Technische Universität München

Fakultät für Informatik

LATENT MATTERS

Amortised Variational Inference With Constrained Optimisation and Learnable Priors

ALEXEJ KLUSHYN

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

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr. Stephan Günnemann
Prüfer der Dissertation:	1.	Prof. Dr. Hans-Joachim Bungartz
	2.	apl. Prof. Dr. Georg Groh

Die Dissertation wurde am 28.06.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 08.10.2021 angenommen.

Alexej Klushyn
Latent Matters: Amortised Variational Inference With Constrained Optimisation and Learnable Priors

ABSTRACT

Probabilistic generative modelling is a powerful framework for describing the world on the basis of observations and accounting for the resulting uncertainty. In order to increase their expressiveness, generative models are often defined as latent-variable models, i.e. by a hierarchy of latent (unobservable) and observable variables, where our knowledge is represented by the corresponding conditional probability distribution. A key challenge in this regard is the inference of the latent variables that underlie our observations and are assumed to capture the characteristics of the data.

The posterior distribution defined by Bayes' theorem provides a latent representation of the observed data. However, it is computationally intractable in all but the simplest cases due to an intractable normalising constant. Amortised variational inference allows us to approximate the posterior distribution by framing inference as an optimisation problem and leveraging the expressive power of neural networks. Nevertheless, current approaches often lead to an over-regularisation of the approximate posterior distribution, which considerably limits its informative value.

The focus of this dissertation is on learning latent representations—with the corresponding generative models—that reflect the factors of variation and topology of the observed data. To this end, we propose a constrained optimisation-based formulation of amortised variational inference and complement it with a powerful empirical Bayes method. We first demonstrate the effectiveness of this approach within the framework of variational autoencoders. We then show in the context of dynamic systems that our approach significantly improves system identification in deep state-space models, which is accompanied by an increased prediction accuracy of the models. Another area where our approach provides substantial benefits is unsupervised metric learning: the learned latent representations allow us to create similarity measures of data which achieve the performance of state-of-the-art supervised learning methods in object tracking and re-identification.

ZUSAMMENFASSUNG

Die probabilistische generative Modellierung ist eine leistungsstarke Methode, um die Welt auf Grundlage von Beobachtungen zu beschreiben und die dabei entstehende Unsicherheit zu berücksichtigen. Um ihre Ausdruckskraft zu erhöhen, werden generative Modelle oft als latente Variablenmodelle definiert, d.h. durch eine Hierarchie von latenten (unbeobachtbaren) und beobachtbaren Variablen, wobei unser Wissen durch die entsprechende bedingte Wahrscheinlichkeitsverteilung repräsentiert wird. Eine zentrale Herausforderung in diesem Zusammenhang ist die Inferenz latenter Variablen, die unseren Beobachtungen zugrunde liegen und von denen angenommen wird, dass sie die Charakteristika der Daten erfassen.

Die durch den Satz von Bayes definierte A-posteriori-Verteilung liefert eine latente Darstellung der beobachteten Daten. Eine Berechnung ist jedoch in meisten Fällen nicht möglich, da die Normierungskonstante nicht berechenbar ist. Amortisierte Variationsinferenz ermöglicht es uns, die A-posteriori-Verteilung zu approximieren, indem wir die Inferenz als Optimierungsproblem auffassen und dabei die Ausdruckskraft neuronaler Netze nutzen. Dennoch führen aktuelle Ansätze oft zu einer Überregularisierung der approximierten A-posteriori-Verteilung, was ihre Aussagekraft erheblich einschränkt.

Der Schwerpunkt dieser Dissertation liegt auf dem Erlernen latenter Repräsentationen – mit den entsprechenden generativen Modellen – welche die Faktoren der Variation und die Topologie der beobachteten Daten widerspiegeln. Zu diesem Zweck stellen wir eine, auf eingeschränkter Optimierung basierende, Formulierung der amortisierten Variationsinferenz vor und ergänzen diese durch eine leistungsstarke empirische Bayes-Methode. Wir demonstrieren die Effektivität dieses Ansatzes zunächst im Rahmen von Variational Autoencodern. Anschließend zeigen wir im Kontext dynamischer Systeme, dass unser Ansatz die Systemidentifikation in tiefen Zustandsraummodellen signifikant verbessert, was mit einer erhöhten Vorhersagegenauigkeit der Modelle einhergeht. Ein weiterer Bereich, in dem unser Ansatz erhebliche Vorteile bietet, ist das unüberwachte Lernen von Metriken: Die erlernten latenten Repräsentationen ermöglichen es uns, Ähnlichkeitsmaße von Daten zu erstellen, welche die Leistung modernster überwachter Lernmethoden bei der Objektverfolgung und Re-Identifizierung erreichen.

PREFACE

During my PhD, I was fortunate to work on various research projects that have been the main source of inspiration for this thesis. Some of these projects resulted in publications, which are related to the contributions of this work as follows:

Chapter 3 addresses the question of how to learn latent representations with variational autoencoders that reflect the factors of variation and topology of observed data. It lays the foundation for the methods presented in the following chapters and incorporates previously published work from

- Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke and Patrick van der Smagt (2019b). 'Learning Hierarchical Priors in VAEs'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. Curran Associates, Inc., pp. 2870–2879, **Spotlight Presentation**

Compared to the publication, the introduced method is explained in more detail; in Section 3.2, an additional analysis provides a deeper insight into the theory of the method. Furthermore, results that have only appeared in the appendix of the publication are incorporated and discussed in Section 3.5.

Chapter 4 deals with the question of how to accurately predict observed dynamic systems. To this end, the method presented in Chapter 3 is extended to enable the learning of deep state-space models. This chapter is based on recently published work:

- Alexej Klushyn, Richard Kurle, Maximilian Soelch, Botond Cseke and Patrick van der Smagt (2021). 'Latent Matters: Learning Deep State-Space Models'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34. Curran Associates, Inc.

In Chapter 5, we return to the framework of variational autoencoders and discuss how the method presented in Chapter 3 can be applied to measure the similarity of data. This chapter incorporates work from the following two publications:

- Nutan Chen*, Alexej Klushyn*, Richard Kurle*, Xueyan Jiang, Justin Bayer and Patrick van der Smagt (2018). 'Metrics for Deep Generative Models'. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 84. PMLR, pp. 1540–1550

* denotes equal contribution

- Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer and Patrick van der Smagt (2020). 'Learning Flat Latent Manifolds With VAEs'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 119. PMLR, pp. 1587–1596

Further publications I have contributed to are listed below. These are mentioned as related work but not incorporated into this thesis.

- Alexej Klushyn, Nutan Chen, Botond Cseke, Justin Bayer and Patrick van der Smagt (2019a). 'Increasing the Generalisation Capacity of Conditional VAEs'. In: *International Conference on Artificial Neural Networks (ICANN)*. Vol. 11728. Springer, pp. 779–791, **Oral Presentation**

- Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick van der Smagt and Stephan Günnemann (2020). 'Continual Learning With Bayesian Neural Networks for Non-Stationary Data'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net

- Nutan Chen, Francesco Ferroni, Alexej Klushyn, Alexandros Paraschos, Justin Bayer and Patrick van der Smagt (2019). 'Fast Approximate Geodesics for Deep Generative Models'. In: *International Conference on Artificial Neural Networks (ICANN)*. Vol. 11728. Springer, pp. 554–566

- Nutan Chen, Alexej Klushyn, Alexandros Paraschos, Djalel Benbouzid and Patrick van der Smagt (2018). 'Active Learning Based on Data Uncertainty and Model Sensitivity'. In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1547–1554

- Nutan Chen*, Alexej Klushyn*, Richard Kurle*, Xueyan Jiang, Justin Bayer and Patrick van der Smagt (2017). 'Metrics for Deep Generative Models Based on Learned Skills'. In: *Advances in Neural Information Processing Systems (NeurIPS), Workshop on Acting and Interacting in the Real World: Challenges in Robot Learning*

ACKNOWLEDGEMENTS

This thesis would not have been possible without the people who have accompanied me over the last four years.

I want to thank first my advisors Prof. Dr. Patrick van der Smagt and Prof. Dr. Hans-Joachim Bungartz. Patrick gave me this unique opportunity and encouraged me to find my own way. Prof. Bungartz helped me not to lose sight of the actual goal.

Special thanks belong to Dr. Botond Cseke, Dr. Djalel Benbouzid, Dr. Nutan Chen, and Richard Kurle. Botond, I only realised how much I had learned from you when I wrote this thesis. Hopefully, you will share your expertise with many more budding researchers. Djalel, I have constantly refused to *play it by ear*, but you made sure I listened to my gut feeling. Nutan and Richard, together we mastered our research projects, and at the crucial moments you gave me confidence.

I feel fortunate to have been part of the Volkswagen MLRL and want to sincerely thank my lab mates for the wonderful time: Adnan Akhundov, Dr. Alexandros Paraschos, Atanas Mirchev, Baris Kayalibay, Eileen Zhang, Felix Frank, Dr. Grady Jensen, Dr. Justin Bayer, Karolina Stosio, Dr. Maximilian Karl, Maximilian Soelch, Michelle Ploetner, and Philip Becker-Ehmck. Working with you was inspiring, challenging, and always incredibly fun.

Besonderer Dank gebührt auch meinen Freunden Fabi, Flo, Alex, Hannah, Lukas, Bardhi, Mel, Alex und Stephie. Ihr habt begnadet dafür gesorgt, dass mein Leben im richtigen Verhältnis zur Arbeit steht. Alex, unsere Läufe haben mir nicht selten den Tag gerettet, insbesondere im letzten Jahr.

Нет слов, чтобы выразить глубину моей благодарности моим родителям Лене и Славе. Вы всегда поощряли меня в любых моих начинаниях. Поэтому я был и остаюсь уверен: в любом моем решении я могу полагаться на вашу поддержку.

Alina, wie sehr du mich in den letzten Jahren unterstützt hast, kann ich nicht in Worte fassen und Worte würden dir auch nicht gerecht werden.

CONTENTS

1 INTRODUCTION 1

2 FUNDAMENTALS 3

2.1 Latent-Variable Models 3

2.1.1 Difficulties With Classical Learning Approaches 4

2.1.2 Variational Inference 5

2.2 Amortised Variational Inference Within the Framework of Variational Autoencoders 8

2.2.1 Variational Autoencoders 8

2.2.2 Beyond Gaussian Posteriors 14

2.2.3 Learnable Priors 15

2.3 Extension to Sequential Data 16

2.3.1 State-Space Models 17

2.3.2 Bayesian Filtering and Smoothing 18

2.3.3 Amortised Variational Inference 23

3 LEARNING INFORMATIVE LATENT REPRESENTATIONS WITH VARIATIONAL AUTOENCODERS 25

3.1 Variational Autoencoders as a Constrained Optimisation Problem 26

3.2 Analysis of the Constrained Optimisation-Based Formulation 28

3.3 Hierarchical Priors for Learning Informative Latent Representations 32

3.4 Graph-Based Interpolation for Verifying Latent Representations 36

3.5 Experimental Results 36

3.6 Summarising Discussion 50

4 EXTENSION TO SEQUENTIAL DATA: LEARNING DEEP STATE-SPACE MODELS 51

4.1 Constrained Optimisation Framework for Improved System Identification 52

4.2 Extended Kalman Variational Autoencoder 60

4.3 Disentangling Static and Dynamic Features for Policy Learning 67

4.4 Experimental Results 69
4.5 Summarising Discussion 85

5 UNSUPERVISED METRIC LEARNING WITH VARIATIONAL AUTOENCODERS 87
5.1 The Latent Space as a Riemannian Manifold 88
5.1.1 Riemannian Geometry in the Context of Variational Autoencoders 89
5.1.2 Approximating Geodesics 92
5.1.3 Visualising the Riemannian Metric Tensor 94
5.1.4 Experimental Results 94
5.1.5 Summarising Discussion 102
5.2 The Latent Space as Euclidean Similarity Metric 103
5.2.1 Flat Latent Manifolds 104
5.2.2 Learning Flat Latent Manifolds With Variational Autoencoders 105
5.2.3 Experimental Results 108
5.2.4 Summarising Discussion 120

6 CONCLUSION AND OUTLOOK 121

A APPENDIX 123
A.1 Model Architectures 123

BIBLIOGRAPHY 129

ACRONYMS

CO	constrained optimisation
CNN	convolutional neural network
DKF	deep Kalman filter
DKS	deep Kalman smoother
DSSM	deep state-space model
DVBF	deep variational Bayes filter
DVBS	deep variational Bayes smoother
EKVAE	extended Kalman variational autoencoder
ELBO	evidence lower bound
EM	expectation maximisation
FMVAE	flat manifold variational autoencoder
GECO	generalized ELBO with constrained optimization
GRU	gated recurrent unit
i.i.d.	independent and identically distributed
IoU	intersection over union
IWAE	importance-weighted autoencoder
KL	Kullback-Leibler divergence
KVAE	Kalman variational autoencoder
LSTM	long short-term memory
ML	maximum likelihood
MLP	multilayer perceptron
MSE	mean squared error
NN	neural network
OLS	ordinary least squares
SGD	stochastic gradient descent
SGVB	stochastic gradient variational Bayes
REWO	reconstruction-error-based weighting of the objective function
RNN	recurrent neural network
RSSM	recurrent state-space model
SGD	stochastic gradient decent
SIR	sequence importance resampling
SSM	state-space model

VAE	variational autoencoder
VHP	variational hierarchical prior
WU	warm up

1 INTRODUCTION

An essential aspect of human learning is to observe. Based on new observations, we create an increasingly precise model of our environment. Accordingly, this model can never be perfect. Both our model and the knowledge of its inaccuracy are crucial sources of information for our decision-making process.

Probabilistic generative modelling follows this principle. It is a powerful framework for describing the world on the basis of observations, taking into account the uncertainty involved. We naturally want generative models to be as expressive as possible. For this reason, they are often defined as latent-variable models, i.e. by a probability distribution over observations conditioned on latent (unobservable) variables. The latent variables are assumed to capture the underlying characteristics of the data, and the conditional distribution represents our knowledge about the observed system. A key challenge in learning such models is the inference of latent variables.

Bayes' theorem defines the posterior distribution over the latent variables, providing a latent representation of the observed data. However, the posterior distribution is computationally intractable in all but the simplest cases due to an intractable normalising constant. Amortised variational inference offers a solution to this problem: it enables the approximation of the posterior distribution by framing inference as an optimisation problem and opens up the possibility to leverage the expressive power of neural networks (Kingma and Welling, 2014; Rezende et al., 2014). Nevertheless, current approaches often lead to an over-regularisation of the approximate posterior distribution (Alemi et al., 2018; Higgins et al., 2017; Tomczak and Welling, 2018), which considerably limits the informative value of the learned latent representation, as we show in our experiments. The informative value of the inferred latent representation, however, has a significant impact on the quality of the learned model (X. Chen et al., 2017; Kingma et al., 2016; Rezende and Mohamed, 2015).

In this thesis, we address the question of how to learn latent representations—with the corresponding models—that reflect the factors of variation and topology of the observed data. Previous work has tackled this by either improving the optimisation process (Bowman et al., 2016; Higgins et al., 2017; Rezende and Viola, 2018) or the capacity of the model (X. Chen et al., 2017; Tomczak and Welling, 2018). Our approach builds on both concepts: we propose a constrained optimisation-based formulation of amortised variational inference and complement it with a powerful empirical Bayes method. Moreover, we

demonstrate that latent representations learned this way are useful for various applications, such as model-based reinforcement learning or unsupervised metric learning.

Chapter 2 covers the fundamentals of Bayesian inference that are essential for understanding our approach and the related methods. First, we motivate and introduce latent-variable models in the context of probabilistic generative modelling. Building upon this, we show how inference in latent-variable models can be framed as an optimisation problem. We then present amortised variational inference in the context of variational autoencoders (VAEs), i.e. for learning latent-variable models on the basis of independent and identically distributed (i.i.d.) data. Finally, we extend the introduced concepts to sequential Bayesian inference, allowing us to build and learn probabilistic models of observed dynamic systems.

In Chapter 3, we introduce our approach within the framework of VAEs and derive the theoretical foundations for the constrained optimisation-based formulation of amortised variational inference. This lays the groundwork for the methods presented in the following chapters of this thesis. In contrast to the original VAE, our method avoids an over-regularisation of the approximate posterior distribution and facilitates learning informative latent representations that reflect the factors of variation and topology of the observed data.

Chapter 4 addresses the question of how to accurately predict observed dynamic systems. To this end, we apply our approach to sequential data by introducing a constrained optimisation framework for learning deep state-space models (DSSMs). Building upon this, we combine amortised variational inference with classic Bayesian filtering and smoothing to improve approximate inference for DSSMs. In contrast to previous approaches, our method facilitates system identification, with the consequence that the learned latent variables represent the true underlying state of the observed dynamic system. The result is a substantial increase in prediction accuracy. Furthermore, we use our method to learn state-space representations where static and dynamic features are disentangled and show how these can be applied in model-based reinforcement learning.

In Chapter 5, we focus on another area where our approach provides considerable advantages: unsupervised metric learning. In order to find a metric that measures the similarity of i.i.d. data, we first analyse the topology of latent spaces learned by VAEs with the help of Riemannian geometry. Based on this, we combine our method introduced in Chapter 3 with a regularisation approach that allows us to learn latent spaces where the Euclidean distance is a similarity measure for observed data. We use this method for object tracking and re-identification and show that it can compete with state-of-the-art supervised learning methods.

2

FUNDAMENTALS

In this chapter, we cover the fundamentals of Bayesian inference that are essential for understanding the methods presented in this thesis. In Section 2.1, we motivate and introduce latent-variable models in the context of probabilistic generative modelling of observed systems. Building upon this, we show how inference in latent-variable models can be framed as an optimisation problem. In Section 2.2, we present amortised variational inference within the framework of variational autoencoders, where the expressive power of neural networks is leveraged. We then extend in Section 2.3 the introduced concepts to sequential Bayesian inference, which allows us to build and learn probabilistic models of observed dynamic systems.

In addition to this chapter, we recommend (Bishop, 2006) and (Murphy, 2012) for a detailed introduction to probabilistic modelling and (Särkkä, 2013) for a comprehensive summary on sequential Bayesian inference. Further references to relevant literature can be found in the related sections of this chapter.

2.1 LATENT-VARIABLE MODELS

In probabilistic generative modelling, we want to approximate the true probability distribution $p^*(\mathbf{x})$ underlying an unknown but observable system, where observed variables $\mathbf{x} \in \mathbb{R}^{D_\mathbf{x}}$ are assumed to be i.i.d. random samples. To this end, we introduce a probabilistic model $p_\theta(\mathbf{x})$ with the aim of determining its parameters θ such that we obtain

$$p^*(\mathbf{x}) \approx p_\theta(\mathbf{x})$$

for each observed $\mathbf{x}$.

We naturally want $p_\theta(\mathbf{x})$ to be as expressive as possible in order to get a sufficiently accurate model of the true probability distribution. For this reason, $p_\theta(\mathbf{x})$ is often defined as a latent-variable model. The term latent refers to unobserved or unobservable, and latent variables $\mathbf{z} \in \mathbb{R}^{D_\mathbf{z}}$ are random variables that are assumed to capture the underlying characteristics/essence of the data. The marginal distribution over the observed variables is thus given by

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) \, \mathrm{d}\mathbf{z}, \tag{2.1}$$

which is also called the marginal likelihood or the model evidence when defined as a function of θ. The joint distribution factorises as

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z}) \, p(\mathbf{z}),$$

where the conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$ is referred to as the likelihood distribution and $p(\mathbf{z})$ as the prior distribution. The latter represents our prior knowledge and is usually defined in advance; therefore, we do not emphasise its parameters.

The latent-variable model can be interpreted as follows: the essence of data is captured by the latent variables, which we refer to as the *latent representation* of the data; the likelihood distribution captures the information how this essence is translated into an observation. This leads to the generative process:

$$\mathbf{z}' \sim p(\mathbf{z}) \quad \rightarrow \quad \mathbf{x}' \sim p_\theta(\mathbf{x}|\mathbf{z}').$$

Such a generative model can be quite expressive. For example, if the prior distribution is discrete, and $p_\theta(\mathbf{x}|\mathbf{z})$ is a Gaussian distribution, then $p_\theta(\mathbf{x})$ is a mixture of Gaussians. For a continuous $p(\mathbf{z})$, $p_\theta(\mathbf{x})$ can be seen as an infinite/continuous mixture, which is a powerful model for approximating the true probability distribution $p^*(\mathbf{x})$.

The focus of this dissertation is on learning latent representations—with the corresponding generative models—that reflect well the factors of variation and topology of the observed data. The necessary fundamentals, e.g. how we can determine the parameters θ or how probabilistic models are parametrised, are presented in the following sections.

2.1.1 *Difficulties With Classical Learning Approaches*

Learning is referred to as the process of finding the optimal parameters θ^* given a dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ consisting of i.i.d. random observations. Under the i.i.d. assumption, the joint probability of the dataset factorises as a product of individual probabilities:

$$p_\theta(\mathcal{D}) = \prod_i p_\theta(\mathbf{x}_i).$$

In maximum likelihood (ML) learning[1]—which is the prevailing method in statistical inference—we want to find the parameters that maximise the log-likelihood:

$$\operatorname*{argmax}_\theta \log p_\theta(\mathcal{D}) = \operatorname*{argmax}_\theta \sum_i \log p_\theta(\mathbf{x}_i).$$

Considering the latent-variable model defined in Eq. (2.1), we obtain:

$$\sum_i \log p_\theta(\mathbf{x}_i) = \sum_i \log \underbrace{\int p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})\, \mathrm{d}\mathbf{z}}_{\text{usually intractable}}. \tag{2.2}$$

1 We refer to (Murphy, 2012) for a detailed introduction to maximum likelihood learning.

However, the integral in Eq. (2.2) is intractable in all but the simplest cases where a closed-form solution exists. This is due to the intractable summation over all possible outcomes, making it infeasible to directly perform ML learning.

An intuitive approach would be to find a way of inferring the latent variable $\mathbf{z}_i$ that corresponds to $\mathbf{x}_i$—with the idea of applying ML learning as if we had a fully-observed model. Bayes' theorem defines a posterior distribution based on our latent-variable model, which in theory allows us to infer the respective latent variables:

$$p_\theta(\mathbf{z}|\mathbf{x}_i) = \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{\underbrace{\int p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})\, \mathrm{d}\mathbf{z}}_{\text{usually intractable}}}.$$

It does, however, also reveal the connection between the posterior and the marginal likelihood, which acts as a normalisation constant. Hence, an intractable marginal likelihood leads to an intractable posterior, and vice versa. In the next section, we introduce a method that allows us to approximate the posterior $p_\theta(\mathbf{z}|\mathbf{x}_i)$ as well as the marginal likelihood $\int p_\theta(\mathbf{x}_i|\mathbf{z})p(\mathbf{z})\, \mathrm{d}\mathbf{z}$ and thus to learn the parameters of our model.

2.1.2 *Variational Inference*

The main idea behind variational inference is to frame the inference of the posterior distribution as an optimisation problem. To this end, we specify a family $\mathcal{Q}$ of densities over the latent variables, which we refer to as the variational family. Each variational distribution $q_i(\mathbf{z}) \in \mathcal{Q}$ is a candidate for approximating the posterior, and we aim to find the most suitable candidate:

$$p_\theta(\mathbf{z}|\mathbf{x}_i) \approx q_i(\mathbf{z}).$$

In this way, Bayesian inference can be turned into an optimisation problem over variational parameters. As supplementary reading to variational inference, we recommend (Blei et al., 2017) and (Beal, 2003).

The next step is to find a way to measure the similarity between $q_i(\mathbf{z})$ and $p_\theta(\mathbf{z}|\mathbf{x}_i)$ in order to define an objective function for our optimisation problem.

2.1.2.1 *Kullback-Leibler Divergence*

The Kullback-Leibler divergence (KL) is an information-theoretic measure of the proximity between two distributions. Typically, one of the distributions represents an approximation, while the other represents an exact probability distribution:

$$\mathrm{KL}(q(\mathbf{z}) \,\|\, p(\mathbf{z})) = \int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z})}\, \mathrm{d}\mathbf{z} = \mathbb{E}_{q(\mathbf{z})}\left[\log \frac{q(\mathbf{z})}{p(\mathbf{z})}\right] \geq 0,$$

which is zero if and only if $q(\mathbf{z}) = p(\mathbf{z})$.

In variational inference, we aim to minimise the KL between the variational distribution and the posterior, where $q_i^*(\mathbf{z})$ is the best approximation of the posterior within the variational family $\mathcal{Q}$:

$$q_i^*(\mathbf{z}) = \underset{q_i(\mathbf{z}) \in \mathcal{Q}}{\operatorname{argmin}} \, \mathrm{KL}\big(q_i(\mathbf{z}) \| \, p_\theta(\mathbf{z}|\mathbf{x}_i)\big) \,. \tag{2.3}$$

Although this objective function is not computable—because it requires us to compute the posterior and thus the integral in Eq. (2.2)—it is still an excellent choice since it can be minimised indirectly, as we elaborate in the next section.

2.1.2.2 *Evidence Lower Bound*

In the following, we show that minimising the KL between the variational distribution and the posterior is equivalent to maximising a lower bound, which is referred to as the evidence lower bound (ELBO):

$$\begin{aligned}
\log p_\theta(\mathbf{x}_i) &\geq \log p_\theta(\mathbf{x}_i) - \overbrace{\mathrm{KL}\big(q_i(\mathbf{z}) \| \, p_\theta(\mathbf{z}|\mathbf{x}_i)\big)}^{\geq 0} \\
&= \mathbb{E}_{q_i(\mathbf{z})}\left[\log \frac{p_\theta(\mathbf{z}|\mathbf{x}_i)\, p_\theta(\mathbf{x}_i)}{q_i(\mathbf{z})}\right] \\
&= \mathbb{E}_{q_i(\mathbf{z})}\left[\log \frac{p_\theta(\mathbf{x}_i, \mathbf{z})}{q_i(\mathbf{z})}\right] \\
&= \mathbb{E}_{q_i(\mathbf{z})}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})}\right] && (2.4) \\
&= \mathbb{E}_{q_i(\mathbf{z})}\big[\log p_\theta(\mathbf{x}_i|\mathbf{z})\big] - \mathrm{KL}\big(q_i(\mathbf{z}) \| \, p(\mathbf{z})\big) && (2.5) \\
&=: \mathcal{F}_{\mathrm{ELBO}}(q_i(\mathbf{z}), \theta; \, \mathbf{x}_i).
\end{aligned}$$

Since we have access to the variational distribution, the ELBO is tractable, for example, via Monte Carlo integration, which we explain in more detail in Section 2.2.1.

Note that the *gap* between the ELBO and the marginal log-likelihood corresponds to $\mathrm{KL}\big(q_i(\mathbf{z}) \| \, p_\theta(\mathbf{z}|\mathbf{x}_i)\big)$ and becomes zero if and only if $q_i(\mathbf{z}) = p_\theta(\mathbf{z}|\mathbf{x}_i)$. Consequently, the variational distribution that maximises the ELBO concurrently minimises the KL between the variational distribution and the posterior:

$$\underset{q_i(\mathbf{z}) \in \mathcal{Q}}{\operatorname{argmin}} \, \mathrm{KL}\big(q_i(\mathbf{z}) \| \, p_\theta(\mathbf{z}|\mathbf{x}_i)\big) = \underset{q_i(\mathbf{z}) \in \mathcal{Q}}{\operatorname{argmax}} \, \mathcal{F}_{\mathrm{ELBO}}(q_i(\mathbf{z}), \theta; \, \mathbf{x}_i). \tag{2.6}$$

On the other hand—and that makes this formulation so elegant—the ELBO is a lower bound on the marginal log-likelihood. Thus, maximising it can be viewed as an alternative to the (intractable) ML estimate:

$$\underset{\theta}{\operatorname{argmax}} \, \log p_\theta(\mathbf{x}_i) \quad \rightarrow \quad \underset{\theta}{\operatorname{argmax}} \, \mathcal{F}_{\mathrm{ELBO}}(q_i(\mathbf{z}), \theta; \, \mathbf{x}_i) \,. \tag{2.7}$$

However, the parameters θ that maximise the ELBO are highly related to $q_i(\mathbf{z})$, which in turn depends on the previous/initial model parameters. This interdependence leads us directly to the optimisation algorithm that we introduce in Section 2.1.2.3.

ALTERNATIVE DERIVATION OF THE ELBO Note, however, that the ELBO is typically derived using Jensen's inequality. Although this alternative derivation does not provide the insight about the relation between the ELBO and the KL in Eq. (2.6), it is often easier to grasp:

$$\begin{aligned}
\log p_\theta(\mathbf{x}_i) &= \log \int p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})\, \mathrm{d}\mathbf{z} \\
&= \log \int q_i(\mathbf{z})\, \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})}\, \mathrm{d}\mathbf{z} \\
&= \log \mathbb{E}_{q_i(\mathbf{z})}\left[\frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})}\right] \\
&\geq \mathbb{E}_{q_i(\mathbf{z})}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})}\right], \qquad \text{(Jensen's inequality)}
\end{aligned}$$

which is identical to Eq. (2.4).

2.1.2.3 *Variational Expectation-Maximisation Algorithm*

In this section, we discuss how to find the optimal model parameters θ^* starting with an some initial parameters $\theta^{(0)}$. As mentioned in the previous section, the variational distribution that maximises $\mathcal{F}_{\text{ELBO}}(q_i(\mathbf{z}), \theta^{(0)}; \mathbf{x}_i)$ is highly related to $\theta^{(0)}$. Since the initial parameters are most likely not optimal, this therefore also applies to the variational distribution

$$q_i^{(1)}(\mathbf{z}) = \operatorname*{argmax}_{q_i(\mathbf{z}) \in \mathcal{Q}} \mathcal{F}_{\text{ELBO}}(q_i(\mathbf{z}), \theta^{(0)}; \mathbf{x}_i). \tag{2.8}$$

Note that this step has to be carried out for each $\mathbf{x}_i$ in $\mathcal{D}$, resulting in N separate variational distributions $\{q_i^{(1)}(\mathbf{z})\}_{i=1}^N$. Due to the (most likely) suboptimal variational distributions, maximising the ELBO in a next step with respect to θ,

$$\theta^{(1)} = \operatorname*{argmax}_{\theta} \sum_i \mathcal{F}_{\text{ELBO}}\left(q_i^{(1)}(\mathbf{z}), \theta; \mathbf{x}_i\right), \tag{2.9}$$

will not immediately lead to the optimal model parameters θ^*. Consequently, we need an alternating optimisation approach that iteratively finds the optimal model parameters.

The variational expectation-maximization (EM) algorithm (e.g. Neal and Hinton, 1998) alternates between an E-step, which infers the variational distributions given the current parameter ($\theta^{(n)}$), and an M-step that maximises the ELBO with respect to the parameters θ given the variational distributions $\{q_i^{(n+1)}(\mathbf{z})\}_{i=1}^N$ obtained from the

E-step—where the superscript (n) denotes the iteration number of the optimisation process:

$$\textbf{E-step:} \quad q_i^{(n+1)}(\mathbf{z}) = \underset{q_i(\mathbf{z}) \in \mathcal{Q}}{\operatorname{argmax}} \, \mathcal{F}_{\text{ELBO}}\big(q_i(\mathbf{z}), \theta^{(n)}; \mathbf{x}_i\big) \quad \forall \mathbf{x}_i \in \mathcal{D}$$

$$\textbf{M-step:} \quad \theta^{(n+1)} = \underset{\theta}{\operatorname{argmax}} \sum_i \mathcal{F}_{\text{ELBO}}\big(q_i^{(n+1)}(\mathbf{z}), \theta; \mathbf{x}_i\big)$$

However, the variational EM algorithm requires us to infer a separate variational distribution for each $\mathbf{x}_i$ in $\mathcal{D}$. This makes variational inference relatively expensive for large datasets, which are increasingly becoming the norm. Furthermore, we only have access to latent variables $\mathbf{z}_i$ that represent the dataset we use to train our model. This is because we do not learn a function/conditional distribution that allows us to directly infer the latent variable of a new observation.

2.2 AMORTISED VARIATIONAL INFERENCE WITHIN THE FRAMEWORK OF VARIATIONAL AUTOENCODERS

In the previous section, we have introduced latent-variable models and shown how to learn the model parameters through variational inference. However, variational inference is relatively expensive for large datasets, and it does not allow us to directly infer latent variables of new observations as it requires a separate variational distribution for each data point. Amortised variational inference solves theses problems. It is the central method in this thesis and enables us to model an approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$ using a constant number of parameters ϕ with respect to the size of the dataset $\mathcal{D}$.

In the following sections, we introduce amortised variational inference within the framework of variational autoencoders (VAEs), which exploit the expressive power of neural networks and provide a computationally-efficient optimisation approach. We start with the classical VAE in Section 2.2.1 and present two concepts of improvement in Sections 2.2.2 and 2.2.3. As supplementary reading to VAEs, we recommend (Kingma and Welling, 2019).

2.2.1 *Variational Autoencoders*

The term *variational autoencoder* has its origins in autoencoders, where a deterministic encoder–decoder pair is used for learning a compressed representation of high-dimensional data. The VAE (Kingma and Welling, 2014; Rezende et al., 2014) can be viewed as a stochastic counterpart with two independently parametrised probabilistic models as encoder–decoder pair: a recognition/inference model $q_\phi(\mathbf{z}|\mathbf{x})$ and a generative model $p_\theta(\mathbf{x}|\mathbf{z})\, p(\mathbf{z})$. The purpose of the recognition model is to approximate the posterior distribution:

$$p_\theta(\mathbf{z}|\mathbf{x}_i) \approx q_\phi(\mathbf{z}|\mathbf{x}_i).$$

Thus, in contrast to variational inference, the posterior is approximated by a conditional distribution, $q_i(\mathbf{z}) \rightarrow q_\phi(\mathbf{z}|\mathbf{x}_i)$, whose model parameters ϕ do not depend on one specific data point $\mathbf{x}_i$. More precisely, they are used to model the relation between observed and latent variables for the entire dataset. This is typically referred to as amortisation, which gives rise to the term amortised variational inference. In Section 2.2.1.1, we show how such a parametrisation is realised in detail by means of neural networks.

As a consequence of amortising the approximate posterior distribution, the ELBO in Eq. (2.5) is defined as a function of the parameters (θ, ϕ):

$$\mathcal{F}_{\text{ELBO}}(\theta, \phi; \mathbf{x}_i) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\theta(\mathbf{x}_i|\mathbf{z})\right]}_{\text{reconstruction term}} - \underbrace{\text{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}_i) \| \, p(\mathbf{z})\big)}_{\text{regularisation term}}. \tag{2.10}$$

In the context of VAEs, the ELBO is often divided into a reconstruction and regularisation term. The former optimises the model towards a good reconstruction and represents the actual stochastic counterpart to autoencoders. The latter regularises the approximate posterior towards the prior distribution $p(\mathbf{z})$. This ensures that the likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ is optimised to process samples from $p(\mathbf{z})$, which is necessary in order to generate realistic data. In Section 2.2.1.3, we explain this concept in more detail using a popular dataset as an example Before that, however, we present in Section 2.2.1.2, how the ELBO is optimised in detail.

2.2.1.1 *Parametrisation With Neural Networks*

Neural networks (NNs) can be viewed as expressive function approximators. They are differentiable and computationally scalable, which makes them a suitable basis for gradient-based optimisation approaches, as we elaborate in Section 2.2.1.2. The term *neural network* encompasses (among others) the classical deep feed-forward neural networks referred to as multilayer perceptrons (MLPs)—as well as convolutional neural networks (CNNs), which are particularly well suited for processing high-dimensional image data. We assume that the reader is familiar with the basics of neural network architectures and refer to (Goodfellow et al., 2016) for a comprehensive introduction.

In this thesis, we are mainly interested in the use of NNs in probabilistic models, i.e. for modelling probability density (or mass) functions, which is based on the idea of defining the distribution parameters as functions of the conditioning variables. This approach allows the aforementioned amortisation of the approximate posterior. For example, if we use a Gaussian distribution, which is the most common case in VAEs, we obtain:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}\big(\mathbf{z}|\, \boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})\big), \quad \{\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\Sigma}_\phi(\mathbf{x})\} = \text{NN}_\phi(\mathbf{x}),$$

where $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ are implemented by means of an NN characterised by the parameters ϕ. Therefore, in case of a new observation, we just need to pass it through the network, and we obtain an approximate posterior distribution over its associated latent variables. The same concept is used in VAEs for modelling the likelihood distribution. Depending on the dataset, it is usually defined as a Gaussian or as a Bernoulli distribution:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \begin{cases} \mathcal{N}\big(\mathbf{x}|\,\boldsymbol{\mu}_\theta(\mathbf{z}),\,\boldsymbol{\Sigma}_\theta(\mathbf{z})\big), & \{\boldsymbol{\mu}_\theta(\mathbf{z}),\,\boldsymbol{\Sigma}_\theta(\mathbf{z})\} = \mathrm{NN}_\theta(\mathbf{z}), \\ \mathrm{Ber}\big(\mathbf{x}|\,\boldsymbol{p}_\theta(\mathbf{z})\big), & \boldsymbol{p}_\theta(\mathbf{z}) = \mathrm{NN}_\theta(\mathbf{z}), \end{cases}$$

where $\boldsymbol{\mu}_\theta(\mathbf{x})$ and $\boldsymbol{\Sigma}_\theta(\mathbf{x})$ or $\boldsymbol{p}_\theta(\mathbf{z})$ are also implemented by means of an NN. Note, however, that in the context of VAEs it is often advantageous to define the covariance of a Gaussian likelihood distribution as a global parameter, $\boldsymbol{\Sigma}_\theta(\mathbf{x}) \to \boldsymbol{\Sigma}$, as we elaborate in Section 3.2.

2.2.1.2 *Stochastic Gradient-Based Optimisation*

The parametrisation of the approximate posterior and the likelihood distribution with neural networks enables us to compute their derivatives, $\nabla_\phi \log q_\phi(\mathbf{z}|\mathbf{x})$ and $\nabla_\theta \log p_\theta(\mathbf{x}|\mathbf{z})$, by means of the backpropagation algorithm (Rumelhart et al., 1986). In the following, we show that this forms the basis for a gradient-based optimisation of the ELBO, where we start with some random initial parameters (θ_0, ϕ_0) and optimise them, in contrast to the variational EM algorithm, *jointly* via gradient ascent:

$$\begin{aligned} \theta_{n+1} &= \theta_n + \gamma \nabla_\theta \mathcal{F}_{\mathrm{ELBO}}(\theta_n, \phi_n), \\ \phi_{n+1} &= \phi_n + \gamma \nabla_\phi \mathcal{F}_{\mathrm{ELBO}}(\theta_n, \phi_n). \end{aligned}$$

Here, γ is the learning rate, and the ELBO is defined with respect to the entire dataset $\mathcal{D}$ as

$$\mathcal{F}_{\mathrm{ELBO}}(\theta, \phi) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\mathcal{F}_{\mathrm{ELBO}}(\theta, \phi; \mathbf{x})\right], \tag{2.11}$$

where $p_\mathcal{D}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \delta(\mathbf{x} - \mathbf{x}_i)$ is the empirical distribution representing $\mathcal{D}$.

However, computing the derivatives $\nabla_{\theta,\phi}\mathcal{F}_{\mathrm{ELBO}}(\theta, \phi)$ is an expensive operation for large datasets since it scales linearly with the size of the dataset. Furthermore, $\mathcal{F}_{\mathrm{ELBO}}(\theta, \phi; \mathbf{x})$ is not analytically solvable and has to be approximated. In both cases, the solution is to express the integral by a finite sum, which is known as Monte Carlo integration:

$$\mathbb{E}_{p(\mathbf{x})}\left[g(\mathbf{x})\right] \approx \frac{1}{M}\sum_{i=1}^{M} g(\mathbf{x}_i), \quad \mathbf{x}_i \sim p(\mathbf{x}). \tag{2.12}$$

Consequently, in order to reduce the computational cost, we can approximate Eq. (2.11) by

$$\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\left[\mathcal{F}_{\text{ELBO}}(\theta, \phi; \mathbf{x})\right] \approx \frac{1}{M}\sum_{i=1}^{M} \mathcal{F}_{\text{ELBO}}(\theta, \phi; \mathbf{x}_i), \quad \mathbf{x}_i \sim p_{\mathcal{D}}(\mathbf{x}),$$

where $M < N$ corresponds to the size of a randomly drawn so-called minibatch. Maximising an objective function based on minibatches is referred to as stochastic gradient ascent—or as stochastic gradient decent (SGD) if the objective function, which in our case would correspond to the negative ELBO, is minimised (see Murphy, 2012, for further details).

The gradient of the ELBO with respect to the generative model parameters θ is obtained as follows:

$$\begin{aligned}\nabla_\theta \mathcal{F}_{\text{ELBO}}(\theta, \phi; \mathbf{x}_i) &= \nabla_\theta \, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\theta(\mathbf{x}_i|\mathbf{z})\right] \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\nabla_\theta \log p_\theta(\mathbf{x}_i|\mathbf{z})\right] \\ &\approx \nabla_\theta \log p_\theta(\mathbf{x}_i|\mathbf{z}_i), \end{aligned} \tag{2.13}$$

where Eq. (2.13) is a Monte Carlo estimate (cf. Eq. (2.12)) on the basis of a single sample $\mathbf{z}_i \sim q_\phi(\mathbf{z}|\mathbf{x}_i)$. Computing the gradient with respect to the variational parameters ϕ is more complicated, however, because the expectation is taken over the approximate posterior distribution, which is a function of ϕ:

$$\begin{aligned}\nabla_\phi \mathcal{F}_{\text{ELBO}}(\theta, \phi; \mathbf{x}_i) &= \nabla_\phi \, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right] \\ &\neq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\nabla_\phi \log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right]. \end{aligned}$$

The reparametrisation trick (Kingma and Welling, 2014; Rezende et al., 2014) solves this problem by applying a change of variables, where the (continuous) random variable $\mathbf{z}$ is replaced by the deterministic expression $\mathbf{z} = \mathbf{g}_\phi(\boldsymbol{\epsilon}, \mathbf{x}_i)$ with the random noise sample $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$. For example, in case of a Gaussian approximate posterior distribution, we get:

$$\mathbf{z} = \underbrace{\boldsymbol{\mu}_\phi(\mathbf{x}_i) + \mathbf{L}_\phi(\mathbf{x}_i)\,\boldsymbol{\epsilon}}_{\mathbf{g}_\phi(\boldsymbol{\epsilon}, \mathbf{x}_i)}, \quad \boldsymbol{\epsilon} \sim \underbrace{\mathcal{N}(0, \mathbb{1})}_{p(\boldsymbol{\epsilon})}, \tag{2.14}$$

where $\mathbf{L}_\phi(\mathbf{x})$ is a lower triangular matrix resulting from the Cholesky decomposition $\boldsymbol{\Sigma}_\phi(\mathbf{x}) = \mathbf{L}_\phi(\mathbf{x})\mathbf{L}_\phi(\mathbf{x})^T$. Note that the reason for using $\mathbf{L}_\phi(\mathbf{x})$ instead of $\boldsymbol{\Sigma}_\phi(\mathbf{x})$ is a significant increase in computational efficiency (see Kingma and Welling, 2019, for details).

The change of variables allows us to compute the gradient with respect to the variational parameters ϕ as follows:

$$\begin{aligned}\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right] &= \nabla_\phi \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right] \\ &= \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_\phi \log \frac{p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right] \\ &\approx \nabla_\phi \log \frac{p_\theta(\mathbf{x}_i|\mathbf{z}_i)\, p(\mathbf{z}_i)}{q_\phi(\mathbf{z}_i|\mathbf{x}_i)}, \quad (2.15)\end{aligned}$$

where Eq. (2.15) is a single-sample Monte Carlo estimate. This method of estimating the gradients is referred to as stochastic gradient variational Bayes (SGVB) (Kingma and Welling, 2014).

In practice, the computation of gradients is handled by software packages for automatic differentiation, such as (Abadi et al., 2016) or (Paszke et al., 2019).

2.2.1.3 *Example: Binarised MNIST*

In order to gain a better understanding of how VAEs work, we conduct a simple experiment on the example of the binarised version of MNIST (Larochelle and Murray, 2011). It is a popular benchmark dataset consisting of 50,000 training, 10,000 validation, and 10,000 test images of handwritten digits (0 to 9), which are 28×28 pixels in size. For visualisation purposes, we use a two-dimensional latent space. The approximate posterior is defined as an isotropic Gaussian distribution, the prior as standard normal distribution, and the likelihood as a product of independent Bernoulli distributions, each representing one pixel. Both conditional distributions are parametrised by an MLP, and we use a two-dimensional latent space for visualisation purposes.

As mentioned before, the reconstruction term in the ELBO (cf. Eq. (2.10)) optimises the model towards a good reconstruction. This means that the VAE learns (i) to encode data into the latent space, where the aggregated posterior $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[q_\phi(\mathbf{z}|\mathbf{x})\right]$ defines the latent representation of the dataset (Figure 2.1, left); and (ii) to decode samples from the aggragated posterior into realistic observations reflecting the respective encoded data point.

The Kullback-Leibler divergence $\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| \, p(\mathbf{z})\big)$, on the other hand, regularises the approximate posterior towards the standard normal prior, as shown in Figure 2.1 (left), such that

$$\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[q_\phi(\mathbf{z}|\mathbf{x})\right] \approx p(\mathbf{z}). \quad (2.16)$$

This ensures that the likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ is optimised for processing samples from $p(\mathbf{z})$ and thus allows the VAE to generate realistic data (cf. Figure 2.1, right).

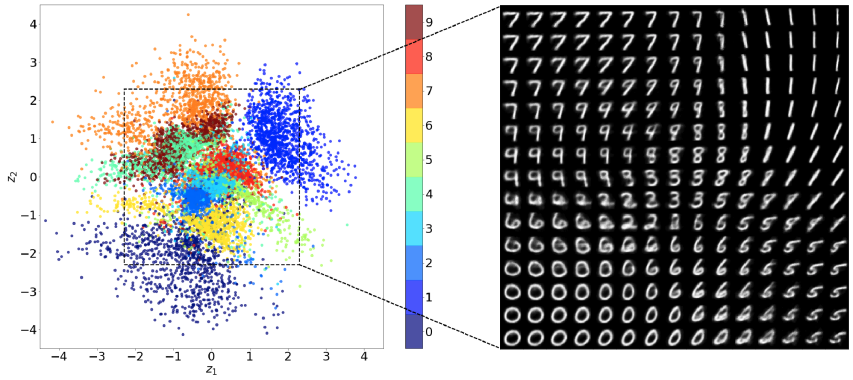


Figure 2.1: VAE trained on the binarised MNIST dataset. The learned two-dimensional latent representation of the dataset is depicted on the left side, and the right side shows generated digits based on the marked area in the latent space. (Klushyn, 2019)

2.2.1.4 *Limitations*

In the following, we identify three main limitations for generative modelling with vanilla VAEs (cf. Kingma and Welling, 2019).

BLURRINESS OF THE GENERATIVE MODEL Generated samples sometimes appear blurry, as can be seen in Figure 2.1 (right). This issue typically occurs if the aggregated posterior does not correspond to the prior distribution (cf. Eq. (2.16))—and thus the generative model is not optimally trained to process samples from the prior. One reason is the limited flexibility of a Gaussian approximate posterior distribution, which we address in Section 2.2.2.

OVER-REGULARISATION OF THE APPROXIMATE POSTERIOR In the vanilla VAE, the aggregated posterior is regularised towards a predefined prior, which is usually the standard normal distribution (cf. Figure 2.1, left). As a consequence, the learned latent representation does not necessarily reflect the topology of the data. To solve this problem, the prior distribution can be learned from data, which we explain in more detail in Section 2.2.3.

