). Finally, it can be observed that NLMM<sub>1</sub> performs well in the steady-state regions, whereas is less accurate in the transition phases.

Table 7.1: Chafee-Infante: Parameters of selected scenarios

method	parameters
	$L=1, \ell=1000, n=1000, t \in [0, 10\text{s}], h=0.02\text{s}, r_{\text{defl}}=10$
RK	$u_{\text{eq}}=1, \mathbf{x}_{\text{eq}}$ guess: $x_{[1:n]}=1, \mathbf{s0}=\text{logspace}(-2,2,r_{\text{defl}})$
POD	$u_{\text{train}}^{\text{POD}}(t) = 5 \cos(t), t \in [0 : 0.02 : 10\text{s}] \leftrightarrow n_s = 501$
NLMM <sub>1</sub>	$x_{r,i}^v(t) = x_{r,0,i}^v, u_i(t) = r_i x_{r,0,i}^v, \sigma_i = \mathbf{zeros}(1, r_{\text{defl}}),$ $r_i = 25 * \mathbf{ones}(1, r_{\text{defl}}), x_{r,0,i}^v = \mathbf{linspace}(-2,2,r_{\text{defl}})$
NLMM <sub>2</sub>	$x_{r,i}^v(t) = e^{\sigma_i t} x_{r,0,i}^v, u_i(t) = r_i x_{r,i}^v(t), \sigma_i = [0, -1, -1],$ $r_i = [1, 25, 25], x_{r,0,i}^v = [1, -2, 2], \{-, \mathbf{linspace}(0,6,10), "\}$
Test	$u_{\text{test},1}(t) = 25 \sin(t), u_{\text{test},2}(t) = 25 (\sin(t) + 1)$

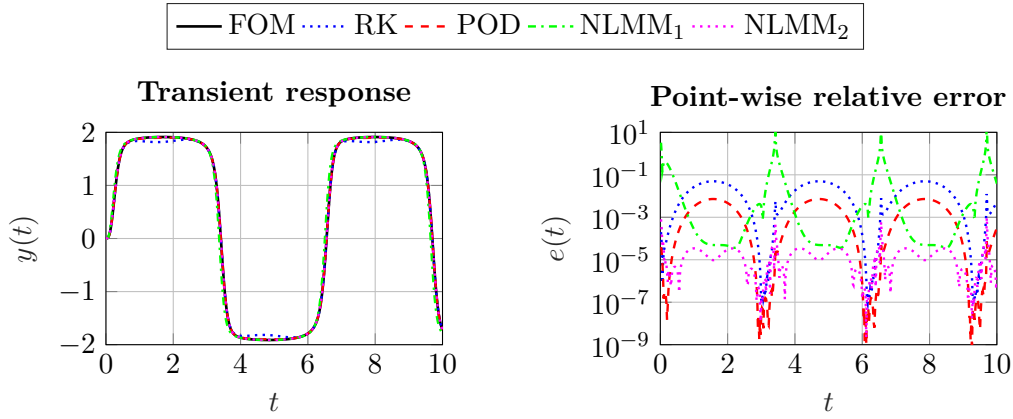
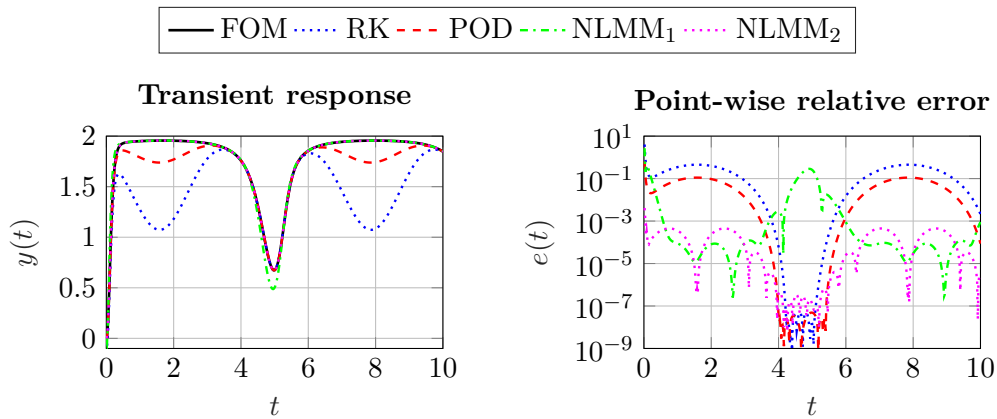
Figure 7.2: Chafee-Infante: Outputs and point-wise relative error  $e_{y,\text{rel},2}^m(t)$  for test signal  $u_{\text{test},1}(t) = 25 \sin(t)$ .Figure 7.3: Chafee-Infante: Outputs and point-wise relative error  $e_{y,\text{rel},2}^m(t)$  for test signal  $u_{\text{test},2}(t) = 25 (\sin(t) + 1)$ .

Table 7.2: Chafee-Infante: Approximation errors  $e_{y,\text{rel},\mathcal{L}_p}^m$ 

input	norm	RK	POD	NLMM <sub>1</sub>	NLMM <sub>2</sub>
$u_{\text{test},1}(t)$	$\mathcal{L}_1$	$2.09 \text{e}^{-2}$	$2.70 \text{e}^{-3}$	$3.45 \text{e}^{-2}$	$1.61 \text{e}^{-5}$
	$\mathcal{L}_2$	$2.86 \text{e}^{-2}$	$3.90 \text{e}^{-3}$	$8.14 \text{e}^{-2}$	$2.02 \text{e}^{-5}$
	$\mathcal{L}_\infty$	$4.96 \text{e}^{-2}$	$7.20 \text{e}^{-3}$	$2.80 \text{e}^{-1}$	$3.39 \text{e}^{-5}$
$u_{\text{test},2}(t)$	$\mathcal{L}_1$	$1.89 \text{e}^{-1}$	$4.31 \text{e}^{-2}$	$1.07 \text{e}^{-2}$	$1.62 \text{e}^{-4}$
	$\mathcal{L}_2$	$2.57 \text{e}^{-1}$	$6.09 \text{e}^{-2}$	$3.02 \text{e}^{-2}$	$2.37 \text{e}^{-4}$
	$\mathcal{L}_\infty$	$4.51 \text{e}^{-1}$	$1.11 \text{e}^{-1}$	$1.39 \text{e}^{-1}$	$4.45 \text{e}^{-4}$

The computation times for the offline and online phase are summarized in Tab. 7.3. The comparison shows that NLMM requires less time than POD to compute the corresponding reduction basis. Nevertheless, we can observe that the simulation of the ROMs in the online phase takes slightly longer than the simulation of the FOM. This indicates that dimensional reduction alone is not sufficient here, since the repetitive evaluation of the nonlinear terms within **implicitEuler** is still the major bottleneck. This could be alleviated by applying hyper-reduction. For comparison, **ode15s** needs around 0.15 s for the simulation of the FOM and approximately 0.08 s for the simulation of the ROMs. Hence, a slight speed-up is achieved when using this solver.

Table 7.3: Chafee-Infante: Computation times

method	reduction time (training/offline)	sim. time (test 1)	sim. time (test 2)
FOM	-	0.3626 s	0.3501 s
RK	0.0352 s	0.4793 s	0.4849 s
POD	0.3721 s (501 NLSEs)	0.4980 s	0.5280 s
NLMM <sub>1</sub>	0.0347 s (10 NLSEs)	0.5132 s	0.4562 s
NLMM <sub>2</sub>	0.0466 s (21 NLSEs)	0.5005 s	0.4673 s

### 7.5.2 FitzHugh-Nagumo model

The FitzHugh-Nagumo (FHN) model is a simplified version of the Hodgkin-Huxley model, which describes the activation and deactivation dynamics of a spiking neuron. It has been previously considered for POD-based [74] and quadratic-bilinear [45, 103] model reduction. The dynamics are governed by the following one-dimensional coupled PDE-ODE system

$$\begin{aligned}\varepsilon \frac{\partial v}{\partial t}(x, t) &= \varepsilon^2 \frac{\partial^2 v}{\partial x^2}(x, t) + f(v(x, t)) - w(x, t) + g, \\ \frac{\partial w}{\partial t}(x, t) &= R v(x, t) - \gamma w(x, t) + g,\end{aligned}$$

with the cubic nonlinearity  $f(v) = v(v - 0.1)(1 - v)$ . The initial and boundary conditions are given by

$$\begin{aligned} v(x, 0) &= 0, & w(x, 0) &= 0, & x &\in [0, L], \\ \frac{\partial v}{\partial x}(0, t) &= -i_0(t), & \frac{\partial v}{\partial x}(L, t) &= 0, & t &\geq 0. \end{aligned}$$

The parameters of the model are set to  $\varepsilon = 0.015$ ,  $R = 0.5$ ,  $\gamma = 2$ ,  $g = 0.05$ ,  $L = 1$ . A spatial discretization using a finite difference scheme with  $\ell$  nodes yields  $n = 2\ell$  degrees of freedom. The model equation is given by (6.1), with  $\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}\mathbf{x} + \tilde{\mathbf{f}}(\mathbf{x}) + \mathbf{B}\mathbf{u}$ ,  $\mathbf{h}(\mathbf{x}) = \mathbf{C}\mathbf{x}$  and  $\mathbf{x} = [\mathbf{v}^\top, \mathbf{w}^\top]^\top$ . The state variables  $v_\ell$  and  $w_\ell$  represent the voltage and recovery voltage at each node. The model is input-affine with  $\mathbf{u}(t) = [i_0(t), 1]^\top$ , where  $u_1(t) = i_0(t)$  denotes the electric current excitation at the left boundary ( $x = 0$ ) and the second input  $u_2(t) = 1$  comes from the constant value  $g$ . The outputs are chosen at the left boundary via the output matrix  $\mathbf{C}$ , i.e.  $\mathbf{y}(t) = [v_1(t), w_1(t)]^\top$ . In this example  $\mathbf{E} \neq \mathbf{I}$ , but the matrix is still diagonal. Thus, it can be efficiently carried to the right-hand side by its inverse  $\mathbf{E}^{-1}$  to obtain an explicit representation with  $\mathbf{E} = \mathbf{I}$ . Note that, for systems with more general, regular  $\mathbf{E}$ , it is advisable to apply the reduction directly to the implicit state-space representation instead of using the inverse.

From basic neuroscience knowledge and reading, we expect the membrane voltage and recovery voltage to lie in the ranges  $v \in [-2, 2 \text{ V}]$  and  $w \in [-1, 1 \text{ V}]$ . Furthermore, depending on the amplitude of the external stimulus  $i_0(t)$ , different neuronal behaviors can occur: a resting behavior for negative to low amplitudes, spiking activity (i.e. limit cycle oscillations) when the stimulus  $i_0(t)$  exceeds a certain threshold, and blocking/saturated behavior for higher currents. Based on this, the reduction parameters for RK, POD and NLMM were chosen (cf. Tab. 7.4).

Table 7.4: FHN model: Parameters of selected scenarios

method	parameters
	$L = 1$ , $\ell = 1000$ , $n = 2000$ , $t \in [0, 8\text{s}]$ , $h = 0.02 \text{ s}$ , $r_{\text{def}} = 20$
RK	$i_{0,\text{eq}} = u_{\text{eq}} = 0.3$ , $\mathbf{x}_{\text{eq}}$ guess: $x_{[1:\ell]} = 0.1$ , $x_{[\ell+1:2\ell]} = 0.04$ , <code>s0=cplxpair([1i*s0Real,-1i*s0Real])</code> with $[0.01, 100 \frac{\text{rad}}{\text{s}}]$
POD	$i_{0,\text{train}}^{\text{POD}}(t) = 5 \cdot 10^4 t^3 e^{-15t}$ , $t \in [0 : 0.02 : 8\text{s}] \leftrightarrow n_s = 401$
NLMM <sub>1</sub>	$x_{r,i}^v(t) = e^{\sigma_i t} x_{r,0,i}^v$ , $i_{0,i}(t) = 1 x_{r,i}^v(t)$ , $\sigma_i = [0, 0, -i\frac{2\pi}{T}, i\frac{2\pi}{T}]$ , $T = 10 \text{ s}$ , $x_{r,0,i}^v = [-2, 2, -2, -2]$ , <code>{-, -, linspace(0,T,10), "}</code>
NLMM <sub>2</sub>	$x_r^v(t) = 10^4 t^3 e^{-15t} + x_{r,0}^v$ , $i_0(t) = 5(x_r^v(t) - x_{r,0}^v)$ , $x_{r,0}^v = -2$ , $K = 51$ time-snapshots in $t_k^* \in [0, 1\text{s}]$ , i.e. <code>linspace(0,1,51)</code>
Test	$i_{0,\text{test},1}(t) = 5 \cdot 10^4 t^3 e^{-15t}$ , $i_{0,\text{test},2}(t) = -300$

For the computation of the linearization point needed for RK, a stationary input current and an initial guess for  $\mathbf{x}_{\text{eq}}$  are selected, before the NLSE  $\mathbf{0} = \mathbf{f}(\mathbf{x}_{\text{eq}}, u_{\text{eq}})$  is solved. The computed linearization point corresponds in this case to an unstable equilibrium, where the

model exhibits periodic spiking oscillation. Since the frequency of oscillation is not known a-priori, the shifts are spread over the wide range  $[0.01, 100 \frac{\text{rad}}{\text{s}}]$ . In NLMM<sub>1</sub>, a combination of two constant signals and a  $y$ -shifted sine/cosine oscillation is employed as signal generators. Hereby, the time period  $T$  and the initial conditions  $x_{r,0,i}^v$  are selected by pre-analyzing the value range that the signals  $x_{r,i}^v(t)$  and  $\dot{x}_{r,i}^v(t)$  cover. This combination of zero signal generators with imaginary shifts is expected to capture both the resting/blocking behavior of the neuron, as well as the spiking activity. In NLMM<sub>2</sub>, the signal generator is chosen such that  $x_r^v(t)$  covers the value range  $[-2, 2\text{V}]$  of the state variables and the generated input  $i_0(t)$  equals the training input for POD. The time interval is limited to  $t_k^* \in [0, 1\text{s}]$ , since the signals do not significantly change as time increases, and more snapshots would only yield redundant basis vectors. For the test phase of FOM and ROMs, two different input signals are chosen: the first one corresponds to the training signal of POD, whereas the second one represents a constant negative current.

The numerical results for both test signals are illustrated in Fig. 7.4 and Fig. 7.5. Moreover, the approximation errors are listed in Tab. 7.5. As expected, we can observe that POD yields the best approximation for the test signal  $i_{0,\text{test},1}(t)$ , followed by NLMM<sub>2</sub>. NLMM<sub>1</sub> and RK can reproduce the basic spiking behavior, but are not as accurate as the previous ones. For  $i_{0,\text{test},2}(t)$  the neuron shows no oscillations, but rather a steady-state (resting) behavior. In this scenario, NLMM<sub>1</sub> performs the best, followed by NLMM<sub>2</sub>. POD is less accurate, since this constant resting information is not contained in the trained snapshot matrix. RK yields moderate results due to the chosen linearization point corresponding to the spiking behavior.

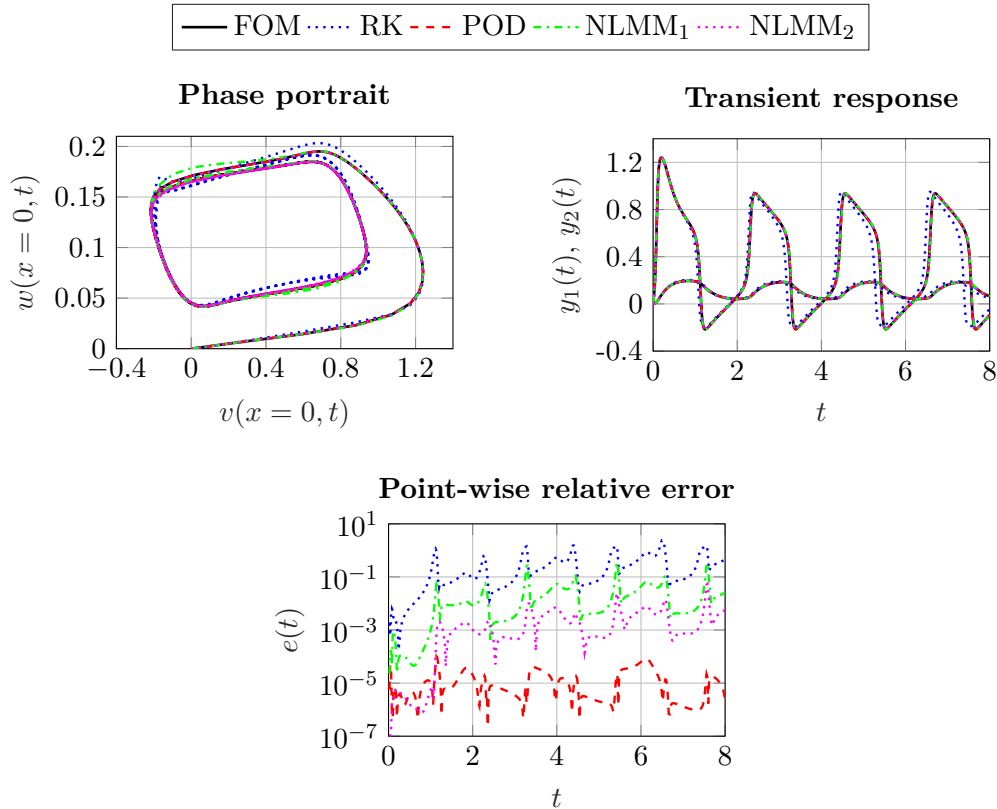


Figure 7.4: FHN model, spiking: Phase portrait, outputs and point-wise relative error  $e_{y,\text{rel},2}^m(t)$  for test signal  $i_{0,\text{test},1}(t) = 5 \cdot 10^4 t^3 e^{-15t}$ .

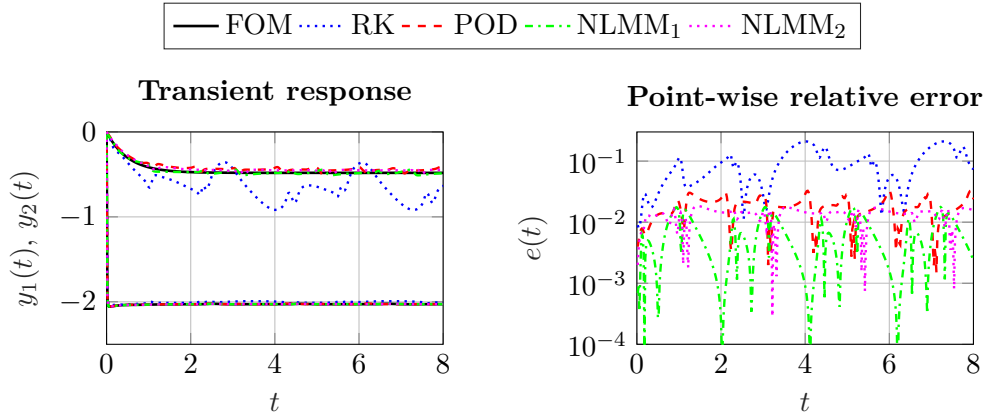


Figure 7.5: FHN model, resting: Outputs and point-wise relative error  $e_{y,\text{rel},2}^m(t)$  for test signal  $i_{0,\text{test},2}(t) = -300$ .

Table 7.5: FHN model: Approximation errors  $e_{y,\text{rel},\mathcal{L}_p}^m$

input	norm	RK	POD	NLMM <sub>1</sub>	NLMM <sub>2</sub>
$i_{0,\text{test},1}(t)$	$\mathcal{L}_1$	$2.03 \text{ e}^{-1}$	$6.67 \text{ e}^{-6}$	$1.34 \text{ e}^{-2}$	$1.50 \text{ e}^{-3}$
	$\mathcal{L}_2$	$3.01 \text{ e}^{-1}$	$7.10 \text{ e}^{-6}$	$2.22 \text{ e}^{-2}$	$2.40 \text{ e}^{-3}$
	$\mathcal{L}_\infty$	$5.88 \text{ e}^{-1}$	$1.38 \text{ e}^{-5}$	$5.39 \text{ e}^{-2}$	$8.80 \text{ e}^{-3}$
$i_{0,\text{test},2}(t)$	$\mathcal{L}_1$	$9.26 \text{ e}^{-2}$	$1.78 \text{ e}^{-2}$	$6.20 \text{ e}^{-3}$	$1.33 \text{ e}^{-2}$
	$\mathcal{L}_2$	$1.09 \text{ e}^{-1}$	$1.91 \text{ e}^{-2}$	$7.90 \text{ e}^{-3}$	$1.37 \text{ e}^{-2}$
	$\mathcal{L}_\infty$	$2.10 \text{ e}^{-1}$	$3.27 \text{ e}^{-2}$	$1.77 \text{ e}^{-2}$	$1.84 \text{ e}^{-2}$

The comparison of the reduction methods in terms of computational effort is given in Tab. 7.6. NLMM requires less time than POD to compute the corresponding reduction basis due to the lower number of NLSEs that have to be solved. Note, however, that the simulation of the FOM with `ode15s` only needs around 12s for test 1 and 17s for test 2. Using this sophisticated solver, POD needs less computational time than NLMM. Remember, though, that several full training simulations would be required during POD to capture the resting, spiking and blocking behavior of the neuron. In NLMM, different dynamic behaviors can be *deliberately* captured without having to perform (long) transient simulations. Regarding the online phase, the speed-up gained through dimensional reduction becomes evident and underlines the advantage of MOR.

Table 7.6: FHN model: Computation times

method	reduction time (training/offline)	sim. time (test 1)	sim. time (test 2)
FOM	-	127.41 s	140.22 s
RK	0.90 s	10.09 s	10.06 s
POD	127.50 s (401 NLSEs)	9.82 s	11.10 s
NLMM <sub>1</sub>	13.14 s (12 NLSEs)	9.74 s	10.43 s
NLMM <sub>2</sub>	32.19 s (51 NLSEs)	9.75 s	10.52 s



## 7.6 Further remarks

In this section some remarks concerning the applicability of NLMM to finite element models and its combination with simulation-free hyper-reduction are given.

### 7.6.1 Applicability to large finite element models

The performance of NLMM has been demonstrated in Section 7.5 by rather “toy”/academic examples. However, the theory discussed here has been recently transferred to nonlinear second-order systems in [75]. Therein, the algorithm has been applied to two geometrically nonlinear structural models, using AMfe [224] as well as Gmsh [104] and ParaView [5]. Although the studied models have still an academic nature, they constitute *two-dimensional, finite element* models with more realistic geometry as the benchmarks employed here. Thus, they allowed for a first demonstration of the feasibility and potential of NLMM with the reduction of finite element models. This will be shown in Chapter 11.

The proposed NLMM algorithm can be considered a *semi non-intrusive* MOR approach. On the one hand, it only requires to *evaluate* the nonlinear function  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  and Jacobian  $\mathbf{A}(\mathbf{x}, \mathbf{u})$  within the Newton-Raphson scheme (cf. lines 3-5) to calculate the basis vectors. This means that the governing equations (6.1) are not required analytically, but merely need to be evaluated within the (commercial) FE software package to compute  $\mathbf{V}_{\text{NLMM}}$ . However, once the reduction basis is calculated, the governing equations are required to obtain the ROM by projection. Hence, classical projection-based MOR cannot be considered fully non-intrusive. Nevertheless, this latter shortcoming can be circumvented by combining NLMM with the non-intrusive identification of *reduced* polynomial tensors  $\mathbf{A}_{2r}$ ,  $\mathbf{A}_{3r}$ , etc. using e.g. [178, 210]. In this way, the reduction can be performed without requiring access to the model equations, but by directly inferring the reduced-order operators in a non-intrusive manner.

Based on the proofs of concept, together with the semi non-intrusive nature of NLMM, we believe that it can be successfully applied in the future to more complex FE applications (e.g. automotive, aeronautics, biomechanics, electromechanics, etc.). Herein, the presented guidelines as well as some system knowledge are crucial for the performance of the method.

### 7.6.2 Simulation-free hyper-reduction

As discussed in Chapter 6, one usually needs to combine dimensional reduction with hyper-reduction to gain higher computational savings during the online testing phase of the ROM. Since NLMM is a simulation-free (linear projection-based) dimensional reduction approach, it makes sense to combine it with a simulation-free hyper-reduction technique in order to completely avoid expensive training simulations.

First of all, NLMM can be combined with a polynomial representation of  $\mathbf{f}(\mathbf{x}, \mathbf{u})$ , or rather of  $\mathbf{f}_r(\mathbf{x}_r, \mathbf{u})$ , using (i) an analytical formulation of the tensors, (ii) finite differences or (iii) an identification of coefficients. The intrusive approach (i) is numerically very efficient, but requires access to the element formulation in the FE code for implementation. In contrast, the non-intrusive approaches (ii) and (iii) are computationally more expensive (i.e. yield higher offline costs), but do not require the full-order operators.

NLMM can also be combined with classical hyper-reduction methods like DEIM or ECSW. In Section 6.5, the common simulation-based and some simulation-free techniques to generate

snapshots for hyper-reduction are explained. In this sense, NLMM could be combined with (1) the standard simulation-based generation of snapshots or (2) the lean procedure based on nonlinear static problems [219]. A third option could be to construct the full-order snapshots  $\mathbf{x}(t_k^*)$  by using the assumed ansatz functions  $x_{r,i}^v(t_k^*)$  from the signal generator as follows:

$$\mathbf{x}(t_k^*) \approx \sum_{i=1}^r \mathbf{v}_{ik} x_{r,i}^v(t_k^*), \quad t_k^* \in [t_0^{\text{SG}}, t_{\text{end}}^{\text{SG}}]. \quad (7.37)$$

Herein, the basis vectors  $\mathbf{v}_{ik}$  are calculated with the NLMM algorithm, while the assumed functions  $x_{r,i}^v(t_k^*)$  play the role of the unknown reduced coordinates  $x_{r,i}(t)$ . After generating  $\mathbf{x}(t_k^*) := \mathbf{x}_k$ , the function snapshot matrix  $\mathbf{F} = [\{\mathbf{f}(\mathbf{x}_k)\}_{k=1}^K]$  or the reduced state snapshots  $\mathbf{x}_{r,k} \leftarrow (6.45)$  can be calculated for DEIM or ECSW.

With this proposed approach, the choice of the characterizing functions  $x_{r,i}^v(t_k^*)$  becomes even more important, as it influences both the dimensional *and* hyper-reduction process. This simulation-free snapshot generation ansatz is only mentioned here as a conceptual idea. Its implementation and numerical efficiency is subject to future research.

## 7.7 Conclusions

In this chapter, the concept of moment matching known from the linear setting is first comprehensively explained for nonlinear systems based on [14]. Then, some simplifications are proposed to approximate the arising PDE, yielding a ready-to-implement, simulation-lean nonlinear moment matching algorithm which relies on the solution of NLSEs. An extensive discussion about the proposed simplifications, the numerical aspects and the degrees of freedom of the algorithm is presented, together with valuable guidelines for practitioners.

All in all, it can be concluded that the proposed algorithm for approximated nonlinear moment matching represents a particular instance of the most general framework from Astolfi, in the sense that a *linear projection* is employed. The resulting scheme requires the solution of some NLSEs, thus allowing for a numerically more efficient computation of the reduction basis (i.e. lower offline costs) in comparison with simulation-based approaches like POD. In addition, it can be concluded that the selection of the signal generator using prior system knowledge plays a crucial role for the performance of the algorithm. Nevertheless, the presented simplifications *and* guidelines provide a powerful tool (instead of a “finished product”), which can be customized for the nonlinear system at hand.

Recommendations for future research directions concern the matching of high-order nonlinear moments/Markov parameters based on [14], the development of an  $\mathcal{H}_2$ -optimal moment matching algorithm for general nonlinear systems (see e.g. [247]), as well as the comparison of our proposed NLMM algorithm with other invariant manifold approximation techniques, system-theoretic and/or data-driven methods. Moreover, the conceptual differences/connection between POD and NLMM should be studied in more detail. It might be possible that POD and our suggested approach in fact coincide for a certain choice of the time integration scheme. This should be investigated further using e.g. the links given in [195] and [29].

## Chapter 8

# Bridging the Gap between Polynomial and Nonlinear Model Reduction

In this chapter we want to establish connections between polynomial and nonlinear model reduction by moment matching. The aim is to offer a unifying view of both worlds.

We have seen that the growing exponential approach is a fundamental tool for analyzing polynomial systems. It is based on the eigenfunction property of *linear* (sub)systems. Motivated by this, we will derive the eigenfunctions of bilinear and quadratic-bilinear systems. Then, we will discuss how these eigenfunctions could be generated via a signal generator to use them within the nonlinear moment matching framework from Astolfi. Finally, we provide the time-domain/signal generator interpretation of Volterra series interpolation.

For simplicity, the presentation is restricted to SISO systems. A similar analysis is possible for the MIMO case as well.

### 8.1 Eigenfunctions of dynamical systems

The concept of eigenfunctions is well-known in mathematics.

Let us consider a linear operator  $\mathcal{A} : f \mapsto y$ , i.e.  $\mathcal{A}f = y$ . An *eigenfunction* of the operator  $\mathcal{A}$  is any function  $f$  such that – when applied to  $\mathcal{A}$  – is only scaled by some factor  $\lambda$  called *eigenvalue*. In other words:  $\mathcal{A}f = \lambda f$ . For example, let us assume that  $\mathcal{A} = \mathbf{A} \in \mathbb{R}^{n \times n}$  with  $\mathbf{A}\mathbf{x} = \mathbf{b}$ . The vector  $\mathbf{v}$  satisfying  $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$  is called eigenvector of the matrix  $\mathbf{A}$ .

The concept of eigenfunctions is also well-known in signals and systems theory.

Let us consider a dynamical system operator  $\mathcal{S}_{i/o} : u(t) \mapsto y(t)$  or  $\mathcal{S}_{i/s} : u(t) \mapsto \mathbf{x}(t)$ . In the context of LTI systems  $\mathcal{S}_{i/o}$  represents the convolution operator (cf. Eq. (3.2)). An eigenfunction of a dynamical system is a signal  $u(t)$  that – when input to the system – produces a response  $y(t) = \lambda u(t)$  with the eigenvalue  $\lambda$ , or  $\mathbf{x}(t) = \lambda u(t)$  with the scaling  $\lambda \in \mathbb{C}^{n \times 1}$ .

#### Eigenfunctions of linear time-invariant systems

It is well known that (complex) exponentials  $u(t) = e^{st}$  are the eigenfunctions of LTI systems, whereas the transfer function  $G(s) = \mathbf{c}^T (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b}$  represents the scaling factor  $\lambda$ :

$$y(t) = G(s) e^{st}, \quad \mathbf{x}(t) = \boldsymbol{\lambda}(s) e^{st} \quad \text{with} \quad G(s) = \mathbf{c}^T \boldsymbol{\lambda}(s), \quad \boldsymbol{\lambda}(s) = (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b}. \quad (8.1)$$

This means that the response of an LTI system to a complex exponential input  $e^{\sigma t}$  is again a complex exponential of the same frequency  $\sigma \in \mathbb{C}$  which is only scaled by  $G(\sigma)$ .

We now want to show how to derive the eigenfunctions  $u(t)$  and the scaling  $\boldsymbol{\lambda}$  of linear time-invariant systems. Inserting the assumption  $\boldsymbol{x}(t) = \boldsymbol{\lambda}u(t)$  with  $\dot{\boldsymbol{x}}(t) = \boldsymbol{\lambda}\dot{u}(t)$  into the system  $\boldsymbol{E}\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t)$ ,  $y(t) = \boldsymbol{c}^\top \boldsymbol{x}(t)$  yields  $n$  linear ODEs for  $u(t)$ :

$$\boldsymbol{E} \boldsymbol{\lambda} \dot{u}(t) = (\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b}) u(t) \iff \boldsymbol{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \dot{u}(t) = \begin{bmatrix} (\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_1 \\ \vdots \\ (\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_n \end{bmatrix} u(t). \quad (8.2)$$

For

$$\frac{(\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_1}{(\boldsymbol{E}\boldsymbol{\lambda})_1} = \dots = \frac{(\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_n}{(\boldsymbol{E}\boldsymbol{\lambda})_n} = s \quad (8.3)$$

all  $n$  linear ODEs become

$$\dot{u}(t) = s u(t), \quad (8.4)$$

with the solution  $u(t) = u_0 e^{st}$ . For the scaling  $\boldsymbol{\lambda}$  it follows:

$$s \boldsymbol{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} (\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_1 \\ \vdots \\ (\boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b})_n \end{bmatrix} \iff s \boldsymbol{E}\boldsymbol{\lambda} = \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b} \iff \boldsymbol{\lambda} = (s\boldsymbol{E} - \boldsymbol{A})^{-1}\boldsymbol{b}. \quad (8.5)$$

With this simple derivation we can show that complex exponentials  $u(t) = u_0 e^{st}$  are indeed the eigenfunctions of LTI systems.

### Application of the eigenfunctions

Complex exponentials are widely used in the context of linear systems theory due to the eigenfunction property. For example: with the Fourier series or Fourier/Laplace transform one can decompose an *arbitrary* signal into a sum of complex exponentials, i.e.  $u(t) = \sum_{k \in \mathbb{Z}} u_k e^{s_k t}$  with  $s_k = \frac{i2\pi k}{T} = ik\omega$ . Then, the response of the LTI system to this arbitrary input is given by a sum of complex exponentials scaled with  $\boldsymbol{\lambda}(s_k) = (s_k \boldsymbol{E} - \boldsymbol{A})^{-1} \boldsymbol{b}$ , i.e.  $\boldsymbol{x}(t) = \sum_{k \in \mathbb{Z}} \boldsymbol{\lambda}(s_k) u_k e^{s_k t}$ . Thus, the well-known procedure to calculate the solution of an LTI system via the Laplace transform is as follows:

- Decomposition of the input signal  $u(t)$  into eigenfunctions  $e^{st}$  (Laplace transform)
- Scaling of the eigenfunctions with  $\boldsymbol{\lambda}(s)$  (transfer function)
- Composition of the output signal  $\boldsymbol{x}(t)$  using the scaled eigenfunctions  $\boldsymbol{\lambda}(s)e^{st}$  (inverse Laplace transform)

## 8.2 Eigenfunction of bilinear and quadratic-bilinear systems

After having revisited the concept of eigenfunctions for linear dynamical systems we now want to derive the eigenfunctions of bilinear and quadratic-bilinear systems. We will see that the latter can be derived from *Bernoulli* differential equations and that they can be represented as a sum of exponential functions after some rearranging via the geometric series.

### Bilinear systems

We first consider bilinear systems. Inserting the assumption  $\mathbf{x}(t) = \boldsymbol{\lambda}u(t)$  with  $\dot{\mathbf{x}}(t) = \boldsymbol{\lambda}\dot{u}(t)$  into  $\mathbf{E}\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{N}\mathbf{x}(t)u(t) + \mathbf{b}u(t)$ ,  $y(t) = \mathbf{c}^\top \mathbf{x}(t)$  yields  $n$  Bernoulli ODEs for  $u(t)$ :

$$\begin{aligned} \mathbf{E}\boldsymbol{\lambda}\dot{u}(t) &= (\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})u(t) + \mathbf{N}\boldsymbol{\lambda}u^2(t) \\ \Leftrightarrow \mathbf{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \dot{u}(t) &= \begin{bmatrix} (\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_1 \\ \vdots \\ (\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_n \end{bmatrix} u(t) + \begin{bmatrix} (\mathbf{N}\boldsymbol{\lambda})_1 \\ \vdots \\ (\mathbf{N}\boldsymbol{\lambda})_n \end{bmatrix} u^2(t) \end{aligned} \quad (8.6)$$

For

$$\frac{(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_1}{(\mathbf{E}\boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_n}{(\mathbf{E}\boldsymbol{\lambda})_n} = s \quad \text{and} \quad \frac{(\mathbf{N}\boldsymbol{\lambda})_1}{(\mathbf{E}\boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{N}\boldsymbol{\lambda})_n}{(\mathbf{E}\boldsymbol{\lambda})_n} = \tilde{n} \quad (8.7)$$

all  $n$  Bernoulli differential equations become

$$\dot{u}(t) = s u(t) + \tilde{n} u^2(t), \quad (8.8)$$

with the solution

$$u(t) = \left( \frac{\tilde{n}}{s} (e^{-st} - 1) + \frac{1}{u_0} e^{-st} \right)^{-1}, \quad \text{where } \frac{\tilde{n}}{s} (e^{-st} - 1) + \frac{1}{u_0} e^{-st} \neq 0, \quad \forall t \geq 0. \quad (8.9)$$

For the scaling  $\boldsymbol{\lambda}$  it follows:

$$\frac{(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_1}{(\mathbf{E}\boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_n}{(\mathbf{E}\boldsymbol{\lambda})_n} = s \quad \Leftrightarrow \quad \boldsymbol{\lambda} = (s\mathbf{E} - \mathbf{A})^{-1}\mathbf{b} \quad (8.10)$$

and

$$\begin{bmatrix} (\mathbf{N}\boldsymbol{\lambda})_1 \\ \vdots \\ (\mathbf{N}\boldsymbol{\lambda})_n \end{bmatrix} = \tilde{n} \mathbf{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \quad \Leftrightarrow \quad \mathbf{N}\boldsymbol{\lambda} = \tilde{n} \mathbf{E}\boldsymbol{\lambda}. \quad (8.11)$$

This means: the eigenfunctions  $u(t)$  of bilinear systems are given by (8.9). The scaling is  $\boldsymbol{\lambda} = (s\mathbf{E} - \mathbf{A})^{-1}\mathbf{b}$  with the further condition  $\mathbf{N}\boldsymbol{\lambda} = \tilde{n} \mathbf{E}\boldsymbol{\lambda}$ , i.e. the scaling is also eigenvector of the matrix  $\mathbf{N}$  with the eigenvalue  $\tilde{n}$ . It should be checked for which value  $s \in \mathbb{C}$  the scaling  $\boldsymbol{\lambda}$  truly is an eigenvector of the matrix  $\mathbf{N}$ .

The eigenfunctions  $u(t)$  of bilinear systems can be represented as sum of exponential functions using the geometric series  $\sum_{k=0}^{\infty} q^k = \frac{1}{1-q}$ ,  $|q| < 1$ . This yields:

$$\begin{aligned} u(t) &= \left( \frac{\tilde{n}}{s} (e^{-st} - 1) + \frac{1}{u_0} e^{-st} \right)^{-1} = \frac{s}{\tilde{n}} e^{st} \left( 1 + \frac{s}{\tilde{n} u_0} \right)^{-1} \sum_{k=0}^{\infty} \left( e^{st} \left( 1 + \frac{s}{\tilde{n} u_0} \right)^{-1} \right)^k \\ &= \frac{s}{\tilde{n}} \sum_{k=1}^{\infty} e^{kst} \left( 1 + \frac{s}{\tilde{n} u_0} \right)^{-k} = \sum_{k=1}^{\infty} \frac{s}{\tilde{n}} \left( 1 + \frac{s}{\tilde{n} u_0} \right)^{-k} e^{kst} \\ &= \sum_{k=1}^{\infty} u_k(s) e^{kst}. \end{aligned} \quad (8.12)$$

This fact could be related to the growing exponential approach. Namely: in Section 4.4.2 we have employed the Volterra series  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \mathbf{x}_k(t)$  and the mentioned approach to derive the scalings  $\boldsymbol{\lambda}_{k,\Delta}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k})$  for a sum of growing exponentials  $\mathbf{u}(t) = \sum_{l_1=1}^r \mathbf{1}_m u_{l_1} e^{s_{l_1} t}$  (cf. Eqs. (4.53), (4.54)). For the SISO and single-tone case with  $u(t) = e^{st}$  the solution is given by  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \boldsymbol{\lambda}_k(s) e^{kst}$  with  $\mathbf{x}_k(t) = \boldsymbol{\lambda}_k(s) u^k(t)$  and the scalings  $\boldsymbol{\lambda}_1(s) = (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b}$ ,  $\boldsymbol{\lambda}_k(s) = (ks\mathbf{E} - \mathbf{A})^{-1} \mathbf{N} \boldsymbol{\lambda}_{k-1}(s)$ . Hence, the question arises whether for an eigenfunction, i.e. an input of the form  $u(t) = \sum_{k=1}^{\infty} u_k(s) e^{kst}$ , the bilinear system would respond with  $\sum_{k=1}^{\infty} \boldsymbol{\lambda}_k(s) u_k(s) e^{kst}$ . This could not be clarified by the author and is thus subject to research.

### Quadratic-bilinear systems

We consider now quadratic-bilinear systems. Inserting the assumption  $\mathbf{x}(t) = \boldsymbol{\lambda} u(t)$  with  $\dot{\mathbf{x}}(t) = \boldsymbol{\lambda} \dot{u}(t)$  into  $\mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t)) + \mathbf{N} \mathbf{x}(t) u(t) + \mathbf{b} u(t)$ ,  $y(t) = \mathbf{c}^T \mathbf{x}(t)$  yields  $n$  Bernoulli ODEs for  $u(t)$ :

$$\begin{aligned} \mathbf{E} \boldsymbol{\lambda} \dot{u}(t) &= (\mathbf{A} \boldsymbol{\lambda} + \mathbf{b}) u(t) + (\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda})) u^2(t) \\ \iff \mathbf{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \dot{u}(t) &= \begin{bmatrix} (\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_1 \\ \vdots \\ (\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_n \end{bmatrix} u(t) + \begin{bmatrix} (\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_1 \\ \vdots \\ (\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_n \end{bmatrix} u^2(t). \end{aligned} \quad (8.13)$$

For

$$\frac{(\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_1}{(\mathbf{E} \boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_n}{(\mathbf{E} \boldsymbol{\lambda})_n} = s, \quad \frac{(\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_1}{(\mathbf{E} \boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_n}{(\mathbf{E} \boldsymbol{\lambda})_n} = h \quad (8.14)$$

all  $n$  Bernoulli differential equations become

$$\dot{u}(t) = s u(t) + h u^2(t), \quad (8.15)$$

with the solution

$$u(t) = \left( \frac{h}{s} (e^{-st} - 1) + \frac{1}{u_0} e^{-st} \right)^{-1}, \quad \text{where } \frac{h}{s} (e^{-st} - 1) + \frac{1}{u_0} e^{-st} \neq 0, \quad \forall t \geq 0. \quad (8.16)$$

For the scaling  $\boldsymbol{\lambda}$  it follows:

$$\frac{(\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_1}{(\mathbf{E} \boldsymbol{\lambda})_1} = \dots = \frac{(\mathbf{A} \boldsymbol{\lambda} + \mathbf{b})_n}{(\mathbf{E} \boldsymbol{\lambda})_n} = s \quad \iff \quad \boldsymbol{\lambda} = (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \quad (8.17)$$

and

$$\begin{bmatrix} (\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_1 \\ \vdots \\ (\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}))_n \end{bmatrix} = h \mathbf{E} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \quad \iff \quad \mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}) = h \mathbf{E} \boldsymbol{\lambda}. \quad (8.18)$$

This means: the eigenfunctions  $u(t)$  of quadratic-bilinear systems are given by (8.16). The scaling is  $\boldsymbol{\lambda} = (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b}$  with the further condition  $\mathbf{N} \boldsymbol{\lambda} + \mathbf{H}(\boldsymbol{\lambda} \otimes \boldsymbol{\lambda}) = h \mathbf{E} \boldsymbol{\lambda}$ , i.e. the scaling is also solution of this quadratic equation. It should be checked for which value  $s \in \mathbb{C}$  the scaling  $\boldsymbol{\lambda}$  truly satisfies this quadratic equation.

Following the same steps as in (8.12) the eigenfunctions of quadratic-bilinear systems can also be represented as sum of exponential functions, i.e.

$$u(t) = \sum_{k=1}^{\infty} u_k(s) e^{kst} \quad \text{with} \quad u_k(s) = \frac{s}{h} \left( 1 + \frac{s}{h u_0} \right)^{-k}. \quad (8.19)$$

In Section 4.4.2 we have employed the Volterra series  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \mathbf{x}_k(t)$  and derived the scalings  $\boldsymbol{\lambda}_{k,\Delta}^{(j_2, \dots, j_k)}(s_{l_1}, \dots, s_{l_k})$  for a sum of growing exponentials  $\mathbf{u}(t) = \sum_{l_1=1}^r \mathbf{1}_m u_{l_1} e^{s_{l_1} t}$  (cf. Eq. (4.58)). For the SISO and single-tone case with  $u(t) = e^{st}$  the solution is given by  $\mathbf{x}(t) = \sum_{k=1}^{\infty} \boldsymbol{\lambda}_k(s) e^{kst}$  with  $\mathbf{x}_k(t) = \boldsymbol{\lambda}_k(s) u^k(t)$  and the scalings  $\boldsymbol{\lambda}_1(s) = (s\mathbf{E} - \mathbf{A})^{-1} \mathbf{b}$ ,  $\boldsymbol{\lambda}_k(s) = (ks\mathbf{E} - \mathbf{A})^{-1} \left( \mathbf{H} \sum_{i=1}^{k-1} (\boldsymbol{\lambda}_i \otimes \boldsymbol{\lambda}_{k-i}) + \mathbf{N} \boldsymbol{\lambda}_{k-1} \right)$ .

**Discussion** The Volterra theory and the growing exponential approach consider exponential input signals  $u(t) = e^{st}$ , i.e., eigenfunctions of the *linear* system. The question arises: why are exponential functions also being considered for (quadratic-)bilinear systems instead of the corresponding eigenfunctions? The reason is simple. The Volterra model allows to decompose a polynomial nonlinear system in linear subsystems, such that complex exponentials can be exploited as eigenfunctions. On the contrary, a decomposition of general input signals into eigenfunctions of (quadratic-)bilinear systems is questionable and not fully clear yet. Thus, the growing exponential approach has established for analyzing (polynomial) nonlinear systems.

### 8.3 Eigenfunctions for nonlinear moment matching

Despite the success of the growing exponential approach we now want to outline how the eigenfunctions of *nonlinear systems* could be applied. First, the input signal should be represented using the eigenfunctions of the nonlinear system. Then, the eigenfunctions are scaled with the scaling/system function  $\boldsymbol{\lambda}$ . Finally, the output signal is composed of the scaled eigenfunctions. However, differences and questions remain:

- Exponential functions are generally not the eigenfunctions of nonlinear systems. Therefore, a new decomposition of the input signal is necessary. The question remains: which class of input signals can be represented by eigenfunctions of a nonlinear system?
- If the decomposition of the input into eigenfunctions succeeds, then a scaling with  $\boldsymbol{\lambda}$  is possible. However the question remains: how can the output signal be composed of the scaled eigenfunctions? The superposition principle does not apply to nonlinear systems, i.e. a sum of eigenfunctions is generally not an eigenfunction. Moreover, the scaling of the eigenfunctions does not form a vector space but a manifold.

All these questions make the application of “nonlinear” eigenfunctions difficult – or at least not as straightforward as complex exponentials.

In the context of nonlinear moment matching we have mentioned that the FOM should ideally be excited by the eigenfunctions of the nonlinear system. This means that the signal generator ODE (7.1a), together with the mapping (7.1b), should generate characterizing eigenfunctions of the system, before an invariant manifold  $\mathbf{x} = \boldsymbol{\nu}(\mathbf{x}_r^v)$  (a linear basis  $\mathbf{V}$ ) is constructed by solving the Sylvester-like PDE (7.10) (approximately).