POSTERIOR COLLAPSE A local optimum of the ELBO is where $q_\phi(\mathbf{z}|\mathbf{x}_i) = p(\mathbf{z})$ for all $\mathbf{x}_i$ in our dataset $\mathcal{D}$. This corresponds to the optimum of the regularisation term $\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p(\mathbf{z})\big)$ and is referred to as posterior collapse. As a consequence, the VAE is not capable of reconstructing data because the approximate posterior is identical for each data point, and thus the generative model is not optimised.

Especially at the beginning of an optimisation process, where the model has not yet learned to reconstruct data, we may end up in such an initially attractive state. One solution, proposed by Bowman et al.

(2016) and Sønderby et al. (2016), is to slowly increase the weight of the KL term in the ELBO from 0 to 1 during the optimisation process, known as annealing or warm up (WU). In Chapter 3, we introduce a novel approach to this problem based on constrained optimisation.

2.2.2 *Beyond Gaussian Posteriors*

In this section, we discuss how to improve the flexibility of the recognition model in order to achieve $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[q(\mathbf{z}|\mathbf{x})\right] \approx p(\mathbf{z})$ and thus to obtain a tighter lower bound on the marginal log-likelihood. This would allow the generative model to be optimally trained for processing samples from the prior, as detailed in the previous section.

A popular method, which we use throughout this thesis, is the importance-weighted autoencoder (IWAE) (Burda et al., 2016). It treats $q_\phi(\mathbf{z}|\mathbf{x})$ as a proposal distribution for importance sampling:

$$\begin{aligned}\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] &\geq \mathcal{F}_{\text{IW-ELBO}}(\theta, \phi; K) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\mathbb{E}_{\mathbf{z}_{1:K}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{1}{K}\sum_{k=1}^{K}\frac{p_\theta(\mathbf{x}|\mathbf{z}_k)\, p_\theta(\mathbf{z}_k)}{q_\phi(\mathbf{z}_k|\mathbf{x})}\right], \end{aligned} \tag{2.17}$$

where K is the number of importance samples; and the higher K, the tighter is lower bound:

$$\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \geq \mathcal{F}_{\text{IW-ELBO}}(\theta, \phi; K) \geq \mathcal{F}_{\text{ELBO}}(\theta, \phi).$$

This can be explained as follows: by using $q_\phi(\mathbf{z}|\mathbf{x})$ as a proposal distribution for importance sampling, the generative model is optimised on the basis of a more complex *implicit* distribution,

$$q_{\text{IW}}(\mathbf{z}|\mathbf{x}) = \mathbb{E}_{\mathbf{z}_{1:K}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\tilde{q}(\mathbf{z}|\mathbf{x}, \mathbf{z}_{1:K})\right],$$

which approaches the true posterior for $K \to \infty$ (Cremer et al., 2017), i.e. $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[q_{\text{IW}}(\mathbf{z}|\mathbf{x})\right] = p(\mathbf{z})$. As a result, the IWAE learns a more accurate generative model than the original VAE.

In order to obtain the latent representation of our data, i.e. to sample from the implicit distribution, we have to use sequence importance resampling (SIR), as described in Algorithm 1.

Algorithm 1 Sampling from the implicit distribution $q_{\text{IW}}(\mathbf{z}|\mathbf{x})$ via SIR

$K \leftarrow$ number of importance samples
for k in $1 \dots K$ **do**
  $\mathbf{z}_k \sim q_\phi(\mathbf{z}|\mathbf{x})$
  $w_k = \frac{p_\theta(\mathbf{x},\mathbf{z}_k)}{q_\phi(\mathbf{z}_k|\mathbf{x})}$
end for
Each $\tilde{w}_k = \frac{w_k}{\sum_{i=1}^{K} w_i}$
$j \sim \text{Categorical}(\tilde{\mathbf{w}})$
Return $\mathbf{z}_j$

Another popular method that we would like to mention in this context, but will not discuss in detail, is normalising flows (Kingma et al., 2016; Rezende and Mohamed, 2015). To learn a more complex approximate posterior distribution, normalising flows use the change of variables method, where the transformation is framed as an optimisation problem. We refer to (Papamakarios et al., 2021) for a comprehensive introduction.

2.2.3 *Learnable Priors*

In the previous section, we have shown how to increase the flexibility of $q_\phi(\mathbf{z}|\mathbf{x})$ in order to learn an aggregated posterior that corresponds to a predefined prior distribution. An alternative approach to obtain a tighter lower bound on the marginal log-likelihood and a more accurate generative model is to improve the complexity of the prior distribution by learning it from data. Furthermore, this has the consequence that the approximate posterior is not restricted by a predefined prior distribution, which allows the model to learn a latent representation that reflects the topology of our data and has thus a higher informative value, as we discuss in detail in Chapter 3.

Learning the prior distribution from data is referred to as empirical Bayes, which is also known as type-II maximum likelihood because the parameters of the prior distribution are usually learned via ML (Murphy, 2012). However, this often results in an undesirable double use of the data (Berger et al., 2006). In the context of amortised variational inference, by contrast, $p_\theta(\mathbf{x}|\mathbf{z})$ is learned based on samples from $q_\phi(\mathbf{z}|\mathbf{x})$ (reconstruction term in the ELBO), and a learnable prior primarily prevents the approximate posterior from being over-regularised by the KL/regularisation term in the ELBO. Consequently, one can show that the optimal prior is the aggregated posterior distribution (Tomczak and Welling, 2018):

$$p^*(\mathbf{z}) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[q_\phi(\mathbf{z}|\mathbf{x})\right]. \tag{2.18}$$

Therefore, posterior inference with VAEs can be considered to approximate empirical Bayes inference (Wang et al., 2019), and a learnable prior $p_\Theta(\mathbf{z}) \approx p^*(\mathbf{z})$ is just supposed to mimic the aggregated posterior, which is realised by

$$\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| \, p_\Theta(\mathbf{z})\big)\right] \tag{2.19}$$

in the ELBO. We refer to prior distributions learned this way as *empirical Bayes priors*.

Popular examples of learnable priors are introduced in X. Chen et al. (2017) and Tomczak and Welling (2018). The former makes use of normalising flows (cf. Section 2.2.2), the latter (referred to as

VampPrior) is based on a finite mixture of approximate posterior distributions conditioned on pseudo-inputs:

$$p_\Theta(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k),$$

where the pseudo-inputs $\mathbf{u}_k$ are also learned from data (cf. Eq. (2.19)); thus, $\Theta = \{\mathbf{u}_1, \mathbf{u}_2 \ldots, \mathbf{u}_K, \phi\}$. Note that the VampPrior is motivated by the fact that the optimal prior distribution is the aggregated posterior (Eq. (2.18)) and is therefore a suitable approximation in this regard.

In Chapter 3, we revisit this topic and introduce a new concept on how to learn approximate posterior/prior distributions in order to obtain more informative latent representations of data.

2.3 EXTENSION TO SEQUENTIAL DATA

In this section, we introduce probabilistic models for sequences of observations $\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ that occur at discrete time steps within a fixed time interval. Therefore, time is considered as a discrete variable in the following discussion.

The observations $\mathbf{x}_{1:T}$ provide information about a dynamic system that can be observed but whose exact dynamics are usually unknown. Furthermore, the system can be externally influenced by optional control signals $\mathbf{u}_{1:T} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_T)$. In Chapter 4, we study such dynamic systems and introduce state-of-the-art methods for modelling them. The necessary fundamentals are presented in this section.

In order to model and predict an observed system, we need to learn the underlying dynamics. To this end, we introduce latent variables $\mathbf{z}_{1:T} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_T)$ that allow us to model the distribution of sequential data $\mathcal{D} = \{\mathbf{x}^{(i)}_{1:T}, \mathbf{u}^{(i)}_{1:T}\}_{i=1}^{N}$ as the marginal

$$\prod_i p_\theta\left(\mathbf{x}^{(i)}_{1:T} \middle|\, \mathbf{u}^{(i)}_{1:T}\right) = \prod_i \int p_\theta\left(\mathbf{x}^{(i)}_{1:T}, \mathbf{z}_{1:T} \middle|\, \mathbf{u}^{(i)}_{1:T}\right) \mathrm{d}\mathbf{z}_{1:T}\,. \tag{2.20}$$

Similar to the non-sequential case (cf. Section 2.1.2), the model parameters in Eq. (2.20) can be learned by maximising a variational lower bound that we refer to as sequential ELBO:

$$\begin{aligned}\log p_\theta\left(\mathbf{x}^{(i)}_{1:T} \middle|\, \mathbf{u}^{(i)}_{1:T}\right) &\geq \mathcal{F}_{\mathrm{ELBO}}\left(q_i(\mathbf{z}_{1:T}), \theta;\, \mathbf{x}^{(i)}_{1:T}, \mathbf{u}^{(i)}_{1:T}\right) \\ &= \mathbb{E}_{q_i(\mathbf{z}_{1:T})}\left[\log \frac{p_\theta\left(\mathbf{x}^{(i)}_{1:T}, \mathbf{z}_{1:T} \middle|\, \mathbf{u}^{(i)}_{1:T}\right)}{q_i(\mathbf{z}_{1:T})}\right].\end{aligned} \tag{2.21}$$

The sequential ELBO forms the basis for the parameter learning methods discussed in the following sections, where we present several approaches to facilitate the optimisation problem.

First, we introduce in Section 2.3.1 state-space models (SSMs), a specific class of sequential latent-variable models that allows us to efficiently model sequences $\mathbf{x}_{1:T}$ of arbitrary length T.

Building upon this, we discuss in Sections 2.3.2 and 2.3.3 different methods for efficiently learning SSMs based on the sequential ELBO in Eq. (2.21). In Section 2.3.2, we introduce Bayesian filtering and smoothing—and extend in Section 2.3.3 the amortised variational inference framework (Section 2.2) to sequential data. The former allows us to analytically compute the optimal $q_i(\mathbf{z}_{1:T})$, the latter to substitute $q_i(\mathbf{z}_{1:T})$ by an amortised approximate posterior distribution.

2.3.1 *State-Space Models*

The joint distribution $p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T})$ can be factorised by applying the chain rule of probability:

$$\begin{aligned} &p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}) \\ &\quad = \prod_{t=1}^{T} p(\mathbf{x}_t, \mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1}) \\ &\quad = \prod_{t=1}^{T} p(\mathbf{x}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t}, \mathbf{u}_{1:t-1})\, p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1}). \end{aligned}$$

However, modelling the conditional distributions $p(\mathbf{x}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t}, \mathbf{u}_{1:t-1})$ and $p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1})$ becomes computationally more expensive as t increases. This is because with each time step the number of conditioning variables increases and so does the number of parameters necessary to represent the conditional distributions.

To model an arbitrary number of conditional variables with a fixed number of parameters, it is common to impose the Markov assumption, which states that *the future is independent of the past given the present*:

$$\begin{aligned} p(\mathbf{z}_t | \cancel{\mathbf{x}_{1:t-1}}, \mathbf{z}_{t-1}, \cancel{\mathbf{z}_{1:t-2}}, \mathbf{u}_{t-1}, \cancel{\mathbf{u}_{1:t-2}}) &= p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}), \\ p(\mathbf{x}_t | \cancel{\mathbf{x}_{1:t-1}}, \mathbf{z}_1, \cancel{\mathbf{z}_{1:t-1}}, \cancel{\mathbf{u}_{1:t-1}}) &= p(\mathbf{x}_t | \mathbf{z}_t). \end{aligned}$$

Consequently, the latent variable $\mathbf{z}_t$ represents the state of the system that contains all information necessary to predict the future state $\mathbf{z}_{t+1}$ as well as the current observation $\mathbf{x}_t$. This is depicted by the graphical model in Figure 2.2. The corresponding probabilistic model is referred to as state-space model (SSM) and defined by the transition model $p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ and the observation model $p(\mathbf{x}_t | \mathbf{z}_t)$. Both are independent of time; thus, the same parameters are used for each time step. As a result, we arrive at the factorisation:

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}) = \prod_{t=1}^{T} p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})\, p(\mathbf{x}_t | \mathbf{z}_t), \tag{2.22}$$

where $p(\mathbf{z}_1 | \mathbf{z}_0, \mathbf{u}_0) = p(\mathbf{z}_1)$ is referred to as the initial or prior distribution.

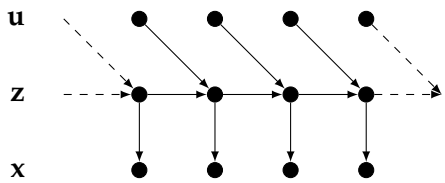


Figure 2.2: Graphical model of state-space models.

2.3.2 *Bayesian Filtering and Smoothing*

In the following, we consider the case where $p_\theta(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ can be computed analytically given the parameters θ. Unlike stationary/non-dynamic systems, where we were not interested in the simple cases where the posterior can be computed analytically, Bayesian filtering and smoothing can be a powerful tool in the context of dynamic systems. It allows us to express the optimal variational distribution by (e.g. Neal and Hinton, 1998)

$$q_i^{(n+1)}(\mathbf{z}_{1:T}) = p_{\theta^{(n)}}\left(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}^{(i)}, \mathbf{u}_{1:T}^{(i)}\right),$$

where the superscript (n) denotes the iteration number of the EM optimisation process introduced in Section 2.1.2.3. As a consequence, we can define the sequential ELBO with respect to the entire dataset $\mathcal{D}$ as

$$\begin{aligned}
&\mathcal{F}_{\text{ELBO}}(\theta, \theta^{(n)}) \\
&\quad = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\Big[\mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T})\right] \\
&\qquad - \underbrace{\mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\left[\log p_{\theta^{(n)}}(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})\right]}_{\text{constant w.r.t. } \theta}\Big],
\end{aligned} \tag{2.23}$$

where $p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ is the empirical distribution representing $\mathcal{D}$. Since the second term in Eq. (2.23) is constant with respect to the optimisation parameters, maximising the ELBO is equivalent to maximising

$$\mathcal{Q}(\theta, \theta^{(n)}) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T})\right].$$

Hence, the variational inference/variational expectation-maximisation problem (cf. Section 2.1.2.3) is simplified to an expectation-maximisation problem with the corresponding EM algorithm:

$$\begin{aligned}
\textbf{E-step:} \quad & \textit{compute} \;\; \mathcal{Q}(\theta, \theta^{(n)}) \\
\textbf{M-step:} \quad & \theta^{(n+1)} = \operatorname*{argmax}_\theta \; \mathcal{Q}(\theta, \theta^{(n)})
\end{aligned}$$

Considering the Markovian structure of SSMs defined in Eq. (2.22), we obtain (Schön et al., 2011):

$$\begin{aligned}\mathcal{Q}(\theta, \theta^{(n)}) = \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Bigg[& \mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})} \left[\log p_\theta(\mathbf{z}_1) \right] \\ & + \sum_{t=2}^{T} \mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})} \left[\log p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \right] \\ & + \sum_{t=1}^{T} \mathbb{E}_{p_{\theta^{(n)}}(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})} \left[\log p_\theta(\mathbf{x}_t | \mathbf{z}_t) \right] \Bigg]. \end{aligned} \tag{2.24}$$

Consequently, we do not need the full posterior distribution, but only the smoothed distributions $p_{\theta^{(n)}}(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$ and the pairwise smoothed distributions $p_{\theta^{(n)}}(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$.

2.3.2.1 *Bayesian Filtering and Smoothing Equations*

The Bayesian filtering equations consist of a *prediction* and *update step* that define the filtered distributions $p(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1})$. The Bayesian smoothing equations extend the filtering equations by a *backward recursion*, which defines the smoothed distributions $p(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$ and pairwise smoothed distributions $p(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$.

PREDICTION STEP In the prediction step, the distribution of the future state—referred to as the predictive distribution—is computed by means of the transition model using the Chapman–Kolmogorov equation:

$$\begin{aligned} & p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-1}) \\ & \quad = \int p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \, p(\mathbf{z}_{t-1} | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-2}) \, \mathrm{d}\mathbf{z}_{t-1}. \end{aligned}$$

UPDATE STEP In the update step, the predictive distribution is updated by the current observation. To this end, we use Bayes' theorem to define the filtered distribution

$$p(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}) = \frac{p(\mathbf{x}_t | \mathbf{z}_t) \, p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-1})}{\int p(\mathbf{x}_t | \mathbf{z}_t) \, p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-1}) \, \mathrm{d}\mathbf{z}_t}.$$

BACKWARD RECURSION The backward recursion is initialised by the filtered distribution at time step T and goes backwards in time using the smoothing equation (Kitagawa, 1987):

$$\begin{aligned} & p(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1}) \\ & \quad = p(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}) \int \frac{p(\mathbf{z}_{t+1} | \mathbf{z}_t, \mathbf{u}_t) \, p(\mathbf{z}_{t+1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})}{p(\mathbf{z}_{t+1} | \mathbf{x}_{1:t}, \mathbf{u}_{1:t})} \, \mathrm{d}\mathbf{z}_{t+1}. \end{aligned} \tag{2.25}$$

Thus, the pairwise smoothed distribution can be computed by *not* marginalising $\mathbf{z}_{t+1}$ in Eq. (2.25) (note that the time steps are shifted by one):

$$p(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1}) = p(\mathbf{z}_{t-1} | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-2}) \frac{p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \, p(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})}{p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-1})}.$$

In Sections 2.3.2.2 and 2.3.2.3, we present analytic solutions of the Bayesian filtering and smoothing equations for linear and nonlinear Gaussian systems. A more extensive introduction to Bayesian filtering and smoothing can be found in (Särkkä, 2013).

2.3.2.2 *Kalman Filter and Smoother*

Linear Gaussian systems are characterised by a linear transition and observation model:

$$\begin{aligned} \mathbf{z}_t &= \mathbf{F}_{t-1} \, \mathbf{z}_{t-1} + \mathbf{B}_{t-1} \, \mathbf{u}_{t-1} + \mathbf{q}_{t-1}, \\ \mathbf{x}_t &= \mathbf{H}_t \, \mathbf{z}_t + \mathbf{r}_t, \end{aligned}$$

where $\mathbf{F}_{t-1}$, $\mathbf{B}_{t-1}$, and $\mathbf{H}_t$ are the transition, control, and observation matrices, respectively. The process noise $\mathbf{q}_{t-1} \sim \mathcal{N}(0, \mathbf{Q}_{t-1})$ and observation noise $\mathbf{r}_t \sim \mathcal{N}(0, \mathbf{R}_t)$ are sampled from zero-mean multivariate normal distributions. Consequently, the corresponding probabilistic versions of the transition and observation model are given as follows:

$$\begin{aligned} p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) &= \mathcal{N}(\mathbf{z}_t | \mathbf{F}_{t-1} \, \mathbf{z}_{t-1} + \mathbf{B}_{t-1} \, \mathbf{u}_{t-1}, \, \mathbf{Q}_{t-1}), \\ p(\mathbf{x}_t | \mathbf{z}_t) &= \mathcal{N}(\mathbf{x}_t | \mathbf{H}_t \, \mathbf{z}_t, \, \mathbf{R}_t). \end{aligned}$$

The Kalman filter algorithm (Kalman et al., 1960) provides the analytic solutions of the *prediction* and *update step* defined in Section 2.3.2.1. Thus, it allows for analytically computing the filtered distribution $p(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1})$ for linear Gaussian systems.

The Kalman smoother algorithm (Rauch et al., 1965) is an extension of the Kalman filter and provides the analytic solutions of the *backward recursion* defined in Section 2.3.2.1, i.e. for analytically computing the smoothed distribution $p(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$ and pairwise smoothed distribution $p(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$.

PREDICTION STEP In linear Gaussian systems, the predictive distribution is obtained through

$$\begin{aligned} \mathbf{m}_t^- &= \mathbf{F}_{t-1} \, \mathbf{m}_{t-1} + \mathbf{B}_{t-1} \, \mathbf{u}_{t-1}, \\ \mathbf{P}_t^- &= \mathbf{F}_{t-1} \, \mathbf{P}_{t-1} \, \mathbf{F}_{t-1}^{\mathrm{T}} + \mathbf{Q}_{t-1}, \end{aligned}$$

where $\mathbf{m}_t^-$ and $\mathbf{P}_t^-$ are the mean and covariance of

$$p(\mathbf{z}_t | \mathbf{x}_{1:t-1}, \mathbf{u}_{1:t-1}) = \mathcal{N}(\mathbf{z}_t | \mathbf{m}_t^-, \, \mathbf{P}_t^-).$$

UPDATE STEP In order to update the predictive distribution with the current observation, we use the linear observation model:

$$\begin{aligned}
\mathbf{v}_t &= \mathbf{x}_t - \mathbf{H}_t\,\mathbf{m}_t^-,\\
\mathbf{S}_t &= \mathbf{H}_t\,\mathbf{P}_t^-\,\mathbf{H}_t^{\mathrm{T}} + \mathbf{R}_t,\\
\mathbf{K}_t &= \mathbf{P}_t^-\,\mathbf{H}_t^{\mathrm{T}}\,\mathbf{S}_t^{-1},\\
\mathbf{m}_t &= \mathbf{m}_t^- + \mathbf{K}_t\,\mathbf{v}_t,\\
\mathbf{P}_t &= \mathbf{P}_t^- - \mathbf{K}_t\,\mathbf{S}_t\,\mathbf{K}_t^{\mathrm{T}},
\end{aligned}$$

which defines the filtered distribution

$$p(\mathbf{z}_t|\,\mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}) = \mathcal{N}(\mathbf{z}_t|\,\mathbf{m}_t,\,\mathbf{P}_t).$$

BACKWARD RECURSION The backward recursion starts with the filtered distribution at time step T—determining $\mathbf{m}_T^{\mathrm{s}} = \mathbf{m}_T$ and $\mathbf{P}_T^{\mathrm{s}} = \mathbf{P}_T$—and goes backwards in time using the recursive equations

$$\mathbf{G}_t = \mathbf{P}_t\,\mathbf{F}_t^{\mathrm{T}}\left[\mathbf{P}_{t+1}^-\right]^{-1}, \tag{2.26}$$

$$\mathbf{m}_t^{\mathrm{s}} = \mathbf{m}_t + \mathbf{G}_t\left[\mathbf{m}_{t+1}^{\mathrm{s}} - \mathbf{m}_{t+1}^-\right], \tag{2.27}$$

$$\mathbf{P}_t^{\mathrm{s}} = \mathbf{P}_t + \mathbf{G}_t\left[\mathbf{P}_{t+1}^{\mathrm{s}} - \mathbf{P}_{t+1}^-\right]\mathbf{G}_t^{\mathrm{T}}. \tag{2.28}$$

Therefore, at time step $t < T$, the backward recursion enables the consideration of all observations up to time step T. As a result, we obtain the smoothed distribution

$$p(\mathbf{z}_t|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T-1}) = \mathcal{N}(\mathbf{z}_t|\,\mathbf{m}_t^{\mathrm{s}},\,\mathbf{P}_t^{\mathrm{s}}).$$

Furthermore, Eqs. (2.26–2.28) allow us to define the joint mean and covariance,

$$\begin{aligned}
\tilde{\mathbf{m}}^{\mathrm{s}} &= \begin{pmatrix} \mathbf{m}_t^{\mathrm{s}} \\ \mathbf{m}_{t-1}^{\mathrm{s}} \end{pmatrix},\\
\tilde{\mathbf{P}}^{\mathrm{s}} &= \begin{pmatrix} \mathbf{P}_t^{\mathrm{s}} & \mathbf{P}_t^{\mathrm{s}}\,\mathbf{G}_{t-1}^{\mathrm{T}} \\ \mathbf{G}_{t-1}\,\mathbf{P}_t^{\mathrm{s}} & \mathbf{P}_{t-1}^{\mathrm{s}} \end{pmatrix},
\end{aligned}$$

of the pairwise smoothed distribution

$$p(\mathbf{z}_t, \mathbf{z}_{t-1}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T-1}) = \mathcal{N}(\mathbf{z}_t, \mathbf{z}_{t-1}|\,\tilde{\mathbf{m}}^{\mathrm{s}},\,\tilde{\mathbf{P}}^{\mathrm{s}}).$$

The Kalman smoother algorithm therefore enables us to compute $\mathcal{Q}(\theta, \theta^{(n)})$ in Eq. (2.24) for linear Gaussian systems.

2.3.2.3 *Extended Kalman Filter and Smoother*

Under the assumption that process and observation noises are additive, nonlinear Gaussian systems can be characterised by

$$\begin{aligned}
\mathbf{z}_t &= \mathbf{f}(\mathbf{z}_{t-1}, \mathbf{u}_{t-1}) + \mathbf{q}_{t-1},\\
\mathbf{x}_t &= \mathbf{h}(\mathbf{z}_t) + \mathbf{r}_t,
\end{aligned}$$

where $\mathbf{f}(\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ and $\mathbf{h}(\mathbf{z}_t)$ are nonlinear functions referred to as the transition and the observation model function. In order to enable an analytic computation, $\mathbf{f}(\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ and $\mathbf{h}(\mathbf{z}_t)$ are locally linearised in the extended Kalman filter/smoother algorithm using a first-order Taylor series expansion, as we describe below.

PREDICTION STEP The prediction step follows the same concept as the original Kalman filter/smoother algorithm, with the difference that the local Jacobian of the transition model function is used for computing the covariance of the predictive distribution:

$$\begin{aligned}\mathbf{m}_t^- &= \mathbf{f}(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}),\\ \mathbf{P}_t^- &= \mathbf{F}_\mathbf{z}(\mathbf{m}_{t-1}, \mathbf{u}_{t-1})\,\mathbf{P}_{t-1}\,\mathbf{F}_\mathbf{z}^\mathrm{T}(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}) + \mathbf{Q}_{t-1},\end{aligned}$$

where

$$\mathbf{F}_\mathbf{z}(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}) = \left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{z}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}}. \tag{2.29}$$

UPDATE STEP According to the same principle as the prediction step, the observation model function is locally linearised around $\mathbf{m}_t^-$ in order to compute the mean and covariance of the filtered distribution:

$$\begin{aligned}\mathbf{v}_t &= \mathbf{x}_t - \mathbf{h}(\mathbf{m}_t^-),\\ \mathbf{S}_t &= \mathbf{H}_\mathbf{z}(\mathbf{m}_t^-)\,\mathbf{P}_t^-\,\mathbf{H}_\mathbf{z}^\mathrm{T}(\mathbf{m}_t^-) + \mathbf{R}_t,\\ \mathbf{K}_t &= \mathbf{P}_t^-\,\mathbf{H}_\mathbf{z}^\mathrm{T}(\mathbf{m}_t^-)\,\mathbf{S}_t^{-1},\\ \mathbf{m}_t &= \mathbf{m}_t^- + \mathbf{K}_t\,\mathbf{v}_t,\\ \mathbf{P}_t &= \mathbf{P}_t^- - \mathbf{K}_t\,\mathbf{S}_t\,\mathbf{K}_t^\mathrm{T},\end{aligned}$$

where

$$\mathbf{H}_\mathbf{z}(\mathbf{m}_t^-) = \left.\frac{\partial \mathbf{h}(\mathbf{z})}{\partial \mathbf{z}}\right|_{\mathbf{m}_t^-}$$

is the local Jacobian of the observation model function.

BACKWARD RECURSION The backward recursion is identical to the Kalman smoother except that the local Jacobian defined in Eq. (2.29) is used as transition matrix:

$$\mathbf{G}_t = \mathbf{P}_t\,\mathbf{F}_\mathbf{z}^\mathrm{T}(\mathbf{m}_t, \mathbf{u}_t)\left[\mathbf{P}_{t+1}^-\right]^{-1}, \tag{2.30}$$

$$\mathbf{m}_t^\mathrm{s} = \mathbf{m}_t + \mathbf{G}_t\left[\mathbf{m}_{t+1}^\mathrm{s} - \mathbf{m}_{t+1}^-\right], \tag{2.31}$$

$$\mathbf{P}_t^\mathrm{s} = \mathbf{P}_t + \mathbf{G}_t\left[\mathbf{P}_{t+1}^\mathrm{s} - \mathbf{P}_{t+1}^-\right]\mathbf{G}_t^\mathrm{T}. \tag{2.32}$$

The Eqs. (2.30–2.32) define the smoothed and pairwise smoothed distribution, $p(\mathbf{z}_t \,|\, \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$ and $p(\mathbf{z}_t, \mathbf{z}_{t-1} \,|\, \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1})$, respectively. Hence, the extended Kalman smoother algorithm allows the computation of $\mathcal{Q}(\theta, \theta^{(n)})$ in Eq. (2.24) for nonlinear Gaussian systems.

However, the necessity of locally-linear approximations is a disadvantage that causes the extended Kalman filter/smoother to perform poorly in problems with significant nonlinearities, as it is the case for image data (Särkkä, 2013). We address this issue in the following section.

2.3.3 *Amortised Variational Inference*

Amortised variational inference enables modelling probability distributions of highly nonlinear data, as demonstrated in Section 2.2.1. This is achieved by introducing an amortised approximate posterior distribution

$$q_i(\mathbf{z}_{1:T}) \quad \rightarrow \quad q_\phi\left(\mathbf{z}_{1:T} \middle| \mathbf{x}_{1:T}^{(i)}, \mathbf{u}_{1:T}^{(i)}\right),$$

also referred to as recognition model. Consequently, we can define the sequential ELBO as

$$\mathcal{F}_{\text{ELBO}}(\theta, \phi) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log \frac{p_\theta(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T})}{q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \right],$$

where $p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ is the empirical distribution representing $\mathcal{D}$. Instead of an EM-based optimisation, amortised variational inference allows us to maximise the ELBO by optimising (θ, ϕ) jointly, as discussed in Section 2.2.1.2.

By taking into account the Markovian structure of SSMs in Eq. (2.22), we obtain (Krishnan et al., 2015):

$$\begin{aligned}
\mathcal{F}_{\text{ELBO}}(\theta, \phi) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Bigg[& \mathbb{E}_{q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log \frac{p(\mathbf{z}_1)}{q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \right] \\
& + \sum_{t=2}^{T} \mathbb{E}_{q_\phi(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log \frac{p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})}{q_\phi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T})} \right] \\
& + \sum_{t=1}^{T} \mathbb{E}_{q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log p_\theta(\mathbf{x}_t | \mathbf{z}_t) \right] \Bigg].
\end{aligned} \tag{2.33}$$

Similar to the framework of variational autoencoders, the sequential ELBO in Eq. (2.33) can be divided into a reconstruction term and a KL term. The reconstruction term is defined by

$$\sum_{t=1}^{T} \mathbb{E}_{q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log p_\theta(\mathbf{x}_t | \mathbf{z}_t) \right],$$

whereas the KL term consists of a *prior* KL,

$$\begin{aligned}
& \mathbb{E}_{q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log \frac{p(\mathbf{z}_1)}{q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \right] \\
& \quad = -\mathrm{KL}\left(q_\phi(\mathbf{z}_1 | \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}) \| \, p(\mathbf{z}_1) \right),
\end{aligned}$$

for the initial time step and *transition* KLs for time steps $T \leq t \leq 2$:

$$\begin{aligned}&\mathbb{E}_{q_\phi(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\log \frac{p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})}{q_\phi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T})} \right] \\ &= -\mathbb{E}_{q_\phi(\mathbf{z}_{t-1} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Big[\mathrm{KL}\big(q_\phi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}) \| \, p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})\big) \Big].\end{aligned}$$

The distribution parameters of $p_\theta(\mathbf{x}_t | \mathbf{z}_t)$, $p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$, and $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ are functions of the conditioning variables. If we use, for example, Gaussian distributions, which is the most common case, we obtain:

$$\begin{aligned}p_\theta(\mathbf{x}_t | \mathbf{z}_t) &= \mathcal{N}\big(\mathbf{x}_t | \boldsymbol{\mu}_\theta(\mathbf{z}_t), \boldsymbol{\Sigma}_\theta(\mathbf{z}_t)\big), \\ p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) &= \mathcal{N}\big(\mathbf{z}_t | \boldsymbol{\mu}_\theta(\mathbf{z}_{t-1}, \mathbf{u}_{t-1}), \boldsymbol{\Sigma}_\theta(\mathbf{z}_{t-1}, \mathbf{u}_{t-1})\big), \\ q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) &= \mathcal{N}\big(\mathbf{z}_t | \boldsymbol{\mu}_\phi(\mathbf{x}_{1:T}, \mathbf{u}_{1:T}), \boldsymbol{\Sigma}_\phi(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})\big).\end{aligned}$$

The functions representing the distribution parameters of $p_\theta(\mathbf{x}_t | \mathbf{z}_t)$ and $p_\theta(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ are usually modelled by MLPs or CNNs, as described in Section 2.2.1.1. In order to process sequential data, as is the case with $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$, recurrent neural networks (RNNs) are typically used. Thus, the RNN is expected to replace the Bayesian filtering and smoothing equations discussed in Section 2.3.2.1. Common RNN architectures are long short-term memorys (LSTMs) (Hochreiter and Schmidhuber, 1997) or gated recurrent units (GRUs) (Cho et al., 2014).[2] This class of SSMs is therefore also referred to as DSSMs.

A key challenge in the context of DSSMs is the implementation of the conditional approximate posterior $q_\phi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T})$ within the neural setting. In Chapter 4, we present established state-of-the-art methods that address this issue and show how combining amortised variational inference with classic Bayesian filtering/smoothing can be a powerful approach in this regard.

2 We refer to (Goodfellow et al., 2016) for a detailed introduction to neural network architectures.

3

LEARNING INFORMATIVE LATENT REPRESENTATIONS WITH VARIATIONAL AUTOENCODERS

The methods and experimental results discussed in this chapter have been previously published in (Klushyn et al., 2019b). Sections 3.1, 3.3, 3.4, and 3.5 are based on revised text from this publication.

This chapter addresses the question of how to learn latent representations with variational autoencoders (VAEs) that reflect the factors of variation and topology of observed data

VAEs are a class of probabilistic latent-variable models for unsupervised learning (Kingma and Welling, 2014; Rezende et al., 2014). They model the unknown distribution of observed data by learning a mapping from a (predefined) prior distribution. The resulting generative model is a useful tool for a wide range of applications. For example, the generation of photo-realistic images (Child, 2020; Razavi et al., 2019) and coherent text (Bowman et al., 2016; Hu et al., 2017)—or the automatic design of chemicals and molecules (Gómez-Bombarelli et al., 2018; Riesselman et al., 2018).

As detailed in Section 2.2.1, VAEs learn the generative model by simultaneously inferring an approximate posterior distribution of latent variables, which is expected to capture semantically meaningful features of the observed data. The informative value of the inferred latent representation has a significant impact on the quality of the learned generative model (X. Chen et al., 2017; Kingma et al., 2016; Rezende and Mohamed, 2015).

The prior distribution over the latent variables is typically defined as a standard normal distribution. However, in the context of VAEs, this often leads to an over-regularised posterior distribution resulting in latent representations that do not reflect the topology of the observed data, as we verify in Section 3.5

To solve this problem, we draw on concepts from previous work, which can be grouped into two categories: optimisation algorithms and empirical Bayes methods. Bowman et al. (2016), Higgins et al. (2017), and Rezende and Viola (2018) have developed optimisation algorithms that facilitate finding minima in the objective function of VAEs—known as the evidence lower bound (ELBO)—which correspond to informative latent representations. X. Chen et al. (2017) and Tomczak and Welling (2018) have proposed empirical Bayes methods for learning complex prior distributions in order to avoid an over-regularisation of the approximate posterior. In the following, we combine these two approaches and show how this enables VAEs

to learn latent representations reflecting the factors of variation and topology of the observed data:

- In Section 3.1, we formulate the ELBO as the Lagrangian of a constrained optimisation problem. To this end, we impose an inequality constraint on the reconstruction error and use the KL between the approximate posterior and the standard normal prior as the optimisation objective. This allows controlling the reconstruction quality achieved by the VAE.
- Section 3.2 provides a detailed analysis of the introduced constrained optimisation-based formulation: we prove that the optimisation problem is concave with respect to the Lagrange multiplier and derive the connection to the ELBO for Bernoulli and Gaussian likelihood distributions.
- In Section 3.3, we introduce a hierarchical empirical Bayes prior and the associated constrained optimisation algorithm. This extends the VAE to a two-level stochastic model, such that the first layer learns the latent representation and the second layer models the empirical Bayes prior. Therefore, the model is capable of learning complex latent representations reflecting the topology of the observed data.
- In Section 3.4, we propose a graph-based interpolation method, which allows us to validate the quality of the learned latent representation on the basis of the manifold hypothesis.
- In Section 3.5, we evaluate our approach. This includes experiments on a moving pendulum, on real-world human motion data, on standard benchmark datasets, and on high-dimensional image data.

3.1 VARIATIONAL AUTOENCODERS AS A CONSTRAINED OPTIMISATION PROBLEM

VAEs model an unknown distribution of i.i.d. data $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ by means of typically lower-dimensional latent variables:

$$\prod_i p_\theta(\mathbf{x}_i) = \prod_i \int p_\theta(\mathbf{x}_i|\mathbf{z})\, p(\mathbf{z})\, \mathrm{d}\mathbf{z}. \tag{3.1}$$

Supplementary to the general introduction to VAEs in Section 2.2.1, we emphasise some important details that are relevant for understanding our proposed method.

The model parameters in Eq. (3.1) are learned through amortised variational inference. This requires introducing an approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x})$, which we use to learn an informative latent representation of the observed data $\mathcal{D}$, allowing the VAE to

generalise to unseen data. For example, $\{\mathbb{E}_{q_\theta(\mathbf{z}|\mathbf{x}_i)}[\mathbf{z}]\}_{i=1}^N$ cluster with respect to some discrete features or important factors of variation in $\mathcal{D}$.

This allows us to learn the model parameters (θ, ϕ) by maximising the ELBO:

$$\begin{aligned}\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\big[\log p_\theta(\mathbf{x})\big] &\geq \mathcal{F}_{\text{ELBO}}(\theta, \phi) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\Big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x}|\mathbf{z})\big] - \mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| \, p(\mathbf{z})\big)\Big], \end{aligned} \tag{3.2}$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ are typically assumed to be diagonal Gaussians, and $p_\mathcal{D}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^N \delta(\mathbf{x} - \mathbf{x}_i)$ represents the empirical distribution of $\mathcal{D}$.

Amortised variational inference was originally formulated as an EM optimisation problem, i.e. as a double-loop algorithm, which is described in Section 2.1.2.3:

$$\min_{\theta,\phi} -\mathcal{F}_{\text{ELBO}}(\theta, \phi) \;\widehat{=}\; \overbrace{\min_\theta}^{\text{M-step}} \underbrace{\min_\phi}_{\text{E-step}} -\mathcal{F}_{\text{ELBO}}(\theta, \phi).$$

In the context of VAEs and neural inference models in general, though, it is common practice to optimise the parameters (θ, ϕ) jointly.

However, it has been shown that local minima with high ELBO values do not necessarily result in informative latent representations (Alemi et al., 2018; Higgins et al., 2017). In order to address this problem, several approaches have been developed, which typically introduce weighting schedules for either the negative expected log-likelihood or the KL in the ELBO (Bowman et al., 2016; Sønderby et al., 2016). This is because a different ratio targets different regions in the rate–distortion plane, either favouring better reconstruction or compression (Alemi et al., 2018).

Rezende and Viola (2018) reformulate the ELBO as the Lagrangian of a constrained optimisation problem[1]. They specify the Kullback-Leibler divergence $\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| \, p(\mathbf{z})\big)$ as optimisation objective and impose the inequality constraint $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\big] \leq \kappa^2$. Here, $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ is defined as the reconstruction-error-related term in $-\log p_\theta(\mathbf{x}|\mathbf{z})$. As a consequence, this formulation allows for a better control of the quality of the reconstructed data. In the resulting Lagrangian

$$\begin{aligned}&\mathcal{L}(\theta, \phi; \lambda) \\ &\quad = \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\Big[\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| \, p(\mathbf{z})\big) + \lambda\Big(\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\big] - \kappa^2\Big)\Big],\end{aligned} \tag{3.3}$$

the Lagrange multiplier λ can be viewed as a weighting term for $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[-\log p_\theta(\mathbf{x}|\mathbf{z})\big]$.

1 We refer to (Boyd et al., 2004) for a detailed introduction to constrained optimisation with Lagrange multipliers.

This approach leads to a similar optimisation objective, as defined by Higgins et al. (2017), with $\beta = 1/\lambda$. Rezende and Viola (2018) propose a descent-ascent algorithm (GECO) for finding the saddle point of the Lagrangian. The parameters (θ, ϕ) are optimised through gradient descent, and λ is updated as

$$\lambda_t = \lambda_{t-1} \cdot \exp\left(\nu \cdot \left(\hat{\mathcal{C}}_t - \kappa^2\right)\right), \tag{3.4}$$

corresponding to a quasi-gradient ascent due to $\Delta\lambda_t \cdot \partial_\lambda \mathcal{L} \geq 0$. Here, ν is the update's learning rate. In the context of stochastic batch gradient training, $\hat{\mathcal{C}}_t \approx \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\right]$ is estimated as the running average $\hat{\mathcal{C}}_t = (1 - \alpha) \cdot \hat{\mathcal{C}}_{\text{ba}} + \alpha \cdot \hat{\mathcal{C}}_{t-1}$, where $\hat{\mathcal{C}}_{\text{ba}}$ is the batch average $\mathbb{E}_{p_\mathcal{D}(\mathbf{x_{ba}})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\right]$.

However, $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ is not optimised in the classical constrained optimisation setting, and thus the parameters θ are not optimised either. To address this issue, we interpret the optimisation problem as a sequence of constrained optimisation problems. To the best of our understanding—this is not explicitly stated in Rezende and Viola (2018)—the GECO algorithm solves

$$\overbrace{\min_\theta}^{\text{M-step}} \underbrace{\max_\lambda \min_\phi}_{\text{E-step}} \mathcal{L}(\theta, \phi; \lambda) \quad \text{s.t.} \quad \lambda \geq 0, \tag{3.5}$$

where the constrained optimisation problem $\max_\lambda \min_\phi \mathcal{L}(\theta, \phi; \lambda)$ and the M-step $\min_\theta \mathcal{L}(\theta, \phi; \lambda)$ are alternately updated. However, in general, the Lagrangian can only be guaranteed to optimise the ELBO if $\lambda = 1$, or in case of $0 \leq \lambda < 1$, a scaled lower bound on the ELBO. In Section 3.2, we show that the optimisation problem in Eq. (3.5) is concave with respect to the newly introduced Lagrange multiplier λ and provide a detailed discussion about the connection to the ELBO.

3.2 ANALYSIS OF THE CONSTRAINED OPTIMISATION-BASED FORMULATION

In Section 3.1, we have reformulated the ELBO as the Lagrangian of a constrained optimisation problem. This can be viewed as an alternative approach for optimising VAEs. Yet it might be helpful to discuss two important points in more detail.

We have argued that the original EM optimisation problem can be formulated as a sequence of constrained optimisation problems (Eq. (3.5)), where the constrained optimisation corresponds to the E-step. In this section, we prove that the optimisation problem in Eq. (3.5) is concave with respect to the newly introduced Lagrange multiplier λ—and show explicitly the connection to the ELBO on the example of Bernoulli and Gaussian likelihood distributions.

3.2.1 *Concavity of the Optimisation Problem*

By formulating the ELBO as the Lagrangian of a constrained optimisation problem, we introduce a Lagrange multiplier λ. This adds an additional optimisation cycle with respect to λ to the classical EM approach. In the following, we analyse if the optimisation problem is concave with respect to the Lagrange multiplier and explain the idea behind the introduced update scheme for λ in Eq. (3.4). As a first step, we derive the optimal $q_i^*(\mathbf{z})$ in distribution space:

$$\mathcal{L}(\theta, q_i(\mathbf{z}); \lambda) = \mathrm{KL}\big(q_i(\mathbf{z}) \| \, p(\mathbf{z})\big) + \lambda \big(\mathbb{E}_{q_i(\mathbf{z})}\big[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big] - \kappa^2\big).$$

For simplicity, the normalisation constraints $\int q_i(\mathbf{z}) \, \mathrm{d}\mathbf{z} = 1$ are not explicitly stated. By setting the derivative $\partial_{q_i} \mathcal{L}(\theta, q_i(\mathbf{z}); \lambda)$ to zero:

$$\partial_{q_i} \left(\int \left(\log \frac{q_i(\mathbf{z})}{p(\mathbf{z})} + \lambda \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z}) - \lambda \, \kappa^2 \right) q_i(\mathbf{z}) \, \mathrm{d}\mathbf{z} \right) \stackrel{!}{=} 0,$$

we obtain

$$q_i^*(\mathbf{z}) = \frac{1}{Z(\lambda)} \, p(\mathbf{z}) \, \exp\big(-\lambda \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big),$$

where $Z(\lambda) = \int p(\mathbf{z}) \exp\big(-\lambda \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big) \, \mathrm{d}\mathbf{z}$ is the normalising factor. The constrained optimisation problem can now be reduced to

$$\max_{\lambda} \min_{q_i(\mathbf{z})} \mathcal{L}(\theta, q_i(\mathbf{z}); \lambda) = \max_{\lambda} \mathcal{L}(\theta, q_i^*(\mathbf{z}); \lambda),$$

with

$$\begin{aligned} \mathcal{L}(\theta, q_i^*(\mathbf{z}); \lambda) &= \mathbb{E}_{q_i^*(\mathbf{z})} \left[\log \frac{p(\mathbf{z}) \exp\big(-\lambda \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big)}{Z(\lambda) \, p(\mathbf{z})} \right] \\ &\quad + \lambda \left(\mathbb{E}_{q_i^*(\mathbf{z})}\big[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big] - \kappa^2 \right) \\ &= -\log Z(\lambda) - \lambda \, \kappa^2. \end{aligned}$$

As a next step, we compute

$$\begin{aligned} \partial_\lambda \mathcal{L}(\theta, q_i^*(\mathbf{z}); \lambda) &= -\frac{1}{Z(\lambda)} \, \partial_\lambda Z(\lambda) - \kappa^2 \\ &= \frac{1}{Z(\lambda)} \int p(\mathbf{z}) \exp\big(-\lambda \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big) \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z}) \, \mathrm{d}\mathbf{z} - \kappa^2 \\ &= \frac{Z(\lambda)}{Z(\lambda)} \int q_i^*(\mathbf{z}) \, \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z}) \, \mathrm{d}\mathbf{z} - \kappa^2 \\ &= \mathbb{E}_{q_i^*(\mathbf{z})}\big[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big] - \kappa^2 \end{aligned} \tag{3.6}$$

and

$$\begin{aligned} \partial_\lambda^2 \mathcal{L}(\theta, q_i^*(\mathbf{z}); \lambda) &= \partial_\lambda \left(-\frac{1}{Z(\lambda)} \, \partial_\lambda Z(\lambda) - \kappa^2 \right) \\ &= -\frac{Z(\lambda) \, \partial_\lambda^2 Z(\lambda) - \partial_\lambda Z(\lambda) \, \partial_\lambda Z(\lambda)}{Z(\lambda)^2} \\ &= \underbrace{-\mathbb{E}_{q_i^*(\mathbf{z})}\big[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})^2\big] + \mathbb{E}_{q_i^*(\mathbf{z})}\big[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\big]^2}_{\leq 0}. \end{aligned} \tag{3.7}$$

Eq. (3.7) can be interpreted as a negative variance term; thus, it is less than or equal to zero. As a result, $\mathcal{L}(\theta, q_i^*; \lambda)$ is concave down with respect to λ and has a global maximum at $\mathbb{E}_{q_i^*(\mathbf{z})}\left[\mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z})\right] = \kappa^2$. In practice, we do not optimise over distribution space, and $q_\phi(\mathbf{z}|\,\mathbf{x})$ is often a highly non-convex function. However, we show empirically in Section 3.5.3 that this does not introduce limitations to our method.