Based on this thought, we will next construct the eigenfunctions of bilinear systems via a signal generator. Let us assume that

$$x_{r,i}^v(t) = \left( \frac{\tilde{n}_i}{\sigma_i} (e^{-\sigma_i t} - 1) + \frac{1}{x_{r,0,i}^v} e^{-\sigma_i t} \right)^{-1}, \quad i = 1, \dots, r, \quad (8.20)$$

with user-defined  $\{\sigma_i, \tilde{n}_i, x_{r,0,i}^v\}$ . These eigenfunctions can be generated via the following signal generator ODE (cf. Eq. (8.8)):

$$\dot{\mathbf{x}}_r^v(t) = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \mathbf{x}_r^v(t) + \begin{bmatrix} \tilde{n}_1 & & \\ & \ddots & \\ & & \tilde{n}_r \end{bmatrix} \begin{bmatrix} x_{r,1}^v(t)^2 \\ \vdots \\ x_{r,r}^v(t)^2 \end{bmatrix} = \mathbf{S}_v \mathbf{x}_r^v(t) + \mathbf{\Lambda}_{\tilde{n}} \begin{bmatrix} x_{r,1}^v(t)^2 \\ \vdots \\ x_{r,r}^v(t)^2 \end{bmatrix}. \quad (8.21)$$

We further use  $u(t) = \mathbf{1}_r^\top \mathbf{x}_r^v(t) = \sum_{i=1}^r x_{r,i}^v(t)$ , i.e. the input is a sum of eigenfunctions of the bilinear system. Now let us assume that we employ a linear projection  $\mathbf{x}(t) = \mathbf{V} \mathbf{x}_r^v(t)$ , where the columns  $\mathbf{v}_1, \dots, \mathbf{v}_r$  of the matrix  $\mathbf{V}$  represent the scalings  $\lambda_i$  of the eigenfunctions (cf. Eqs. (8.10), (8.11)). In other words:  $\mathbf{v}_i = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} \iff \mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} - \mathbf{b} \mathbf{1}_r^\top = \mathbf{0}$  and  $\mathbf{N} \mathbf{v}_i = \tilde{n}_i \mathbf{E} \mathbf{v}_i \iff \mathbf{N} \mathbf{V} = \mathbf{E} \mathbf{V} \mathbf{\Lambda}_{\tilde{n}}$ . Inserting the linear projection ansatz and the signal generator into the bilinear system yields:

$$\begin{aligned} \mathbf{E} \mathbf{V} \mathbf{S}_v \mathbf{x}_r^v(t) + \mathbf{E} \mathbf{V} \mathbf{\Lambda}_{\tilde{n}} \begin{bmatrix} x_{r,1}^v(t)^2 \\ \vdots \\ x_{r,r}^v(t)^2 \end{bmatrix} &= \mathbf{A} \mathbf{V} \mathbf{x}_r^v(t) + \mathbf{N} \mathbf{V} \mathbf{x}_r^v(t) \mathbf{1}_r^\top \mathbf{x}_r^v(t) + \mathbf{b} \mathbf{1}_r^\top \mathbf{x}_r^v(t) \\ \iff \underbrace{(\mathbf{E} \mathbf{V} \mathbf{S}_v - \mathbf{A} \mathbf{V} - \mathbf{b} \mathbf{1}_r^\top)}_{=\mathbf{0}} \mathbf{x}_r^v(t) + \mathbf{E} \mathbf{V} \mathbf{\Lambda}_{\tilde{n}} \underbrace{\left( \begin{bmatrix} x_{r,1}^v(t)^2 \\ \vdots \\ x_{r,r}^v(t)^2 \end{bmatrix} - \mathbf{x}_r^v(t) \mathbf{1}_r^\top \mathbf{x}_r^v(t) \right)}_{\stackrel{!}{=} \mathbf{0}} &= \mathbf{0}. \end{aligned} \quad (8.22)$$

The second term next to the well-known Sylvester equation disappears only for reduced order  $r = 1$  (because then  $x_r^v(t)^2 = \mathbf{x}_r^v(t) \mathbf{1}^\top \mathbf{x}_r^v(t)$  holds), or alternatively for eigenvalues  $\tilde{n}_i = 0$  of  $\mathbf{N}$  or linearly dependent eigenvectors in  $\mathbf{V}$ .

Why is this not working? We have assumed that the input signal  $u(t) = \sum_{i=1}^r x_{r,i}^v(t)$  is a sum of eigenfunctions of the original model generated by the signal generator and the state

$$\mathbf{x}(t) = \mathbf{V} \mathbf{x}_r^v(t) = \sum_{i=1}^r \mathbf{v}_i x_{r,i}^v(t), \quad \mathbf{v}_i = (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b}, \quad \mathbf{N} \mathbf{v}_i = \tilde{n}_i \mathbf{E} \mathbf{v}_i \quad (8.23)$$

is also a sum of the scaled eigenfunctions. However, in the nonlinear case the sum of eigenfunctions is generally not an eigenfunction. Therefore a linear projection  $\mathbf{x}(t) = \mathbf{V} \mathbf{x}_r^v(t)$  leads to a contradiction.

A nonlinear projection  $\mathbf{x}(t) = \boldsymbol{\nu}(\mathbf{x}_r^v(t))$  with  $\boldsymbol{\nu}(\mathbf{x}_r^v) : \mathbb{R}^r \rightarrow \mathbb{R}^n$  could take into account how linear combinations of eigenfunctions  $u(t) = \sum_{i=1}^r x_{r,i}^v(t)$  affect the original nonlinear model. The projection does not define a vector space but a manifold. The remaining problem is that the PDE (7.10) needs to be solved for  $\boldsymbol{\nu}(\mathbf{x}_r^v)$ . The procedure is thus promising, but still subject to further research.



## 8.4 Signal generator interpretation of Volterra series interpolation

In Section 5.4 we have discussed the Volterra series interpolation framework for the reduction of bilinear systems. We have seen that the Krylov vectors can be computed in an Arnoldi-like manner (cf. (5.45a)) or via the solution of the *state-independent* Sylvester equations (5.43a) or (5.52a). On the other hand, in Section 7.1 we have discussed the nonlinear moment matching framework from Astolfi for the reduction of general nonlinear state-space systems. We have seen that the approach exploits the concept of signal generators and requires the solution of a nonlinear, state-dependent Sylvester-like PDE (7.10) to compute  $\boldsymbol{\nu}(\boldsymbol{x}_r^v)$ .

At this point we asked ourselves how the Volterra series interpolation could be embedded in the more general framework from Astolfi. In other words, the question raised how the signal generator and the input are implicitly being chosen to obtain the state-independent Sylvester equation (5.43a). This will be discussed in the following. It will turn out that by applying a linear projection and a linear signal generator to the bilinear system/subsystems (4.25) we gain the structure of the Sylvester equations. Moreover, we will discuss the required input  $u(t)$  to be able to factor out  $\boldsymbol{x}_r^v(t)$ .

First of all we want to motivate the use of a linear projection for each subsystem. The Volterra series interpolation framework uses the following linear ansatz (with  $\boldsymbol{x}_r(t) \stackrel{!}{=} \boldsymbol{x}_r^v(t)$ ):

$$\boldsymbol{x}(t) \approx \boldsymbol{V} \boldsymbol{x}_r^v(t) = \sum_{i=1}^r \boldsymbol{v}_i x_{r,i}^v(t) = \sum_{i=1}^r \sum_{k=1}^{\infty} \boldsymbol{v}_i^{(k)} x_{r,i}^v(t), \quad \boldsymbol{v}_i = \sum_{k=1}^{\infty} \boldsymbol{v}_i^{(k)} \Leftrightarrow \boldsymbol{V} = \sum_{k=1}^{\infty} \boldsymbol{V}^{(k)}. \quad (8.24)$$

On the other hand, we know from the Volterra series that  $\boldsymbol{x}(t) = \sum_{k=1}^{\infty} \boldsymbol{x}_k(t)$ . Thus, it seems reasonable to assign the matrices  $\boldsymbol{V}^{(k)}$  to each  $k$ -th subsystem:

$$\boldsymbol{x}(t) = \sum_{k=1}^{\infty} \boldsymbol{x}_k(t) \stackrel{!}{=} \sum_{k=1}^{\infty} \boldsymbol{V}^{(k)} \boldsymbol{x}_r^v(t) \implies \boldsymbol{x}_k(t) = \boldsymbol{V}^{(k)} \boldsymbol{x}_r^v(t) = \sum_{i=1}^r \boldsymbol{v}_i^{(k)} x_{r,i}^v(t). \quad (8.25)$$

Motivated by the structure of the Sylvester equations (5.43a) or (5.52a) having a shift matrix  $\boldsymbol{S}_v$ , it seems reasonable to use a linear signal generator:

$$\dot{\boldsymbol{x}}_r^v(t) = \boldsymbol{S}_v \boldsymbol{x}_r^v(t), \quad \boldsymbol{x}_r^v(0) = \boldsymbol{x}_{r,0}^v \Leftrightarrow \boldsymbol{x}_r^v(t) = e^{\boldsymbol{S}_v t} \boldsymbol{x}_{r,0}^v, \quad x_{r,i}^v(t) = e^{\sigma_i t} x_{r,0,i}^v. \quad (8.26)$$

Inserting the linear projection and the linear signal generator into the bilinear system or into the subsystems (4.25) yields:

$$\boldsymbol{E} \boldsymbol{V} \boldsymbol{S}_v \boldsymbol{x}_r^v(t) = \boldsymbol{A} \boldsymbol{V} \boldsymbol{x}_r^v(t) + \boldsymbol{N} \boldsymbol{V} \boldsymbol{x}_r^v(t) u(t) + \boldsymbol{b} u(t) \quad (8.27)$$

and

$$\begin{aligned} \boldsymbol{E} \boldsymbol{V}^{(1)} \boldsymbol{S}_v \boldsymbol{x}_r^v(t) &= \boldsymbol{A} \boldsymbol{V}^{(1)} \boldsymbol{x}_r^v(t) + \boldsymbol{b} u(t), \\ \boldsymbol{E} \boldsymbol{V}^{(k)} \boldsymbol{S}_v \boldsymbol{x}_r^v(t) &= \boldsymbol{A} \boldsymbol{V}^{(k)} \boldsymbol{x}_r^v(t) + \boldsymbol{N} \boldsymbol{V}^{(k-1)} \boldsymbol{x}_r^v(t) u(t). \end{aligned} \quad (8.28)$$

The question is now how the input  $u(t)$  should be chosen to factor out  $\boldsymbol{x}_r^v(t)$  and gain the well-known Sylvester equations. For this purpose the input should fulfill two conditions:

$$u(t) = \mathbf{1}_r^T \boldsymbol{x}_r^v(t) \stackrel{!}{=} \sum_{i=1}^r e^{\sigma_i t} x_{r,0,i}^v \quad \text{and} \quad \boldsymbol{x}_r^v(t) u(t) \stackrel{!}{=} \boldsymbol{U}_v^T \boldsymbol{x}_r^v(t). \quad (8.29)$$

The first equation is well-known from the linear case, i.e.  $\mathbf{b}u(t) = \mathbf{b}\mathbf{1}_r^\top \mathbf{x}_r^v(t)$ . The second equation requires more analysis. The  $i$ -th entry of the right-hand side is given by  $(\mathbf{U}_v^\top \mathbf{x}_r^v(t))_i = \sum_{i_u=1}^r u_{i_u,i}^v x_{r,i_u}^v(t) = \sum_{i_u=1}^r u_{i_u,i}^v e^{\sigma_{i_u}t} x_{r,0,i_u}^v$ . By looking closely at the left- and right-hand side of the second condition we can get the expression for the input:

$$\begin{aligned} x_{r,i}^v(t)u(t) &= e^{\sigma_i t} x_{r,0,i}^v \sum_{i_u=1}^r \sum_{l=1}^r u_{i_u,l}^v e^{(\sigma_{i_u}-\sigma_l)t} \frac{x_{r,0,i_u}^v}{x_{r,0,l}^v} \\ &:= \sum_{i_u=1}^r e^{\sigma_i t} x_{r,0,i}^v u_{i_u,i}^v e^{(\sigma_{i_u}-\sigma_i)t} \frac{x_{r,0,i_u}^v}{x_{r,0,i}^v} \\ &= \sum_{i_u=1}^r u_{i_u,i}^v e^{\sigma_{i_u}t} x_{r,0,i_u}^v = (\mathbf{U}_v^\top \mathbf{x}_r^v(t))_i. \end{aligned} \quad (8.30)$$

This means that the first subsystem is excited by  $u(t) = \sum_{i=1}^r e^{\sigma_i t} x_{r,0,i}^v$ , whereas the higher subsystems are excited by  $u(t) = \sum_{i_u=1}^r \sum_{l=1}^r u_{i_u,l}^v e^{(\sigma_{i_u}-\sigma_l)t} \frac{x_{r,0,i_u}^v}{x_{r,0,l}^v}$ . Physically it does not make sense that higher subsystems are excited with a different input than the first subsystem. We only proceed in this way to arrive at the desired Sylvester equations. Moreover, we will see next that some terms need to be neglected in order to be compliant with the projection ansatz and get the Volterra series interpolation.<sup>1</sup>

According to the applied projection ansatz (8.25) and the Volterra series interpolation, the response of the first subsystem should read

$$\mathbf{x}_1(t) = \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v = \sum_{i=1}^r \mathbf{v}_i^{(1)} x_{r,i}^v(t). \quad (8.31)$$

The first subsystem is linear with respect to the input term  $\mathbf{b}u(t)$ . Therefore exponential functions  $e^{\sigma_i t}$  are scaled with the matrix  $(\sigma_i \mathbf{E} - \mathbf{A})^{-1}$ :

$$\mathbf{b}u(t) := \sum_{i=1}^r \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v \implies \mathbf{x}_1(t) = \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v. \quad (8.32)$$

The response of the second subsystem should read

$$\mathbf{x}_2(t) = \sum_{i=1}^r \sum_{l_1=1}^r u_{i,l_1}^v (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} (\sigma_{l_1} \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v = \sum_{i=1}^r \mathbf{v}_i^{(2)} x_{r,i}^v(t). \quad (8.33)$$

The second subsystem is linear w.r.t. the input term  $\mathbf{N}\mathbf{x}_1(t)u(t)$ . Therefore exponential functions  $e^{\sigma_i t}$  are scaled (similarly to the first subsystem) with the matrix  $(\sigma_i \mathbf{E} - \mathbf{A})^{-1}$ :

$$\mathbf{N}\mathbf{x}_1(t)u(t) := \sum_{i=1}^r \mathbf{a}_i e^{\sigma_i t} x_{r,0,i}^v \implies \mathbf{x}_2(t) = \sum_{i=1}^r (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{a}_i e^{\sigma_i t} x_{r,0,i}^v. \quad (8.34)$$

---

<sup>1</sup>It would be more meaningful to use the input  $u(t) = \sum_{i=1}^r e^{\sigma_i t} x_{r,0,i}^v + \sum_{i_u=1}^r \sum_{l=1}^r u_{i_u,l}^v e^{(\sigma_{i_u}-\sigma_l)t} \frac{x_{r,0,i_u}^v}{x_{r,0,l}^v}$  for all subsystems. Nevertheless, with this ansatz more terms need to be neglected to be compliant with a linear projection. For the sake of brevity we will not present this explicitly. The procedure is similar to the one described here for the mentioned ansatz.

The multiplication  $\mathbf{x}_1(t)u(t)$  between two terms of the state and four terms of the input would result in a total of eight terms (for  $r=2$ ). To get the desired result four terms must be neglected. This is illustrated in the following:

$$\begin{aligned} \mathbf{x}_1(t)u(t) &= (\mathbf{v}_1^{(1)} e^{\sigma_1 t} x_{r,0,1}^v + \mathbf{v}_2^{(1)} e^{\sigma_2 t} x_{r,0,2}^v) (u_{1,1}^v + u_{2,2}^v + u_{1,2}^v e^{(\sigma_1 - \sigma_2)t} \frac{x_{r,0,1}^v}{x_{r,0,2}^v} + u_{2,1}^v e^{(\sigma_2 - \sigma_1)t} \frac{x_{r,0,2}^v}{x_{r,0,1}^v}) \\ &= \sum_{i_x=1}^r \mathbf{v}_{i_x}^{(1)} e^{\sigma_{i_x} t} x_{r,0,i_x}^v \sum_{i_u=1}^r \sum_{l_1=1}^r u_{i_u,l_1}^v e^{(\sigma_{i_u} - \sigma_{l_1})t} \frac{x_{r,0,i_u}^v}{x_{r,0,l_1}^v} \\ &\stackrel{i_x=l_1}{=} \sum_{i=1}^r \sum_{l_1=1}^r \mathbf{v}_{l_1}^{(1)} e^{\sigma_{l_1} t} x_{r,0,l_1}^v u_{i,l_1}^v e^{(\sigma_i - \sigma_{l_1})t} \frac{x_{r,0,i}^v}{x_{r,0,l_1}^v} = \sum_{i=1}^r \sum_{l_1=1}^r u_{i,l_1}^v \mathbf{v}_{l_1}^{(1)} e^{\sigma_i t} x_{r,0,i}^v. \end{aligned} \quad (8.35)$$

This means that only the terms  $i_x = l_1$  are considered and  $r^2$  terms are neglected. The procedure is the same for higher subsystems. The input for the  $k$ -th subsystem reads:

$$\begin{aligned} \mathbf{x}_{k-1}(t)u(t) &= \sum_{i_x=1}^r \mathbf{v}_{i_x}^{(k-1)} e^{\sigma_{i_x} t} x_{r,0,i_x}^v \sum_{i_u=1}^r \sum_{l_{k-1}=1}^r u_{i_u,l_{k-1}}^v e^{(\sigma_{i_u} - \sigma_{l_{k-1}})t} \frac{x_{r,0,i_u}^v}{x_{r,0,l_{k-1}}^v} \\ &:= \sum_{i=1}^r \sum_{l_{k-1}=1}^r \mathbf{v}_{l_{k-1}}^{(k-1)} e^{\sigma_{l_{k-1}} t} x_{r,0,l_{k-1}}^v u_{i,l_{k-1}}^v e^{(\sigma_i - \sigma_{l_{k-1}})t} \frac{x_{r,0,i}^v}{x_{r,0,l_{k-1}}^v} \\ &= \sum_{i=1}^r \sum_{l_{k-1}=1}^r u_{i,l_{k-1}}^v \mathbf{v}_{l_{k-1}}^{(k-1)} e^{\sigma_i t} x_{r,0,i}^v \iff \mathbf{V}^{(k-1)} \mathbf{U}_v^\top \mathbf{x}_r^v(t). \end{aligned} \quad (8.36)$$

The solution of the  $k$ -th subsystem is then given by

$$\mathbf{x}_k(t) = \sum_{i=1}^r \sum_{l_{k-1}=1}^r u_{i,l_{k-1}}^v (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{N} \mathbf{v}_{l_{k-1}}^{(k-1)} e^{\sigma_i t} x_{r,0,i}^v = \sum_{i=1}^r \mathbf{v}_i^{(k)} x_{r,i}^v(t). \quad (8.37)$$

To sum up: we can obtain the bilinear Sylvester equations by applying a linear projection and a linear signal generator, whereby the input  $u(t)$  needs to fulfill two conditions. Moreover, certain terms need to be neglected to be conform with the linear projection ansatz and the Volterra series interpolation conditions.

## Implicit Volterra series interpolation

We have discussed in Section 5.4 that the implicit Volterra series interpolation represents a special case of the most general framework for diagonal weighting matrices  $\mathbf{U}_v, \mathbf{U}_w$ . More importantly, we have seen that the implicit Volterra interpolation conditions (5.47) are equivalent to interpolating the special linear system  $\Sigma = (\mathbf{A} + \eta_{i,i} \mathbf{N}, \mathbf{b}, \mathbf{c}^\top, \mathbf{E})$ . We now provide a time-domain/signal generator interpretation for implicit Volterra series interpolation. A similar result is discussed in [92, Sec. 3.4].

For implicit Volterra the input of the  $k$ -th subsystem is given by

$$u(t) = \eta_{i,i}, \quad (8.38)$$

i.e. a constant input, where the amplitude represents the diagonal weights  $\eta_{i,i} = u_{i,i}^v$ .

Then, the subsystems read

$$\begin{aligned} \mathbf{E} \dot{\mathbf{x}}_1(t) &= \mathbf{A} \mathbf{x}_1(t) + \mathbf{b} u(t), \\ \mathbf{E} \dot{\mathbf{x}}_k(t) &= \mathbf{A} \mathbf{x}_k(t) + \eta_{i,i} \mathbf{N} \mathbf{x}_{k-1}(t), \quad k \geq 2, \end{aligned} \quad (8.39)$$

or the bilinear system becomes

$$\mathbf{E} \dot{\mathbf{x}}(t) = (\mathbf{A} + \eta_{i,i} \mathbf{N}) \mathbf{x}(t) + \mathbf{b} u(t). \quad (8.40)$$

Since the above system is linear, we know that for an exponential input  $u(t) = \sum_{i=1}^r e^{\sigma_i t} x_{r,0,i}^v$  (i.e. a linear signal generator) the solution is

$$\mathbf{x}(t) = \sum_{i=1}^r (\sigma_i \mathbf{E} - (\mathbf{A} + \eta_{i,i} \mathbf{N}))^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v. \quad (8.41)$$

Using the Neumann series  $(\mathbf{I} - \mathbf{T})^{-1} = \sum_{k=0}^{\infty} \mathbf{T}^k$  for  $\|\mathbf{T}\| < 1$  we receive the implicit Volterra series interpolation:

$$\begin{aligned} \mathbf{x}(t) &= \sum_{i=1}^r \left( (\sigma_i \mathbf{E} - \mathbf{A}) \left( \mathbf{I} - (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \eta_{i,i} \mathbf{N} \right) \right)^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v \\ &= \sum_{i=1}^r \sum_{k=0}^{\infty} \left( (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \eta_{i,i} \mathbf{N} \right)^k (\sigma_i \mathbf{E} - \mathbf{A})^{-1} \mathbf{b} e^{\sigma_i t} x_{r,0,i}^v \\ &= \sum_{i=1}^r \sum_{k=1}^{\infty} (\eta_{i,i})^{k-1} \boldsymbol{\lambda}_k^{\square}(\sigma_i, \dots, \sigma_i) e^{\sigma_i t} x_{r,0,i}^v \\ &= \sum_{i=1}^r \mathbf{v}_i e^{\sigma_i t} x_{r,0,i}^v. \end{aligned} \quad (8.42)$$

## PART IV

### NONLINEAR SECOND-ORDER SYSTEMS



## Chapter 9

# Fundamentals of Model Reduction for Nonlinear Mechanical Systems

In this part of the thesis we focus on the reduction of nonlinear *second-order* systems. They arise in many engineering applications, e.g. in structural dynamics, MEMS, vibroacoustics, biomechanics, etc. Nonlinear terms typically originate from (i) nonlinear geometric behavior (i.e. large deformations), (ii) material nonlinearities (e.g. viscoelasticity), or (iii) nonlinear boundary conditions (e.g. contact interactions).

This chapter deals with the mathematical and model reduction fundamentals for nonlinear mechanical systems. First, the considered equations of motion and the assumed system properties are introduced. Then, the numerical time integration of the FOM is discussed. After that, the focus is laid on model reduction based on both linear and nonlinear projection. In the linear projection framework we will turn our attention to the concept of basis augmentation with modal derivatives (MDs). We will revisit their original derivation [130] based on the perturbation of the linearized eigenvalue problem and also discuss the static derivatives. In the nonlinear projection framework we will concentrate on the emerging convective term and the time integration of manifold-ROMs. Then, the special case of a quadratic manifold (QM) is treated, which can be parametrized with modes and modal derivatives.

This chapter lays the foundation for the upcoming Chapters 10 and 11. Certain parts are based on the corresponding sections of [75] and [73].

### 9.1 System representation

We consider a large-scale, nonlinear time-invariant, exponentially stable, MIMO second-order model of the form

$$\mathbf{M} \ddot{\mathbf{q}}(t) + \mathbf{D} \dot{\mathbf{q}}(t) + \mathbf{f}(\mathbf{q}(t)) = \mathbf{B} \mathbf{F}(t), \quad \mathbf{q}(0) = \mathbf{q}_0, \quad \dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_0, \quad (9.1a)$$

$$\mathbf{y}(t) = \mathbf{C} \mathbf{q}(t), \quad (9.1b)$$

with non-singular mass matrix  $\mathbf{M} \in \mathbb{R}^{n \times n}$ , displacements  $\mathbf{q}(t) \in \mathbb{R}^n$ , input forces  $\mathbf{F}(t) \in \mathbb{R}^m$ , outputs  $\mathbf{y}(t) \in \mathbb{R}^p$  and the smooth mapping  $\mathbf{f}(\mathbf{q}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ . Note that the considered equations of motion represent a special case of the more general system representation

$$\mathbf{M} \ddot{\mathbf{q}}(t) + \mathbf{f}_{\text{int}}(\dot{\mathbf{q}}(t), \mathbf{q}(t)) = \mathbf{f}_{\text{ext}}(\dot{\mathbf{q}}(t), \mathbf{q}(t), t), \quad \mathbf{q}(0) = \mathbf{q}_0, \quad \dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_0, \quad (9.2a)$$

$$\mathbf{y}(t) = \mathbf{h}(\dot{\mathbf{q}}(t), \mathbf{q}(t)), \quad (9.2b)$$

with  $\mathbf{f}_{\text{int}}(\dot{\mathbf{q}}(t), \mathbf{q}(t)) = \mathbf{D}\dot{\mathbf{q}}(t) + \mathbf{f}(\mathbf{q}(t))$  denoting the internal forces,  $\mathbf{f}_{\text{ext}}(\dot{\mathbf{q}}(t), \mathbf{q}(t), t) = \mathbf{B}\mathbf{F}(t)$  representing the time- and sometimes even displacement-dependent external forces, and the function  $\mathbf{h}(\dot{\mathbf{q}}(t), \mathbf{q}(t)) = \mathbf{C}\mathbf{q}$  as output mapping.

We assume that the mass matrix is symmetric positive definite and non-singular, i.e.  $\mathbf{M} = \mathbf{M}^\top \succ \mathbf{0}$ ,  $\det(\mathbf{M}) \neq 0$ . Moreover, we assume that the tangential stiffness matrix at the equilibrium point  $\mathbf{q}_{\text{eq}} = \mathbf{0}$ , i.e.  $\mathbf{K}(\mathbf{q}_{\text{eq}}) = \left. \frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}} \right|_{\mathbf{q}_{\text{eq}}}$ , is also symmetric positive definite. The modeling of damping in nonlinear dynamic analysis is not a trivial task. Thus, zero damping ( $\mathbf{D} = \mathbf{0}$ ) or a linear Rayleigh damping  $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}(\mathbf{q}_{\text{eq}})$  with  $\alpha, \beta \geq 0$  are often assumed. The latter case leads to a symmetric positive definite damping matrix. All these mentioned assumptions imply that the equilibrium point  $\mathbf{q}_{\text{eq}}$  is exponentially stable. For  $\mathbf{D} = \mathbf{0}$  the equilibrium point is said to be marginally/neutrally stable.

The second-order nonlinear system (9.1) can be reformulated in (an implicit) state-space representation as:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}(t) \\ \ddot{\mathbf{q}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{I}\dot{\mathbf{q}}(t) \\ -\mathbf{f}(\mathbf{q}(t)) - \mathbf{D}\dot{\mathbf{q}}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{B} \end{bmatrix} \mathbf{F}(t), \quad (9.3a)$$

$$\mathbf{y}(t) = \begin{bmatrix} \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix}. \quad (9.3b)$$

## 9.2 Time integration

The numerical simulation of nonlinear second-order systems is usually performed via the *generalized- $\alpha$*  scheme [61]. It constitutes a generalization of the previously proposed Newmark [189] and HHT- $\alpha$  schemes [121]. The algorithmic procedure is summarized in Algorithm 9.1, which is schematically leaned on [105] (see also [Bil19]).

The generalized- $\alpha$  method is an implicit solver composed of two steps. First, the displacements  $\mathbf{q}_{k+1}$  and velocities  $\dot{\mathbf{q}}_{k+1}$  at the next time-step are obtained from the solutions at the previous time-step  $k$  using the step-size  $h$  and the parameters  $\beta \in [0, \frac{1}{2}]$ ,  $\gamma \in [0, 1]$  (cf. lines 7-10). Secondly, two additional parameters  $\alpha_f$  and  $\alpha_m$  are employed to interpolate the displacements, velocities, accelerations and external forces between the  $k$ -th and  $k+1$ -th time-step (cf. lines 11-15). For the special choice  $\alpha_m = \alpha_f = 0$  one gets Newmark's scheme, whereas for  $\alpha_m = 0$ ,  $\alpha_f = \alpha$  one obtains the HHT- $\alpha$  scheme. These two  $\alpha$ -parameters are crucial to adjust the degree of numerical damping for high frequencies. [105]

Inserting the time-discretized equations in the system representation yields the residual

$$\mathbf{res}(\mathbf{q}_{k+1}) = \mathbf{M}\ddot{\mathbf{q}}_{k+1-\alpha_m} + \mathbf{D}\dot{\mathbf{q}}_{k+1-\alpha_f} + \mathbf{f}(\mathbf{q}_{k+1-\alpha_f}) - \mathbf{f}_{\text{ext},k+1-\alpha_f} \stackrel{!}{=} \mathbf{0}. \quad (9.4)$$

The Newton-Raphson method searches then for the solution  $\mathbf{q}_{k+1}$ . The analytical Jacobian of the residual  $\mathbf{K}_{\text{dyn},k+1}^{\text{iter}}(\mathbf{q}_{k+1}^{\text{iter}}) = \partial \mathbf{res}(\mathbf{q}_{k+1}) / \partial \mathbf{q}_{k+1}$  for every iteration step "iter" reads

$$\mathbf{K}_{\text{dyn},k+1}^{\text{iter}}(\mathbf{q}_{k+1}^{\text{iter}}) = \frac{1 - \alpha_m}{\beta h^2} \mathbf{M} + \frac{(1 - \alpha_f)\gamma}{\beta h} \mathbf{D} + (1 - \alpha_f) \mathbf{K}(\mathbf{q}_{k+1-\alpha_f}^{\text{iter}}). \quad (9.5)$$

The correction  $\Delta \mathbf{q}_{k+1}^i$  is computed by solving a LSE (cf. line 21). The next solution is then calculated by the update  $\mathbf{q}_{k+1}^{\text{iter}+1} = \mathbf{q}_{k+1}^{\text{iter}} + \Delta \mathbf{q}_{k+1}^{\text{iter}}$  (cf. lines 22-25). The Newton-Raphson loop is repeated until the norm of the residual undershoots a specified tolerance.



**Algorithm 9.1** Generalized- $\alpha$  time integration scheme

**Input:** Initial conditions  $\mathbf{q}_0, \dot{\mathbf{q}}_0 \in \mathbb{R}^n$ , parameters  $\gamma, \beta, \alpha_f$  and  $\alpha_m$ , time range  $t = [t_0, \dots, t_{\text{end}}]$ , step-size  $h$ , residual error tolerance  $\text{tol}$ , start residual  $\|\mathbf{res}_0\| = 1$

**Output:** Set of displacements  $\mathbf{q}_k$  at discrete time steps  $t_k$  for  $k = 0, \dots, N$

- 1: **▼ Initialization**
- 2:  $\ddot{\mathbf{q}}_0 = \mathbf{M}^{-1}(\mathbf{f}_{\text{ext}}(t_0) - \mathbf{f}(\mathbf{q}_0) - \mathbf{D}\dot{\mathbf{q}}_0)$  **► solve LSE**
- 3:  $k \leftarrow 0$
- 4: **while**  $t < t_{\text{end}}$  **do** **► time-marching loop**
- 5:    $k \leftarrow k + 1$
- 6:    $t_{k+1} = t_k + h$  **► time increment**
- 7:   **▼ prediction**
- 8:    $\mathbf{q}_{k+1}^i = \mathbf{q}_k^i + h\dot{\mathbf{q}}_k^i + (\frac{1}{2} - \beta)h^2\ddot{\mathbf{q}}_k^i$
- 9:    $\dot{\mathbf{q}}_{k+1}^i = \dot{\mathbf{q}}_k^i + (1 - \gamma)h\ddot{\mathbf{q}}_k^i$
- 10:    $\ddot{\mathbf{q}}_{k+1}^i = \mathbf{0}$
- 11:   **▼ mid-point in generalized-alpha scheme**
- 12:    $\mathbf{q}_{k+1-\alpha_f}^i = (1 - \alpha_f)\mathbf{q}_{k+1}^i + \alpha_f\mathbf{q}_k^i$
- 13:    $\dot{\mathbf{q}}_{k+1-\alpha_f}^i = (1 - \alpha_f)\dot{\mathbf{q}}_{k+1}^i + \alpha_f\dot{\mathbf{q}}_k^i$
- 14:    $\ddot{\mathbf{q}}_{k+1-\alpha_m}^i = (1 - \alpha_m)\ddot{\mathbf{q}}_{k+1}^i + \alpha_m\ddot{\mathbf{q}}_k^i$
- 15:    $\mathbf{f}_{\text{ext},k+1-\alpha_f}^i = (1 - \alpha_f)\mathbf{f}_{\text{ext},k+1}^i + \alpha_f\mathbf{f}_{\text{ext},k}^i$
- 16:   **▼ initial residual evaluation**
- 17:    $i \leftarrow 0$
- 18:    $\mathbf{res}_{k+1}^i = \mathbf{M}\ddot{\mathbf{q}}_{k+1-\alpha_m}^i + \mathbf{D}\dot{\mathbf{q}}_{k+1-\alpha_f}^i + \mathbf{f}(\mathbf{q}_{k+1-\alpha_f}^i) - \mathbf{f}_{\text{ext},k+1-\alpha_f}^i$
- 19:   **while**  $\|\mathbf{res}_{k+1}^i\| > \text{tol}$  **do** **► Newton-Raphson loop**
- 20:      $\mathbf{K}_{\text{dyn},k+1}^i(\mathbf{q}_{k+1}^i) = \frac{1-\alpha_m}{\beta h^2}\mathbf{M} + \frac{(1-\alpha_f)\gamma}{\beta h}\mathbf{D} + (1 - \alpha_f)\mathbf{K}(\mathbf{q}_{k+1-\alpha_f}^i)$
- 21:      $\Delta\mathbf{q}_{k+1}^i = -\left(\mathbf{K}_{\text{dyn},k+1}^i(\mathbf{q}_{k+1}^i)\right)^{-1}\mathbf{res}_{k+1}^i$  **► solve LSE**
- 22:     **▼ correction**
- 23:      $\mathbf{q}_{k+1}^{i+1} = \mathbf{q}_{k+1}^i + \Delta\mathbf{q}_{k+1}^i$
- 24:      $\dot{\mathbf{q}}_{k+1}^{i+1} = \dot{\mathbf{q}}_{k+1}^i + \frac{\gamma}{\beta h}\Delta\mathbf{q}_{k+1}^i$
- 25:      $\ddot{\mathbf{q}}_{k+1}^{i+1} = \ddot{\mathbf{q}}_{k+1}^i + \frac{1}{\beta h^2}\Delta\mathbf{q}_{k+1}^i$
- 26:     **▼ mid-point in generalized-alpha scheme**
- 27:      $\mathbf{q}_{k+1-\alpha_f}^{i+1} = (1 - \alpha_f)\mathbf{q}_{k+1}^{i+1} + \alpha_f\mathbf{q}_k^{i+1}$
- 28:      $\dot{\mathbf{q}}_{k+1-\alpha_f}^{i+1} = (1 - \alpha_f)\dot{\mathbf{q}}_{k+1}^{i+1} + \alpha_f\dot{\mathbf{q}}_k^{i+1}$
- 29:      $\ddot{\mathbf{q}}_{k+1-\alpha_m}^{i+1} = (1 - \alpha_m)\ddot{\mathbf{q}}_{k+1}^{i+1} + \alpha_m\ddot{\mathbf{q}}_k^{i+1}$
- 30:      $\mathbf{f}_{\text{ext},k+1-\alpha_f}^{i+1} = (1 - \alpha_f)\mathbf{f}_{\text{ext},k+1}^{i+1} + \alpha_f\mathbf{f}_{\text{ext},k}^{i+1}$
- 31:     **▼ iterative residual evaluation**
- 32:      $\mathbf{res}_{k+1}^{i+1} = \mathbf{M}\ddot{\mathbf{q}}_{k+1-\alpha_m}^{i+1} + \mathbf{D}\dot{\mathbf{q}}_{k+1-\alpha_f}^{i+1} + \mathbf{f}(\mathbf{q}_{k+1-\alpha_f}^{i+1}) - \mathbf{f}_{\text{ext},k+1-\alpha_f}^{i+1}$
- 33:      $i \leftarrow i + 1$  **► iter  $\leftarrow$  iter + 1**

### 9.3 Nonlinear reduction based on linear projection

The goal of nonlinear model reduction is to find a ROM of smaller dimension  $r \ll n$  using again a projection framework. The reduction of second-order systems is usually performed by

an *orthogonal* Galerkin projection rather than by an oblique Petrov-Galerkin projection. The former fulfills the principle of *virtual work* and, thus, preserves the symmetry and definiteness of the original matrices as well as the stability of the FOM.

Similar to state-space systems, we will distinguish between linear and nonlinear projection. In this section, the reduction of (9.1) based on linear projection is discussed.

### Linear Galerkin projection

A common way to reduce nonlinear second-order systems is to apply the well-known linear Galerkin projection ansatz  $\mathbf{q}(t) \approx \mathbf{V}\mathbf{q}_r(t)$ . Hereby, the displacements  $\mathbf{q}(t) \in \mathbb{R}^n$  are approximated by a linear combination of the basis vectors in  $\mathbf{V} \in \mathbb{R}^{n \times r}$  and the reduced displacements  $\mathbf{q}_r(t) \in \mathbb{R}^r$ . Inserting the ansatz  $\mathbf{q}(t) = \mathbf{V}\mathbf{q}_r(t) + \mathbf{e}(t)$  in (9.1a) and premultiplying the overdetermined system with the *orthogonal* projector  $\mathbf{\Pi} = \mathbf{V}(\mathbf{V}^\top \mathbf{V})^{-1} \mathbf{V}^\top$  leads to

$$\mathbf{\Pi} \left( \underbrace{M\mathbf{V}\ddot{\mathbf{q}}_r(t) + D\mathbf{V}\dot{\mathbf{q}}_r(t) + \mathbf{f}(\mathbf{V}\mathbf{q}_r(t)) - \mathbf{B}\mathbf{F}(t) - \boldsymbol{\varepsilon}(t)}_{=\boldsymbol{\xi}(\mathbf{V}\mathbf{q}_r(t), \mathbf{F}(t))} \right) = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{\Pi}(\boldsymbol{\xi}(\cdot, \cdot) - \boldsymbol{\varepsilon}(t)) = \mathbf{0}. \quad (9.6)$$

Enforcing the *Galerkin* condition  $\mathbf{V}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\mathbf{\Pi} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , the residual then vanishes and only the term  $\mathbf{\Pi} \boldsymbol{\xi}(\cdot, \cdot) = \mathbf{0}$  remains. This finally yields the (square) ROM

$$\mathbf{M}_r \ddot{\mathbf{q}}_r(t) + \mathbf{D}_r \dot{\mathbf{q}}_r(t) + \mathbf{V}^\top \mathbf{f}(\mathbf{V}\mathbf{q}_r(t)) = \mathbf{B}_r \mathbf{F}(t), \quad \mathbf{q}_r(0) = \mathbf{q}_{r,0}, \quad \dot{\mathbf{q}}_r(0) = \dot{\mathbf{q}}_{r,0}, \quad (9.7a)$$

$$\mathbf{y}_r(t) = \mathbf{C}_r \mathbf{q}_r(t), \quad (9.7b)$$

with reduced matrices  $\{\mathbf{M}_r, \mathbf{D}_r\} = \mathbf{V}^\top \{\mathbf{M}, \mathbf{D}\} \mathbf{V}$ ,  $\mathbf{B}_r = \mathbf{V}^\top \mathbf{B}$ ,  $\mathbf{C}_r = \mathbf{C} \mathbf{V}$ , initial conditions  $\{\mathbf{q}_r(0), \dot{\mathbf{q}}_r(0)\} = (\mathbf{V}^\top \mathbf{V})^{-1} \mathbf{V}^\top \{\mathbf{q}_0, \dot{\mathbf{q}}_0\}$  and the reduced nonlinear function  $\mathbf{f}_r(\mathbf{q}_r) = \mathbf{V}^\top \mathbf{f}(\mathbf{V}\mathbf{q}_r)$  with  $\mathbf{f}_r(\mathbf{q}_r) : \mathbb{R}^r \rightarrow \mathbb{R}^r$ .

In this linear projective framework, the main task is to efficiently compute a dimensional reduction basis  $\mathbf{V}$  that comprises the most dominant nonlinear dynamics to ensure a good approximation, measured e.g. point-wise by  $\|\mathbf{q}(t) - \mathbf{V}\mathbf{q}_r(t)\|_{(\cdot)}$  or  $\|\mathbf{y}(t) - \mathbf{y}_r(t)\|_{(\cdot)}$  with a suitable vector norm  $(\cdot) = \{1, 2, \infty, \dots\}$  (see Section 2.6). For the effective evaluation of the nonlinear term  $\mathbf{f}_r(\mathbf{q}_r)$  the hyper-reduction methods from Section 6.5 can be applied.

### Time integration

Similar to the FOM, the time integration of the ROM (9.7) is also accomplished using the generalized- $\alpha$  scheme. To this end, Algorithm 9.1 needs only small modifications.

First, the initial reduced acceleration  $\ddot{\mathbf{q}}_{r,0}$  is computed using the reduced quantities  $\mathbf{M}_r$ ,  $\mathbf{D}_r$ ,  $\mathbf{V}^\top \mathbf{f}(\mathbf{V}\mathbf{q}_r)$ ,  $\mathbf{V}^\top \mathbf{f}_{\text{ext}}(t)$  and the initial conditions  $\mathbf{q}_{r,0}$ ,  $\dot{\mathbf{q}}_{r,0}$ . Moreover, the residual equation in lines 18 and 32 should be replaced by

$$\mathbf{res}(\mathbf{q}_{r,k+1}) = \mathbf{M}_r \ddot{\mathbf{q}}_{r,k+1-\alpha_m} + \mathbf{D}_r \dot{\mathbf{q}}_{r,k+1-\alpha_f} + \mathbf{V}^\top \mathbf{f}(\mathbf{V}\mathbf{q}_{r,k+1-\alpha_f}) - \mathbf{V}^\top \mathbf{f}_{\text{ext},k+1-\alpha_f}. \quad (9.8)$$

Differentiating with respect to  $\mathbf{q}_{r,k+1}$  the Jacobian of the residual now reads

$$\mathbf{K}_{\text{dyn},k+1}^{\text{iter}}(\mathbf{q}_{r,k+1}^{\text{iter}}) = \frac{1-\alpha_m}{\beta h^2} \mathbf{M}_r + \frac{(1-\alpha_f)\gamma}{\beta h} \mathbf{D}_r + (1-\alpha_f) \mathbf{V}^\top \mathbf{K}(\mathbf{V}\mathbf{q}_{r,k+1-\alpha_f}^{\text{iter}}) \mathbf{V}. \quad (9.9)$$

### Reduction approaches

The question is now how to efficiently compute the reduction matrix  $\mathbf{V}$  needed for projection. In general, the same methods described for nonlinear state-space systems in Section 6.3.3 can be applied for nonlinear second-order systems as well, i.e. POD, TPWL, pure linear basis and basis augmentation. What is more, some reduction methods have been initially developed for second-order systems and then transferred to state-space models (and vice versa). This is the case with the basis augmentation approach using modes and *modal derivatives*, which have their origin in the context of nonlinear structural dynamics [129, 130, 240]. We will concentrate on this simulation-free reduction concept in the following.

The key idea is to first compute some dominant modes of the linearized, second-order system, and then to augment the reduction basis with perturbation derivatives capturing the nonlinear behavior (cf. ansatz (6.13)).

### Vibration modes of the linearized model

The nonlinear structural system (9.1) is first linearized around a linearization  $(\bar{\mathbf{q}}, \bar{\mathbf{F}})$  or equilibrium point  $(\mathbf{q}_{\text{eq}}, \mathbf{F}_{\text{eq}})$ . Possible ways to compute the linearization/equilibrium point are discussed in [73]. Very often one selects  $\mathbf{q}_{\text{eq}} = \mathbf{0}$  and  $\mathbf{F}_{\text{eq}} = \mathbf{0}$ , which leads to  $\mathbf{f}(\mathbf{q}_{\text{eq}}) = \mathbf{0}$  and  $\dot{\mathbf{q}}_{\text{eq}} = \mathbf{0}$ ,  $\dot{\mathbf{q}}_{\text{eq}} = \mathbf{0}$ . The Taylor series expansion of  $\mathbf{f}(\mathbf{q}(t))$  around  $(\mathbf{q}_{\text{eq}}, \mathbf{F}_{\text{eq}})$  yields

$$\begin{aligned} M\Delta\ddot{\mathbf{q}}(t) + D\Delta\dot{\mathbf{q}}(t) + \mathbf{f}(\mathbf{q}_{\text{eq}}) + \mathbf{K}_{\text{eq}}\Delta\mathbf{q}(t) &= \mathbf{B}\Delta\mathbf{F}(t), \quad \Delta\mathbf{q}(0) = \Delta\mathbf{q}_0, \quad \Delta\dot{\mathbf{q}}(0) = \Delta\dot{\mathbf{q}}_0, \\ \Delta\mathbf{y}(t) &= \mathbf{C}\Delta\mathbf{q}(t), \end{aligned} \quad (9.10)$$

where  $\Delta\mathbf{q}(t) = \mathbf{q}(t) - \mathbf{q}_{\text{eq}}$ ,  $\Delta\mathbf{F}(t) = \mathbf{F}(t) - \mathbf{F}_{\text{eq}}$  and  $\Delta\mathbf{y}(t) = \mathbf{y}(t) - \mathbf{y}_{\text{eq}}$  represent the deviation from the linearization point. The tangential stiffness matrix is given by

$$\mathbf{K}_{\text{eq}} = \mathbf{K}(\mathbf{q}_{\text{eq}}) = \left. \frac{\partial \mathbf{f}(\mathbf{q}(t))}{\partial \mathbf{q}(t)} \right|_{\mathbf{q}_{\text{eq}}}. \quad (9.11)$$

The quadratic eigenvalue problem is then derived by setting the damping to zero ( $\mathbf{D} = \mathbf{0}$ ) and considering the homogeneous system ( $\Delta\mathbf{F}(t) = \mathbf{0}$ ). In this undamped free motion case (i.e.  $\lambda_{1/2} = \pm i\omega$ ,  $\delta = 0$ ,  $\omega \neq 0$ ), the ansatz for the displacements and accelerations is (cf. Fn. 1)

$$\Delta\mathbf{q}(t) = \sum_{i=1}^{2n} C_i \phi_{i,\text{eq}} e^{\pm i\omega_{i,\text{eq}} t}, \quad \Delta\ddot{\mathbf{q}}(t) = \sum_{i=1}^{2n} -\omega_{i,\text{eq}}^2 C_i \phi_{i,\text{eq}} e^{\pm i\omega_{i,\text{eq}} t}, \quad (9.12)$$

where the scalings  $C_i$  are determined through the initial conditions. Inserting the above ansatz into (9.10) with  $\mathbf{D} = \mathbf{0}$ ,  $\Delta\mathbf{F}(t) = \mathbf{0}$  and  $\mathbf{f}(\mathbf{q}_{\text{eq}}) = \mathbf{0}$ , and canceling the time-dependent term  $e^{i\omega_{i,\text{eq}} t}$  finally delivers the quadratic eigenvalue problem

$$(\mathbf{K}_{\text{eq}} - \omega_{i,\text{eq}}^2 \mathbf{M})\phi_{i,\text{eq}} = \mathbf{0} \quad \iff \quad \mathbf{K}_{\text{eq}} \Phi_{\text{eq}} - \mathbf{M} \Phi_{\text{eq}} \Omega_{\text{eq}}^2 = \mathbf{0}. \quad (9.13)$$

Hereby, the undamped eigenfrequencies  $\{\omega_{i,\text{eq}}\}_{i=1}^n$  and the eigenmodes  $\{\phi_{i,\text{eq}}\}_{i=1}^n$  are encoded in the matrices  $\Omega_{\text{eq}} = \text{diag}(\omega_{1,\text{eq}}, \dots, \omega_{n,\text{eq}}) \in \mathbb{R}^{n \times n}$  and  $\Phi_{\text{eq}} = [\phi_{1,\text{eq}}, \dots, \phi_{n,\text{eq}}] \in \mathbb{R}^{n \times n}$ . Typically, the eigenmodes are normalized using the  $\mathbf{M}$ -weighted inner product. Moreover, they are orthogonalized to each other in both the  $\mathbf{M}$ - and  $\mathbf{K}$ -norm:  $\phi_{i,\text{eq}}^T \mathbf{M} \phi_{j,\text{eq}} = \delta_{ij}$ ,  $\phi_{i,\text{eq}}^T \mathbf{K}_{\text{eq}} \phi_{j,\text{eq}} = \omega_{i,\text{eq}}^2 \delta_{ij}$ . The reduction basis is then given by  $\mathbf{V}_\phi = [\phi_{1,\text{eq}}, \dots, \phi_{r,\text{eq}}] \in \mathbb{R}^{n \times r}$ .

### Perturbation of eigenmodes

The vibration modes of the linearized system  $\phi_i(\mathbf{q}_{\text{eq}})$  depend on the chosen linearization point  $\mathbf{q}_{\text{eq}}$ . Hence, the change of the modes with respect to the linearization point is of interest now. To this end, the perturbation of the modes  $\phi_i(\mathbf{q}_{\text{eq}})$ ,  $i = 1, \dots, r$  with respect to the amplitude  $\eta_j(t)$  of the modes  $\phi_j(\mathbf{q}_{\text{eq}})$ ,  $j = 1, \dots, r$  is considered in the following. Note that the modal amplitudes  $\eta_j(t)$  correspond to the reduced coordinates  $\mathbf{q}_r(t) = [q_{r,1}(t), \dots, q_{r,r}(t)]^\top = [\eta_1(t), \dots, \eta_r(t)]^\top$  in the projection ansatz  $\mathbf{q}(t) \approx \mathbf{V}_\phi \mathbf{q}_r(t) = \sum_{i=1}^r \phi_{i,\text{eq}} \eta_i(t)$ .

The perturbation of the linearized eigenvalue problem (9.13) with respect to the amplitude  $\eta_j(t)$  of mode  $\phi_{j,\text{eq}}$  yields (after applying the product rule): [130, 240]

$$\begin{aligned} \frac{\partial}{\partial \eta_j(t)} \left( \mathbf{K}(\mathbf{q}_{\text{eq}}) - \omega_i^2(\mathbf{q}_{\text{eq}}) \mathbf{M} \right) \phi_i(\mathbf{q}_{\text{eq}}) &= \mathbf{0} \\ \implies \left( \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} - \frac{\partial \omega_{i,\text{eq}}^2}{\partial \eta_j(t)} \mathbf{M} \right) \phi_{i,\text{eq}} + \left( \mathbf{K}_{\text{eq}} - \omega_{i,\text{eq}}^2 \mathbf{M} \right) \frac{\partial \phi_{i,\text{eq}}}{\partial \eta_j(t)} &= \mathbf{0}. \end{aligned} \quad (9.14)$$

Rearranging (9.14) yields the following linear system of equations

$$\left( \mathbf{K}_{\text{eq}} - \omega_{i,\text{eq}}^2 \mathbf{M} \right) \frac{\partial \phi_{i,\text{eq}}}{\partial \eta_j(t)} = \left( \frac{\partial \omega_{i,\text{eq}}^2}{\partial \eta_j(t)} \mathbf{M} - \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \right) \phi_{i,\text{eq}}. \quad (9.15)$$

The so-called *modal derivative*  $\theta_{ij} = \partial \phi_{i,\text{eq}} / \partial \eta_j(t)$  represents the derivative of mode  $\phi_{i,\text{eq}}$  with respect to the amplitude  $\eta_j(t)$  of mode  $\phi_{j,\text{eq}}$ . Note, however, that the obtained system constitutes a *singular* linear system of equations (LSE), since the matrix  $(\mathbf{K}_{\text{eq}} - \omega_{i,\text{eq}}^2 \mathbf{M})$  is singular. Therefore, special care has to be taken to be able to solve the singular system (9.15). Different methods to tackle the singularity are mentioned in the upcoming paragraph, in order to compute the modal derivatives under mass consideration.

The right-hand side of the above calculation formula is composed of two parts: the derivative of the eigenfrequencies and the derivative of the tangential stiffness matrix. The derivative of the squared  $i$ -th eigenfrequency  $\partial \omega_{i,\text{eq}}^2 / \partial \eta_j(t)$  can be obtained, if we premultiply the equation (9.14) by  $\phi_{i,\text{eq}}^\top$  and exploit the eigenvalue problem (9.13) together with the normalization condition  $\phi_{i,\text{eq}}^\top \mathbf{M} \phi_{i,\text{eq}} = 1$ . This yields (cf. [105, Sec. 6.11], [73])

$$\frac{\partial \omega_{i,\text{eq}}^2}{\partial \eta_j(t)} = \phi_{i,\text{eq}}^\top \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \phi_{i,\text{eq}}. \quad (9.16)$$

The derivative of the tangential stiffness matrix  $\partial \mathbf{K}_{\text{eq}} / \partial \eta_j(t)$  can be calculated either analytically within the finite element assembly procedure, or numerically via a finite difference scheme using the step width  $h$ :

- Forward difference:  $\left. \frac{\partial \mathbf{K}(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \approx \frac{\mathbf{K}(\mathbf{q}_{\text{eq}} + \phi_{j,\text{eq}} \cdot h) - \mathbf{K}(\mathbf{q}_{\text{eq}})}{h}$ ,
- Backward difference:  $\left. \frac{\partial \mathbf{K}(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \approx \frac{\mathbf{K}(\mathbf{q}_{\text{eq}}) - \mathbf{K}(\mathbf{q}_{\text{eq}} - \phi_{j,\text{eq}} \cdot h)}{h}$ ,
- Central difference:  $\left. \frac{\partial \mathbf{K}(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \approx \frac{\mathbf{K}(\mathbf{q}_{\text{eq}} + \phi_{j,\text{eq}} \cdot h) - \mathbf{K}(\mathbf{q}_{\text{eq}} - \phi_{j,\text{eq}} \cdot h)}{2h}$ .

We preferably employed the central difference scheme in AMfe, as it is more accurate and likely to yield a symmetric array. The choice of a suitable step width is discussed in [224].

**Modal derivatives handling the singular left-hand side** In order to compute the modal derivatives under mass consideration, the singular system (9.15) has to be solved. One way to achieve this consists in imposing an additional constraint to the modal derivative  $\boldsymbol{\theta}_{ij}$  to obtain a regular LSE with a unique solution. The additional condition for the modal derivatives requires that the norm of the vibration mode  $\boldsymbol{\phi}_{i,\text{eq}}^\top \mathbf{M} \boldsymbol{\phi}_{i,\text{eq}} = 1$  remains unchanged w.r.t. the amplitude  $\eta_j(t)$ . With symmetric  $\mathbf{M}$ , this imposition leads to the following constraint for the modal derivatives (see e.g. [73] for the derivation):

$$\frac{\partial}{\partial \eta_j(t)} \left( \boldsymbol{\phi}_{i,\text{eq}}^\top \mathbf{M} \boldsymbol{\phi}_{i,\text{eq}} \right) = 0 \quad \Longrightarrow \quad \boldsymbol{\phi}_{i,\text{eq}}^\top \mathbf{M}^\top \frac{\partial \boldsymbol{\phi}_{i,\text{eq}}}{\partial \eta_j(t)} = 0. \quad (9.17)$$

This means that the derivative  $\boldsymbol{\theta}_{ij}$  should be orthogonal to the vibration mode  $\boldsymbol{\phi}_{i,\text{eq}}$  w.r.t. the  $\mathbf{M}$ -weighted inner product. Using this imposition the modal derivatives can be calculated by one of the following approaches:

- **Nelson's method:** Proposed in [188] and further described in [129, 130, 240, 224].
- **Direct method:** Introduced in [105, Sec. 6.11.3] and applied in [132, 251]. The idea is to augment the singular system (9.15) with the condition (9.17).

$$\begin{bmatrix} (\mathbf{K}_{\text{eq}} - \omega_{i,\text{eq}}^2 \mathbf{M}) & -\mathbf{M} \boldsymbol{\phi}_{i,\text{eq}} \\ -(\mathbf{M} \boldsymbol{\phi}_{i,\text{eq}})^\top & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta}_{ij} \\ \frac{\partial \omega_{i,\text{eq}}^2}{\partial \eta_j(t)} \end{bmatrix} = \begin{bmatrix} -\frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \boldsymbol{\phi}_{i,\text{eq}} \\ 0 \end{bmatrix}. \quad (9.18)$$

- **Pseudoinverse and least-squares approaches:** The underdetermined LSE (9.15) can be solved using e.g. `pinv` and `lsqminnorm` in MATLAB.

All three strategies are well explained in [73]. Nelson's method is already integrated in AMfe. We also implemented the direct method for comparison reasons.

**Static modal derivatives excluding mass consideration** In many applications, the mass terms included in (9.15) are neglected, leading to the simplified LSE

$$\mathbf{K}_{\text{eq}} \left. \frac{\partial \boldsymbol{\phi}_{i,\text{eq}}}{\partial \eta_j(t)} \right|_s = - \left. \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \boldsymbol{\phi}_{i,\text{eq}} \right|_s, \quad (9.19)$$

where  $\boldsymbol{\theta}_{s,ij} = \left. \frac{\partial \boldsymbol{\phi}_{i,\text{eq}}}{\partial \eta_j(t)} \right|_s$  denotes the so-called *static* modal derivative (SMD). The omission of the shift-term  $-\omega_{i,\text{eq}}^2 \mathbf{M}$  leads to a regular linear system of equations, since the left-hand side of (9.19) is only composed of  $\mathbf{K}_{\text{eq}}$ . This matrix is constant and independent of  $\omega_{i,\text{eq}}$ , meaning that only one factorization of  $\mathbf{K}_{\text{eq}}$  is needed to compute all SMDs.

The right-hand side of (9.19) becomes simpler than in (9.15) due to the omission of the eigenfrequencies derivative  $\partial \omega_{i,\text{eq}}^2 / \eta_j(t)$ . What is more, it can be rewritten as

$$\left. \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \boldsymbol{\phi}_{i,\text{eq}} \right|_s = \left. \frac{\partial \mathbf{K}(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \boldsymbol{\phi}_{i,\text{eq}} := \left. \frac{\partial^2 \mathbf{f}(\mathbf{q})}{\partial \eta_j(t) \partial \mathbf{q}} \right|_{\mathbf{q}_{\text{eq}}} \left. \frac{\partial \mathbf{q}}{\partial \eta_i(t)} \right|_{\mathbf{q}_{\text{eq}}} = \left. \frac{\partial^2 \mathbf{f}(\mathbf{q})}{\partial \eta_j(t) \partial \eta_i(t)} \right|_{\mathbf{q}_{\text{eq}}} \quad (9.20)$$

using the relation (9.11) for the tangential stiffness matrix and expressing the displacement field at  $\mathbf{q}_{\text{eq}}$  as  $\mathbf{q}(t)|_{\mathbf{q}_{\text{eq}}} = \boldsymbol{\phi}_{i,\text{eq}} \eta_i(t) + \boldsymbol{\phi}_{j,\text{eq}} \eta_j(t)$ . Since  $\partial^2 \mathbf{f} / \partial \eta_i \eta_j = \partial^2 \mathbf{f} / \partial \eta_j \eta_i$  holds, the right-hand side, and consequently the SMDs, are *symmetric* with respect to  $i$  and  $j$ :

$$\left. \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_j(t)} \boldsymbol{\phi}_{i,\text{eq}} \right|_s = \left. \frac{\partial \mathbf{K}_{\text{eq}}}{\partial \eta_i(t)} \boldsymbol{\phi}_{j,\text{eq}} \right|_s \quad \Longrightarrow \quad \boldsymbol{\theta}_{s,ij} = \left. \frac{\partial \boldsymbol{\phi}_{i,\text{eq}}}{\partial \eta_j(t)} \right|_s = \left. \frac{\partial \boldsymbol{\phi}_{j,\text{eq}}}{\partial \eta_i(t)} \right|_s = \boldsymbol{\theta}_{s,ji}. \quad (9.21)$$

### Augmentation of the reduction basis

In many publications, e.g. [282, 224, 251, 75], it has been shown that the reduction of a nonlinear structural system with a pure linear reduction basis  $\mathbf{V}_\phi = [\phi_{1,\text{eq}}, \dots, \phi_{r,\text{eq}}]$  containing only vibration modes usually yields unsatisfactory results. In order to construct a reduction basis that also captures the nonlinear behavior of the FOM, the linear basis  $\Phi_r = [\phi_{1,\text{eq}}, \dots, \phi_{r,\text{eq}}] \in \mathbb{R}^{n \times r}$  is augmented with (S)MDs.<sup>1</sup>

The calculated MDs result in a third-order tensor  $\Theta \in \mathbb{R}^{n \times r \times r}$ , which is unfolded into a matrix using the 1-mode matricization

$$\Theta_{r,2} = [\theta_{11} \ \theta_{12} \ \cdots \ \theta_{1r} \ \cdots \ \theta_{r1} \ \theta_{r2} \ \cdots \ \theta_{rr}] \in \mathbb{R}^{n \times r^2}. \quad (9.22)$$

In the *non-symmetric* case of the MDs  $\theta_{ij} = \partial \phi_{i,\text{eq}} / \partial \eta_j(t)$ , the reduction basis is composed of  $r$  vibration modes  $\Phi_r \in \mathbb{R}^{n \times r}$  and  $r^2$  modal derivatives  $\Theta_{r,2} = [\theta_{11}, \dots, \theta_{rr}] \in \mathbb{R}^{n \times r^2}$ , yielding an augmented basis  $\mathbf{V}_{\text{aug}} = [\Phi_r, \Theta_{r,2}] \in \mathbb{R}^{n \times r + r^2}$ . In the *symmetric* case of the SMDs, where  $\theta_{s,ij} = \theta_{s,ji}$ , only  $o = r(r+1)/2$  distinct derivatives are included in the basis, yielding  $\Theta_{s,o} \in \mathbb{R}^{n \times o}$  and  $\mathbf{V}_{\text{aug}} = [\Phi_r, \Theta_{s,o}] \in \mathbb{R}^{n \times r + o}$ .

The raw reduction basis composed of both vibration modes and (S)MDs is generally not full rank, since the column vectors are usually redundant or linear dependent to each other. Thus, a deflation via SVD is often employed. The deflated basis is constructed with  $r_{\text{def}}$  left singular vectors corresponding to the leading singular values:  $\mathbf{V}_{\text{aug}}^{\text{defl}} = [\mathbf{u}_1, \dots, \mathbf{u}_{\text{def}}]$ . The deflated reduced order may be chosen at wish, or selected according to a given tolerance  $\varepsilon$ .

## 9.4 Nonlinear reduction based on nonlinear projection

In this section, we discuss the reduction of nonlinear second-order systems using a nonlinear projection framework (cf. Section 6.4 for nonlinear state-space systems).

### Nonlinear Galerkin projection

Another way of reducing (9.1) consists in applying a nonlinear Galerkin projection, where the approximation ansatz reads

$$\mathbf{q}(t) = \boldsymbol{\nu}(\mathbf{q}_r(t)) + \mathbf{e}(t), \quad (9.23)$$

with the smooth nonlinear mapping  $\boldsymbol{\nu}(\mathbf{q}_r) : \mathbb{R}^r \rightarrow \mathbb{R}^n$ . The first and second time-derivative of the ansatz are then given by

$$\dot{\mathbf{q}} = \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r)}{\partial \mathbf{q}_r} \dot{\mathbf{q}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_r} \dot{\mathbf{q}}_r, \quad (9.24)$$

$$\ddot{\mathbf{q}} = \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r)}{\partial \mathbf{q}_r} \ddot{\mathbf{q}}_r + \frac{\partial^2 \boldsymbol{\nu}(\mathbf{q}_r)}{\partial \mathbf{q}_r^2} (\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) = \widetilde{\mathbf{V}}_{\mathbf{q}_r} \ddot{\mathbf{q}}_r + \mathbf{d}\widetilde{\mathbf{V}}_{\mathbf{q}_r} (\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r), \quad (9.25)$$

<sup>1</sup>A similar approach is to employ Krylov vectors (cf. Eq. (11.8)) and their corresponding Krylov derivatives (see e.g. [175], [Him18]) for the basis augmentation approach.

with the Jacobian and the Hessian:

$$\widetilde{\mathbf{V}}(\mathbf{q}_r) = \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r)}{\partial \mathbf{q}_r} \in \mathbb{R}^{n \times r}, \quad d\widetilde{\mathbf{V}}(\mathbf{q}_r) = \frac{\partial^2 \boldsymbol{\nu}(\mathbf{q}_r)}{\partial \mathbf{q}_r^2} = \frac{\partial \widetilde{\mathbf{V}}(\mathbf{q}_r)}{\partial \mathbf{q}_r} \in \mathbb{R}^{n \times r^2}. \quad (9.26)$$

The tangential Jacobian  $\widetilde{\mathbf{V}}_{\mathbf{q}_r}$  spans the  $r$ -dimensional tangent space  $\widetilde{\mathcal{V}} = \mathcal{T}_{\mathbf{q}_r} \mathcal{M} = \text{ran}(\widetilde{\mathbf{V}}_{\mathbf{q}_r})$  of the manifold  $\mathcal{M} = \{\mathbf{q}_r \in \mathbb{R}^r : \mathbf{q} = \boldsymbol{\nu}(\mathbf{q}_r)\}$ .

Inserting the ansatz and its derivatives in (9.1a) yields an overdetermined system of equations with the residual  $\boldsymbol{\varepsilon}(t)$ . To obtain a square ROM, we project the resulting system onto the tangent space  $\widetilde{\mathcal{V}} = \text{ran}(\widetilde{\mathbf{V}}_{\mathbf{q}_r})$  orthogonally to this same space. This is accomplished by premultiplying the overdetermined system with the projector  $\widetilde{\boldsymbol{\Pi}} = \widetilde{\mathbf{V}}_{\mathbf{q}_r} (\widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{q}_r})^{-1} \widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top$ , where  $\widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{q}_r}$  is assumed non-singular, leading to

$$\widetilde{\boldsymbol{\Pi}} \left( \underbrace{M \widetilde{\mathbf{V}}_{\mathbf{q}_r} \ddot{\mathbf{q}}_r + M d\widetilde{\mathbf{V}}_{\mathbf{q}_r} (\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) + D \widetilde{\mathbf{V}}_{\mathbf{q}_r} \dot{\mathbf{q}}_r + \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r)) - \mathbf{B}\mathbf{F}}_{=\boldsymbol{\xi}(\boldsymbol{\nu}(\mathbf{q}_r(t)), \mathbf{F}(t))} - \boldsymbol{\varepsilon}(t) \right) = \mathbf{0}. \quad (9.27)$$

Enforcing the Galerkin condition  $\widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , which implies  $\widetilde{\boldsymbol{\Pi}} \boldsymbol{\varepsilon}(t) = \mathbf{0}$ , the residual then vanishes and only the term  $\widetilde{\boldsymbol{\Pi}} \boldsymbol{\xi}(\cdot, \cdot) = \mathbf{0}$  remains. This yields the nonlinearly projected ROM

$$\widetilde{\mathbf{M}}_r \ddot{\mathbf{q}}_r(t) + \widetilde{\mathbf{p}}_r(t) + \widetilde{\mathbf{D}}_r \dot{\mathbf{q}}_r(t) + \widetilde{\mathbf{V}}(\mathbf{q}_r(t))^\top \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r(t))) = \widetilde{\mathbf{B}}_r \mathbf{F}(t), \quad (9.28a)$$

$$\mathbf{y}_r(t) = \mathbf{C} \boldsymbol{\nu}(\mathbf{q}_r(t)), \quad (9.28b)$$

with reduced matrices  $\{\widetilde{\mathbf{M}}_r, \widetilde{\mathbf{D}}_r\} = \widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top \{M, D\} \widetilde{\mathbf{V}}_{\mathbf{q}_r}$  and  $\widetilde{\mathbf{B}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top \mathbf{B}$ , the convective term  $\widetilde{\mathbf{p}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_r}^\top M d\widetilde{\mathbf{V}}_{\mathbf{q}_r} (\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r)$  and  $\mathbf{q}_r(0) = \arg \min_{\mathbf{q}_{r,0}} \|\boldsymbol{\nu}(\mathbf{q}_{r,0}) - \mathbf{q}_0\|_2^2$ ,  $\dot{\mathbf{q}}_r(0) = (\widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}})^{-1} \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top \dot{\mathbf{q}}_0$ .

Note again that the computation of the initial condition  $\mathbf{q}_r(0)$  requires the solution of a nonlinear least-squares problem via e.g. `lsqnonlin` [138]. Further note that the reduced nonlinear mapping  $\widetilde{\mathbf{f}}_r(\mathbf{q}_r(t)) = \widetilde{\mathbf{V}}(\mathbf{q}_r(t))^\top \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r(t)))$  can be simplified using the hyper-reduction techniques from Section 6.5.

Due to the second derivative (9.25) with the Hessian  $d\widetilde{\mathbf{V}}_{\mathbf{q}_r}$ , an additional convective term  $\widetilde{\mathbf{p}}_r(t)$  pops up in the ROM (9.28). Special care must be taken with this part as it tends to destabilize the simulation of the ROM and often leads to convergence problems. This will be discussed in the next subsection.

**Remark 9.1.** Note that the use of a linear projection  $\mathbf{q} \approx \mathbf{V}\mathbf{q}_r$  constitutes a special case of the most general nonlinear projection  $\mathbf{q} \approx \boldsymbol{\nu}(\mathbf{q}_r)$  or the power series ansatz

$$\mathbf{q} = \sum_{k=1}^N \mathbf{V}^{(k)} \mathbf{q}_r^{(k)} = \mathbf{V}^{(1)} \mathbf{q}_r^{(1)} + \mathbf{V}^{(2)} \mathbf{q}_r^{(2)} + \dots \quad (9.29)$$

with  $\mathbf{V}^{(k)} \in \mathbb{R}^{n \times r^k}$  and  $\mathbf{q}_r^{(k)} = \overbrace{\mathbf{q}_r \otimes \dots \otimes \mathbf{q}_r}^{k \text{ times}} \in \mathbb{R}^{r^k}$ . The special case of a quadratic manifold will be discussed later.  $\triangle$

## Time integration

The numerical simulation of the manifold-ROM (9.28) is more complicated than the one for (9.7) due to the projection onto a displacement-dependent subspace. To keep it simple, a Newmark scheme (i.e.  $\alpha_m = \alpha_f = 0$ ) has been applied in [132] and [224, Ch. 7] to simulate quadratic manifold-ROMs. Based on these works, we present the slightly more general Newmark scheme for manifold-ROMs in Algorithm 9.2. This procedure has been (re)implemented and enhanced in AMfe during the semester thesis [Bil19].

Like in the state-space case (cf. Section 6.4.2), the reduced matrices  $\widetilde{\mathbf{M}}_r$  and  $\widetilde{\mathbf{D}}_r$  also change over time due to the state-dependent basis  $\widetilde{\mathbf{V}}_{q_r}$ . Thus, special care must be taken with the rank deficiency/regularity of the matrix  $\widetilde{\mathbf{V}}_{q_r}^\top \widetilde{\mathbf{V}}_{q_r}$ . Moreover, the arising convective term must be taken into account during the time integration. The residual then reads [224, Ch. 7]

$$\begin{aligned} \mathit{res}(\mathbf{q}_{r,k+1}) &= \widetilde{\mathbf{M}}_{r,k+1} \ddot{\mathbf{q}}_{r,k+1} + \widetilde{\mathbf{p}}_{r,k+1} + \widetilde{\mathbf{D}}_{r,k+1} \dot{\mathbf{q}}_{r,k+1} + \widetilde{\mathbf{f}}_{r,k+1}(\mathbf{q}_{r,k+1}) - \widetilde{\mathbf{f}}_{\text{ext},r,k+1} \\ &= \widetilde{\mathbf{V}}_{q_r,k+1}^\top \left( \mathbf{M} \widetilde{\mathbf{V}}_{q_r,k+1} \ddot{\mathbf{q}}_{r,k+1} + \mathbf{M} d\widetilde{\mathbf{V}}_{q_r,k+1} \dot{\mathbf{q}}_{r,k+1} \otimes \dot{\mathbf{q}}_{r,k+1} + \mathbf{D} \widetilde{\mathbf{V}}_{q_r,k+1} \dot{\mathbf{q}}_{r,k+1} + \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_{r,k+1})) - \mathbf{f}_{\text{ext},k+1} \right) \\ &= \widetilde{\mathbf{V}}_{q_r,k+1}^\top \cdot \mathit{res}_{\text{full}}(\mathbf{q}_{r,k+1}). \end{aligned} \quad (9.30)$$

The Jacobian of the residual  $\mathbf{K}_{\text{dyn},k+1}(\mathbf{q}_{r,k+1}) = \partial \mathit{res}(\mathbf{q}_{r,k+1}) / \partial \mathbf{q}_{r,k+1}$  is obtained by the product rule and is *exactly* given by

$$\begin{aligned} \mathbf{K}_{\text{dyn},k+1}(\mathbf{q}_{r,k+1}) &= \frac{1}{\beta h^2} \widetilde{\mathbf{V}}_{q_r,k+1}^\top \mathbf{M} \widetilde{\mathbf{V}}_{q_r,k+1} + \frac{\gamma}{\beta h} \left( 2\widetilde{\mathbf{V}}_{q_r,k+1}^\top \mathbf{M} d\widetilde{\mathbf{V}}_{q_r,k+1} \dot{\mathbf{q}}_{r,k+1} + \widetilde{\mathbf{V}}_{q_r,k+1}^\top \mathbf{D} \widetilde{\mathbf{V}}_{q_r,k+1} \right) \\ &\quad + \widetilde{\mathbf{V}}_{q_r,k+1}^\top \mathbf{K}(\boldsymbol{\nu}(\mathbf{q}_{r,k+1})) \widetilde{\mathbf{V}}_{q_r,k+1} \\ &\quad + \widetilde{\mathbf{V}}_{q_r,k+1}^\top \left( \mathbf{M} d\widetilde{\mathbf{V}}_{q_r,k+1} \ddot{\mathbf{q}}_{r,k+1} + \mathbf{D} d\widetilde{\mathbf{V}}_{q_r,k+1} \dot{\mathbf{q}}_{r,k+1} \right) + d\widetilde{\mathbf{V}}_{q_r,k+1}^\top \mathit{res}_{\text{full}}(\mathbf{q}_{r,k+1}). \end{aligned} \quad (9.31)$$

The exact analytical Jacobian involves several terms. However, in our simulation runs we experienced sometimes convergence problems with this Jacobian, especially for the (one-sided clamped) cantilever beam. An action that helped was the introduction of an *approximated* Jacobian  $\mathbf{K}_{\text{dyn,approx},k+1}(\mathbf{q}_{r,k+1})$  leaving out the convective related parts [132]. The regularity of this Jacobian is essential for the computation of  $\Delta \mathbf{q}_{r,k+1}^i$  (cf. lines 22-23).

## Reduction approaches

After having presented the nonlinear Galerkin projection ansatz and the time integration of the manifold-ROM, the question is now how to compute the reduction mapping  $\boldsymbol{\nu}(\mathbf{q}_r)$ .

One possibility could be to solve the second-order partial differential equation (11.18) arising in the generalization of nonlinear moment matching to second-order systems (cf. Chapter 11). This is system-theoretically attractive, but practically difficult. Another approach could be to employ neural networks and convolutional autoencoders to learn this manifold from data. This has been done so far for nonlinear first-order systems (cf. [199, 159]), but could also be attempted for second-order models. Lastly, one can select a particular ansatz (9.29) for the projection and then determine the reduction matrices  $\mathbf{V}^{(k)}$ . This strategy has been followed in [132, 223] using a quadratic manifold approach with modes and modal derivatives.



**Algorithm 9.2** Newmark time integration scheme for manifold-ROM

**Input:** Initial conditions  $\mathbf{q}_{r,0}, \dot{\mathbf{q}}_{r,0} \in \mathbb{R}^r$ , parameters  $\gamma, \beta$ , time range  $t = [t_0, \dots, t_{\text{end}}]$ , step-size  $h$ , residual error tolerance  $\text{tol}$ , start residual  $\|\text{res}_0\| = 1$

**Output:** Set of displacements  $\mathbf{q}_{r,k}$  at discrete time steps  $t_k$  for  $k = 0, \dots, N$

- 1: **Initialization**
- 2:  $[\mathbf{q}_0, \widetilde{\mathbf{V}}_{\mathbf{q}_r}, d\widetilde{\mathbf{V}}_{\mathbf{q}_r}] \leftarrow \text{Manifold}(\mathbf{q}_{r,0})$  **Full-order solution using manifold**
- 3:  $\widetilde{\mathbf{M}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top M \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}, \quad \widetilde{\mathbf{D}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top D \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}, \quad \widetilde{\mathbf{f}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top \mathbf{f}(\mathbf{q}_0), \quad \widetilde{\mathbf{f}}_{\text{ext},r} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top \mathbf{f}_{\text{ext}}(t_0)$
- 4:  $\widetilde{\mathbf{p}}_r = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}}^\top M d\widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}} (\dot{\mathbf{q}}_{r,0} \otimes \dot{\mathbf{q}}_{r,0})$
- 5:  $\ddot{\mathbf{q}}_{r,0} = \widetilde{\mathbf{M}}_r^{-1} (\widetilde{\mathbf{f}}_{\text{ext},r} - \widetilde{\mathbf{f}}_r - \widetilde{\mathbf{D}}_r \dot{\mathbf{q}}_{r,0} - \widetilde{\mathbf{p}}_r)$  **solve LSE**
- 6:  $k \leftarrow 0$
- 7: **while**  $t < t_{\text{end}}$  **do** **time-marching loop**
- 8:    $k \leftarrow k + 1$
- 9:    $t_{k+1} = t_k + h$  **time increment**
- 10:   **prediction**
- 11:    $\mathbf{q}_{r,k+1}^i = \mathbf{q}_{r,k}^i + h \dot{\mathbf{q}}_{r,k}^i + (\frac{1}{2} - \beta) h^2 \ddot{\mathbf{q}}_{r,k}^i$
- 12:    $\dot{\mathbf{q}}_{r,k+1}^i = \dot{\mathbf{q}}_{r,k}^i + (1 - \gamma) h \ddot{\mathbf{q}}_{r,k}^i$
- 13:    $\ddot{\mathbf{q}}_{r,k+1}^i = \mathbf{0}$
- 14:   **initial residual evaluation**
- 15:    $i \leftarrow 0$
- 16:    $[\mathbf{q}_{k+1}^i, \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}, d\widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}] \leftarrow \text{Manifold}(\mathbf{q}_{r,k+1}^i)$  **Full-order solution using manifold**
- 17:    $\widetilde{\mathbf{M}}_{r,k+1}^i = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top M \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}, \quad \widetilde{\mathbf{D}}_{r,k+1}^i = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top D \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}$
- 18:    $\widetilde{\mathbf{f}}_{r,k+1}^i = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top \mathbf{f}(\mathbf{q}_{k+1}^i), \quad \widetilde{\mathbf{f}}_{\text{ext},r,k+1}^i = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top \mathbf{f}_{\text{ext}}(t_{k+1})$
- 19:    $\widetilde{\mathbf{p}}_{r,k+1}^i = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top M d\widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i} (\dot{\mathbf{q}}_{r,k+1}^i \otimes \dot{\mathbf{q}}_{r,k+1}^i)$
- 20:    $\text{res}_{k+1}^i = \widetilde{\mathbf{M}}_{r,k+1}^i \ddot{\mathbf{q}}_{r,k+1}^i + \widetilde{\mathbf{p}}_{r,k+1}^i + \widetilde{\mathbf{D}}_{r,k+1}^i \dot{\mathbf{q}}_{r,k+1}^i + \widetilde{\mathbf{f}}_{r,k+1}^i - \widetilde{\mathbf{f}}_{\text{ext},r,k+1}^i$
- 21:   **while**  $\|\text{res}_{k+1}^i\| > \text{tol}$  **do** **Newton-Raphson loop**
- 22:      $\mathbf{K}_{\text{dyn,approx},k+1}^i(\mathbf{q}_{r,k+1}^i) = \frac{1}{\beta h^2} \widetilde{\mathbf{M}}_{r,k+1}^i + \frac{\gamma}{\beta h} \widetilde{\mathbf{D}}_{r,k+1}^i + \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}^\top \mathbf{K}(\mathbf{q}_{k+1}^i) \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^i}$
- 23:      $\Delta \mathbf{q}_{r,k+1}^i = - \left( \mathbf{K}_{\text{dyn,approx},k+1}^i(\mathbf{q}_{r,k+1}^i) \right)^{-1} \text{res}_{k+1}^i$  **solve LSE**
- 24:     **correction**
- 25:      $\mathbf{q}_{r,k+1}^{i+1} = \mathbf{q}_{r,k+1}^i + \Delta \mathbf{q}_{r,k+1}^i$
- 26:      $\dot{\mathbf{q}}_{r,k+1}^{i+1} = \dot{\mathbf{q}}_{r,k+1}^i + \frac{\gamma}{\beta h} \Delta \mathbf{q}_{r,k+1}^i$
- 27:      $\ddot{\mathbf{q}}_{r,k+1}^{i+1} = \ddot{\mathbf{q}}_{r,k+1}^i + \frac{1}{\beta h^2} \Delta \mathbf{q}_{r,k+1}^i$
- 28:     **iterative residual evaluation**
- 29:      $[\mathbf{q}_{k+1}^{i+1}, \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}, d\widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}] \leftarrow \text{Manifold}(\mathbf{q}_{r,k+1}^{i+1})$  **Full-order solution using manifold**
- 30:      $\widetilde{\mathbf{M}}_{r,k+1}^{i+1} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}^\top M \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}, \quad \widetilde{\mathbf{D}}_{r,k+1}^{i+1} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}^\top D \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}$
- 31:      $\widetilde{\mathbf{f}}_{r,k+1}^{i+1} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}^\top \mathbf{f}(\mathbf{q}_{k+1}^{i+1}), \quad \widetilde{\mathbf{f}}_{\text{ext},r,k+1}^{i+1} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}^\top \mathbf{f}_{\text{ext}}(t_{k+1})$
- 32:      $\widetilde{\mathbf{p}}_{r,k+1}^{i+1} = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}}^\top M d\widetilde{\mathbf{V}}_{\mathbf{q}_{r,k+1}^{i+1}} (\dot{\mathbf{q}}_{r,k+1}^{i+1} \otimes \dot{\mathbf{q}}_{r,k+1}^{i+1})$
- 33:      $\text{res}_{k+1}^{i+1} = \widetilde{\mathbf{M}}_{r,k+1}^{i+1} \ddot{\mathbf{q}}_{r,k+1}^{i+1} + \widetilde{\mathbf{p}}_{r,k+1}^{i+1} + \widetilde{\mathbf{D}}_{r,k+1}^{i+1} \dot{\mathbf{q}}_{r,k+1}^{i+1} + \widetilde{\mathbf{f}}_{r,k+1}^{i+1} - \widetilde{\mathbf{f}}_{\text{ext},r,k+1}^{i+1}$
- 34:      $i \leftarrow i + 1$  **iter**  $\leftarrow$  **iter**  $+ 1$

The quadratic manifold ansatz that has been commonly used in the just mentioned publications is given by

$$\mathbf{q}(t) \approx \sum_{i=1}^r \phi_{i,\text{eq}} q_{r,i}(t) + \sum_{i=1}^r \sum_{j=1}^r \boldsymbol{\theta}_{ij}^{\text{sym}} q_{r,i}(t) q_{r,j}(t) = \boldsymbol{\Phi}_r \mathbf{q}_r(t) + \boldsymbol{\Theta}_{r^2}^{\text{sym}} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)), \quad (9.32)$$

where  $\boldsymbol{\theta}_{ij}^{\text{sym}}$  are the *symmetrized* version of the MDs from (9.15), i.e.  $\boldsymbol{\theta}_{ij}^{\text{sym}} = \frac{1}{2}(\boldsymbol{\theta}_{ij} + \boldsymbol{\theta}_{ji})$ . After the symmetrization, it holds  $\boldsymbol{\Theta}_{ijk}^{\text{sym}} = \boldsymbol{\Theta}_{ikj}^{\text{sym}}$ , or in other words,  $\boldsymbol{\Theta}_{\text{sym}}^{(2)} = \boldsymbol{\Theta}_{\text{sym}}^{(3)}$ . In numerical examples the authors rather employ the ansatz

$$\mathbf{q}(t) \approx \sum_{i=1}^r \phi_{i,\text{eq}} q_{r,i}(t) + \sum_{i=1}^r \sum_{j=1}^r \boldsymbol{\theta}_{s,ij} q_{r,i}(t) q_{r,j}(t) = \boldsymbol{\Phi}_r \mathbf{q}_r(t) + \boldsymbol{\Theta}_{s,r^2} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)), \quad (9.33)$$

using the popular, inherently symmetric SMDs  $\boldsymbol{\theta}_{s,ij}$  from (9.19). Exemplarily, the Jacobian  $\widetilde{\mathbf{V}}_{\mathbf{q}_r}$  and Hessian  $d\widetilde{\mathbf{V}}_{\mathbf{q}_r}$  of the quadratic manifold ansatz (9.32) are consequently given by

$$\widetilde{\mathbf{V}}_{\mathbf{q}_r} = \boldsymbol{\Phi}_r + \boldsymbol{\Theta}_{r^2}^{\text{sym}} (\mathbf{1}_r \otimes \mathbf{q}_r + \mathbf{q}_r \otimes \mathbf{1}_r), \quad d\widetilde{\mathbf{V}}_{\mathbf{q}_r} = 2 \boldsymbol{\Theta}_{r^2}^{\text{sym}}, \quad (9.34)$$

with the vector of ones  $\mathbf{1}_r = [1, \dots, 1]^\top \in \mathbb{R}^r$ .

Finally, we remind again about the differences between the basis augmentation and the quadratic manifold procedure. While the basis augmentation approach employs a linear basis  $\mathbf{V}_{\text{aug}} \in \mathbb{R}^{n \times r+r^2}$  and an ansatz

$$\mathbf{q}(t) \approx \mathbf{V}_{\text{aug}} \mathbf{q}_{r,\text{aug}}(t) = [\boldsymbol{\Phi}_r, \boldsymbol{\Theta}_{r^2}] \mathbf{q}_{r,\text{aug}}(t) \quad (9.35)$$

with the corresponding reduced coordinates  $\mathbf{q}_{r,\text{aug}}(t) \in \mathbb{R}^{r+r^2}$

$$\mathbf{q}_{r,\text{aug}}(t) = [q_{r,1}(t) \quad \cdots \quad q_{r,r}(t) \quad | \quad q_{r,11}(t) \quad \cdots \quad q_{r,1r}(t) \quad \cdots \quad q_{r,rr}(t)]^\top, \quad (9.36)$$

the quadratic manifold approach exploits the ansatz

$$\mathbf{q}(t) \approx \mathbf{V}^{(1)} \mathbf{q}_r(t) + \mathbf{V}^{(2)} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)) = \boldsymbol{\Phi}_r \mathbf{q}_r(t) + \boldsymbol{\Theta}_{r^2}^{\text{sym}} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)) \quad (9.37)$$

with the reduced coordinates  $\mathbf{q}_r(t) = [q_{r,1}(t), \dots, q_{r,r}(t)]^\top \in \mathbb{R}^r$ . This means: the augmentation of the reduction basis leads to an augmented reduced vector  $\mathbf{q}_{r,\text{aug}}(t)$  with additional reduced coordinates, whereas in the quadratic manifold ansatz these additional degrees of freedom are constrained to behave quadratically, i.e.  $q_{r,ij}(t) \stackrel{!}{=} q_{r,i}(t) q_{r,j}(t)$ . Hence, this quadratic enslavement allows for a smaller reduced order, but may be the wrong ansatz for systems that do not behave quadratically (e.g. cantilever beam [223]).

# Chapter 10

## Novel Modal Derivatives for Nonlinear Analysis and Model Reduction

In the previous chapter we have reviewed the original derivation of modal derivatives based on perturbing the linearized eigenvalue problem w.r.t. the linearization point. This derivation yields a singular linear system of equations under mass consideration, and also involves the derivatives of eigenfrequencies (cf. (9.15)). In practical applications the mass terms are usually neglected, leading to a regular LSE to compute the static modal derivatives (cf. (9.19)).

This chapter presents a novel derivation for modal derivatives based on the Volterra series representation for nonlinear structural systems. Motivated from polynomial state-space systems (Part II), we had the idea to employ the variational equation approach to the polynomial(ized) version of the nonlinear second-order system (9.1). This yields cascaded, nonlinearly coupled subsystem state equations that can be used to derive the modes and the novel modal derivatives. It turns out that our derivation using the Volterra series representation yields a calculation formula which is almost identical to the conventional definition, except for the fact that a sum/subtraction of eigenfrequencies results. Thus, the resulting LSE is only singular, if the sum/subtraction of eigenfrequencies is again an eigenfrequency. Moreover, if the eigenfrequencies cancel out the definition for the SMDs results, without having to explicitly neglect the mass terms.

In addition to the novel derivation, we discuss some possible implications and applications of the gained new derivatives for nonlinear structural dynamics: the explanation of nonlinear effects like internal resonances, the gained analytical solution via truncated Volterra series, the formulation of novel quadratic manifold approaches for model reduction, and the connection of the Volterra series approach to the Harmonic Balance method. Finally, we present some numerical results for our preliminary examination comparing the novel derivatives with the conventional ones and using all them for reduction purposes (basis augmentation and QM).

This chapter supplements our paper [73] with the content from Sections 10.4.3, 10.5 and Appendix B.

### 10.1 Polynomial system representation

First, we approximate the nonlinear function  $\mathbf{f}(\mathbf{q})$  in (9.1) by a Taylor series at the equilibrium point  $\mathbf{q}_{\text{eq}}$ :

$$\mathbf{f}(\mathbf{q}) = \mathbf{f}(\mathbf{q}_{\text{eq}}) + \frac{\partial \mathbf{f}(\mathbf{q}_{\text{eq}})}{\partial \mathbf{q}} (\mathbf{q} - \mathbf{q}_{\text{eq}}) + \frac{1}{2!} \frac{\partial^2 \mathbf{f}(\mathbf{q}_{\text{eq}})}{\partial \mathbf{q}^2} (\mathbf{q} - \mathbf{q}_{\text{eq}})^{(2)} + \frac{1}{3!} \frac{\partial^3 \mathbf{f}(\mathbf{q}_{\text{eq}})}{\partial \mathbf{q}^3} (\mathbf{q} - \mathbf{q}_{\text{eq}})^{(3)} + \dots$$

Setting  $\mathbf{q}_{\text{eq}} = \mathbf{0}$  and assuming  $\mathbf{f}(\mathbf{q}_{\text{eq}}) = \mathbf{0}$ , we can write

$$\mathbf{f}(\mathbf{q}) = \mathbf{K}_1 \mathbf{q} + \mathbf{K}_2 (\mathbf{q} \otimes \mathbf{q}) + \mathbf{K}_3 (\mathbf{q} \otimes \mathbf{q} \otimes \mathbf{q}) + \dots, \quad (10.1)$$

where the polynomial matrices are defined here as  $\mathbf{K}_k = \frac{1}{k!} \frac{\partial^k \mathbf{f}(\mathbf{q}_{\text{eq}})}{\partial \mathbf{q}^k} \in \mathbb{R}^{n \times n^k}$ , i.e. including the coefficients  $1/k!$ . Truncating the series after the cubic term yields the polynomial system

$$\mathbf{M} \ddot{\mathbf{q}}(t) + \mathbf{D} \dot{\mathbf{q}}(t) + \mathbf{K}_1 \mathbf{q}(t) + \mathbf{K}_2 (\mathbf{q}(t) \otimes \mathbf{q}(t)) + \mathbf{K}_3 (\mathbf{q}(t) \otimes \mathbf{q}(t) \otimes \mathbf{q}(t)) = \mathbf{B} \mathbf{F}(t). \quad (10.2)$$

Note that the nonlinear forces of any geometric nonlinear system exhibiting a linear St. Venant-Kirchhoff material are a cubic function of the displacements, i.e.  $\mathbf{f}(\mathbf{q}) = \mathcal{O}(\mathbf{q}^3)$ . Thus, in such case the Taylor series converges after the third term, so that the representation (10.2) is exact and not an approximation. Further note that the matrices  $\mathbf{K}_1$ ,  $\mathbf{K}_2$  and  $\mathbf{K}_3$  represent the 1-mode matricizations of the corresponding second-order  $\mathcal{K}_1 \in \mathbb{R}^{n \times n}$ , third-order  $\mathcal{K}_2 \in \mathbb{R}^{n \times n \times n}$  and fourth-order  $\mathcal{K}_3 \in \mathbb{R}^{n \times n \times n \times n}$  tensors.