Furthermore, Eq. (3.6) allows for a better interpretation of the update rule $\lambda_t = \lambda_{t-1} \exp\left(\propto (\hat{\mathcal{C}}_t - \kappa^2)\right)$ (see definition in Eq. (3.4)). The comparison of the two equations shows that the update of λ is aligned with the gradient. However, due to the use of the exponential function, the step size varies in an elegant way: the larger the distance to the optimum the larger the update step, and vice versa. This leads to a faster convergence of the optimisation procedure.

3.2.2 *Connection to the Evidence Lower Bound*

In Section 3.1, we argued $\min_\theta \max_\lambda \min_\phi \mathcal{L}(\theta, \phi; \lambda)$ optimises the ELBO if and only if $\lambda = 1$. In the following, we draw a connection between the Lagrangian and the negative ELBO—and clarify the role of the Lagrange multiplier on the example of Bernoulli and Gaussian likelihood distributions.

Note that the negative ELBO and the Lagrangian are not necessarily identical, but optimising them:

$$\min_\theta \min_\phi \; -\mathcal{F}_{\text{ELBO}}(\theta, \phi),$$
$$\min_\theta \max_\lambda \min_\phi \; \mathcal{L}(\theta, \phi; \lambda) \quad \text{s.t.} \quad \lambda \geq 0,$$

can lead to the same parameters (θ, ϕ). In order to figure out under which conditions this is the case, we look at the gradients:

$$\exists \lambda \geq 0 \;:\; -\nabla_{\theta,\phi}\, \mathcal{F}_{\text{ELBO}}(\theta, \phi) = \nabla_{\theta,\phi}\, \mathcal{L}(\theta, \phi; \lambda).$$

Since the KL term is identical in $\mathcal{F}_{\text{ELBO}}(\theta, \phi)$ and $\mathcal{L}(\theta, \phi; \lambda)$ (cf. Eq. (3.2) and Eq. (3.3)), we focus on the difference between $-\log p_\theta(\mathbf{x}|\mathbf{z})$ and $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$. The latter is defined as the reconstruction-error-related term in the negative log-likelihood. To do this, we analyse $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ on the example of two frequently used likelihood distributions in the context of VAEs: the Bernoulli and the Gaussian distribution.

3.2.2.1 *Bernoulli Likelihood Distribution*

In case of a Bernoulli likelihood distribution, the probability of success, $\mathbf{p}$, is a function of the latent variable:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{d=1}^{D} \big(\underbrace{f_\theta^{(d)}(\mathbf{z})}_{p_d} \big)^{x_d} \left(1 - f_\theta^{(d)}(\mathbf{z})\right)^{1-x_d},$$

where D is the dimensionality of $\mathbf{x}$. We define the reconstruction-error-related term $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ as the binary cross entropy, which is a natural choice:

$$\mathcal{C}_\theta(\mathbf{x}, \mathbf{z}) = -\sum_{d=1}^{D} x_d \log f_\theta^{(d)}(\mathbf{z}) + (1 - x_d) \log \left(1 - f_\theta^{(d)}(\mathbf{z})\right).$$

As a consequence, $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ is identical to $-\log p_\theta(\mathbf{x}|\mathbf{z})$. It follows that the negative ELBO and the Lagrangian are identical for $\lambda = 1$ (cf. Eq. (3.2) and Eq. (3.3)):

$$-\mathcal{F}_{\text{ELBO}}(\theta, \phi) = \mathcal{L}(\theta, \phi; \lambda = 1).$$

Thus, we obtain

$$-\nabla_{\theta,\phi} \mathcal{F}_{\text{ELBO}}(\theta, \phi) = \nabla_{\theta,\phi} \mathcal{L}(\theta, \phi; \lambda = 1).$$

Note that for $0 \leq \lambda < 1$, we optimise a lower bound on the ELBO since the impact of the KL, which can be interpreted as a regularisation term, is increased. Conversely, $\mathcal{L}(\theta, \phi; \lambda)$ does not represent a lower bound on $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right]$ anymore if $\lambda > 1$.

3.2.2.2 *Gaussian Likelihood Distribution*

The Gaussian likelihood distribution in VAEs is typically defined to have a diagonal covariance matrix (Kingma and Welling, 2014). This is due to improving computational efficiency. Furthermore, current research, e.g. by Rybkin et al. (2020), shows that parametrising the covariance matrix by a global standard deviation σ (i.e. $\mathbf{\Sigma} = \mathbb{1}\sigma^2$) achieves better results in terms of generation quality and numerical stability. The mean $\boldsymbol{\mu}$, by contrast, is defined as a function of the latent variable. As a consequence, we obtain:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_d - \overbrace{f_\theta^{(d)}(\mathbf{z})}^{\mu_d})^2}{2\sigma^2} \right).$$

The reconstruction-error-related term $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ is specified as

$$\mathcal{C}_\theta(\mathbf{x}, \mathbf{z}) = \sum_{d=1}^{D} \frac{\left(x_d - f_\theta^{(d)}(\mathbf{z})\right)^2}{2\sigma^2},$$

which corresponds to a scaled squared error. For this reason, the gradients are identical for $\lambda = 1$ and a given σ:

$$-\nabla_{\theta,\phi} \mathcal{F}_{\text{ELBO}}(\theta, \phi) = \nabla_{\theta,\phi} \mathcal{L}(\theta, \phi; \lambda = 1).$$

Note that σ can be either predefined or learned in a separate loop.

As in case of the Bernoulli likelihood distribution, a lower bound on the ELBO is optimised if $0 \leq \lambda < 1$. For $\lambda > 1$, minimising $\mathcal{L}(\theta, \phi; \lambda)$ does not result in maximising a lower bound on $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\big[\log p_\theta(\mathbf{x})\big]$ due to the ratio between the KL and the reconstruction term.

3.3 HIERARCHICAL PRIORS FOR LEARNING INFORMATIVE LATENT REPRESENTATIONS

In this section, we propose a hierarchical empirical Bayes prior for VAEs and integrate it with the constrained optimisation method derived in Section 3.1. The goal is to incentivise the learning of informative latent representations as well as to avoid over-regularising the posterior. We achieve this by increasing the complexity of the prior distribution and by providing an optimisation method to learn such models. An introduction to empirical Bayes in the context of amortised variational inference can be found in Section 2.2.3.

3.3.1 *Variational Hierarchical Prior*

It has been shown by Tomczak and Welling (2018) that the optimal empirical Bayes prior is the aggregated posterior distribution $p^*(\mathbf{z}) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\big[q_\phi(\mathbf{z}|\mathbf{x})\big]$. We follow Tomczak and Welling (2018) to approximate the aggregated posterior by a mixture distribution. However, we opt for a continuous mixture in form of a hierarchical model

$$p^*(\mathbf{z}) \approx p_\Theta(\mathbf{z}) = \int p_\Theta(\mathbf{z}|\boldsymbol{\zeta})\, p(\boldsymbol{\zeta})\, \mathrm{d}\boldsymbol{\zeta},$$

with a standard normal $p(\boldsymbol{\zeta})$. As a result, intuitively, our approach inherently favours the learning of continuous latent features.

In order to learn a precise conditional $p_\Theta(\mathbf{z}|\boldsymbol{\zeta})$, we use an importance-weighted lower bound (Burda et al., 2016) on the optimal empirical Bayes prior, i.e. on the aggregated posterior distribution:

$$\begin{aligned}\mathbb{E}_{p^*(\mathbf{z})}\big[\log p_\Theta(\mathbf{z})\big] &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\, \big[\log p_\Theta(\mathbf{z})\big] \\ &\leq \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\, \underbrace{\mathbb{E}_{\boldsymbol{\zeta}_{1:K}\sim q_\Phi(\boldsymbol{\zeta}|\mathbf{z})}\left[\log \frac{1}{K}\sum_{k=1}^{K} \frac{p_\Theta(\mathbf{z}|\boldsymbol{\zeta}_k)\, p(\boldsymbol{\zeta}_k)}{q_\Phi(\boldsymbol{\zeta}_k|\mathbf{z})}\right]}_{=\mathcal{F}_{\text{VHP}}(\Theta,\Phi;\mathbf{z})},\end{aligned}$$

where K is the number of importance samples. A more detailed explanation of the importance-weighted lower bound can be found in Section 2.2.2. We refer to this model as variational hierarchical prior (VHP). The VHP defines an upper bound on the KL, which leads to a hierarchical model with two stochastic layers:

$$\begin{aligned}\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\Big[\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\Theta(\mathbf{z})\big)\Big] &\leq \mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\, \Big[\log q_\phi(\mathbf{z}|\mathbf{x}) - \mathcal{F}_{\text{VHP}}(\Theta, \Phi;\, \mathbf{z})\Big]. \qquad (3.8)\end{aligned}$$

Learning the VHP as part of the VAE is in line with our optimisation problem in Section 3.1 because Eq. (3.8) is an upper bound on the KL and thus a lower bound on the ELBO. As a result, we obtain the Lagrangian

$$\begin{aligned}&\mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \lambda) \\ &\quad = \mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi) + \lambda \left(\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z}) \right] - \kappa^2 \right) \end{aligned} \tag{3.9}$$

and arrive at the constrained optimisation problem

$$\underbrace{\min_{\Theta, \Phi}}_{\substack{\text{Empirical} \\ \text{Bayes}}} \overbrace{\min_{\theta}}^{\text{M-step}} \underbrace{\max_{\lambda} \min_{\phi}}_{\text{E-step}} \mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \lambda) \quad \text{s.t.} \quad \lambda \geq 0. \tag{3.10}$$

3.3.2 *Optimisation Algorithm*

The constrained optimisation problem in Eq. (3.10) can be optimised by the following double-loop algorithm: in the *outer loop* the bound is updated with respect to (Θ, Φ); in the *inner loop* the optimisation problem $\min_\theta \max_\lambda \min_\phi \mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \lambda)$ is solved by applying an update scheme for λ or $\beta = 1/\lambda$. In the following, we use the β-parametrisation to be in line with, for instance, Higgins et al. (2017) and Sønderby et al. (2016).

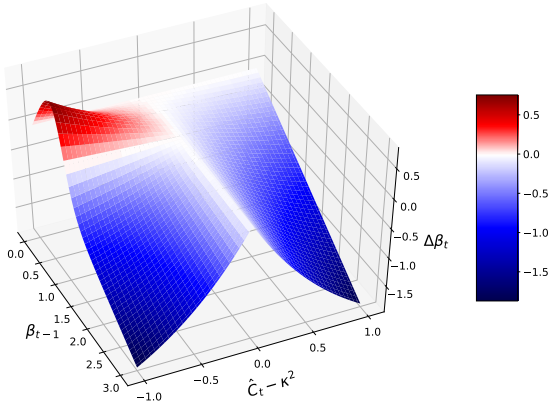
In the GECO update scheme (Eq. (3.4)), β increases/decreases until $\hat{\mathcal{C}}_t = \kappa^2$. However, if the constraint is fulfilled, we want to obtain a tight lower bound on the log-likelihood. As discussed in Section 3.1, this holds when $\beta = 1$ (ELBO)—for $\beta > 1$, we would optimise a scaled lower bound on the ELBO. Therefore, we propose to replace the corresponding β-version of Eq. (3.4) by

$$\beta_t = \beta_{t-1} \cdot \exp\left[\nu \cdot f_\beta\left(\beta_{t-1}, \hat{\mathcal{C}}_t - \kappa^2; \tau\right) \cdot \left(\hat{\mathcal{C}}_t - \kappa^2\right) \right], \tag{3.11}$$

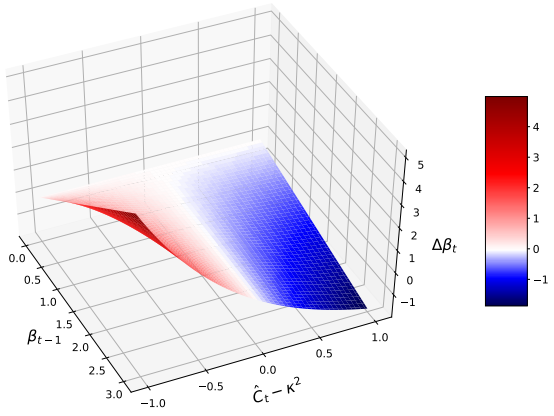
where we define

$$f_\beta(\beta, \delta; \tau) = \left(1 - H(\delta)\right) \cdot \tanh\left(\tau \cdot (\beta - 1)\right) - H(\delta).$$

Here, $H(\bullet)$ is the Heaviside function, and we introduce the slope parameter τ. A visualisation of the β-update scheme, as well as a comparison to GECO, is shown in Figure 3.1. The update can be interpreted as follows: if the constraint is violated, i.e. $\hat{\mathcal{C}}_t > \kappa^2$, the update scheme is equivalent to Eq. (3.4). In case the constraint is fulfilled, the tanh term guarantees that the optimisation process finishes at $\beta = 1$ to obtain/optimise the ELBO at the end of the training. Thus, we impose $\beta \in (0, 1]$, which is reasonable since $\beta < \beta_{\max}$ does not violate the constraint.



(a) REWO



(b) GECO

Figure 3.1: β-update schemes of REWO and GECO: $\Delta\beta_t = \beta_t - \beta_{t-1}$ as a function of β_{t-1} and $\hat{C}_t - \kappa^2$ for $\nu = 1$ and $\tau = 3$.

Note that there are alternative ways to modify Eq. (3.4), as shown in Section 3.5.1. However, in our experiments Eq. (3.11) achieves the best results.

The double-loop approach in Eq. (3.10) is often computationally inefficient. Therefore, we choose to run the inner loop only until the constraint is fulfilled. That is, we optimise Eq. (3.10) with respect to (θ, ϕ) and skip the outer loop (empirical Bayes) updates when the constraint is not satisfied. It turned out that the bound updates were often skipped in the initial phase, but rarely skipped later on. Hence, the algorithm behaves as layer-wise pre-training (Bengio et al., 2007). For these reasons, we propose Algorithm 2 (REWO) that separates the training into two phases: an initial phase where we only optimise the reconstruction error and a main phase where all parameters are updated jointly.

Algorithm 2 (REWO) Reconstruction-error-based weighting of the objective function

Initialise $t = 1$
Initialise $\beta \ll 1$
Initialise InitialPhase = True
while training **do**
  Read current data batch $\mathbf{x}^{\text{ba}}_{1:T}$
  Sample from variational posterior $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}_{\text{ba}})$
  Compute $\hat{\mathcal{C}}_{\text{ba}} = \frac{1}{\text{batch\_size}} \sum_i \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z}_i)$
  Compute $\hat{\mathcal{C}}_t = (1-\alpha) \cdot \hat{\mathcal{C}}_{\text{ba}} + \alpha \cdot \hat{\mathcal{C}}_{t-1}, \quad (\hat{\mathcal{C}}_0 = \hat{\mathcal{C}}_{\text{ba}})$
  if $\hat{\mathcal{C}}_t < \kappa^2$ **then**
    InitialPhase = False
  end if
  if InitialPhase **then**
    Optimise $\mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \beta)$ w.r.t. θ, ϕ
  else
    $\beta \leftarrow \beta \cdot \exp\left[\nu \cdot f_\beta\left(\beta, \hat{\mathcal{C}}_t - \kappa^2; \tau\right) \cdot \left(\hat{\mathcal{C}}_t - \kappa^2\right)\right]$
    Optimise $\mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \beta)$ w.r.t. $\theta, \phi, \Theta, \Phi$
  end if
  $t \leftarrow t + 1$
end while

The initial phase starts with $\beta \ll 1$ to enforce a reconstruction optimisation. Thus, the model is first trained to achieve a good encoding of the data through $q_\phi(\mathbf{z}|\mathbf{x})$, which is measured by the reconstruction error. In order to prevent β of becoming smaller than the initial value during the first iteration steps, we do not update β as long as the condition $\hat{\mathcal{C}}_t < \kappa^2$ is not fulfilled. A good encoding is required to learn the conditionals $q_\Phi(\boldsymbol{\zeta}|\mathbf{z})$ and $p_\Theta(\mathbf{z}|\boldsymbol{\zeta})$ in the second stochastic layer. In the main phase, i.e. as soon as $\hat{\mathcal{C}}_t < \kappa^2$ is fulfilled, we additionally start to optimise the parameters of the second stochastic layer (Θ, Φ) and to update β. This approach avoids posterior collapse in both stochastic layers and helps the VHP to learn an informative latent representation of the observed data. See Section 3.5.1 for empirical evidence.

3.3.3 *Summarising Note*

The proposed method—which is a combination of an ELBO-like Lagrangian and an importance-weighted lower bound—can be interpreted as follows: the posterior of the first stochastic layer $q_\phi(\mathbf{z}|\mathbf{x})$ can learn an informative latent representation due to the hierarchical empirical Bayes prior. The parameters of the prior are optimised by applying an importance-weighted lower bound on the optimal empirical Bayes prior, which is the aggregated posterior $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}[q_\phi(\mathbf{z}|\mathbf{x})]$. Despite the diagonal Gaussian $q_\Phi(\boldsymbol{\zeta}|\mathbf{z})$, the importance weighting allows us to learn a precise conditional $p_\Theta(\mathbf{z}|\boldsymbol{\zeta})$ from the standard normal distribution $p(\boldsymbol{\zeta})$ to the aggregated posterior (cf. Section 2.2.2). Note that for

this purpose we could have also used, for example, normalising flows (Rezende and Mohamed, 2015) instead of importance weighting.

3.4 GRAPH-BASED INTERPOLATION FOR VERIFYING LATENT REPRESENTATIONS

A major reason for using the VHP is to facilitate the learning of informative latent representations, as it allows for less over-regulation of the posterior distribution.

To verify the quality of the latent representations, we build on the manifold hypothesis defined by Cayton (2005) and Rifai et al. (2011a). It can be summarised by the following assumption: real-world data presented in high-dimensional spaces is likely to concentrate in the vicinity of nonlinear sub-manifolds of much lower dimensionality, where distance reflects the similarity of data. Following this hypothesis, we can evaluate the quality of a learned latent representation—i.e. if the latent representation reflects the topology of the data—by reconstructing interpolations between points on the latent manifold.

To realise the above idea, we propose a graph-based method (N. Chen et al., 2019) that summarises the continuous latent space by a graph consisting of a finite number of nodes $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$. The nodes are N random samples from the (learned) prior distribution:

$$\mathbf{z}_n, \boldsymbol{\zeta}_n \sim p_\Theta(\mathbf{z}|\boldsymbol{\zeta})\, p(\boldsymbol{\zeta}), \quad n = 1, \ldots, N.$$

The graph is constructed by connecting each node through undirected edges to its k-nearest neighbours. The edge weights are defined as the Euclidean distances between the related node pairs in the latent space.

Once the graph is built, interpolation between two additional data points $\mathbf{x}_i$ and $\mathbf{x}_j$ can be realised as follows: first, both data points are encoded $\mathbf{z}_{(\cdot)} = \mu_\phi(\mathbf{x}_{(\cdot)})$, where $\mu_\phi(\mathbf{x}_{(\cdot)})$ is the mean of $q_\phi(\mathbf{z}|\mathbf{x}_{(\cdot)})$, and added as new nodes to the existing graph. Afterwards, a classic search algorithm such as $A^\star$ can be used to find the shortest path between $\mathbf{z}_i$ and $\mathbf{z}_j$ through the graph. The result is a sequence $\mathbf{Z}_{\text{path}} = (\mathbf{z}_i, \mathbf{Z}_{\text{sub}}, \mathbf{z}_j)$—with $\mathbf{Z}_{\text{sub}} \subseteq \mathbf{Z}$—that represents the shortest path on the learned latent manifold. Finally, we obtain the interpolation by reconstructing $\mathbf{Z}_{\text{path}}$ to the observable space.

3.5 EXPERIMENTAL RESULTS

We conduct the following experiments to validate our approach. In Section 3.5.1, we demonstrate that our method learns to represent the factor of variation in the data of a moving pendulum. Section 3.5.2 deals with the generation of human movements based on the learned latent representations of real-world human motion data (CMU Graphics Lab Motion Capture Database). In Section 3.5.3, we evaluate the

marginal log-likelihood on standard datasets, such as MNIST, Fashion-MNIST, and OMNIGLOT. In Section 3.5.4, the method is compared on the high-dimensional image datasets 3D Faces and 3D Chairs. The model architectures used in our experiments can be found in Appendix A.1.1.

3.5.1 *Artificial Pendulum Dataset*

We have created a dataset of 15,000 images of a moving pendulum (see Figure 3.7). Each image has a size of 16×16 pixels, and the joint angles are distributed uniformly in the range $[0, 2\pi)$. Thus, the joint angle is the only factor of variation.

3.5.1.1 *Comparison of Different Optimisation Approaches*

Figure 3.2 shows latent representations of the pendulum data learned by the VHP when applying REWO or GECO (we use the same κ in both cases). The variance of the posterior's standard deviation, expressed by the greyscale, measures whether the contribution to the ELBO is equally distributed over all data points.

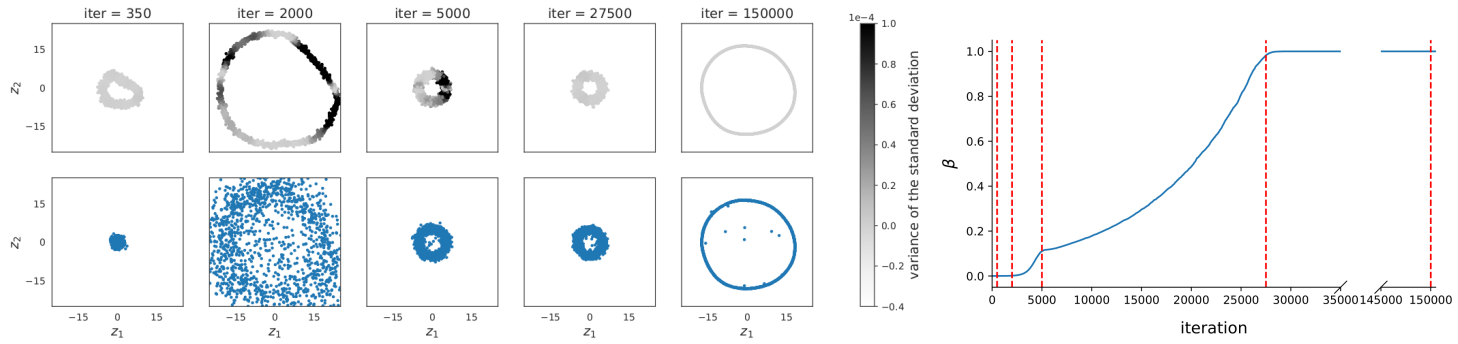
Furthermore, we compare REWO and GECO to the classical amortised variational inference approach. In the latter, we optimise the VAE objective (not the Lagrangian) defined as the combination of an ELBO and an importance-weighted bound, similar to Eq. (3.9). The main difference is that $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$ is replaced by the negative log-likelihood. Additionally, we combine amortised variational inference with warm up (WU) (Sønderby et al., 2016): a linear annealing schedule of β, which is an established method for improving the optimisation process and for avoiding posterior collapse. The related plots can be found in Figure 3.3.

To validate how informative the learned latent representations are, we apply a linear regression (Figure 3.4), where the rotation angles

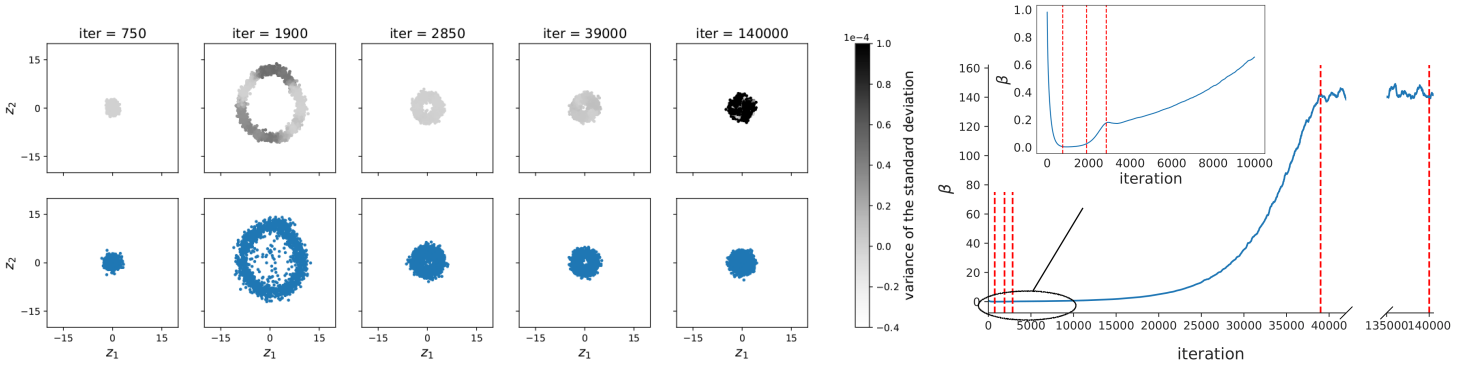
Table 3.1: Prediction of the pendulum's joint angle by means of an OLS regression on the learned latent representation. The encodings are transformed to polar coordinates, and the rotation angles are used as labels.

method	absolute error
VHP + REWO	0.054
VHP + GECO	0.53
VHP*	0.49
VHP* + WU (20 epochs)	0.20
VHP* + WU (200 epochs)	0.31

*VAE objective optimised through amortised variational inference

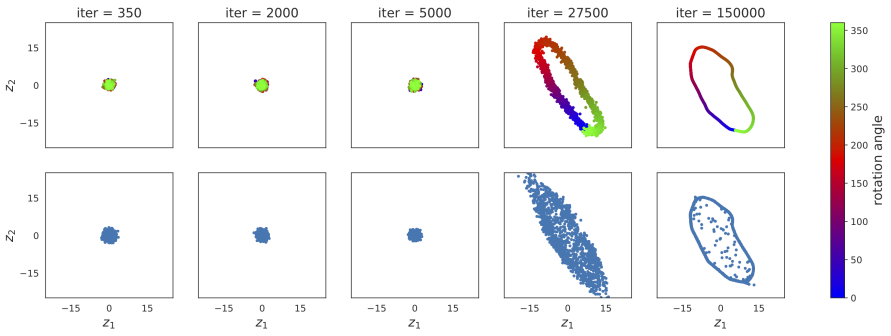


(a) VHP + REWO

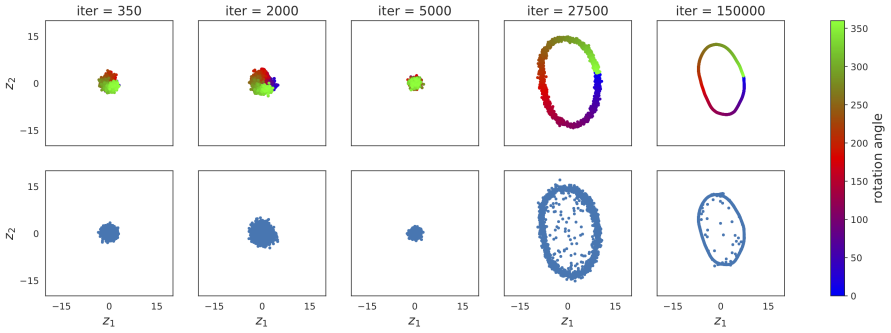


(b) VHP + GECO

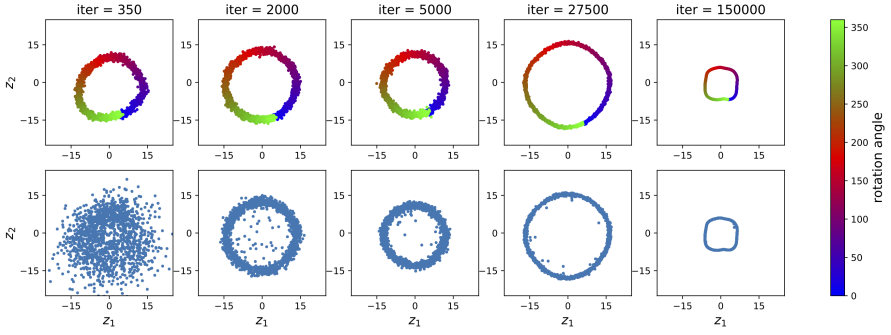
Figure 3.2: (left) Latent representation of the pendulum data at different iteration steps when optimising $\mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \beta)$ with REWO or GECO. The top row shows the approximate posterior: the greyscale encodes the variance of its standard deviation. The bottom row shows the hierarchical prior. (right) β as a function of the iteration steps: red lines mark the visualised iteration steps.



(a) VHP



(b) VHP + WU (20 epochs)



(c) VHP + WU (200 epochs)

Figure 3.3: Latent representation of the pendulum data at different iteration steps when optimising the VHP via amortised variational inference (+WU). The top row shows samples from the approximate posterior, where the colour encodes the rotation angle of the pendulum. The bottom row shows samples from the hierarchical prior.

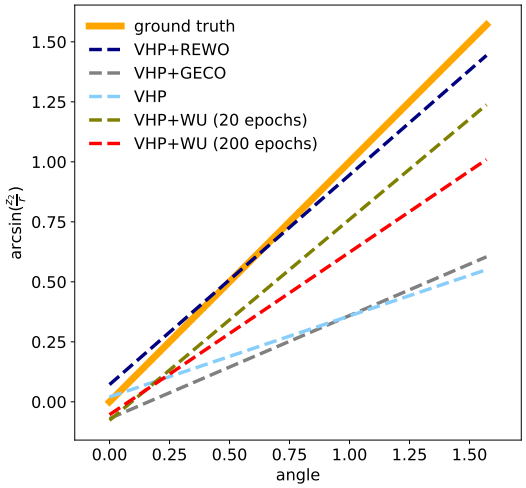


Figure 3.4: Prediction of the pendulum's joint angle by an OLS regression on the latent representations learned by the VHP when applying REWO, GECO, or amortised variational inference (+WU). The absolute errors can be found in Table 3.1.

(ground truth) function as labels. For this purpose, the encodings are transformed to polar coordinates, where the encoded rotation angle is computed by $\arcsin(z_2/r)$. Here, z_2 is the second dimension of the latent space, and the radius r is estimated from the learned latent representation. The goal is to predict the joint angle of the pendulum as precise as possible. Table 3.1 shows the absolute errors of the ordinary least squares (OLS) regression on the latent representations learned by the different optimisation approaches. As a result, REWO leads to the most precise representation of the ground truth.

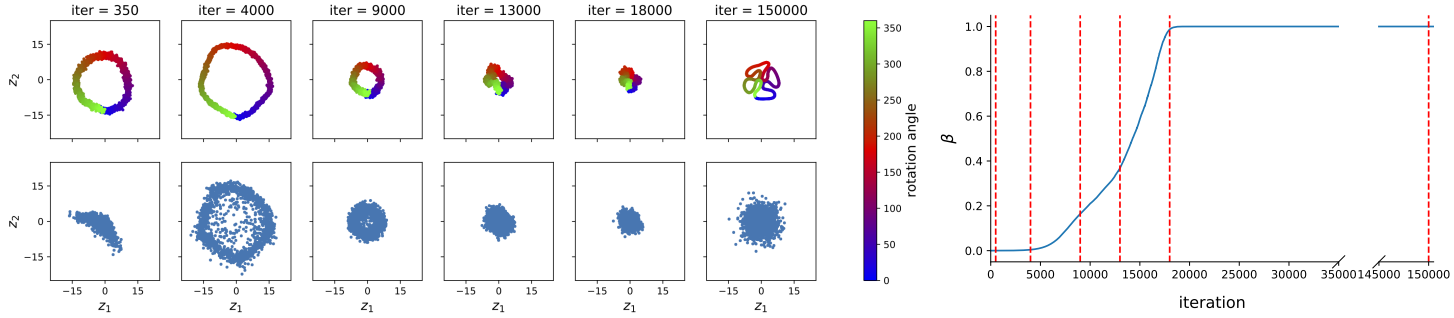
3.5.1.2 *Variational Hierarchical Prior Without Importance Weighting*

In the context of VAEs, optimising an importance-weighted bound instead of the original ELBO results in a more expressive posterior distribution (Cremer et al., 2017). In the following, we demonstrate the impact of the importance-weighted bound in the second stochastic layer in our model on the learned latent representation. Without importance weighing, the VHP learns poor encodings, as shown in Figure 3.5a. This is because the model compensates the less expressive posterior $q_\Phi(\boldsymbol{\zeta}|\mathbf{z})$ (compared to the original VHP implementation) by restricting $q_\phi(\mathbf{z}|\mathbf{x})$, as discussed in Section 3.3.

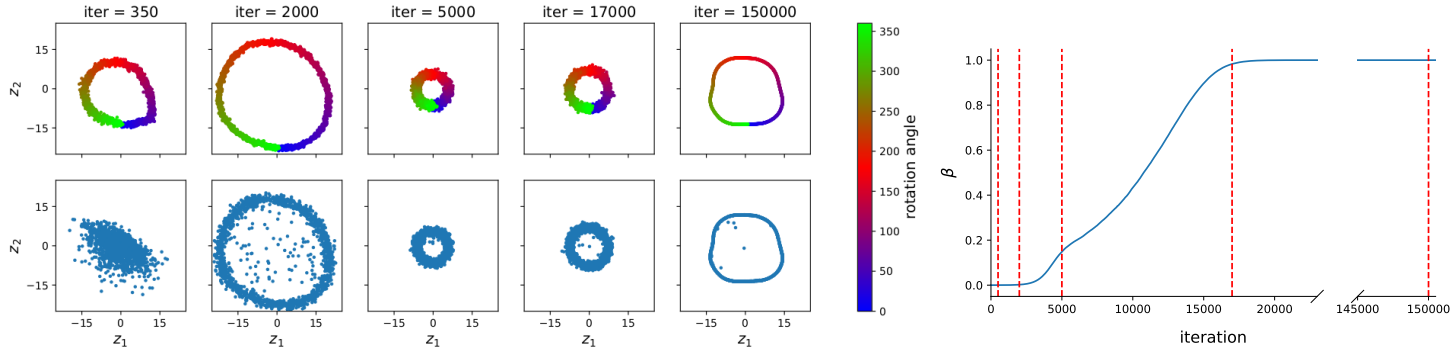
3.5.1.3 *Alternative Update Scheme for the Lagrange Multiplier*

In the following, we compare our proposed update rule in Eq. (3.11) with an alternative one in order to elaborate on our design choices. The alternative λ-update scheme also guarantees $\lambda \geq 1$ and thus $\beta \leq 1$:

$$\lambda_t = 1 + \tau \cdot \gamma_t, \quad \text{with} \quad \gamma_t = \gamma_{t-1} \cdot \exp\left(\nu \cdot (\hat{C}_t - \kappa^2)\right). \tag{3.12}$$



(a) VHP + REWO (with ELBO instead of importance-weighted bound in the second stochastic layer)



(b) VHP + REWO (with different β-update scheme)

Figure 3.5: (left) Latent representation of the pendulum data at different iteration steps when optimising $\mathcal{L}_{\text{VHP-VAE}}(\theta, \phi, \Theta, \Phi; \beta)$. The top row shows the approximate posterior: the colour encodes the rotation angle of the pendulum. The bottom row shows samples from the hierarchical prior. (right) β as a function of the iteration steps: red lines mark the visualised iteration steps.

where ν is defined as the update's learning rate, and τ is a slope parameter. Eq. (3.12) defines the β-update scheme as

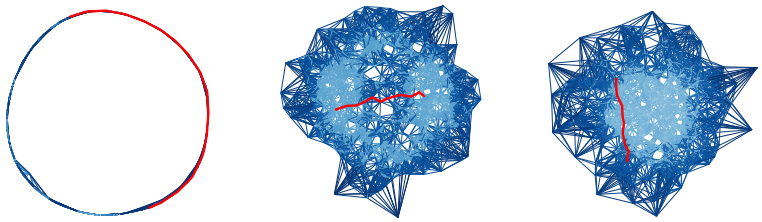
$$\beta_t = \frac{1}{1 + \tau \cdot \gamma_t}. \tag{3.13}$$

Figure 3.5b shows the latent representation learned by the VHP when applying REWO with the alternative update rule from Eq. (3.13). In our experiments, Eq. (3.13) was more difficult to tune than the proposed β-update in Eq. (3.11), resulting in less informative latent representations (cf. Figure 3.5b and Figure 3.2a). Furthermore, unlike the update rule in Eq. (3.11), the update scheme in Eq. (3.13) allows any $\beta > 0$ to be chosen as the initial value.

3.5.1.4 *Graph-Based Interpolation*

Finally, we use the graph-based interpolation method described in Section 3.4 to compare the latent representations learned by the VHP and the importance-weighted autoencoder (IWAE) (Burda et al., 2016).

The graphs in Figure 3.6 are built with 1000 samples from the prior distribution of the respective model. The red curves depict the interpolation on the learned latent manifold. As start and end point, we choose pendulum images with a joint angle of 0 and 180 degrees, respectively. The reconstructions of the interpolations are shown in Figure 3.7. The top row (VHP + REWO) shows a smooth change of the joint angles, whereas the middle (VHP + GECO) and bottom row (IWAE) contain discontinuities that result in unrealistic interpolations.



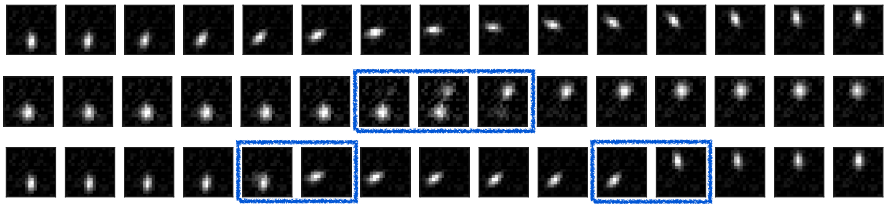
(a) VHP + REWO **(b)** VHP + GECO **(c)** IWAE

Figure 3.6: Graph-based interpolations on the learned latent manifold. The graphs are based on samples of the respective prior distributions. The red curves depict the interpolations, the bluescale indicates the edge weight.

3.5.2 *Human Motion Capture Database*

The CMU Graphics Lab Motion Capture Database[2] consists of a large number of human motion recordings, which have been recorded by

2 mocap.cs.cmu.edu

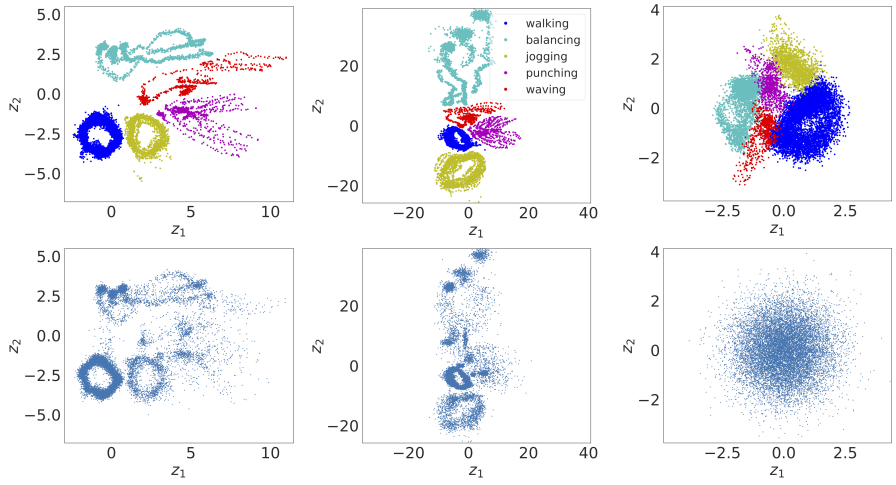


top: VHP + REWO, middle: VHP + GECO, bottom: IWAE

Figure 3.7: Pendulum reconstructions of the graph-based interpolation on the learned latent manifold, shown in Figure 3.6. Discontinuities are marked by blue boxes.

using a motion capture system. Human subjects wear 41 markers while walking, dancing, etc. The data is pre-processed as described by N. Chen et al. (2015) such that each frame is represented by a 50-dimensional feature vector. In our experiments, we use five different motions: walking, balancing, jogging, punching, and waving.

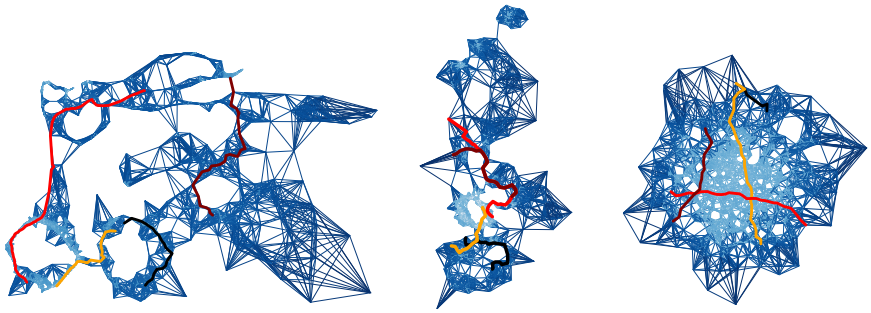
We compare our method with the VampPrior (Tomczak and Welling, 2018) and the IWAE. Samples from the prior and aggregated approximate posterior of the three methods are shown in Figure 3.8. As expected, for both the VHP and VampPrior the latent representations of different movements are separated. In both cases the learned prior matches the aggregated posterior. By contrast, the IWAE is restricted through the Gaussian prior and cannot learn a spatially separated representation of the different motions.



(a) VHP + REWO **(b)** VampPrior **(c)** IWAE

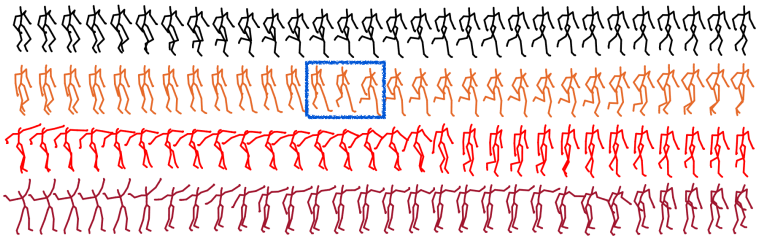
Figure 3.8: Latent representation of human motion data: approximate posterior (top) and prior (bottom). The colour encodes the five human motions. The different sample densities are caused by a different amount of data points for each motion.

Figure 3.9 shows four generated interpolations using our graph-based interpolation method: between two frames within the same

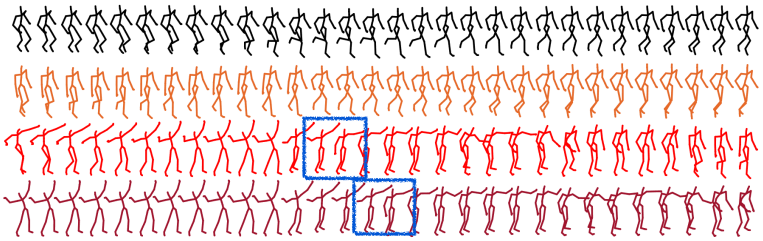


(a) VHP + REWO **(b)** VampPrior **(c)** IWAE

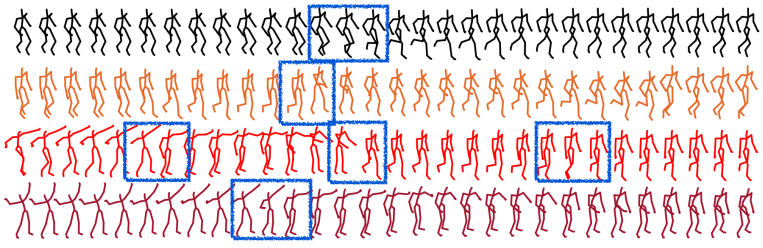
Figure 3.9: Graph-based interpolation of human motions. The graphs are based on the (learned) prior distributions, depicted in Figure 3.8. The bluescale indicates the edge weight. The coloured lines represent four interpolated movements depicted in Figure 3.10.



(a) VHP + REWO



(b) VampPrior



(c) IWAE

Figure 3.10: Human-movement reconstructions of the graph-based interpolations in Figure 3.9. The blue boxes mark discontinuities.

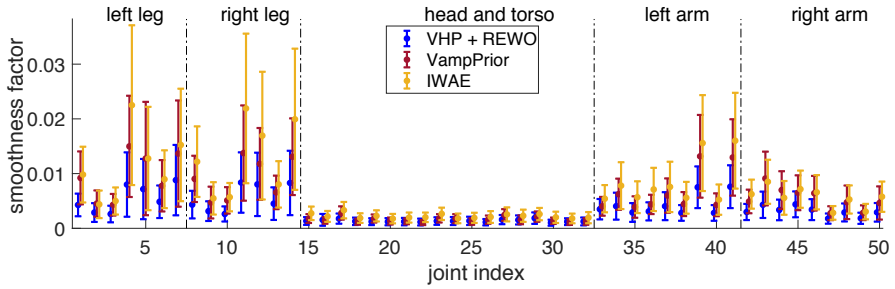


Figure 3.11: Smoothness measure of the human-movement interpolations. For each joint, the mean and standard deviation of the smoothness factor are displayed. Smaller values correspond to smoother movements.

motion (black line) and of different motions (orange, red, and maroon). The reconstructions are depicted in Figure 3.10. In contrast to the IWAE, the VampPrior and the VHP enable smooth interpolations.

Figure 3.11 depicts the movement smoothness factor, which we define as the root mean square of the second order finite difference along the interpolated path. Thus, smaller values correspond to smoother movements. For each of the three methods, the smoothness factor is averaged across 10 graphs, each with 100 interpolations. The starting and ending points are randomly selected. As a result, the latent representation learned by the VHP leads to smoother movement interpolations than in case of the VampPrior and the IWAE.

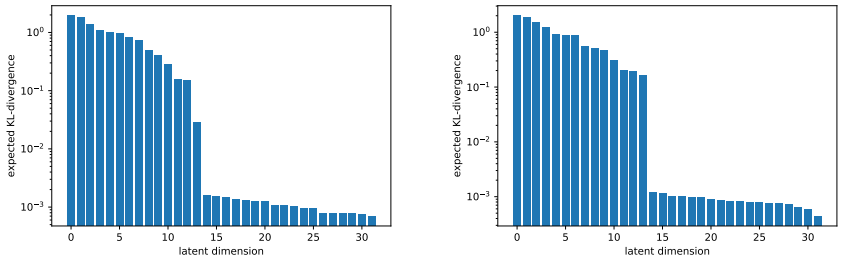
3.5.3 *Evaluation on MNIST, Fashion-MNIST, and OMNIGLOT*

We compare our method quantitatively with the VampPrior and the IWAE on MNIST (Larochelle and Murray, 2011; Lecun et al., 1998), Fashion-MNIST (Xiao et al., 2017), and OMNIGLOT (Lake et al., 2015).

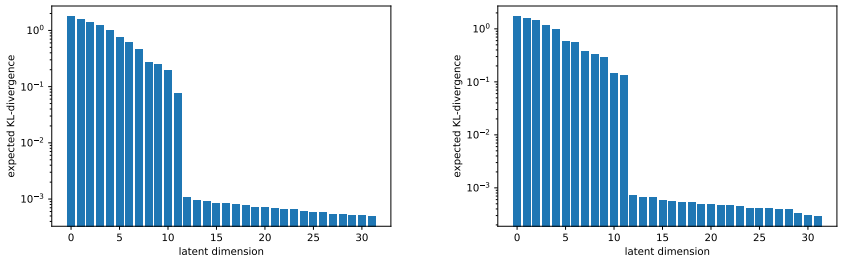
To this end, we report the marginal log-likelihood on the respective test set. Following the test protocol of previous work by Tomczak and Welling (2018), we evaluate the log-likelihood using importance sampling with 5,000 samples (Burda et al., 2016). The results are

Table 3.2: Negative test log-likelihood estimated with 5,000 importance samples.