As already discussed in Section 4.1.1, the polynomial matrices  $\mathbf{K}_1$ ,  $\mathbf{K}_2$  and  $\mathbf{K}_3$  can be obtained from different strategies [178, 224]: (i) they can be calculated analytically within the finite element assembly procedure, (ii) numerically via finite differences, or (iii) identified via e.g. the Implicit Condensation and Expansion (ICE) method [120] or the Enforced Displacement (ED) method [182, 210]. The tensors are *symmetric*, since they originate from the elastic potential  $\mathcal{V} = \mathcal{O}(\mathbf{q}^4)$  as follows:

$$\begin{aligned} \mathbf{f}(\mathbf{q}) &= \frac{\partial \mathcal{V}}{\partial \mathbf{q}}, & \mathbf{K}_1 &= \frac{\partial \mathbf{f}}{\partial \mathbf{q}} = \frac{\partial^2 \mathcal{V}}{\partial \mathbf{q}^2}, & \mathbf{K}_2 &= \frac{1}{2} \frac{\partial^2 \mathbf{f}}{\partial \mathbf{q}^2} = \frac{1}{2} \frac{\partial^3 \mathcal{V}}{\partial \mathbf{q}^3}, & \mathbf{K}_3 &= \frac{1}{6} \frac{\partial^3 \mathbf{f}}{\partial \mathbf{q}^3} = \frac{1}{6} \frac{\partial^4 \mathcal{V}}{\partial \mathbf{q}^4}, \\ f_a &= \frac{\partial \mathcal{V}}{\partial q_a}, & K_{1ab} &= \frac{\partial f_a}{\partial q_b} = \frac{\partial^2 \mathcal{V}}{\partial q_a \partial q_b}, & K_{2abc} &= \frac{1}{2} \frac{\partial^2 f_a}{\partial q_b \partial q_c} = \frac{1}{2} \frac{\partial^3 \mathcal{V}}{\partial q_a \partial q_b \partial q_c}, & K_{3abcd} &= \frac{1}{6} \frac{\partial^3 f_a}{\partial q_b \partial q_c \partial q_d} = \dots \end{aligned}$$

for  $a, b, c, d = 1, \dots, n$ .

## 10.2 Variational equations

The Volterra series representation [221] allows to describe a nonlinear system by an infinite series of cascaded, linear *subsystems*. The solution of the nonlinear system is then given by the sum over all sub-solutions:  $\mathbf{q}(t) = \sum_{k=1}^{\infty} \mathbf{q}_k(t)$ . To obtain the subsystem state equations, the so-called *variational equation approach* [221] is employed (cf. Section 4.3.2). To this end, it is assumed that the response of the system to an input of the form  $\alpha \mathbf{F}(t)$  can be written as a sum of sub-responses:

$$\begin{aligned} \mathbf{q}(t) &= \alpha \mathbf{q}_1(t) + \alpha^2 \mathbf{q}_2(t) + \dots, \\ \ddot{\mathbf{q}}(t) &= \alpha \ddot{\mathbf{q}}_1(t) + \alpha^2 \ddot{\mathbf{q}}_2(t) + \dots \end{aligned} \quad (10.3)$$

Inserting the assumed input and the assumed response into (10.2) yields (for  $\mathbf{D} = \mathbf{0}$ )

$$\begin{aligned} \mathbf{M}(\alpha \ddot{\mathbf{q}}_1(t) + \alpha^2 \ddot{\mathbf{q}}_2(t) + \dots) + \mathbf{K}_1(\alpha \mathbf{q}_1(t) + \alpha^2 \mathbf{q}_2(t) + \dots) \\ + \mathbf{K}_2(\mathbf{q}(t) \otimes \mathbf{q}(t)) + \mathbf{K}_3(\mathbf{q}(t) \otimes \mathbf{q}(t) \otimes \mathbf{q}(t)) = \mathbf{B} \alpha \mathbf{F}(t). \end{aligned} \quad (10.4)$$

Note that, with the calculation rules for Kronecker products, it holds:

$$\begin{aligned}\mathbf{q}(t) \otimes \mathbf{q}(t) &= (\alpha \mathbf{q}_1(t) + \alpha^2 \mathbf{q}_2(t) + \dots) \otimes (\alpha \mathbf{q}_1(t) + \alpha^2 \mathbf{q}_2(t) + \dots) \\ &= \alpha^2 \mathbf{q}_1(t) \otimes \mathbf{q}_1(t) + \alpha^3 (\mathbf{q}_1(t) \otimes \mathbf{q}_2(t) + \mathbf{q}_2(t) \otimes \mathbf{q}_1(t)) + \dots, \quad (10.5) \\ \mathbf{q}(t) \otimes \mathbf{q}(t) \otimes \mathbf{q}(t) &= \alpha^3 \mathbf{q}_1(t) \otimes \mathbf{q}_1(t) \otimes \mathbf{q}_1(t) + \dots.\end{aligned}$$

Since the above differential equation must hold for all  $\alpha$ , coefficients of like powers of  $\alpha$  can be equated. This yields the variational equations (subsystem state equations):

$$\alpha : \quad \mathbf{M} \ddot{\mathbf{q}}_1(t) + \mathbf{K}_1 \mathbf{q}_1(t) = \mathbf{B} \mathbf{F}(t), \quad \mathbf{q}_1(0) = \mathbf{q}_0, \quad (10.6a)$$

$$\alpha^2 : \quad \mathbf{M} \ddot{\mathbf{q}}_2(t) + \mathbf{K}_1 \mathbf{q}_2(t) = -\mathbf{K}_2 (\mathbf{q}_1(t) \otimes \mathbf{q}_1(t)), \quad \mathbf{q}_2(0) = \mathbf{0}, \quad (10.6b)$$

$$\begin{aligned}\alpha^3 : \quad \mathbf{M} \ddot{\mathbf{q}}_3(t) + \mathbf{K}_1 \mathbf{q}_3(t) &= -\mathbf{K}_2 (\mathbf{q}_1(t) \otimes \mathbf{q}_2(t) + \mathbf{q}_2(t) \otimes \mathbf{q}_1(t)) - \mathbf{K}_3 (\mathbf{q}_1(t) \otimes \mathbf{q}_1(t) \otimes \mathbf{q}_1(t)), \quad (10.6c) \\ &\vdots\end{aligned}$$

Each  $k$ -th subsystem state equation is linear in  $\mathbf{q}_k(t)$ , but depends nonlinearly on the solution of the previous subsystems  $\mathbf{q}_{k-1}(t)$ , etc. (cf. (10.6b) and (10.6c)).

## 10.3 Derivation of modes and modal derivatives

The variational equations are exploited in the following to derive the modes and the new modal derivatives.

### 10.3.1 Modes (First subsystem)

The undamped ( $\mathbf{D} = \mathbf{0}$ ), homogeneous ( $\mathbf{F}(t) = \mathbf{0}$ ) subsystem (10.6a) is considered first. The ansatz for the solution (with  $\dot{\mathbf{q}}_1(0) = \mathbf{0}^1$ ) is given by a free oscillation with the scalings  $c_i$ , the eigenmodes  $\phi_i$  and the eigenfrequencies  $\omega_i$ :

$$\mathbf{q}_1(t) = \sum_{i=1}^n c_i \phi_i \cos(\omega_i t). \quad (10.7)$$

Inserting this ansatz (together with  $\ddot{\mathbf{q}}_1(t)$ ) into (10.6a) and canceling the time-dependent term  $c_i \cos(\omega_i t)$ , yields the quadratic eigenvalue problem

$$\left( \mathbf{K}_1 - \omega_i^2 \mathbf{M} \right) \phi_i = \mathbf{0}, \quad \forall i = 1, \dots, n. \quad (10.8)$$

The scalings  $c_i$  are determined by the initial condition of the first subsystem ( $\dot{\mathbf{q}}_1(0) = \mathbf{0}$ )

$$\mathbf{q}_1(0) = \sum_{i=1}^n c_i \phi_i \iff \Phi \mathbf{c} = \mathbf{q}_1(0) \iff \mathbf{c} = \Phi^{-1} \mathbf{q}_1(0) \quad (10.9)$$

using the fact that  $\Phi^T \mathbf{M} = \Phi^{-1}$  (due to  $\Phi^T \mathbf{M} \Phi = \mathbf{I}$ ) to solve the equation for  $\mathbf{c} = [c_1, \dots, c_n]^T$ .

<sup>1</sup> Note that for general initial conditions  $\mathbf{q}_1(0)$ ,  $\dot{\mathbf{q}}_1(0)$  the ansatz is given by  $\mathbf{q}_1(t) = \sum_{i=1}^n c_i \phi_i \cos(\omega_i t + \alpha_i)$ , or equivalently by  $\mathbf{q}_1(t) = \sum_{i=1}^n \phi_i (A_i \cos(\omega_i t) + B_i \sin(\omega_i t))$ , where  $c_i = \sqrt{A_i^2 + B_i^2}$ ,  $\alpha_i = \arctan(-\frac{B_i}{A_i})$ .

### 10.3.2 Modal derivatives (Second subsystem)

The second subsystem (10.6b) is considered now. Note that the right-hand side is composed of the quadratic coupling  $\mathbf{q}_1(t) \otimes \mathbf{q}_1(t)$ . Employing the ansatz for the first subsystem (10.7), we obtain (for  $n = 2$  and the trigonometric product-to-sum identities<sup>2</sup>):

$$\begin{aligned}
\mathbf{q}_1(t) \otimes \mathbf{q}_1(t) &= (c_1 \boldsymbol{\phi}_1 \cos(\omega_1 t) + c_2 \boldsymbol{\phi}_2 \cos(\omega_2 t)) \otimes (c_1 \boldsymbol{\phi}_1 \cos(\omega_1 t) + c_2 \boldsymbol{\phi}_2 \cos(\omega_2 t)) \\
&= c_1^2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_1) \cos^2(\omega_1 t) + c_2^2 (\boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_2) \cos^2(\omega_2 t) \\
&\quad + c_1 c_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_1) \cos(\omega_1 t) \cos(\omega_2 t) \\
&= \frac{1}{2} c_1^2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_1) (\cos(2\omega_1 t) + 1) + \frac{1}{2} c_2^2 (\boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_2) (\cos(2\omega_2 t) + 1) \\
&\quad + \frac{1}{2} c_1 c_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_1) (\cos((\omega_1 + \omega_2)t) + \cos((\omega_1 - \omega_2)t)).
\end{aligned} \tag{10.10}$$

Looking at the form of the obtained right-hand side and using the *method of undetermined coefficients*, we choose the following ansatz for the (particular) solution of the second subsystem (for  $n = 2$ ):

$$\begin{aligned}
\mathbf{q}_2(t) &= \sum_{i=1}^2 \sum_{j=1}^2 \frac{1}{2} c_i c_j \left( \tilde{\boldsymbol{\theta}}_{ij} \cos((\omega_i + \omega_j)t) + \tilde{\tilde{\boldsymbol{\theta}}}_{ij} \cos((\omega_i - \omega_j)t) \right) \\
&= \frac{1}{2} c_1^2 \left( \tilde{\boldsymbol{\theta}}_{11} \cos(2\omega_1 t) + \tilde{\tilde{\boldsymbol{\theta}}}_{11} \right) + \frac{1}{2} c_2^2 \left( \tilde{\boldsymbol{\theta}}_{22} \cos(2\omega_2 t) + \tilde{\tilde{\boldsymbol{\theta}}}_{22} \right) \\
&\quad + \frac{1}{2} c_1 c_2 \left( (\tilde{\boldsymbol{\theta}}_{12} + \tilde{\boldsymbol{\theta}}_{21}) \cos((\omega_1 + \omega_2)t) + (\tilde{\tilde{\boldsymbol{\theta}}}_{12} + \tilde{\tilde{\boldsymbol{\theta}}}_{21}) \cos((\omega_1 - \omega_2)t) \right).
\end{aligned} \tag{10.11}$$

Inserting this ansatz (together with  $\ddot{\mathbf{q}}_2(t)$ ) into (10.6b) and ordering the terms, leads to<sup>3</sup>:

$$\mathbf{0} = \frac{1}{2} c_1^2 \cos(2\omega_1 t) \underbrace{\left( (-2\omega_1)^2 \mathbf{M} + \mathbf{K}_1 \right) \tilde{\boldsymbol{\theta}}_{11} + \mathbf{K}_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_1)}_{=0} \tag{10.12a}$$

$$+ \frac{1}{2} c_1^2 \underbrace{\left( \mathbf{K}_1 \tilde{\tilde{\boldsymbol{\theta}}}_{11} + \mathbf{K}_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_1) \right)}_{=0} \tag{10.12b}$$

$$+ \frac{1}{2} c_2^2 \cos(2\omega_2 t) \underbrace{\left( (-2\omega_2)^2 \mathbf{M} + \mathbf{K}_1 \right) \tilde{\boldsymbol{\theta}}_{22} + \mathbf{K}_2 (\boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_2)}_{=0} \tag{10.12c}$$

$$+ \frac{1}{2} c_2^2 \underbrace{\left( \mathbf{K}_1 \tilde{\tilde{\boldsymbol{\theta}}}_{22} + \mathbf{K}_2 (\boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_2) \right)}_{=0} \tag{10.12d}$$

$$\begin{aligned}
+ \frac{1}{2} c_1 c_2 \cos((\omega_1 + \omega_2)t) &\underbrace{\left( (-\omega_1 - \omega_2)^2 \mathbf{M} + \mathbf{K}_1 \right) (\tilde{\boldsymbol{\theta}}_{12} + \tilde{\boldsymbol{\theta}}_{21})}_{\dots} \\
&\quad + \underbrace{\mathbf{K}_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_1)}_{=0}
\end{aligned} \tag{10.12e}$$

$$\begin{aligned}
+ \frac{1}{2} c_1 c_2 \cos((\omega_1 - \omega_2)t) &\underbrace{\left( (-\omega_1 + \omega_2)^2 \mathbf{M} + \mathbf{K}_1 \right) (\tilde{\tilde{\boldsymbol{\theta}}}_{12} + \tilde{\tilde{\boldsymbol{\theta}}}_{21})}_{\dots} \\
&\quad + \underbrace{\mathbf{K}_2 (\boldsymbol{\phi}_1 \otimes \boldsymbol{\phi}_2 + \boldsymbol{\phi}_2 \otimes \boldsymbol{\phi}_1)}_{=0} .
\end{aligned} \tag{10.12f}$$

<sup>2</sup> $\cos x \cos y = 1/2 (\cos(x+y) + \cos(x-y))$  and therefore  $\cos^2 x = \cos x \cos x = 1/2 (\cos(2x) + 1)$ .

<sup>3</sup>Actually, there should be eight brackets for eight unknowns, instead of six brackets for eight unknowns. However, multiplying (10.12e) and (10.12f) out, leads to the same LSE and thus yields  $\tilde{\boldsymbol{\theta}}_{12} = \tilde{\boldsymbol{\theta}}_{21}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{12} = \tilde{\tilde{\boldsymbol{\theta}}}_{21}$ .

Since the previous equation must hold for all  $t$ , only the six brackets should be equal to zero. As  $\tilde{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ji}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ji}$  holds (cf. Appendix B), it follows from all brackets:

$$\left(\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M}\right) \tilde{\boldsymbol{\theta}}_{ij} = -\mathbf{K}_2 \left(\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j\right), \quad \forall i, j = 1, \dots, n, \quad (10.13a)$$

$$\left(\mathbf{K}_1 - (\omega_i - \omega_j)^2 \mathbf{M}\right) \tilde{\tilde{\boldsymbol{\theta}}}_{ij} = -\mathbf{K}_2 \left(\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j\right), \quad \forall i, j = 1, \dots, n. \quad (10.13b)$$

**Equivalent description for the right-hand side** The right-hand side of equation (10.13) can be equivalently represented by the right-hand side of (9.19). First, we rewrite the right-hand side of (9.19) using the Einstein notation and the definitions  $\mathcal{K}_{2abc} = \frac{1}{2} \partial^2 f_a / \partial q_b \partial q_c = \frac{1}{2} \partial K_{1ab} / \partial q_c$  and  $(\boldsymbol{\phi}_j)_c = \partial q_c / \partial \eta_j(t)$ :

$$\frac{\partial K_{1ab}}{\partial \eta_j(t)} (\boldsymbol{\phi}_i)_b = \frac{\partial K_{1ab}}{\partial q_c} \frac{\partial q_c}{\partial \eta_j(t)} (\boldsymbol{\phi}_i)_b := \frac{\partial^2 f_a}{\partial q_b \partial q_c} (\boldsymbol{\phi}_j)_c (\boldsymbol{\phi}_i)_b = 2 \mathcal{K}_{2abc} (\boldsymbol{\phi}_j)_c (\boldsymbol{\phi}_i)_b. \quad (10.14)$$

For symmetric  $\mathbf{K}_2$ , the identity  $\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j) = \mathbf{K}_2 (\boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i)$  holds (cf. Eq. (2.16)). Thus:

$$\left. \frac{\partial \mathbf{K}_1(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \boldsymbol{\phi}_i := 2 \mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j) = \mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j + \boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i). \quad (10.15)$$

Using this relationship, (10.13) can be rewritten as:

$$\left(\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M}\right) \tilde{\boldsymbol{\theta}}_{ij} = -\frac{1}{2} \left. \frac{\partial \mathbf{K}_1(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \boldsymbol{\phi}_i, \quad \forall i, j = 1, \dots, n, \quad (10.16a)$$

$$\left(\mathbf{K}_1 - (\omega_i - \omega_j)^2 \mathbf{M}\right) \tilde{\tilde{\boldsymbol{\theta}}}_{ij} = -\frac{1}{2} \left. \frac{\partial \mathbf{K}_1(\mathbf{q})}{\partial \eta_j(t)} \right|_{\mathbf{q}_{\text{eq}}} \boldsymbol{\phi}_i, \quad \forall i, j = 1, \dots, n. \quad (10.16b)$$

The new modal derivatives  $\tilde{\boldsymbol{\theta}}_{ij}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij}$  describe the vibration shapes of the second subsystem associated to the oscillations  $\cos((\omega_i + \omega_j)t)$  and  $\cos((\omega_i - \omega_j)t)$  (cf.  $\mathbf{q}_2(t)$  in (10.11)).

**Comments on the gained new derivatives** Interestingly, our novel derivation yields a calculation formula (10.16) which is almost identical to the conventional equation (9.15) or rather (9.19), except for the fact that a sum/subtraction of eigenfrequencies results (the factor 1/2 on the right-hand side is only a definition issue).

The modal derivatives  $\tilde{\boldsymbol{\theta}}_{ij}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij}$  are equal, if the mass term is neglected, yielding  $\tilde{\boldsymbol{\theta}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ij} = \boldsymbol{\theta}_{s,ij}$ . More importantly: the derivatives  $\tilde{\tilde{\boldsymbol{\theta}}}_{ii}$  (where the eigenfrequencies cancel out) correspond also to the SMDs  $\boldsymbol{\theta}_{s,ij}$ , without having to explicitly neglect the mass term.

The new modal derivatives  $\tilde{\boldsymbol{\theta}}_{ij}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij}$  are symmetric ( $\tilde{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ji}$ ,  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ji}$ ), even if the mass term is not neglected. This seems more plausible than the non-symmetric MDs (9.15).

In the conventional definition (9.15) the matrix  $(\mathbf{K}_1 - \omega_i^2 \mathbf{M})$  is always singular. By contrast, the LSE (10.16) is regular as long as the sum  $\omega_i + \omega_j$  or subtraction  $|\omega_i - \omega_j|$  of eigenfrequencies is *not* an eigenfrequency of the linearized system, yielding regular matrices  $(\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M})$  or  $(\mathbf{K}_1 - (\omega_i - \omega_j)^2 \mathbf{M})$ . In other words, the LSE (10.16) is only singular, if the sum/subtraction of eigenfrequencies is again an eigenfrequency. This fact can be related to the occurrence of *internal resonances* in nonlinear structural dynamics.

## 10.4 Impact of the new modal derivatives

In this section we discuss three possible applications of the gained new derivatives for nonlinear structural dynamics. Firstly, we give an *analytical* solution for the nonlinear system (9.1) via the truncated Volterra series. This analytical (simulation-free) solution can be compared to the solution obtained via simulation to assess the validity of the Volterra series representation. Secondly, the new derivatives serve to define novel quadratic manifold approaches for model order reduction. Thirdly, we show the connection and differences between the Volterra series approach and the Harmonic Balance method using the novel derivatives.

### 10.4.1 Analytical solution via truncated Volterra series

Using the ansatz for the first (10.7) and second subsystem (10.11), we can give an analytical solution in terms of the modes and new modal derivatives via the *truncated* Volterra series:

$$\mathbf{q}_{1:2}(t) = \underbrace{\sum_{i=1}^n c_i \boldsymbol{\phi}_i \cos(\omega_i t)}_{\mathbf{q}_1(t)} + \underbrace{\sum_{i=1}^n \sum_{j=1}^n \frac{1}{2} c_i c_j \left( \tilde{\boldsymbol{\theta}}_{ij} \cos((\omega_i + \omega_j)t) + \tilde{\tilde{\boldsymbol{\theta}}}_{ij} \cos((\omega_i - \omega_j)t) \right)}_{\mathbf{q}_2(t)}. \quad (10.17)$$

This analytical *approximated* solution can be compared with the solution of the *whole* nonlinear system. The latter could be computed e.g. (i) using the Volterra model by taking a higher (but finite) number of subsystems into account

$$\begin{aligned} & \sum_{i=1}^n c_i \boldsymbol{\phi}_i \cos(\omega_i t) + \sum_{i=1}^n \sum_{j=1}^n \frac{1}{2} c_i c_j \left( \tilde{\boldsymbol{\theta}}_{ij} \cos((\omega_i + \omega_j)t) + \tilde{\tilde{\boldsymbol{\theta}}}_{ij} \cos((\omega_i - \omega_j)t) \right) + \dots \\ & = \sum_{k=1}^{\infty} \mathbf{q}_k(t) \approx \sum_{k=1}^N \mathbf{q}_k(t) =: \mathbf{q}_{1:N}(t), \end{aligned} \quad (10.18)$$

(ii) by superposing the nonlinear normal modes (NNMs) computed via Harmonic Balance/shooting and path continuation methods yielding  $\mathbf{q}_{\text{NNM}}(t)$ , or (iii) via an expensive simulation of the nonlinear system to obtain the true (numerical) solution  $\mathbf{q}_{\text{sim}}(t)$ . Moreover, each nonlinear mode (e.g. the first one, i.e.  $i, j = 1$ ) of the first and second subsystem

$$\mathbf{q}_{1:2}^1(t) = c_1 \boldsymbol{\phi}_1 \cos(\omega_1 t) + \frac{1}{2} c_1^2 \left( \tilde{\boldsymbol{\theta}}_{11} \cos(2\omega_1 t) + \tilde{\tilde{\boldsymbol{\theta}}}_{11} \right) \quad (10.19)$$

could be compared with the corresponding nonlinear normal mode (e.g. the first one NNM1) of the whole nonlinear system.

### Reformulated analytical solution

We now define some new modal derivatives

$$\bar{\boldsymbol{\theta}}_{ij} = \frac{1}{2} (\tilde{\boldsymbol{\theta}}_{ij} + \tilde{\tilde{\boldsymbol{\theta}}}_{ij}) \quad \text{and} \quad \hat{\boldsymbol{\theta}}_{ij} = \frac{1}{2} (\tilde{\boldsymbol{\theta}}_{ij} - \tilde{\tilde{\boldsymbol{\theta}}}_{ij}), \quad (10.20)$$

to reformulate the solution  $\mathbf{q}_2(t)$  in (10.11) and (10.17) a little differently. The idea is to use trigonometric identities to rewrite the terms  $\cos((\omega_i \pm \omega_j)t)$  as *products* of cosines and sines.



Using the trigonometric product-to-sum identities<sup>4</sup>, the solution of the second subsystem can be rewritten as:

$$\begin{aligned}
\mathbf{q}_2(t) &= \sum_{i=1}^n \sum_{j=1}^n \frac{1}{2} c_i c_j \left( \tilde{\boldsymbol{\theta}}_{ij} \cos((\omega_i + \omega_j)t) + \tilde{\tilde{\boldsymbol{\theta}}}_{ij} \cos((\omega_i - \omega_j)t) \right) \\
&= \sum_{i=1}^n \sum_{j=1}^n \frac{1}{4} c_i c_j \left( \underbrace{(\tilde{\boldsymbol{\theta}}_{ij} + \tilde{\tilde{\boldsymbol{\theta}}}_{ij})}_{2\bar{\boldsymbol{\theta}}_{ij}} (\cos((\omega_i + \omega_j)t) + \cos((\omega_i - \omega_j)t)) \right. \\
&\quad \left. - \underbrace{(\tilde{\boldsymbol{\theta}}_{ij} - \tilde{\tilde{\boldsymbol{\theta}}}_{ij})}_{2\hat{\boldsymbol{\theta}}_{ij}} (-\cos((\omega_i + \omega_j)t) + \cos((\omega_i - \omega_j)t)) \right) \\
&= \sum_{i=1}^n \sum_{j=1}^n c_i c_j \left( \bar{\boldsymbol{\theta}}_{ij} \cos(\omega_i t) \cos(\omega_j t) - \hat{\boldsymbol{\theta}}_{ij} \sin(\omega_i t) \sin(\omega_j t) \right).
\end{aligned} \tag{10.21}$$

This yields the analytical truncated Volterra series solution

$$\mathbf{q}_{1:2}(t) = \underbrace{\sum_{i=1}^n c_i \boldsymbol{\phi}_i \cos(\omega_i t)}_{\mathbf{q}_1(t)} + \underbrace{\sum_{i=1}^n \sum_{j=1}^n c_i c_j \left( \bar{\boldsymbol{\theta}}_{ij} \cos(\omega_i t) \cos(\omega_j t) - \hat{\boldsymbol{\theta}}_{ij} \sin(\omega_i t) \sin(\omega_j t) \right)}_{\mathbf{q}_2(t)}. \tag{10.22}$$

This reformulated analytical solution, including products rather than sums of eigenfrequencies, will show its applications in the following subsection 10.4.2.

Note that in our paper [73, Sec. 5.2] we have discussed the equivalence of the modal derivatives  $\tilde{\boldsymbol{\theta}}_{ij}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij}$ . They are equal, if and only if the eigenfrequency  $\omega_i = 0$  or  $\omega_j = 0$ , meaning that the system has rigid-body motions. In such case,  $\bar{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ij}$  and  $\hat{\boldsymbol{\theta}}_{ij} = \mathbf{0}$  holds, and consequently the solution of the second subsystem  $\mathbf{q}_2(t)$  in (10.22) simplifies.

### 10.4.2 Novel quadratic manifold approaches for model reduction

Another possible application of the new modal derivatives could be to formulate novel quadratic manifold projection approaches for model order reduction.

#### Novel quadratic manifold approach (1)

We propose a first novel quadratic manifold approach based on equation (10.22) for the case that  $\bar{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ij}$ , i.e.  $\hat{\boldsymbol{\theta}}_{ij} = \mathbf{0}$ . Using the ansatz  $q_{r,i}(t) = c_i \cos(\omega_i t)$ ,  $i = 1, \dots, r$  for the reduced coordinates and employing the derivatives  $\bar{\boldsymbol{\theta}}_{ij} = \frac{1}{2}(\tilde{\boldsymbol{\theta}}_{ij} + \tilde{\tilde{\boldsymbol{\theta}}}_{ij})$ , it follows

$$\mathbf{q}(t) \approx \sum_{i=1}^r \boldsymbol{\phi}_i q_{r,i}(t) + \sum_{i=1}^r \sum_{j=1}^r \bar{\boldsymbol{\theta}}_{ij} q_{r,i}(t) q_{r,j}(t) = \boldsymbol{\Phi}_r \mathbf{q}_r(t) + \bar{\boldsymbol{\Theta}}_{r^2} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)). \tag{10.23}$$

Note that this approach looks similar to the previous ones (9.32) and (9.33). However, they are not the same, since  $\boldsymbol{\theta}_{ij}^{\text{sym}} \neq \bar{\boldsymbol{\theta}}_{ij} = 1/2(\tilde{\boldsymbol{\theta}}_{ij} + \tilde{\tilde{\boldsymbol{\theta}}}_{ij})$  and  $\boldsymbol{\theta}_{s,ij} \neq \hat{\boldsymbol{\theta}}_{ij} = 1/2(\tilde{\boldsymbol{\theta}}_{ij} - \tilde{\tilde{\boldsymbol{\theta}}}_{ij})$ .

<sup>4</sup> $\cos x \cos y = 1/2(\cos(x+y) + \cos(x-y))$  and  $\sin x \sin y = 1/2(-\cos(x+y) + \cos(x-y))$ .

### Novel quadratic manifold approach (2)

We propose a second novel quadratic manifold approach based on equation (10.22). We again use the ansatz  $q_{r,i}(t) = c_i \cos(\omega_i t)$ ,  $i = 1, \dots, r$  for the reduced displacements, but now together with the corresponding reduced velocities  $\dot{q}_{r,i}(t) = -c_i \omega_i \sin(\omega_i t)$ . This leads to

$$\begin{aligned} \mathbf{q}(t) &\approx \sum_{i=1}^r \phi_i q_{r,i}(t) + \sum_{i=1}^r \sum_{j=1}^r \bar{\theta}_{ij} q_{r,i}(t) q_{r,j}(t) - \frac{1}{\omega_i \omega_j} \hat{\theta}_{ij} \dot{q}_{r,i}(t) \dot{q}_{r,j}(t) \\ &= \Phi_r \mathbf{q}_r(t) + \bar{\Theta}_{r,2} (\mathbf{q}_r(t) \otimes \mathbf{q}_r(t)) - \hat{\Theta}_{r,2} (\dot{\mathbf{q}}_r(t) \otimes \dot{\mathbf{q}}_r(t)), \end{aligned} \quad (10.24)$$

where the factor  $1/\omega_i \omega_j$  is included to exactly obtain the same solution as in (10.22). Interestingly, this factor is often used for weighting and ranking the conventional MDs (cf. [224], [73, Sec. 3.4.2]). The (until now) rather heuristic employment of this factor could be perhaps explained more system-theoretically by the analytical solution (10.22) and the quadratic manifold approach (10.24).

In contrast to the other approaches, note that the above ansatz also considers the quadratic coupling of the velocities  $\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r$ . Thus, the ansatz is both displacement- and velocity-dependent, i.e.  $\mathbf{q}(t) \approx \boldsymbol{\nu}(\mathbf{q}_r(t), \dot{\mathbf{q}}_r(t))$ . To the best of the author's knowledge such a nonlinear manifold has not been proposed in the model reduction community before. We believe that this ansatz could be advantageous for *advection-dominated phenomena* possessing a quadratic term in the velocities. Thus, it has potential for further research and application in the field. However, the projection framework (and the time integration scheme) for such an ansatz becomes even more difficult than for  $\mathbf{q}(t) \approx \boldsymbol{\nu}(\mathbf{q}_r(t))$ . This is illustrated in Appendix B.

### 10.4.3 Connection to the Harmonic Balance method

In the following, we will show the connection of the novel derivatives (derived from the variational equations and Volterra series ansatz) to the well-known Harmonic Balance method.

#### Harmonic Balance method

The Harmonic Balance (HB) is an approximation method to compute *periodic* steady-state solutions of a nonlinear ODE system. It is usually employed to compute the NNMs *approximately* or to calculate the nonlinear frequency response function (NLFRRF) for several excitation frequencies and amplitudes. Recently, the conventional MDs have been applied to compute the NNMs and the NLFRRF efficiently, i.e. in a reduced subspace [281, 250, 251].

The key idea of the HB method is to use a truncated Fourier series ansatz and then determine the Fourier coefficients by solving a nonlinear system of algebraic equations. In this regard the HB method is conceptually different from numerical time integration, since it avoids the expensive *transient* simulation but rather allows to compute *steady-state* responses.

The (infinite) Fourier series ansatz for the displacement, velocity and acceleration is [142]

$$\mathbf{q}(t) = \operatorname{Re} \left\{ \sum_{k=0}^{\infty} \mathbf{Q}_k e^{ik\omega t} \right\}, \quad \dot{\mathbf{q}}(t) = \operatorname{Re} \left\{ \sum_{k=0}^{\infty} ik\omega \mathbf{Q}_k e^{ik\omega t} \right\}, \quad \ddot{\mathbf{q}}(t) = \operatorname{Re} \left\{ \sum_{k=0}^{\infty} -(k\omega)^2 \mathbf{Q}_k e^{ik\omega t} \right\},$$

where  $\mathbf{Q}_k$  are the sought Fourier coefficients. Inserting the (truncated) ansatz into the equations of motion (9.1a) and canceling the time-dependent part yields the residual [142]

$$\mathbf{R}_k = \left( -(k\omega)^2 \mathbf{M} + ik\omega \mathbf{D} \right) \mathbf{Q}_k + \mathbf{F}_k - \mathbf{F}_{\text{ext},k}, \quad k = 0, \dots, H \quad (10.25)$$

where  $\mathbf{F}_k$  and  $\mathbf{F}_{\text{ext},k}$  represent the Fourier coefficients of the series expansion for  $\mathbf{f}(\mathbf{q})$  and  $\mathbf{f}_{\text{ext}}$ , respectively. Due to the nonlinear coupling of the Fourier coefficients  $\mathbf{Q}_k$  in the internal force vector  $\mathbf{F}_k(\mathbf{Q}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_H)$ , the residual equations  $\mathbf{R}_k$  cannot be treated individually for each  $k$ . Instead, they have to be solved simultaneously, yielding a  $n(H+1)$  set of equations:

$$\begin{aligned} \mathbf{R}_0(\mathbf{Q}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_H) &= \mathbf{0} \\ \mathbf{R}_1(\mathbf{Q}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_H) &= \mathbf{0} \\ &\vdots \\ \mathbf{R}_H(\mathbf{Q}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_H) &= \mathbf{0}. \end{aligned} \quad (10.26)$$

The dimensions of the residuals and Fourier coefficients is  $\mathbf{R}_0, \mathbf{Q}_0 \in \mathbb{R}^n$  and  $\mathbf{R}_k, \mathbf{Q}_k \in \mathbb{C}^n$  with  $k = 1, \dots, H$ . Next, we briefly discuss how the coefficients  $\mathbf{F}_{\text{ext},k}$  and  $\mathbf{F}_k$  are calculated to set up the residual (10.25). Then, we mention how the set of equations (10.26) are solved via continuation. Note that a preliminary implementation of the HB method has been carried out within AMfe in Python during the semester thesis [Bil19].

**External forces** For the external forces  $\mathbf{f}_{\text{ext}}$  a Fourier series expansion is also assumed:

$$\mathbf{f}_{\text{ext}} = \text{Re} \left\{ \sum_{k=0}^{\infty} \mathbf{F}_{\text{ext},k} e^{ik\omega t} \right\}. \quad (10.27)$$

The Fourier coefficients are calculated in the following way. First, a fast Fourier transform (`fft` in Python) is applied to the discrete time-series  $\{\mathbf{f}_{\text{ext}}(t_n)\}_{n=0}^{N-1}$ , yielding

$$\mathbf{F}_{\text{ext},k} = \text{FFT} \left[ \{\mathbf{f}_{\text{ext}}(t_n)\} \right] = \sum_{n=0}^{N-1} \mathbf{f}_{\text{ext},n} e^{-i2\pi \frac{kn}{N}}, \quad k = 0, \dots, N-1. \quad (10.28)$$

Afterwards, the first  $H+1$  vectors  $\{\mathbf{F}_{\text{ext},k}\}_{k=0}^H$  are selected and normalized with the number of samples  $N$

$$\mathbf{F}_{\text{ext},k} = \frac{1}{N} \{\mathbf{F}_{\text{ext},k}\}_{k=0}^H, \quad k = 0, \dots, H \quad (10.29)$$

to obtain the truncated Fourier series coefficients  $\mathbf{F}_{\text{ext},k}$  required for (10.25).

**Internal forces and AFT scheme** The computation of the Fourier coefficients of the internal forces  $\mathbf{F}_k$  is not trivial. In [142] several methods are discussed, whereby the Alternating Frequency-Time (AFT) scheme is the most general one. The idea is to alternate between the frequency- and time-domain to calculate the coefficients as: [141]

$$\mathbf{F}_k = \text{FFT} \left[ \mathbf{f} \left( \text{iFFT}[\{\mathbf{Q}_k\}], \text{iFFT}[\{ik\omega \mathbf{Q}_k\}] \right) \right], \quad (10.30)$$

where FFT and iFFT represent the forward and inverse fast Fourier transform, respectively. The procedure is as follows. First, the Fourier coefficients  $\mathbf{Q}_k$  are transformed in the time-domain to get the displacement vector

$$\mathbf{q}_n = \mathbf{q}(t_n) = \text{iFFT} \left[ \{\mathbf{Q}_k\} \right] = \frac{1}{N} \sum_{k=0}^{N-1} \mathbf{Q}_k e^{i2\pi \frac{kn}{N}}, \quad n = 0, \dots, N-1. \quad (10.31)$$

The same can be done for the velocity, if the internal force vector is also dependent on it:  $\dot{\mathbf{q}}_n = \text{iFFT}[\{ik\omega \mathbf{Q}_k\}]$ . Then, the displacement (and velocity) time-series is inserted into the nonlinear function, in order to obtain the time samples  $\mathbf{f}_n = \mathbf{f}(\mathbf{q}_n, \dot{\mathbf{q}}_n, t_n)$ . Finally, this time-series is transformed in the frequency-domain with the FFT:

$$\mathbf{F}_k = \text{FFT} \left[ \{\mathbf{f}(\mathbf{q}_n, \dot{\mathbf{q}}_n, t_n)\} \right] = \sum_{n=0}^{N-1} \mathbf{f}_n e^{-i2\pi \frac{kn}{N}}, \quad k = 0, \dots, N-1. \quad (10.32)$$

Afterwards, the first  $H+1$  vectors  $\{\mathbf{F}_k\}_{k=0}^H$  are selected and normalized

$$\mathbf{F}_k = \frac{1}{N} \{\mathbf{F}_k\}_{k=0}^H, \quad k = 0, \dots, H \quad (10.33)$$

to obtain the truncated Fourier series coefficients  $\mathbf{F}_k$  required for (10.25).

**Continuation** Once the residual equations (10.26) have been set up, the remaining question is how to solve them. Usually, the HB method is used to compute e.g. the nonlinear frequency response function (NLFRF). Thus, the free parameter  $\omega$  is not sampled equidistantly a-priori, but included in the vector of unknowns  $\mathbf{X} = [\mathbf{Q}^\top, \omega]^\top$ . The residual is also augmented with an additional equation  $p(\mathbf{X})=0$  (e.g. the arc-length parametrization [241]), yielding [142]

$$\mathbf{R}_{\text{aug}}(\mathbf{X}) = \mathbf{0}, \quad \text{with} \quad \mathbf{R}_{\text{aug}}(\mathbf{X}) = [\mathbf{R}^\top(\mathbf{X}), p(\mathbf{X})]^\top, \quad \mathbf{X} = [\mathbf{Q}_0^\top, \dots, \mathbf{Q}_H^\top, \omega]^\top, \quad (10.34)$$

where  $\mathbf{R}_{\text{aug}}(\mathbf{X}) \in \mathbb{C}^{n(H+1)+1}$ ,  $\mathbf{X} \in \mathbb{C}^{n(H+1)+1}$ . To generate a sequence of solution points  $\mathbf{X}_\ell$  in the parameter range  $\omega_s \leq \omega_\ell \leq \omega_e$  the above problem is solved via *path continuation* using a predictor-corrector method. In the correction step, the nonlinear system of algebraic equations (10.34) is solved by the Newton-Raphson method, wherefore the analytical Jacobian  $\mathbf{J} = \frac{\partial \mathbf{R}_{\text{aug}}}{\partial \mathbf{X}}$  is required. The reader is referred to [Bil19] for valuable details on the computation of the Jacobian.

## Difference between the Volterra and the Harmonic Balance approach

In the Harmonic Balance method, the Fourier ansatz is substituted directly in the nonlinear system (9.1) to gain *nonlinear* algebraic equations that are solved *simultaneously*. In the Volterra approach, however, we employ the subsystem state equations (10.6) from the variational approach to derive *linear* algebraic equations that are solved *sequentially*. The Volterra approach shares several similarities with the multiple-scale analysis and perturbation methods described in [187]. On the other hand, the Harmonic Balance is a technique for approximating e.g. the NNMs (which span an invariant manifold). Comparisons between the Volterra and the Harmonic Balance approach can be found in [201, 63].

Motivated from the Harmonic Balance method, in the following we will insert the analytical ansatz (10.17) that we obtained from the Volterra approach (recall Eq. (10.10)) into the polynomial nonlinear system (10.2). For simplicity we consider only one mode ( $i, j=1$ ), but we also include the cosine term  $\cos(3\omega t)$  from the third subsystem solution  $\mathbf{q}_3(t)$ :

$$\begin{aligned} \mathbf{q}(t) &= \underbrace{c\phi \cos(\omega t)}_{\alpha} + \underbrace{c^2\tilde{\theta} \cos(2\omega t)}_{\beta} + \underbrace{c^2\tilde{\tilde{\theta}} \cos(0\omega t)}_{\gamma} + \underbrace{c^3\phi^{(3)} \cos(3\omega t)}_{\delta}, \\ \ddot{\mathbf{q}}(t) &= -c\omega^2\phi \cos(\omega t) - 4c^2\omega^2\tilde{\theta} \cos(2\omega t) - 9c^3\omega^2\phi^{(3)} \cos(3\omega t). \end{aligned} \quad (10.35)$$

The above Volterra ansatz shares similarities with a truncated ( $H=3$ ) Fourier series ansatz, where  $\tilde{\tilde{\theta}} \hat{=} \mathbf{Q}_0$ ,  $\phi \hat{=} \mathbf{Q}_1$ ,  $\tilde{\theta} \hat{=} \mathbf{Q}_2$  and  $\phi^{(3)} \hat{=} \mathbf{Q}_3$ . Note, however, that the equation (10.35) is parametrized in the amplitude/scaling  $c$  of the linear mode  $\phi$ .

In order to insert the ansatz (10.35) into the polynomial system (10.2) we need to determine multiple components. For instance:

$$\mathbf{K}_2(\mathbf{q} \otimes \mathbf{q}) = \mathbf{K}_2 \left( \alpha^{(2)} + \beta^{(2)} + \gamma^{(2)} + \delta^{(2)} + 2\alpha \otimes \beta + 2\alpha \otimes \gamma + \dots + 2\gamma \otimes \delta \right), \quad (10.36a)$$

$$\mathbf{K}_3(\mathbf{q} \otimes \mathbf{q} \otimes \mathbf{q}) = \mathbf{K}_3 \left( \alpha^{(3)} + \dots + 3\alpha^{(2)} \otimes \beta + 3\beta^{(2)} \otimes \alpha + \dots + 4\alpha \otimes \beta \otimes \gamma + \dots \right), \quad (10.36b)$$

where the individual terms are given in Appendix B. Collecting the terms for the first four cosine factors that are included in the ansatz yields:

- $c \cos(\omega t)$ :  $(\mathbf{K}_1 - \omega^2 \mathbf{M}) \phi + \mathbf{K}_2 \left( c^2 \phi \otimes \tilde{\theta} + c^2 \phi \otimes \tilde{\tilde{\theta}} + c^4 \tilde{\theta} \otimes \phi^{(3)} \right) +$   
 $\mathbf{K}_3 \left( \frac{3}{4} c^2 \phi \otimes \phi \otimes \phi + \frac{3}{4} c^4 \phi \otimes \phi \otimes \phi^{(3)} + \frac{3}{4} c^6 \tilde{\theta} \otimes \tilde{\theta} \otimes \phi^{(3)} + \frac{3}{2} c^4 \phi \otimes \tilde{\theta} \otimes \tilde{\theta} \right.$   
 $\left. + 3c^4 \phi \otimes \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} + \frac{3}{2} c^6 \phi \otimes \phi^{(3)} \otimes \phi^{(3)} + 2c^4 \phi \otimes \tilde{\theta} \otimes \tilde{\tilde{\theta}} + 2c^6 \tilde{\theta} \otimes \tilde{\tilde{\theta}} \otimes \phi^{(3)} \right)$
- $c^2 \cos(0\omega t)$ :  $\mathbf{K}_1 \tilde{\tilde{\theta}} + \mathbf{K}_2 \left( \frac{1}{2} \phi \otimes \phi + \frac{1}{2} c^2 \tilde{\theta} \otimes \tilde{\theta} + c^2 \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} + \frac{1}{2} c^4 \phi^{(3)} \otimes \phi^{(3)} \right) +$   
 $\mathbf{K}_3 \left( c^4 \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} + \frac{3}{4} c^2 \phi \otimes \phi \otimes \tilde{\theta} + \frac{3}{2} c^2 \phi \otimes \phi \otimes \tilde{\tilde{\theta}} + \frac{3}{2} c^4 \tilde{\theta} \otimes \tilde{\theta} \otimes \tilde{\tilde{\theta}} \right.$   
 $\left. + \frac{3}{2} c^3 \phi \otimes \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} + \frac{3}{2} c^4 \tilde{\theta} \otimes \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} + \frac{3}{2} c^5 \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} \otimes \phi^{(3)} + \frac{3}{2} c^6 \tilde{\tilde{\theta}} \otimes \phi^{(3)} \otimes \phi^{(3)} + c^4 \phi \otimes \tilde{\theta} \otimes \phi^{(3)} \right)$
- $c^2 \cos(2\omega t)$ :  $(\mathbf{K}_1 - 4\omega^2 \mathbf{M}) \tilde{\theta} + \mathbf{K}_2 \left( \frac{1}{2} \phi \otimes \phi + c^2 \phi \otimes \phi^{(3)} + 2c^2 \tilde{\theta} \otimes \tilde{\tilde{\theta}} \right)$   
 $\mathbf{K}_3 \left( \frac{3}{4} c^4 \tilde{\theta} \otimes \tilde{\theta} \otimes \tilde{\theta} + \frac{3}{2} c^2 \phi \otimes \phi \otimes \tilde{\theta} + \frac{3}{2} c^2 \phi \otimes \phi \otimes \tilde{\tilde{\theta}} + \frac{3}{2} c^4 \tilde{\theta} \otimes \tilde{\theta} \otimes \tilde{\tilde{\theta}} \right.$   
 $\left. + \frac{3}{2} c^6 \tilde{\theta} \otimes \phi^{(3)} \otimes \phi^{(3)} + c^4 \phi \otimes \tilde{\theta} \otimes \phi^{(3)} + 2c^4 \phi \otimes \tilde{\tilde{\theta}} \otimes \phi^{(3)} \right)$
- $c^3 \cos(3\omega t)$ :  $(\mathbf{K}_1 - 9\omega^2 \mathbf{M}) \phi^{(3)} + \mathbf{K}_2 \left( \phi \otimes \tilde{\theta} + 2\phi \otimes \tilde{\tilde{\theta}} \right)$   
 $+ \mathbf{K}_3 \left( \frac{1}{4} \phi \otimes \phi \otimes \phi + \frac{3}{4} c^6 \phi^{(3)} \otimes \phi^{(3)} \otimes \phi^{(3)} + \frac{3}{2} c^2 \phi \otimes \phi \otimes \phi^{(3)} + \frac{3}{4} c^2 \phi \otimes \tilde{\theta} \otimes \tilde{\theta} \right.$   
 $\left. + \frac{3}{2} c^4 \tilde{\theta} \otimes \tilde{\theta} \otimes \phi^{(3)} + \frac{3}{2} c^4 \tilde{\tilde{\theta}} \otimes \tilde{\tilde{\theta}} \otimes \phi^{(3)} + 2c^2 \phi \otimes \tilde{\theta} \otimes \tilde{\tilde{\theta}} \right).$  (10.37)

These are  $4n$  nonlinear equations for the unknown vectors  $\phi, \tilde{\theta}, \tilde{\tilde{\theta}}, \phi^{(3)} \in \mathbb{R}^n$  and the frequency  $\omega$ . The equations are dependent on the scaling  $c$  of mode  $\phi$ . In order to obtain a well-determined nonlinear system of equations, we may use e.g. the constraint equation  $\phi^\top \mathbf{M} \phi = 1$  or another parametrization to obtain  $4n + 1$  equations. The influence of high powers of the scaling  $c$  decreases for small amplitudes  $c \rightarrow 0$ .

Considering only the terms until the power  $c^1$  yields the well-known equation for the linear mode  $\phi$ :

$$\left(\mathbf{K}_1 - \omega^2 \mathbf{M}\right) \phi = \mathbf{0}. \quad (10.38)$$

Considering the terms until the power  $c^2$  yields the linear mode  $\phi$  and the novel modal derivatives  $\tilde{\theta}$  and  $\tilde{\tilde{\theta}}$  (the factor 1/2 is only a definition issue due to the ansatz (10.35)):

$$\begin{aligned} \left(\mathbf{K}_1 - \omega^2 \mathbf{M}\right) \phi &= \mathbf{0} \\ \mathbf{K}_1 \tilde{\tilde{\theta}} &= -\frac{1}{2} \mathbf{K}_2 (\phi \otimes \phi) \\ \left(\mathbf{K}_1 - 4\omega^2 \mathbf{M}\right) \tilde{\theta} &= -\frac{1}{2} \mathbf{K}_2 (\phi \otimes \phi). \end{aligned} \quad (10.39)$$

Considering the terms until the power  $c^3$  we see that the frequency  $\omega$  and the “linear” mode  $\phi$  become dependent on the scaling  $c$ :

$$\begin{aligned} \left(\mathbf{K}_1 - \omega^2 \mathbf{M}\right) \phi &= -c^2 \left( \mathbf{K}_2 (\phi \otimes \tilde{\theta} + \phi \otimes \tilde{\tilde{\theta}}) + \frac{1}{4} \mathbf{K}_3 (\phi \otimes \phi \otimes \phi) \right) \\ \mathbf{K}_1 \tilde{\tilde{\theta}} &= -\frac{1}{2} \mathbf{K}_2 (\phi \otimes \phi) \\ \left(\mathbf{K}_1 - 4\omega^2 \mathbf{M}\right) \tilde{\theta} &= -\frac{1}{2} \mathbf{K}_2 (\phi \otimes \phi) \\ \left(\mathbf{K}_1 - 9\omega^2 \mathbf{M}\right) \phi^{(3)} &= -\mathbf{K}_2 (\phi \otimes \tilde{\theta} + 2\phi \otimes \tilde{\tilde{\theta}}) - \frac{1}{4} \mathbf{K}_3 (\phi \otimes \phi \otimes \phi). \end{aligned} \quad (10.40)$$

This means: linear modes after Eq. (10.38) are sufficient to describe the behavior for small amplitudes (until  $c^1$ ). Modal derivatives after Eq. (10.39) are important to capture the dynamic behavior for bigger amplitudes (until  $c^2$ ). If powers  $c^3$  are also considered, then the equations (10.40) become nonlinear and the frequency  $\omega$  is *no longer* independent from the amplitude. Thus, this system needs to be solved simultaneously in contrast to the equations (10.39) that can be solved one after the other.

To summarize: the obtained equations (10.40) represent a closed-form approximate solution to the HB equations (10.26) for the case when the Volterra/Fourier series is truncated ( $H=3$ ) and the nonlinearities have a polynomial form. This analytical analysis yields insightful results on the qualitative dependency of the scaling  $c$  and on the connection of the novel modal derivatives to the HB method. In practice one could gradually increase the complexity and e.g. proceed as follows:

1. Compute the (first) eigenfrequency  $\omega$ , the mode  $\phi$  and the modal derivatives  $\tilde{\theta}, \tilde{\tilde{\theta}}$  as solution to the linear systems of equations (10.39).
2. Compute  $\omega, \phi, \tilde{\theta}, \tilde{\tilde{\theta}}$  and  $\phi^{(3)}$  for different amplitudes  $c > 0$  as solution to the nonlinear system of equations (10.40). As initial guess for small amplitudes  $c \ll 1$  one could employ the solution from (10.39). The tensors  $\mathbf{K}_2$  and  $\mathbf{K}_3$  can be computed via finite differences of the tangential stiffness matrix  $\mathbf{K}_1$  or the internal forces  $\mathbf{f}(\mathbf{q})$ .
3. Solve the HB equations (10.34) for a truncation index  $H=3$  or higher.
4. Compute the NNMs via shooting (i.e. forward time integration). The NNMs are the (numerically) exact homogeneous solutions of the nonlinear system (9.1).
5. Compare  $1 \Leftrightarrow 2 \Leftrightarrow 3 \Leftrightarrow 4$  to obtain evidence about the validity region of the different approximations 1, 2, 3 to the NNMs.

## 10.5 Numerical examination

The numerical examination of the novel modal derivatives was accomplished during the thesis [Bil19]. In this section we will only show an excerpt of the obtained results. The examination was mainly conducted on a *cantilever* and *clamped-clamped* beam model. Both beams have length  $L = 2$  m, height  $h = 5$  cm and are modeled using a St. Venant-Kirchhoff material with Young's modulus  $E = 70$  GPa, Poisson's ratio  $\nu = 0.3$  and density  $\rho = 2700 \frac{\text{kg}}{\text{m}^3}$ . The cantilever beam has  $n = 1624$  dofs, whereas the clamped-clamped beam has  $n = 1614$ . The interested reader is referred to [Bil19] for more details and further numerical results.

First of all, the computation of the symmetric tensor  $\mathbf{K}_2 \in \mathbb{R}^{n \times n^2}$  was implemented in AMfe. Since an analytical formulation of  $\mathbf{K}_2 = \frac{1}{2} \frac{\partial^2 \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}^2}$  within the assembly procedure is lengthy, we decided to accomplish the calculation via central finite differences. We applied the latter scheme to both the internal forces  $\mathbf{f}(\mathbf{q})$  and the tangential stiffness matrix  $\mathbf{K}_1$  to compare the efficiency and the results of both strategies. Due to symmetry we do not require  $n^2$  or  $n$  loop evaluations, but instead calculated only the non-redundant entries. Once the obtained  $\mathbf{K}_2$  was validated w.r.t. symmetry for an appropriate step width  $h$ , we verified numerically via the mentioned examples that the equivalent description (10.15) holds.

For the computation of the conventional modal derivatives (9.15), the Nelson's method together with the derivative of the tangential stiffness matrix  $\partial \mathbf{K}_{\text{eq}} / \partial \eta_j(t)$  were already available in AMfe. We further implemented the calculation via finite difference schemes applied to the modes, i.e.  $\boldsymbol{\theta}_{ij} = \frac{\partial \phi_i(\mathbf{q}_{\text{eq}})}{\partial \eta_j(t)}$ , and the direct method (9.18). After several comparisons, we decided to proceed with the direct method due to its numerical robustness.

The computation of the novel modal derivatives (10.13) was implemented similarly to the already available static MDs (9.19). Only the coefficient matrix had to be adapted to account for the sum/subtraction of eigenfrequencies. Despite the symmetry, we computed  $\tilde{\boldsymbol{\theta}}_{ij}$ ,  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij}$  for all  $i, j = 1, \dots, r$  to verify the expected results. The computation of  $\bar{\boldsymbol{\theta}}_{ij}$  and  $\hat{\boldsymbol{\theta}}_{ij}$  was easily accomplished according to (10.20).

Before using the different modal derivatives for reduction purposes, we first examined their (dis)similarity using the concept of subspace angles and modal assurance criterion (MAC). The MAC is a measure for the similarity (or linear dependence) between the vectors of two subspaces  $\boldsymbol{\chi} = [\boldsymbol{\chi}_1, \dots, \boldsymbol{\chi}_k] \in \mathbb{R}^{n \times k}$  and  $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_m] \in \mathbb{R}^{n \times m}$ :

$$\text{MAC}(\boldsymbol{\chi}_a, \boldsymbol{\psi}_b) = \frac{(\boldsymbol{\chi}_a^T \boldsymbol{\psi}_b)^2}{\boldsymbol{\chi}_a^T \boldsymbol{\chi}_a \boldsymbol{\psi}_b^T \boldsymbol{\psi}_b} \in [0, 1], \quad \forall a = 1, \dots, k, \quad \forall b = 1, \dots, m. \quad (10.41)$$

In our case we computed e.g.  $\text{MAC}(\boldsymbol{\theta}_{ij}, \tilde{\tilde{\boldsymbol{\theta}}}_{ij})$  for  $i, j = 1, \dots, r$ , so that the MAC matrix is of dimension  $\mathbb{R}^{r^2 \times r^2}$ . We calculated both the Auto-MAC and the Cross-MAC for all possible combinations of the modal derivatives to gain insight about their correlation. In Figures 10.1 and 10.2 the results for the cantilever beam model with  $r = 6$  are depicted. Please note the ordering according to (9.22). In general it can be observed that the derivatives corresponding to the block 30 – 35, i.e.  $\boldsymbol{\theta}_{56}, \dots, \boldsymbol{\theta}_{65}$ , are less correlated to the others. Remarkable is also that every sixth row and column has values close to zero. However, in the Auto-MAC for the static MDs one can notice the linear dependency of  $\boldsymbol{\theta}_{s,56}, \dots, \boldsymbol{\theta}_{s,65}$  to  $\boldsymbol{\theta}_{s,16}$ ,  $\boldsymbol{\theta}_{s,26}$ ,  $\boldsymbol{\theta}_{s,36}$ , etc. through the blue stripes. This is due to the fact that  $\phi_6$  is a *transverse* mode, so that the corresponding SMDs represent *in-plane* motions sharing similarity to other in-plane modes. Finally, the symmetry of the novel derivatives can be recognized via the off-diagonal entries.

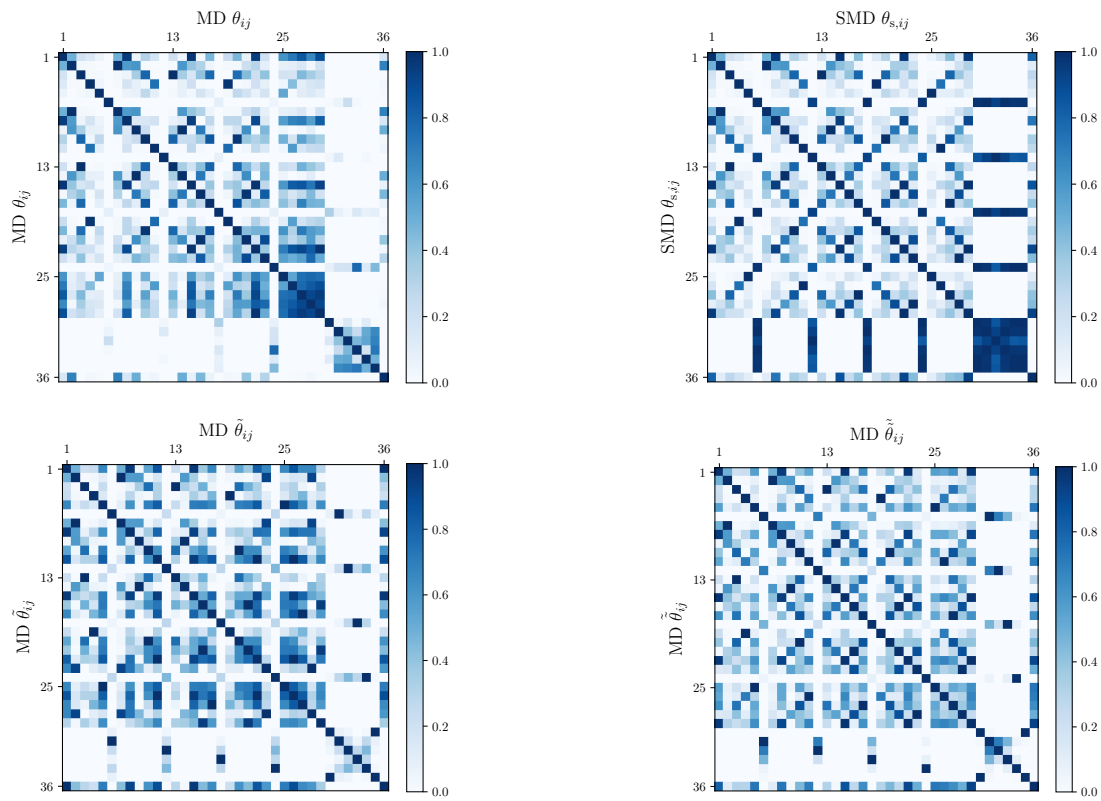


Figure 10.1: Auto-MAC of the original/static/tilde/double tilde MDs for a cantilever beam

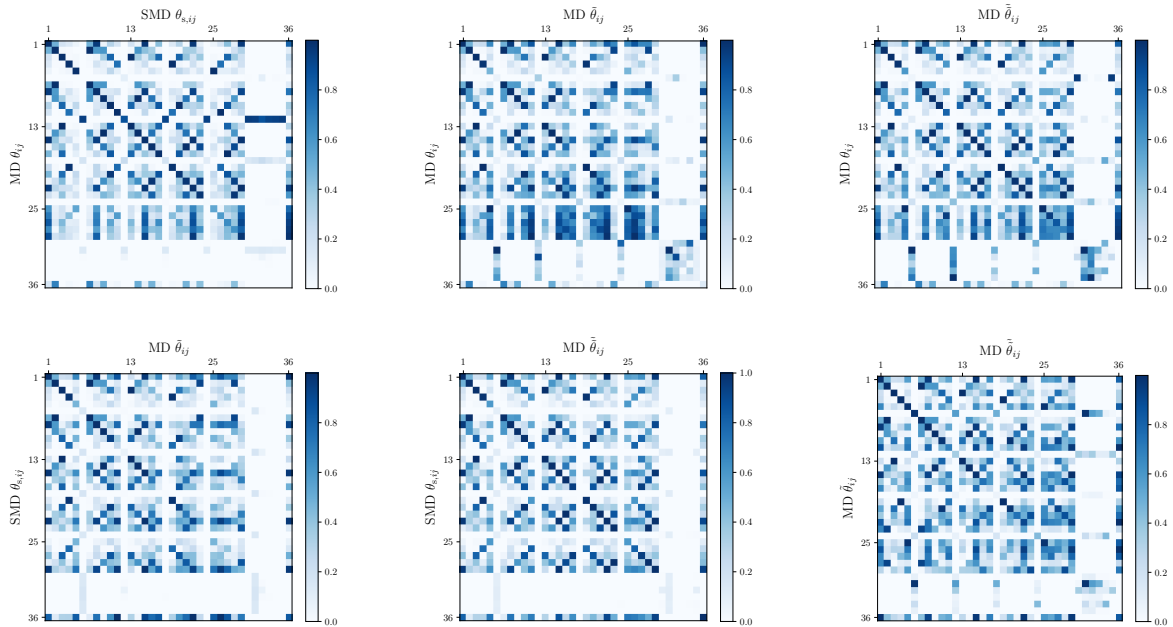


Figure 10.2: Cross-MAC of the original/static/tilde/double tilde MDs for a cantilever beam



In order to qualitatively visualize the shape of the novel modal derivatives, we computed them also for a 3D plate (leaned on [224]). Figures 10.4, 10.5, 10.6 and 10.7 show an excerpt of the calculated original, static, tilde and double tilde MDs respectively. The complete examination (for  $r = 6$ ) can be found in [Bil19]. We can see that the first three vibration modes are transverse:  $\phi_1$  is the first bending mode,  $\phi_2$  is the first torsion mode and  $\phi_3$  is the second bending mode. Thus, the corresponding (S)MDs represent longitudinal/in-plane motions. On the contrary, the fourth mode  $\phi_4$  is in-plane leading to transverse (S)MDs.

In addition to the MAC plots and the qualitative visualization, we also evaluated the derivatives with respect to their information content. To this end, we performed a singular value decomposition of the bases  $\Theta_{r,2}$ ,  $\Theta_{s,r,2}$ ,  $\tilde{\Theta}_{r,2}$  and  $\tilde{\tilde{\Theta}}_{r,2}$ . The idea is to observe how the symmetry of the static and novel modal derivatives affects the choice of the deflated reduced order. The results for the cantilever example with  $r=6$  are given in Figure 10.3. In case of the original MDs almost all  $r^2$  vectors are required to undershoot the selected tolerance. This is a consequence of the dissimilarity between the MDs, especially  $\theta_{ij} \neq \theta_{ji}$ . On the other hand, we can see that in cases 10.3b, 10.3c and 10.3d only 21 derivatives are required to undershoot the tolerance. This is in accordance with the number  $o=r(r+1)/2$  of *distinct* derivatives due to the symmetry property. Thus, we conclude that the static and new derivatives demand a smaller reduced order to achieve a certain quality threshold. In other words, in a fair comparison using the same reduced order for all subspaces the static and novel MDs are expected to (slightly) beat the original (symmetrized) MDs.

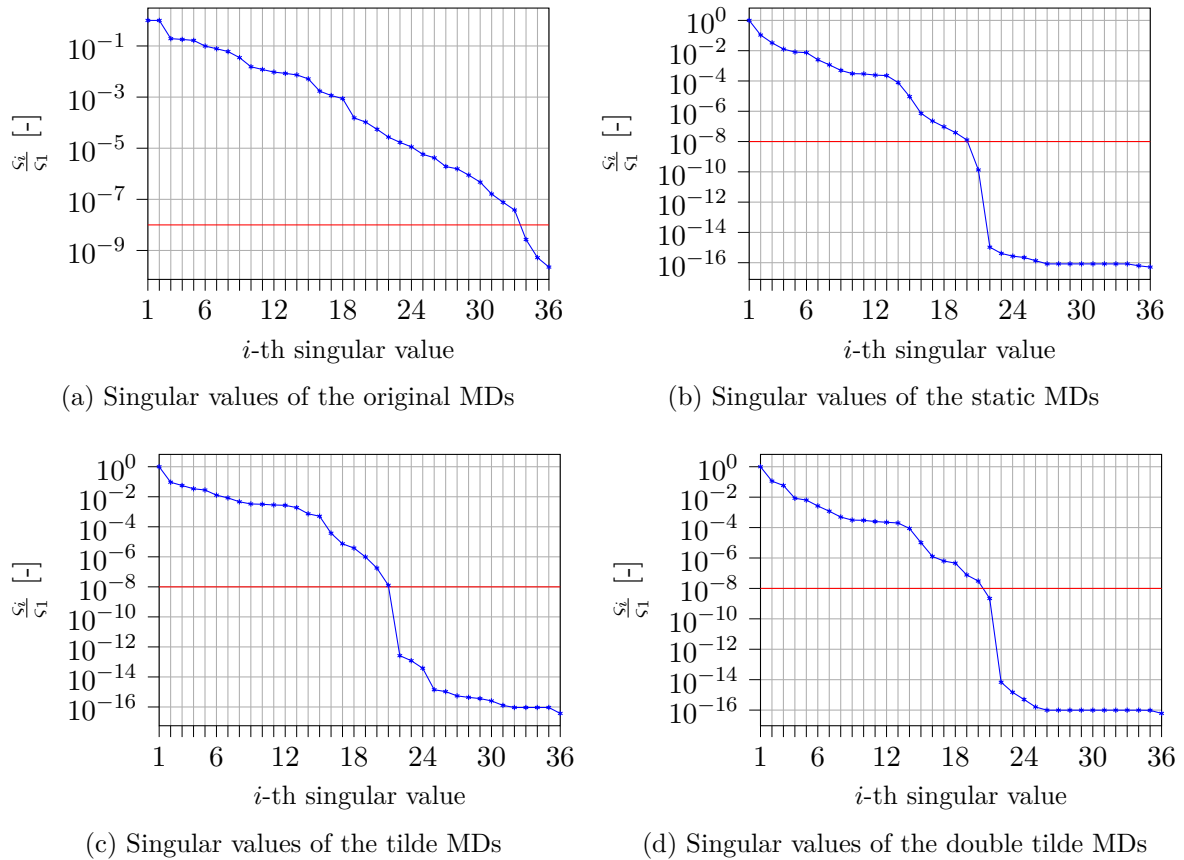


Figure 10.3: Singular values of the original/static/tilde/double tilde MDs for a cantilever beam

**Original MDs**

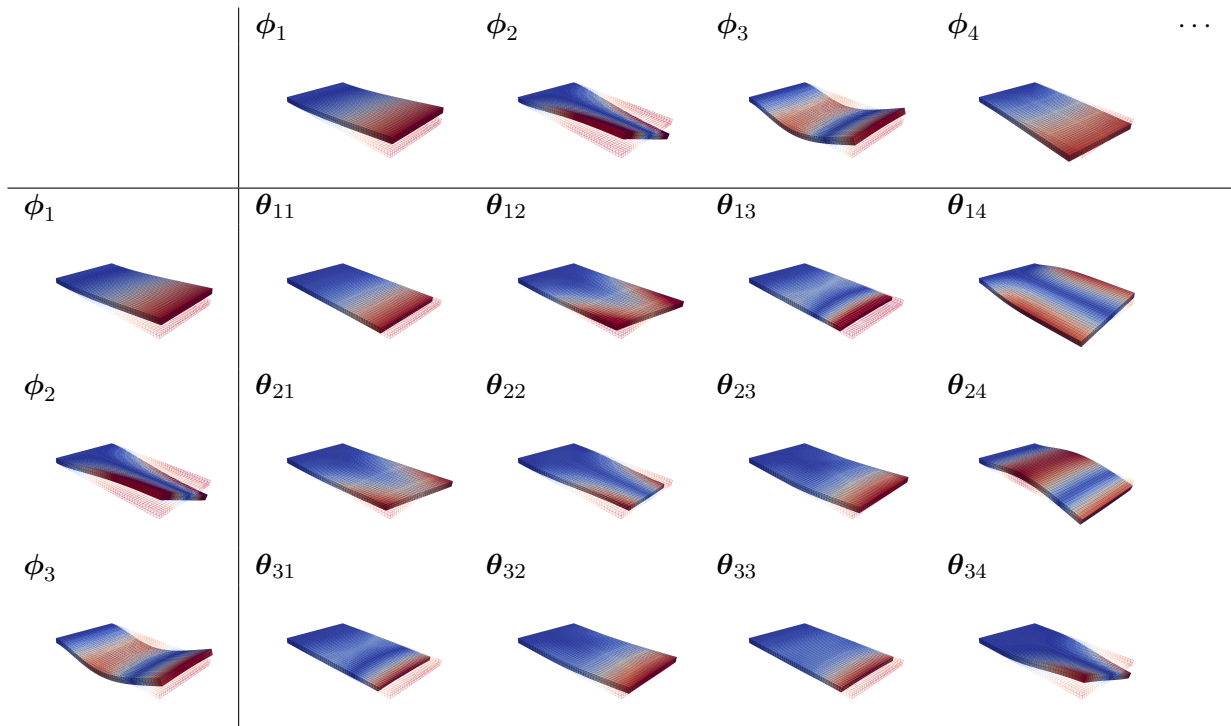


Figure 10.4: Original MDs for a plate (scalefactor VM=20, original MDs=20)

**Static MDs**

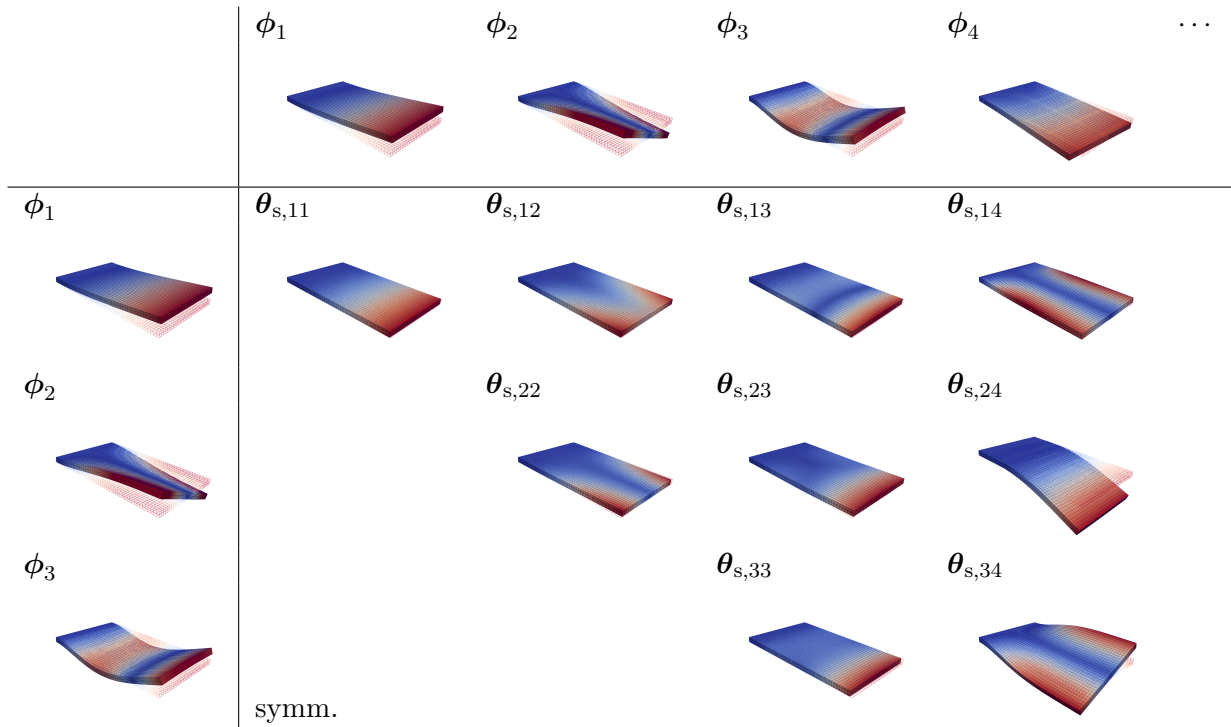


Figure 10.5: Static MDs for a plate (scalefactor VM=20, static MDs=100)

**Tilde MDs**

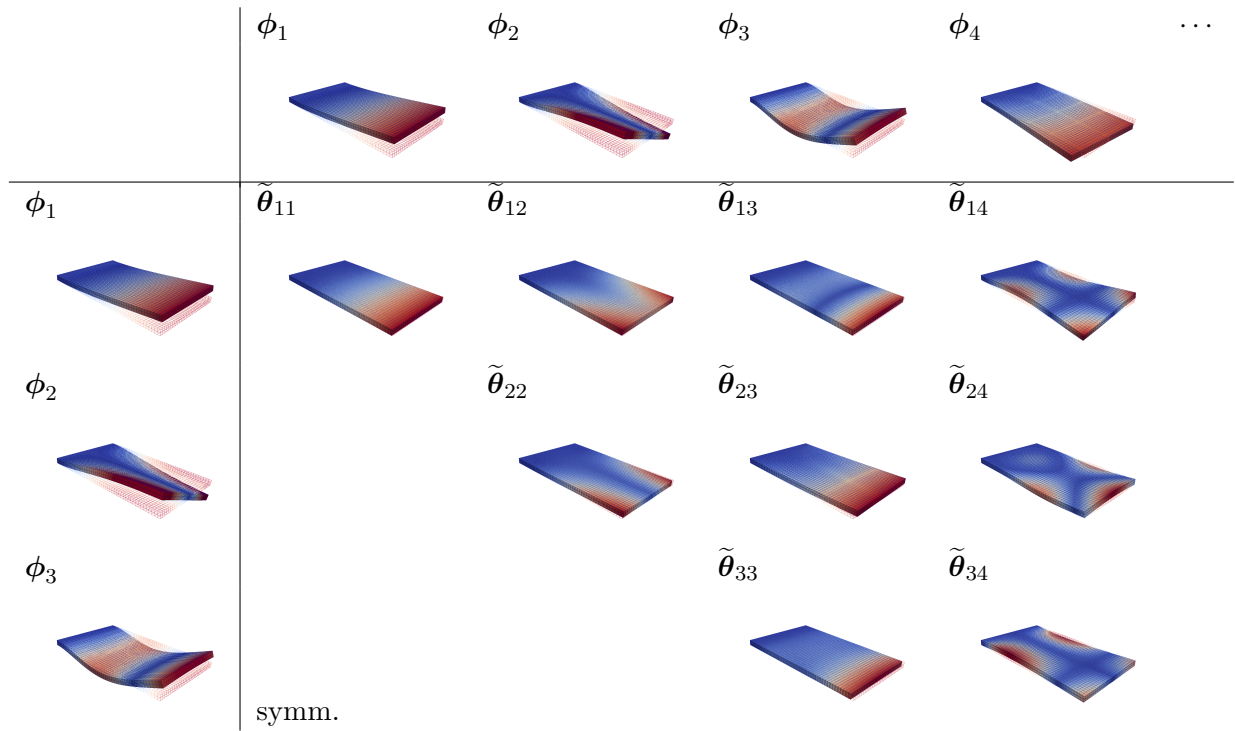


Figure 10.6: Tilde MDs for a plate (scalefactor VM=20, tilde MDs=100)

**Double tilde MDs**

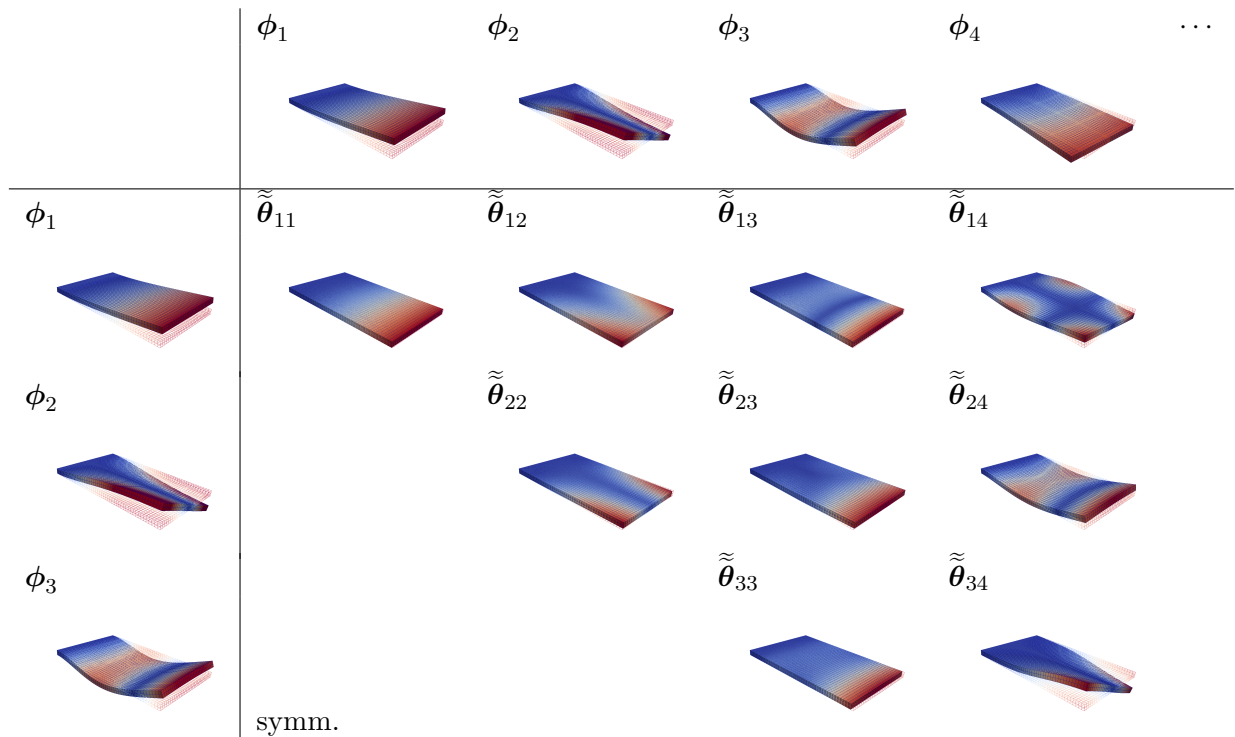


Figure 10.7: Double tilde MDs for a plate (scalefactor VM=20, double tilde MDs=100)

After the bases comparisons, we employed the MDs for reduction in both a basis augmentation (9.35) and quadratic manifold (9.37) setting. The results for the basis augmentation approach applied to the cantilever and clamped-clamped beam can be found in [Bil19]. In general it can be concluded that all derivatives perform similarly, whereby the static and novel MDs beat the original ones for equal reduced order. In the QM setting we experienced convergence problems during the ROM-simulations for the cantilever beam, as already observed by [223]. This is due to the fact that the behavior of the cantilever beam (showing large rotations) does not fit into a quadratic manifold. Thus, we only present here the results for the clamped-clamped beam. As Figures 10.8 and 10.9 show, the quadratic enslavement of the QM ansatz is perfectly suited to capture the dynamic behavior of the middle node of the beam (force  $F(t) = 5 \cdot 10^5 (\sin(72 \cdot 2\pi t) + \sin(100 \cdot 2\pi t))$  applied at the whole top boundary). All MDs show similar performance ( $r = 6$ ). Note however that the original MDs need to be symmetrized, whereas the static and novel ones are inherently symmetric.

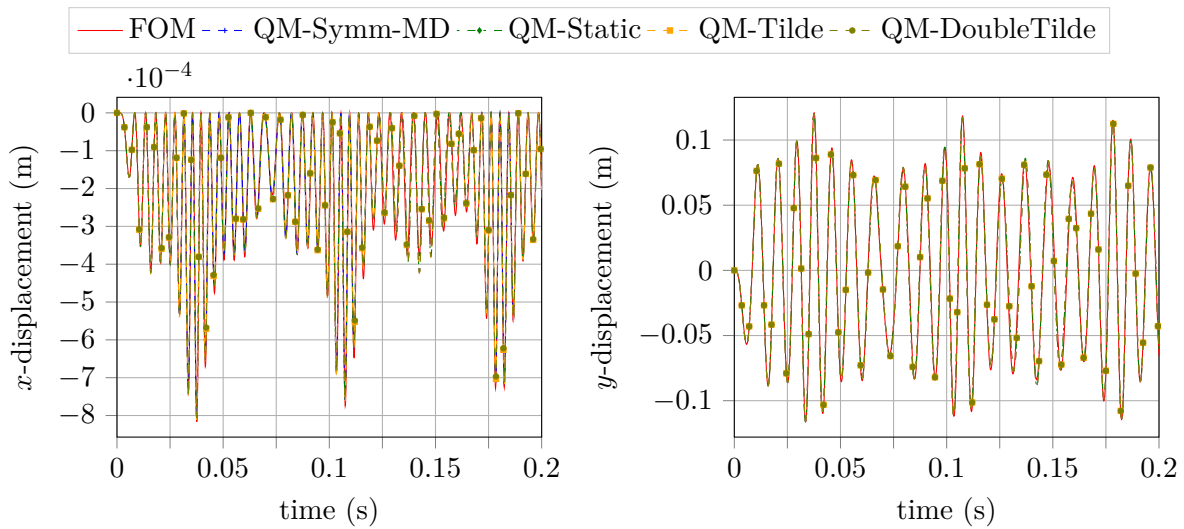


Figure 10.8: Displacements at the middle node of clamped-clamped beam in QM simulation

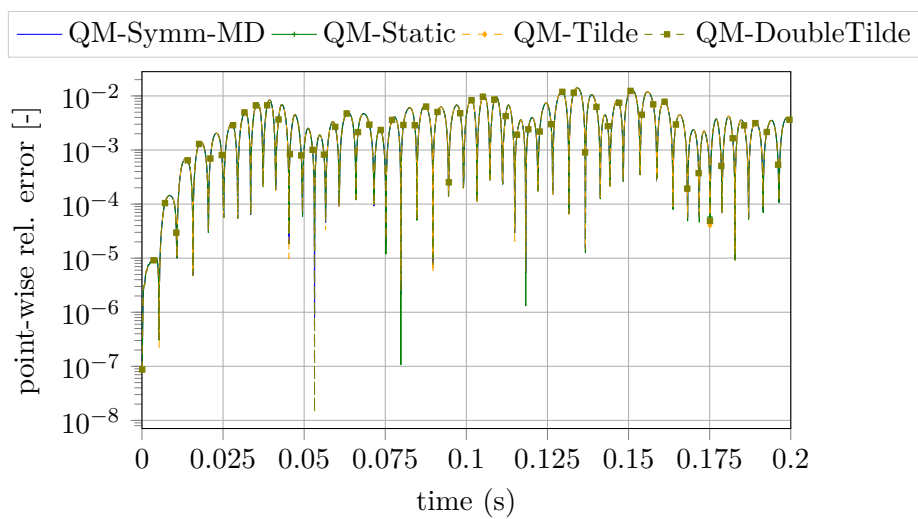


Figure 10.9: Relative error  $\frac{\|\mathbf{y}(t) - \mathbf{y}_r^m(t)\|_2}{\|\mathbf{y}(t)\|_2}$  of the clamped-clamped beam in QM simulation

## 10.6 Conclusions

This chapter presents a novel derivation for modal derivatives using the Volterra series representation of the polynomialized second-order system. In comparison to the original derivation, we believe that our approach is system-theoretically more meaningful and insightful. For instance, our novel derivation yields two regular LSEs that become singular only if the sum/subtraction of eigenfrequencies is again an eigenfrequency. This allows to retrieve the SMDs when eigenfrequencies cancel out and to explain the occurrence of internal resonances. Moreover, the novel MDs are inherently symmetric in contrast to the original ones.

The novel derivatives show promising applications for both nonlinear system analysis and model reduction. For the former purpose, the gained analytical solution (10.17) can be compared with the NNMs – approximated via the Harmonic Balance method or calculated via shooting. This way one can assess the quality and validity region of the different approximations. Moreover, the solution to the linear algebraic equations (10.39) can be compared with the solution to the nonlinear system of equations (10.40) (obtained by substituting the Volterra ansatz into the polynomial system) or to the HB residual equations (10.34).

In the context of model reduction, the new derivatives serve to formulate two novel quadratic manifold approaches. In the first one (10.23) the derivatives  $\bar{\Theta}_{r,2}$  should be employed, whereas the second one (10.24) depends quadratically on the reduced displacements **and** velocities. Both QM approaches need to be further examined in the future. Especially the second one could prove useful for advection-dominated phenomena.

The preliminary reduction results have shown that the novel derivatives perform slightly better than the original ones, and comparably well in comparison to the SMDs. The evaluation of the ROMs was accomplished in time-domain via simulation runs for certain inputs. Due to the sake of time we could not validate the ROMs in frequency-domain via the NLFRF or the NNMs. However, we could almost finish the implementation of the HB method in AMfe. In the future, the conceptual ideas and comparisons proposed here could be pursued to gain further insight about nonlinear system dynamics.



# Chapter 11

## Second-Order Nonlinear Moment Matching

In the previous chapters we have focused on the concept of modes and modal derivatives for the reduction of nonlinear mechanical systems. It was also mentioned that Krylov vectors and their corresponding perturbation derivatives can be applied as well.

In this chapter we follow a different path. We exploit the concept of moment matching for linear second-order systems and then transfer it to the nonlinear case using the center manifold theory and the ideas from Astolfi [14].

After a brief review about the frequency-domain, we will present the steady-state/time-domain interpretation of moment matching for linear second-order systems. We then transfer this reduction method to the nonlinear second-order case, hereby providing the corresponding signal generator and the second-order nonlinear Sylvester-like PDE. Afterwards, similar simplifications as for state-space systems (cf. Section 7.2) are proposed to achieve a simulation-free second-order nonlinear moment matching algorithm.

This chapter represents an edited version of the publication [75]. Certain sections are only explained briefly, since a similar discussion has already been presented for state-space systems in Section 7.3. In turn, we put more emphasis on the steady-state notion of moments and explain the numerical examples in more detail.

### 11.1 Moment matching for linear mechanical systems

We first consider a linear second-order system of the form (see (9.10) with  $\mathbf{q}_{\text{eq}} = \mathbf{0}$ ,  $\mathbf{F}_{\text{eq}} = \mathbf{0}$ )

$$\mathbf{M} \ddot{\mathbf{q}}(t) + \mathbf{D} \dot{\mathbf{q}}(t) + \mathbf{K} \mathbf{q}(t) = \mathbf{B} \mathbf{F}(t), \quad \mathbf{q}(0) = \mathbf{q}_0, \quad \dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_0, \quad (11.1a)$$

$$\mathbf{y}(t) = \mathbf{C} \mathbf{q}(t), \quad (11.1b)$$

with the non-singular symmetric positive definite mass matrix  $\mathbf{M} \in \mathbb{R}^{n \times n}$ , symmetric positive definite stiffness matrix  $\mathbf{K} \in \mathbb{R}^{n \times n}$  and similar system properties discussed in Section 9.1.

With the state vector  $\mathbf{x}(t) = [\mathbf{q}(t)^\top, \dot{\mathbf{q}}(t)^\top]^\top$  the corresponding implicit first-order (state-space) realization is given by

$$\mathcal{E} \dot{\mathbf{x}}(t) = \mathcal{A} \mathbf{x}(t) + \mathcal{B} \mathbf{u}(t), \quad \mathbf{y}(t) = \mathcal{C} \mathbf{x}(t), \quad (11.2)$$

where

$$\mathcal{E} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix}, \quad \mathcal{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K} & -\mathbf{D} \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{B} \end{bmatrix}, \quad \mathcal{C} = [\mathbf{C} \quad \mathbf{0}]. \quad (11.3)$$

This equivalent state-space representation helps us to analyze the dynamic behavior of the second-order system via the input-output equations (3.2) and (3.6), as well as to define similar system-theoretic concepts. For example, the transfer function can be derived by applying the Laplace transform to (11.1) or by using the state-space representation (11.2):

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B} = \mathbf{C}(s^2\mathbf{M} + s\mathbf{D} + \mathbf{K})^{-1}\mathbf{B}. \quad (11.4)$$

Note, however, that the state-space representation is only employed for theoretical considerations. The reduction is accomplished directly on the second-order level.

### 11.1.1 Notion of moments and Krylov subspaces

Besides modal truncation, the concept of moment matching by rational Krylov subspaces is also very common to reduce linear second-order systems (cf. [46, 30, 236]).

**Definition 11.1.** The moments  $\mathbf{m}_\ell(\sigma)$  of  $\mathbf{G}(s)$  at the complex expansion point  $\sigma \in \mathbb{C}$  are equivalent to the moments of  $\mathbf{G}(s+\sigma) = \mathbf{C}(s^2\mathbf{M} + s\mathbf{D}_\sigma + \mathbf{K}_\sigma)^{-1}\mathbf{B}$  at  $\sigma=0$ . Thus, it follows

$$\begin{aligned} \mathbf{m}_\ell(\sigma) &= (-1)^\ell \mathbf{C} \left( (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{E} \right)^\ell (\sigma\mathbf{E} - \mathbf{A})^{-1} \mathbf{B} \Big|_{\sigma=0} \\ &= (-1)^\ell [\mathbf{C} \ \mathbf{0}] \begin{bmatrix} \mathbf{K}_\sigma^{-1} \mathbf{D}_\sigma & \mathbf{K}_\sigma^{-1} \mathbf{M} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix}^\ell \begin{bmatrix} \mathbf{K}_\sigma^{-1} \mathbf{B} \\ \mathbf{0} \end{bmatrix}, \end{aligned} \quad (11.5)$$

where  $\mathbf{K}_\sigma = \mathbf{K} + \sigma\mathbf{D} + \sigma^2\mathbf{M}$  and  $\mathbf{D}_\sigma = \mathbf{D} + 2\sigma\mathbf{M}$ . For example,  $\mathbf{m}_0(\sigma) = \mathbf{C} \mathbf{K}_\sigma^{-1} \mathbf{B}$ ,  $\mathbf{m}_1(\sigma) = -\mathbf{C} \mathbf{K}_\sigma^{-1} \mathbf{D}_\sigma \mathbf{K}_\sigma^{-1} \mathbf{B}$ ,  $\mathbf{m}_2(\sigma) = \mathbf{C} \mathbf{K}_\sigma^{-1} \mathbf{D}_\sigma \mathbf{K}_\sigma^{-1} \mathbf{D}_\sigma \mathbf{K}_\sigma^{-1} \mathbf{B} - \mathbf{C} \mathbf{K}_\sigma^{-1} \mathbf{M} \mathbf{K}_\sigma^{-1} \mathbf{B}$ .  $\blacktriangle$

Depending on the considered damping, two different Krylov subspaces for second-order systems can be distinguished to achieve implicit moment matching:

- For general damping ( $\mathbf{D} \neq \mathbf{0}$ ) second-order Krylov subspaces

$$\mathcal{K}_q^{2\text{nd}} \left( \mathbf{K}_\sigma^{-1} \mathbf{D}_\sigma, \mathbf{K}_\sigma^{-1} \mathbf{M}, \mathbf{K}_\sigma^{-1} \mathbf{B} \right) \supseteq \text{ran}(\mathbf{V}), \quad (11.6a)$$

$$\mathcal{K}_q^{2\text{nd}} \left( \mathbf{K}_\mu^{-\text{T}} \mathbf{D}_\mu^\text{T}, \mathbf{K}_\mu^{-\text{T}} \mathbf{M}^\text{T}, \mathbf{K}_\mu^{-\text{T}} \mathbf{C}^\text{T} \right) \supseteq \text{ran}(\mathbf{W}), \quad (11.6b)$$

are employed. These Krylov subspaces yield a two-stage Arnoldi-like recurrence, aka. *second-order Arnoldi* (SOAR). Further details on this case are available in [46, 236].

- For proportional ( $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}$ ) or zero ( $\mathbf{D} = \mathbf{0}$ ) damping, the classical first-order Krylov subspaces

$$\mathcal{K}_q^{1\text{st}} \left( \mathbf{K}_\sigma^{-1} \mathbf{M}, \mathbf{K}_\sigma^{-1} \mathbf{B} \right) \supseteq \text{ran}(\mathbf{V}), \quad (11.7a)$$

$$\mathcal{K}_q^{1\text{st}} \left( \mathbf{K}_\mu^{-\text{T}} \mathbf{M}^\text{T}, \mathbf{K}_\mu^{-\text{T}} \mathbf{C}^\text{T} \right) \supseteq \text{ran}(\mathbf{W}), \quad (11.7b)$$

can be employed instead, yielding – exemplarily for  $\mathbf{V} = [\mathbf{V}_0, \dots, \mathbf{V}_{q-1}]$  – the one-stage Arnoldi-like recurrence

$$(\mathbf{K} + \sigma\mathbf{D} + \sigma^2\mathbf{M}) \mathbf{V}_0 = \mathbf{B}, \quad (\mathbf{K} + \sigma\mathbf{D} + \sigma^2\mathbf{M}) \mathbf{V}_\ell = \mathbf{M} \mathbf{V}_{\ell-1}, \quad \ell = 1, \dots, q-1. \quad (11.8)$$

More details are available in [30, 236]. We will focus on this case in the following.