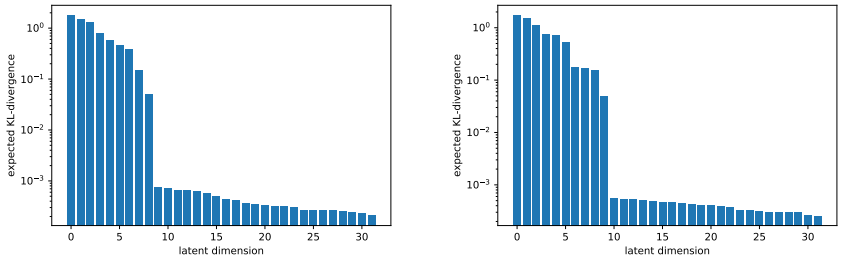
	dynamic MNIST	static MNIST	Fashion-MNIST	OMNIGLOT
VHP + REWO	78.88	82.74	225.37	101.78
VHP + GECO	95.01	96.32	234.73	108.97
VampPrior	80.42	84.02	232.78	101.97
IWAE (L=1)	81.36	84.46	226.83	101.57
IWAE (L=2)	80.66	82.83	225.39	101.83



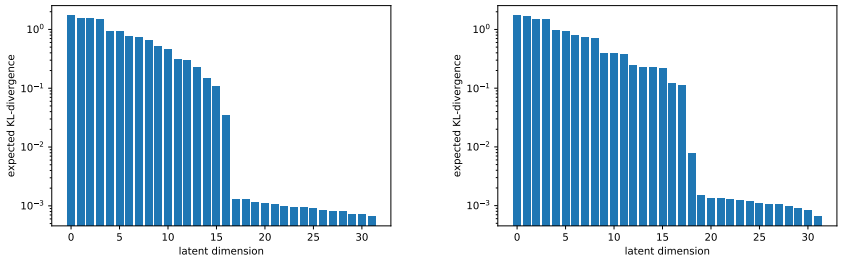
(a) static MNIST



(b) dynamic MNIST



(c) Fashion-MNIST



(d) OMNIGLOT

Figure 3.12: VHP optimised with REWO (left) and WU (right). The plots show $\mathrm{KL}\big(q_\Phi(\zeta_d|\mathbf{x}) \;||\; p(\zeta_d)\big)$ depending on the latent dimension d. The dimensions are sorted by the KL value, and the histograms are displayed on a logarithmic scale.

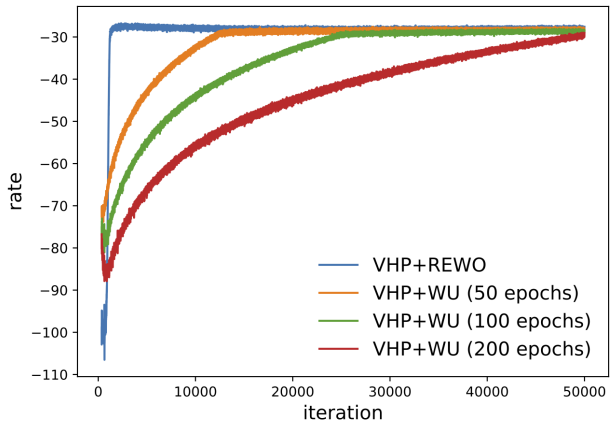


Figure 3.13: Curse of the rate (which corresponds to $-\mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi)$) as a function of the iteration step for different optimisation strategies on static MNIST data.

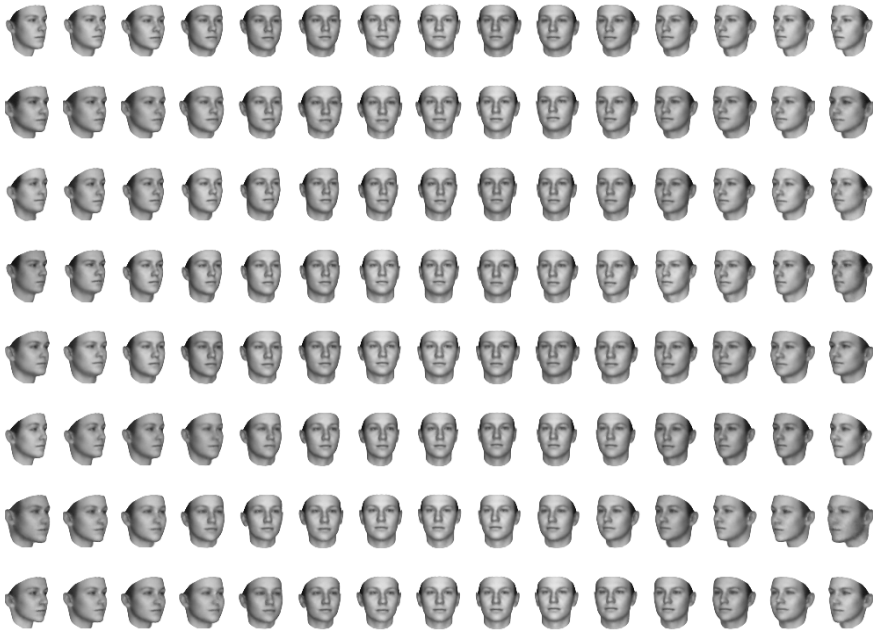
reported in Table 3.2. VHP + REWO performs as good or better than state-of-the-art on these datasets. The same κ is used for training VHP with REWO and GECO. The two stochastic layer hierarchical IWAE does not perform better than VHP + REWO, supporting our claim that a flexible prior in the first stochastic layer and a powerful approximate posterior (importance sampling) in the second stochastic layer is sufficient.

Furthermore, REWO prevents the posterior from collapsing and leads to a similar amount of active units as WU. We show this, similar to Sønderby et al. (2016), by evaluating $\mathrm{KL}\big(q_\Phi(\zeta_d|\mathbf{x}) \;||\; p(\zeta_d)\big)$, where $\prod_{d=1}^{D} q_\Phi(\zeta_d|\mathbf{x}) = q_\Phi(\boldsymbol{\zeta}|\mathbf{x})$, and D is the dimensionality of the second stochastic layer represented by the latent variable $\boldsymbol{\zeta}$. The main difference to WU, however, is that REWO can learn an informative latent representation of the data after much fewer iterations. This can be shown by analysing the rate, which corresponds to $-\mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi)$ in Eq. (3.8), and is motivated by the rate–distortion theory (Shannon, 1948). Figure 3.13 displays the course of the rate depending on the iteration step for different optimisation strategies on static MNIST data. Note that this result is consistent with the experimental findings in Section 3.5.1 (cf. Figures 3.2 and 3.3).

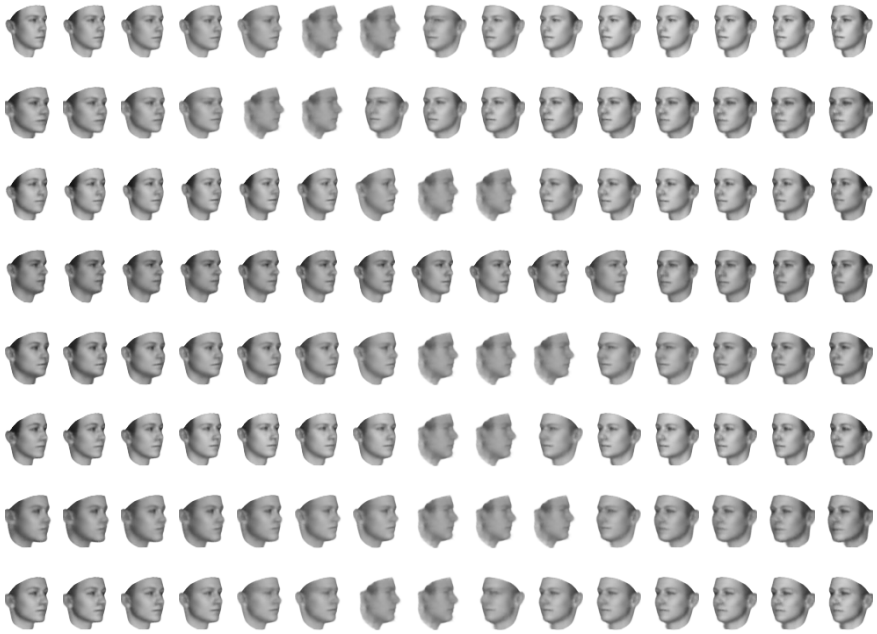
3.5.4 *Qualitative Results on 3D Chairs and 3D Faces*

We have generated the 3D Faces dataset (Paysan et al., 2009) based on images of 2000 faces with 37 views each. The 3D Chairs (Aubry et al., 2014) dataset consists of 1393 chair images with 62 views each. In both cases, the images have a size of 64×64 pixels.

We compare our approach to the IWAE, using a 32-dimensional latent space. The learned encodings are evaluated qualitatively with the graph-based interpolation method. Figures 3.14a and 3.15a show

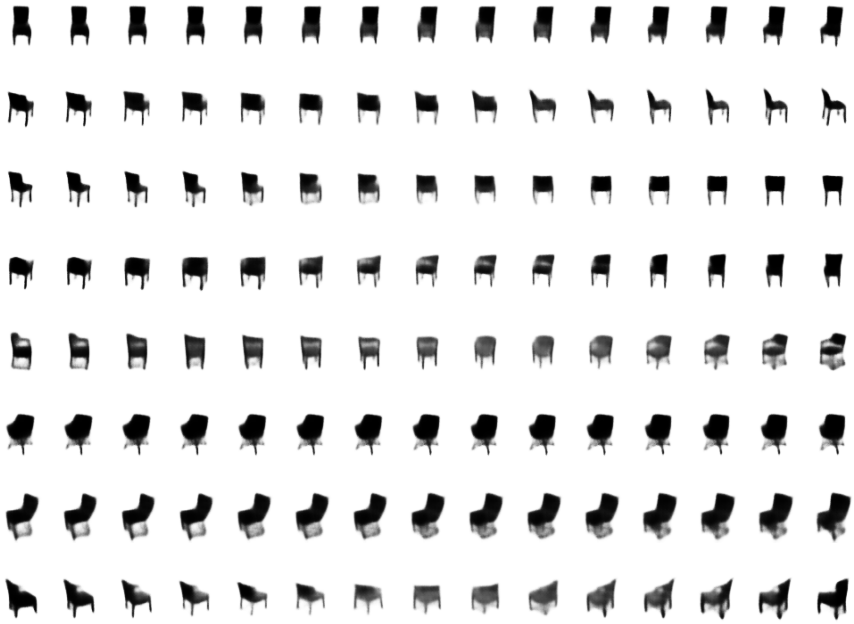


(a) VHP + REWO

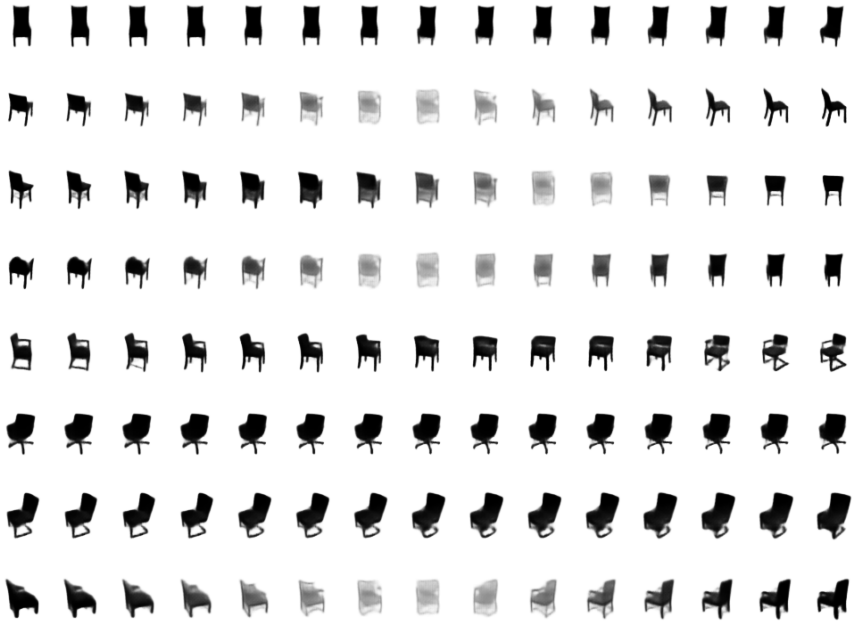


(b) IWAE

Figure 3.14: 3D Faces: graph-based interpolations between two data points from the test set on the learned 32-dimensional latent manifold. The graph is built by using samples from the (learned) prior distribution.



(a) VHP + REWO



(b) IWAE

Figure 3.15: 3D Chairs: graph-based interpolations between two data points from the test set on the learned 32-dimensional latent manifold. The graph is built by using samples from the (learned) prior distribution.

interpolations on the latent manifold, learned by the VHP + REWO. Compared to the IWAE (Figures 3.14b and 3.15b), they are less blurry and smoother, i.e. more realistic.

3.6 SUMMARISING DISCUSSION

In this chapter, we have addressed the question of how to learn informative latent representations of data with VAEs. To this end, we have reformulated the ELBO as the Lagrangian of a constrained optimisation problem. We have proposed a hierarchical empirical Bayes prior to enable richer latent representations of observed data. In order to learn the hierarchical prior, we have extended the constrained optimisation approach to hierarchical models by using an importance-weighted bound on the optimal empirical Bayes prior. Concurrently, we have introduced the associated optimisation algorithm to facilitate good encodings.

We have shown that the learned hierarchical prior is indeed non-trivial; moreover, it is well-adapted to the latent representation reflecting the topology of the data. In contrast to the original VAE, our method provides informative latent representations and performs particularly well on data where the relevant features change continuously. In case of the moving pendulum (Section 3.5.1), the model has learned to represent the factor of variation in the data, which allows us to predict the pendulum's angle by a simple ordinary least squares regression. Furthermore, we have shown in this context that our proposed optimisation algorithm REWO leads to a significant improvement of the latent representation's interpretability compared to previous optimisation methods. The experiments on real-world human motion data in Section 3.5.2 and on the high-dimensional 3D Faces and 3D Chairs datasets in Section 3.5.4 have demonstrated that the learned hierarchical prior leads to smoother and more realistic interpolations than a standard normal prior or the VampPrior. Moreover, we have achieved test log-likelihoods (Section 3.5.3) comparable to or better than state-of-the-art on established benchmark datasets.

The capability of our model to learn complex latent representations that reflect the topology of the data is a useful tool for various VAE-based approaches. For this reason, the introduced method forms the basis for the following chapters in this dissertation: in Chapter 4, we extend it to deep state-space models in order to improve the prediction accuracy of dynamic systems; and Chapter 5 addresses the issue of learning encodings that allow measuring the similarity of data in the context of unsupervised metric learning.

4

EXTENSION TO SEQUENTIAL DATA: LEARNING DEEP STATE-SPACE MODELS

The methods and experimental results discussed in this chapter have been previously published in (Klushyn et al., 2021). Sections 4.1, 4.2, 4.3, and 4.4 are based on revised text from this publication.

This chapter addresses the question of how to learn deep state-space models (DSSMs) in order to obtain accurate temporal predictions of observed dynamic systems. To this end, we extend the constrained optimisation approach introduced in Chapter 3 to sequential data.

A limitation frequently encountered in machine learning—and science in general—is systems that can be (partially) observed, but whose exact dynamics and constraints are unknown. Nevertheless, an accurate model of the system is essential because we want, for example, to forecast weather, traffic flow, or electricity consumption (e.g. Rangapuram et al., 2018; Salinas et al., 2020). Further areas of application are model-based reinforcement learning and model predictive control. Here, the learned model of an unknown dynamic system is used for planning, e.g. for learning to fly a drone (Becker-Ehmck et al., 2020) or to make a humanoid robot run (Levine and Koltun, 2013). However, learning models that are accurate enough for forecasting or planning is the subject of current research, especially in image-based domains (e.g. Hafner et al., 2019).

DSSMs enable temporal predictions by learning the underlying dynamic system of observed sequential data. They infer a typically lower-dimensional latent representation—similar to VAEs—which is then used to learn a transition model that takes into account the dynamics and constraints of the observed system. The idea is that learning a transition model based on low-dimensional latent variables is more feasible than on the basis of high-dimensional sensory data, such as images.

DSSMs are typically trained by maximising the sequential evidence lower bound (ELBO). We show that high ELBO values, however, do not indicate the model has learned the underlying dynamics, i.e. to accurately predict the observed system. We address this problem as follows:

- In Section 4.1, we propose a constrained optimisation framework, including an empirical Bayes method, as a general approach for learning DSSMs. To this end, we extend the method introduced in Chapter 3 to DSSMs. We do this by formulating the sequential

ELBO as the Lagrangian of a constrained optimisation problem and by introducing a hierarchical empirical Bayes prior distribution for the initial time step. Concurrently, we introduce the associated optimisation algorithm, an extension of REWO (Algorithm 2) and show how our framework can be applied to existing models.

- Building upon the constrained optimisation framework, we introduce in Section 4.2 the extended Kalman variational autoencoder, which combines amortised variational inference with classic Bayesian filtering/smoothing to model dynamics more accurately than RNN-based DSSMs.
- In Section 4.3, we demonstrate how to learn state-space representations with the extended Kalman variational autoencoder, where static and dynamic features are disentangled—and how to use these in the context of model-based reinforcement learning for defining reward functions and validating the learned model.
- In Section 4.4, we evaluate our proposed method. This includes experiments on image data of a moving pendulum and on the reacher environment of Deepmind's control suite, where we use angle as well as high-dimensional image data as observations.

4.1 CONSTRAINED OPTIMISATION FRAMEWORK FOR IMPROVED SYSTEM IDENTIFICATION

4.1.1 *A Rate–Distortion Perspective on Deep State-Space Models*

DSSMs (e.g. Karl et al., 2017; Krishnan et al., 2015; Watter et al., 2015) model an unknown distribution of observed sequential data $\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ by means of typically lower-dimensional latent variables $\mathbf{z}_{1:T}$ that represent the underlying state of the system. To achieve this, the Markov assumption is imposed. It states that the future state $\mathbf{z}_{t+1}$ as well as the current observation $\mathbf{x}_t$ solely depend on $\mathbf{z}_t$:

$$\begin{aligned} p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T} | \, \mathbf{u}_{1:T}) &= p_\theta(\mathbf{x}_{1:T} | \mathbf{z}_{1:T}) \, p_\psi(\mathbf{z}_{1:T} | \, \mathbf{u}_{1:T}) \\ &= p(\mathbf{z}_1) \, p_\theta(\mathbf{x}_1 | \, \mathbf{z}_1) \prod_{t=2}^{T} p_\psi(\mathbf{z}_t | \, \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \, p_\theta(\mathbf{x}_t | \, \mathbf{z}_t), \end{aligned} \tag{4.1}$$

where $\mathbf{u}_{1:T}$ are optional control signals (actions), and the use of different parameters (θ, ψ) will become important in the course of this chapter. Note that a detailed introduction to state-space models can be found in Section 2.3.

The model parameters in Eq. (4.1) are often learned through amortised variational inference (e.g. Karl et al., 2017; Krishnan et al., 2015). This requires introducing a recognition model $q_\phi(\mathbf{z}_{1:T} | \, \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ that

learns—in combination with the transition model $p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$—the dynamics underlying the observed data. The resulting objective function is known as sequential ELBO:

$$\begin{aligned}\mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\log p(\mathbf{x}_{1:T}|\,\mathbf{u}_{1:T})\right] &\geq \mathcal{F}_{\text{ELBO}}(\theta,\psi,\phi)\\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\log \frac{p_\theta(\mathbf{x}_{1:T}|\,\mathbf{z}_{1:T})\,p_\psi(\mathbf{z}_{1:T}|\,\mathbf{u}_{1:T})}{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\right],\end{aligned} \tag{4.2}$$

where $p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})$ is the empirical distribution representing the dataset $\mathcal{D} = \{\mathbf{x}^{(i)}_{1:T}, \mathbf{u}^{(i)}_{1:T}\}_{i=1}^N$.

In the context of generative models, the ELBO can be divided into a reconstruction term (distortion) and a compression term (rate) (Alemi et al., 2018). We transfer this approach to deep state-space models, where the distortion,

$$\mathcal{D}(\theta,\phi) = -\,\mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\log p_\theta(\mathbf{x}_{1:T}|\,\mathbf{z}_{1:T})\right], \tag{4.3}$$

optimises the model's ability for reconstructing observations—whereas the rate,

$$\mathcal{R}(\phi,\psi) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\mathrm{KL}\big(q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})\,\|\,p_\psi(\mathbf{z}_{1:T}|\,\mathbf{u}_{1:T})\big)\right], \tag{4.4}$$

enables learning the underlying dynamics. Eq. (4.3) and (4.4) lead to the following general formulation of the ELBO:

$$\mathcal{F}_{\text{ELBO}}(\theta,\psi,\phi) = -\mathcal{D}(\theta,\phi) - \mathcal{R}(\psi,\phi). \tag{4.5}$$

Balancing the ratio between distortion and rate during optimisation can be an effective approach to improve the learning of DSSMs, as we discuss in the following section.

4.1.2 *The Sequential Evidence Lower Bound as Lagrangian Function*

High ELBO values do not necessarily imply that the model has learned the underlying system dynamics of the observed data, as we verify in Section 4.4.2. This is because different combinations of rate and distortion can result in the same ELBO value. Previous work addresses this issue by introducing weighting schedules for either $\mathcal{D}(\theta,\phi)$ or $\mathcal{R}(\psi,\phi)$ (e.g. Bowman et al., 2016) since a different ratio favours either better reconstruction or compression (Alemi et al., 2018). However, we demonstrate in Section 4.4 that balancing reconstruction and compression with predefined annealing schedules often does not achieve the desired result.

In Chapter 3, we have defined the objective function of VAEs as the Lagrangian of a constrained optimisation problem, which allows for controlling the model's reconstruction quality. We transfer this

approach to DSSMs to guarantee a good reconstruction—i.e. a low $\mathcal{D}(\theta, \phi)$—and thus provide a sufficient basis for learning the underlying dynamic system. To this end, we formulate the sequential ELBO as the Lagrangian of a constrained optimisation problem by specifying the rate $\mathcal{R}(\psi, \phi)$ in Eq. (4.5) as optimisation objective and by imposing an inequality constraint $\mathcal{C}(\theta, \phi) \leq \kappa^2$. Here, $\mathcal{C}(\theta, \phi)$ is defined as the reconstruction-error-related term in $\mathcal{D}(\theta, \phi)$, which is, for instance, the cross-entropy in case of a Bernoulli or the mean squared error in case of Gaussian likelihood. Further details can be found in Section 3.2. The corresponding Lagrangian is

$$\mathcal{L}(\theta, \psi, \phi; \lambda) = \mathcal{R}(\psi, \phi) + \lambda \left(\mathcal{C}(\theta, \phi) - \kappa^2 \right),$$

where the Lagrange multiplier λ can be viewed as a weighting term for the distortion.

As discussed in Section 3.1, the original EM algorithm for optimising the ELBO, $\min_{\theta,\psi} \min_{\phi} -\mathcal{F}_{\text{ELBO}}(\theta, \psi, \phi)$, provides the following connection to the constrained optimisation problem:

$$\overbrace{\min_{\theta} \min_{\psi}}^{\text{M-step}} \underbrace{\max_{\lambda} \min_{\phi}}_{\text{E-step}} \mathcal{L}(\theta, \psi, \phi; \lambda) \quad \text{s.t.} \quad \lambda \geq 0, \tag{4.6}$$

where, unlike in the original EM algorithm, we want $q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ to additionally satisfy the inequality constraint $\mathcal{C}(\theta, \phi) \leq \kappa^2$ in the E-step. This is achieved by $\max_\lambda \min_\phi \mathcal{L}(\theta, \psi, \phi; \lambda)$, as detailed in Section 3.2. The observation and transition model are optimised by $\min_\theta \mathcal{L}(\theta, \psi, \phi; \lambda)$ and $\min_\psi \mathcal{L}(\theta, \psi, \phi; \lambda)$, respectively, on the basis of $q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$. However, it can only be guaranteed that $\mathcal{L}(\theta, \psi, \phi; \lambda)$ optimises a lower bound on $\log p(\mathbf{x}_{1:T} | \mathbf{u}_{1:T})$ if and only if $1 \geq \lambda \geq 0$, as we have shown Section 3.2.

4.1.2.1 *Learning the Initial Distribution*

It is common practice to define the initial/prior distribution $p(\mathbf{z}_1)$ as standard normal distribution (e.g. Fraccaro et al., 2017; Krishnan et al., 2015). However, in case of a standard normal $p(\mathbf{z}_1)$, the prior KL in the ELBO can cause an over-regularisation of the approximate posterior and thus of the transition model as well. Furthermore, if the discrepancy between prior and posterior is too large, we may obtain a broken generative model, where $p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ is not trained to process samples from $p(\mathbf{z}_1)$. We provide empirical evidence in Section 4.4.1.

This issue often arises in the context of neural models trained by stochastic gradient methods, where batching typically happens by cutting the time-series data into equally-sized short-length units to alleviate possible vanishing-gradient problems. Thus, one can assume that initial-state samples uniformly cover the manifold of the learned

latent representation. As a consequence, the marginal approximate posterior is the optimal initial distribution $p^*(\mathbf{z}_1) = \mathbb{E}_{p_\mathcal{D}(\mathbf{x},\mathbf{u})}\left[q_\phi(\mathbf{z}_1|\,\mathbf{x},\mathbf{u})\right]$ (cf. Tomczak and Welling, 2018)—and an empirical Bayes prior $p_{\psi_0}(\mathbf{z}_1)$ must have the complexity to approximate $p^*(\mathbf{z}_1) \approx p_{\psi_0}(\mathbf{z}_1)$.

For this reason, we propose to learn a hierarchical empirical Bayes prior, $p_{\psi_0}(\mathbf{z}_1) = \int p_{\psi_0}(\mathbf{z}_1|\boldsymbol{\zeta})\, p(\boldsymbol{\zeta})\, \mathrm{d}\boldsymbol{\zeta}$, as part of the DSSM by applying the VHP approach introduced in Section 3.3. The VHP defines, by means of an approximate distribution $q_{\phi_0}(\boldsymbol{\zeta}|\,\mathbf{z}_1)$, a VAE-like lower bound on the optimal empirical Bayes prior:

$$\begin{aligned}\mathbb{E}_{p^*(\mathbf{z}_1)}\left[\log p_{\psi_0}(\mathbf{z}_1)\right] &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\log p_{\psi_0}(\mathbf{z}_1)\right] \\ &\geq \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\mathcal{F}_{\mathrm{VHP}}(\psi_0,\phi_0;\mathbf{z}_1)\right],\end{aligned}$$

where

$$\mathcal{F}_{\mathrm{VHP}}(\psi_0,\phi_0;\mathbf{z}_1) = \mathbb{E}_{q_{\phi_0}(\boldsymbol{\zeta}|\,\mathbf{z}_1)}\left[\log \frac{p_{\psi_0}(\mathbf{z}_1|\,\boldsymbol{\zeta})\, p(\boldsymbol{\zeta})}{q_{\phi_0}(\boldsymbol{\zeta}|\,\mathbf{z}_1)}\right], \tag{4.7}$$

and $p(\boldsymbol{\zeta})$ is defined as standard normal distribution. Note that in contrast to the VAE-based approach in Eq. (3.8) (Chapter 3), we do not use the computationally expensive importance weighting. This is because, in DSSMs, a slightly less expressive empirical Bayes prior does not affect the learned latent representation, as we verify in Section 4.4.1.

The VHP (Eq. (4.7)) defines an upper bound on the rate $\mathcal{R}(\psi,\phi,\psi_0) \mathrel{\widehat{=}} \mathcal{R}(\psi,\phi)$ (cf. Eq. (4.4)), where ψ_0 denotes the now learnable parameters of the prior:

$$\begin{aligned}\mathcal{R}(\psi,\phi,\psi_0) &\leq \mathcal{R}(\psi,\phi,\psi_0,\phi_0) \\ &= \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\Bigg[\log q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T},\mathbf{u}_{1:T}) \\ &\quad - \mathcal{F}_{\mathrm{VHP}}(\psi_0,\phi_0;\mathbf{z}_1) - \sum_{t=2}^{T}\log p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})\Bigg].\end{aligned} \tag{4.8}$$

Learning the VHP as part of the model is consistent with our optimisation problem since the upper bound on $\mathcal{R}(\psi,\phi,\psi_0)$ introduces a lower bound on the sequential ELBO. As a result, we obtain the Lagrangian

$$\mathcal{L}(\theta,\psi,\phi,\psi_0,\phi_0;\lambda) = \mathcal{R}(\psi,\phi,\psi_0,\phi_0) + \lambda\left(\mathcal{C}(\theta,\phi) - \kappa^2\right), \tag{4.9}$$

with the corresponding optimisation problem

$$\underbrace{\min_{\psi_0,\phi_0}}_{\substack{\text{Empirical}\\ \text{Bayes}}}\ \overbrace{\min_{\theta,\psi}}^{\text{M-step}}\ \underbrace{\max_{\lambda}\min_{\phi}}_{\text{E-step}}\ \mathcal{L}(\theta,\psi,\phi,\psi_0,\phi_0;\lambda)\ \text{ s.t. }\ \lambda \geq 0, \tag{4.10}$$

which extends the constrained optimisation problem in Eq. (4.6) by the optimisation of the empirical Bayes prior.

Algorithm 3 REWO for deep state-space models

Initialise $t = 1$
Initialise $\beta_0 = 1$
Initialise INITIALPHASE = TRUE
while training **do**
  Compute $\hat{\mathcal{C}}_{\text{ba}}$ (batch average)
  $\hat{\mathcal{C}}_t = (1-\alpha) \cdot \hat{\mathcal{C}}_{\text{ba}} + \alpha \cdot \hat{\mathcal{C}}_{t-1}, \quad (\hat{\mathcal{C}}_0 = \hat{\mathcal{C}}_{\text{ba}})$
  $\beta_t \leftarrow \beta_{t-1} \cdot \exp\left[\nu \cdot f_\beta\left(\beta_{t-1}, \hat{\mathcal{C}}_t - \kappa^2; \tau_1, \tau_2\right) \cdot \left(\hat{\mathcal{C}}_t - \kappa^2\right)\right]$
  if $\hat{\mathcal{C}}_t \leq \kappa^2$ **then**
    INITIALPHASE = FALSE
  end if
  if INITIALPHASE **then**
    Optimise $\mathcal{L}(\theta, \psi, \phi, \psi_0, \phi_0; \beta_t)$ w.r.t. θ, ϕ
  else
    Optimise $\mathcal{L}(\theta, \psi, \phi, \psi_0, \phi_0; \beta_t)$ w.r.t. $\theta, \psi, \phi, \psi_0, \phi_0$
  end if
  $t \leftarrow t + 1$
end while

4.1.2.2 *Optimisation Algorithm*

In order to find the saddle point of the Lagrangian in Eq. (4.10), we propose Algorithm 3, which is an extension of REWO (Algorithm 2) to deep state-space models. It ensures through a special update scheme for λ that we optimise a lower bound on $\log p(\mathbf{x}_{1:T} | \mathbf{u}_{1:T})$ at the end of training (that is $1 \geq \lambda \geq 0$)—and allows efficiently learning the parameters of the hierarchical empirical Bayes prior (ψ_0, ϕ_0) and the transition model (ψ).

Similar to Chapter 3, we use the β-parametrisation of the Lagrange multiplier, $\beta = {}^1/_\lambda$, to be in line with previous literature (e.g. Higgins et al., 2017; Sønderby et al., 2016).

Algorithm 3 divides the constrained optimisation process into two phases, an initial and a main phase, which explains the use of different parameters for the transition and observation model. In the initial phase, the model is optimised to reduce the reconstruction error in order to learn the features of individual observations $\mathbf{x}_t$. For this purpose, we optimise the bound solely with respect to (θ, ϕ), and β is updated by applying a similar update scheme as introduced in Section 3.3:

$$\beta_t = \beta_{t-1} \cdot \exp\left[\nu \cdot f_\beta\left(\beta_{t-1}, \mathcal{C}_t - \kappa^2; \tau_1, \tau_2\right) \cdot \left(\mathcal{C}_t - \kappa^2\right)\right],$$

where t is the iteration step of the optimisation, and f_β is defined as

$$f_\beta(\beta, \delta; \tau_1, \tau_2) = \big(1 - H(\delta)\big) \cdot \tanh\big(\tau_1 \cdot (\beta - 1)\big) - \tau_2 \cdot H(\delta).$$

H is the Heaviside function, and τ_1 and τ_2 are slope parameters. A visualisation of this update scheme can be found in Figure 3.1a.

The main phase starts as soon as the inequality constraint $\mathcal{C}(\theta, \phi) \leq \kappa^2$ is satisfied. This serves as a starting point for learning

the transition model and the hierarchical empirical Bayes prior, i.e. to learn the system dynamics. To do this, we optimise all parameters $(\theta, \psi, \phi, \psi_0, \phi_0)$ jointly.

4.1.3 *Application to Existing Models*

Our constrained optimisation framework can be applied to any DSSM whose objective function is covered by the general rate–distortion formulation of the ELBO (Eq. 4.5). In Section 4.4, we demonstrate our approach on the smoother versions of deep Kalman filters (DKFs) (Krishnan et al., 2015), deep variational Bayes filters (DVBFs) (Karl et al., 2017), recurrent state-space models (RSSMs) (Hafner et al., 2019), and Kalman variational autoencoders (KVAEs) (Fraccaro et al., 2017). In the following we integrate DKF/DKS and DVBF/DVBS with our constrained optimisation framework by defining the respective distortion $\mathcal{D}(\theta, \phi)$ and rate $\mathcal{R}(\psi, \phi, \psi_0, \phi_0)$, which includes the VHP. This allows us to formulate the Lagrangian of the constrained optimisation problem in Eq. (4.10).

4.1.3.1 *Deep Kalman Filter and Smoother*

ORIGINAL EVIDENCE LOWER BOUND (SMOOTHER VERSION) The objective function introduced by Krishnan et al. (2015) for training deep Kalman smoothers (DKSs) is

$$\mathcal{F}^{\text{DKS}}_{\text{ELBO}}(\theta, \psi, \phi) = -\mathcal{D}_{\text{DKS}}(\theta, \phi) - \mathcal{R}_{\text{DKS}}(\psi, \phi).$$

The distortion is defined by

$$\mathcal{D}_{\text{DKS}}(\theta, \phi) = -\,\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\sum_{t=1}^{T} \log p_\theta(\mathbf{x}_t | \mathbf{z}_t) \right],$$

with the joint approximate posterior factorising as

$$q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = \prod_{t=1}^{T} q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}).$$

Consequently, the rate is given by

$$\begin{aligned}
&\mathcal{R}_{\text{DKS}}(\psi, \phi) \\
&\quad = \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Bigg[\mathrm{KL}\left(q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \,\|\, p(\mathbf{z}_1) \right) \\
&\qquad + \sum_{t=2}^{T} \mathrm{KL}\left(q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \,\|\, p_\psi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \right) \Bigg],
\end{aligned}$$

where $p(\mathbf{z}_1)$ is a standard normal distribution. See (Krishnan et al., 2015) for further implementation details. Note that the filter version (DKF) is obtained by replacing $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ with $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t})$.

VHP-BASED EVIDENCE LOWER BOUND (SMOOTHER VERSION) In the following, we integrate the VHP with DKS:

$$\mathcal{F}_{\text{ELBO}}^{\text{VHP-DKS}}(\theta, \psi, \phi, \psi_0, \phi_0) = -\mathcal{D}_{\text{DKS}}(\theta, \phi) - \mathcal{R}_{\text{VHP-DKS}}(\psi, \phi, \psi_0, \phi_0). \tag{4.11}$$

Note that the distortion remains identical to DKS. By replacing the standard normal prior $p(\mathbf{z}_1)$ in $\mathcal{R}_{\text{DKS}}(\psi, \phi)$ with the VHP defined in Eq. (4.7), we get:

$$\begin{aligned}
&\mathcal{R}_{\text{VHP-DKS}}(\psi, \phi, \psi_0, \phi_0) \\
&= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_\phi(\mathbf{z}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Bigg[\log q_\phi(\mathbf{z}_1 | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) - \mathcal{F}_{\text{VHP}}(\psi_0, \phi_0; \mathbf{z}_1) \\
&\quad + \sum_{t=2}^{T} \text{KL}\left(q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \| \, p_\psi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \right) \Bigg].
\end{aligned}$$

The filter version (VHP-DKF) is obtained, as with DKF, by replacing $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ with $q_\phi(\mathbf{z}_t | \mathbf{x}_{1:t}, \mathbf{u}_{1:t})$.

4.1.3.2 *Deep Variational Bayes Filter and Smoother*

ORIGINAL EVIDENCE LOWER BOUND (SMOOTHER VERSION) The original model was introduced in (Karl et al., 2017). In the following, we refer to the updated version presented in (Karl et al., 2019). The locally-linear transition model is described in Section 4.2.1. The corresponding objective function for training deep variational Bayes smoothers (DVBSs) is

$$\mathcal{F}_{\text{ELBO}}^{\text{DVBS}}(\theta, \psi, \phi, \psi_0, \phi_0) = -\mathcal{D}_{\text{DVBS}}(\theta, \phi, \psi_0, \phi_0) - \mathcal{R}_{\text{DVBS}}(\psi, \phi, \psi_0, \phi_0).$$

The distortion is defined by

$$\begin{aligned}
&\mathcal{D}_{\text{DVBS}}(\theta, \phi, \psi_0, \phi_0) \\
&= -\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_{\phi_0}(\zeta | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_\phi(\mathbf{z}_{2:T} | f_{\psi_0}(\zeta), \mathbf{x}_{2:T}, \mathbf{u}_{1:T})} \Bigg[\log p_\theta(\mathbf{x}_1 | f_{\psi_0}(\zeta)) \\
&\quad + \sum_{t=2}^{T} \log p_\theta(\mathbf{x}_t | \mathbf{z}_t) \Bigg],
\end{aligned}$$

where $f_{\psi_0}(\zeta) \mathrel{\widehat{=}} \mathbf{z}_1$ mimics an empirical Bayes prior that is learned from data, and the joint approximate posterior factorises as

$$\begin{aligned}
&q_\phi(\mathbf{z}_{2:T} | f_{\psi_0}(\zeta), \mathbf{x}_{2:T}, \mathbf{u}_{1:T}) \\
&\quad = q_\phi(\mathbf{z}_2 | f_{\psi_0}(\zeta), \mathbf{x}_{2:T}, \mathbf{u}_{1:T}) \prod_{t=3}^{T} q_\phi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}).
\end{aligned}$$

Therefore, the rate is given by

$$\begin{aligned}
&\mathcal{R}_{\text{DVBS}}(\phi, \psi, \psi_0, \phi_0) \\
&= \mathbb{E}_{p_{\mathcal{D}}} \mathbb{E}_{q_{\phi_0}(\zeta|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_\phi(\mathbf{z}_{2:T}|\, f_{\psi_0}(\zeta), \mathbf{x}_{2:T}, \mathbf{u}_{1:T})} \Bigg[\mathrm{KL}\big(q_{\phi_0}(\zeta|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \| \, p(\zeta)\big) \\
&\quad + \mathrm{KL}\left(q_\phi(\mathbf{z}_2|\, f_{\psi_0}(\zeta), \mathbf{x}_{2:T}, \mathbf{u}_{1:T}) \| \, p_\psi(\mathbf{z}_2|\, f_{\psi_0}(\zeta), \mathbf{u}_1)\right) \\
&\quad + \sum_{t=3}^{T} \mathrm{KL}\left(q_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}) \| \, p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})\right) \Bigg],
\end{aligned}$$

where $p(\zeta)$ is a standard normal distribution.

The conditional approximate posterior is implemented as the product of two distributions (Karl et al., 2019):

$$q_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}) \propto p_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \times q_\phi(\mathbf{z}_t|\,\mathbf{x}_{t:T}, \mathbf{u}_{t:T}). \tag{4.12}$$

Further implementation details can be found in (Karl et al., 2017) and (Karl et al., 2019). Note that the filter version (DVBF) is obtained by replacing $q_\phi(\mathbf{z}_t|\,\mathbf{x}_{t:T}, \mathbf{u}_{t:T})$ in Eq. (4.12) with $q_\phi(\mathbf{z}_t|\,\mathbf{x}_t)$.

VHP-BASED EVIDENCE LOWER BOUND (SMOOTHER VERSION) In the following, we integrate the VHP with DVBS:

$$\begin{aligned}
&\mathcal{F}_{\text{ELBO}}^{\text{VHP-DVBS}}(\theta, \psi, \phi, \psi_0, \phi_0) \\
&\quad = -\mathcal{D}_{\text{VHP-DVBS}}(\theta, \phi) - \mathcal{R}_{\text{VHP-DVBS}}(\psi, \phi, \psi_0, \phi_0).
\end{aligned}$$

By replacing the deterministic transformation $f_{\psi_0}(\zeta)$ with the VHP defined in Eq. (4.7), the marginal approximate posterior simplifies to $q_\phi(\mathbf{z}_t|\,\mathbf{x}_{t:T}, \mathbf{u}_{t:T})$ for all time steps *including the initial time step*. As a result, the joint approximate posterior factorises as

$$q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = q_\phi(\mathbf{z}_1|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \prod_{t=2}^{T} q_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}).$$

Thus, the distortion is given by

$$\begin{aligned}
&\mathcal{D}_{\text{VHP-DVBS}}(\theta, \phi) \\
&\quad = -\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\sum_{t=1}^{T} \log p_\theta(\mathbf{x}_t|\,\mathbf{z}_t) \right],
\end{aligned}$$

and the rate is defined as

$$\begin{aligned}&\mathcal{R}_{\text{VHP-DVBS}}(\psi, \phi, \psi_0, \phi_0)\\ &= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{q_\phi(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\Bigg[\log q_\phi(\mathbf{z}_1|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) - \mathcal{F}_{\text{VHP}}(\psi_0, \phi_0;\, \mathbf{z}_1)\\ &\quad - \sum_{t=2}^{T} \mathrm{KL}\left(q_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T}) \,\|\, p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})\right)\Bigg].\end{aligned}$$

The conditional approximate posterior $q_\phi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t-1:T})$ is implemented as for DVBS, see Eq. (4.12). The filter version (VHP-DVBF) is obtained, as with DVBF, by replacing $q_\phi(\mathbf{z}_t|\,\mathbf{x}_{t:T}, \mathbf{u}_{t:T})$ in Eq. (4.12) with $q_\phi(\mathbf{z}_t|\,\mathbf{x}_t)$.

4.2 EXTENDED KALMAN VARIATIONAL AUTOENCODER

The constrained optimisation framework facilitates system identification, as we show in our experiments. However, to achieve high prediction accuracies, the model itself should not prove to be a limiting factor in learning a precise description of the dynamic system.

The amortised variational inference framework requires us to introduce an approximate posterior distribution $q(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$. The optimal $q(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ is the posterior distribution $p(\mathbf{z}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$, which can be computed analytically through Bayesian filtering or smoothing for simple models as linear Gaussian systems. A detailed introduction can be found in Section 2.3.2. In neural models it is therefore common to learn the parameters of the approximate posterior through an $\text{RNN}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})$, which is expected to replace classic Bayesian filtering/smoothing, as discussed in Section 2.3.3. However, this can result in learning a less accurate model of the dynamic system, as we show in Section 4.4.2.

To address this issue, Fraccaro et al. (2017) use an auxiliary-variable model in the KVAE:

$$p(\mathbf{x}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_{1:T}|\,\mathbf{u}_{1:T}) = p(\mathbf{x}_{1:T}|\,\mathbf{a}_{1:T})\, p(\mathbf{a}_{1:T}|\,\mathbf{z}_{1:T})\, p(\mathbf{z}_{1:T}|\,\mathbf{u}_{1:T}).$$

It is based on linear Gaussian $p(\mathbf{a}_t|\,\mathbf{z}_t, \mathbf{h}_{t-1})$ and $p(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{h}_{t-1}, \mathbf{u}_{t-1})$, whose model parameters are conditioned on a deterministic hidden state $\mathbf{h}_{t-1} = \text{LSTM}(\mathbf{a}_{1:t-1})$ for modelling nonlinear dynamic systems. This allows analytically computing $p(\mathbf{z}_{1:T}|\mathbf{a}_{1:T}, \mathbf{u}_{1:T})$ by Kalman filtering or smoothing (cf. Section 2.3.2.2); and leads to the posterior approximation

$$q(\mathbf{z}_{1:T}, \mathbf{a}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = p(\mathbf{z}_{1:T}|\,\mathbf{a}_{1:T}, \mathbf{u}_{1:T})\, q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T}).$$

However, as a consequence of the LSTM-based deterministic path in the transition model, the KVAE learns a non-Markovian state representation in $\mathbf{z}$, as we verify in Section 4.4.3.

In order to learn a Markovian state-space—i.e. the state is fully represented by $\mathbf{z}_t$—we introduce the extended Kalman variational autoencoder (EKVAE). We dispense with the LSTM-based deterministic path by using a transition model with a nonlinear dependence on $\mathbf{z}_{t-1}$. To compute the posterior, we leverage the concept of extended Kalman filtering/smoothing (see Section 2.3.2.3) but avoid the computational expensive linearisation (first-order Taylor expansion) of the transition and observation model. This is achieved (i) by learning the Jacobian of the dynamic system as a function of the current state, which we refer to as neural linearisation (Section 4.2.1); and (ii) by introducing an auxiliary-variable model similar to the KVAE (Section 4.2.2). The EKVAE can be used as filter or smoother. In the following, we focus on the more complex **smoother version** and refer to Section 4.2.5 for the filter version.

4.2.1 *Neural Linearisation of the Dynamic Model*

We model the (unknown) nonlinear dynamic system by a Gaussian transition model that is locally linear with respect to discrete time steps (Karl et al., 2017; Watter et al., 2015):

$$\begin{aligned} &p_\psi(\mathbf{z}_{t+1}|\,\mathbf{z}_t, \mathbf{u}_t) \\ &\quad = \mathcal{N}\big(\mathbf{z}_{t+1}|\,\mathbf{F}_\psi(\mathbf{z}_t, \mathbf{u}_t)\,\mathbf{z}_t + \mathbf{B}_\psi(\mathbf{z}_t, \mathbf{u}_t)\,\mathbf{u}_t,\, \mathbf{Q}_\psi(\mathbf{z}_t, \mathbf{u}_t)\big), \end{aligned} \tag{4.13}$$

where $\mathbf{F}_\psi$, $\mathbf{B}_\psi$, and $\mathbf{Q}_\psi$ are modelled by linear combinations of M base matrices, for example:

$$\mathbf{F}_\psi(\mathbf{z}_t, \mathbf{u}_t) = \sum_{m=1}^{M} \alpha_\psi^{(m)}(\mathbf{z}_t, \mathbf{u}_t)\,\mathbf{F}^{(m)}$$

weighted by

$$\boldsymbol{\alpha}_\psi(\mathbf{z}_t, \mathbf{u}_t) = \mathrm{softmax}\big(\mathbf{g}_\psi(\mathbf{z}_t, \mathbf{u}_t)\big) \in \mathbb{R}^M. \tag{4.14}$$

The base matrices $\big\{\mathbf{F}^{(m)}, \mathbf{B}^{(m)}, \mathbf{Q}^{(m)}\big\}_{m=1}^{M}$ are learned parameters, and $\mathbf{g}_\psi(\mathbf{z}_t, \mathbf{u}_t)$ is implemented as a neural network.

Next, we make the connection to extended Kalman filtering/smoothing, where the *prediction step* is based on the local Jacobian (first-order Taylor expansion) of the nonlinear dynamic model function $\mathbf{z}_{t+1} = \mathbf{f}(\mathbf{z}_t, \mathbf{u}_t) + \mathbf{q}_t$, which is *unknown* in our case. Our transition model in Eq. (4.13) learns to represent the local Jacobian as a function of the current state (Watter et al., 2015), for example:

$$\mathbf{F}_\psi(\mathbf{z}_t, \mathbf{u}_t) \mathrel{\widehat{=}} \left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{z}}\right|_{\mathbf{z}_t, \mathbf{u}_t}, \tag{4.15}$$

which we refer to as neural linearisation approach This allows us to apply the extended Kalman filter or smoother algorithm but avoid the computationally expensive linearisation (first-order Taylor expansion) in the *prediction step*, as we derive in Section 4.2.3.