Besides the multimoment case and block Krylov subspaces, we may also consider the multipoint setting and tangential Krylov subspaces. In case of different shifts  $\{\sigma_i\}_{i=1}^r$  and  $\{\mu_i\}_{i=1}^r$  with single multiplicities  $q_1 = \dots = q_r = 1$ , the subspaces (11.7) become

$$\text{span} \left\{ \mathbf{K}_{\sigma_1}^{-1} \mathbf{B} \mathbf{r}_1, \dots, \mathbf{K}_{\sigma_r}^{-1} \mathbf{B} \mathbf{r}_r \right\} \supseteq \text{ran}(\mathbf{V}), \quad (11.9a)$$

$$\text{span} \left\{ \mathbf{K}_{\mu_1}^{-\top} \mathbf{C}^\top \mathbf{l}_1, \dots, \mathbf{K}_{\mu_r}^{-\top} \mathbf{C}^\top \mathbf{l}_r \right\} \supseteq \text{ran}(\mathbf{W}), \quad (11.9b)$$

yielding the following tangential multipoint moment matching conditions:

$$\mathbf{G}(\sigma_i) \mathbf{r}_i = \mathbf{G}_r(\sigma_i) \mathbf{r}_i, \quad \Leftrightarrow \quad \mathbf{C} \mathbf{K}_{\sigma_i}^{-1} \mathbf{B} \mathbf{r}_i = \mathbf{C}_r \mathbf{K}_{r, \sigma_i}^{-1} \mathbf{B}_r \mathbf{r}_i, \quad i = 1, \dots, r, \quad (11.10a)$$

$$\mathbf{l}_i^\top \mathbf{G}(\mu_i) = \mathbf{l}_i^\top \mathbf{G}_r(\mu_i), \quad \Leftrightarrow \quad \mathbf{l}_i^\top \mathbf{C} \mathbf{K}_{\mu_i}^{-1} \mathbf{B} = \mathbf{l}_i^\top \mathbf{C}_r \mathbf{K}_{r, \mu_i}^{-1} \mathbf{B}_r, \quad i = 1, \dots, r. \quad (11.10b)$$

Note that (in a two-sided reduction) the reduced matrices are  $\{\mathbf{M}_r, \mathbf{D}_r, \mathbf{K}_r\} = \mathbf{W}^\top \{\mathbf{M}, \mathbf{D}, \mathbf{K}\} \mathbf{V}$ ,  $\mathbf{B}_r = \mathbf{W}^\top \mathbf{B}$ ,  $\mathbf{C}_r = \mathbf{C} \mathbf{V}$ . Convenient right and left tangential directions  $\mathbf{r}_i \in \mathbb{C}^m$  and  $\mathbf{l}_i \in \mathbb{C}^p$  should be chosen in the tangential case. Besides, the shifts  $\sigma_i, \mu_i \in \mathbb{C}$  cannot be quadratic eigenvalues of the triple  $(\mathbf{M}, \mathbf{D}, \mathbf{K})$ , i.e.  $\sigma_i, \mu_i \notin \lambda^2(\mathbf{M}, \mathbf{D}, \mathbf{K})$ .

### 11.1.2 Equivalence of Krylov subspaces and Sylvester equations

Similar to the state-space case, the bases of the input and output Krylov subspaces (11.9) can be interpreted as the solution  $\mathbf{V}$  and  $\mathbf{W}$  of the following second-order Sylvester equations:

$$\mathbf{M} \mathbf{V} \mathbf{S}_v^2 + \mathbf{D} \mathbf{V} \mathbf{S}_v + \mathbf{K} \mathbf{V} = \mathbf{B} \mathbf{R}, \quad (11.11a)$$

$$\mathbf{M}^\top \mathbf{W} \mathbf{S}_w^{2\top} + \mathbf{D}^\top \mathbf{W} \mathbf{S}_w^\top + \mathbf{K}^\top \mathbf{W} = \mathbf{C}^\top \mathbf{L}. \quad (11.11b)$$

The input interpolation data  $\{\sigma_i, \mathbf{r}_i\}$  is specified by the matrices  $\mathbf{S}_v = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{C}^{r \times r}$  and  $\mathbf{R} = [\mathbf{r}_1, \dots, \mathbf{r}_r] \in \mathbb{C}^{m \times r}$ , where the pair  $(\mathbf{R}, \mathbf{S}_v)$  is observable. Similarly, the output interpolation data  $\{\mu_i, \mathbf{l}_i\}$  is encoded in the matrices  $\mathbf{S}_w = \text{diag}(\mu_1, \dots, \mu_r) \in \mathbb{C}^{r \times r}$  and  $\mathbf{L} = [\mathbf{l}_1, \dots, \mathbf{l}_r] \in \mathbb{C}^{p \times r}$ , where the pair  $(\mathbf{S}_w, \mathbf{L}^\top)$  is controllable.

### 11.1.3 Time-domain interpretation of linear moment matching

In addition to the frequency-domain perception of moment matching (cf. (11.10)), one can also interpret this concept in the time-domain. Similar to state-space systems (cf. Section 3.6), we will first introduce a signal generator and then interpret the moments as the steady-state response of the system (11.1) interconnected with this signal generator.

#### Notion of second-order linear signal generator

Consider the linear signal generator

$$\dot{\mathbf{q}}_r^v(t) = \mathbf{S}_v \mathbf{q}_r^v(t), \quad \mathbf{q}_r^v(0) = \mathbf{q}_{r,0}^v \neq \mathbf{0}, \quad (11.12a)$$

$$\ddot{\mathbf{q}}_r^v(t) = \mathbf{S}_v \dot{\mathbf{q}}_r^v(t), \quad \dot{\mathbf{q}}_r^v(0) = \dot{\mathbf{q}}_{r,0}^v \neq \mathbf{0}, \quad (11.12b)$$

$$\mathbf{F}(t) = \mathbf{R} \mathbf{q}_r^v(t), \quad (11.12c)$$

where it is assumed that the pair  $(\mathbf{R}, \mathbf{S}_v)$  is observable,  $(\mathbf{S}_v, \mathbf{q}_{r,0}^v)$  is excitable and  $\lambda(\mathbf{S}_v) \cap \lambda^2(\mathbf{M}, \mathbf{D}, \mathbf{K}) = \emptyset$ .

### Steady-state response of interconnected system

The response of the interconnected system (11.1)+(11.12) can be given over the state-space representation (11.2) similarly to (3.82), i.e.  $\mathbf{q}(t) = \mathbf{q}_t(t) + \mathbf{q}_{ss}(t)$ . This yields the next lemma.

**Lemma 11.1** (Steady-state notion of linear input moments). The 0-th tangential moments  $\mathbf{m}_0(\sigma_i, \mathbf{r}_i)$  at  $\{\sigma_i, \mathbf{r}_i\}$  are related to the (well-defined) steady-state response

$$\begin{aligned} \mathbf{y}_{ss}(t) &= \sum_{i=1}^r \mathbf{C} \underbrace{(\sigma_i^2 \mathbf{M} + \sigma_i \mathbf{D} + \mathbf{K})^{-1} \mathbf{B} \mathbf{r}_i}_{\mathbf{v}_i} \underbrace{e^{\sigma_i t} \mathbf{q}_{r,0,i}^v}_{\mathbf{q}_{r,i}^v(t)} \\ &= \sum_{i=1}^r \mathbf{m}_0(\sigma_i, \mathbf{r}_i) e^{\sigma_i t} \mathbf{q}_{r,0,i}^v = \mathbf{C} \mathbf{V} \mathbf{q}_r^v(t), \quad \mathbf{m}_0(\sigma_i, \mathbf{r}_i) = \mathbf{C} \mathbf{v}_i, \end{aligned} \quad (11.13)$$

of the interconnected system (11.1)+(11.12), where  $\mathbf{V}$  is the unique solution of the Sylvester equation (11.11a) or  $\mathbf{v}_i = (\sigma_i^2 \mathbf{M} + \sigma_i \mathbf{D} + \mathbf{K})^{-1} \mathbf{B} \mathbf{r}_i$  is coming from an Arnoldi process.

### Moment matching by interconnection

**Theorem 11.1** (Steady-state-based linear moment matching). *Consider the interconnection of system (11.1) with the linear signal generator (11.12), where the triple  $(\mathbf{S}_v, \mathbf{R}, \mathbf{q}_{r,0}^v)$  is such that  $(\mathbf{R}, \mathbf{S}_v)$  is observable,  $(\mathbf{S}_v, \mathbf{q}_{r,0}^v)$  is excitable and  $\lambda(\mathbf{S}_v) \cap \lambda^2(\mathbf{M}, \mathbf{D}, \mathbf{K}) = \emptyset$ . Let  $\mathbf{V}$  be the solution of (11.11a) and  $\mathbf{W}$  such that  $\det(\mathbf{W}^T \mathbf{V}) \neq 0$  (e.g.  $\mathbf{W} = \mathbf{V}$ ). Furthermore, let  $\mathbf{q}_0 = \mathbf{V} \mathbf{q}_{r,0}^v$ ,  $\dot{\mathbf{q}}_0 = \mathbf{V} \dot{\mathbf{q}}_{r,0}^v$  with  $\mathbf{q}_{r,0}^v \neq \mathbf{0}$ ,  $\dot{\mathbf{q}}_{r,0}^v \neq \mathbf{0}$  arbitrary. Then, the (asymptotically stable) second-order ROM  $(\mathbf{M}_r, \mathbf{D}_r, \mathbf{K}_r, \mathbf{B}_r, \mathbf{C}_r)$  exactly matches the (well-defined) steady-state response of the output of the FOM, i.e.  $\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{y}_r(t) = \mathbf{C} \mathbf{q}(t) - \mathbf{C} \mathbf{V} \mathbf{q}_r(t) = \mathbf{0} \forall t$ .*

**Corollary 11.1.** *Consequently, moment matching for linear second-order systems can be interpreted as the exact matching of the steady-state response of the FOM and ROM*

$$\begin{aligned} \mathbf{y}_{ss}(t) &= \sum_{i=1}^r \mathbf{C} (\sigma_i^2 \mathbf{M} + \sigma_i \mathbf{D} + \mathbf{K})^{-1} \mathbf{B} \mathbf{r}_i \mathbf{q}_{r,i}^v(t) \\ &\equiv \sum_{i=1}^r \mathbf{C}_r (\sigma_i^2 \mathbf{M}_r + \sigma_i \mathbf{D}_r + \mathbf{K}_r)^{-1} \mathbf{B}_r \mathbf{r}_i \mathbf{q}_{r,i}^v(t) = \mathbf{y}_{r,ss}(t) \end{aligned} \quad (11.14)$$

when both are excited with exponential input signals  $\mathbf{F}(t) = \mathbf{R} \mathbf{q}_r^v(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{q}_{r,0}^v$ . For other input signals the steady-state response is interpolated. Note that the transient response of the FOM is also matched or rather vanishes, if the initial conditions are chosen like above.

### Derivation of second-order linear Sylvester equation

The Sylvester equation (11.11a) can be derived using the notion of signal generators. To this end, first insert the linear approximation ansatz  $\mathbf{q}(t) = \mathbf{V} \mathbf{q}_r(t)$  with  $\mathbf{q}_r(t) \stackrel{!}{=} \mathbf{q}_r^v(t)$  in the state equation (11.1a). Then, the linear signal generator (11.12) is plugged in, yielding

$$(\mathbf{M} \mathbf{V} \mathbf{S}_v^2 + \mathbf{D} \mathbf{V} \mathbf{S}_v + \mathbf{K} \mathbf{V} - \mathbf{B} \mathbf{R}) \cdot \mathbf{q}_r^v(t) = \mathbf{0}. \quad (11.15)$$

Since (11.15) holds for  $\mathbf{q}_r^v(t) = e^{\mathbf{S}_v t} \mathbf{q}_{r,0}^v$ , the state vector  $\mathbf{q}_r^v(t)$  can be factored out and the *constant* (state-independent) linear Sylvester equation (11.11a) of dimension  $n \times r$  is obtained.

## 11.2 Steady-state-based second-order nonlinear moment matching

The steady-state-based interpretation of moment matching can be carried over to nonlinear second-order systems. For the generalization we follow the same steps proposed by Astolfi for nonlinear first-order systems. We first introduce a second-order nonlinear signal generator and then interpret the moments as the steady-state of an interconnected system.

### Notion of second-order nonlinear signal generator

Consider the nonlinear signal generator

$$\dot{\mathbf{q}}_r^v(t) = \mathbf{s}_v(\mathbf{q}_r^v(t)), \quad \mathbf{q}_r^v(0) = \mathbf{q}_{r,0}^v \neq \mathbf{0}, \quad (11.16a)$$

$$\ddot{\mathbf{q}}_r^v(t) = \frac{\partial \mathbf{s}_v(\mathbf{q}_r^v(t))}{\partial \mathbf{q}_r^v(t)} \cdot \mathbf{s}_v(\mathbf{q}_r^v(t)), \quad \dot{\mathbf{q}}_r^v(0) = \dot{\mathbf{q}}_{r,0}^v \neq \mathbf{0}, \quad (11.16b)$$

$$\mathbf{F}(t) = \mathbf{r}(\mathbf{q}_r^v(t)), \quad (11.16c)$$

where  $\mathbf{s}_v(\mathbf{q}_r^v): \mathbb{R}^r \rightarrow \mathbb{R}^r$ ,  $\mathbf{r}(\mathbf{q}_r^v): \mathbb{R}^r \rightarrow \mathbb{R}^m$  are smooth mappings. It is assumed that the signal generator is observable and neutrally stable.

### Steady-state response of interconnected system

The response of the interconnected system (9.1)+(11.16) can be given using the corresponding nonlinear state-space representation (9.3) similarly to (7.2), i.e.  $\mathbf{q}(t) = \mathbf{q}_t(t) + \boldsymbol{\nu}(\mathbf{q}_r^v(t))$ . This leads to the following lemma.

**Lemma 11.2** (Steady-state notion of nonlinear input moments). The 0-th nonlinear moments  $\mathbf{m}_0(\mathbf{s}_v(\mathbf{q}_r^v(t)), \mathbf{r}(\mathbf{q}_r^v(t)), \mathbf{q}_{r,0}^v)$  at  $\{\mathbf{s}_v(\mathbf{q}_r^v), \mathbf{r}(\mathbf{q}_r^v), \mathbf{q}_{r,0}^v\}$  are related to the (locally well-defined) steady-state response

$$\mathbf{y}_{ss}(t) = \mathbf{h}(\boldsymbol{\nu}(\mathbf{q}_r^v(t))) = \mathbf{C}\boldsymbol{\nu}(\mathbf{q}_r^v(t)) := \mathbf{m}_0(\mathbf{s}_v(\mathbf{q}_r^v(t)), \mathbf{r}(\mathbf{q}_r^v(t)), \mathbf{q}_{r,0}^v) \quad (11.17)$$

of the interconnected system (9.1)+(11.16), where the mapping  $\boldsymbol{\nu}(\mathbf{q}_r^v)$ , defined in a neighborhood of  $\mathbf{q}_{r,eq}^v = \mathbf{0}$ , is the unique solution of the *second-order* Sylvester-like PDE

$$\begin{aligned} \mathbf{M} \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r^v)}{\partial \mathbf{q}_r^v} \frac{\partial \mathbf{s}_v(\mathbf{q}_r^v)}{\partial \mathbf{q}_r^v} \mathbf{s}_v(\mathbf{q}_r^v) + \mathbf{M} \frac{\partial^2 \boldsymbol{\nu}(\mathbf{q}_r^v)}{\partial \mathbf{q}_r^{v2}} \mathbf{s}_v(\mathbf{q}_r^v) \otimes \mathbf{s}_v(\mathbf{q}_r^v) \\ + \mathbf{D} \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r^v)}{\partial \mathbf{q}_r^v} \mathbf{s}_v(\mathbf{q}_r^v) + \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r^v)) = \mathbf{B} \mathbf{r}(\mathbf{q}_r^v). \end{aligned} \quad (11.18)$$

### Nonlinear moment matching by interconnection

**Theorem 11.2** (Steady-state-based nonlinear moment matching). *Consider the interconnection of system (9.1) with the nonlinear signal generator (11.16), where the triple  $(\mathbf{s}_v, \mathbf{r}, \mathbf{q}_{r,0}^v)$*

is assumed observable and neutrally stable. Let  $\boldsymbol{\nu}(\mathbf{q}_r^v)$  be the unique solution of the Sylvester-like PDE (11.18) and  $\boldsymbol{\omega}(\cdot)$  such that  $\det(\widetilde{\mathbf{W}}_{\mathbf{q}_r}^\top \widetilde{\mathbf{V}}_{\mathbf{q}_r}) \neq 0$ . Furthermore, let  $\mathbf{q}_0 = \boldsymbol{\nu}(\mathbf{q}_{r,0}^v)$ ,  $\dot{\mathbf{q}}_0 = \widetilde{\mathbf{V}}_{\mathbf{q}_{r,0}^v} \dot{\mathbf{q}}_{r,0}^v$  with  $\mathbf{q}_{r,0}^v \neq \mathbf{0}$ ,  $\dot{\mathbf{q}}_{r,0}^v \neq \mathbf{0}$  arbitrary. Then, the (exponentially stable) ROM (9.28) exactly matches the (locally well-defined) steady-state response of the output of the FOM, i.e.  $\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{y}_r(t) = \mathbf{C}\mathbf{q}(t) - \mathbf{C}\boldsymbol{\nu}(\mathbf{q}_r(t)) = \mathbf{0} \forall t$ .

**Corollary 11.2.** *Thus, nonlinear moment matching can be interpreted as the exact matching of the steady-state response of the FOM and ROM*

$$\begin{aligned} \mathbf{y}_{\text{ss}}(t) &= \mathbf{C}\boldsymbol{\nu}(\mathbf{q}_r^v(t)) = \mathbf{m}_0(\mathbf{s}_v(\mathbf{q}_r^v(t)), \mathbf{r}(\mathbf{q}_r^v(t)), \mathbf{q}_{r,0}^v), \\ &\equiv \mathbf{C}\boldsymbol{\nu}(\mathbf{q}_{r,\text{ss}}(t)) = \mathbf{m}_{r,0}(\mathbf{s}_v(\mathbf{q}_r^v(t)), \mathbf{r}(\mathbf{q}_r^v(t)), \mathbf{q}_{r,0}^v) = \mathbf{y}_{r,\text{ss}}(t), \end{aligned} \quad (11.19)$$

when both are excited with the signal generator (11.16). For other input signals the steady-state response is interpolated.

### Derivation of second-order nonlinear Sylvester-like partial differential equation

The Sylvester-like PDE (11.18) represents the nonlinear counterpart of the linear equation (11.15). Thus, the PDE has been derived similarly as follows. First, the nonlinear approximation ansatz  $\mathbf{q}(t) = \boldsymbol{\nu}(\mathbf{q}_r(t))$  with  $\mathbf{q}_r(t) \stackrel{!}{=} \mathbf{q}_r^v(t)$  is inserted in the state equation (9.1a). Afterwards, the nonlinear signal generator (11.16) is plugged in, yielding (11.18). Note that, as opposed to the linear Sylvester equation (11.11a) of dimension  $n \times r$ , the PDE (11.18) is a *nonlinear, state-dependent* equation of dimension  $n \times 1$ . Further note that the PDE contains not only first, but also second-order partial derivatives w.r.t. the unknown  $\boldsymbol{\nu}(\mathbf{q}_r^v(t))$ .

### Families of reduced-order models achieving second-order nonlinear moment matching

Similar to state-space systems (cf. Sections 3.6.4 and 7.1.4), one could also consider defining *non-projective* families of ROMs achieving second-order nonlinear moment matching. We will not deepen this further, since we mainly use the projection-based approaches (9.7) and (9.28). As already mentioned, we employ a one-sided reduction to preserve the stability of the FOM.

## 11.3 Approximated second-order nonlinear moment matching

The second-order nonlinear PDE (11.18) is difficult to solve for  $\boldsymbol{\nu}(\mathbf{q}_r^v(t))$ . Thus, in [75] we have proposed similar step-by-step simplifications as in [72] to achieve a feasible, simulation-free reduction method for nonlinear mechanical systems. Here we will not repeat the simplifications for all three signal generator cases again, as it is similar to what discussed in Section 7.2. Instead, we focus only on the most general nonlinear signal generator case.

### (i) Linear projection

Applying a linear projection  $\mathbf{q}(t) = \boldsymbol{\nu}(\mathbf{q}_r^v(t)) = \mathbf{V}\mathbf{q}_r^v(t)$ , the PDE (11.18) becomes the following algebraic nonlinear system of equations

$$\mathbf{M}\mathbf{V} \frac{\partial \mathbf{s}_v(\mathbf{q}_r^v(t))}{\partial \mathbf{q}_r^v(t)} \mathbf{s}_v(\mathbf{q}_r^v(t)) + \mathbf{D}\mathbf{V} \mathbf{s}_v(\mathbf{q}_r^v(t)) + \mathbf{f}(\mathbf{V}\mathbf{q}_r^v(t)) - \mathbf{B}\mathbf{r}(\mathbf{q}_r^v(t)) = \mathbf{0}, \quad (11.20)$$

where the triple  $(\mathbf{s}_v(\mathbf{q}_r^v(t)), \mathbf{r}(\mathbf{q}_r^v(t)), \mathbf{q}_r^v(t))$  is user-defined. Note that the term corresponding to the second-order partial derivative of  $\nu(\mathbf{q}_r^v(t))$  vanishes when a linear projection is applied.

### (ii) Column-wise consideration

System (11.20) consists of  $n$  equations for  $n \cdot r$  unknowns in  $\mathbf{V} \in \mathbb{R}^{n \times r}$ , i.e. it is underdetermined. To overcome this problem, we propose to consider the equation column-wise for each  $\mathbf{v}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, r$

$$\mathbf{M} \mathbf{v}_i \frac{\partial s_{v_i}(q_{r,i}^v(t))}{\partial q_{r,i}^v(t)} s_{v_i}(q_{r,i}^v(t)) + \mathbf{D} \mathbf{v}_i s_{v_i}(q_{r,i}^v(t)) + \mathbf{f}(\mathbf{v}_i q_{r,i}^v(t)) - \mathbf{B} \mathbf{r}_i(q_{r,i}^v(t)) = \mathbf{0}, \quad (11.21)$$

with  $q_{r,i}^v(t) \in \mathbb{R}$  and  $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_r]$ . Please be aware that, in contrast to the linear setting, a column-wise construction of  $\mathbf{V}$  using columns  $\mathbf{v}_i$  satisfying (11.21) does generally not fulfill the “true” equation (11.20).

### (iii) Time discretization

The nonlinear equation (11.21) is still state-dependent. Thus, we propose to discretize the equation with *time-snapshots*  $\{t_k^*\}$ ,  $k = 1, \dots, K$ . With user-defined  $s_{v_i}(q_{r,i}^v(t_k^*))$ ,  $\mathbf{r}_i(q_{r,i}^v(t_k^*))$  and  $q_{r,0,i}^v$ , the following time-independent equation results

$$\mathbf{M} \mathbf{v}_{ik} \frac{\partial s_{v_i}(q_{r,i}^v(t_k^*))}{\partial q_{r,i}^v(t_k^*)} s_{v_i}(q_{r,i}^v(t_k^*)) + \mathbf{D} \mathbf{v}_{ik} s_{v_i}(q_{r,i}^v(t_k^*)) + \mathbf{f}(\mathbf{v}_{ik} q_{r,i}^v(t_k^*)) - \mathbf{B} \mathbf{r}_i(q_{r,i}^v(t_k^*)) = \mathbf{0}, \quad (11.22)$$

which can be solved for each  $\mathbf{v}_{ik} \in \mathbb{R}^n$ , with  $i = 1, \dots, r$  and  $k = 1, \dots, K$ . The discrete solution  $q_{r,i}^v(t_k^*)$  of the nonlinear signal generator equation (11.16) must be given or computed via simulation before solving equation (11.22).

These simplifications yield the second-order nonlinear moment matching algorithm 11.1:

---

#### Algorithm 11.1 Second-order NLMM (SO-NLMM)

---

**Input:**  $\mathbf{M}$ ,  $\mathbf{D}$ ,  $\mathbf{f}(\mathbf{q})$ ,  $\mathbf{B}$ ,  $\mathbf{K}(\mathbf{q})$ ,  $q_{r,i}^v(t_k^*)$ ,  $\dot{q}_{r,i}^v(t_k^*)$ ,  $\ddot{q}_{r,i}^v(t_k^*)$ ,  $\mathbf{r}_i(q_{r,i}^v(t_k^*))$ , initial guesses  $\mathbf{v}_{0,ik}$ , deflated order  $r_{\text{def}}$

**Output:** orthogonal basis  $\mathbf{V}$

- 1: **for**  $i = 1 : r$  **do**      ▶ e.g.  $r$  different shifts  $\sigma_i$
  - 2:    **for**  $k = 1 : K$  **do**    ▶ e.g.  $K$  samples in each shift
  - 3:      $\text{fun} = @(\mathbf{v}) \mathbf{M} \mathbf{v} \ddot{q}_{r,i}^v(t_k^*) + \mathbf{D} \mathbf{v} \dot{q}_{r,i}^v(t_k^*) + \mathbf{f}(\mathbf{v} q_{r,ik}^v) - \mathbf{B} \mathbf{r}_i(q_{r,ik}^v)$       ▶ residual (11.22)
  - 4:      $\text{Jfun} = @(\mathbf{v}) \mathbf{M} \ddot{q}_{r,i}^v(t_k^*) + \mathbf{D} \dot{q}_{r,i}^v(t_k^*) + \mathbf{K}(\mathbf{v} q_{r,ik}^v) q_{r,ik}^v$       ▶ Jacobian of residual
  - 5:      $\mathbf{v}_{ik} = \text{NewtonRaphson}(\text{fun}, \mathbf{v}_{0,ik}, \text{Jfun})$       ▶ call Alg. 6.2
  - 6:      $\mathbf{V}(:, (i-1)*K+k) \leftarrow \mathbf{v}_{ik}$
  - 7:      $\mathbf{V} = \text{gramSchmidt}(\mathbf{v}_{ik}, \mathbf{V})$       ▶ optional
  - 8:  $[\mathbf{U}, \text{Sigma}, \sim] = \text{svd}(\mathbf{V}, \text{'econ'})$ ;  $\mathbf{V} = \mathbf{U}(:, 1:r_{\text{def}})$       ▶ deflation is optional
- 

Since the discussion regarding the computational aspects, the approximated moments and the limitations also applies here, we refrain from further details and instead refer the reader to [75] and Sections 7.2 and 7.3.

## 11.4 Numerical examples

The SO-NLMM algorithm is illustrated by means of two numerical examples: a cantilever beam and a S-shaped structure. A third example using a similar clamped-clamped beam as in Section 10.5 is not shown for the sake of brevity.

We use the open-source research code AMfe (cf. [224]) for the setup and numerical simulation of the finite element models. Gmsh [104] and ParaView [5] serve hereby as mesh generation and post-processing tools, respectively. For the numerical time integration of FOMs and ROMs (9.7) we employ the implicit generalized- $\alpha$  scheme 9.1 with default parameters.

The SO-NLMM algorithm 11.1 has been implemented in Python using a self-programmed Newton-Raphson scheme.<sup>1</sup> We compare SO-NLMM with POD and the basis augmentation approach with vibration modes (VMs) and modal derivatives (MDs).

### 11.4.1 Cantilever beam

The cantilever beam has a length of  $L = 3$  m and a height of  $h = 10$  cm. It is made of steel, which is modeled as linear Kirchhoff material with  $E = 210$  GPa,  $\nu = 0.3$  and  $\rho = 1 \cdot 10^4 \frac{\text{kg}}{\text{m}^3}$ . The 2D model is discretized using 246 triangular Tri6 elements with quadratic shape functions, yielding (after Dirichlet boundary conditions)  $n = 1224$  degrees of freedom in  $x$ - and  $y$ -direction. The model exhibits *geometric nonlinear* behavior due to the used quadratic Green-Lagrange strain tensor, resulting in a *cubic* function  $\mathbf{f}(\mathbf{q})$  of the nodal displacements. The model equation is given by (9.1), where  $\mathbf{D} = \mathbf{0}$  is assumed. The input force  $F(t)$  is applied at the tip in negative  $y$ -direction. The output  $y(t)$  is the  $y$ -displacement of the tip.

We apply Algorithm 11.1 using a single signal generator with  $K = 10$  or  $K = 20$  equidistant time-snapshots in the interval  $t \in [0, 1]$  s. For the *training phase* of SO-NLMM and POD, we use the signal generator  $q_r^v(t) = \sin(10t)$  – i.e.  $\omega_{\text{train}} = 10 \frac{\text{rad}}{\text{s}}$  – with corresponding  $\dot{q}_r^v(t)$ ,  $\ddot{q}_r^v(t)$  and the training input  $F_{\text{train}}(t) = r(q_r^v(t)) = 10^8 \cdot q_r^v(t)$ . For the *test phase* of FOM and ROMs we apply the different input  $F_{\text{test}}(t) = 10^8 \cdot \sin(31t)$  – i.e.  $\omega_{\text{test}} = 31 \frac{\text{rad}}{\text{s}}$ . We compare both approaches with ROMs obtained via  $\mathbf{V}_\phi$  containing only linear vibration modes  $\phi_i$ , and an augmented basis  $\mathbf{V}_{\text{aug}}$  containing VMs and SMDs – for  $\mathbf{q}_{\text{eq}} = \mathbf{0}$  – to capture the nonlinear behavior. The numerical integration of FOM and ROMs is accomplished by the generalized- $\alpha$  scheme with fixed step-size  $h = 0.001$  s in the interval  $[t_0, t_{\text{end}}] = [0, 1]$  s.

In Figures 11.1, 11.2 and 11.3 the results for  $K = 10$  are depicted. Note that  $\mathbf{V}_\phi$  is composed of  $r_\phi = 3$  VMs, whereas  $\mathbf{V}_{\text{aug}}$  contains  $r_\phi = 3$  VMs and  $r_\phi \cdot (r_\phi + 1)/2 = 6$  SMDs, i.e.  $r_{\text{aug}} = 9$ . The POD and SO-NLMM bases contain  $r = 10$  vectors, respectively. In Figures 11.4, 11.5 and 11.6 the results for  $r_\phi = 5$ ,  $r_{\text{aug}} = 20$  and  $r = 20$  are shown.

In terms of approximation quality we can see that the pure linear basis  $\mathbf{V}_\phi$  cannot capture the nonlinear behavior at all. Both SO-NLMM and POD yield satisfactory results, being POD superior in both cases. SO-NLMM yields slightly better results than the augmentation approach for  $r = 10$  and performs equally good for  $r = 20$ . For this latter reduced order one can practically not see the mismatch between FOM and ROMs. In terms of computational effort POD required  $\approx 37$  s (simulation + SVD), whereas SO-NLMM needed only  $\approx 2.6$  s ( $K = 10$ ) or  $\approx 6$  s ( $K = 20$ ) to compute the reduction basis during the offline phase. The basis augmentation approach is also very efficient, requiring less than 1 s in both cases.

<sup>1</sup>The Python implementation of SO-NLMM, the employed test scripts as well as some videos are available at <https://zenodo.org/record/2611120>.

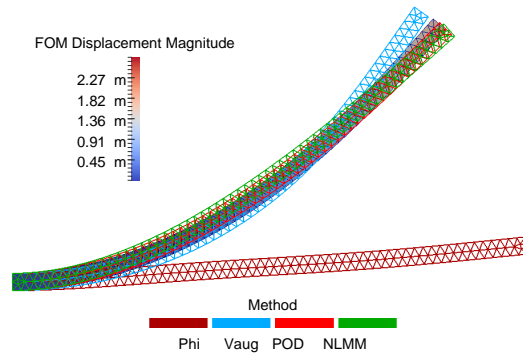


Figure 11.1: Snapshot of the cantilever beam simulation ( $r_\phi = 3, r_{\text{aug}} = 9, r = 10$ )

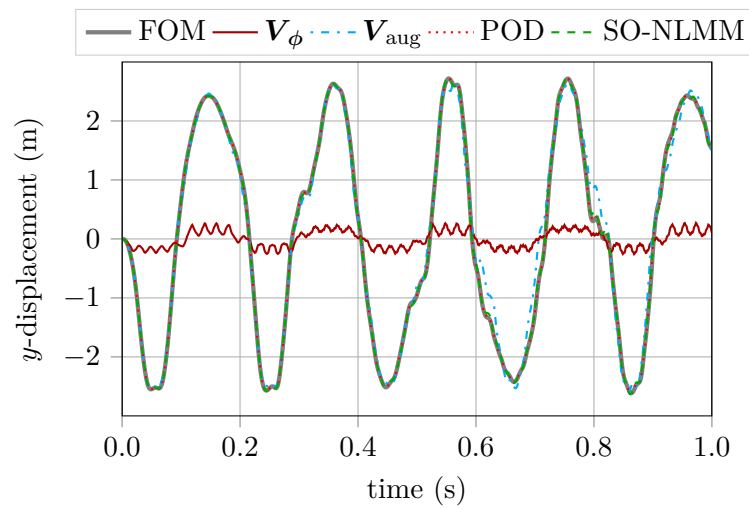


Figure 11.2:  $y$ -displacement of beam's tip for FOM and ROMs ( $r_\phi = 3, r_{\text{aug}} = 9, r = 10$ )

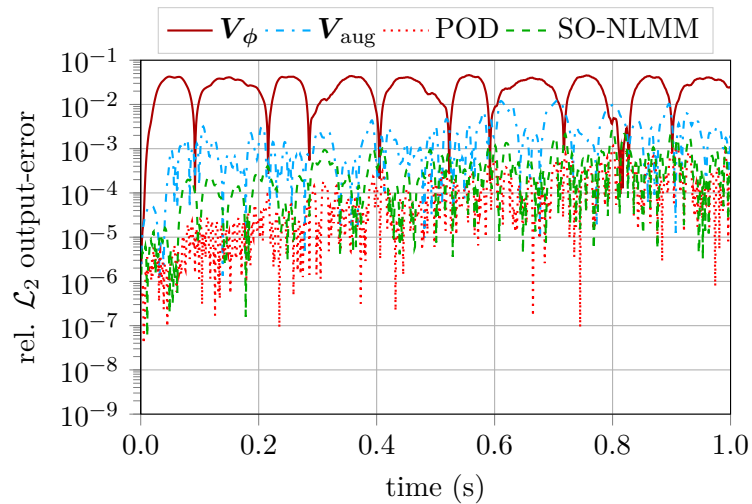


Figure 11.3:  $\frac{|y(t) - y_r^m(t)|}{\|y\|_{\mathcal{L}_2}}$ -error for different reduction methods ( $r_\phi = 3, r_{\text{aug}} = 9, r = 10$ )

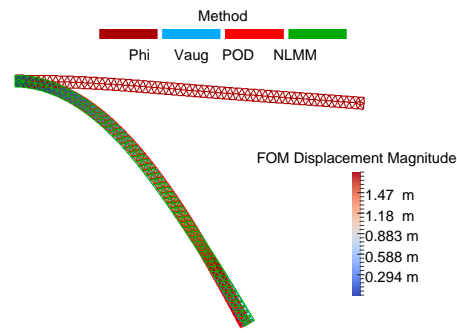


Figure 11.4: Snapshot of the cantilever beam simulation ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ )

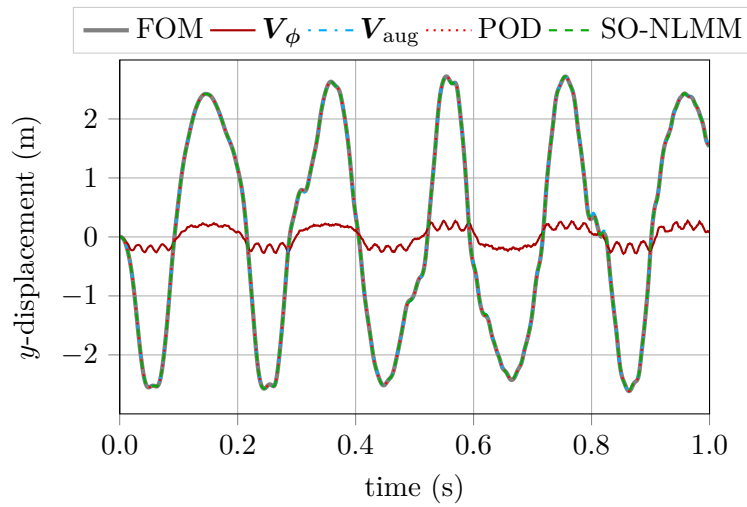


Figure 11.5:  $y$ -displacement of beam's tip for FOM and ROMs ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ )

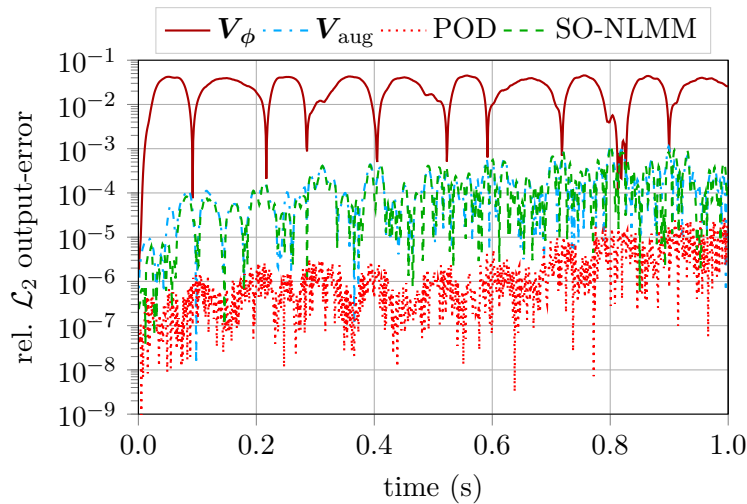


Figure 11.6:  $\frac{|y(t) - y_r^i(t)|}{\|y\|_{\mathcal{L}_2}}$ -error for different reduction methods ( $r_\phi = 5, r_{\text{aug}} = 20, r = 20$ )



### 11.4.2 S-shaped structure

The geometry of the considered S-shaped structure can be seen in Figure 11.7. The structure is 32 m long, 11 m high and clamped on both sides. The origin of the coordinate system is located at the middle of the structure, meaning that it is symmetric w.r.t. the  $y$ -axis (16 m right/left, 5.5 m up/down). The employed St. Venant-Kirchhoff material has the properties  $E = 2.1$  GPa,  $\nu = 0.3$  and  $\rho = 7867 \frac{\text{kg}}{\text{m}^3}$ . The model is discretized using 136 triangular Tri6 elements with quadratic shape functions, yielding (after Dirichlet boundary conditions)  $n = 742$  degrees of freedom in  $x$ - and  $y$ -direction. Zero physical damping  $\mathbf{D} = \mathbf{0}$  is again assumed, whereby numerical damping is introduced through the generalized- $\alpha$  scheme. The loading force  $F(t)$  is applied at the coordinate  $(0, 5.5)$  in negative  $y$ -direction. The observed node is indicated by a magenta point in Figure 11.7 and is approximately located at  $(-5.1, -1.0)$ .

We apply Algorithm 11.1 using a single signal generator with  $K = 20$  equidistant time-snapshots in the interval  $t \in [0, 5 \text{ s}]$ . For the *training phase* of SO-NLMM and POD, we use the signal generator  $q_r^v(t) = \sin(4t)$  – i.e.  $\omega_{\text{train}} = 4 \frac{\text{rad}}{\text{s}}$  – with corresponding  $\dot{q}_r^v(t)$ ,  $\ddot{q}_r^v(t)$  and the training input  $F_{\text{train}}(t) = r(q_r^v(t)) = 4 \cdot 10^6 \cdot q_r^v(t)$ . For the *test phase* of FOM and ROMs we apply the different input  $F_{\text{test}}(t) = 4 \cdot 10^6 \cdot \sin(6t)$  – i.e.  $\omega_{\text{test}} = 6 \frac{\text{rad}}{\text{s}}$ . We again compare POD and SO-NLMM with ROMs obtained via a pure linear basis  $\mathbf{V}_\phi$  and an augmented basis  $\mathbf{V}_{\text{aug}}$  containing VMs and SMDs (for  $\mathbf{q}_{\text{eq}} = \mathbf{0}$ ). The numerical time integration is conducted with a fixed step-size  $h = 0.01 \text{ s}$  in the interval  $[t_0, t_{\text{end}}] = [0, 5 \text{ s}]$ .

The numerical results for  $r_\phi = 5$ ,  $r_{\text{aug}} = 20$ ,  $r = 20$  are illustrated in Figures 11.8 and 11.9. The pure linear basis performs badly, whereas SO-NLMM and the augmented basis yield a comparably good approximation. POD leads again to the smallest error. Note that the training and test signals employed here do not differ too much from each other. However, considering a completely different test scenario would reveal the training dependency of POD (and maybe also of SO-NLMM). The advantage of NLMM is that a complete transient simulation of the FOM is not required to apply different training signal generators.

For the computation of the bases during the offline phase POD required  $\approx 16.7 \text{ s}$  (simulation + SVD), while SO-NLMM needed only  $\approx 1.8 \text{ s}$  ( $K = 20$ ) and the basis augmentation approach less than 1 s. Thus, the simulation-free approaches are more efficient than the simulation-based method. The simulation of the FOM and ROMs during the online phase takes  $\approx 17 \text{ s}$  and  $\approx 14 \text{ s}$ , respectively. This small speed-up is a consequence of the application of dimensional reduction only. We have not applied hyper-reduction yet, since our primary goal was to evaluate the performance of SO-NLMM in terms of approximation quality and offline costs. For further speed-up one could consider applying ECSW with the simulation-free generation of snapshots proposed in Section 7.6.2.

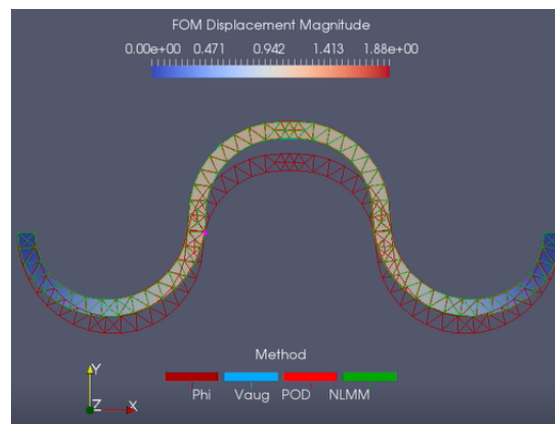


Figure 11.7: Snapshot of the simulation for the S-shaped structure ( $r_\phi=5, r_{\text{aug}}=20, r=20$ )

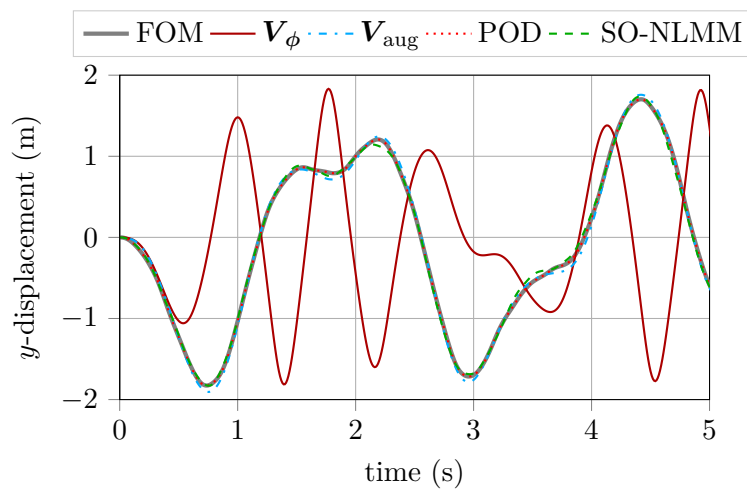


Figure 11.8:  $y$ -displacement of observed node for FOM and ROMs ( $r_\phi=5, r_{\text{aug}}=20, r=20$ )

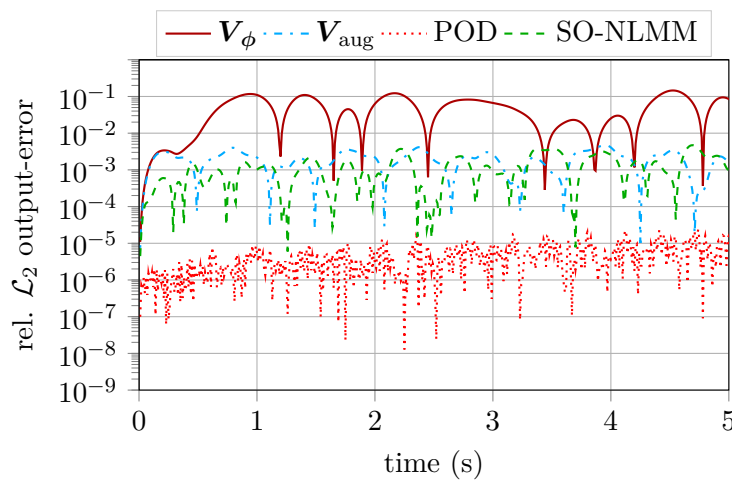


Figure 11.9:  $\frac{|y(t) - y_r^t(t)|}{\|y\|_{\mathcal{L}_2}}$ -error for different reduction methods ( $r_\phi=5, r_{\text{aug}}=20, r=20$ )

## 11.5 Conclusions

Based on the ideas from Astolfi [14] we have extended the concept of moment matching to nonlinear second-order systems. We have derived the corresponding Sylvester-like PDE and employed our simplifications to achieve a feasible algorithm relying on the solution of nonlinear systems of equations. As in the state-space case, the selection of appropriate signal generators is essential for the performance of the algorithm.

It can be concluded that SO-NLMM is slightly different from the concept of modal derivatives. The former method emerges from the approximation of the nonlinear manifold  $\boldsymbol{\nu}(\mathbf{q}_r^v)$ , whereas the modal derivatives result from the perturbation of the eigenvalue problem or from the Volterra series representation of the solution  $\mathbf{q}(t)$ . Both approaches are simulation-free, since they do not rely on the simulation of the FOM to construct the basis. However, SO-NLMM depends on the choice of signal generators while the modal derivatives rely on the selection of a linearization point  $\mathbf{q}_{\text{eq}}$ .

We have provided a first comparison study between SO-NLMM and (static) modal derivatives. In the future, a more exhaustive examination between both techniques is required to gain further insight about the conceptual similarities and differences. Moreover, a comparison between the QM ansatz with VMs+MDs and a nonlinear projection based on the (approximate) solution  $\boldsymbol{\nu}(\mathbf{q}_r^v)$  of the PDE (11.18) could be carried out.



PART V  
CLOSURE AND APPENDICES



# Chapter 12

## Concluding Remarks

This thesis studies system-theoretic model order reduction of nonlinear dynamical systems. In this chapter we summarize the main contributions of the thesis and draw conclusions from the most important results. We also provide some ideas for future research endeavors in the context of nonlinear model reduction.

### Summary and conclusions

Most of the existing methods for linear model order reduction rely (in one way or another) on system-theoretic concepts. For instance, modal truncation is grounded on the eigenmodes, balanced truncation relies on the Gramians, moment matching interpolates the transfer function and  $\mathcal{H}_2$ -optimal approaches require the  $\mathcal{H}_2$ -norm. On the contrary, many nonlinear model reduction techniques rely on expensive simulations of the original model to construct the ROM. Hence, in this thesis we have focused on simulation-free methods to avoid training simulations and lower the offline computational costs of the reduction process. In this regard, we have investigated system-theoretic concepts and reduction approaches for polynomial, nonlinear state-space and nonlinear second-order systems. In the context of polynomial systems the Volterra series representation has played a crucial role to obtain e.g. multivariable transfer functions required for Krylov-based reduction. In the field of nonlinear state-space systems we have turned our attention to the center manifold theory, the steady-state interpretation of moments and the concept of eigenfunctions. Finally, in the realm of nonlinear mechanical systems we have focused on modal derivatives and the Harmonic Balance method.