4.2.2 *Linear Auxiliary-Variable Model*

The observation model often needs to learn highly nonlinear functions, especially in case of high-dimensional sensory data, such as images. In order to enable an analytic computation of the posterior but avoid an expensive linearisation of the observation model, we introduce auxiliary variables $\mathbf{a}_{1:T}$ with a linear dependence on $\mathbf{z}_{1:T}$. As in (Fraccaro et al., 2017), the nonlinear mapping from $\mathbf{a}_{1:T}$ to the high-dimensional observations $\mathbf{x}_{1:T}$ is learned by a VAE's encoder–decoder pair, $q_\phi(\mathbf{a}_t|\,\mathbf{x}_t)$ and $p_\theta(\mathbf{x}_t|\,\mathbf{a}_t)$.

Since the entire dynamics are modelled by the transition model in $\mathbf{z}_{1:T}$, and $\mathbf{a}_t$ can be viewed as a low-dimensional representation of $\mathbf{x}_t$, we obtain the observation model:

$$p(\mathbf{x}_{1:T}, \mathbf{a}_{1:T}|\,\mathbf{z}_{1:T}) = \prod_{t=1}^{T} p_\theta(\mathbf{x}_t|\,\mathbf{a}_t)\, p_\psi(\mathbf{a}_t|\,\mathbf{z}_t). \tag{4.16}$$

In contrast to Fraccaro et al. (2017), we propose a time-invariant (no conditioning on $\mathbf{h}_t$) auxiliary-variable model,

$$p_\psi(\mathbf{a}_t|\,\mathbf{z}_t) = \mathcal{N}(\mathbf{a}_t|\,\mathbf{H}\,\mathbf{z}_t,\, \mathbf{R}), \tag{4.17}$$

where $\mathbf{H}$ and $\mathbf{R}$ are globally learned or predefined. The time-invariant $\mathbf{H}$ additionally allows us to learn state-space representations where static and dynamic features are disentangled, as we discuss in Section 4.3. By using $\mathbf{a}_{1:T}$ as pseudo observations, our *update step* corresponds to the classical Kalman filter/smoother algorithm because we do not need to linearise the observation model (see Section 4.2.3). In combination with the neural linearisation approach, we can now compute the filtered and smoothed distributions, $p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:t}, \mathbf{u}_{1:t-1})$ and $p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})$, respectively. Note, however, that these are generally not optimal because we have a nonlinear Gaussian system that is locally linearised. As a result, we can define the recognition model (**smoother version**) as follows:

$$q(\mathbf{z}_{1:T}, \mathbf{a}_{1:T}|\,\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = \prod_{t=1}^{T} p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:T}, \mathbf{u}_{1:T-1}) \prod_{t=1}^{T} q_\phi(\mathbf{a}_t|\,\mathbf{x}_t). \tag{4.18}$$

4.2.3 *Connection to Extended Kalman Filtering and Smoothing*

In Section 2.3.2.3, we have introduced the extended Kalman filter and smoother. In the following, we provide an analysis of how extended Kalman filtering/smoothing is applied in combination with the neural linearisation-based transition model defined in Eq. (4.13) and the auxiliary-variable model defined in Eq. (4.17). To this end, we first consider the *prediction step* that allows analytically computing

$$p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-1}) = \mathcal{N}(\mathbf{z}_t|\,\mathbf{m}_t^-, \mathbf{P}_t^-),$$

given the filtered distribution

$$p_\psi(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-1}) = \mathcal{N}(\mathbf{z}_{t-1}|\,\mathbf{m}_{t-1}, \mathbf{P}_{t-1}),$$

where $\mathbf{m}$ refers to the mean and $\mathbf{P}$ to the covariance of a Gaussian distribution.

The nonlinear dynamic model is typically defined as

$$\mathbf{z}_t = \mathbf{f}(\mathbf{z}_{t-1}, \mathbf{u}_{t-1}) + \mathbf{q}_{t-1}.$$

In extended Kalman filtering/smoothing, the dynamic model function $\mathbf{f}(\mathbf{z}, \mathbf{u})$ is locally linearised by means of a first-order Taylor expansion, which allows applying the Kalman filter/smoother algorithm, as we describe in the following. The prediction step is defined by:

$$\begin{aligned}
\mathbf{m}_t^- &= \mathbf{f}(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}),\\
\mathbf{F}_{t-1} &= \left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{z}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}},\\
\mathbf{P}_t^- &= \mathbf{F}_{t-1}\,\mathbf{P}_{t-1}\,\mathbf{F}_{t-1}^{\mathrm{T}} + \mathbf{Q}_{t-1}.
\end{aligned}$$

In case of an unknown dynamic model function, we can approximate $\mathbf{f}(\mathbf{z}, \mathbf{u})$ by a function that is locally linear with respect to discrete time steps. This allows formulating the prediction step of the mean as

$$\mathbf{m}_t^- = \left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{z}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}} \cdot \mathbf{m}_{t-1} + \left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}} \cdot \mathbf{u}_{t-1}.$$

In our proposed transition model (Eq. (4.13)), the above Jacobians are modelled by

$$\left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{z}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}} \approx \mathbf{F}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}), \tag{4.19}$$

$$\left.\frac{\partial \mathbf{f}(\mathbf{z}, \mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{m}_{t-1}, \mathbf{u}_{t-1}} \approx \mathbf{B}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}). \tag{4.20}$$

Eq. (4.19) and (4.20) allow defining the prediction step of the extended Kalman filter/smoother as

$$\mathbf{m}_t^- = \mathbf{F}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}) \cdot \mathbf{m}_{t-1} + \mathbf{B}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}) \cdot \mathbf{u}_{t-1}, \tag{4.21}$$

$$\mathbf{F}_{t-1} = \mathbf{F}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}), \tag{4.22}$$

$$\begin{aligned}
\mathbf{Q}_{t-1} &= \mathbf{Q}_\psi(\mathbf{m}_{t-1}, \mathbf{u}_{t-1}),\\
\mathbf{P}_t^- &= \mathbf{F}_{t-1}\,\mathbf{P}_{t-1}\,\mathbf{F}_{t-1}^{\mathrm{T}} + \mathbf{Q}_{t-1}.
\end{aligned} \tag{4.23}$$

The *update step* corresponds to the classical Kalman filter/smoother due to the linear Gaussian $p_\psi(\mathbf{a}_t|\,\mathbf{z}_t)$ (Eq. (4.17)). The *backward recursion* is defined by $\mathbf{m}_t^-$, $\mathbf{F}_{t-1}$, and $\mathbf{P}_t^-$ in Eqs. (4.21–4.23). Therefore, it is identical to the Kalman smoother. Details regarding the Kalman filtering/smoothing algorithm can be found in Section 2.3.2.2.

4.2.4 *Integration With the Constrained Optimisation Framework*

To integrate the extended Kalman variational autoencoder (EKVAE) with the constrained optimisation framework introduced in Section 4.1, we define in the following the distortion $\mathcal{D}(\theta, \phi)$ and rate $\mathcal{R}(\psi, \phi, \psi_0, \phi_0)$ based on the transition, observation, and recognition model in Eqs. (4.13, 4.16, 4.18)—and the VHP in Eq. (4.7). A detailed derivation (filter and smoother version) can be found in Section 4.2.5.

The distortion of the EKVAE is simply defined by the encoder–decoder pair:

$$\mathcal{D}(\theta, \phi) = -\,\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \left[\sum_{t=1}^{T} \mathbb{E}_{q_\phi(\mathbf{a}_t | \mathbf{x}_t)} \left[\log p_\theta(\mathbf{x}_t | \mathbf{a}_t) \right] \right]. \tag{4.24}$$

Deriving the rate is more complicated: (i) we need to perform a sample-based optimisation of transition parameters ψ. This is especially crucial for $g_\psi(\mathbf{z}_t, \mathbf{u}_t)$ in Eq. (4.14), where an optimisation solely via extended Kalman smoothing, i.e. via deterministic mean values (cf. Section 4.2.3), does not cover the range of application and would therefore cause a poorly trained transition model. (ii) Our recognition model $q(\mathbf{z}_{1:T}, \mathbf{a}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$ (cf. Eq. (4.18)) does *not* contain the computationally more expensive pairwise smoothed distributions $p_\psi(\mathbf{z}_t, \mathbf{z}_{t-1} | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})$ but only smoothed distributions $p_\psi(\mathbf{z}_t | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})$. However, an optimisation of $p_\psi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ based on samples $(\mathbf{z}_t, \mathbf{z}_{t-1})$ from smoothed distributions would lead to an inaccurate transition model.

To address these issues, we use the rate

$$\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{q_\phi(\mathbf{a}_{1:T} | \mathbf{x}_{1:T})} \Big[\sum_{t=1}^{T} \log \frac{q_\phi(\mathbf{a}_t | \mathbf{x}_t)}{p_\psi(\mathbf{a}_t | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-1})} \Big] \tag{4.25}$$

as our starting point. But instead of computing $p_\psi(\mathbf{a}_t | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-1})$ analytically, we solve the corresponding integral (derived from the Bayesian filtering equations, see Section. 4.2.5.2) only with respect to $\mathbf{z}_t$ in closed form and marginalise $\mathbf{z}_{t-1}$ via Monte Carlo integration:

$$\begin{aligned}
&\log p_\psi(\mathbf{a}_t | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-1}) \\
&\quad = \log \iint p_\psi(\mathbf{a}_t | \mathbf{z}_t)\, p_\psi(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})\, p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-2})\, \mathrm{d}\mathbf{z}_t\, \mathrm{d}\mathbf{z}_{t-1} \\
&\quad \geq \mathbb{E}_{p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})} \left[\log \frac{p_\psi(\mathbf{a}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})\, p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-2})}{p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})} \right].
\end{aligned} \tag{4.26}$$

In this context, the distribution $p_\psi(\mathbf{a}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ plays a crucial role as it includes all transition parameters ψ and decouples $\mathbf{z}_t$ from $\mathbf{z}_{t-1}$. It therefore allows a sample-based optimisation of the transition model on the basis of the smoothed distribution $p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})$.

By combining Eq. (4.25) with Eq. (4.26)—the complete derivation can be found in Section. 4.2.5.2—we obtain the following rate (**smoother version**):

$$\begin{aligned}
&\mathcal{R}(\psi, \phi, \psi_0, \phi_0) \\
&= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \mathbb{E}_{q(\mathbf{z}_{1:T}, \mathbf{a}_{1:T} | \mathbf{x}_{1:T}, \mathbf{u}_{1:T})} \Bigg[\mathcal{R}_{\text{initial}}(\psi, \phi, \psi_0, \phi_0; \mathbf{z}_1, \mathbf{a}_{1:T}, \mathbf{x}_1, \mathbf{u}_{1:T-1}) \\
&\quad + \sum_{t=2}^{T} \log \frac{q_\phi(\mathbf{a}_t | \mathbf{x}_t)\, p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})}{p_\psi(\mathbf{a}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) p_\psi(\mathbf{z}_{t-1} | \mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-2})} \Bigg],
\end{aligned} \tag{4.27}$$

where the VHP introduced in Section 4.1.2.1 is learned via

$$\begin{aligned}
&\mathcal{R}_{\text{initial}}(\psi, \phi, \psi_0, \phi_0; \mathbf{z}_1, \mathbf{a}_{1:T}, \mathbf{x}_1, \mathbf{u}_{1:T-1}) \\
&\quad = \log \frac{q_\phi(\mathbf{a}_1 | \mathbf{x}_1)}{p_\psi(\mathbf{a}_1 | \mathbf{z}_1)} + \log p_\psi(\mathbf{z}_1 | \mathbf{a}_{1:T}, \mathbf{u}_{1:T-1}) - \mathcal{F}_{\text{VHP}}(\psi_0, \phi_0; \mathbf{z}_1).
\end{aligned} \tag{4.28}$$

Eq. (4.24) and (4.27) allow us to define the Lagrangian of the constrained optimisation problem defined Eq. (4.10) and thus to integrate the EKVAE with our constrained optimisation framework.

4.2.5 *Derivation of the Extended Kalman Variational Autoencoder*

In the following, we derive $\mathcal{F}_{\text{ELBO}}$ of the EKVAE, i.e. the distortion $\mathcal{D}(\theta, \phi)$ and rate $\mathcal{R}(\psi, \phi, \psi_0, \phi_0)$ in Eq. (4.24) and (4.27). To this end, we start with the generative model that defines $p(\mathbf{x}_{1:T} | \mathbf{u}_{1:T})$.

4.2.5.1 *Generative Model*

In addition to the latent variables $\mathbf{z}_t$, we use the auxiliary variables $\mathbf{a}_t$ to facilitate extended Kalman filtering/smoothing and ζ to model the empirical Bayes prior:

$$\begin{aligned}
&p(\mathbf{x}_{1:T} | \mathbf{u}_{1:T}) \\
&\quad = \iiint p(\mathbf{x}_{1:T}, \mathbf{a}_{1:T}, \mathbf{z}_{1:T}, \zeta | \mathbf{u}_{1:T})\, \mathrm{d}\mathbf{a}_{1:T}\, \mathrm{d}\mathbf{z}_{1:T}\, \mathrm{d}\zeta \\
&\quad = \iiint p(\mathbf{x}_{1:T} | \mathbf{a}_{1:T}, \cancel{\mathbf{z}_{1:T}}, \cancel{\zeta}, \cancel{\mathbf{u}_{1:T}})\, p(\mathbf{a}_{1:T} | \mathbf{z}_{1:T}, \cancel{\zeta}, \cancel{\mathbf{u}_{1:T}}) \\
&\qquad\quad p(\mathbf{z}_{1:T} | \zeta, \mathbf{u}_{1:T-1})\, p(\zeta)\, \mathrm{d}\mathbf{a}_{1:T}\, \mathrm{d}\mathbf{z}_{1:T}\, \mathrm{d}\zeta \\
&\quad = \iiint p(\mathbf{x}_{1:T} | \mathbf{a}_{1:T})\, p(\mathbf{a}_{1:T} | \mathbf{z}_{1:T})\, p(\mathbf{z}_{1:T} | \zeta, \mathbf{u}_{1:T}) \\
&\qquad\quad p(\zeta)\, \mathrm{d}\mathbf{a}_{1:T}\, \mathrm{d}\mathbf{z}_{1:T}\, \mathrm{d}\zeta \\
&\quad = \iiint \prod_{t=1}^{T} \Big(p_\theta(\mathbf{x}_t | \mathbf{a}_t)\, p_\psi(\mathbf{a}_t | \mathbf{z}_t) \Big) \prod_{t=2}^{T} \Big(p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) \Big) \\
&\qquad\quad p(\mathbf{z}_1 | \zeta)\, p(\zeta)\, \mathrm{d}\mathbf{a}_{1:T}\, \mathrm{d}\mathbf{z}_{1:T}\, \mathrm{d}\zeta.
\end{aligned} \tag{4.29}$$

4.2.5.2 *Evidence Lower Bound (Smoother Version)*

$$
\begin{aligned}
&\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\left[\log p(\mathbf{x}_{1:T}|\,\mathbf{u}_{1:T})\right] \\
&\quad\geq \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\log\frac{p(\mathbf{x}_{1:T}|\,\mathbf{a}_{1:T})}{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\right. \\
&\qquad\left.+\log\iint\prod_{t=2}^{T}\Big(p(\mathbf{a}_t|\,\mathbf{z}_t)\,p(\mathbf{z}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})\Big)\,p(\mathbf{a}_1|\,\mathbf{z}_1)\,p(\mathbf{z}_1|\,\zeta)\,p(\zeta)\,\mathrm{d}\mathbf{z}_{1:T}\,\mathrm{d}\zeta\right] \\
&\quad= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\log\frac{p(\mathbf{x}_{1:T}|\,\mathbf{a}_{1:T})}{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}+\log\iint p(\mathbf{a}_1|\,\mathbf{z}_1)\,p(\mathbf{z}_1|\,\zeta)\,p(\zeta)\,\mathrm{d}\mathbf{z}_1\,\mathrm{d}\zeta\right. \\
&\qquad\left.+\sum_{t=2}^{T}\log\iint\underbrace{p(\mathbf{a}_t|\,\mathbf{z}_t)}_{\text{observation}}\overbrace{p(\mathbf{z}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})}^{\text{transition}}\underbrace{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1},\mathbf{u}_{1:t-2})}_{\text{filtered distribution}}\,\mathrm{d}\mathbf{z}_t\,\mathrm{d}\mathbf{z}_{t-1}\right] \qquad (4.30) \\
&\quad= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\log\frac{p(\mathbf{x}_{1:T}|\,\mathbf{a}_{1:T})}{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}+\log\iint p(\mathbf{a}_1|\,\mathbf{z}_1)\,p(\mathbf{z}_1|\,\zeta)\,p(\zeta)\,\mathrm{d}\mathbf{z}_1\,\mathrm{d}\zeta\right. \\
&\qquad\left.+\sum_{t=2}^{T}\log\int\overbrace{p(\mathbf{a}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})}^{\int p(\mathbf{a}_t|\,\mathbf{z}_t)\,p(\mathbf{z}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})\,\mathrm{d}\mathbf{z}_t}\;p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1},\mathbf{u}_{1:t-2})\,\mathrm{d}\mathbf{z}_{t-1}\right] \\
&\quad\geq \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\log\frac{p(\mathbf{x}_{1:T}|\,\mathbf{a}_{1:T})}{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\right. \\
&\qquad+\mathbb{E}_{p(\mathbf{z}_1|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}\left[\log p(\mathbf{a}_1|\,\mathbf{z}_1)-\log p(\mathbf{z}_1|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})+\overbrace{\mathbb{E}_{q(\zeta|\mathbf{z}_1)}\left[\log\frac{p(\mathbf{z}_1|\,\zeta)\,p(\zeta)}{q(\zeta|\,\mathbf{z}_1)}\right]}^{\mathcal{F}_{\text{VHP}}(\psi_0,\phi_0;\mathbf{z}_1)\text{ in Eq. (4.28)}}\right] \\
&\qquad\left.+\sum_{t=2}^{T}\mathbb{E}_{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}\left[\log p(\mathbf{a}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})+\log\underbrace{\frac{\overbrace{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1},\mathbf{u}_{1:t-2})}^{\text{filtered distribution}}}{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}}_{\text{smoothed distribution}}\right]\right] \\
&\quad= \underbrace{\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x}_{1:T},\mathbf{u}_{1:T})}\,\mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\sum_{t=1}^{T}\log p(\mathbf{x}_t|\,\mathbf{a}_t)\right]}_{-\mathcal{D}(\theta,\phi)\text{ in Eq. (4.24)}} \\
&\qquad-\mathbb{E}_{p(\mathbf{z}_1|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}\underbrace{\mathbb{E}_{q(\zeta|\,\mathbf{z}_1)}\left[\log\frac{q(\mathbf{a}_1|\,\mathbf{x}_1)}{p(\mathbf{a}_1|\,\mathbf{z}_1)}+\log\frac{p(\mathbf{z}_1|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}{p(\mathbf{z}_1|\,\zeta)}+\log\frac{q(\zeta|\,\mathbf{z}_1)}{p(\zeta)}\right]}_{\mathcal{R}_{\text{initial}}(\psi,\phi,\psi_0,\phi_0;\mathbf{z}_1,\mathbf{a}_{1:T},\mathbf{x}_1,\mathbf{u}_{1:T-1})\text{ in Eq. (4.27)}} \\
&\qquad\left.-\sum_{t=2}^{T}\mathbb{E}_{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}\left[\log\frac{q(\mathbf{a}_t|\,\mathbf{x}_t)}{p(\mathbf{a}_t|\,\mathbf{z}_{t-1},\mathbf{u}_{t-1})}+\log\frac{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:T},\mathbf{u}_{1:T-1})}{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1},\mathbf{u}_{1:t-2})}\right]\right]
\end{aligned}
$$

4.2.5.3 *Evidence Lower Bound (Filter Version)*

The filter version of the EKVAE corresponds to the smoother version with the difference of replacing $p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})$ by $p_\psi(\mathbf{z}_t|\,\mathbf{a}_{1:t}, \mathbf{u}_{1:t-1})$. In contrast to a closed-form evaluation, this enables a sample-based optimisation of the transition-model parameters ψ, as discussed in Section 4.2.4. Consequently, we obtain:

$$
\begin{aligned}
&\mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\left[\log p(\mathbf{x}_{1:T}|\,\mathbf{u}_{1:T})\right] \\
&\geq \mathbb{E}_{p_\mathcal{D}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T})}\, \mathbb{E}_{q(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})}\left[\sum_{t=1}^{T} \log p(\mathbf{x}_t|\,\mathbf{a}_t)\right. \\
&\quad - \mathbb{E}_{p(\mathbf{z}_1|\,\mathbf{a}_1)}\, \mathbb{E}_{q(\zeta|\,\mathbf{z}_1)}\left[\log \frac{q(\mathbf{a}_1|\,\mathbf{x}_1)}{p(\mathbf{a}_1|\,\mathbf{z}_1)} + \log \frac{p(\mathbf{z}_1|\,\mathbf{a}_1)}{p(\mathbf{z}_1|\,\zeta)} + \log \frac{q(\zeta|\,\mathbf{z}_1)}{p(\zeta)}\right] \\
&\quad \left. - \sum_{t=2}^{T} \mathbb{E}_{p(\mathbf{z}_{t-1}|\,\mathbf{a}_{1:t-1}, \mathbf{u}_{1:t-2})}\left[\log \frac{q(\mathbf{a}_t|\,\mathbf{x}_t)}{p(\mathbf{a}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})}\right]\right].
\end{aligned}
$$

4.3 DISENTANGLING STATIC AND DYNAMIC FEATURES FOR POLICY LEARNING

Learning DSSMs—in the context of model-based reinforcement learning and model predictive control—that are accurate enough for planning is subject of current research (e.g. Hafner et al., 2019). Furthermore, substantial engineering effort has to be invested into the design of cost/reward functions. This is because reward functions based on observable data—for instance, the Euclidean distance to a desired position of a robot arm—do not take into account the dynamics and constraints of a system.

In Sections 4.1 and 4.2, we have introduced the constrained optimisation framework and the EKVAE for learning accurate models of dynamic systems, where the dynamics as well as the constraints of the system are encoded in the transition model $p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$. To validate the learned transition model, we propose a method for defining reward functions by means of the latent representation in order to learn policies exclusively through $p_\psi(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$.

4.3.1 *Disentangling Static and Dynamic Features*

Disentanglement, as typically employed in literature, refers to independence among features in a representation (e.g. Bengio et al., 2013). In the context of latent-variable models, the aim is to represent different features by different dimensions in latent space (e.g. Higgins et al., 2017). In order to predict dynamic systems, DSSMs learn a representation of the system's state in the latent space. The state can usually be split into *static* and *dynamic* features, for example, the position and

velocity of a moving robot arm, where the position can be inferred from a single frame, while inferring the velocity requires a sequence of frames.

The EKVAE is capable of disentangling static and dynamic features in the latent space due to its architecture: static features are represented separately by the auxiliary variables $\mathbf{a}_t$, which are learned via the encoder–decoder pair. By defining $\mathbf{H}$ in $p_\psi(\mathbf{a}_t|\,\mathbf{z}_t)$ (Eq. (4.17)) as rectangular identity matrix (cf. Figure 4.19),

$$\mathbf{H} = \left(\delta_{ij}\right) \in \mathbb{R}^{D_\mathbf{a} \times D_\mathbf{z}}, \tag{4.31}$$

the model learns a state-space representation where the first $D_\mathbf{a}$ dimensions of $\mathbf{z}_t$ correspond to static features $\mathbf{a}_t \in \mathbb{R}^{D_\mathbf{a}}$, such that $z_t^{(d)} = a_t^{(d)}$ for $d = 1, 2, \ldots, D_\mathbf{a}$. The remaining $D_\mathbf{z} - D_\mathbf{a}$ dimensions of $\mathbf{z}_t$ represent dynamic features, as we demonstrate in Sections 4.4.2 and 4.4.4.

4.3.2 *Encoding the Reward Function*

In the context of model-based reinforcement learning, the state of a dynamic system often consists of position and velocity. For this reason, we discuss on the example of disentangled position–velocity representations how such state-space representations can be used to learn policies $\pi_\omega(\mathbf{u}_t|\,\mathbf{z}_t)$ that enable a dynamic system to reach a goal state. This can be, for example, a certain position of a robot arm or a certain angular velocity at which a pendulum should rotate.

The EKVAE allows us to use an observation for encoding a goal position $\mathbf{p}_g = \mathbf{a}$ directly through $q_\phi(\mathbf{a}|\,\mathbf{x})$—or a goal velocity $\mathbf{v}_g$ through $p_\psi(\mathbf{z}|\,\mathbf{a}_{1:T}, \mathbf{u}_{1:T-1})\, q_\phi(\mathbf{a}_{1:T}|\,\mathbf{x}_{1:T})$, where $\mathbf{v}_g \in \mathbb{R}^{D_\mathbf{v}}$ is represented by the last $D_\mathbf{v} = D_\mathbf{z} - D_\mathbf{a}$ dimensions of $\mathbf{z}$. As a result, we can define two independent reward functions—based on an encoded $\mathbf{p}_g$ or $\mathbf{v}_g$—that target dimensions in $\mathbf{z}_t$ either representing the position or the velocity:

$$r_t^{\text{pos}}(\mathbf{z}_t, \mathbf{p}_g) = -\sum_{d=1}^{D_\mathbf{a}} \left(z_t^{(d)} - p_g^{(d)}\right)^2, \tag{4.32}$$

$$r_t^{\text{vel}}(\mathbf{z}_t, \mathbf{v}_g) = -\sum_{d=1}^{D_\mathbf{v}} \left(z_t^{(D_\mathbf{a}+d)} - v_g^{(d)}\right)^2, \tag{4.33}$$

where the negative mean squared error is a natural choice in the context of state-space representations motivated by the Euclidean distance metric (cf. Section 3.4). Depending on the task, we can use the reward function in *either* Eq. (4.32) *or* Eq. (4.33) to learn a policy by means of

$$J(\omega, \psi) = \sum_{t=1}^{H-1} \mathbb{E}_{\pi_\omega(\mathbf{u}_t|\,\mathbf{z}_t)}\, \mathbb{E}_{p_\psi(\mathbf{z}_{t+1}|\,\mathbf{z}_t, \mathbf{u}_t)} \left[r_{t+1}\right], \tag{4.34}$$

where $J(\omega, \psi)$ is maximised with respect to ω, and H is the planning horizon.

4.4 EXPERIMENTAL RESULTS

We validate our approach on image data of a moving pendulum and on the reacher environment of Deepmind's control suite (Tassa et al., 2018), where we use angle as well as high-dimensional image data. The pendulum dataset was originally introduced in (Karl et al., 2017) and consists of 500 sequences with 15 images each, which have a size of 16×16 pixels. The reacher dataset consists of 2000 sequences with 30 time steps each. We use two versions in our experiments: the first contains partially observed system states, i.e. the angle of the first and second joint; in the second version, observations are represented by RGB images of 64×64 pixels in size.

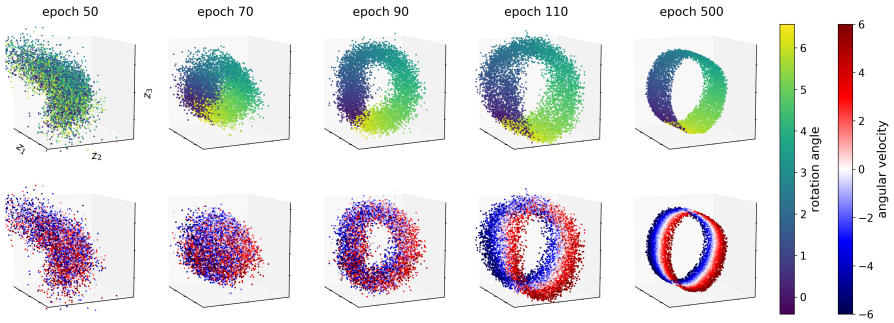
In our experiments, we use smoothing posteriors during training. This allows inferring an accurate state-space representation of partially observed systems already in the initial time step. In Section 4.4.1, we demonstrate our constrained optimisation framework on the example of deep Kalman smoothers (DKSs) (Krishnan et al., 2015)—and show in Sec. 4.4.2 that it significantly improves learning the underlying dynamics of observed systems on the example of DKSs, deep variational Bayes smoothers (DVBSs) (Karl et al., 2017), and extended Kalman variational autoencoders (EKVAEs). Furthermore, we verify in Section 4.4.3 that RNN-based transition models, as is the case in Kalman variational autoencoders (KVAEs) (Fraccaro et al., 2017) and recurrent state-space models (RSSMs) (Hafner et al., 2019), lead to a non-Markovian state space. In Section 4.4.4, we demonstrate learning disentangled position–velocity representations with the EKVAE. These are used to validate the learned dynamic model in the context of model-based reinforcement learning. The model architectures can be found in Appendix A.1.2.

4.4.1 *Demonstrating the Constrained Optimisation Framework*

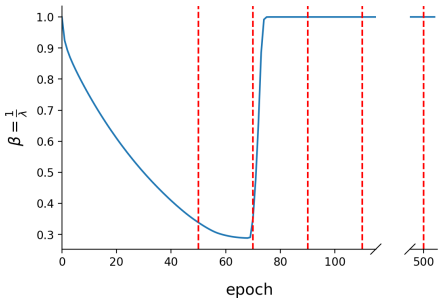
First, we demonstrate the constrained optimisation framework introduced in Section 4.1 on the example of the DKS, which we train on image data of a moving pendulum (see Figure 4.3). We refer to this model as VHP-DKS (CO), implying that it is trained through constrained optimisation (CO) with the variational hierarchical prior (VHP) as part of the model. The objective function can be found in Eq. (4.11).

Karl et al. (2017) have shown that the DKS is not capable of learning the underlying dynamics of the moving pendulum—i.e. to accurately predict the observed system—when trained classically or by applying (linear) annealing. In the following, we demonstrate that our constrained optimisation framework solves this problem.

Figure 4.1 shows the optimisation process of VHP-DKS (CO), where Figure 4.1a depicts the learned state-space representation of the pendulum at selected optimisation steps (epochs). As mentioned in Sec-



(a) State-space representation at selected epochs



(b) Course of β

Figure 4.1: VHP-DKS (CO): Learning the state-space representation of a pendulum (image data) with our constrained optimisation framework. Distortion and rate are balanced by β. It is updated (cf. Algorithm 3) such that the model first reduces the reconstruction error (constraint) by learning the rotation angle (see epoch 70). As soon as the constraint is satisfied, β increases and the model starts learning the underlying dynamics, i.e. to represent the angular velocity.

tion 4.1.2.1, we use the β-parametrisation of the Lagrange multiplier, $\beta = 1/\lambda$, to be in line with previous literature. The ratio between distortion and rate is balanced by β, which is updated (cf. Algorithm 3) to reduce first the reconstruction error (constraint) by learning the rotation angle (see epoch 70 in Figure 4.1a). As soon as the constraint is satisfied, β increases, and the model starts learning the underlying dynamics, i.e. to represent the angular velocity.

In Figure 4.2, we compare annealing with constrained optimisation and demonstrate the effect of the VHP. Constrained optimisation (Figure 4.2, middle & top) enables the DKS to learn the underlying system dynamics, which is indicated by the barrel shape of the learned representation (cf. Karl et al., 2017; Watter et al., 2015) and verified in Section 4.4.2. By contrast, training the DKS with annealing (Figure 4.2, bottom) does not lead to a state-space representation where the model learns to infer the angular velocity of the pendulum (cf. Table 4.1), which confirms the experimental findings in (Karl et al., 2017). Furthermore, we show that the VHP (Figure 4.2, top) significantly improves the quality of generated sequences. This is because the VHP learns

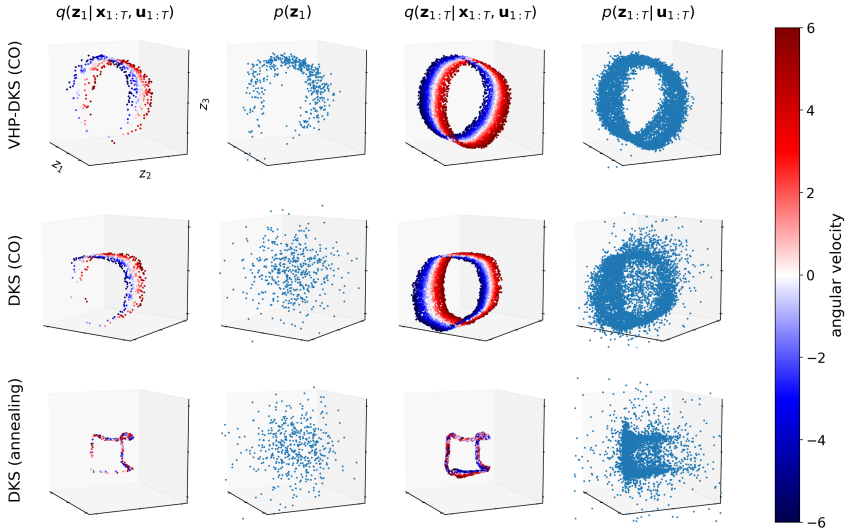
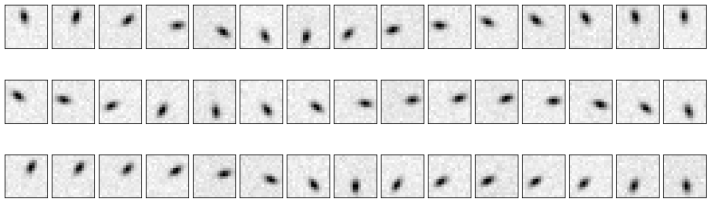
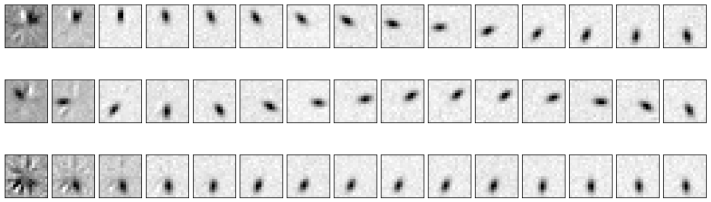


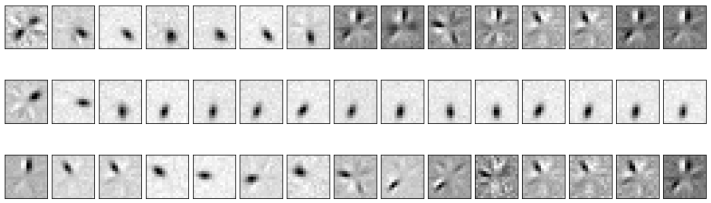
Figure 4.2: Pendulum (image data). In contrast to annealing (bottom), constrained optimisation (CO) (middle & top) enables the model to learn the underlying system dynamics, as we verify in Table 4.1. Furthermore, the VHP (top) significantly improves the quality of generated sequences $p(\mathbf{z}_{1:T}|\,\mathbf{u}_{1:T})$, as verified in Figure 4.3.



(a) VHP-DKS (CO)



(b) DKS (CO)



(c) DKS (annealing)

Figure 4.3: Generated sequences of a moving pendulum. In order to generate realistic data, the prior has to match the inferred latent manifold. This is realised by the VHP, which learns a prior $p(\mathbf{z}_1)$ corresponding to $q(\mathbf{z}_1|\,\mathbf{z}_{1:T}, \mathbf{u}_{1:T})$, cf. Figure 4.2 (top).

a $p(\mathbf{z}_1)$ that matches the manifold of the initial $q(\mathbf{z}_1 | \mathbf{z}_{1:T}, \mathbf{u}_{1:T})$. As a consequence, the manifold of the generated sequences $p(\mathbf{z}_{1:T} | \mathbf{u}_{1:T})$ corresponds to $q(\mathbf{z}_{1:T} | \mathbf{u}_{1:T})$. This leads to realistic generated pendulum movements, as verified in Figure 4.3.

4.4.2 *Prediction Accuracy: The Impact of the State-Space Representation*

In this section, we demonstrate that our constrained optimisation framework improves learning the underlying dynamics of observed systems on the example of DKSs, DVBSs, and EKVAEs.

4.4.2.1 *Pendulum Dataset*

Table 4.1 shows, on the example of pendulum image data, that the constrained optimisation framework significantly improves system identification in contrast to (linear) annealing. This is indicated by a high correlation between inferred and ground-truth states—i.e. rotation angle ϕ and angular velocity $\dot{\phi}$—which we measure through R^2 of an OLS regression. Since the pendulum can make a full 360 degree turn, the model learns to represent the rotation angle ϕ by a circle, resulting in a barrel-shaped state-space representation (cf. Figure 4.2). Therefore, we perform three OLS regressions on the learned representations. In the first two, we use $\sin(\phi)$ and $\cos(\phi)$ as ground truth (cf. Karl et al., 2017), where $R^{2\ (\text{ang. } \phi)}_{\text{OLS reg.}}$ is the corresponding mean. $R^{2\ (\text{vel. } \dot{\phi})}_{\text{OLS reg.}}$ refers to the third OLS regression with $\dot{\phi}$ as ground truth.

Table 4.1: Pendulum (image data). The constrained optimisation framework significantly improves system identification, as indicated by the high correlation (R^2) between inferred and ground-truth states ($\phi, \dot{\phi}$). In contrast to high ELBO values, high R^2 values coincide with an increase of the models' prediction accuracy (mean squared error (MSE)).

model	test ELBO	$R^{2\ (\text{ang. } \phi)}_{\text{OLS reg.}}$	$R^{2\ (\text{vel. } \dot{\phi})}_{\text{OLS reg.}}$	$\text{MSE}^{\ (\text{smoothed})}_{\text{predict}}$
VHP-EKVAE (CO)	**807.3**	**0.992**	**0.998**	**1.99E-4**
EKVAE (CO)	805.9	0.957	0.991	3.53E-4
EKVAE (annealing)	804.2	0.687	0.339	1.94E-3
VHP-DVBS (CO)	804.3	**0.992**	0.989	5.63E-4
DVBS (CO)	803.8	0.985	0.980	9.41E-4
DVBS (annealing)	803.1	0.795	0.237	4.67E-3
VHP-DKS (CO)	804.7	0.973	0.990	1.73E-3
DKS (CO)	804.1	0.912	0.962	2.36E-3
DKS (annealing)	804.0	0.330	0.040	2.12E-2

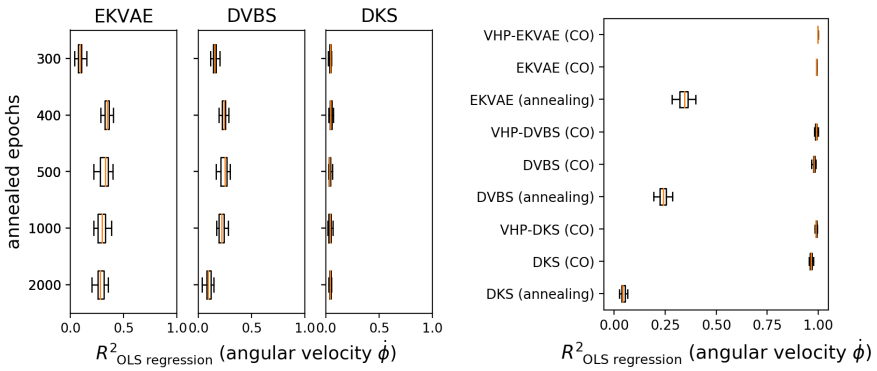


Figure 4.4: Pendulum (image data). Statistic evaluation of different annealing schedules. For this purpose, we measure the correlation between inferred and ground-truth angular velocity by R^2 (OLS regression) and compare the best schedule with CO (right), cf. Table 4.1. The statistics are based on 25 experimental runs each and indicate that CO facilitates system identification.

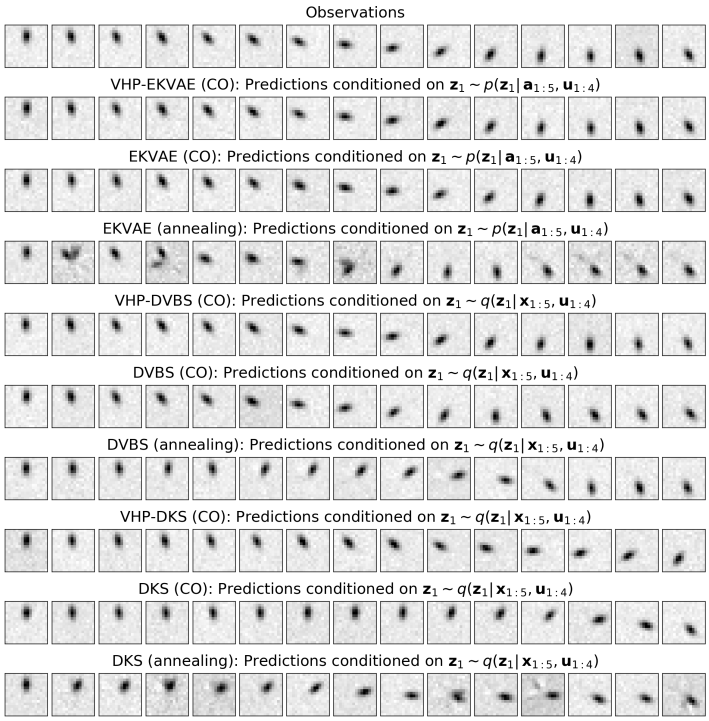


Figure 4.5: Predicted sequences of a moving pendulum conditioned on $\mathbf{z}_1 \sim q(\mathbf{z}_1|\,\mathbf{x}_{1:5}, \mathbf{u}_{1:4})$ or, in case of the EKVAE, on $\mathbf{z}_1 \sim p(\mathbf{z}_1|\,\mathbf{a}_{1:5}, \mathbf{u}_{1:4})$, where $\mathbf{a}_{1:5} \sim q(\mathbf{a}_{1:5}|\,\mathbf{x}_{1:5})$. The average prediction accuracy, measured by the MSE, can be found in Table 4.1.

A high correlation between inferred and ground-truth states goes hand in hand with an increase of the models' prediction accuracy. This is measured in Table 4.1 by $\text{MSE}^{(\text{smoothed})}_{\text{predict}}$, the MSE of 500 predicted sequences $\mathbf{x}_{1:15}$ (15 times steps). The predictions are conditioned on

$\mathbf{z}_1 \sim q(\mathbf{z}_1 | \mathbf{x}_{1:5}, \mathbf{u}_{1:4})$ or, in case of the EKVAE, on $\mathbf{z}_1 \sim p(\mathbf{z}_1 | \mathbf{a}_{1:5}, \mathbf{u}_{1:4})$, where the auxiliary variables are obtained through $\mathbf{a}_{1:5} \sim q(\mathbf{a}_{1:5} | \mathbf{x}_{1:5})$. Conditioning on $\mathbf{z}_1$ allows us to additionally verify whether the model has learned a good state-space representation in the initial time step. Note that high ELBO values, by contrast, do not indicate the model has learned to accurately predict the observed system.

Supplementary to Table 4.1, we show in Figure 4.5 the corresponding predicted sequences of the moving pendulum. Figure 4.4 provides a statistic evaluation of different linear annealing schedules compared with CO, where the correlation between inferred and ground-truth angular velocity is measured based on 25 experimental runs each. The results indicate that constrained optimisation facilitates system identification. Figures 4.6–4.14 show visualisations of the state-space representations learned by the different models and provide further evaluations including reconstructed observations and one-step predictions.

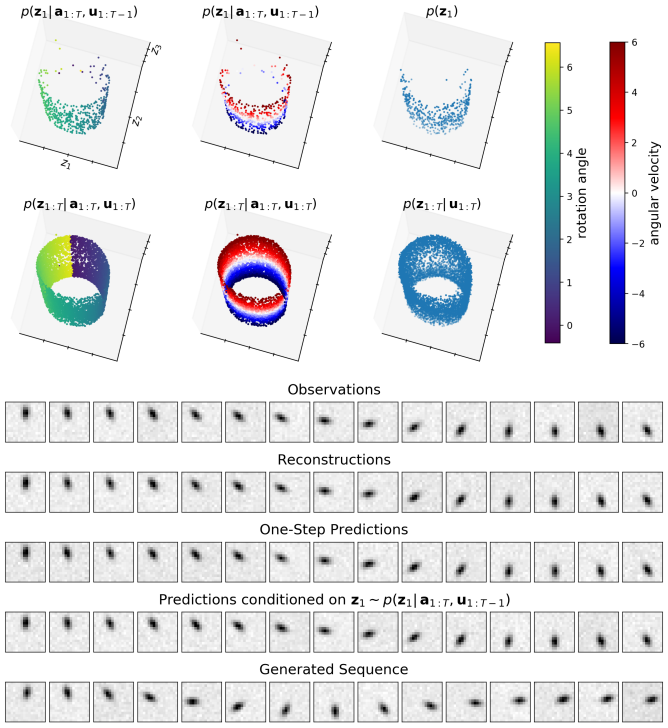


Figure 4.6: VHP-EKVAE (CO) trained on pendulum image data (supplementary to Table 4.1). In combination with the constrained optimisation framework, the EKVAE identifies the dynamic system of the pendulum and learns to predict it accurately.

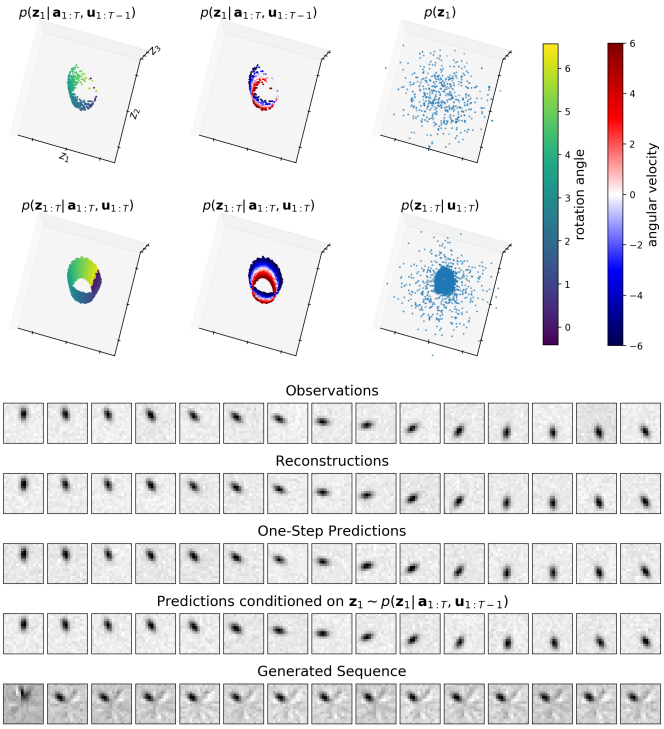


Figure 4.7: EKVAE (CO) trained on pendulum image data (supplementary to Table 4.1). Without the VHP, the EKVAE identifies the dynamic system of the pendulum but does not learn to process samples from the prior, which results in a broken generative model.