The most important statements and conclusions learned from each chapter are presented in the following:

### Chapter 3

- Interconnecting a system with a linear signal generator  $(\mathbf{S}_v, \mathbf{R}, \mathbf{x}_{r,0}^v)$  corresponds to exciting the system with exponential input signals of the form  $\mathbf{u}(t) = \mathbf{R} e^{\mathbf{S}_v t} \mathbf{x}_{r,0}^v$ .
- In addition to the classical interpretation of moment matching as the interpolation of the transfer function at certain shifts, one can also interpret this concept as the interpolation of the steady-state response of the system interconnected with a linear signal generator.
- The rational Krylov subspace method from [84] allows not only the approximate solution of linear Lyapunov equations, but also the cumulative reduction of LTI systems. Therefore, the method has been implemented in the sssMOR toolbox. CRKSM constitutes a good alternative to IRKA, not only because of the adaptive choice of interpolation data, but also due to the automatic selection of the reduced order.

### Chapter 4

- Starting from a nonlinear input-affine system, there are different methods to obtain a polynomial system (Taylor series, polynomialization) and/or to transform the latter to a special class (Carleman bilinearization, quadratic-bilinearization). Their respective properties, advantages and disadvantages have been discussed, enriching the existing literature.
- In addition to the Picard iteration, the variational equation approach [221] also allows to obtain the Volterra series representation and the multivariable convolution kernels via homogeneous subsystem state equations. Both methods have been applied and discussed.
- The growing exponential approach represents an easier way to obtain transfer functions for polynomial systems than the multidimensional Laplace transform of the kernels and output equations. Based upon the eigenfunction property, the solution to a sum of growing exponentials is directly expressed in time-domain via the generalized transfer functions, without requiring an inverse Laplace transform like for (4.37) and (4.41). We have applied the approach to bilinear and quadratic-bilinear systems.
- Instead of the Kronecker notation often used in the literature, we employed the more insightful sum notation for the MIMO transfer functions. This increases the awareness about the  $(j_2, \dots, j_k)$  combinations resulting from the multiple bilinear matrices  $\mathbf{N}_1, \dots, \mathbf{N}_m$ .

### Chapter 5

- The regular kernels and transfer functions allow the generalization of many system-theoretic concepts to the bilinear setting. We have extended the pole-residue formulation to the MIMO case and employed the result to express the  $\mathcal{H}_2$ -norm. We also have discussed our BSSSMOR implementation of `blyapchol` to solve large bilinear Lyapunov equations.
- In the context of Krylov-based reduction of polynomial systems one can distinguish between subsystem and Volterra series interpolation. For MIMO subsystem interpolation we have proposed to sum the moments over all  $(j_2, \dots, j_k)$  combinations of the transfer functions to reduce the dimension of the Krylov subspace.
- Exploiting the bilinear Sylvester equations, we have provided the until now missing Volterra series interpolation conditions for the MIMO case. The framework has been further extended to the multimoment setting and implemented in an efficient Arnoldi-like manner in the BSSSMOR toolbox (`volterraBrk`). Only if *infinite* Volterra series interpolation is desired, the corresponding bilinear Sylvester equations should be solved.
- We have extended the concept of  $\mathcal{H}_2$ -pseudo-optimality to the bilinear case and presented the iteration-free algorithm BPOK. Its properties and usage have also been mentioned.

### Chapter 6

- The reduction of nonlinear state-space systems using a linear projection is well understood. Moreover, several established techniques (e.g. POD, TPWL, basis augmentation) exist to construct the linear subspaces. The latter have proven successful in lots of applications, but may require many basis vectors to capture the nonlinear dynamics satisfactorily.
- The use of a nonlinear projection constitutes a very promising way to reduce dynamical systems, allowing for a smaller reduced order than with linear subspaces. However, nonlinear projection-based reduction is still in its infancy. Moreover, the projection onto the tangent space of a manifold results in a more complicated ROM.
- There exist different strategies to accomplish hyper-reduction (e.g. polynomial representation, PWL, DEIM, manifold-based), all having their advantages and applicability.



## Chapter 7

- Astolfi's extension of moment matching to the nonlinear case requires the solution of a Sylvester-like PDE to calculate the nonlinear reduction mapping. We have proposed certain simplifications (linear projection, column-wise consideration, time discretization) to approximate the PDE and achieve a feasible algorithm that relies on the solution of NLSEs.
- An extensive discussion about the simplifications has been presented, together with guidelines for the selection of the reduction parameters and the use of the algorithm. The numerical examples show the efficiency of our simulation-free approach in comparison to POD.
- We have discussed the semi non-intrusive nature of NLMM, as well as its combination with simulation-free hyper-reduction. These remarks, the practical guidelines and some prior system knowledge make NLMM well applicable to large finite element models.

## Chapter 8

- The eigenfunctions of bilinear and quadratic-bilinear systems have been derived from Bernoulli differential equations. Moreover, we have explained how they can be constructed via a signal generator and exploited in the nonlinear moment matching framework from Astolfi. However, a nonlinear manifold is indispensable in this setting to take the superposition of eigenfunctions properly into account (superposition principle does not hold).
- We have discussed how the signal generator and the input need to be chosen to obtain the state-independent bilinear Sylvester equations employed in the Volterra series interpolation.

## Chapter 9

- For the linear projection-based reduction of nonlinear second-order systems we have concentrated on the concept of basis augmentation with modal derivatives. The original derivation of modal derivatives is based on the perturbation of the linearized eigenvalue problem and yields a singular LSE under mass consideration. Hence, the mass terms are often neglected to obtain the static modal derivatives, which are symmetric.
- In the context of nonlinear projection we have focused on the emerging convective term and the time integration of manifold-ROMs. Moreover, we have treated the special case of a quadratic manifold, which can be parametrized with modes and modal derivatives.

## Chapter 10

- We have presented a novel derivation for modal derivatives based on the application of the variational equation approach to the polynomial(ized) second-order system. The gained derivatives are inherently symmetric and allow the explanation of internal resonances due to the sum/subtraction of eigenfrequencies in the coefficient matrix.
- The new derivatives can be employed to give an analytical solution via the truncated Volterra series, to formulate novel quadratic manifold approaches for model reduction and to show the differences between the Volterra series and Harmonic Balance method.

## Chapter 11

- We have extended the method of moment matching from linear to nonlinear mechanical systems. The generalization yields a second-order PDE that has been gradually simplified to achieve a simulation-free second-order nonlinear moment matching algorithm.
- SO-NLMM has been implemented in the research code AMfe. Numerical results for geometrically nonlinear structural models demonstrate the performance of the algorithm.

## Future research topics

Some perspectives for future research have already been mentioned throughout the thesis. In the following we present some more open questions for possible research:

- We have initiated the implementation of the BSSSMOR and QBSSSMOR toolboxes for bilinear and quadratic-bilinear model reduction. However, due to the lack of time they could not be developed to their full extent. Thus, it would be interesting to further implement procedural benchmarks, analysis functions and reduction algorithms to increase the functionality. Eventually, the toolboxes could be made open-source to foster collaborations.
- We have compared our NLMM algorithm to POD and to a pure linear basis computed with rational Krylov. However, a comparison with other nonlinear Krylov-based reduction methods is still missing. For instance, a numerical comparison between NLMM and Volterra series interpolation for (quadratic-)bilinear systems (e.g. using `volterraBrk`) should definitely be carried out in the future. Comparisons between NLMM and  $\mathcal{H}_2$ -optimal approaches – such as (T)BIRKA and TQB-IRKA [35] – are naturally possible, but not fair/reasonable. The same applies to a comparison between NLMM and the quadratic-bilinear Loewner framework [96], since the former is a reduction and the latter an identification approach. In our opinion it would be more appropriate to compare the data-driven approach [231] with the recent time-domain Loewner framework [143].
- The parametrized families of reduced models achieving moment matching have been theoretically discussed in this thesis, but not practically employed. Thus, the exploitation of the free parameters  $\Delta(\mathbf{x}_r)$  and  $\mathbf{W}$  to enforce certain properties (e.g. stability), as well as the numerical comparison between the projection-free and projection-based families of ROMs is topic of future research. In this context, it would be also interesting to compare the ROM obtained via our bilinear  $\mathcal{H}_2$ -pseudo-optimal algorithm BPORK with the ROM obtained by nonlinear moment matching in a non-projective manner.
- The ideas for an output NLMM algorithm presented in [70] could be further pursued. The ultimate goal would be to combine both algorithms to obtain a two-sided reduction method for approximated nonlinear moment matching. Hereby, the approximation quality as well as the preservation of important properties (such as stability) should be examined both from a theoretical and numerical perspective.
- The existing approaches for simulation-free hyper-reduction, as well as our proposed method based on assumed ansatz functions, could be applied to further speed the online evaluation of the ROM. Moreover, it would be very interesting to combine NLMM with a polynomial system representation gained either from an intrusive or non-intrusive approach. The latter method would make NLMM applicable in an industrial context.
- A further topic of research concerns nonlinear projection-based model reduction. Throughout the thesis we have proposed several conceptual ideas for (quadratic) manifold reduction, such as the exploitation of the bases  $\mathbf{V}^{(1)}$  and  $\mathbf{V}^{(2)}$  from subsystem interpolation, or the use of eigen- (Krylov-)vectors and their corresponding derivatives. Moreover, we have presented novel quadratic manifold approaches based on the newly derived modal derivatives. All these methods could be implemented and tested via numerical examples. Furthermore, a comparison between system-theoretic manifold reduction and recent data-driven approaches based on manifold/deep learning would be interesting.
- The novel modal derivatives and their promising applications should be examined in more detail. Moreover, the initiated implementation of the HB method could be followed up.

## Appendix A

### Taylor series expansion of non-input-affine nonlinear systems

In Section 4.1.1 we have discussed the Taylor series expansion of a nonlinear *input-affine* system (4.2). Although this representation arise in many engineering applications, it is not as general as the one given in (6.1). In fact, this latter representation allows to describe systems that are also nonlinear w.r.t. the inputs, whereas the representation (4.2) does not cover such type of nonlinearities. For instance, the systems  $\dot{x} = -x + u^2$ ,  $\dot{x} = \cos(x) + xu^2$  or  $\dot{x} = x^3 + \tanh(u)$  are *not* input-affine and can therefore only be described by (6.1).

The Taylor series expansion of  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  at the equilibrium point  $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$  yields

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) + \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} \right|_{\text{eq}} (\mathbf{x} - \mathbf{x}_{\text{eq}}) + \frac{1}{2!} \left. \frac{\partial^2 \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}^2} \right|_{\text{eq}} (\mathbf{x} - \mathbf{x}_{\text{eq}})^{(2)} + \dots \quad (\text{A.1})$$

$$+ \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}} \right|_{\text{eq}} (\mathbf{u} - \mathbf{u}_{\text{eq}}) + \frac{1}{2!} \left. \frac{\partial^2 \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}^2} \right|_{\text{eq}} (\mathbf{u} - \mathbf{u}_{\text{eq}})^{(2)} + \dots \quad (\text{A.2})$$

$$+ \frac{1}{2!} \left. \frac{\partial^2 \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x} \partial \mathbf{u}} \right|_{\text{eq}} (\mathbf{x} - \mathbf{x}_{\text{eq}})(\mathbf{u} - \mathbf{u}_{\text{eq}}) + \frac{1}{3!} \left. \frac{\partial^3 \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}^2 \partial \mathbf{u}} \right|_{\text{eq}} (\mathbf{x} - \mathbf{x}_{\text{eq}})^{(2)}(\mathbf{u} - \mathbf{u}_{\text{eq}}) + \dots \quad (\text{A.3})$$

Setting  $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = (\mathbf{0}, \mathbf{0})$  and assuming  $\mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = \mathbf{0}$ , we can write [221, p. 114]

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{F}_{10} \mathbf{x} + \mathbf{F}_{20}(\mathbf{x} \otimes \mathbf{x}) + \dots + \mathbf{F}_{01} \mathbf{u} + \mathbf{F}_{02}(\mathbf{u} \otimes \mathbf{u}) + \dots \quad (\text{A.4})$$

$$+ \mathbf{F}_{11} \mathbf{x} \mathbf{u} + \mathbf{F}_{21}(\mathbf{x} \otimes \mathbf{x}) \mathbf{u} + \dots \approx \sum_{k=0}^N \sum_{l=0}^N \mathbf{F}_{kl} \mathbf{x}^{(k)} \mathbf{u}^{(l)}, \quad (\text{A.5})$$

where  $\mathbf{F}_{00} \equiv \mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = \mathbf{0}$ . The partial derivatives of  $\mathbf{f}(\mathbf{x}, \mathbf{u})$  w.r.t.  $\mathbf{x}$  in (A.1), w.r.t.  $\mathbf{u}$  in (A.2) and the mixed derivatives in (A.3) are given by

$$\mathbf{F}_{k0} = \frac{1}{k!} \left. \frac{\partial^k \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}^k} \right|_{\text{eq}} \in \mathbb{R}^{n \times n^k}, \quad \mathbf{F}_{0l} = \frac{1}{l!} \left. \frac{\partial^l \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}^l} \right|_{\text{eq}} \in \mathbb{R}^{n \times m^l}, \quad (\text{A.6})$$

$$\mathbf{F}_{kl} = \frac{1}{(k+l)!} \left. \frac{\partial^{k+l} \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}^k \partial \mathbf{u}^l} \right|_{\text{eq}} \in \mathbb{R}^{n \times n^k \times m^l}.$$

Please note that  $\mathbf{F}_{kl}$  represents a 3-dimensional array, while the matrices  $\mathbf{F}_{k0}$  and  $\mathbf{F}_{0l}$  correspond to the 1-mode matricization of the corresponding tensors. In order to explain this in more detail, we provide an example in the following.

*Example A.1* (Quadratic-quadratic system). Truncating the Taylor series expansion after the second-order term yields

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) \approx \mathbf{F}_{10} \mathbf{x} + \mathbf{F}_{20}(\mathbf{x} \otimes \mathbf{x}) + \mathbf{F}_{01} \mathbf{u} + \mathbf{F}_{02}(\mathbf{u} \otimes \mathbf{u}) + \mathbf{F}_{11} \mathbf{x} \mathbf{u}, \quad (\text{A.7})$$

where  $\mathbf{F}_{10} \equiv \mathbf{A} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{F}_{20} \equiv \mathbf{H} \in \mathbb{R}^{n \times n^2}$ ,  $\mathbf{F}_{01} \equiv \mathbf{B} \in \mathbb{R}^{n \times m}$ ,  $\mathbf{F}_{02} \in \mathbb{R}^{n \times m^2}$  and  $\mathbf{F}_{11} \equiv \mathcal{N} \in \mathbb{R}^{n \times n \times m}$ . The matrix  $\mathbf{F}_{20} \hat{=} \mathcal{H}^{(1)}$  represents the 1-mode matricization of the third-order tensor  $\mathcal{H} \in \mathbb{R}^{n \times n \times n}$ . Similarly, the matrix  $\mathbf{F}_{02} \hat{=} \mathcal{B}^{(1)}$  represents the 1-mode matricization of the third-order tensor  $\mathcal{B} \in \mathbb{R}^{n \times m \times m}$ . The layers  $\mathcal{N}(:, :, j) \hat{=} \mathbf{N}_j \in \mathbb{R}^{n \times n}$  of the third-order tensor  $\mathcal{N}$  can also be arranged in a matrix  $\bar{\mathbf{N}} = [\mathbf{N}_1 \cdots \mathbf{N}_m] \in \mathbb{R}^{n \times n \cdot m}$ , which corresponds to the 1-mode matricization  $\mathcal{N}^{(1)}$ . This allows to rewrite (A.7) as

$$\begin{aligned} \mathbf{f}(\mathbf{x}, \mathbf{u}) &= \mathbf{A} \mathbf{x} + \mathbf{H}(\mathbf{x} \otimes \mathbf{x}) + \mathbf{B} \mathbf{u} + \mathbf{F}_{02}(\mathbf{u} \otimes \mathbf{u}) + \bar{\mathbf{N}}(\mathbf{u} \otimes \mathbf{x}), \\ &= \mathbf{A} \mathbf{x} + \mathbf{H}(\mathbf{x} \otimes \mathbf{x}) + \mathbf{B} \mathbf{u} + \mathbf{F}_{02}(\mathbf{u} \otimes \mathbf{u}) + \sum_{j=1}^m \mathbf{N}_j \mathbf{x} u_j, \end{aligned} \quad (\text{A.8})$$

which describes a dynamical system that is quadratic in state, quadratic in input and bilinear in state and input. △

## Appendix B

### Additional material for novel modal derivatives

#### Symmetry property of novel modal derivatives

In the following we show that the novel modal derivatives are indeed symmetric w.r.t. the indices  $i$  and  $j$ , i.e.  $\tilde{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ji}$  and  $\tilde{\tilde{\boldsymbol{\theta}}}_{ij} = \tilde{\tilde{\boldsymbol{\theta}}}_{ji}$ . The “proof” is exemplary shown for  $\tilde{\boldsymbol{\theta}}_{ij}$  in (10.13a), but applies similarly for  $\tilde{\boldsymbol{\theta}}_{ij}$  in (10.13b) too. The matrix  $\mathbf{K}_2 \in \mathbb{R}^{n \times n^2}$  is assumed symmetric.

From the bracket in (10.12e) it follows:

$$\begin{aligned} (\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M}) (\tilde{\boldsymbol{\theta}}_{ij} + \tilde{\boldsymbol{\theta}}_{ji}) &= -\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j + \boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i) \\ &= -\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j) - \mathbf{K}_2 (\boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i). \end{aligned} \quad (\text{B.1})$$

Multiplying the left-hand side out and comparing the terms yields

$$(\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M}) \tilde{\boldsymbol{\theta}}_{ij} = -\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j), \quad (\text{B.2a})$$

$$(\mathbf{K}_1 - (\omega_j + \omega_i)^2 \mathbf{M}) \tilde{\boldsymbol{\theta}}_{ji} = -\mathbf{K}_2 (\boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i). \quad (\text{B.2b})$$

Since  $\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j) = \mathbf{K}_2 (\boldsymbol{\phi}_j \otimes \boldsymbol{\phi}_i)$  holds and the left-hand side is also equal, it follows that  $\tilde{\boldsymbol{\theta}}_{ij} = \tilde{\boldsymbol{\theta}}_{ji}$ , i.e. the modal derivatives are symmetric. Thus, we can also rewrite (B.1) as

$$(\mathbf{K}_1 - (\omega_i + \omega_j)^2 \mathbf{M}) (2\tilde{\boldsymbol{\theta}}_{ij}) = -2\mathbf{K}_2 (\boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_j), \quad (\text{B.3})$$

from which Eq. (10.13a) results.

#### Quadratic manifold ansatz with velocities

The novel quadratic manifold ansatz (10.24) depends nonlinearly on the reduced displacements  $\mathbf{q}_r$  and the reduced velocities  $\dot{\mathbf{q}}_r$ :

$$\mathbf{q} \approx \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r) := \boldsymbol{\Phi}_r \mathbf{q}_r + \bar{\boldsymbol{\Theta}}_{r^2}(\mathbf{q}_r \otimes \mathbf{q}_r) - \hat{\boldsymbol{\Theta}}_{r^2}(\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r). \quad (\text{B.4})$$

Differentiating this twice with respect to time yields

$$\dot{\mathbf{q}} = \boldsymbol{\Phi}_r \dot{\mathbf{q}}_r + \bar{\boldsymbol{\Theta}}_{r^2}(\dot{\mathbf{q}}_r \otimes \mathbf{q}_r + \mathbf{q}_r \otimes \dot{\mathbf{q}}_r) - \hat{\boldsymbol{\Theta}}_{r^2}(\ddot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r + \dot{\mathbf{q}}_r \otimes \ddot{\mathbf{q}}_r), \quad (\text{B.5})$$

$$\begin{aligned} \ddot{\mathbf{q}} &= \boldsymbol{\Phi}_r \ddot{\mathbf{q}}_r + \bar{\boldsymbol{\Theta}}_{r^2}(\ddot{\mathbf{q}}_r \otimes \mathbf{q}_r + \mathbf{q}_r \otimes \ddot{\mathbf{q}}_r) + 2\bar{\boldsymbol{\Theta}}_{r^2}(\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) \\ &\quad - \hat{\boldsymbol{\Theta}}_{r^2}(\ddot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r + \dot{\mathbf{q}}_r \otimes \ddot{\mathbf{q}}_r) - 2\hat{\boldsymbol{\Theta}}_{r^2}(\ddot{\mathbf{q}}_r \otimes \ddot{\mathbf{q}}_r). \end{aligned} \quad (\text{B.6})$$

From these equations the following Jacobians and Hessians can be derived:

$$\widetilde{\mathbf{V}}_{\mathbf{q}_r} = \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)}{\partial \mathbf{q}_r} = \boldsymbol{\Phi}_r + \bar{\boldsymbol{\Theta}}_{r,2}(\mathbf{1}_r \otimes \mathbf{q}_r + \mathbf{q}_r \otimes \mathbf{1}_r), \quad (\text{B.7})$$

$$d\widetilde{\mathbf{V}}_{\mathbf{q}_r} = \frac{\partial^2 \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)}{\partial \mathbf{q}_r^2} = 2 \bar{\boldsymbol{\Theta}}_{r,2}, \quad (\text{B.8})$$

$$\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} = \frac{\partial \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)}{\partial \dot{\mathbf{q}}_r} = -\hat{\boldsymbol{\Theta}}_{r,2}(\mathbf{1}_r \otimes \dot{\mathbf{q}}_r + \dot{\mathbf{q}}_r \otimes \mathbf{1}_r), \quad (\text{B.9})$$

$$d\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} = \frac{\partial^2 \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)}{\partial \dot{\mathbf{q}}_r^2} = -2 \hat{\boldsymbol{\Theta}}_{r,2}, \quad (\text{B.10})$$

with the vector of ones  $\mathbf{1}_r = [1, \dots, 1]^\top \in \mathbb{R}^r$ . Inserting these Jacobians and Hessians into (B.5) and (B.6) yields

$$\dot{\mathbf{q}} = \widetilde{\mathbf{V}}_{\mathbf{q}_r} \dot{\mathbf{q}}_r + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r, \quad (\text{B.11})$$

$$\ddot{\mathbf{q}} = \widetilde{\mathbf{V}}_{\mathbf{q}_r} \ddot{\mathbf{q}}_r + d\widetilde{\mathbf{V}}_{\mathbf{q}_r}(\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r + d\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r}(\ddot{\mathbf{q}}_r \otimes \ddot{\mathbf{q}}_r). \quad (\text{B.12})$$

The first derivative is composed of two terms (cf. with Eq. (9.24)), whereas the second derivative contains four terms (cf. with Eq. (9.25)). What is more, one of the terms depends on the third time-derivative  $\ddot{\mathbf{q}}_r$ , also known as *jerk*.

The ansatz (B.4) together with its derivatives (B.11) and (B.12) can then be inserted into the equations of motion

$$M\ddot{\mathbf{q}} + D\dot{\mathbf{q}} + \mathbf{f}(\mathbf{q}) = \mathbf{f}_{\text{ext}}. \quad (\text{B.13})$$

This leads to the overdetermined system

$$\begin{aligned} M\widetilde{\mathbf{V}}_{\mathbf{q}_r} \ddot{\mathbf{q}}_r + Md\widetilde{\mathbf{V}}_{\mathbf{q}_r}(\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) + M\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r + Md\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r}(\ddot{\mathbf{q}}_r \otimes \ddot{\mathbf{q}}_r) \\ + D\widetilde{\mathbf{V}}_{\mathbf{q}_r} \dot{\mathbf{q}}_r + D\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r + \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)) = \mathbf{f}_{\text{ext}} + \boldsymbol{\varepsilon}. \end{aligned} \quad (\text{B.14})$$

The projection is performed onto the tangent space of the displacement and velocity-dependent manifold  $\boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)$ , i.e. onto  $\text{ran}(\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})$ . The projection is conducted orthogonally to this same space. Therefore, we multiply the above overdetermined system from the left with  $(\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top$  and enforce the Galerkin condition  $(\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top \boldsymbol{\varepsilon} = \mathbf{0}$ . Neglecting the jerk-related term leads to

$$\begin{aligned} (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top M\widetilde{\mathbf{V}}_{\mathbf{q}_r} \ddot{\mathbf{q}}_r + (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top Md\widetilde{\mathbf{V}}_{\mathbf{q}_r}(\dot{\mathbf{q}}_r \otimes \dot{\mathbf{q}}_r) + (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top M\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r \\ + (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top D\widetilde{\mathbf{V}}_{\mathbf{q}_r} \dot{\mathbf{q}}_r + (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top D\widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r} \ddot{\mathbf{q}}_r + (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top \mathbf{f}(\boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)) = (\widetilde{\mathbf{V}}_{\mathbf{q}_r} + \widetilde{\mathbf{V}}_{\dot{\mathbf{q}}_r})^\top \mathbf{f}_{\text{ext}}. \end{aligned} \quad (\text{B.15})$$

Due to the additional terms, the time integration of the above manifold-ROM is much more involved than the one for (9.28).

Note that the projection framework for the novel ansatz  $\mathbf{q} \approx \boldsymbol{\nu}(\mathbf{q}_r, \dot{\mathbf{q}}_r)$  is only mentioned here for the sake of completeness. We did not implement it due to the lack of time and the right application (AMfe includes geometric nonlinearities, but is not advection-dominant). Nevertheless, as already mentioned, this manifold ansatz seems promising for the reduction of nonlinear problems with quadratic velocity terms, e.g. in the Navier-Stokes equation.

### Volterra ansatz inserted in polynomial system

The individual terms of Eq. (10.36a) are ( $\mathbf{K}_2$  is symmetric, trigonometric identities<sup>1</sup>):

$$\begin{aligned}
\alpha^{(2)} &: c^2 \boldsymbol{\phi} \otimes \boldsymbol{\phi} \left( \frac{1}{2} \cos(2\omega t) + \frac{1}{2} \right), \\
\beta^{(2)} &: c^4 \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} \left( \frac{1}{2} \cos(4\omega t) + \frac{1}{2} \right), \\
\gamma^{(2)} &: c^4 \tilde{\tilde{\boldsymbol{\theta}}} \otimes \tilde{\tilde{\boldsymbol{\theta}}}, \\
\delta^{(2)} &: c^6 \boldsymbol{\phi}^{(3)} \otimes \boldsymbol{\phi}^{(3)} \left( \frac{1}{2} \cos(6\omega t) + \frac{1}{2} \right), \\
2\alpha \otimes \beta &: c^3 \boldsymbol{\phi} \otimes \tilde{\boldsymbol{\theta}} (\cos(3\omega t) + \cos(\omega t)), \\
2\alpha \otimes \gamma &: 2c^3 \boldsymbol{\phi} \otimes \tilde{\tilde{\boldsymbol{\theta}}} \cos(\omega t), \\
2\alpha \otimes \delta &: c^4 \boldsymbol{\phi} \otimes \boldsymbol{\phi}^{(3)} (\cos(4\omega t) + \cos(2\omega t)), \\
2\beta \otimes \gamma &: 2c^4 \tilde{\boldsymbol{\theta}} \otimes \tilde{\tilde{\boldsymbol{\theta}}} \cos(2\omega t), \\
2\beta \otimes \delta &: c^5 \tilde{\boldsymbol{\theta}} \otimes \boldsymbol{\phi}^{(3)} (\cos(5\omega t) + \cos(\omega t)), \\
2\gamma \otimes \delta &: 2c^5 \tilde{\tilde{\boldsymbol{\theta}}} \otimes \boldsymbol{\phi}^{(3)} \cos(3\omega t).
\end{aligned} \tag{B.16}$$

The individual terms of Eq. (10.36b) are ( $\mathbf{K}_3$  is symmetric, trigonometric identities<sup>2</sup>):

$$\begin{aligned}
\alpha^{(3)} &: \frac{1}{4} c^3 \boldsymbol{\phi} \otimes \boldsymbol{\phi} \otimes \boldsymbol{\phi} (\cos(3\omega t) + 3 \cos(\omega t)), \\
\beta^{(3)} &: \frac{1}{4} c^6 \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} (\cos(6\omega t) + 3 \cos(2\omega t)), \\
\gamma^{(3)} &: c^6 \tilde{\tilde{\boldsymbol{\theta}}} \otimes \tilde{\tilde{\boldsymbol{\theta}}} \otimes \tilde{\tilde{\boldsymbol{\theta}}}, \\
\delta^{(3)} &: \frac{1}{4} c^9 \boldsymbol{\phi}^{(3)} \otimes \boldsymbol{\phi}^{(3)} \otimes \boldsymbol{\phi}^{(3)} (\cos(9\omega t) + 3 \cos(3\omega t)), \\
3\alpha^{(2)} \otimes \beta &: \frac{3}{4} c^4 \boldsymbol{\phi} \otimes \boldsymbol{\phi} \otimes \tilde{\boldsymbol{\theta}} (\cos(4\omega t) + 2 \cos(2\omega t) + 1), \\
3\alpha^{(2)} \otimes \gamma &: \frac{3}{4} c^4 \boldsymbol{\phi} \otimes \boldsymbol{\phi} \otimes \tilde{\tilde{\boldsymbol{\theta}}} (2 \cos(2\omega t) + 2), \\
3\alpha^{(2)} \otimes \delta &: \frac{3}{4} c^5 \boldsymbol{\phi} \otimes \boldsymbol{\phi} \otimes \boldsymbol{\phi}^{(3)} (\cos(5\omega t) + 2 \cos(3\omega t) + \cos(\omega t)), \\
3\beta^{(2)} \otimes \alpha &: \frac{3}{4} c^5 \boldsymbol{\phi} \otimes \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} (\cos(5\omega t) + 2 \cos(\omega t) + \cos(3\omega t)), \\
3\beta^{(2)} \otimes \gamma &: \frac{3}{4} c^6 \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} \otimes \tilde{\tilde{\boldsymbol{\theta}}} (2 \cos(4\omega t) + 2), \\
3\beta^{(2)} \otimes \delta &: \frac{3}{4} c^7 \tilde{\boldsymbol{\theta}} \otimes \tilde{\boldsymbol{\theta}} \otimes \boldsymbol{\phi}^{(3)} (\cos(7\omega t) + 2 \cos(3\omega t) + \cos(\omega t)), \\
4\alpha \otimes \beta \otimes \gamma &: c^5 \boldsymbol{\phi} \otimes \tilde{\boldsymbol{\theta}} \otimes \tilde{\tilde{\boldsymbol{\theta}}} (2 \cos(3\omega t) + 2 \cos(\omega t)), \\
4\alpha \otimes \beta \otimes \delta &: c^6 \boldsymbol{\phi} \otimes \tilde{\boldsymbol{\theta}} \otimes \boldsymbol{\phi}^{(3)} (\cos(6\omega t) + \cos(4\omega t) + \cos(2\omega t) + 1), \\
4\alpha \otimes \gamma \otimes \delta &: c^6 \boldsymbol{\phi} \otimes \tilde{\tilde{\boldsymbol{\theta}}} \otimes \boldsymbol{\phi}^{(3)} (2 \cos(4\omega t) + 2 \cos(2\omega t)), \\
4\beta \otimes \gamma \otimes \delta &: c^7 \tilde{\boldsymbol{\theta}} \otimes \tilde{\tilde{\boldsymbol{\theta}}} \otimes \boldsymbol{\phi}^{(3)} (2 \cos(5\omega t) + 2 \cos(\omega t)),
\end{aligned} \tag{B.17}$$

<sup>1</sup> $\cos x \cos y = 1/2 \cos(x+y) + 1/2 \cos(x-y)$ , especially  $\cos^2(x) = 1/2 \cos(2x) + 1/2$  and  $\cos(x) = \cos(-x)$ .

<sup>2</sup> $\cos x \cos y \cos z = 1/4 (\cos(x+y+z) + \cos(x+y-z) + \cos(x-y+z) + \cos(-x+y+z))$ , especially  $\cos^3(x) = 1/4 \cos(3x) + 3/4 \cos x$ .

and

$$\begin{aligned}
3\gamma^{(2)} \otimes \alpha &: \frac{3}{4}c^5\phi \otimes \tilde{\theta} \otimes \tilde{\theta}(2\cos(\omega t) + 2), \\
3\gamma^{(2)} \otimes \beta &: \frac{3}{4}c^6\tilde{\theta} \otimes \tilde{\theta} \otimes \tilde{\theta}(2\cos(2\omega t) + 2), \\
3\gamma^{(2)} \otimes \delta &: \frac{3}{4}c^7\tilde{\theta} \otimes \tilde{\theta} \otimes \phi^{(3)}(2\cos(3\omega t) + 2), \\
3\delta^{(2)} \otimes \alpha &: \frac{3}{4}c^7\phi \otimes \phi^{(3)} \otimes \phi^{(3)}(\cos(7\omega t) + 2\cos(\omega t) + \cos(5\omega t)), \\
3\delta^{(2)} \otimes \beta &: \frac{3}{4}c^8\tilde{\theta} \otimes \phi^{(3)} \otimes \phi^{(3)}(\cos(8\omega t) + 2\cos(2\omega t) + \cos(4\omega t)), \\
3\delta^{(2)} \otimes \gamma &: \frac{3}{4}c^8\tilde{\theta} \otimes \phi^{(3)} \otimes \phi^{(3)}(2\cos(6\omega t) + 2).
\end{aligned} \tag{B.18}$$



## Bibliography

- [1] AL-BAIYAT, S. A., BETTAYEB, M., and AL-SAGGAF, U. M. “New model reduction scheme for bilinear systems”. In: *International Journal of Systems Science* 25.10 (1994), pp. 1631–1642.
- [2] AHMAD, M. I., BAUR, U., and BENNER, P. “Implicit Volterra series interpolation for model reduction of bilinear systems”. In: *Journal of Computational and Applied Mathematics* 316 (2017), pp. 15–28.
- [3] ANTOULAS, A. C., BENNER, P., and FENG, L. “Model reduction by iterative error system approximation”. In: *Mathematical and Computer Modelling of Dynamical Systems* 24.2 (2018), pp. 103–118.
- [4] ANTOULAS, A. C., GOSEA, I. V., and IONITA, A. C. “Model reduction of bilinear systems in the Loewner framework”. In: *SIAM Journal on Scientific Computing* 38.5 (2016), B889–B916.
- [5] AHRENS, J., GEVECI, B., and LAW, C. “Paraview: An end-user tool for large data visualization”. In: *The visualization handbook* 717 (2005).
- [6] ANTOULAS, A. C. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.
- [7] ANTOULAS, A. C. “On pole placement in model reduction”. In: *Automatisierungstechnik* 55.9 (2007), pp. 443–448.
- [8] ARNOLDI, W. E. “The principle of minimized iterations in the solution of the matrix eigenvalue problem”. In: *Quarterly of applied mathematics* 9.1 (1951), pp. 17–29.
- [9] ASCHER, U. M., RUUTH, S. J., and SPITERI, R. J. “Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations”. In: *Applied Numerical Mathematics* 25.2-3 (1997), pp. 151–167.
- [10] ANTOULAS, A. C. and SORENSEN, D. C. “Approximation of large-scale dynamical systems: an overview”. In: *International Journal of Applied Mathematics and Computer Science* 11.5 (2001), pp. 1093–1121.
- [11] ASTRID, P. et al. “Missing point estimation in models described by proper orthogonal decomposition”. In: *IEEE Transactions on Automatic Control* 53.10 (2008), pp. 2237–2251.
- [12] ASTOLFI, A. “A new look at model reduction by moment matching for linear systems”. In: *2007 46th IEEE Conference on Decision and Control*. IEEE. 2007, pp. 4361–4366.
- [13] ASTOLFI, A. “Model reduction by moment matching for nonlinear systems”. In: *2008 47th IEEE Conference on Decision and Control*. IEEE. 2008, pp. 4873–4878.
- [14] ASTOLFI, A. “Model reduction by moment matching for linear and nonlinear systems”. In: *IEEE TAC* 55.10 (2010), pp. 2321–2336.
- [15] ASTOLFI, A. “Model reduction by moment matching, steady-state response and projections”. In: *49th IEEE CDC*. 2010, pp. 5344–5349.

- [16] ÅSTRÖM, K. J. and WITTENMARK, B. *Adaptive Control*. 2nd. Addison-Wesley Longman Publishing Company, 1994.
- [17] BAI, Z. et al. “Error bound for reduced system model by Padé approximation via the Lanczos process”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 18.2 (1999), pp. 133–141.
- [18] BAI, Z. “Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems”. In: *Applied numerical mathematics* 43.1-2 (2002), pp. 9–44.
- [19] BARRAULT, M. et al. “An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations”. In: *Comptes Rendus Mathématique* 339.9 (2004), pp. 667–672.
- [20] BENNER, P. and BREITEN, T. “Interpolation-Based  $\mathcal{H}_2$ -Model Reduction of Bilinear Control Systems”. In: *SIAM Journal on Matrix Analysis and Applications* 33.3 (2012), pp. 859–885.
- [21] BENNER, P. and BREITEN, T. “Low rank methods for a class of generalized Lyapunov equations and related issues”. In: *Numerische Mathematik* 124.3 (2013), pp. 441–470.
- [22] BENNER, P. and BREITEN, T. “Two-sided projection methods for nonlinear model order reduction”. In: *SIAM Journal on Scientific Computing* 37.2 (2015), B239–B260.
- [23] BENNER, P., BREITEN, T., and DAMM, T. “Generalised tangential interpolation for model reduction of discrete-time MIMO bilinear systems”. In: *International Journal of Control* 84.8 (2011), pp. 1398–1407.
- [24] BAUR, U., BENNER, P., and FENG, L. “Model order reduction for linear and nonlinear systems: a system-theoretic perspective”. In: *Archives of Computational Methods in Engineering* 21.4 (2014), pp. 331–358.
- [25] BREITEN, T. and DAMM, T. “Krylov subspace methods for model order reduction of bilinear control systems”. In: *Systems & Control Letters* 59.8 (2010), pp. 443–450.
- [26] BENNER, P. and DAMM, T. “Lyapunov equations, energy functionals, and model order reduction of bilinear and stochastic systems”. In: *SIAM Journal on Control and Optimization* 49.2 (2011), pp. 686–711.
- [27] BRUNI, C., DIPILLO, G., and KOCH, G. “On the mathematical models of bilinear systems”. In: *Ricerca Di Automatica* 2.1 (1971), pp. 11–26.
- [28] BOLLHÖFER, M. and EPPLER, A. “A structure preserving FGMRES method for solving large Lyapunov equations”. In: *Progress in Industrial Mathematics at ECMI 2010*. Springer, 2012, pp. 131–136.
- [29] BERTRAM, C. and FASSBENDER, H. “A link between gramian based model order reduction and moment matching”. In: *arXiv preprint arXiv:2003.03101* (2020).
- [30] BEATTIE, C. A. and GUGERCIN, S. “Krylov-based model reduction of second-order systems with proportional damping”. In: *44th IEEE Conference on Decision and Control*. IEEE. 2005, pp. 2278–2283.
- [31] BEATTIE, C. A. and GUGERCIN, S. “A trust region method for optimal  $\mathcal{H}_2$  model reduction”. In: *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*. IEEE. 2009, pp. 5370–5375.
- [32] BEATTIE, C. and GUGERCIN, S. “Realization-independent  $\mathcal{H}_2$ -approximation”. In: *51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pp. 4953–4958.

- [33] BEATTIE, C. A. and GUGERCIN, S. “Model reduction by rational interpolation”. In: *Model Reduction and Algorithms: Theory and Applications* 15 (2017), pp. 297–334.
- [34] BENNER, P. and GOYAL, P. “Balanced truncation model order reduction for quadratic-bilinear control systems”. In: *arXiv preprint arXiv:1705.00160* (2017).
- [35] BENNER, P., GOYAL, P., and GUGERCIN, S. “ $\mathcal{H}_2$ -Quasi-Optimal Model Order Reduction for Quadratic-Bilinear Control Systems”. In: *SIAM Journal on Matrix Analysis and Applications* 39.2 (2018), pp. 983–1032.
- [36] BENNER, P., GUGERCIN, S., and WILLCOX, K. “A survey of projection-based model reduction methods for parametric dynamical systems”. In: *SIAM review* 57.4 (2015), pp. 483–531.
- [37] BERKOOZ, G., HOLMES, P., and LUMLEY, J. L. “The proper orthogonal decomposition in the analysis of turbulent flows”. In: *Annual review of fluid mechanics* 25.1 (1993), pp. 539–575.
- [38] BENNER, P., KÜRSCHNER, P., and SAAK, J. “An improved numerical method for balanced truncation for symmetric second-order systems”. In: *Mathematical and Computer Modelling of Dynamical Systems* 19.6 (2013), pp. 593–615.
- [39] BENNER, P., KÜRSCHNER, P., and SAAK, J. “Self-generating and efficient shift parameters in ADI methods for large Lyapunov and Sylvester equations”. In: *Electronic Transactions on Numerical Analysis (ETNA)* 43 (2014), pp. 142–162.
- [40] BENNER, P., KÜRSCHNER, P., and SAAK, J. “Frequency-limited balanced truncation with low-rank approximations”. In: *SIAM Journal on Scientific Computing* 38.1 (2016), A471–A499.
- [41] BENNER, P., LI, J.-R., and PENZL, T. “Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems”. In: *Numerical Linear Algebra with Applications* 15.9 (2008), pp. 755–777.
- [42] BENNER, P., LI, R.-C., and TRUHAR, N. “On the ADI method for Sylvester equations”. In: *Journal of Computational and Applied Mathematics* 233.4 (2009), pp. 1035–1045.
- [43] BRUNTON, S. L., PROCTOR, J. L., and KUTZ, J. N. “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. In: *Proc. Natl. Acad. Sci. (PNAS)* 113.15 (2016), pp. 3932–3937.
- [44] BENNER, P. and QUINTANA-ORTI, E. S. “Solving stable generalized Lyapunov equations with the matrix sign function”. In: *Numerical Algorithms* 20.1 (1999), pp. 75–100.
- [45] BREITEN, T. “Interpolatory Methods for Model Reduction of Large-Scale Dynamical Systems”. PhD thesis. Otto-von-Guericke Universität Magdeburg, 2013.
- [46] BAI, Z. and SU, Y. “Dimension reduction of large-scale second-order dynamical systems via a second-order Arnoldi method”. In: *SIAM Journal on Scientific Computing* 26.5 (2005), pp. 1692–1709.
- [47] BAI, Z. and SKOOGH, D. “A projection method for model reduction of bilinear dynamical systems”. In: *Linear algebra and its applications* 415.2-3 (2006), pp. 406–425.
- [48] BENNER, P. and STYKEL, T. “Model order reduction for differential-algebraic equations: a survey”. In: *Surveys in Differential-Algebraic Equations IV*. Springer, 2017, pp. 107–160.

- [49] BARTELS, R. H. and STEWART, G. W. “Solution of the matrix equation  $AX+XB=C$ ”. In: *Communications of the ACM* 15.9 (1972), pp. 820–826.
- [50] CARLBERG, K. et al. “The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows”. In: *Journal of Computational Physics* 242 (2013), pp. 623–647.
- [51] CARR, J. *Applications of centre manifold theory*. Applied Mathematical Sciences. Vol. 35. Springer-Verlag, 1981.
- [52] CASTAGNOTTO, A. et al. “sss & sssMOR: Analysis and reduction of large-scale dynamic systems in MATLAB”. In: *at-Automatisierungstechnik* 65.2 (2017), pp. 134–150.
- [53] CRAIG, R. and BAMPTON, M. “Coupling of substructures for dynamic analyses”. In: *AIAA Journal* 6.7 (1968), pp. 1313–1319.
- [54] CARLBERG, K., BARONE, M., and ANTIL, H. “Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction”. In: *Journal of Computational Physics* 330 (2017), pp. 693–734.
- [55] CARLBERG, K., BOU-MOSLEH, C., and FARHAT, C. “Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations”. In: *International Journal for Numerical Methods in Engineering* 86.2 (2011), pp. 155–181.
- [56] CHATURANTABUT, S., BEATTIE, C., and GUGERCIN, S. “Structure-preserving model reduction for nonlinear port-Hamiltonian systems”. In: *SIAM Journal on Scientific Computing* 38.5 (2016), B837–B865.
- [57] CHOI, Y. and CARLBERG, K. “Space–Time Least-Squares Petrov–Galerkin Projection for Nonlinear Model Reduction”. In: *SIAM Journal on Scientific Computing* 41.1 (2019), A26–A58.
- [58] CARLBERG, K., CHOI, Y., and SARGSYAN, S. “Structure-preserving nonlinear model reduction for finite-volume models”. In: *Journal of Computational Physics* 330 (2017), pp. 693–734.
- [59] CRUZ VARONA, M. and GEBHART, R. *Nonlinear Model Order Reduction: A system-theoretic viewpoint*. MoRePaS IV, ScienceOpen. 2018.
- [60] CRUZ VARONA, M. and GEBHART, R. “Impulse response of bilinear systems based on Volterra series representation”. <https://arxiv.org/abs/1812.05360>. 2018.
- [61] CHUNG, J. and HULBERT, G. M. “A Time Integration Algorithm for Structural Dynamics With Improved Numerical Dissipation: The Generalized- $\alpha$  Method”. In: *Journal of Applied Mechanics* 60.2 (1993), pp. 371–375.
- [62] CHAHLAOUI, Y. et al. “Second-order balanced truncation”. In: *Linear algebra and its applications* 415.2-3 (2006), pp. 373–384.
- [63] CHENG, C. M. et al. “Volterra-series-based nonlinear system modeling and its engineering applications: A state-of-the-art review”. In: *Mechanical Systems and Signal Processing* 87 (2017), pp. 340–364.
- [64] CHEN, Y. “Model order reduction for nonlinear systems”. MA thesis. Massachusetts Institute of Technology, 1999.
- [65] CHIDAMBARA, M. “Two simple techniques for the simplification of large dynamic systems”. In: *Joint Automatic Control Conference*. 7. 1969, pp. 669–674.

- [66] CONDON, M. and IVANOV, R. “Nonlinear systems—algebraic gramians and model reduction”. In: *COMPEL-Int. J. Comp. Math. Electr. Electron. Eng.* 24.1 (2005), pp. 202–219.
- [67] CHAFEE, N. and INFANTE, E. F. “A bifurcation problem for a nonlinear partial differential equation of parabolic type”. In: *Applicable Analysis* 4.1 (1974), pp. 17–37.
- [68] CRUZ VARONA, M., NABI, M., and LOHMANN, B. “Automatic adaptive sampling in parametric model order reduction by matrix interpolation”. In: *International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE. 2017, pp. 472–477.
- [69] CASTAGNOTTO, A., PANZER, H. K. F., and LOHMANN, B. “Fast  $\mathcal{H}_2$ -optimal model order reduction exploiting the local nature of Krylov-subspace methods”. In: *2016 European Control Conference (ECC)*. IEEE. 2016, pp. 1958–1969.
- [70] CRUZ VARONA, M., PAK, M., and LOHMANN, B. “Towards Output Krylov Subspace-Based Nonlinear Moment Matching”. In: *Joint 8th IFAC Symposium on Mechatronic Systems and 11th IFAC Symposium on Nonlinear Control Systems*. 2019.
- [71] CRUZ VARONA, M. et al. “Model Reduction of Dynamical Systems by Approximated Nonlinear Moment Matching”. Submitted to *at-Automatisierungstechnik*.
- [72] CRUZ VARONA, M. et al. “Practicable Simulation-Free Model Order Reduction by Nonlinear Moment Matching”. <https://arxiv.org/abs/1901.10750>. 2019.
- [73] CRUZ VARONA, M. et al. “A novel derivation for modal derivatives based on Volterra series representation and its use in nonlinear model order reduction”. In: *7th Conference on Computational Methods in Structural Dynamics and Earthquake Engineering*. ECCOMAS, 2019, pp. 2376–2394.
- [74] CHATURANTABUT, S. and SORENSEN, D. C. “Nonlinear model reduction via discrete empirical interpolation”. In: *SIAM Journal on Scientific Computing* 32.5 (2010), pp. 2737–2764.
- [75] CRUZ VARONA, M., SCHNEUCKER, N., and LOHMANN, B. “Nonlinear Moment Matching for the Simulation-Free Reduction of Structural Systems”. In: *IFAC-PapersOnLine* 52.16 (2019), pp. 328–333.
- [76] CARLBERG, K., TUMINARO, R., and BOGGS, P. “Preserving Lagrangian structure in nonlinear model reduction with application to structural dynamics”. In: *SIAM Journal on Scientific Computing* 37.2 (2015), B153–B184.
- [77] DAMM, T. “Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations”. In: *Numerical Linear Algebra with Applications* 15.9 (2008), pp. 853–871.
- [78] DAVISON, E. “A method for simplifying linear dynamic systems”. In: *IEEE Transactions on Automatic Control* 11.1 (1966), pp. 93–101.
- [79] DEMMEL, J. W. *Applied numerical linear algebra*. SIAM, 1997.
- [80] DRMAC, Z. and GUGERCIN, S. “A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions”. In: *SIAM Journal on Scientific Computing* 38.2 (2016), A631–A648.
- [81] DALESSANDRO, P., ISIDORI, A., and RUBERTI, A. “Realization and structure theory of bilinear dynamical systems”. In: *SIAM Journal on Control* 12.3 (1974), pp. 517–535.