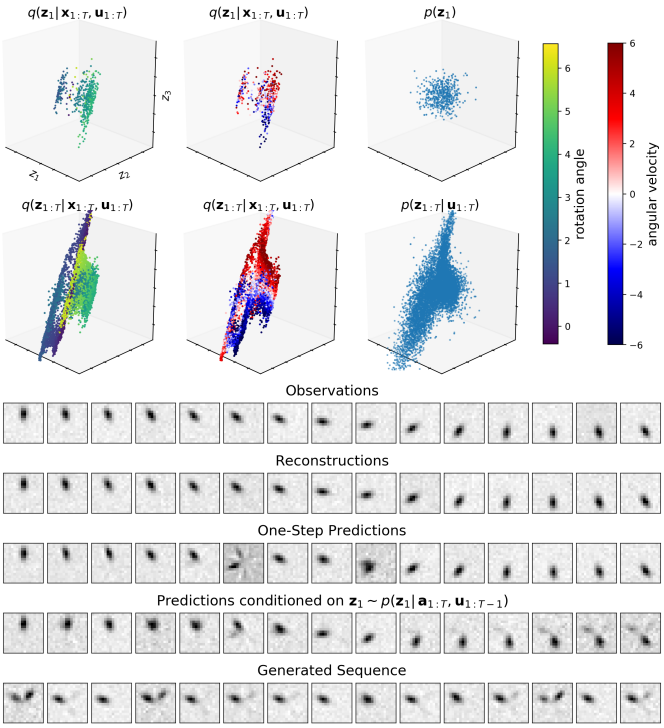


Figure 4.8: EKVAE (annealing) trained on pendulum image data (supplementary to Table 4.1). Without the constrained optimisation framework, the EKVAE does not learn to accurately predict the observed dynamic system.

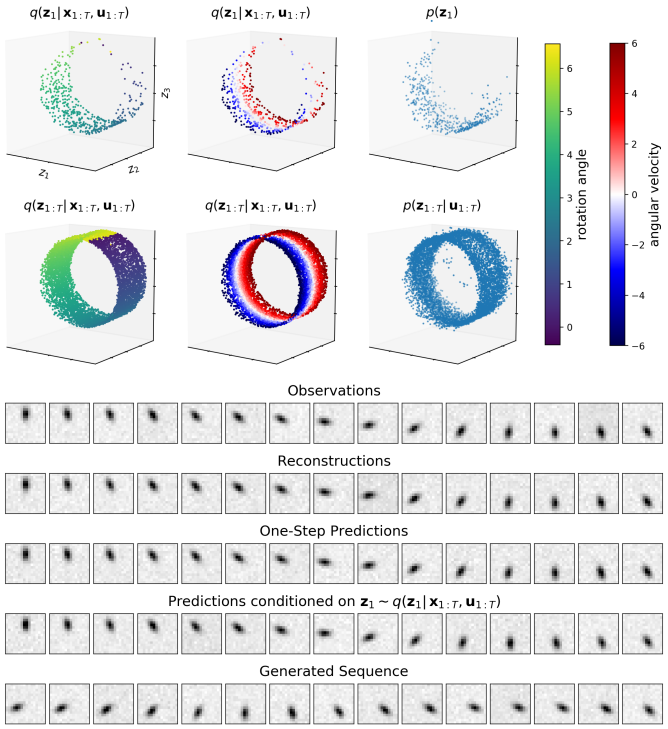


Figure 4.9: VHP-DVBS (CO) trained on pendulum image data (supplementary to Table 4.1). In combination with the constrained optimisation framework, DVBS identifies the dynamic system of the pendulum and learns to predict it accurately.

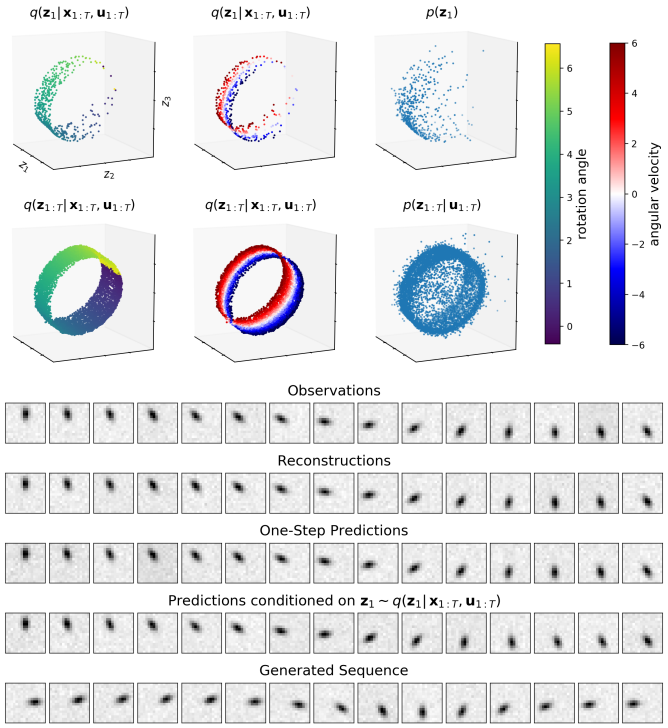


Figure 4.10: DVBS (CO) trained on pendulum image data (supplementary to Table 4.1). The original empirical Bayes prior proposed for DVBS leads to a poorer generative model than the VHP.

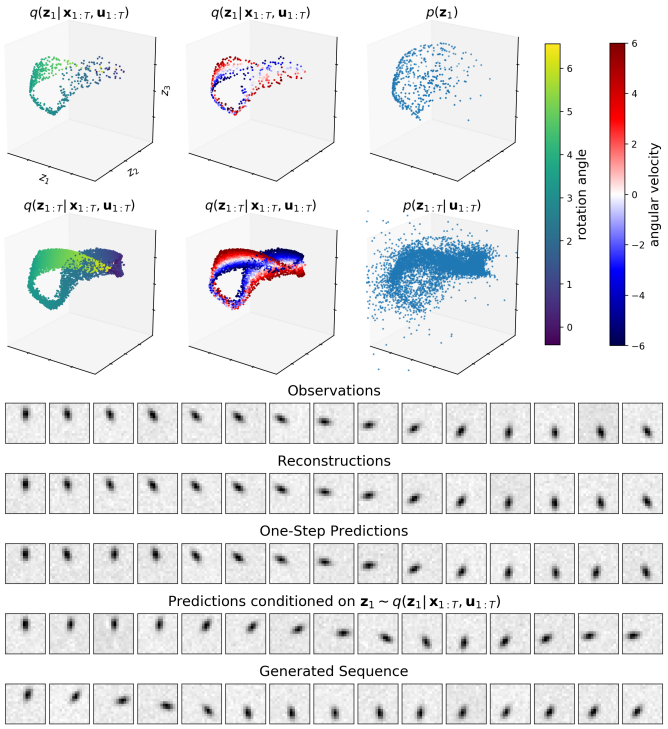


Figure 4.11: DVBS (annealing) trained on pendulum image data (supplementary to Table 4.1). Without the constrained optimisation framework, DVBS does not learn to accurately predict the observed dynamic system.

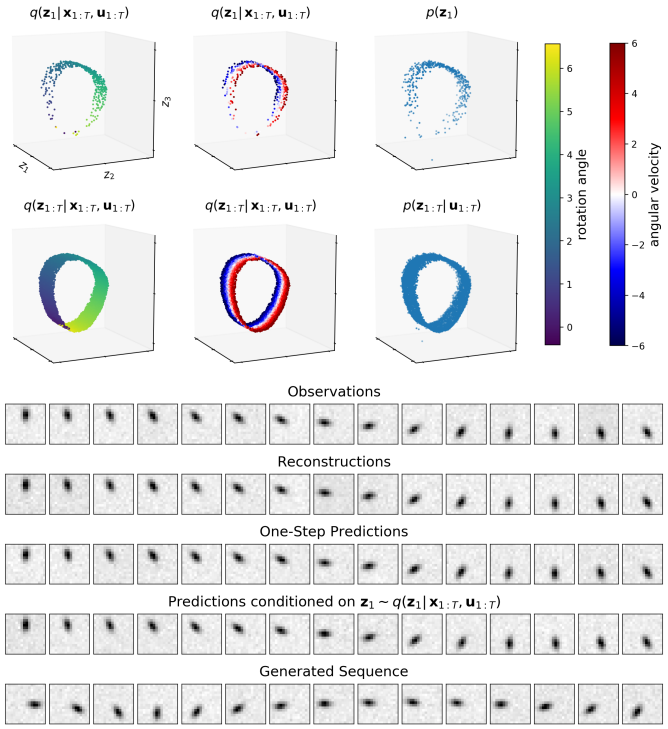


Figure 4.12: VHP-DKS (CO) trained on pendulum image data (supplementary to Table 4.1). In combination with the constrained optimisation framework, DKS identifies the dynamic system of the pendulum and learns to predict it accurately.

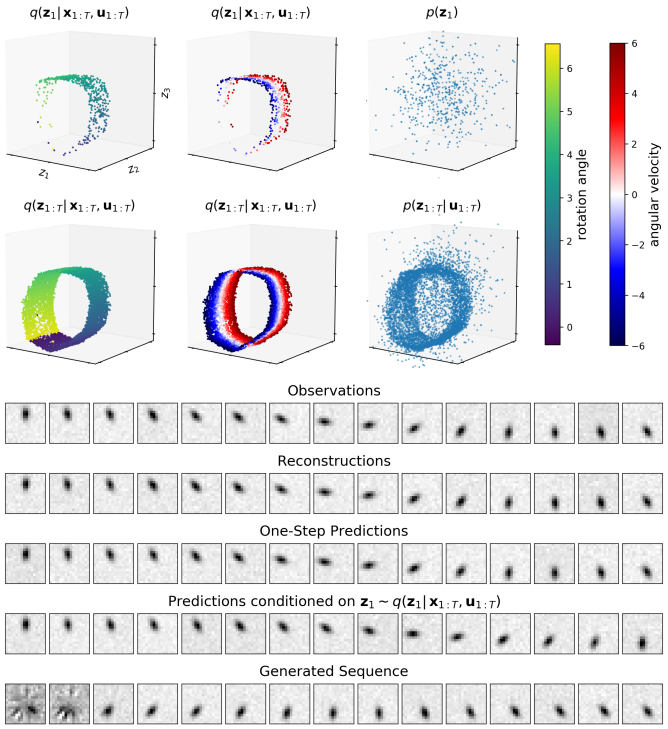


Figure 4.13: DKS (CO) trained on pendulum image data (supplementary to Table 4.1). Without the VHP, DKS identifies the dynamic system of the pendulum but does not learn to process samples from the prior, which results in a broken generative model.

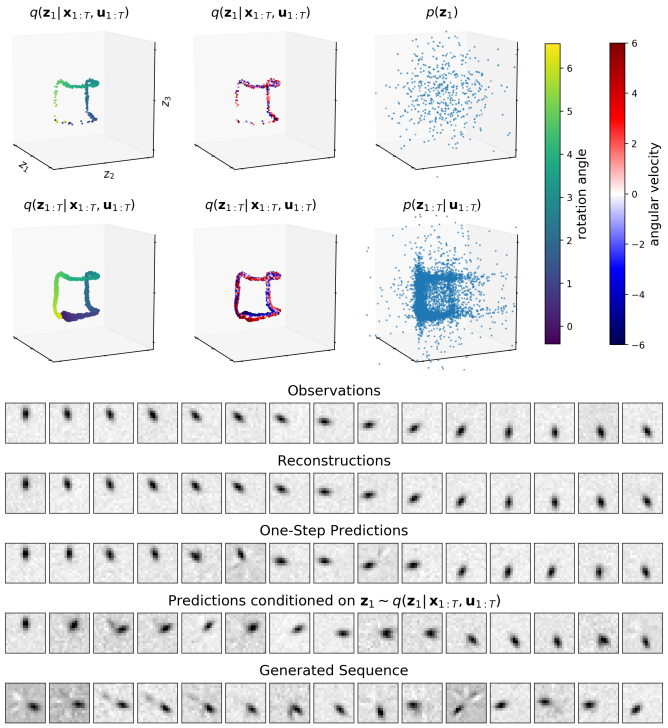


Figure 4.14: DKS (annealing) trained on pendulum image data (supplementary to Table 4.1). Without the constrained optimisation framework, DKS does not learn to accurately predict the observed dynamic system.

4.4.2.2 *Reacher Dataset*

Table 4.2 compares the constrained optimisation framework with (linear) annealing on the example of the reacher environment, where we use angle as well as high-dimensional image data (cf. Figure 4.16) as observations.

As in the pendulum experiments, we measure the correlation between inferred and ground-truth states—i.e. the joint angles (ϕ, ψ) and the related angular velocities $(\dot{\phi}, \dot{\psi})$—through R^2 of an OLS regression. In case of angle data (Table 4.3a), we perform *four* OLS regressions on the learned representations with $(\phi, \psi, \dot{\phi}, \dot{\psi})$ as ground truth. In case of image data (Table 4.3b), we perform *five* OLS regressions on the learned representations because we use $\sin(\phi)$ and $\cos(\phi)$, instead of ϕ, as ground truth, where $R^{2\ (\text{ang. } \phi)}_{\text{OLS reg.}}$ refers to the corresponding mean. As in the pendulum experiments, this is necessary for image data since the model learns to represent the first joint angle ϕ of the reacher by a circle (cf. Figure 4.15). This is because the first joint can make, in contrast to the second one, a full 360 degree turn.

A high correlation between inferred and ground-truth states coincides with an increase of the models' prediction accuracy. This is measured by $\text{MSE}^{\ (\text{smoothed})}_{\text{predict}}$, the MSE of 500 predicted sequences $\mathbf{x}_{1:30}$ (30 times steps). The predictions are conditioned on $\mathbf{z}_1 \sim q(\mathbf{z}_1 | \mathbf{x}_{1:5}, \mathbf{u}_{1:4})$ or, in case of the EKVAE, on $\mathbf{z}_1 \sim p(\mathbf{z}_1 | \mathbf{a}_{1:5}, \mathbf{u}_{1:4})$, where $\mathbf{a}_{1:5} \sim q(\mathbf{a}_{1:5} | \mathbf{x}_{1:5})$.

Figure 4.15 shows the disentangled 5-dimensional position–velocity representation learned by the VHP-EKVAE (CO) (see Section 4.3 for a description of our disentanglement method). The first three dimensions represent the two joint angles (ϕ, ψ), the last two dimensions

Table 4.2: Reacher. The correlation between inferred and ground-truth states $(\phi, \psi, \dot{\phi}, \dot{\psi})$ is measured by R^2 (OLS regression). The models' prediction accuracy is measured by the MSE.

(a) angle data

model	$R^{2\ (\text{ang. } \phi)}_{\text{OLS reg.}}$	$R^{2\ (\text{ang. } \psi)}_{\text{OLS reg.}}$	$R^{2\ (\text{vel. } \dot{\phi})}_{\text{OLS reg.}}$	$R^{2\ (\text{vel. } \dot{\psi})}_{\text{OLS reg.}}$	$\text{MSE}^{\ (\text{smoothed})}_{\text{predict}}$
VHP-EKVAE (CO)	0.988	**0.997**	**0.989**	0.986	**2.13E-5**
EKVAE (annealing)	0.712	0.835	0.881	0.339	4.38E-4
VHP-DVBS (CO)	**0.990**	0.994	0.979	**0.991**	2.75E-4
DVBS (annealing)	0.897	0.949	0.963	0.778	4.17E-4
VHP-DKS (CO)	0.984	0.991	0.986	0.980	3.52E-4
DKS (annealing)	0.693	0.781	0.965	0.016	1.12E-3

(b) image data (cf. Figure 4.15 and Figure 4.16)

model	$R^{2\ (\text{ang. } \phi)}_{\text{OLS reg.}}$	$R^{2\ (\text{ang. } \psi)}_{\text{OLS reg.}}$	$R^{2\ (\text{vel. } \dot{\phi})}_{\text{OLS reg.}}$	$R^{2\ (\text{vel. } \dot{\psi})}_{\text{OLS reg.}}$	$\text{MSE}^{\ (\text{smoothed})}_{\text{predict}}$
VHP-EKVAE (CO)	**0.980**	**0.986**	**0.991**	**0.987**	**1.64E-4**
EKVAE (annealing)	0.672	0.052	0.668	0.091	1.82E-3

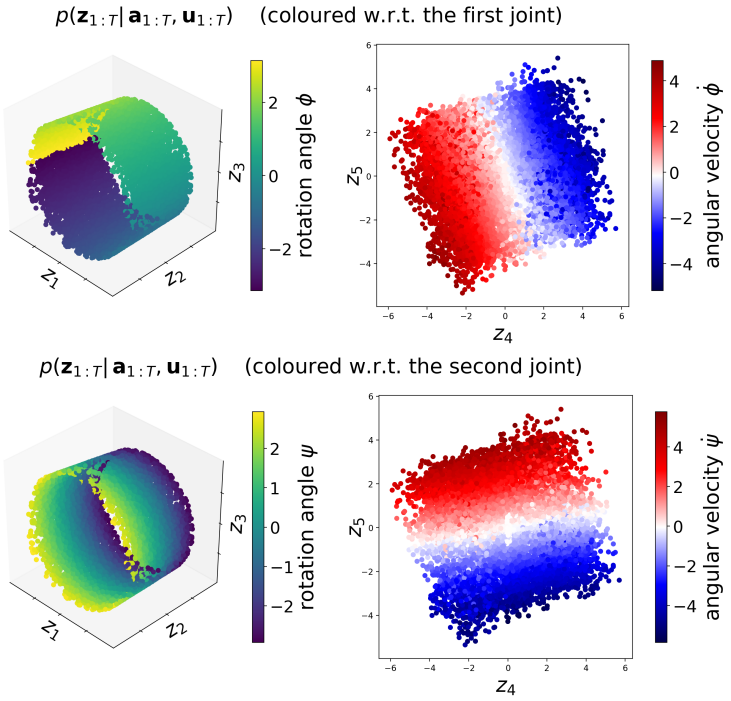


Figure 4.15: Reacher (image data). Disentangled 5-dimensional position–velocity representation learned by the VHP-EKVAE (CO). The first three dimensions represent the two joint angles (ϕ, ψ), the last two dimensions represent the respective angular velocities $(\dot{\phi}, \dot{\psi})$.

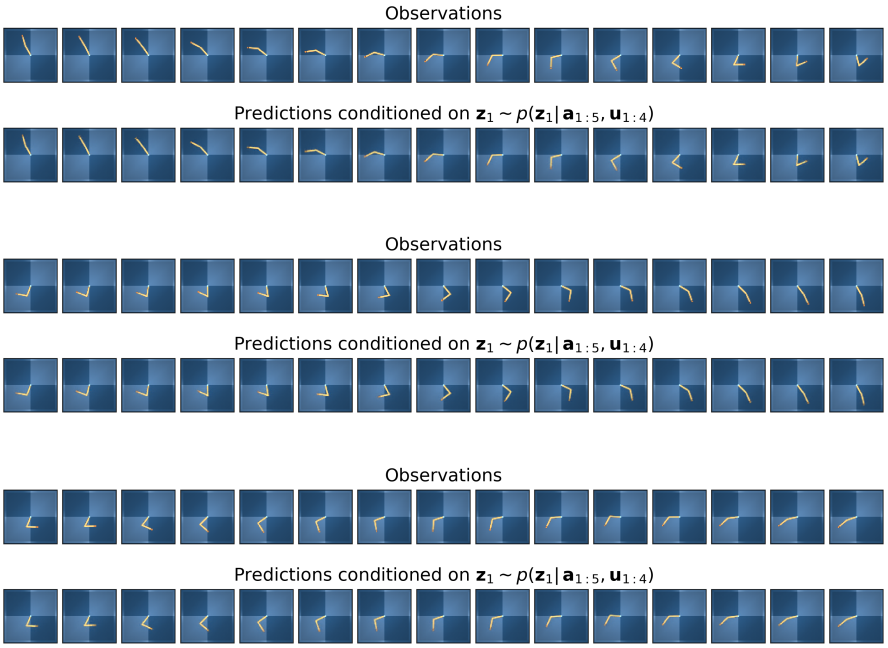


Figure 4.16: VHP-EKVAE (CO). Predicted sequences (first 15 time steps) of the moving reacher conditioned on the smoothed distribution $\mathbf{z}_1 \sim p(\mathbf{z}_1|\,\mathbf{a}_{1:5}, \mathbf{u}_{1:4})$, where the auxiliary variables are obtained through $\mathbf{a}_{1:5} \sim q(\mathbf{a}_{1:5}|\,\mathbf{x}_{1:5})$.

represent the respective angular velocities $(\dot{\phi}, \dot{\psi})$. The barrel shape indicates the model has learned that the first joint can make a 360 degree turn, whereas the second joint is restricted to avoid self-collisions. This is verified by the experiments in Figure 4.21.

Predicted sequences (first 15 time steps) of the moving reacher can be found in Figure 4.16. They are conditioned on the smoothed distribution $\mathbf{z}_1 \sim p(\mathbf{z}_1 | \mathbf{a}_{1:5}, \mathbf{u}_{1:4})$, where the auxiliary variables are obtained through $\mathbf{a}_{1:5} \sim q(\mathbf{a}_{1:5} | \mathbf{x}_{1:5})$. The average prediction accuracy measured by the MSE can be found in Table 4.3b.

4.4.3 *Limitations of Recurrent-Neural-Network-Based Transition Models*

In this section, we show that as a consequence of the LSTM-based transition models, the KVAE and the RSSM learn a non-Markovian state space, i.e. not all information about the state is encoded in $\mathbf{z}_t$, but partially in the LSTM. This is indicated by the low correlation (R^2) between the inferred and ground-truth angular velocity, when trained on pendulum image data (see Table 4.3). The OLS regressions are performed identically to Section 4.4.2.

In order to verify that the KVAE and the RSSM learn a non-Markovian state space, we compare in Table 4.3 the accuracy (MSE) of 500 predicted sequences $\mathbf{x}_{1:15}$ (15 time steps), which are either conditioned on the smoothed or the filtered posterior (referred to as $\text{MSE}_{\text{predict}}^{(\text{smoothed})}$ and $\text{MSE}_{\text{predict}}^{(\text{filtered})}$). This allows us to isolate the influence of the LSTM on the model's prediction accuracy, as we elaborate below.

The KVAE uses the transition model $p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{h}_{t-1}, \mathbf{u}_{t-1})$, where $\mathbf{h}_{t-1} = \text{LSTM}(\mathbf{a}_{1:t-1})$. Table 4.3 shows that predictions conditioned on $\{\mathbf{z}_1 \sim p(\mathbf{z}_1 | \mathbf{a}_{1:5}, \mathbf{u}_{1:4}),\ \mathbf{h}_1 = \text{LSTM}(\mathbf{a}_1)\}$ (denoted by $\text{MSE}_{\text{predict}}^{(\text{smoothed})}$) are significantly less accurate than predicted sequences conditioned on $\{\mathbf{z}_5 \sim p(\mathbf{z}_5 | \mathbf{a}_{1:5}, \mathbf{u}_{1:4}),\ \mathbf{h}_5 = \text{LSTM}(\mathbf{a}_{1:5})\}$ (denoted by $\text{MSE}_{\text{predict}}^{(\text{filtered})}$). Figure 4.17 shows that the predicted position of the pendulum in the initial time step is identical to the observed position. Thus, we can

Table 4.3: Pendulum (image data). The correlation between inferred and ground-truth states $(\phi, \dot{\phi})$ is measured by R^2 (OLS regression). The models' prediction accuracy is measured by the MSE.

model	$R^2_{\text{OLS reg.}}^{(\text{ang. } \phi)}$	$R^2_{\text{OLS reg.}}^{(\text{vel. } \dot{\phi})}$	$\text{MSE}_{\text{predict}}^{(\text{smoothed})}$	$\text{MSE}_{\text{predict}}^{(\text{filtered})}$
VHP-KVAE (CO)	0.989	0.043	2.87E-3	4.24E-4
KVAE (annealing)	0.652	0.134	3.16E-3	6.67E-4
VHP-RSSM (CO)	0.915	0.086	3.10E-3	4.80E-4
RSSM (annealing)	0.158	0.060	3.23E-3	6.94E-4

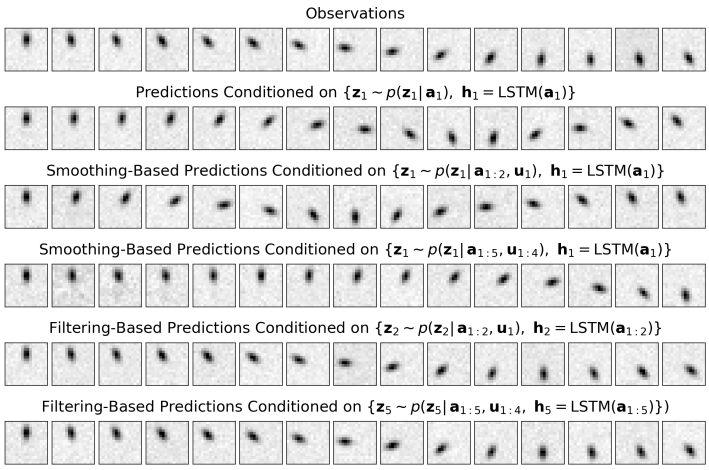


Figure 4.17: VHP-KVAE (CO). The predictions demonstrate that the KVAE encodes the angular velocity of the pendulum in $\mathbf{h}_t = \text{LSTM}(\mathbf{a}_{1:t})$ and not in $\mathbf{z}_t$, cf. Table 4.3.

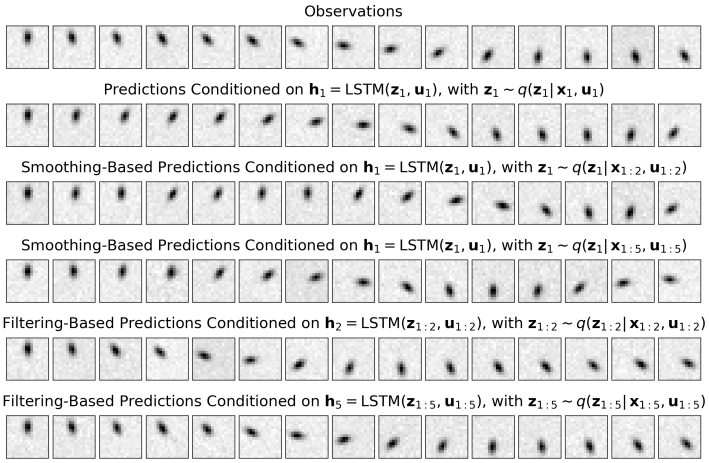


Figure 4.18: VHP-RSSM (CO). The predictions demonstrate that the RSSM encodes the angular velocity of the pendulum in $\mathbf{h}_t = \text{LSTM}(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$ and not in $\mathbf{z}_t$, cf. Table 4.3.

conclude that the low accuracies of the smoothing-based predictions are due to missing information about the dynamics, i.e. the angular velocity of the pendulum. When smoothing back to the initial time step, this information can only be provided by $\mathbf{z}_1$ since $\mathbf{h}_1 = \text{LSTM}(\mathbf{a}_1)$ does not have access to sequential data and therefore cannot infer any dynamics. Consequently, we state that the angular velocity is encoded in $\mathbf{h}_t = \text{LSTM}(\mathbf{a}_{1:t})$ and can only be inferred for $t \geq 2$, as shown in Figure 4.17.

The same line of argument can be applied to the RSSM. It uses the transition model $p(\mathbf{z}_t|\,\mathbf{h}_{t-1})$, where $\mathbf{h}_{t-1} = \text{LSTM}(\mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1})$. The experimental results in Table 4.3 and Figure 4.18 show that the RSSM learns a non-Markovian state representation in $\mathbf{z}$ since the angular velocity of the pendulum is inferred by the LSTM.

4.4.4 *Model Validation: Policy Learning With Disentangled Position–Velocity Representations*

In the following, we use the approach introduced in Section 4.3 to learn disentangled position–velocity representations with the EKVAE. Figure 4.19 demonstrates, on the example of pendulum image data, how defining $\mathbf{H}$ in $p(\mathbf{a}_t|\,\mathbf{z}_t)$ (Eq. (4.17)) as rectangular identity matrix (Figure 4.19, left) allows disentangling the rotation angle and the angular velocity in the latent space. Furthermore, Figure 4.19 shows how different permutations of $\mathbf{H}$ change the dimensions where the model learns to represent the rotation angle or angular velocity. The learned disentangled representation of the reacher (image data) can be found in Figure 4.15.

As a next step, we validate the learned models in the context of model-based reinforcement learning. To this end, we use the method presented in Section 4.3.2 for defining reward functions based on disentangled position–velocity representations. This allows us to learn policies exclusively through $p(\mathbf{z}_t|\,\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$.

Figure 4.20 shows the visualisation of different policies that are learned based on the disentangled position–velocity representation of the pendulum (image data) in Figure 4.19 (left). The policies are tested on the original pendulum environment, which was also used to generate the dataset. The first example (top) demonstrates the pendulum swing-up, which is achieved by encoding the goal position for $r_t^{\text{pos}}(\mathbf{z}_t, \mathbf{p}_g)$ and using an action interval of $7 \geq \mathbf{a} \geq -7$. The second and third example (middle and bottom) demonstrate steady clockwise and counter-clockwise rotations of the pendulum with different angular velocities using an action interval of $30 \geq \mathbf{a} \geq -30$. We achieve this by encoding the goal angular velocity for $r_t^{\text{vel}}(\mathbf{z}_t, \mathbf{v}_g)$: in Figure 4.20 (middle), we use 50% of the maximum speed and in Fig-

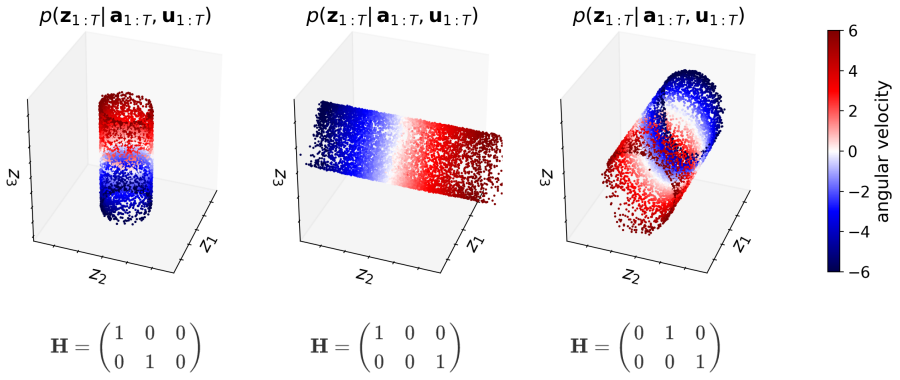


Figure 4.19: Pendulum (image data). Disentangled 3-dimensional position–velocity representations learned by the VHP-EKVAE (CO). Depending on $\mathbf{H}$, the model learns to represent the angular velocity $\dot{\phi}$ solely by one of the three dimensions of the latent space.

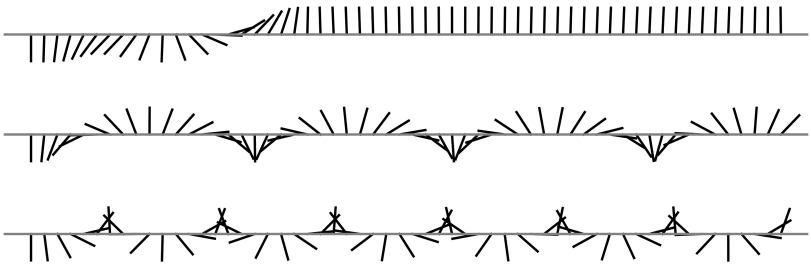


Figure 4.20: VHP-EKVAE (CO). Visualisation of different policies that we learned based on the disentangled position–velocity representation in Figure 4.19 (left). Swinging up the pendulum by encoding the goal position for $r_t^{\text{pos}}(\mathbf{z}_t, \mathbf{p}_g)$ (top). Steady rotating the pendulum by encoding the goal angular velocity for $r_t^{\text{vel}}(\mathbf{z}_t, \mathbf{v}_g)$: clockwise with 50% of the maximum speed (middle); and counter-clockwise with 85% of the maximum speed defined by the dataset (bottom).

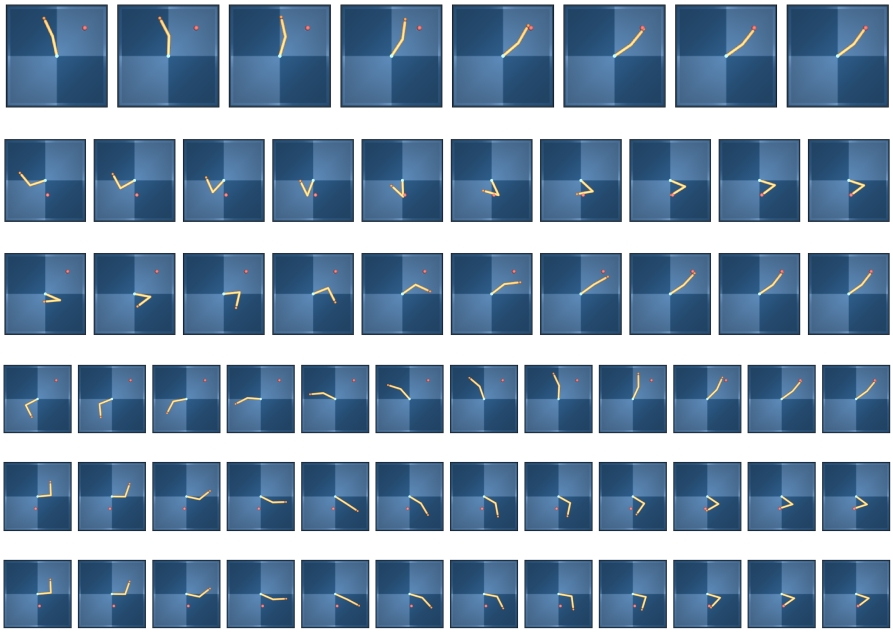


Figure 4.21: VHP-EKVAE (CO). Policy for reaching an encoded goal position (red dot) that we learned based on the disentangled position–velocity representation in Figure 4.15. The results show that the EKVAE has learned an accurate model of the observed system.

ure 4.20 (bottom) 85% of the maximum speed defined by the dataset. The experiments verify that the EKVAE has learned an accurate model of the pendulum. Furthermore, they demonstrate the variety of applications for disentangled position–velocity representations and the related policy learning approach.

Figure 4.21 shows the visualisation of the policy that is learned based on the disentangled position–velocity representation of the reacher (image data) in Figure 4.15. The respective goal position used

for defining the reward function is denoted by the red dot. The policies are tested in the reacher environment of Deepmind's control suite. Our results show that the EKVAE has learned an accurate model of the reacher environment that avoids self-collisions and ensures precise reaching of a desired position.

4.5 SUMMARISING DISCUSSION

In this chapter, we have dealt with the question of how to learn DSSMs in order to obtain accurate temporal predictions of the observed dynamic system. We have addressed this problem by proposing a constrained optimisation framework as a general approach for learning DSSMs. To this end, we have derived a general Lagrangian formulation of the sequential ELBO on the basis of distortion and rate—and extended the hierarchical empirical Bayes prior (VHP) as well as the associated optimisation algorithm introduced in Chapter 3 to DSSMs. Building upon the constrained optimisation framework, we have introduced the EKVAE that leverages the concept of extended Kalman filtering/smoothing for approximate inference and is capable of learning disentangled position–velocity representations. Furthermore, we have proposed a method—in the context of model-based reinforcement learning—for defining/encoding reward functions by means of disentangled position–velocity representations, which allows us to learn policies exclusively through the transition model.

We have shown in Section 4.4.2 that high ELBO values do not imply the model has learned the underlying dynamics of the data, i.e. to accurately predict the observed system. Our proposed constrained optimisation framework addresses this issue. Indeed, the experiments on the pendulum and reacher data have demonstrated that it significantly improves system identification using DKSs, DVBSs, and EKVAEs—which goes hand in hand with an increase of the models' prediction accuracy. Furthermore, we have shown in Section 4.4.1 that the hierarchical empirical Bayes prior substantially improves the quality of generated sequences. In Section 4.4.3, we have verified that KVAEs and RSSMs learn a non-Markovian state space due to their RNN-based transition models, i.e. not all information about the state of the system is captured by the latent variables. In Section 4.4.4, we have learned disentangled position–velocity representations of the pendulum and the reacher with our EKVAE—and used these to define reward functions for model-based reinforcement learning. In this way, we have verified that the model of the pendulum environment is precise enough for learning policies that allow us to swing the pendulum up or make it rotate with a constant angular velocity. Moreover, we have shown that the EKVAE has learned an accurate model of the reacher environment that avoids self-collisions and ensures the precise reaching of a desired position.

In the conducted experiments, we have primarily used our policy-learning method to validate the learned dynamic model of the observed system. In future studies, it can be compared with state-of-the-art policy-learning methods, which is particularly interesting for environments where rewards are not available.

5

UNSUPERVISED METRIC LEARNING WITH VARIATIONAL AUTOENCODERS

In this chapter, we return to the framework of variational autoencoders (VAEs) and show how VAEs can be applied to measure the similarity of data in an unsupervised way.

A variety of algorithms rely on suitable distance metrics. Popular examples are k-nearest neighbours (Altman, 1992); kernel methods, such as KernelPCA (Schölkopf et al., 1997) and KernelNMF (Li and Ding, 2006); or interpolation methods for creating new data (Caruso and Quarta, 1998). Applications in computer vision, such as object tracking over time (Bewley et al., 2016; Wojke et al., 2017), depend on the quality of similarity scores.

It can be challenging, however, to define such distance metrics. Applying the Euclidean distance, for instance, on image data gives each pixel the same importance; thus, the background might have a higher impact on the similarity score than the actual object of interest. Similar issues occur with other predefined metrics, for example, the Minkowski distance, since they come with certain assumptions on the data. Therefore, it is often not straightforward to find a mathematical description of similarity.

As a consequence, learning a distance metric directly from data, referred to as *metric learning*, has become a popular approach (Kulis et al., 2013; Tosi et al., 2014; Weinberger et al., 2006; Xing et al., 2002).

In this chapter, we demonstrate how metric learning can be addressed by the framework of variational autoencoders. VAEs are well suited for learning similarity metrics because they provide an encoder–decoder pair that captures underlying characteristics of the data, as detailed in Section 2.2.1. Furthermore, we show in our experiments that VAEs are capable of generating data with a continuous change of similarity when combined with an accurate distance metric.

The latent representation learned by VAEs, however, cannot be directly used for measuring similarities. This is because the objective function of VAEs does not incentivise learning a topology- or distance-preserving mapping to the latent space. Moreover, as discussed in Chapter 3, VAEs regularise the latent space to be compact, i.e. to remove low-density regions, due to the standard normal prior distribution. In the following sections, we show how Riemannian geometry provides a solution to this problem.

In Section 5.1, we define the latent space of VAEs as a Riemannian manifold. This allows us to compute geodesics, which takes into account the gradients of the decoder. Geodesics are length-minimising

curves on a Riemannian manifold, and we show that they can be used to define an accurate measure of similarity between data points. In other words: the information that allows evaluating the similarity of the data is contained in the latent variables *and* the decoder. To define a distance metric, we use both sources of information.

In Section 5.2, we go one step further and learn latent spaces that can directly act as a Euclidean similarity metric. For this purpose, we use the VHP-VAE introduced in Chapter 3. Due to the empirical Bayes prior, it can learn complex latent representations that reflect the topology of the data. To facilitate a Euclidean distance-preserving mapping to the latent space, we extend the objective function of the VHP-VAE by a regularisation term for the Riemannian metric tensor.

5.1 THE LATENT SPACE AS A RIEMANNIAN MANIFOLD

The methods and experimental results discussed in this section have been previously published in (Chen, Klushyn*, Kurle*, Jiang, Bayer and Smagt, 2018).*

The latent-variable model learned by VAEs,

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})\, p(\mathbf{z}), \tag{5.1}$$

characterises the observable data $\mathbf{x}$ by typically lower-dimensional latent variables $\mathbf{z}$, as detailed in Section 2.1.

In the following, we demonstrate how to use these learned characteristics for defining a distance measure $\delta(\mathbf{x}_i, \mathbf{x}_j)$. The latter is supposed to adequately reflect the similarity of different data points $\mathbf{x}_i$ and $\mathbf{x}_j$. Since VAEs allow for inferring the corresponding latent variables $\mathbf{z}_i$ and $\mathbf{z}_j$, the Euclidean distance $||\mathbf{z}_i - \mathbf{z}_j||_2$ seems to be an obvious similarity metric. This has the implicit assumption that moving a certain distance in latent spaces moves us proportionally far in observable space: $||\mathbf{z}_i - \mathbf{z}_j||_2 \propto ||\mathbf{x}_i - \mathbf{x}_j||_2$.

However, this assumption is often a fallacy. In order to model the data correctly, stark discontinuities in the likelihood function $p(\mathbf{x}|\mathbf{z})$ are virtually always present due to the predefined prior distribution. To see this, we note that the prior can be expressed as the posterior aggregated over the data:

$$p(\mathbf{z}) = \mathbb{E}_{p(\mathbf{x})}\big[p(\mathbf{z}|\mathbf{x})\big].$$

Thus, independent of the topology of the data manifold, the latent manifold corresponds to the prior distribution. In case of VAEs, the prior is typically the standard normal distribution; thus, the latent manifold does not include regions of low density. As a consequence, separated data in the observation space (e.g. sets of points from different classes) is placed next to each other in the latent space. This

can solely be realised by high gradients in the posterior $p(\mathbf{z}|\mathbf{x})$—and therefore in the likelihood $p(\mathbf{x}|\mathbf{z})$—at the respective borders.

Concurrently, these gradients can provide the necessary information for defining a similarity metric. In order to harness this information, we propose to make use of Riemannian geometry, a mathematical framework for studying the curvature of spaces.

Tosi et al. (2014) define the latent space of Gaussian-process latent-variable models as a Riemannian manifold. We follow their line of argument and define the VAE's latent space as a Riemannian manifold. This allows computing the length of curves on the latent manifold based on the Jacobian of the likelihood function and thus with respect to the observable space. The length-minimising curves are referred to as geodesics, and we show that their Riemannian length can be interpreted as an accurate similarity measure of the data.

In the following, we introduce our VAE-based approach for learning similarity metrics:

- In Section 5.1.1, Riemannian geometry is integrated with the framework of variational autoencoders by defining the VAE's latent space as a Riemannian manifold.
- In Section 5.1.2, we propose a method for finding/approximating geodesics between different data points, as the length of geodesics provides a measure of similarity.
- In Section 5.1.3, we address the question of how to analyse the Riemannian metric tensor by providing a visualisation method.
- In Section 5.1.4, we evaluate our proposed method. This includes experiments on a moving pendulum, on binarised MNIST, on a simulated robot arm, and on real-world human motion data.

5.1.1 *Riemannian Geometry in the Context of Variational Autoencoders*

Supplementary to the general introduction to VAEs in Section 2.2.1, we emphasise in the following paragraph some important details that are relevant for understanding our proposed method.

VAEs learn the parameters of the generative model in Eq. (5.1) by maximising the evidence lower bound (ELBO):

$$\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\big[\log p_\theta(\mathbf{x})\big] \geq \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}|\mathbf{z})\, p_\theta(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right], \tag{5.2}$$

where $p_{\mathcal{D}}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \delta(\mathbf{x} - \mathbf{x}_i)$ is the empirical distribution of the observed data $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{N}$. This requires learning an approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$—with the idea that $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ infer the underlying characteristics of $\mathcal{D}$ and enable the model therefore to generalise to unseen data. For example, $\big\{\mathbb{E}_{q_\theta(\mathbf{z}|\mathbf{x}_i)}[\mathbf{z}]\big\}_{i=1}^{N}$ cluster

with respect to some discrete features or important factors of variation in the data.

In order to use these encoded features for learning a similarity metric, two criteria need to be considered. First, for measuring $\delta(\mathbf{x}_i, \mathbf{x}_j)$, we must be able to infer the corresponding latent variables. This is addressed by the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$, which allows encoding observable data into the latent space. Second, the Jacobian of the likelihood function has to be taken into account. This is because, by maximising the ELBO in Eq. (5.2), the latent manifold is regularised towards the standard normal prior distribution (see Section 2.2.1). To harness the information of the likelihood's Jacobian, we define the latent space of the VAE as a Riemannian manifold.

5.1.1.1 *Riemannian Distance Function*

A Riemannian manifold is a differentiable manifold, which is additionally equipped with a metric tensor $\mathbf{G}$. The latter provides mathematical spaces with a measure for distances and angles. A simple example is the inner product of two vectors in the Euclidean space, $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\mathrm{T} \mathbb{1} \mathbf{b}$. The Euclidean spaces is a special case of a Riemannian manifold referred to as the Euclidean metric, where $\mathbf{G} = \mathbb{1}$.

By defining the latent space of VAEs as a Riemannian manifold M, an inner product on the tangent space $T_\mathbf{z} M$ is assigned to each point $\mathbf{z} \in M$ on the latent manifold:

$$\langle \mathbf{a}, \mathbf{b} \rangle_\mathbf{z} = \mathbf{a}^\mathrm{T} \mathbf{G}(\mathbf{z}) \mathbf{b}, \tag{5.3}$$

where $\mathbf{a}, \mathbf{b} \in T_\mathbf{z} M$. The Riemannian metric tensor

$$\mathbf{G}(\mathbf{z}) = \mathbf{J}(\mathbf{z})^\mathrm{T} \mathbf{J}(\mathbf{z})$$

is defined by the Jacobian of the likelihood $p_\theta(\mathbf{x}|\mathbf{z})$. In the context of VAEs, the Jacobian is determined by the decoder $\mathbf{f}_\theta : \mathbb{R}^{D_\mathbf{z}} \to \mathbb{R}^{D_\mathbf{x}}$:

$$\mathbf{J} = \frac{\partial \mathbf{f}_\theta(\mathbf{z})}{\partial \mathbf{z}} \in \mathbb{R}^{D_\mathbf{x} \times D_\mathbf{z}}. \tag{5.4}$$

In case of a Gaussian likelihood, the decoder typically returns the likelihood's mean since the variance is learned as a global parameter—in case of a Bernoulli likelihood, the decoder returns the probability of success.

Since a Riemannian manifold is differentiable, the inner product in Eq. (5.3) allows computing the length of curves on the manifold with respect to the observable space. Assume a curve $\gamma : [0, 1] \to \mathbb{R}^{D_\mathbf{z}}$

in the latent space of the VAE: the length of γ in the observable space—referred to as the Riemannian length—is defined by

$$\begin{aligned} L(\gamma) &= \int_0^1 \left\| \frac{\partial \mathbf{f}_\theta(\gamma(t))}{\partial t} \right\|_2 \mathrm{d}t \\ &= \int_0^1 \left\| \frac{\partial \mathbf{f}_\theta(\gamma(t))}{\partial \gamma(t)} \frac{\partial \gamma(t)}{\partial t} \right\|_2 \mathrm{d}t \\ &= \int_0^1 \left\| \mathbf{J} \frac{\partial \gamma(t)}{\partial t} \right\|_2 \mathrm{d}t \\ &= \int_0^1 \underbrace{\sqrt{\langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle_{\gamma(t)}}}_{\phi(t)} \mathrm{d}t, \qquad \text{(cf. Eq. (5.3))} \end{aligned} \tag{5.5}$$

where $\phi(t)$ can be interpreted as the rate of change at point $\gamma(t)$. Therefore, we refer to it as velocity. The velocity is used in our experiments (Section 5.1.4) for measuring the smoothness of a trajectory in the latent space with respect to the observable space.

As a next step, we define the *similarity metric* as the Riemannian distance

$$D = \min_\gamma L(\gamma), \tag{5.6}$$

which is the Riemannian length of the shortest path between two data points according to Eq. (5.5). In order to compute the Riemannian distance, we need to find the shortest path

$$\gamma_{\text{geodesic}} = \operatorname*{argmin}_\gamma L(\gamma),$$

which is referred to as the geodesic. However, computing the geodesic is not trivial. In Section 5.1.2, we address this issue and introduce an approach for approximating geodesics in the context of VAEs.