- [82] DRUSKIN, V., KNIZHNERMAN, L., and SIMONCINI, V. “Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation”. In: *SIAM Journal on Numerical Analysis* 49.5 (2011), pp. 1875–1898.
- [83] DRUSKIN, V. and SIMONCINI, V. “Adaptive rational Krylov subspaces for large-scale dynamical systems”. In: *Systems & Control Letters* 60.8 (2011), pp. 546–560.
- [84] DRUSKIN, V., SIMONCINI, V., and ZASLAVSKY, M. “Adaptive tangential interpolation in rational Krylov subspaces for MIMO dynamical systems”. In: *SIAM Journal on Matrix Analysis and Applications* 35.2 (2014), pp. 476–498.
- [85] EVERSON, R. and SIROVICH, L. “Karhunen–Loeve procedure for gappy data”. In: *JOSA A* 12.8 (1995), pp. 1657–1664.
- [86] FENG, L., ANTOULAS, A. C., and BENNER, P. “Some a posteriori error bounds for reduced-order modelling of (non-)parametrized linear systems”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 51.6 (2017), pp. 2127–2158.
- [87] FARHAT, C. et al. “Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency”. In: *International Journal for Numerical Methods in Engineering* 98.9 (2014), pp. 625–662.
- [88] FARHAT, C., CHAPMAN, T., and AVERY, P. “Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models”. In: *International Journal for Numerical Methods in Engineering* 102.5 (2015), pp. 1077–1110.
- [89] FELDMANN, P. and FREUND, R. W. “Efficient linear circuit analysis by Padé approximation via the Lanczos process”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 14.5 (1995), pp. 639–649.
- [90] FLAGG, G. M. and GUGERCIN, S. “On the ADI method for the Sylvester equation and the optimal- $H_2$  points”. In: *Applied Numerical Mathematics* 64 (2013), pp. 50–58.
- [91] FLAGG, G. and GUGERCIN, S. “Multipoint Volterra series interpolation and  $H_2$  optimal model reduction of bilinear systems”. In: *SIAM Journal on Matrix Analysis and Applications* 36.2 (2015), pp. 549–579.
- [92] FLAGG, G. M. “Interpolation methods for the model reduction of bilinear systems”. PhD thesis. Virginia Polytechnic Institute and State University, 2012.
- [93] FREUND, R. W. “Passive reduced-order modeling via Krylov-subspace methods”. In: *CACSD. Conference Proceedings. IEEE International Symposium on Computer-Aided Control System Design (Cat. No. 00TH8537)*. IEEE. 2000, pp. 261–266.
- [94] FREUND, R. W. “Model reduction methods based on Krylov subspaces”. In: *Acta Numerica* 12 (2003), pp. 267–319.
- [95] GUGERCIN, S. and ANTOULAS, A. C. “Model reduction of large-scale systems by least squares”. In: *Linear algebra and its applications* 415.2-3 (2006), pp. 290–321.
- [96] GOSEA, I. V. and ANTOULAS, A. C. “Data-driven model order reduction of quadratic-bilinear systems”. In: *Numerical Linear Algebra with Applications* 25.6 (2018).
- [97] GUGERCIN, S., ANTOULAS, A. C., and BEATTIE, C. “ $\mathcal{H}_2$  model reduction for large-scale linear dynamical systems”. In: *SIAM Journal on Matrix Analysis and Applications* 30.2 (2008), pp. 609–638.

- [98] GALLIVAN, K., GRIMME, G., and VAN DOOREN, P. “A rational Lanczos algorithm for model reduction”. In: *Numerical Algorithms* 12.1 (1996), pp. 33–63.
- [99] GRASEDYCK, L., HACKBUSCH, W., and KHOROMSKIJ, B. N. “Solution of large scale algebraic matrix Riccati equations by use of hierarchical matrices”. In: *Computing* 70.2 (2003), pp. 121–165.
- [100] GIBILARO, L. and LEES, F. “The reduction of complex transfer function models to simple models using the method of moments”. In: *Chemical Engineering Science* 24.1 (1969), pp. 85–93.
- [101] GLOVER, K. “All optimal Hankel-norm approximations of linear multivariable systems and their  $L^\infty$ -error bounds”. In: *International Journal of Control* 39.6 (1984), pp. 1115–1193.
- [102] GRAY, W. S. and MESKO, J. “Energy functions and algebraic Gramians for bilinear systems”. In: *IFAC Proceedings Volumes* 31.17 (1998), pp. 101–106.
- [103] GOYAL, P. K. “System-theoretic model order reduction for bilinear and quadratic-bilinear systems”. PhD thesis. Otto-von-Guericke Universität Magdeburg, 2018.
- [104] GEUZAINÉ, C. and REMACLE, J.-F. “Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities”. In: *International Journal for Numerical Methods in Engineering* 79.11 (2009), pp. 1309–1331.
- [105] GÉRADIN, M. and RIXEN, D. J. *Mechanical vibrations: theory and application to structural dynamics*. John Wiley & Sons, 2014.
- [106] GRIMME, E. J. “Krylov Projection Methods for Model Reduction”. PhD thesis. Uni. Illinois at Urbana Champaign, 1997.
- [107] GUSTAVSEN, B. and SEMLYEN, A. “Rational approximation of frequency domain responses by vector fitting”. In: *IEEE Transactions on Power Delivery* 14.3 (1999), pp. 1052–1061.
- [108] GRIMME, E. J., SORENSEN, D. C., and VAN DOOREN, P. “Model reduction of state space systems via an implicitly restarted Lanczos method”. In: *Numerical algorithms* 12.1 (1996), pp. 1–31.
- [109] GU, C. “Model order reduction of nonlinear dynamical systems”. PhD thesis. University of California, Berkeley, 2011.
- [110] GU, C. “QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 30.9 (2011), pp. 1307–1320.
- [111] GUGERCIN, S. et al. “Structure-preserving tangential interpolation for model reduction of port-Hamiltonian systems”. In: *Automatica* 48.9 (2012), pp. 1963–1974.
- [112] GUGERCIN, S. “An iterative SVD-Krylov based method for model reduction of large-scale dynamical systems”. In: *Linear Algebra and its Applications* 428.8-9 (2008), pp. 1964–1986.
- [113] GUYAN, R. J. “Reduction of stiffness and mass matrices”. In: *AIAA Journal* 3.2 (1965), pp. 380–380.
- [114] GOLUB, G. H. and VAN LOAN, C. F. *Matrix Computations*. 4th. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

- 
- [115] GALLIVAN, K., VANDENDORPE, A., and VAN DOOREN, P. “Model reduction of MIMO systems via tangential interpolation”. In: *SIAM Journal on Matrix Analysis and Applications* 26.2 (2004), pp. 328–349.
- [116] GALLIVAN, K., VANDENDORPE, A., and VAN DOOREN, P. “Sylvester equations and projection-based model reduction”. In: *Journal of Computational and Applied Mathematics* 162.1 (2004), pp. 213–229.
- [117] HESTHAVEN, J. S., ROZZA, G., STAMM, B., et al. *Certified reduced basis methods for parametrized partial differential equations*. Vol. 590. Springer, 2016.
- [118] HAMMARLING, S. J. “Numerical solution of the stable, non-negative definite Lyapunov equation”. In: *IMA Journal of Numerical Analysis* 2.3 (1982), pp. 303–323.
- [119] HYLAND, D. and BERNSTEIN, D. “The optimal projection equations for model reduction and the relationships among the methods of Wilson, Skelton, and Moore”. In: *IEEE Transactions on Automatic Control* 30.12 (1985), pp. 1201–1211.
- [120] HOLLKAMP, J. J. and GORDON, R. W. “Reduced-order models for nonlinear response prediction: Implicit condensation and expansion”. In: *Journal of Sound and Vibration* 318.4-5 (2008), pp. 1139–1153.
- [121] HILBER, H. M., HUGHES, T. J., and TAYLOR, R. L. “Improved numerical dissipation for time integration algorithms in structural dynamics”. In: *Earthquake Engineering & Structural Dynamics* 5.3 (1977), pp. 283–292.
- [122] HORN, R. A. and JOHNSON, C. R. *Matrix Analysis*. Cambridge University Press, 2012.
- [123] HESTENES, M. R. and STIEFEL, E. *Methods of conjugate gradients for solving linear systems*. Vol. 49. 1. NBS Washington, DC, 1952.
- [124] HUANG, J. *Nonlinear Output Regulation: Theory and Applications*. SIAM Advances in Design and Control, 2004.
- [125] HURTY, W. “Dynamic analysis of structural systems using component modes”. In: *AIAA Journal* 3.4 (1965), pp. 678–685.
- [126] IONESCU, T. C. and ASTOLFI, A. “Families of reduced order models that achieve nonlinear moment matching”. In: *American Control Conference (ACC)*. IEEE. 2013, pp. 5518–5523.
- [127] IONESCU, T. C. and ASTOLFI, A. “Nonlinear moment matching-based model order reduction”. In: *IEEE TAC* 61.10 (2016), pp. 2837–2847.
- [128] IONESCU, T. C., ASTOLFI, A., and COLANERI, P. “Families of moment matching based, low order approximations for linear systems”. In: *Systems & Control Letters* 64 (2014), pp. 47–56.
- [129] IDELSOHN, S. R. and CARDONA, A. “A load-dependent basis for reduced nonlinear structural dynamics”. In: *Computers & Structures* 20.1-3 (1985), pp. 203–210.
- [130] IDELSOHN, S. R. and CARDONA, A. “A reduction method for nonlinear structural dynamic analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 49.3 (1985), pp. 253–279.
- [131] ISIDORI, A. *Nonlinear Control Systems*. Third edition. Springer, 1995.
- [132] JAIN, S. et al. “A quadratic manifold for model order reduction of nonlinear structural dynamics”. In: *Computers & Structures* 188 (2017), pp. 80–94.



- [133] JAIMOUKHA, I. M. and KASENALLY, E. M. “Krylov subspace methods for solving large Lyapunov equations”. In: *SIAM Journal on Numerical Analysis* 31.1 (1994), pp. 227–251.
- [134] JAIMOUKHA, I. M. and KASENALLY, E. M. “Implicitly restarted Krylov subspace methods for stable partial realizations”. In: *SIAM Journal on Matrix Analysis and Applications* 18.3 (1997), pp. 633–652.
- [135] JBILLOU, K. and RIQUET, A. “Projection methods for large Lyapunov matrix equations”. In: *Linear Algebra and its Applications* 415.2-3 (2006), pp. 344–358.
- [136] JORDAN, D. and SMITH, P. *Nonlinear ordinary differential equations: an introduction for scientists and engineers*. Fourth edition. Vol. 10. Oxford University Press, 2007.
- [137] JAIN, S. and TISO, P. “Simulation-free hyper-reduction for geometrically nonlinear structural dynamics: A quadratic manifold lifting approach”. In: *Journal of Computational and Nonlinear Dynamics* 13.7 (2018).
- [138] JAIN, S. and TISO, P. “Hyper-reduction over nonlinear manifolds for large nonlinear mechanical systems”. In: *J. Comput. Nonlin. Dyn.* 14.8 (2019), p. 081008.
- [139] KOLDA, T. G. and BADER, B. W. “Tensor decompositions and applications”. In: *SIAM review* 51.3 (2009), pp. 455–500.
- [140] KERSCHEN, G. et al. “Nonlinear normal modes, Part I: A useful framework for the structural dynamicist”. In: *Mechanical Systems and Signal Processing* 23.1 (2009), pp. 170–194.
- [141] KRACK, M. and GROSS, J. “The Harmonic Balance Method and its application to nonlinear vibrations: Introduction and current state of the art”. In: *Mechanical Systems and Signal Processing* (2018).
- [142] KRACK, M. and GROSS, J. *Harmonic Balance for Nonlinear Vibration Problems*. Springer, 2019.
- [143] KARACHALIOS, D., GOSEA, I. V., and ANTOULAS, A. C. “On Bilinear Time Domain Identification”. In: *arXiv preprint arXiv:2003.08711* (2020).
- [144] KUNKEL, P. and MEHRMANN, V. *Differential-algebraic equations: analysis and numerical solution*. Vol. 2. European Mathematical Society, 2006.
- [145] KRENER, A. J. “Linearization and bilinearization of control systems”. In: *Proc. 1974 Allerton Conf. on Circuit and System Theory*. Vol. 834. Monticello. 1974.
- [146] KRENER, A. J. “The construction of optimal linear and nonlinear regulators”. In: *Systems, Models and Feedback: Theory and Applications*. Ed. by A. ISIDORI and T. J. TARN. Springer, 1992, pp. 301–322.
- [147] KAWANO, Y. and SCHERPEN, J. M. “Model reduction by differential balancing based on nonlinear Hankel operators”. In: *IEEE Transactions on Automatic Control* 62.7 (2016), pp. 3293–3308.
- [148] KERLER-BACK, J. and STYKEL, T. “Model reduction for linear and nonlinear magneto-quasistatic equations”. In: *International Journal for Numerical Methods in Engineering* 111.13 (2017), pp. 1274–1299.
- [149] KAWANO, Y. and SCHERPEN, J. “Empirical Differential Gramians for Nonlinear Model Reduction”. In: *arXiv preprint arXiv:1902.09836* (2019).

- [150] KRESSNER, D. and TOBLER, C. “Low-rank tensor Krylov subspace methods for parametrized linear systems”. In: *SIAM Journal on Matrix Analysis and Applications* 32.4 (2011), pp. 1288–1316.
- [151] KÜRSCHNER, P. “Efficient low-rank solution of large-scale matrix equations”. PhD thesis. Shaker Verlag Aachen, 2016.
- [152] KÜRSCHNER, P. “Balanced truncation model order reduction in limited time intervals for large systems”. In: *Advances in Computational Mathematics* 44.6 (2018), pp. 1821–1844.
- [153] KUNISCH, K. and VOLKWEIN, S. “Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics”. In: *SIAM Journal on Numerical Analysis* 40.2 (2002), pp. 492–515.
- [154] KUNISCH, K. and VOLKWEIN, S. “Control of the Burgers Equation by a reduced-order approach using Proper Orthogonal Decomposition”. In: *Journal of Optimization Theory and Applications* 102.2 (1999), pp. 345–371.
- [155] LANCZOS, C. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.
- [156] LATTIMER, A. M. “Model Reduction of Nonlinear Fire Dynamics Models”. PhD thesis. Virginia Tech, 2016.
- [157] LAUB, A. et al. “Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms”. In: *IEEE Transactions on Automatic Control* 32.2 (1987), pp. 115–122.
- [158] LIN, Y., BAO, L., and WEI, Y. “Order reduction of bilinear MIMO dynamical systems using new block Krylov subspaces”. In: *Computers & Mathematics with Applications* 58.6 (2009), pp. 1093–1102.
- [159] LEE, K. and CARLBERG, K. T. “Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders”. In: *Journal of Computational Physics* 404 (2020). Article 108973.
- [160] LOHMANN, B. and EID, R. “Efficient order reduction of parametric and nonlinear models by superposition of locally reduced models”. In: *Methoden und Anwendungen der Regelungstechnik. Erlangen-Münchener Workshops*. 2007, pp. 27–36.
- [161] LIANG, Y. et al. “Proper orthogonal decomposition and its applications Part I: Theory”. In: *Journal of Sound and vibration* 252.3 (2002), pp. 527–544.
- [162] LITZ, L. *Reduktion der Ordnung linearer Zustandsraummodelle mittels modaler Verfahren*. Stuttgart: Hochschulverlag, 1979.
- [163] LJUNG, L. *System identification: Theory for the User*. 2nd. Prentice Hall Information and System Sciences Series, 1999.
- [164] LALL, S., MARSDEN, J. E., and GLAVAKI, S. “A subspace approach to balanced truncation for model reduction of nonlinear control systems”. In: *Int. Journal of Robust and Nonlinear Control* 12.6 (2002), pp. 519–535.
- [165] LALL, S., MARSDEN, J. E., and GLAVAKI, S. “Empirical model reduction of controlled nonlinear systems”. In: *IFAC Proceedings Volumes* 32.2 (1999), pp. 2598–2603.

- [166] LIENEMANN, J., RUDNYI, E. B., and KORVINK, J. G. “MST MEMS model order reduction: Requirements and benchmarks”. In: *Linear Algebra and its Applications* 415.2-3 (2006), pp. 469–498.
- [167] LANG, N., SAAK, J., and STYKEL, T. “Balanced truncation model reduction for linear time-varying systems”. In: *Mathematical and Computer Modelling of Dynamical Systems* 22.4 (2016), pp. 267–281.
- [168] LEHOUCQ, R. B., SORENSEN, D. C., and YANG, C. *ARPACK users’ guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. Vol. 6. SIAM, 1998.
- [169] LI, J.-R. and WHITE, J. “Low rank solution of Lyapunov equations”. In: *SIAM Journal on Matrix Analysis and Applications* 24.1 (2002), pp. 260–280.
- [170] LIENEMANN, J., YOUSEFI, A., and KORVINK, J. G. “Nonlinear heat transfer modeling”. In: *Dimension reduction of large-scale systems*. Springer, 2005, pp. 327–331.
- [171] MAYO, A. and ANTOULAS, A. “A framework for the solution of the generalized realization problem”. In: *Linear algebra and its applications* 425.2-3 (2007), pp. 634–662.
- [172] MARSHALL, S. “An approximate method for reducing the order of a linear system”. In: *International Journal of Control* 10.102 (1966), pp. 642–643.
- [173] MCCORMICK, G. P. “Computability of global solutions to factorable nonconvex programs: Part I Convex underestimating problems”. In: *Mathematical programming* 10.1 (1976), pp. 147–175.
- [174] MEINDL, M. et al. “Model order reduction in thermoacoustic stability analysis”. In: *Proceedings of the 9th Vienna International Conference on Mathematical Modelling*. 2018.
- [175] MEYER, C. H. et al. “Model Order Reduction for Parametric Non-linear Mechanical Systems: State of the Art and Future Research”. In: *PAMM* 17.1 (2017), pp. 37–40.
- [176] MEYER, C. D. *Matrix Analysis and Applied Linear Algebra*. Vol. 71. SIAM, 2000.
- [177] MEYER, C. H., GRUBER, F. M., and RIXEN, D. J. “Complex Modal Derivatives For Model Reduction Of Nonclassically Damped, Geometrically Nonlinear Structures”. In: *7th International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering*. 2019.
- [178] MIGNOLET, M. P. et al. “A review of indirect/non-intrusive reduced order modeling of nonlinear geometric structures”. In: *Journal of Sound and Vibration* 332.10 (2013), pp. 2437–2460.
- [179] MEIER, L. and LUENBERGER, D. “Approximation of linear constant systems”. In: *IEEE Transactions on Automatic Control* 12.5 (1967), pp. 585–588.
- [180] MARTINS, N., LIMA, L. T., and PINTO, H. J. “Computing dominant poles of power system transfer functions”. In: *IEEE Transactions on Power Systems* 11.1 (1996), pp. 162–170.
- [181] MOORE, B. “Principal component analysis in linear systems: Controllability, observability, and model reduction”. In: *IEEE Transactions on Automatic Control* 26.1 (1981), pp. 17–32.

- [182] MURAVYOV, A. A. and RIZZI, S. A. “Determination of nonlinear stiffness with application to random vibration of geometrically nonlinear structures”. In: *Computers & Structures* 81.15 (2003), pp. 1513–1523.
- [183] MULLIS, C. and ROBERTS, R. “Synthesis of minimum roundoff noise fixed point digital filters”. In: *IEEE Transactions on Circuits and Systems* 23.9 (1976), pp. 551–562.
- [184] MEHRMANN, V. and STYKEL, T. “Balanced truncation model reduction for large-scale systems in descriptor form”. In: *Dimension Reduction of Large-Scale Systems*. Springer, 2005, pp. 83–115.
- [185] MEYER, D. G. and SRINIVASAN, S. “Balancing and model reduction for second-order form linear systems”. In: *IEEE Transactions on Automatic Control* 41.11 (1996), pp. 1632–1644.
- [186] NAYFEH, A. H. “Order reduction of retarded nonlinear systems—the method of multiple scales versus center-manifold reduction”. In: *Nonlinear Dynamics* 51.4 (2008), pp. 483–500.
- [187] NAYFEH, A. H. *Perturbation methods*. Wiley, 1973.
- [188] NELSON, R. B. “Simplified calculation of eigenvector derivatives”. In: *AIAA Journal* 14.9 (1976), pp. 1201–1205.
- [189] NEWMARK, N. M. “A method of computation for structural dynamics”. In: *Journal of the engineering mechanics division* 85.3 (1959), pp. 67–94.
- [190] NAYFEH, A. H. and MOOK, D. T. *Nonlinear oscillations*. Wiley, 1979.
- [191] NEGRI, F., MANZONI, A., and AMSALLEM, D. “Efficient model reduction of parametrized systems by matrix discrete empirical interpolation”. In: *Journal of Computational Physics* 303 (2015), pp. 431–454.
- [192] NAKATSUKASA, Y., SÈTE, O., and TREFETHEN, L. N. “The AAA algorithm for rational approximation”. In: *SIAM Journal on Scientific Computing* 40.3 (2018), A1494–A1522.
- [193] NOCEDAL, J. and WRIGHT, S. *Numerical optimization*. Springer Science & Business Media, 2006.
- [194] ODABASIOGLU, A., CELIK, M., and PILEGGI, L. T. “PRIMA: Passive reduced-order interconnect macromodeling algorithm”. In: *The Best of ICCAD*. Springer, 2003, pp. 433–450.
- [195] OPMEER, M. R. “Model order reduction by balanced proper orthogonal decomposition and by rational interpolation”. In: *IEEE Transactions on Automatic Control* 57.2 (2011), pp. 472–477.
- [196] PAIGE, C. C. “Computational variants of the Lanczos method for the eigenproblem”. In: *IMA Journal of Applied Mathematics* 10.3 (1972), pp. 373–381.
- [197] PANZER, H. K. et al. “A greedy rational Krylov method for  $\mathcal{H}_2$ -pseudooptimal model order reduction with preservation of stability”. In: *2013 American Control Conference*. IEEE, 2013, pp. 5512–5517.
- [198] PANZER, H. K. “Model order reduction by Krylov subspace methods with global error bounds and automatic choice of parameters”. PhD thesis. Technische Universität München, 2014.

- [199] PAWAR, S. et al. “A deep learning enabler for nonintrusive reduced order modeling of fluid flows”. In: *Physics of Fluids* 31.8 (2019).
- [200] PEETERS, M. et al. “Nonlinear normal modes, Part II: Toward a practical computation using numerical continuation techniques”. In: *Mechanical Systems and Signal Processing* 23.1 (2009), pp. 195–216.
- [201] PENG, Z. K. et al. “Comparisons between harmonic balance and nonlinear output frequency response function in nonlinear system analysis”. In: *Journal of Sound and Vibration* 311.1-2 (2008), pp. 56–73.
- [202] PENZL, T. “A cyclic low-rank Smith method for large sparse Lyapunov equations”. In: *SIAM Journal on Scientific Computing* 21.4 (1999), pp. 1401–1418.
- [203] PFALLER, M. R. et al. “Using parametric model order reduction for inverse analysis of large nonlinear cardiac simulations”. In: *International Journal for Numerical Methods in Biomedical Engineering* 36.4 (2020).
- [204] PEHERSTORFER, B., GUGERCIN, S., and WILLCOX, K. “Data-driven reduced model construction with time-domain Loewner models”. In: *SIAM Journal on Scientific Computing* 39.5 (2017), A2152–A2178.
- [205] PHILLIPS, J. R. “Projection frameworks for model reduction of weakly nonlinear systems”. In: *Proceedings of the 37th Annual Design Automation Conference*. 2000, pp. 184–189.
- [206] PHILLIPS, J. R. “Projection-based approaches for model reduction of weakly nonlinear, time-varying systems”. In: *IEEE TCAD* 22.2 (2003), pp. 171–187.
- [207] PILLAGE, L. T. and ROHRER, R. A. “Asymptotic waveform evaluation for timing analysis”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 9.4 (1990), pp. 352–366.
- [208] PAIGE, C. C. and SAUNDERS, M. A. “Solution of sparse indefinite systems of linear equations”. In: *SIAM journal on numerical analysis* 12.4 (1975), pp. 617–629.
- [209] PAIGE, C. C. and SAUNDERS, M. A. “LSQR: An algorithm for sparse linear equations and sparse least squares”. In: *ACM Transactions on Mathematical Software (TOMS)* 8.1 (1982), pp. 43–71.
- [210] PEREZ, R., WANG, X., and MIGNOLET, M. P. “Nonintrusive structural dynamic reduced order modeling for large deformations: enhancements for complex structures”. In: *Journal of Computational and Nonlinear Dynamics* 9.3 (2014), p. 031008.
- [211] QUARTERONI, A., MANZONI, A., and NEGRI, F. *Reduced basis methods for partial differential equations: an introduction*. Vol. 92. Springer, 2015.
- [212] RAYLEIGH, J. W. S. B. *The theory of sound*. Vol. 2. Dover, 1945.
- [213] REISS, J. et al. “The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena”. In: *SIAM Journal on Scientific Computing* 40.3 (2018), A1322–A1344.
- [214] REWIENSKI, M. J. “A trajectory piecewise-linear approach to model order reduction of nonlinear dynamical systems”. PhD thesis. Massachusetts Institute of Technology, 2003.

- [215] ROMMES, J. and MARTINS, N. “Efficient computation of multivariable transfer function dominant poles using subspace acceleration”. In: *IEEE Transactions on Power Systems* 21.4 (2006), pp. 1471–1483.
- [216] ROBERTS, A. J. *Model emergent dynamics in complex systems*. Vol. 20. SIAM, 2014.
- [217] ROBERTS, A. “Low-dimensional modelling of dynamics via computer algebra”. In: *Computer physics communications* 100.3 (1997), pp. 215–230.
- [218] ROSENBERG, R. M. “The normal modes of nonlinear n-degree-of-freedom systems”. In: *Journal of Applied Mechanics* 29.1 (1962), pp. 7–14.
- [219] RUTZMOSER, J. and RIXEN, D. “A lean and efficient snapshot generation technique for the Hyper-Reduction of nonlinear structural dynamics”. In: *Computer Methods in Applied Mechanics and Engineering* 325 (2017), pp. 330–349.
- [220] REIS, T. and STYKEL, T. “Balanced truncation model reduction of second-order systems”. In: *Mathematical and Computer Modelling of Dynamical Systems* 14.5 (2008), pp. 391–406.
- [221] RUGH, W. J. *Nonlinear system theory. The Volterra/Wiener approach*. J. H. U. Press, 1981.
- [222] RUHE, A. “Rational Krylov sequence methods for eigenvalue computation”. In: *Linear Algebra and its Applications* 58 (1984), pp. 391–405.
- [223] RUTZMOSER, J. B. et al. “Generalization of quadratic manifolds for reduced order modeling of nonlinear structural dynamics”. In: *Computers & Structures* 192 (2017), pp. 196–209.
- [224] RUTZMOSER, J. B. “Model Order Reduction for Nonlinear Structural Dynamics: Simulation-free Approaches”. PhD thesis. Technische Universität München, 2018.
- [225] REWIENSKI, M. and WHITE, J. “A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 22.2 (2003), pp. 155–170.
- [226] SASSANO, M. and ASTOLFI, A. “Dynamic approximate solutions of the HJ inequality and of the HJB equation for input-affine nonlinear systems”. In: *IEEE Transactions on Automatic Control* 57.10 (2012), pp. 2490–2503.
- [227] SASSANO, M. and ASTOLFI, A. “Dynamic generalized controllability and observability functions with applications to model reduction and sensor deployment”. In: *Automatica* 50.5 (2014), pp. 1349–1359.
- [228] SCARCIOTTI, G. and ASTOLFI, A. “Model Reduction of Neutral Linear and Nonlinear Time-Invariant Time-Delay Systems With Discrete and Distributed Delays.” In: *IEEE TAC* 61.6 (2016), pp. 1438–1451.
- [229] SCARCIOTTI, G. and ASTOLFI, A. *Nonlinear Model Reduction by Moment Matching*. Foundations, Trends in Systems, and Control, Now Publishers Inc., 2017. URL: <https://ieeexplore.ieee.org/document/8187282>.
- [230] SCARCIOTTI, G. and ASTOLFI, A. “A review on model reduction by moment matching for nonlinear systems”. In: *Feedback Stabilization of Controlled Dynamical Systems*. Springer, 2017, pp. 29–52.

- [231] SCARCIOTTI, G. and ASTOLFI, A. “Data-driven model reduction by moment matching for linear and nonlinear systems”. In: *Automatica* 79 (2017), pp. 340–351.
- [232] SAAD, Y. *Iterative methods for sparse linear systems*. SIAM, 2003.
- [233] SAAD, Y. *Numerical methods for large eigenvalue problems: revised edition*. Vol. 66. SIAM, 2011.
- [234] SAAD, Y. “Variations on Arnoldi’s method for computing eigenelements of large unsymmetric matrices”. In: *Linear algebra and its applications* 34 (1980), pp. 269–295.
- [235] SAAD, Y. “Numerical Solution of Large Lyapunov Equations”. In: *Signal Processing, Scattering and Operator Theory, and Numerical Methods, Proc. MTNS-89*. Birkhauser, 1990, pp. 503–511.
- [236] SALIMBAHRAMI, S. B. “Structure preserving order reduction of large scale second order models”. PhD thesis. Technische Universität München, 2005.
- [237] STOER, J. and BULIRSCH, R. *Introduction to numerical analysis*. Vol. 12. Springer Science & Business Media, 2013.
- [238] SCHMID, P. J. “Dynamic mode decomposition of numerical and experimental data”. In: *Journal of Fluid Mechanics* 656 (2010), pp. 5–28.
- [239] SCHERPEN, J. M. A. “Balancing for nonlinear systems”. In: *Systems & Control Letters* 21.2 (1993), pp. 143–153.
- [240] SLAATS, P., DE JONGH, J., and SAUREN, A. “Model reduction tools for nonlinear structural dynamics”. In: *Computers & Structures* 54.6 (1995), pp. 1155–1171.
- [241] SEYDEL, R. *Practical bifurcation and stability analysis*. Vol. 5. Springer Science & Business Media, 2009.
- [242] SHAMASH, Y. “Model reduction using the Routh stability criterion and the Padé approximation technique”. In: *International Journal of Control* 21.3 (1975), pp. 475–484.
- [243] SHAMPINE, L. F. *Numerical Solution of Ordinary Differential Equations*. Vol. 4. CRC Press, 1994.
- [244] SILVEIRA, L. M. et al. “A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits”. In: *Computer Methods in Applied Mechanics and Engineering* 169.3-4 (1999), pp. 377–389.
- [245] SIMONCINI, V. “A new iterative method for solving large-scale Lyapunov matrix equations”. In: *SIAM Journal on Scientific Computing* 29.3 (2007), pp. 1268–1288.
- [246] SIMONCINI, V. “Computational methods for linear matrix equations”. In: *SIAM Review* 58.3 (2016), pp. 377–441.
- [247] SCARCIOTTI, G., JIANG, Z.-P., and ASTOLFI, A. “Data-driven constrained optimal model reduction”. In: *European Journal of Control* (2019).
- [248] SAAK, J., KÖHLER, M., and BENNER, P. *M-M.E.S.S.-1.0.1 – The Matrix Equations Sparse Solvers library*. DOI:10.5281/zenodo.50575. see also: <https://www.mpi-magdeburg.mpg.de/projects/mess>. Apr. 2016.
- [249] SELGA, R. C., LOHMANN, B., and EID, R. “Stability preservation in projection-based model order reduction of large scale systems”. In: *European Journal of Control* 18.2 (2012), pp. 122–132.

- [250] SOMBROEK, C. et al. “Bridging the gap between nonlinear normal modes and modal derivatives”. In: *Nonlinear Dynamics, Volume 1*. Springer, 2016, pp. 349–361.
- [251] SOMBROEK, C. S. M. et al. “Numerical computation of nonlinear normal modes in a modal derivative subspace”. In: *Computers & Structures* 195 (2018), pp. 34–46.
- [252] SHAW, S. W. and PIERRE, C. “Normal modes for non-linear vibratory systems”. In: *Journal of Sound and Vibration* 164.1 (1993), pp. 85–124.
- [253] SANDBERG, H. and RANTZER, A. “Balanced truncation of linear time-varying systems”. In: *IEEE Transactions on Automatic Control* 49.2 (2004), pp. 217–229.
- [254] SON, N. T. and STYKEL, T. “Solving parameter-dependent Lyapunov equations using the reduced basis method with application to parametric model order reduction”. In: *SIAM Journal on Matrix Analysis and Applications* 38.2 (2017), pp. 478–504.
- [255] SAAD, Y. and SCHULTZ, M. H. “GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems”. In: *SIAM Journal on scientific and statistical computing* 7.3 (1986), pp. 856–869.
- [256] SIU, T. and SCHETZEN, M. “Convergence of Volterra series representation and BIBO stability of bilinear systems”. In: *International journal of systems science* 22.12 (1991), pp. 2679–2684.
- [257] SHANK, S. D., SIMONCINI, V., and SZYLD, D. B. “Efficient low-rank solution of generalized Lyapunov equations”. In: *Numerische Mathematik* 134.2 (2016), pp. 327–342.
- [258] SHOKOOHI, S., SILVERMAN, L., and VAN DOOREN, P. “Linear time-variable systems: Balancing and model reduction”. In: *IEEE Transactions on Automatic Control* 28.8 (1983), pp. 810–822.
- [259] STYKEL, T. “Gramian-based model reduction for descriptor systems”. In: *Mathematics of Control, Signals and Systems* 16.4 (2004), pp. 297–319.
- [260] SWISCHUK, R. et al. “Projection-based model reduction: Formulations for physics-based machine learning”. In: *Computers & Fluids* 179 (2019), pp. 704–717.
- [261] TISO, P. and RIXEN, D. J. “Discrete empirical interpolation method for finite element structural dynamics”. In: *Topics in Nonlinear Dynamics, Volume 1*. Springer, 2013, pp. 203–212.
- [262] VAN DER VORST, H. A. “Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems”. In: *SIAM Journal on scientific and Statistical Computing* 13.2 (1992), pp. 631–644.
- [263] VARGA, A. “Enhanced modal approach for model reduction”. In: *Mathematical Modelling of Systems* 1.2 (1995), pp. 91–105.
- [264] VERHULST, F. *Nonlinear Differential Equations and Dynamical Systems*. Second edition. Springer, 1996.
- [265] VILLEMAGNE, C. and SKELTON, R. E. “Model reductions using a projection formulation”. In: *International Journal of Control* 46.6 (1987), pp. 2141–2169.
- [266] VANDEREYCKEN, B. and VANDEWALLE, S. “A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations”. In: *SIAM Journal on Matrix Analysis and Applications* 31.5 (2010), pp. 2553–2579.
- [267] WACHSPRESS, E. L. “Iterative solution of the Lyapunov matrix equation”. In: *Applied Mathematics Letters* 1.1 (1988), pp. 87–90.



- [268] WAKASA, T. “Exact eigenvalues and eigenfunctions associated with linearization for Chafee-Infante problem”. In: *Funkcialaj Ekvacioj* 49.2 (2006), pp. 321–336.
- [269] WALTER, W. *Ordinary Differential Equations*. Springer, 1998.
- [270] WILLCOX, K. “Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition”. In: *Computers & fluids* 35.2 (2006), pp. 208–226.
- [271] WILSON, D. A. “Optimum solution of model-reduction problem”. In: *Proceedings of the Institution of Electrical Engineers*. Vol. 117. 6. IET. 1970, pp. 1161–1165.
- [272] WOLF, T. et al. “Passivity and structure preserving order reduction of linear port-Hamiltonian systems using Krylov subspaces”. In: *European Journal of Control* 16.4 (2010), pp. 401–406.
- [273] WOLF, T. “ $\mathcal{H}_2$  pseudo-optimal model order reduction”. PhD thesis. Technische Universität München, 2014.
- [274] WILLCOX, K. and PERAIRE, J. “Balanced model reduction via the proper orthogonal decomposition”. In: *AIAA journal* 40.11 (2002), pp. 2323–2330.
- [275] WOLF, T. and PANZER, H. K. “The ADI iteration for Lyapunov equations implicitly performs  $\mathcal{H}_2$  pseudo-optimal model order reduction”. 2013. URL: <https://arxiv.org/abs/1309.3985>.
- [276] WOLF, T., PANZER, H., and LOHMANN, B. “Gramian-based error bound in model reduction by Krylov subspace methods”. In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 3587–3592.
- [277] WOLF, T., PANZER, H. K., and LOHMANN, B. “ $\mathcal{H}_2$  pseudo-optimality in model order reduction by Krylov subspace methods”. In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 3427–3432.
- [278] WOLF, T., PANZER, H. K., and LOHMANN, B. “On the residual of large-scale Lyapunov equations for Krylov-based approximate solutions”. In: *2013 American Control Conference*. IEEE. 2013, pp. 2606–2611.
- [279] WOLF, T., PANZER, H. K., and LOHMANN, B. “ADI iteration for Lyapunov equations: A tangential approach and adaptive shift selection”. In: *Applied Numerical Mathematics* 109 (2016), pp. 85–95.
- [280] WIRTZ, D., SORENSEN, D. C., and HAASDONK, B. “A posteriori error estimation for DEIM reduced nonlinear dynamical systems”. In: *SIAM Journal on Scientific Computing* 36.2 (2014), A311–A338.
- [281] WEEGER, O., WEVER, U., and SIMEON, B. “Nonlinear frequency response analysis of structural vibrations”. In: *Computational Mechanics* 54.6 (2014), pp. 1477–1495.
- [282] WEEGER, O., WEVER, U., and SIMEON, B. “On the use of modal derivatives for nonlinear model order reduction”. In: *International Journal for Numerical Methods in Engineering* 108.13 (2016), pp. 1579–1602.
- [283] YOUSEFI, A. “Preserving Stability in Model and Controller Reduction with application to embedded systems”. PhD thesis. Technische Universität München, 2006.
- [284] ZAKIAN, V. “Simplification of linear time-invariant systems by moment approximants”. In: *International Journal of Control* 18.3 (1973), pp. 455–460.
- [285] ZHANG, L. and LAM, J. “On  $H_2$  model reduction of bilinear systems”. In: *Automatica* 38.2 (2002), pp. 205–216.

## Selected talks and workshops

The following list includes some selected talks held by the author at seminars and workshops. The slides are available online.

- [Cru16a] CRUZ VARONA, M. *Interpolation-based model reduction of nonlinear systems*. CSC Seminar, Max-Planck-Institut für Dynamik komplexer technischer Systeme. <https://mediatum.ub.tum.de/node?id=1339234>. Feb. 2016.
- [Cru16b] CRUZ VARONA, M. *Krylov subspace model reduction for bilinear and MIMO quadratic-bilinear systems*. Workshop on Model Order Reduction and Optimization, Opatija. <https://mediatum.ub.tum.de/node?id=1339238>. Sept. 2016.
- [Cru17a] CRUZ VARONA, M. *On the  $\mathcal{H}_2$ -pseudo-optimal bilinear model reduction*. Applied Numerical Analysis Seminar, Virginia Tech, Blacksburg. <https://mediatum.ub.tum.de/node?id=1367498>. Apr. 2017.
- [Cru17b] CRUZ VARONA, M. *Two-sided Moment Matching-Based Reduction for MIMO Quadratic-Bilinear Systems*. 11. Elgersburg Workshop. <https://mediatum.ub.tum.de/node?id=1367497>. Feb. 2017.
- [Cru18] CRUZ VARONA, M. *An Introduction to Model Order Reduction: from linear to nonlinear dynamical systems*. Seminar Automatisierungstechnik, Institut für Regelungs- und Steuerungssysteme (IRS), Karlsruhe. <https://mediatum.ub.tum.de/node?id=1446315>. June 2018.
- [Cru19] CRUZ VARONA, M. *Parametric Model Order Reduction: An Introduction*. Model Order Reduction Summer School (MORSS), Eindhoven, <https://mediatum.ub.tum.de/node?id=1538582>. Sept. 2019.
- [CFL17] CRUZ VARONA, M., FIORELISIO JUNIOR, E., and LOHMANN, B. *Krylov subspace methods for model reduction of MIMO quadratic-bilinear systems*. 3rd Workshop on Model Reduction of Complex Dynamical Systems (MODRED), Odense. <https://mediatum.ub.tum.de/node?id=1344241>. Jan. 2017.
- [CL15a] CRUZ VARONA, M. and LOHMANN, B. *Model Order Reduction of Linear Time-Varying Systems: Some straightforward approaches*. MOR 4 MEMS Workshop, Karlsruhe. <https://mediatum.ub.tum.de/node?id=1286603>. Nov. 2015.
- [CL15b] CRUZ VARONA, M. and LOHMANN, B. *Time-Varying Parametric Model Order Reduction by Matrix Interpolation*. Model Reduction of Parametrized Systems (MoRePaS) III, Trieste. <https://mediatum.ub.tum.de/node?id=1286602>. Oct. 2015.
- [COL16] CRUZ VARONA, M., OLCAY, E., and LOHMANN, B. *Interpolation-based  $\mathcal{H}_2$ -pseudo-optimal model reduction of bilinear systems*. GMA Workshop, Fachausschuss 1.30, Anif. <https://mediatum.ub.tum.de/node?id=1339237>. Sept. 2016.

## Supervised student theses

The following student theses were written at the Chair of Automatic Control under essential scientific, technical and content-related supervision from the author of this dissertation. Due to the lack of time, only two student theses could be made available online. If the reader is interested in the details, code and/or documentation of any of the theses, he/she can contact me or the author of the student thesis.

- [Bil19] BILFINGER, P. *Numerical Examination of Volterra Series-Based Modal Derivatives and their Impact on Nonlinear Model Order Reduction*. Semester thesis. Sept. 2019.
- [Fio16] FIORELISIO JUNIOR, E. *Krylov Subspace Methods for Model Reduction of MIMO Quadratic-Bilinear Systems*. Master thesis. Oct. 2016.
- [Geb17] GEBHART, R. *Impulsantwort bilinearer Systeme – Systemtheorie und Modellreduktion*. Semesterarbeit. Mar. 2017. URL: <https://mediatum.ub.tum.de/node?id=1416277>.
- [Gri16] GRIMM, F. *Model Order Reduction of nonlinear, parametric systems using Trajectory Piecewise Linear Approximation*. Master thesis. May 2016.
- [Hei17] HEIDENREICH, P. *Efficient Krylov-Based Low Rank Solution for Generalised Lyapunov Equations*. Semester thesis. Mar. 2017.
- [Hei18] HEIDENREICH, P. *Krylov Subspace Theory and Non-Linear Balanced Truncation as an Optimal Control Problem*. Master thesis. Feb. 2018.
- [Him18] HIMPSL, F. *Simulation-free model order reduction of non-linear state-space models by means of basis vector derivatives*. Master thesis, Co-supervised by C. Lerch. Jan. 2018.
- [Lep17] LEPPMEIER, A. *Generation of nonlinear benchmark models for model order reduction*. Semester thesis. Mar. 2017.
- [Olc16] OLCAY, E. *Interpolation-based  $\mathcal{H}_2$ -Optimal Model Reduction of Bilinear Systems*. Master thesis. July 2016.
- [Röt18] RÖTZER, P. *Implementation of Model Reduction of Bilinear Systems and Extension of the Multipoint Volterra Series Interpolation*. Bachelor thesis. Nov. 2018. URL: <https://mediatum.ub.tum.de/node?id=1469768>.
- [Sch18] SCHNEUCKER, N. *Development and Numerical Research on a Simulation-Free Method for Nonlinear Model Order Reduction*. Bachelor thesis. Nov. 2018.
- [Suk16] SUK, J. *Simulationsbasierte Modellreduktion von parametrisierten nichtlinearen Systemen*. Bachelorarbeit. Sept. 2016.
- [Suk17] SUK, J. *Applied Simulation-Free Nonlinear Model Order Reduction Through Moment Matching*. Semester thesis. Oct. 2017.

## Co-supervised student theses

The following student theses were written at the Chair of Automatic Control under scientific, technical and content-related co-supervision from the author of this dissertation. If the reader is interested in the details and/or documentation of any of the theses, he/she can contact me, the other supervisor or the author of the student thesis.

- [Jes15] JESCHEK, L. *Implementation and Investigation of the Newton-IRKA Algorithm for Krylov-Based,  $\mathcal{H}_2$ -Optimal Model Order Reduction*. Semester thesis, Co-supervised with A. Castagnotto. July 2015.
- [Lan17] LANG, J. *Projection-based Parametric Model Order Reduction for 3D-0D Coupled Cardiac Mechanics*. Master thesis, In cooperation with M. Pfaller (Institute for Computational Mechanics). Sept. 2017.
- [Män18] MÄNNIG, V. *Model Order Reduction in Vibro Acoustics: An application of the sss/sssMOR toolboxes on vibro acoustic FEM and IFEM models*. Bachelor thesis, In cooperation with L. Moheit (Chair of Vibro-Acoustics of Vehicles and Machines). Aug. 2018.
- [Mec18] MECHLER, M. *Projection-based Hyperreduction for 3D-0D Coupled Cardiovascular Mechanics*. Master thesis, In cooperation with M. Pfaller (Institute for Computational Mechanics). June 2018.
- [Mor15] MOREIRA SILVA, J. L. *Untersuchung des Konvergenzverhaltens vom IRKA-Algorithmus und Implementierung einer modifizierten IRKA-Strategie*. Semesterarbeit, Ko-betreut mit A. Castagnotto. Aug. 2015.
- [Tho16] THOMANN, F. *Modellordnungsreduktion von gekoppelten thermoakustischen Problemen*. Semesterarbeit, In Kooperation mit M. Meindl (Professur für Thermofluidynamik). Dec. 2016.