5.1.1.2 *Particularity of Importance Weighted Autoencoders*

In order to learn a richer latent representation and to obtain a more accurate generative model—compared to the classical VAE—we use the importance-weighted autoencoder (IWAE) (Burda et al., 2016; Cremer et al., 2017) in our experiments. As detailed in Section 2.2.2, the approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$ in IWAEs is treated as a proposal distribution for importance sampling. This leads to a tighter lower bound than the original ELBO in Eq. (5.2):

$$\begin{aligned} &\mathbb{E}_{p_\mathcal{D}(\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right] \\ &\quad \geq \mathbb{E}_{p_\mathcal{D}(\mathbf{x})} \mathbb{E}_{\mathbf{z}_{1:K} \sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(\mathbf{x}|\mathbf{z}_k)\, p_\theta(\mathbf{z}_k)}{q_\phi(\mathbf{z}_k|\mathbf{x})} \right]. \end{aligned} \tag{5.7}$$

However, as $q_\phi(\mathbf{z}|\mathbf{x})$ in Eq. (5.7) functions as a proposal distribution, it does not reflect the latent representation of our data. This is an important aspect that has to be taken into account when computing Riemannian distances. In order to obtain the latent representation, we use sequence importance resampling (SIR) (cf. Algorithm 1), as described in Section 2.2.2.

5.1.2 *Approximating Geodesics*

As discussed in the previous section, we define the similarity of data points by the Riemannian distance in Eq. (5.6). In the context of VAEs, the Riemannian distance is the length of the shortest path between two encoded data points on the latent (Riemannian) manifold, referred to as the geodesic.

However, VAEs learn a complex mapping from the latent to the observable space by means of neural networks that define the parameters of the likelihood. This is realised by high gradients resulting in a complex topology of the latent manifold, as we verify in Section 5.1.4. Therefore, it is not possible to analytically compute geodesics. In the following, we propose an approach for approximating geodesics in latent space of VAEs.

In order to approximate the geodesic γ_{geodesic} between two encoded latent variables $\mathbf{z}_0$ and $\mathbf{z}_1$, we use a neural network $\mathbf{g}_\omega : \mathbb{R} \to \mathbb{R}^{D_\mathbf{z}}$ that learns to map time steps $t \in [0, 1]$ to the corresponding latent variables. This allows approximating the Riemannian length in Eq. (5.5) by Monte Carlo integration:

$$L(\gamma) \approx L(\omega) = \frac{1}{T} \sum_{i=0}^{T} \sqrt{\langle \dot{\mathbf{g}}_\omega(t_i), \dot{\mathbf{g}}_\omega(t_i) \rangle_{\mathbf{g}_\omega(t_i)}}, \tag{5.8}$$

where the time steps $t_{1:T} \sim \mathcal{U}(0, 1)$ are sampled from a continuous uniform distribution. We obtain an approximation of the geodesic by minimising $L(\omega)$ with respect to ω and subject to the constraint that the curve starts at $\mathbf{z}_0$ and ends at $\mathbf{z}_1$. This results in the following constrained optimisation problem:

$$\min_\omega L(\omega) \quad \text{s.t.} \quad \mathbf{g}_\omega(0) = \mathbf{z}_0,\ \mathbf{g}_\omega(1) = \mathbf{z}_1. \tag{5.9}$$

5.1.2.1 *Satisfying the Boundary Constraints*

To satisfy the boundary constraints in Eq. (5.9), we shift and rescale the curve predicted by $\mathbf{g}_\omega(t)$:

$$\hat{\gamma}_{\text{geodesic}}(t) = \mathbf{a} \circ \mathbf{g}_\omega(t) - \mathbf{b},$$

where

$$a_i = \frac{z_0^{(i)} - z_1^{(i)}}{g_\omega^{(i)}(0) - g_\omega^{(i)}(1)},$$
$$b_i = \frac{z_0^{(i)}\, g_\omega^{(i)}(1) - z_1^{(i)}\, g_\omega^{(i)}(0)}{g_\omega^{(i)}(0) - g_\omega^{(i)}(1)}.$$

The advantage of this approach is that the optimisation problem in Eq. (5.9) is simplified to $\min_\omega L(\omega)$.

5.1.2.2 *Smoothing the Riemannian Metric Tensor*

The Riemannian length in Eq. (5.8) is approximated by Monte Carlo integration with a finite number of time steps. As a consequence, the Riemannian metric tensor $\mathbf{G}$ is only evaluated at these time steps during the minimisation $L(\omega)$—the gradients between the time steps are not taken into account. As a result, the approximate geodesic does not always correspond to the shortest path on the Riemannian manifold since it uses in some cases shortcuts through small regions with high gradients. This is because such small regions have a lower impact on the Riemannian distance when it is approximated by the finite sum in Eq. (5.9).

To ensure the approximated geodesic corresponds to the shortest path on the Riemannian manifold, we smooth $\mathbf{G}$ by means of a low-rank approximation, as described in (Jiang, 2014):

$$\tilde{\mathbf{G}} = \mathbf{U}_r \operatorname{diag}\left\{ \frac{s_i^3}{s_i^2 + \lambda_s} \right\}_{i=1}^{r} \mathbf{V}_r^{\mathrm{T}}, \tag{5.10}$$

where $\lambda_s > 0$ acts as a regularisation coefficient, and r is the rank of $\mathbf{G}$. The low-rank approximation in Eq. (5.10) is computed by a compact singular value decomposition $\mathbf{G} = \mathbf{U}_r \mathbf{S}_r \mathbf{V}_r^{\mathrm{T}}$: the diagonal elements of $\mathbf{S}_r$ are the singular values s_i of the Riemannian metric tensor; $\mathbf{U}_r$ and $\mathbf{V}_r$ are the corresponding left- and right-singular vectors.

In order to provide an intuitive interpretation, we show that $\tilde{\mathbf{G}}$ is equivalent to a linear smoother (Buja et al., 1989):

$$\mathbf{U}_r \operatorname{diag}\left\{ \frac{s_i^3}{s_i^2 + \lambda_s} \right\}_{i=1}^{r} \mathbf{V}_r^{\mathrm{T}} = \underbrace{\mathbf{U}_r \operatorname{diag}\left\{ \frac{s_i^2}{s_i^2 + \lambda_s} \right\}_{i=1}^{r} \mathbf{U}_r^{\mathrm{T}}}_{\text{smoothing matrix}} \mathbf{G},$$

where λ_s nonlinearly rescales the singular values of $\mathbf{G}$. This increases the gap between the smaller and the dominant singular values.

By replacing $\mathbf{G}$ in Eq. (5.8) by $\tilde{\mathbf{G}}$, we obtain the objective function for approximating geodesics:

$$\mathcal{L}_{\text{geodesic}}(\omega) = \frac{1}{T} \sum_{i=0}^{T} \sqrt{\dot{\mathbf{g}}_\omega^{\mathrm{T}}(t_i)\, \tilde{\mathbf{G}}\big(\mathbf{g}_\omega(t_i)\big)\, \dot{\mathbf{g}}_\omega(t_i)}\,,$$

with $t_{1:T} \sim \mathcal{U}(0,1)$.

5.1.2.3 *Avoiding Local Minima*

We found that the initialisation of $\mathbf{g}_\omega$ significantly influences whether the optimisation $\min_\omega \mathcal{L}_{\text{geodesic}}(\omega)$ gets stuck in a local minimum. Therefore, we propose to use a Bézier curve (De Casteljau, 1986) as initial approximation of the geodesic for pre-training $\mathbf{g}_\omega$.

The Bézier curve is determined by the starting and ending point, $\mathbf{z}_0$ and $\mathbf{z}_1$, as well as a set of C additional control points $\{\mathbf{z}_c\}_{c=1}^{C}$, each sampled from a separate uniform distribution. The means of these uniform distributions are

$$\boldsymbol{\mu}_c = \mathbf{z}_0 + \frac{c}{C+1}(\mathbf{z}_1 - \mathbf{z}_0), \quad \text{where} \quad c = 1, 2, \ldots, C.$$

The support is defined to be orthogonal to the vector $(\mathbf{z}_1 - \mathbf{z}_0)$ and to have a range of $\|\mathbf{z}_1 - \mathbf{z}_0\|/2$. This approach allows creating a variety of different curves.

In order to find a suitable initial approximation of the geodesic, we generate N Bézier curves and fit a separate neural network to each curve. The model $\mathbf{g}_\omega$ with the best validation value is used for optimising $\mathcal{L}_{\text{geodesic}}(\omega)$.

5.1.3 *Visualising the Riemannian Metric Tensor*

Analysing the likelihood function with respect to the latent representation can provide useful information about the quality of the model, for example, whether high gradients occur at the boundaries of encoded classes. For this purpose, we introduce the magnification factor (Bishop et al., 1997):

$$MF = \sqrt{\det(\mathbf{G})}, \tag{5.11}$$

which allows visualising the Riemannian metric tensor. In order to get an intuitive understanding of the magnification factor, it is helpful to first consider two *equidimensional* spaces: the rule for changing variables $\mathrm{d}\mathbf{x} = |\det(\mathbf{J})|\,\mathrm{d}\mathbf{z}$ expresses the relation between infinitesimal volumes of these two equidimensional spaces. The relation between spaces of different dimensions is expressed by $\sqrt{\det(\mathbf{J}^T\mathbf{J})}$, i.e. by the *MF*. As a consequence, the magnification factor represents the extent of change of an infinitesimal volume mapped from the latent to the observable space.

5.1.4 *Experimental Results*

We evaluate our approach by conducting a series of experiments on four datasets: a moving pendulum, the binarised MNIST digit dataset, a simulated KUKA robot arm, and real-world human motion data.

In our experiments, we analyse the topology of the latent manifold learned by an IWAE and visualise how high gradients in the likelihood

function realise the compact encoding of continuous movements or different classes. In this context, we empirically show that geodesics reflect the similarity of data in contrast to Euclidean interpolations in the latent space. In Section 5.1.4.1, we demonstrate that the Riemannian distance is a linear function of the pendulum's joint angle. Section 5.1.4.2 addresses how different classes can be distinguished in the latent space. In Sections 5.1.4.3 and 5.1.4.4, we show how geodesics can be used for efficient motion planning.

The model architectures used in our experiments can be found in Appendix A.1.3.

5.1.4.1 *Artificial Pendulum Dataset*

We created a dataset of a moving pendulum, identical to the one in Section 3.5.1. It consists of 15,000 images (see Figure 5.2) with a size of 16×16 pixels, and the joint angles are distributed uniformly in the range $[0, 2\pi)$. Therefore, the joint angle is the only degree of freedom.

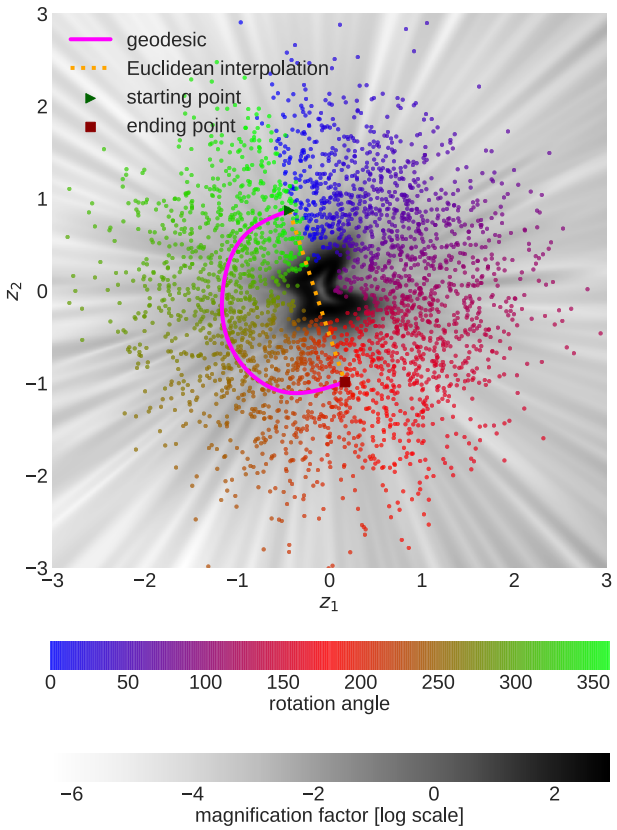


Figure 5.1: Latent representation of pendulum data learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold. In contrast to the Euclidean interpolation, the geodesic follows the data manifold as this is the shortest path on the Riemannian manifold.

Figure 5.1 shows the two-dimensional latent manifold of pendulum data learned by an IWAE. The grayscale in the background repre-

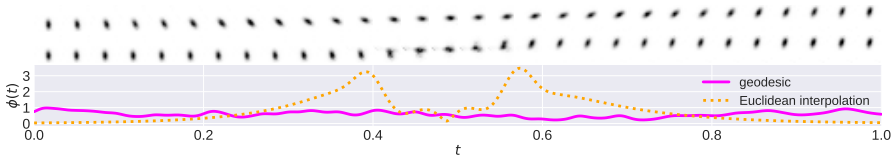


Figure 5.2: Reconstruction of the geodesic (top row) and the Euclidean interpolation (second row). The velocity $\phi(t)$ along the respective trajectories is a measure of smoothness with respect to the observable space.

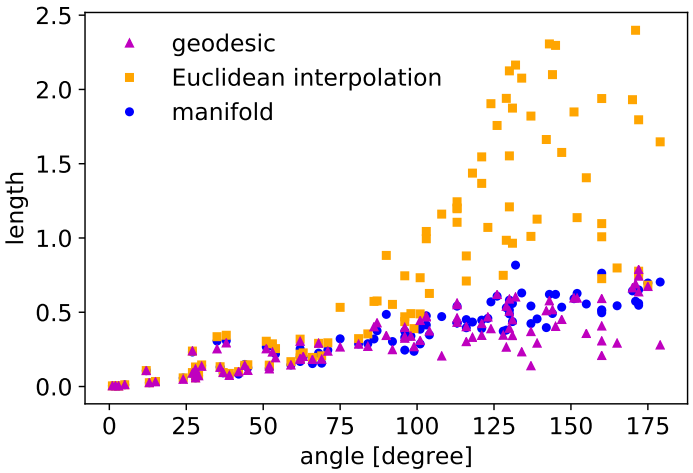


Figure 5.3: Riemannian length of geodesics, Euclidean interpolations, and encoded sequences of the moving pendulum (manifold) that functions as ground truth. The Riemannian distance (Riemannian length of the geodesic) is a linear function of the pendulum's rotation angle, i.e. an accurate similarity metric.

sents the magnification factor defined in Section 5.1.3, Eq. (5.11). It visualises the gradient of the likelihood function, i.e. the topology of the Riemannian manifold. Therefore, in contrast to the Euclidean interpolation, the geodesic follows the data manifold as this is the shortest path on the Riemannian manifold.

In order to compare the geodesic with the Euclidean interpolation, we reconstruct both trajectories in Figure 5.2 and use the velocity $\phi(t)$ defined in Eq. (5.5) for measuring the smoothness of both trajectories with respect to the observable space. In contrast to the Euclidean-interpolation-based reconstructions, the geodesic-based reconstructions (top row) show a uniform and smooth rotation of the pendulum. This is also confirmed by their Riemannian length that is 0.827 in case of the Euclidean interpolation, whereas the Riemannian distance is 0.538.

As a next step, we demonstrate that the Riemannian distance adequately reflects the similarity of different data points. Figure 5.3 shows the Riemannian distance of 100 random geodesics and Euclidean interpolations, with a rotation angle of $(0, 180]$ degrees. As ground truth, we additionally encode 100 sequences of a moving pendulum

and compute the Riemannian length of the interpolations along the encoded data points of each sequence. The idea behind this is: we know the continuous movement of the pendulum is equivalent to a continuous change of similarity in the (image) data. Therefore, our approximate geodesic follows the data manifold in Figure 5.1, and the Riemannian distance corresponds to the Riemannian length of the encoded sequences in Figure 5.3. As a result, the Riemannian distance is a linear function of the pendulum's joint angle and thus a precise similarity metric.

5.1.4.2 *Binarised MNIST*

In order to demonstrate how high gradients in the likelihood function realise the compact encoding of the data, we evaluate our model on the binarised version of MNIST (Larochelle and Murray, 2011), a benchmark dataset with ten different classes. It consists of 50,000 training, 10,000 validation, and 10,000 test images of handwritten digits (0 to 9), which are 28×28 pixels in size.

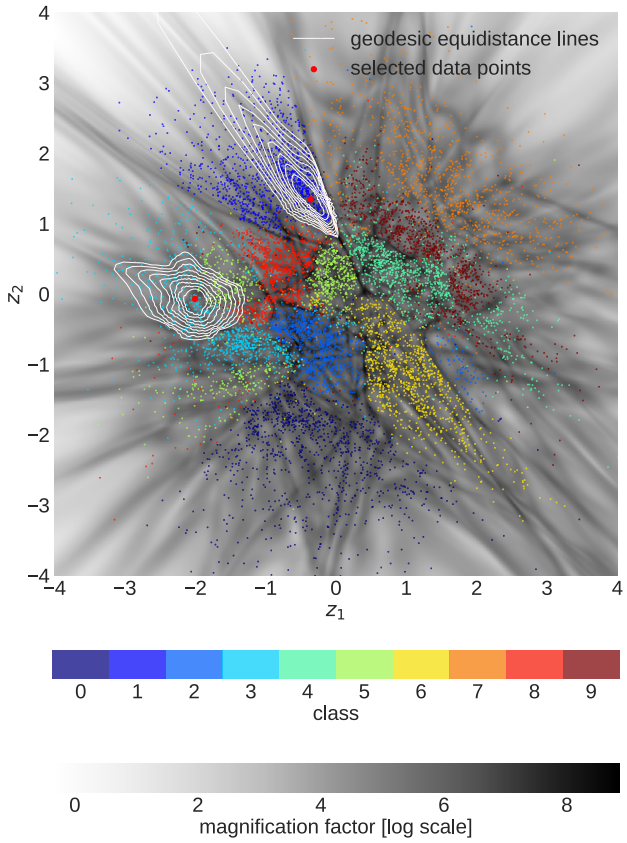


Figure 5.4: Latent representation of MNIST data learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold. The equidistance lines refer to the Riemannian distance to the selected data point. This example illustrates how Riemannian geometry allows for separating different classes (digits) in the latent space.

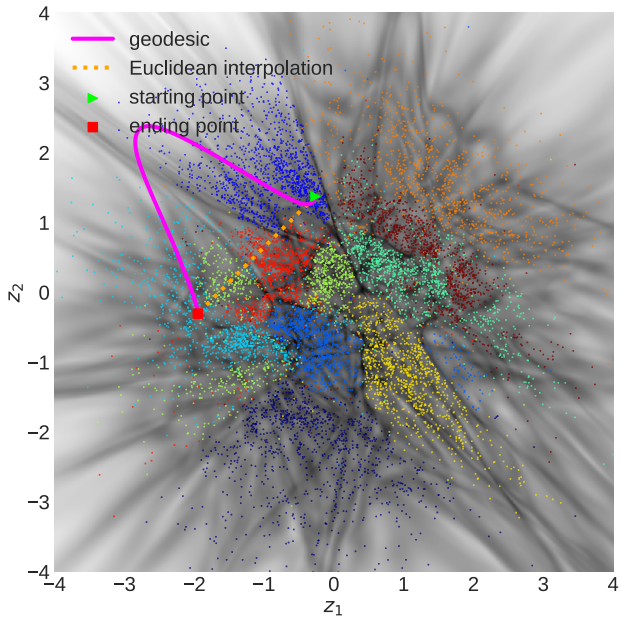


Figure 5.5: Latent representation of MNIST data learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold. In contrast to the Euclidean interpolation, the geodesic does not pass regions with high gradients that separate different classes (digits). This leads to a more accurate similarity measure as shown by the smooth interpolation in Figure 5.6.

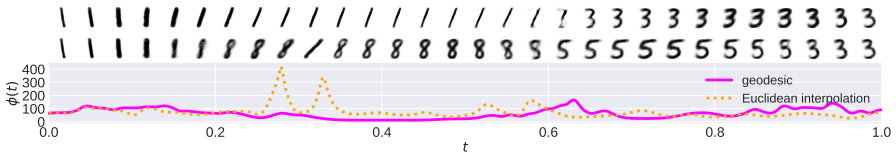


Figure 5.6: Reconstruction of the geodesic (top row) and the Euclidean interpolation (second row). The velocity $\phi(t)$ along the respective trajectories is a measure of smoothness with respect to the observable space.

Figure 5.4 shows the two-dimensional latent representation of MNIST data learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold and illustrates how high gradients in the likelihood function realise the compact encoding of the data. The equidistance lines refer to the Riemannian distance to the selected data point and illustrate how the similarity of data varies in the latent space. The course of the equidistance lines demonstrates that Riemannian geometry enables for separating classes since the Riemannian distance between data of the same class is smaller than between data of different classes. This is especially useful for state-of-the-art methods that lead to very tight boundaries with high gradients, as the IWAE. By contrast, it is mostly not possible to distinguish encodings of different classes by their Euclidean distance. This demonstrates that Euclidean distances in the latent space do not reflect the similarity of data.

In Figure 5.5, the geodesic does do not pass regions with high gradients that separate different classes. This leads to a more accurate similarity measure as shown in Figure 5.6. The geodesic-based reconstructions (top row) show a smooth change of the digit. The abrupt changes in the Euclidean-interpolation-based reconstructions are caused by high velocities at the transitions between different classes. As a result, the Riemannian length of the Euclidean interpolation is 74.3, whereas the Riemannian distance is 62.9.

5.1.4.3 *KUKA Robot Arm*

As a next step, we use our approach to interpolate the movement of a KUKA robot arm with six degrees of freedom. For this purpose, we generated a dataset where the end effector moves along a circle with a radius of 0.4 meters. It consists of 6284 training and 150 test time steps of six-dimensional joint angles. The latter corresponds to one complete circumnavigation. At each time step, the joint angles were obtained by inverse kinematics.

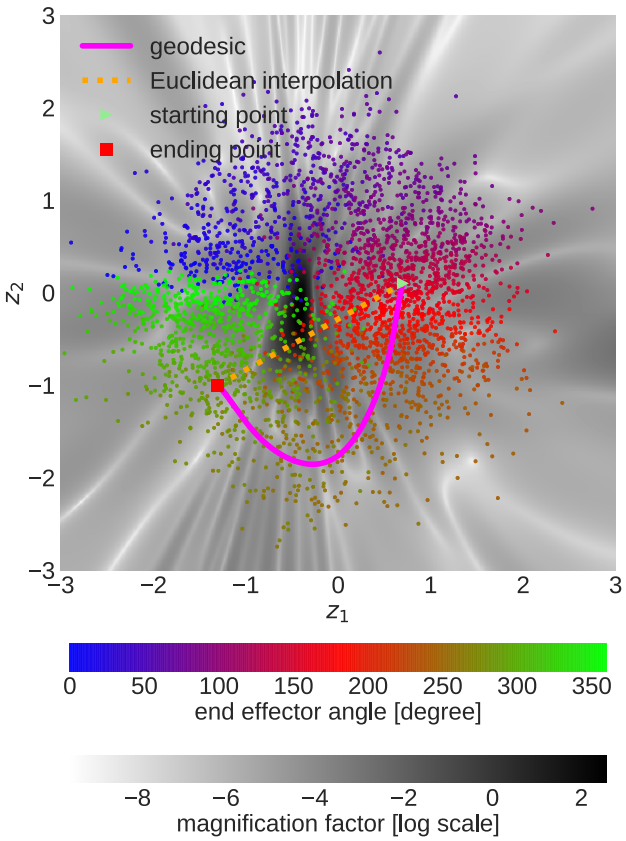


Figure 5.7: Latent representation of the robot-arm movement learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold. In contrast to the Euclidean interpolation, the geodesic follows the data manifold as this is the shortest path on the Riemannian manifold.

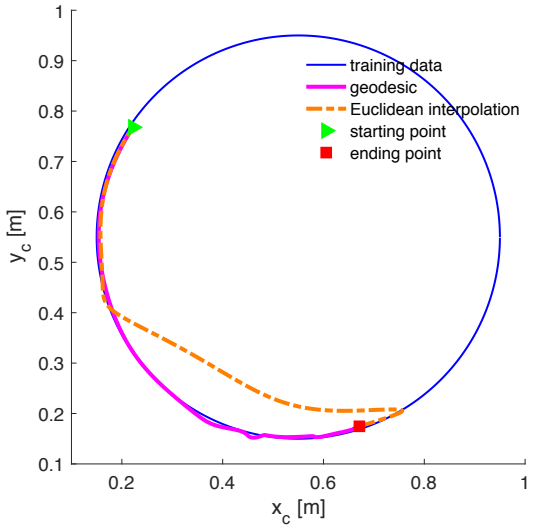


Figure 5.8: Reconstructed end-effector trajectories in the Cartesian space: x_c and y_c represent the position of the end effector (z_c is constant). The smoothness of the geodesic and Euclidean interpolation are measured by $\phi(t)$ in Figure 5.9.

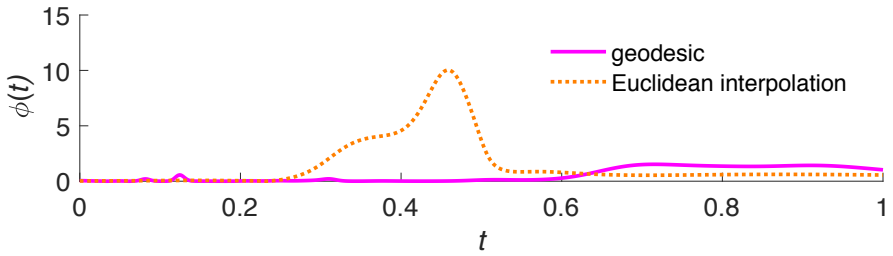


Figure 5.9: The velocity $\phi(t)$ along the geodesic and the Euclidean interpolation, depicted in Figure 5.8, is a measure of smoothness with respect to the observable space.

To efficiently plan motions, constraints were applied in prior work. For example, by constraining the end effector to move in a two- instead of three-dimensional Cartesian space (Berenson et al., 2009). By contrast, our method does not explicitly require such constraints since it is based on the learned latent manifold and thus on the training data.

In order to show that the geodesic (Figure 5.7) allows a more efficient motion planning than the Euclidean interpolation, we reconstruct both trajectories and use forward kinematics for visualising the path of the end effector in Cartesian space (Figure 5.8). As shown by the velocity in Figure 5.9, the reconstructed geodesic corresponds to a smooth movement on the data manifold. This also results in a shorter Riemannian distance of 0.54, compared to the Riemannian length of the Euclidean interpolation, which is 1.48.

In the next section, we extend this experiment to real-world human motion data with a 50-dimensional joint vector.

5.1.4.4 *Human Motion Capture Database*

The CMU Graphics Lab Motion Capture Database[1] consists of a large number of human motion recordings, which were recorded by using a motion capture system. Human subjects wear 41 markers while walking, dancing, etc. The data is pre-processed as described by N. Chen et al. (2015) such that each frame is represented by a 50-dimensional feature vector. Similar to prior work (e.g. Bitzer et al., 2008; Taylor et al., 2007), we evaluate our approach on the walking movements of subject 35, which is a subset of the dataset we used in Section 3.5.2.

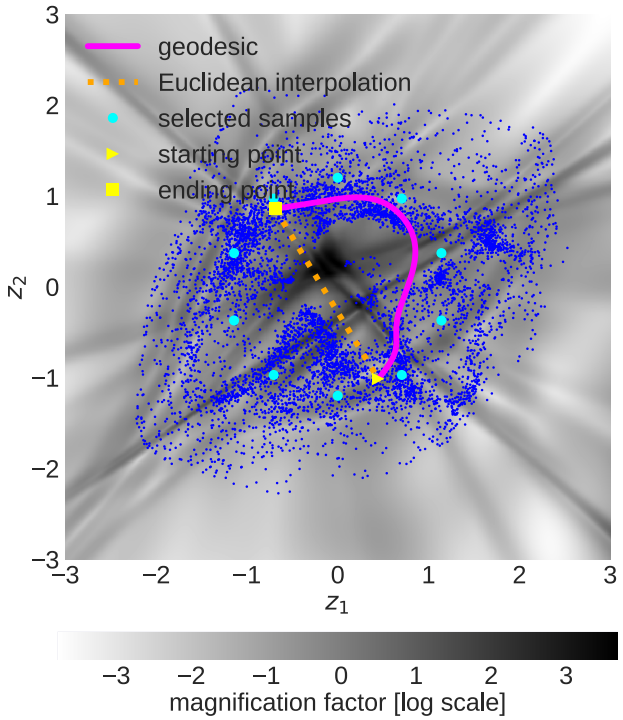


Figure 5.10: Latent representation of human motion data learned by an IWAE. The magnification factor visualises the topology of the Riemannian manifold. In contrast to the Euclidean interpolation, the geodesic follows the data manifold (blue points) as this is the shortest path on the Riemannian manifold. Reconstructions of selected samples (cyan points) can be found in Figure 5.11.

The learned latent representation of the walking movements, as well as selected reconstructions, can be found in Figures 5.10 and 5.11 The reconstruction of the geodesic (top row) shows a realistic and smooth walking movement (Figure 5.12). By contrast, the Euclidean interpolation passes two regions with high MF value, which cause abrupt changes in the movement. Apart from that, the body poses

1 mocap.cs.cmu.edu

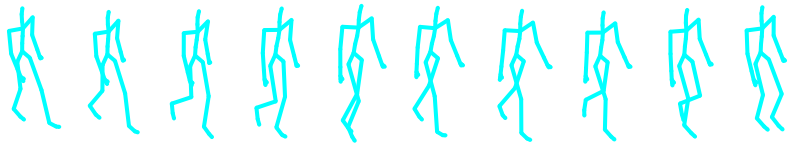


Figure 5.11: Selected reconstructions of human walking movements (cyan points in Figure 5.10). One circle in the latent space corresponds to a complete walking movement in the observation space.

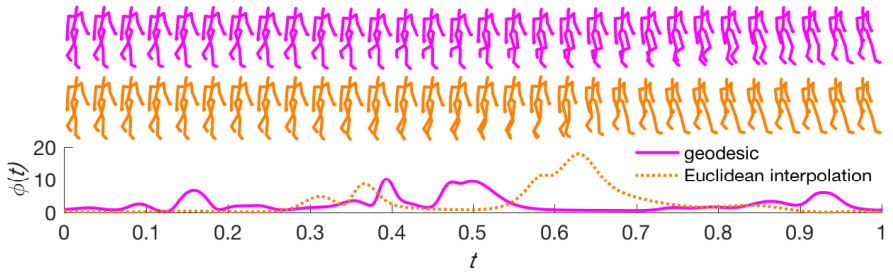


Figure 5.12: Reconstruction of the geodesic (top row) and the Euclidean interpolation (second row). The velocity $\phi(t)$ along the respective trajectories is a measure of smoothness with respect to the observable space.

hardly change. This is also reflected by the Riemannian length of the Euclidean interpolation, which is 2.89. The Riemanian distance, on the other hand, is 2.57.

5.1.5 *Summarising Discussion*

In this section, we have analysed the topology of the latent manifold learned by VAEs in order to find a distance metric that measures the similarity of data. For this purpose, we have defined the latent space as a Riemannian manifold and have visualised the gradient of the likelihood function by means of the magnification factor. We have shown that geodesics on the latent manifold reflect, in contrast to Euclidean interpolations, the similarity of data since their computation is based on the Riemannian metric tensor and thus on the gradient of the likelihood function. As a result, we have demonstrated that the Riemannian distance—i.e. the Riemannian length of the geodesic—is an accurate metric of measuring the similarity of data.

In Section 5.1.4.1, we have shown that the Riemannian distance is a linear function of the pendulum's joint angle. Section 5.1.4.2 has addressed how the Riemannian distance can be used to identify different classes in the latent space. In Sections 5.1.4.3 and 5.1.4.4, we have demonstrated how geodesics can be applied for efficient motion planning.

The capability of our approach for defining a similarity metric by means of the learned latent representation has formed the basis for

further work: in (N. Chen, Klushyn, Paraschos, Benbouzid and Smagt, 2018), we have proposed a novel algorithm for active learning and have shown its utility for generating smooth movements on the example of an anthropomorphic robot arm.

The main limitation of our approach, however, is the approximation of the geodesics, which involves a computationally expensive non-convex optimisation based on the Jacobian of the likelihood function. In (N. Chen, Ferroni, Klushyn, Paraschos, Bayer and Smagt, 2019), we have shown how the major computational demand of approximating geodesics can be sidestepped by solving a related graph-based shortest path problem instead. This entails pre-computing a graph in the latent space, where the encoded data points are the nodes and the edge weights are defined by the Riemannian length of the Euclidean interpolation between these nodes.

Nevertheless, these two approaches of approximating geodesics are not always applicable in practice. For example, in case of object tracking, the similarity metric has to be evaluated in real-time on new data. In Section 5.2, we address this problem by learning latent representations that act directly as Euclidean similarity metric.

5.2 THE LATENT SPACE AS EUCLIDEAN SIMILARITY METRIC

The methods and experimental results discussed in this section have been previously published in (N. Chen, Klushyn, Ferroni, Bayer and Smagt, 2020). Section 5.2.3.3 is based on revised text from this publication.

VAEs regularise the latent space to be compact, i.e. to remove low-density regions. As a consequence, the learned latent representation cannot be directly used as a distance metric for measuring the similarity of data. For this reason, we have defined the latent space of VAEs as a Riemannian manifold in Section 5.1. This introduces a distance metric, referred to as the Riemannian distance, that accurately reflects the similarity of data by taking into account the gradient of the likelihood function. Concurrently, computing the Riemannian distance requires solving a computationally expensive non-convex optimisation problem, as discussed in Section 5.1.2. Hence, this approach is not suitable for applications in real-time environments.

The aforementioned issue can be addressed by learning latent representations where the similarity of data is directly reflected by the Euclidean distance. This implies that the Euclidean distance in the latent space is proportional to the Riemannian distance, which would make similarity measures computationally very efficient. We refer to such latent manifolds that are isometric to the Euclidean space as *flat latent manifolds*.

In order to learn flat latent manifolds, we propose a method that combines the VHP-VAE introduced in Chapter 3 with the Riemannian-

geometry approach discussed in Section 5.1. Due to the empirical Bayes prior distribution, the VHP-VAE is capable of learning complex latent representations that reflect the topology of the data. To incentivise a distance-preserving mapping between the latent and observable space, we extend the objective function of the VHP-VAE by a regularisation term which regularises the Riemannian metric tensor towards a scaled identity matrix. As a consequence, the model learns a latent representation where the Euclidean metric is a proxy for the similarity of data. This results in a computationally efficient distance metric that is practical for applications in real-time scenarios. In the following, we introduce our approach in detail:

- In Section 5.2.1, we define flat latent manifolds in the context of VAEs and Riemannian geometry.
- In Section 5.2.2, we introduce a method for learning flat latent manifolds, which is an extension of the VHP-VAE (Chapter 3) and the associated constrained optimisation algorithm REWO.
- In Section 5.2.3, we evaluate our proposed method. This includes experiments on real-world human motion data, on binarised MNIST, and on the MOT16 object-tracking database.

5.2.1 *Flat Latent Manifolds*

In this section, we specify under which conditions the latent representation of data learned by VAEs can act as Euclidean similarity metric. The idea is to obtain a computationally efficient similarity measure that avoids solving the non-convex optimisation problem discussed in Section 5.1.3.

In Section 5.1.1, we defined the latent space of VAEs as a Riemannian manifold. This allows computing the Riemannian length of a trajectory in the latent space $\gamma : [0, 1] \to \mathbb{R}^{D_z}$ with respect to the observable space:

$$L(\gamma) = \int_0^1 \sqrt{\dot{\gamma}(t)^{\mathrm{T}}\, \mathbf{G}\big(\gamma(t)\big)\, \dot{\gamma}(t)}\, \mathrm{d}t, \tag{5.12}$$

where $\mathbf{G}$ is the Riemannian metric tensor, and $\dot{\gamma}(t)$ is the velocity. As demonstrated in Section 5.1.4, the Riemannian distance $D = \min_\gamma L(\gamma)$ between two data points is an accurate similarity metric.

If it is proportional to the Riemannian distance, the Euclidean distance in the latent space therefore reflects the similarity of data:

$$\|\mathbf{z}(1) - \mathbf{z}(0)\|_2 \;\propto\; \min_\gamma L(\gamma).$$

This presupposes that the Riemannian metric tensor in Eq. (5.12) corresponds to a scaled identity matrix, i.e. to a scaled Euclidean metric:

$$\mathbf{G}(\mathbf{z}) = c^2\, \mathbb{1} \quad \forall \quad \mathbf{z} \in \mathbb{R}^{D_z}, \tag{5.13}$$

where c is the scaling factor. As a consequence, the geodesic $\gamma_{\text{geodesic}} = \text{argmin}_{\gamma} L(\gamma)$ is identical to the Euclidean interpolation. Furthermore, the velocity $\dot{\gamma}(t)$ is constant due to the flat topology of the Riemannian manifold defined by $\mathbf{G}$. We refer to a latent manifold with such properties therefore as *flat latent manifold* (Lee, 2006).

As a result, the latent space of VAEs can act as Euclidean similarity metric if the generative model fulfils the condition in Eq. (5.13). In the next section, we discuss how to learn flat latent manifolds with VAEs.

5.2.2 *Learning Flat Latent Manifolds With Variational Autoencoders*

In order to learn flat latent manifolds, the VAE has to be capable of modelling latent representations that correspond to the topology of the data manifold. Moreover, the Riemannian metric tensor, determined by the Jacobian of the likelihood function, has to be a scaled Euclidean metric, as described in Section 5.1.1. For this reason, we propose to apply the hierarchical empirical Bayes prior introduced in Chapter 3—in order to enable our model to learn complex latent representations—and to regularise the curvature of the likelihood function such that $\mathbf{G} \propto \mathbb{1}$.

As demonstrated in Section 3.5, the VHP-VAE learns latent representations that reflect the topology of the data manifold. However, it is not guaranteed that the Euclidean distance between encoded data in the latent space is a sufficient distance metric with respect to the observable space. This is because it is not incentivised by the objective function (Eq. (3.9)). Therefore, we extend the objective function of the VHP-VAE by a regularisation term.

5.2.2.1 *Mixup-Based Regularisation of the Jacobian*

VAEs map the latent space through a continuous function $\mathbf{f}_\theta(\mathbf{z})$, the decoder, to the observable space. More precisely: the decoder determines the parameters of the likelihood function. The Riemannian metric tensor is defined by $\mathbf{G}_\theta(\mathbf{z}) = \mathbf{J}_\theta(\mathbf{z})^{\mathrm{T}}\mathbf{J}_\theta(\mathbf{z})$, where $\mathbf{J}$ is the Jacobian of the decoder, as stated in Eq. (5.4).

For regularising $\mathbf{G}$ towards a scaled Euclidean metric, we use a stochastic approximation (first order Taylor series expansion) of the Jacobian (Rifai et al., 2011b):

$$\mathbf{J}_\theta^{(i)}(\mathbf{z}) = \lim_{\sigma \to 0} \frac{1}{\sigma} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\sigma^2)} \left[\mathbf{f}_\theta(\mathbf{z} + \epsilon\, \mathbf{e}_i) - \mathbf{f}_\theta(\mathbf{z}) \right],$$

where $\mathbf{J}_\theta^{(i)} \in \mathbb{R}^{D_{\mathbf{x}}}$ is the Jacobian of the i-th latent dimension, and $\mathbf{e}_i$ is the respective standard basis vector. This approximation method leads to a faster computation of the gradient and avoids the second-order-derivative problem of piece-wise-linear layers in neural networks.

Another important detail is that the impact of the regularisation is limited to regions in the latent space where data is available. To

address this issue, we propose to use *mixup* (Zhang et al., 2018), a data-augmentation method that was introduced in the context of supervised learning. We integrate this method with the VAE framework by applying it to latent variables. It allows augmenting data by interpolating between two encoded data points $\mathbf{z}_i$ and $\mathbf{z}_j$:

$$\mathbf{g}(\mathbf{z}_i, \mathbf{z}_j, \rho) = (1-\rho)\,\mathbf{z}_i + \rho\,\mathbf{z}_j, \tag{5.14}$$

with $\mathbf{x}_{i,j} \sim p_{\mathcal{D}}(\mathbf{x})$, $\mathbf{z}_{i,j} \sim q_\phi(\mathbf{z}|\mathbf{x}_{i,j})$, and $\rho \sim p(\rho) = U(-\rho_0, 1+\rho_0)$. In contrast to (Zhang et al., 2018), where the data augmentation is limited to solely convex combinations due to $\rho_0 = 0$, we define $\rho_0 > 0$ to take into account the outer edge of the data manifold. As a result, we obtain the following regularisation term:

$$\begin{aligned}&\mathcal{F}_{\text{reg}}\left(\theta, \phi; c^2\right)\\ &\quad= \mathbb{E}_{\mathbf{x}_{i,j}\sim p_{\mathcal{D}}(\mathbf{x})}\,\mathbb{E}_{\mathbf{z}_{i,j}\sim q_\phi(\mathbf{z}|\mathbf{x}_{i,j})}\,\mathbb{E}_{\rho\sim p(\rho)}\left[\left\|\mathbf{G}_\theta\big(\mathbf{g}(\mathbf{z}_i, \mathbf{z}_j, \rho)\big) - c^2\,\mathbb{1}\right\|_F^2\right].\end{aligned}$$

Furthermore, we define the scaling factor, inspired by batch normalisation, as the mean of the diagonal elements of $\mathbf{G}$ over batch samples and latent dimensions $D_{\mathbf{z}}$:

$$c^2 = \mathbb{E}_{\mathbf{x}_{i,j}\sim p_{\mathcal{D}}(\mathbf{x})}\,\mathbb{E}_{\mathbf{z}_{i,j}\sim q_\phi(\mathbf{z}|\mathbf{x}_{i,j})}\,\mathbb{E}_{\rho\sim p(\rho)}\left[\frac{1}{D_{\mathbf{z}}}\operatorname{tr}\Big(\mathbf{G}_\theta\big(\mathbf{g}(\mathbf{z}_i, \mathbf{z}_j, \rho)\big)\Big)\right]. \tag{5.15}$$

An ablation study regarding the impact of *mixup* (Eq. (5.14)) and the normalisation of the scaling factor (Eq. (5.15)) can be found in Section 5.2.3.1.

5.2.2.2 *Flat Manifold Variational Autoencoder*

In the next step, the constrained optimisation approach introduced in Chapter 3 is extended for learning flat latent manifolds. For this purpose, we integrate $\mathcal{F}_{\text{reg}}\left(\theta, \phi; c^2\right)$ with the Lagrangian defined in Eq. (3.9). The resulting model is referred to as flat manifold variational autoencoder (FMVAE).

As detailed in Section 3.3, the VHP approximates the optimal empirical Bayes prior, $p^*(\mathbf{z}) \approx p_\Theta(\mathbf{z})$, by optimising an importance-weighted bound on the aggregated posterior. This introduces a lower bound on the ELBO since the VHP defines an upper bound on the KL:

$$\begin{aligned}&\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\Big[\mathrm{KL}\big(q_\phi(\mathbf{z}|\mathbf{x})\,\|\,p_\Theta(\mathbf{z})\big)\Big] \leq \mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi)\\ &= \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\,\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\,\mathbb{E}_{\boldsymbol{\zeta}_{1:K}\sim q_\Phi(\boldsymbol{\zeta}|\mathbf{z})}\left[\log q_\phi(\mathbf{z}|\mathbf{x}) - \log\frac{1}{K}\sum_{k=1}^{K}\frac{p_\Theta(\mathbf{z}|\boldsymbol{\zeta}_k)\,p(\boldsymbol{\zeta}_k)}{q_\Phi(\boldsymbol{\zeta}_k|\mathbf{z})}\right],\end{aligned}$$

where K is the number of importance samples, and $p(\boldsymbol{\zeta})$ is the standard normal distribution.

By combining $\mathcal{F}_{\text{VHP-VAE}}(\phi, \Theta, \Phi)$ with the regularisation term, we obtain a looser upper bound on the KL:

$$\mathcal{F}\left(\theta, \phi, \Theta, \Phi; c^2\right) = \mathcal{F}_{\text{VHP-VAE}}\left(\phi, \Theta, \Phi\right) + \eta\, \mathcal{F}_{\text{reg}}\left(\theta, \phi; c^2\right),$$

where η is a hyper parameter determining the impact of the regularisation. We follow the line of argument in Chapter 3 and reformulate the resulting ELBO as the Lagrangian of a constrained optimisation problem:

$$\begin{aligned}&\mathcal{L}_{\text{VHP-FMVAE}}\left(\theta, \phi, \Theta, \Phi; \lambda, c^2\right)\\ &\quad= \mathcal{F}\left(\theta, \phi, \Theta, \Phi; c^2\right) + \lambda\left(\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\right] - \kappa^2\right),\end{aligned} \tag{5.16}$$

with the optimisation objective $\mathcal{F}\left(\theta, \phi, \Theta, \Phi; c^2\right)$ and the inequality constraint $\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\right] \leq \kappa^2$. The Lagrange multiplier λ can be viewed as a weighting term for $\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})$, which is defined as the reconstruction-error-related term in $-\log p_\theta(\mathbf{x}|\mathbf{z})$.

5.2.2.3 *Optimisation Algorithm*

To optimise the Lagrangian $\mathcal{L}_{\text{VHP-FMVAE}}\left(\theta, \phi, \Theta, \Phi; \lambda, c^2\right)$ in Eq. (5.16), we propose Algorithm 4, which is an extension of REWO (Algorithm 2) in Section 3.3. It divides the constrained optimisation process into two phases: an initial and a main phase. Like in Chapter 3, we use the β-parametrisation of the Lagrange multiplier, $\beta = {}^1/_\lambda$, to be in line with previous literature (e.g. Higgins et al., 2017; Sønderby et al., 2016).

In the initial phase, the goal is to reduce the reconstruction error in order to force the model to learn an informative encoding of the data. For this purpose, we define $\beta \ll 1 (\lambda \gg 1)$ and optimise the bound solely with respect to (θ, ϕ). Furthermore, β is not updated as long as the inequality constraint $\mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\mathcal{C}_\theta(\mathbf{x}, \mathbf{z})\right] \leq \kappa^2$ is not fulfilled.

The main phase starts as soon as the constraint is satisfied, i.e. a meaningful encoding is achieved. This serves as a starting point for learning a flat latent manifold and the corresponding empirical Bayes prior distribution. To do this, we optimise all parameters $(\theta, \phi, \Theta, \Phi)$ jointly and start to update β by applying the update scheme introduced in Section 3.3:

$$\beta_t = \beta_{t-1} \cdot \exp\left[\nu \cdot f_\beta\left(\beta_{t-1}, \mathcal{C}_t - \kappa^2; \tau\right) \cdot \left(\mathcal{C}_t - \kappa^2\right)\right],$$

where f_β is defined as

$$f_\beta(\beta, \delta; \tau) = \left(1 - H(\delta)\right) \cdot \tanh\left(\tau \cdot (\beta - 1)\right) - H(\delta).$$

H is the Heaviside function, and τ is a slope parameter. A visualisation of this update scheme can be found in Figure 3.1a.

Algorithm 4 REWO with *mixup*-based regularisation of the Jacobian

Initialise $t = 1$
Initialise $\beta \ll 1$
Initialise InitialPhase = True
while training **do**
  Read current data batch $\mathbf{x}_{\text{ba}}$
  Sample from variational posterior $\mathbf{z}_{\text{ba}} \sim q_\phi(\mathbf{z}|\mathbf{x}_{\text{ba}})$
  Shuffle samples from variational posterior $\mathbf{z}'_{\text{ba}} = \text{shuffle}(\mathbf{z}_{\text{ba}})$
  Sample from uniform $\rho_{\text{ba}} \sim p(\rho)$
  Augment data $\mathbf{z}^{\text{aug}}_{\text{ba}} = \mathbf{g}(\mathbf{z}_{\text{ba}}, \mathbf{z}'_{\text{ba}}, \rho_{\text{ba}})$
  Compute $c^2 = \frac{1}{\text{batch\_size}} \sum_i \frac{1}{D_\mathbf{z}} \text{tr}\big(\mathbf{G}(\mathbf{z}_i^{\text{aug}})\big)$
  Compute $\hat{\mathcal{C}}_{\text{ba}} = \frac{1}{\text{batch\_size}} \sum_i \mathcal{C}_\theta(\mathbf{x}_i, \mathbf{z}_i)$
  Compute $\hat{\mathcal{C}}_t = (1-\alpha) \cdot \hat{\mathcal{C}}_{\text{ba}} + \alpha \cdot \hat{\mathcal{C}}_{t-1}, \quad (\hat{\mathcal{C}}_0 = \hat{\mathcal{C}}_{\text{ba}})$
  if $\hat{\mathcal{C}}_t < \kappa^2$ **then**
    InitialPhase = False
  end if
  if InitialPhase **then**
    Optimise $\mathcal{L}_{\text{VHP-FMVAE}}\left(\theta, \phi, \Theta, \Phi; \beta, c^2\right)$ w.r.t. θ, ϕ
  else
    $\beta \leftarrow \beta \cdot \exp\left[\nu \cdot f_\beta\left(\beta, \hat{\mathcal{C}}_t - \kappa^2; \tau\right) \cdot \left(\hat{\mathcal{C}}_t - \kappa^2\right)\right]$
    Optimise $\mathcal{L}_{\text{VHP-FMVAE}}\left(\theta, \phi, \Theta, \Phi; \beta, c^2\right)$ w.r.t. $\theta, \phi, \Theta, \Phi$
  end if
  $t \leftarrow t + 1$
end while

As β increases, the impact of the regularisation term $\mathcal{F}_{\text{reg}}(\theta, \phi, c^2)$ becomes greater. Hence, the latent space is converted smoothly into a flat manifold. This regularisation method exploits the fact that the latent representation learned by the VHP-VAE (without regularisation) already reflects the topology of the data manifold, as shown in Section 3.5. Thus, the regularisation fine-tunes the likelihood function. As a result, the latent space of VHP-FMVAEs can act as Euclidean similarity metric, which we demonstrate in Section 5.2.3.

5.2.3 *Experimental Results*

We evaluate our approach by conducting a series of experiments on three datasets: human motion data, the binarised MNIST digit dataset, and MOT16. The latter is a benchmark dataset for multiple object tracking in real-world scenarios.

In our experiments, we verify that Euclidean distances in the latent space of VHP-FMVAEs reflect the Riemannian distance, i.e. the similarity of data. Furthermore, we compare the VHP-FMVAE with the VHP-VAE to demonstrate the effect of the regularisation term. The topology of the learned latent manifolds is analysed by means of the Riemannian metric tensor. In this context, we use the condition number and the magnification factor to evaluate whether $\mathbf{G} \approx c^2 \mathbb{1}$.

The condition number is defined as $k(\mathbf{G}) = \frac{s_{\max}(\mathbf{G})}{s_{\min}(\mathbf{G})}$, where $s_{\max}$ and $s_{\min}$ are maximal and minimal singular values of $\mathbf{G}$. Therefore, the condition number measures the ratio between the highest and lowest elongation of the latent space with respect to the observable space. The magnification factor $MF = \sqrt{\det(\mathbf{G})}$ represents the extent of change of an infinitesimal volume mapped from the latent to the observable space. A detailed introduction can be found in Section 5.1.3. To compare magnification factors of different models, we normalise them by their mean. In case of a flat latent manifold, the condition number and the normalised *MF* equal approximately to 1.

In Sections 5.2.3.1 and 5.2.3.2, we analyse the topology of the learned latent manifolds by means of the condition number and the magnification factor—and show that the VHP-FMVAE learns flat latent manifolds. In Section 5.2.3.3, we use our similarity metric for re-identification and object tracking, where the performance of our unsupervised approach nears that of state-of-the-art supervised methods.

The model architectures used in our experiments can be found in Appendix A.1.4.

5.2.3.1 *Human Motion Capture Database*

We select five different movements to evaluate our method on the CMU Graphics Lab Motion Capture Database[2]: walking (subject 35), jogging (subject 35), balancing (subject 49), punching (subject 143), and kicking (subject 74). After pre-procession, each frame is represented by a 50-dimensional feature vector. Note that the dataset is not balanced: the walking subset, for example, contains more data points than the jogging subset.

CONDITION NUMBER AND NORMALISED MAGNIFICATION FACTOR In order to compare the VHP-FMVAE with the VHP-VAE, we examine the topology of the learned latent manifolds by computing the condition number and normalised *MF* shown in Figure 5.13. Both metrics are based on 3,000 samples, and they indicate the VHP-FMVAE has learned a latent space that is close to a flat manifold, in contrast to the VHP-VAE. Furthermore, we demonstrate the impact of the scaled identity term $c^2\,\mathbb{1}$ (Eq. (5.15)) and of *mixup* (Eq. (5.14)). For this purpose, we provide an ablation study where the VHP-FMVAE is trained without applying *mixup* and without including $c^2\,\mathbb{1}$ in the regularisation term. The corresponding latent representations are shown in Figures 5.14 and 5.15.

VISUALISATIONS OF THE LEARNED LATENT MANIFOLDS Figure 5.14 depicts the latent representations of human motion data learned by the VHP-FMVAE and the VHP-VAE. The magnification

2 mocap.cs.cmu.edu

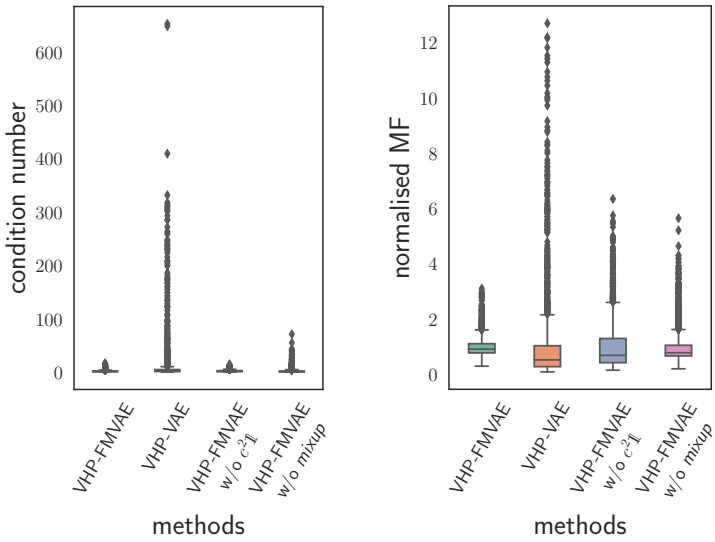


Figure 5.13: Human motion data: if the condition number and the normalised *MF* correspond approximately to 1, this implies that $\mathbf{G} \propto \mathbb{1}$. The box plots are based on 3,000 samples.

factor visualises the topology of the latent manifold. To enable a visual comparison of the two models, we define the upper range of the colour bar related to the VHP-FMVAE by

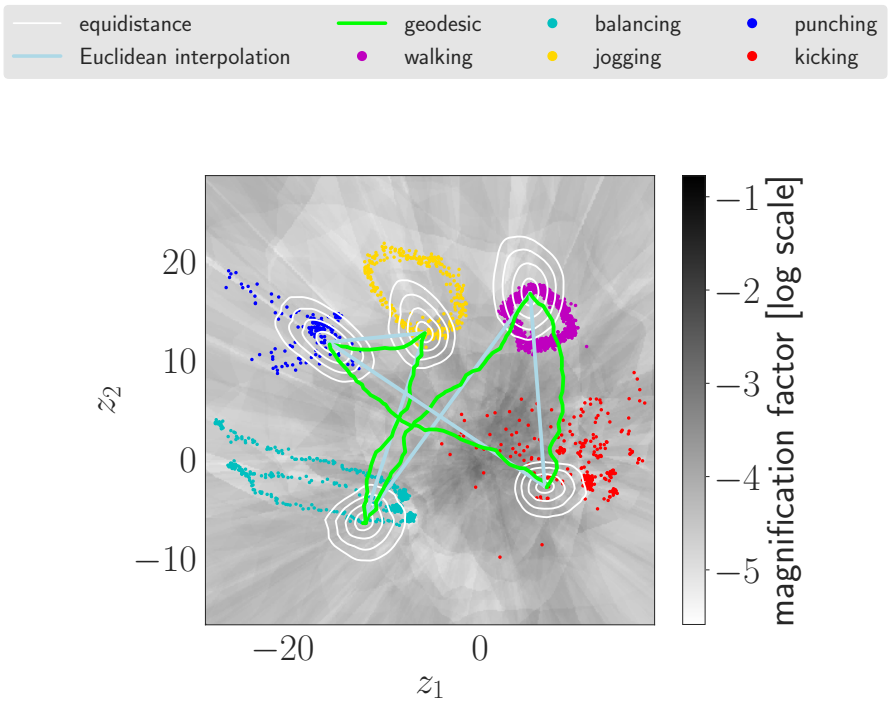
$$MF^{\max}_{\text{VHP-FMVAE}} = \frac{\max\big(MF_{\text{VHP-VAE}}(\text{grid})\big) \cdot \text{mean}\big(MF_{\text{VHP-FMVAE}}(\text{data})\big)}{\text{mean}\big(MF_{\text{VHP-VAE}}(\text{data})\big)},$$

where $\max\big(MF_{\text{VHP-VAE}}(\text{grid})\big)$ is the maximum value in the latent space depicted in Figure 5.14b, and $MF(\text{data})$ is computed with encoded training data. Here, the idea is to normalise $MF^{\max}_{\text{VHP-FMVAE}}$ with respect to both models.

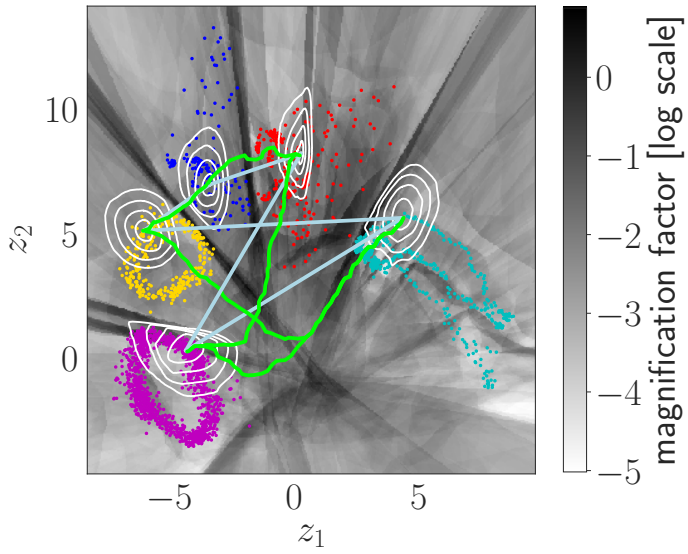
The equidistance lines refer to the Riemannian distance to the respective encoded data point and illustrate how the similarity of data varies in the latent space. In case of the VHP-FMVAE, the contour lines are homogeneous circles. This is because the regularisation term smoothens the latent space towards $\mathbf{G} \approx c^2\,\mathbb{1}$. Without regularisation, we obtain a latent space with regions of high *MF* values that lead to distorted contour lines.

As a next step, we analyse the latent representations of the individual movements. Jogging, for example, is a large-range movement compared with walking. In Figure 5.14a (VHP-FMVAE), the latent representation of jogging therefore occupies a larger area than walking. By contrast, in case of the VHP-VAE (Figure 5.14b), the latent representation of walking occupies a larger area than the one of jogging.

The comparison of Figures 5.14a and 5.15a illustrates the effect of *mixup* (Eq. (5.14)). Without data augmentation, the influence of the regularisation term is limited to regions where data is available. This

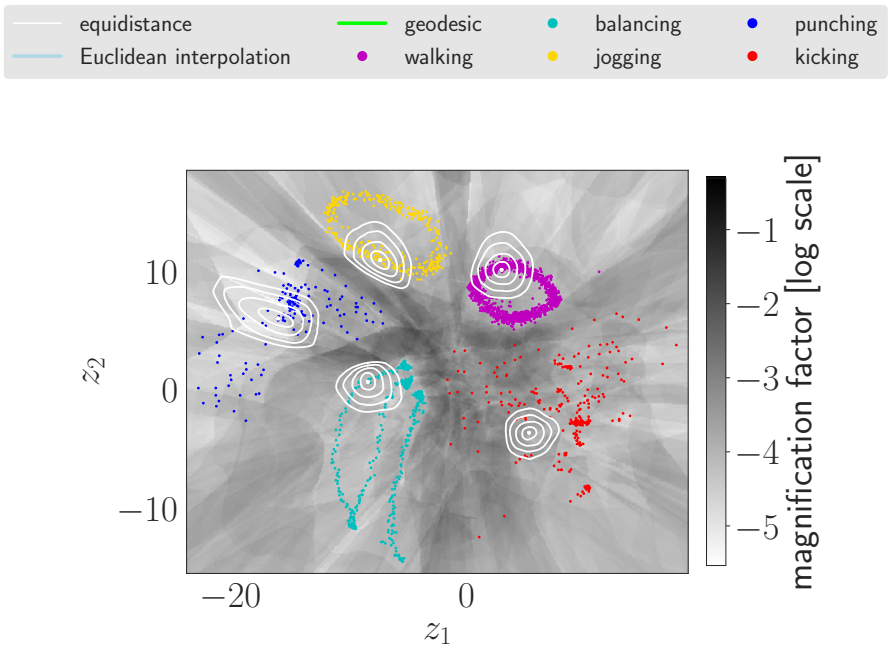


(a) VHP-FMVAE

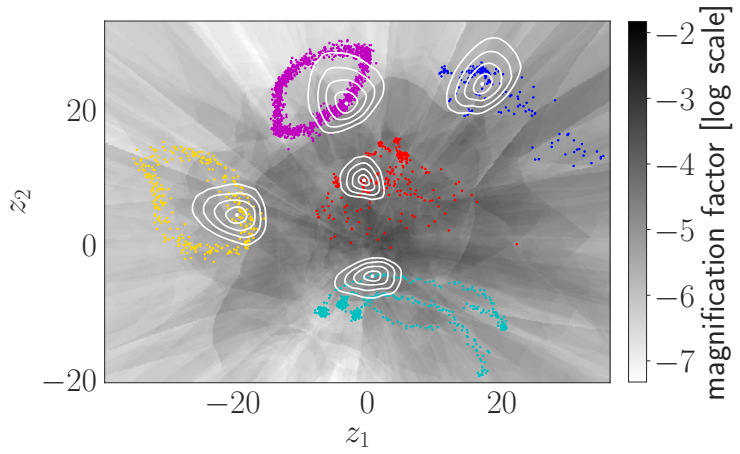


(b) VHP-VAE

Figure 5.14: Latent representation of human motion data: the magnification factor visualises the topology of the Riemannian manifold. The equidistance lines refer to the Riemannian distance to the respective encoded data point. Note: round, homogeneous contour plots indicate that $\mathbf{G} \propto \mathbb{1}$. Reconstructions of the Euclidean interpolations and the geodesics can be found in Figure 5.16. A statistical evaluation of the length ratios is summarised in Table 5.1.



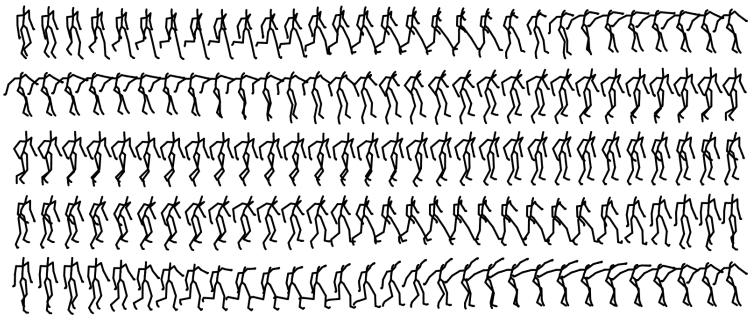
(a) VHP-FMVAE without *mixup*



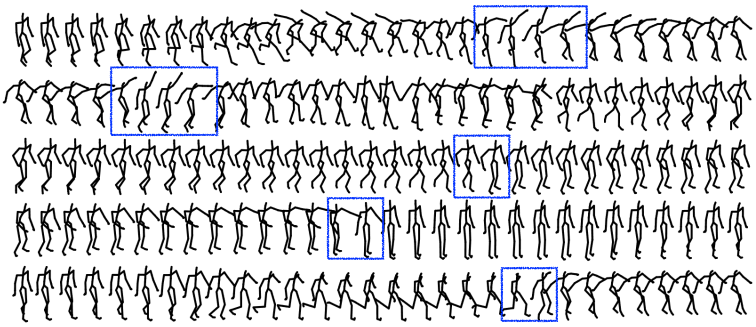
(b) VHP-FMVAE without $c^2\,\mathbb{1}$

Figure 5.15: Latent representation of human motion data: the magnification factor visualises the topology of the Riemannian manifold. The equidistance lines refer to the Riemannian distance to the respective encoded data point. The comparison to Figure 5.14a illustrates the effect of *mixup* and $c^2\,\mathbb{1}$ on the learned latent manifold.

is verified by high *MF* values between the different movements. As a further experiment, we demonstrate the impact of the identity term $c^2\,\mathbb{1}$ (Eq. (5.15)) on the latent manifold learned by the VHP-FMVAE (see Figures 5.14a and 5.15b). Without the identity term, the Riemmanian metric tensor is regularised towards zero—with the consequence that the model does not learn a flat latent manifold.



(a) VHP-FMVAE



(b) VHP-VAE

Figure 5.16: Human motion data: reconstructions of the Euclidean interpolations in Figure 5.14. Note that the blue boxes mark discontinuities in the motions.

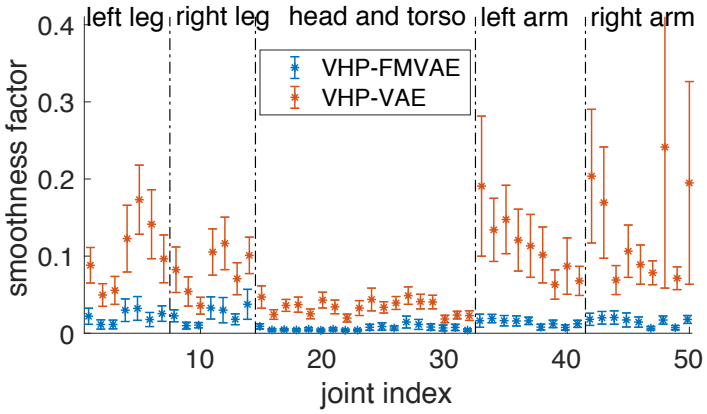


Figure 5.17: Smoothness measure of reconstructed human motions. The smoothness measure is based on 100 reconstructed Euclidean interpolations in the latent space. We display the mean and standard deviation for each joint. Note that the smaller the value, the smoother the movement.

INTERPOLATIONS BETWEEN DIFFERENT MOVEMENTS Figure 5.16 shows the reconstructions of the Euclidean interpolations in Figure 5.14. The blue boxes mark discontinuities in the reconstructed

human motions. The reconstructions demonstrate that Euclidean interpolations in the latent space of the VHP-VAE contain abrupt changes in the motions, as indicated by the high *MF* values in Figure 5.14b.

In order to obtain an accurate statistic, we randomly sample 100 data-point pairs in the latent space and reconstruct the Euclidean interpolation between each pair. Figure 5.17 depicts the average smoothness factor for each joint. The smoothness factor is defined as the second derivative of the joint with respect to time. As a result, the Euclidean interpolations in the latent space of the VHP-FMVAE lead to smoother movements compared to the VHP-VAE.

VERIFICATION OF THE EUCLIDEAN SIMILARITY METRIC In the following, we verify whether Euclidean interpolations in the latent space indeed correspond to geodesics. For this purpose, we approximate the geodesics with the graph-based approach introduced by N. Chen et al. (2019). The graph consists of 14,400 nodes with 12 neighbours each. The nodes are sampled from a two-dimensional uniform distribution representing the latent space.

In our experiment, we compare 100 geodesics to their corresponding Euclidean interpolations. An example of five geodesics with their corresponding Euclidean interpolations can be found in Figure 5.14. Table 5.1 displays the ratio between Euclidean distances in the latent space and the corresponding lengths of the geodesics. Furthermore, we list the ratio between Riemannian lengths of Euclidean interpolations and Riemannian distances (Riemannian lengths of geodesics). If the length ratios are close to 1, Euclidean interpolations correspond to geodesics. As a result, Euclidean distances in the latent space of the VHP-FMVAE reflect Riemannian distances, i.e. the similarity of data.

Table 5.1: Human motion data: verifying that the latent space can act as Euclidean similarity metric. The table shows the length ratio of Euclidean interpolations to their corresponding geodesics (latent space). Additionally, we list the ratio between Riemannian lengths of Euclidean interpolations and Riemannian distances (observable space).

method	length ratio (latent space)	length ratio (observable space)
VHP-FMVAE	**0.93 ± 0.03**	**1.02 ± 0.06**
VHP-VAE	0.82 ± 0.10	1.23 ± 0.20

5.2.3.2 *Binarised MNIST*

Similar to Section 5.1.4.2, we evaluate our model on the binarised version of MNIST (Larochelle and Murray, 2011). This benchmark dataset consists of 50,000 training, 10,000 validation, and 10,000 test images of handwritten digits (0 to 9), which are 28×28 pixels in size.

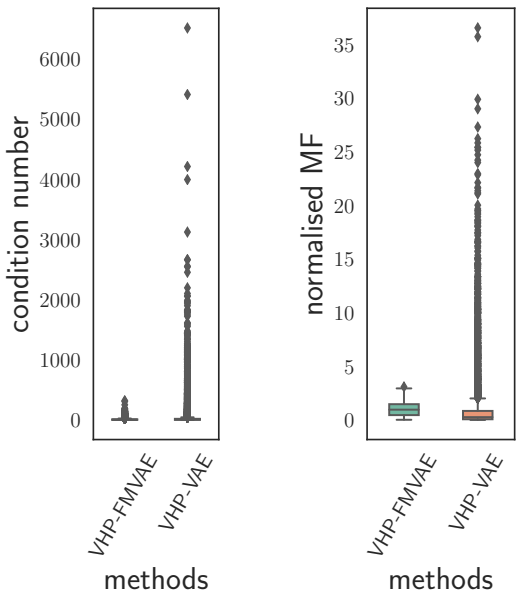


Figure 5.18: MNIST: if both the condition number and the normalised *MF* correspond approximately to 1, this indicates that $\mathbf{G} \propto \mathbb{1}$. The box plots are based on 10,000 samples.

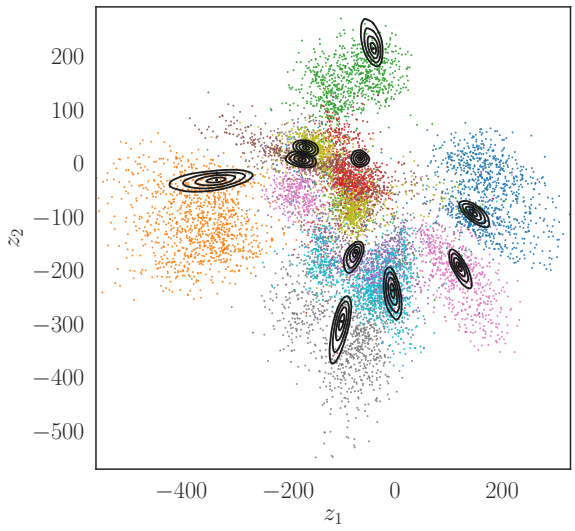
The condition number and normalised *MF* depicted in Figure 5.18 allow analysing the topology of the latent manifolds learned by the VHP-FMVAE and the VHP-VAE. Both metrics are based on 10,000 samples, and they indicate that the VHP-FMVAE, in contrast to the VHP-VAE, has learned a latent space that is close to a flat manifold.

Figure 5.19 shows the corresponding latent representations of the MNIST digits. The equidistance lines refer to the Riemannian distance to the respective encoded data point. The homogeneous contour plots in Figure 5.19a indicate that $\mathbf{G} \propto \mathbb{1}$.

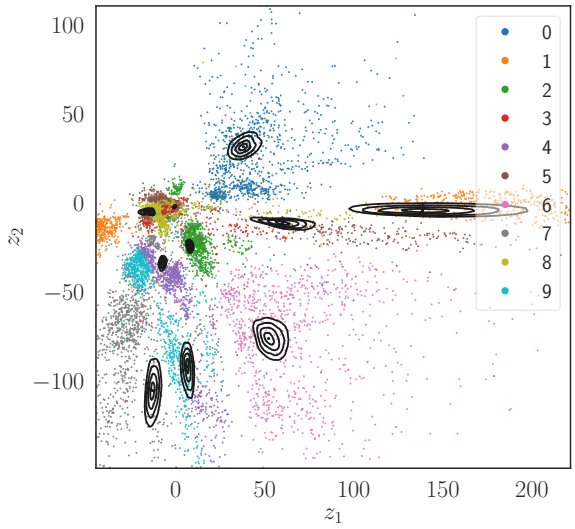
In Table 5.2, we verify that the VHP-FMVAE has learned a flat latent manifold by comparing 100 geodesics to their associated Euclidean interpolations. The procedure is identical to Section 5.2.3.1. The table shows the ratio between Euclidean distances in the latent space and

Table 5.2: MNIST: verifying that the latent space can act as Euclidean similarity metric. The table shows the length ratio of Euclidean interpolations to their corresponding geodesics (latent space). Additionally, we list the ratio between Riemannian lengths of Euclidean interpolations and Riemannian distances (observable space).

method	length ratio (latent space)	length ratio (observable space)
VHP-FMVAE	**0.92 ± 0.05**	**1.01 ± 0.08**
VHP-VAE	0.70 ± 0.31	1.13 ±0.22



(a) VHP-FMVAE



(b) VHP-VAE

Figure 5.19: Latent representation of MNIST data: the equidistance lines refer to the Riemannian distance to the respective encoded data point. Round, homogeneous contour plots indicate that $\mathbf{G} \propto \mathbb{1}$.

the corresponding lengths of the geodesics. In addition, we list the ratio between Riemannian lengths of Euclidean interpolations and Riemannian distances.

5.2.3.3 *MOT16 Object Tracking Database*

In this section, we evaluate our approach on the MOT16 object tracking database (Milan et al., 2016), which is a large-scale person re-

identification dataset containing both static and dynamic scenes from diverse cameras. We compare our method with two baselines: SORT (Bewley et al., 2016) and DeepSORT (Wojke et al., 2017).

SORT is a simple online and real-time tracking method, which uses bounding box intersection over union (IoU) for associating detections between frames and Kalman filters for the track predictions. It relies on good two-dimensional bounding box detections from a separate detector and suffers from ID switching when tracks overlap in the image. The two-dimensional detections are obtained from a neural network, and SORT associates measurements of each frame to tracks that are initiated, kept, or removed over time. The IoU overlap is used as a distance/similarity function between a given track box and a measurement box, and all boxes are optimally associated using the Hungarian algorithm.

DeepSORT extends the original SORT algorithm to integrate appearance information based on a deep appearance descriptor, which helps with re-identification in the case of such overlaps or missed detections. The deep appearance descriptor is trained using a *supervised* cosine metric learning approach (Wojke and Bewley, 2018) and outputs a fixed vector output per object, which contains appearance information. During online application, the vector is used with nearest-neighbour queries to establish measurement-to-track associations, instead of just the IoU overlap used by the vanilla SORT.

In our experiments, we use the latent manifolds learned by the VHP-FMVAE and the VHP-VAE as a drop-in replacement to the fixed vector outputted by the appearance descriptor of DeepSORT, effectively only running the encoder during evaluation. Consequently, the dimension of the latent space is identical to the dimension of the fixed vector. The candidate object locations of the pre-generated detections for SORT, DeepSORT, and our methods are taken from (Yu et al., 2016).

Table 5.3 shows that the performance of VHP-FMVAE-SORT is better than that of VHP-VAE-SORT and even close to the performance of supervised learning (DeepSORT). All methods rely on the same underlying detector for object candidates and identical Kalman filter parameters. Compared to baseline SORT, which does not utilise any appearance information, DeepSORT has 2.54 times, VHP-VAE-SORT has 2.14 times, VHP-FMVAE-SORT ($\eta = 300$) has 2.41 times, and VHP-FMVAE-SORT ($\eta = 3000$) has 2.48 times fewer ID switches. Whilst the supervised DeepSORT descriptor has the least, using unsupervised VAEs with flat latent manifolds has only 2.2% more switches, without the need for labels.

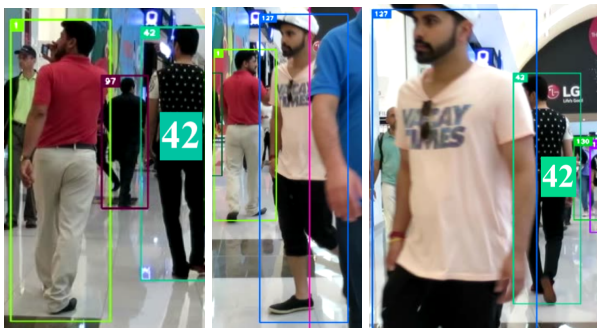
Furthermore, by ensuring a quasi-Euclidean latent space, one can query nearest-neighbours efficiently via data-structures such as k-dimensional trees. Figure 5.20 shows an example of the results. In other examples, VHP-FMVAE-SORT works similar as DeepSORT.

Table 5.3: Comparisons between different descriptors for the purposes of object tracking and re-identification (Ristani et al., 2016). The bold and the red numbers denote the best results among all methods and among unsupervised methods, respectively.

Method	Type	$IDF_1\uparrow$	IDP↑	IDR↑	Recall↑	Precision↑	FAR↓	MT↑	PT↓	ML↓	FP↓	FN↓	IDs↓	FM↓	MOTA ↑	MOTP ↑	MOTAL↑
VHP-FMVAE-SORT $\eta = 300$	unsupervised	63.7	77.0	54.3	65.0	92.3	**1.12**	158	269	90	**5950**	38592	616	**1143**	59.1	81.8	59.7
VHP-FMVAE-SORT $\eta = 3000$	unsupervised	64.2	**77.6**	54.8	65.1	**92.3**	1.13	162	265	90	6026	38515	598	1163	59.1	81.8	59.7
VHP-VAE-SORT	unsupervised	60.5	72.3	52.1	65.8	91.4	1.28	170	266	**81**	6820	37739	693	1264	59.0	81.6	59.6
SORT	n.a.	57.0	67.4	49.4	66.4	90.6	1.44	158	275	84	7643	37071	1486	1515	58.2	**81.9**	59.5
DeepSORT	supervised	**64.7**	76.9	**55.8**	**66.7**	91.9	1.22	**180**	**250**	87	6506	**36747**	**585**	1165	**60.3**	81.6	**60.8**

↑ indicates that the higher the score, the better the performance
↓ indicates that the lower the score, the better the performance

$IDF_1(\uparrow)$: ID F_1 score
IDP(↑): ID precision
IDR(↑): ID recall
FAR(↓): false alarm ratio
MT(↑): mostly tracked trajectory
PT(↓): partially tracked trajectory
ML(↓): mostly lost trajectory
FP(↓): false positives
FN(↓): false negatives
IDs(↓): number of times an ID switches to a different previously tracked object
FM(↓): fragmentations
MOTA(↑): multi-object tracking accuracy
MOTP(↑): multi-object tracking precision
MOTAL(↑): log tracking accuracy



(a) VHP-FMVAE-SORT with $\eta = 3000$

(b) VHP-VAE-SORT

(c) SORT

(d) DeepSORT

Figure 5.20: MOT16: example identity switches between overlapping tracks. For VHP-FMVAE-SORT, the track 42 gets occluded, but is re-identified correctly when again visible. For VHP-VAE-SORT and DeepSORT, the occluding track gets assigned the same ID as the track it occludes (42/61), and subsequently keeps this (erroneous) track. For vanilla SORT, track 3260 gets occluded and when subsequently visible, it gets assigned a new ID 3421.

5.2.4 *Summarising Discussion*

In this section, we have introduced flat manifold variational autoencoders. We have shown that this class of VAEs learns latent representations where the Euclidean metric reflects the similarity of data, making measures computationally very efficient. This is realised by defining the latent space as a Riemannian manifold and by combining a powerful empirical Bayes prior, introduced in Chapter 3, with a regularisation method that constrains the Riemannian metric tensor to be a scaled identity matrix.

We have conducted experiments on human motion data, the binarised MNIST digit dataset, and the MOT16 object tracking database, demonstrating the effectiveness of our proposed method for measuring the similarity of data. In Sections 5.2.3.1 and 5.2.3.2, we have analysed the topology of the learned latent manifolds by means of the condition number and the magnification factor. The results have shown that our method learns flat latent manifolds, where Euclidean interpolations correspond to geodesics. In Section 5.2.3.3, we have used our similarity metric for object tracking and re-identification, where the performance of our unsupervised learning approach nears that of state-of-the-art supervised learning methods.

However, a limitation of our method is the low effectiveness of the *mixup*-based Jacobian regularisation in high-dimensional latent spaces. In order to learn a flat latent manifold, the Riemannian metric tensor has to be regularised towards a scaled identity matrix for the *entire* latent space, i.e. also in regions where no data is available. Data-augmentation methods such as *mixup* can provide this data, but the efficiency of the regularisation suffers from the curse of dimensionality. This is because the amount of data required for regularising the Riemannian metric tensor increases exponentially with the dimension of the latent space. This issue can be addressed by future research.

6 CONCLUSION AND OUTLOOK

In this thesis, we have addressed the question of how to learn latent representations—in the context of probabilistic generative modelling—that reflect the factors of variation and topology of the observed data. To this end, we have proposed a constrained optimisation-based formulation of amortised variational inference and complemented it with a powerful empirical Bayes method.

In Chapter 3, we have introduced this approach within the framework of VAEs and thus laid the foundation for the further methods presented in this thesis. For this purpose, we have reformulated the ELBO as the Lagrangian of a constrained optimisation problem and mathematically proven that this is a valid alternative for learning generative latent-variable models. Building upon this, we have proposed the VHP, an empirical Bayes method for learning the prior distribution by means of a lower bound on the optimal prior, which is the aggregated posterior distribution. Concurrently, we have introduced the associated constrained optimisation algorithm REWO. In contrast to the original VAE, our method provides informative latent representations reflecting the topology of observed data, with the VHP preventing an over-regularisation of the approximate posterior distribution. Moreover, the learned prior distribution is non-trivial and well-adapted to the latent representation.

In Chapter 4, we have applied our approach to sequential data of dynamic systems by introducing a constrained optimisation framework for learning DSSMs. To this end, we have derived a general Lagrangian formulation of the sequential ELBO on the basis of distortion and rate, as well as extended the VHP and REWO to DSSMs. Building on the constrained optimisation framework, we have introduced the EKVAE, which combines extended Kalman filtering/smoothing with amortised variational inference and a neural linearisation approach to model dynamics more accurately than RNN-based DSSMs. Our experimental evaluations have demonstrated that applying the constrained optimisation framework to established DSSMs facilitates system identification, with the consequence that the learned latent variables represent the true underlying state of the observed dynamic system. The result is a substantial increase in prediction accuracy. In this context, we have shown that the EKVAE achieves a significantly higher prediction accuracy than state-of-the-art (RNN-based) models. Furthermore, we have verified that the EKVAE can learn disentangled position–velocity representations and demonstrated how these can be used for model-

based reinforcement learning to define/encode reward functions and learn policies.

In Chapter 5, we have focused on another area where our approach provides substantial benefits: unsupervised metric learning. In order to find a metric that measures the similarity of i.i.d. data, we have analysed the topology of latent spaces learned by VAEs. For this purpose, we have defined the latent space as a Riemannian manifold and shown that the Riemannian distance—i.e. the Riemannian length of geodesics—is an accurate similarity metric. To enable computationally efficient similarity measures, we have introduced the VHP-FMVAE. This method allows us to learn latent spaces where Euclidean distances are proportional to Riemannian distances and therefore reflect the similarity of data. This is realised by combining our method introduced in Chapter 3 with a regularisation approach that constrains the Riemannian metric tensor to be a scaled identity matrix. We have used the VHP-FMVAE to learn similarity metrics for object tracking and re-identification, where our unsupervised learning approach can compete with state-of-the-art supervised learning methods.

In conclusion, we have shown in this work that amortised variational inference in latent-variable models and sequential latent-variable models can be significantly improved by combining it with constrained optimisation and empirical Bayes priors. Our experimental results have emphasised the importance of the latent representation for the quality of generative models and the prediction accuracy of DSSMs. Moreover, we have demonstrated how latent representations can be used for various applications, such as model-based reinforcement learning or unsupervised metric learning.

For future research, another class of latent-variable models is of particular interest: conditional latent-variable models. They are applied for modelling one-to-many mappings, where a single condition has several equivalent solutions—and the parameters are typically learned using the framework of conditional VAEs. In (Klushyn et al., 2019a), we have shown that already a simple empirical Bayes approach can increase the generalisation capacity of conditional VAEs, leading to a larger variety of realistic generations. This is because classical conditional VAEs suffer from a difficulty to represent multimodal distributions, which we have verified is a consequence of the optimisation approach and the predefined, typically unimodal prior distribution. Our constrained optimisation-based formulation, as well as the VHP and REWO, can be straightforwardly adapted to conditional latent-variable models—and we expect, based on our previous results, that this will considerably improve the performance of conditional VAEs.

A

APPENDIX

A.1 MODEL ARCHITECTURES

A.1.1 *Model Architectures Chapter 3*

Table A.1: Model architectures. FC refers to fully-connected layers. GatedFC/-GatedConv denote pairs of fully-connected/convolutional layers multiplied element-wise, where one of the layers (gate) always uses sigmoid activations.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Input	256 (flattened 16×16)
	$1e$-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation. Gaussian.
		$q_\Phi(\boldsymbol{\zeta}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\Theta(\mathbf{z}\|\boldsymbol{\zeta})$	FC 256, 256, 256, 256. ReLU activation.
		Others	κ = 0.02, ν = 5, K = 16
		Graph	1,000 nodes, 18 neighbours.
CMU Human	Adam	Input	50
	$1e$-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation. Gaussian.
		$q_\Phi(\boldsymbol{\zeta}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\Theta(\mathbf{z}\|\boldsymbol{\zeta})$	FC 256, 256, 256, 256. ReLU activation.
		Others	κ = 0.02, ν = 1, K = 32
		Graph	2,530 nodes, 15 neighbours.
3D Faces,	Adam	Input	64×64×1
3D Chairs	$5e$-4	Latents	32
		$q_\phi(\mathbf{z}\|\mathbf{x})$	Conv 32×5×5 (stride 2), 32×3×3 (stride 1), 48×5×5 (stride 2), 48×3×3 (stride 1), 64×5×5 (stride 2), 64×3×3 (stride 1), 96×5×5 (stride 2), 96×3×3 (stride 1). FC 256. ReLU activation
		$p_\theta(\mathbf{x}\|\mathbf{z})$	Deconv reverse of encoder. ReLU activation. Bernoulli.
		$q_\Phi(\boldsymbol{\zeta}\|\mathbf{z})$	FC 256, 256. ReLU activation.
		$p_\Theta(\mathbf{z}\|\boldsymbol{\zeta})$	FC 256, 256. ReLU activation.
		Others	κ = 0.2, ν = 1, K = 16
		Graph	10,000 nodes (faces), 8,637 nodes (chairs), 18 neighbours.
dynamic MNIST,	Adam	Input	28×28×1
static MNIST,	$5e$-4	Latents	32
Fashion-MNIST,		$q_\phi(\mathbf{z}\|\mathbf{x})$	GatedConv 32×7×7 (stride 1) , 32×3×3 (stride 2), 64×5×5 (stride 1), 64×3×3 (stride 2), 64×3×3 (stride 1)
OMNIGLOT		$p_\theta(\mathbf{x}\|\mathbf{z})$	GatedFC 784. GatedConv 64×3×3 (stride 1), 64×3×3 (stride 1), 64×3×3 (stride 1), 64×3×3 (stride 1). linear activation. Bernoulli.
		$q_\Phi(\boldsymbol{\zeta}\|\mathbf{z})$	FC 256, 256. ReLU activation.
		$p_\Theta(\mathbf{z}\|\boldsymbol{\zeta})$	FC 256, 256. ReLU activation.
		Others	κ = 0.18 (dynamic MNIST, static MNIST, OMNIGLOT), κ = 0.31 (Fashion-MNIST), ν = 1, K = 16

A.1.2 *Model Architectures Chapter 4*

Table A.2: Model architectures of EKVAE. FC refers to fully-connected layers.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Observations	256 (flattened 16×16)
	$1e$-3	Time Steps	15
		Actions	1
		Latents	3
		$q_\phi(\mathbf{a}_t\|\mathbf{x}_t)$	FC 128, 128, 128. ReLU activation.
		$p_\theta(\mathbf{x}_t\|\mathbf{a}_t)$	FC 128, 128, 128. ReLU activation. Gaussian.
		Number of Base Matrices M	16
		α-Network	FC 64. ReLU activation.
		$q_{\phi_0}(\boldsymbol{\zeta}\|\mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1\|\boldsymbol{\zeta})$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.03$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 300$
Reacher (angle data)	Adam	Observations	2
	$1e$-3	Time Steps	30
		Actions	2
		Latents	4
		$q_\phi(\mathbf{a}_t\|\mathbf{x}_t)$	FC 128. ReLU activation.
		$p_\theta(\mathbf{x}_t\|\mathbf{a}_t)$	FC 128. ReLU activation. Gaussian.
		Number of Base Matrices M	8
		α-Network	FC 64, 64. ReLU activation.
		$q_{\phi_0}(\boldsymbol{\zeta}\|\mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1\|\boldsymbol{\zeta})$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.3$, $\tau_1 = 1$, $\tau_2 = 0.001$, $\nu = 10$
Reacher (image data)	Adam	Observations	64×64×3
	$5e$-3	Time Steps	30
		Actions	2
		Latents	5
		$q_\phi(\mathbf{a}_t\|\mathbf{x}_t)$	Conv 32×5×5 (stride 2), 64×5×5 (stride 2), 128×5×5 (stride 2). FC 256. ReLU activation.
		$p_\theta(\mathbf{x}_t\|\mathbf{a}_t)$	Deconv reverse of encoder. ReLU activation. Gaussian.
		Number of Base Matrices M	8
		α-Network	FC 64, 64. ReLU activation.
		$q_{\phi_0}(\boldsymbol{\zeta}\|\mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1\|\boldsymbol{\zeta})$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.2$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 30$

Table A.3: Model architectures of DKS. FC refers to fully-connected layers.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Observations	256 (flattened 16×16)
	1e-3	Time Steps	15
		Actions	1
		Latents	3
		$q_\phi(\mathbf{z}_t \mid \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$	BiLSTM 128. sigmoid activation. FC 64. ReLU activation.
		$p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$	FC 128, 128, 128. ReLU activation. Gaussian.
		$p_\theta(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$	FC 128, 128, 128. ReLU activation.
		$q_{\phi_0}(\zeta \mid \mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1 \mid \zeta)$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.03$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 300$
Reacher (angle data)	Adam	Observations	2
	1e-3	Time Steps	30
		Actions	2
		Latents	4
		$q_\phi(\mathbf{z}_t \mid \mathbf{x}_{1:T}, \mathbf{u}_{1:T})$	BiLSTM 128. sigmoid activation. FC 64. ReLU activation.
		$p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$	FC 128. ReLU activation. Gaussian.
		$p_\theta(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$	FC 128, 128, 128. ReLU activation.
		$q_{\phi_0}(\zeta \mid \mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1 \mid \zeta)$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.2$, $\tau_1 = 1$, $\tau_2 = 0.001$, $\nu = 10$

Table A.4: Model architectures of DVBS. FC refers to fully-connected layers.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Observations	256 (flattened 16×16)
	1e-3	Time Steps	15
		Actions	1
		Latents	3
		$q_\phi(\mathbf{z}_t \mid \mathbf{x}_{t:T}, \mathbf{u}_{t:T})$	LSTM 128. sigmoid activation. FC 64. ReLU activation.
		$p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$	FC 128, 128, 128. ReLU activation. Gaussian.
		Number of Base Matrices M	16
		α-Network	FC 64. ReLU activation.
		$q_{\phi_0}(\zeta \mid \mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1 \mid \zeta)$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.03$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 300$
Reacher (angle data)	Adam	Observations	2
	1e-3	Time Steps	30
		Actions	2
		Latents	4
		$q_\phi(\mathbf{z}_t \mid \mathbf{x}_{t:T}, \mathbf{u}_{t:T})$	LSTM 128. sigmoid activation. FC 64. ReLU activation.
		$p_\theta(\mathbf{x}_t \mid \mathbf{z}_t)$	FC 128. ReLU activation. Gaussian.
		Number of Base Matrices M	8
		α-Network	FC 64, 64. ReLU activation.
		$q_{\phi_0}(\zeta \mid \mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1 \mid \zeta)$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.3$, $\tau_1 = 1$, $\tau_2 = 0.001$, $\nu = 10$

Table A.5: Model architectures of KVAE. FC refers to fully-connected layers.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Observations	256 (flattened 16×16)
	1e-3	Time Steps	15
		Actions	1
		Latents	3
		$q_\phi(\mathbf{a}_t\|\mathbf{x}_t)$	FC 128, 128, 128. ReLU activation.
		$p_\theta(\mathbf{x}_t\|\mathbf{a}_t)$	FC 128, 128, 128. ReLU activation. Gaussian.
		Number of Base Matrices M	16
		α-Network	FC 64. ReLU activation.
		Dynamics Parameter Network	LSTM 64. sigmoid activation.
		$q_{\phi_0}(\boldsymbol{\zeta}\|\mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1\|\boldsymbol{\zeta})$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.03$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 300$

Table A.6: Model architectures of RSSM. FC refers to fully-connected layers.

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Observations	256 (flattened 16×16)
	1e-3	Time Steps	15
		Actions	1
		Latents	3
		$q_\phi(\mathbf{z}_t\|h_{t-1}, \mathbf{x}_{t:T}, \mathbf{u}_{t:T})$	LSTM 128. sigmoid activation. FC 64. ReLU activation.
		$p_\theta(\mathbf{x}_t\|\mathbf{z}_t)$	FC 128, 128, 128. ReLU activation. Gaussian.
		$p_\theta(\mathbf{z}_t\|h_{t-1})$	FC 128, 128, 128. ReLU activation.
		Deterministic State Model	LSTM 64. sigmoid activation.
		$q_{\phi_0}(\boldsymbol{\zeta}\|\mathbf{z}_1)$	FC 64, 64. ReLU activation.
		$p_{\psi_0}(\mathbf{z}_1\|\boldsymbol{\zeta})$	FC 64, 64. ReLU activation.
		Others	$\kappa = 0.03$, $\tau_1 = 10$, $\tau_2 = 0.01$, $\nu = 300$

A.1.3 *Model Architectures Chapter 5.1*

Table A.7: Model architectures. FC refers to fully-connected layers.

	Optimiser	Implementation Details	
Geodesics	Adam	Input	1
	$1e$-2	Output	2
		$g_\omega(t)$	FC 150, 150. tanh activation.
		Others	$T = 500$

Dataset	Optimiser	Implementation Details	
Pendulum	Adam	Input	256 (flattened 16×16)
	$1e$-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 512, 512. tanh activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 512, 512. tanh activation. Gaussian.
		Others	$K = 50$
MNIST	Adam	Input	784 (flattened 28×28)
	$1e$-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 512, 512. tanh activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	residual 128, 128, 128, 128, 128, 128, 128. tanh activation. Gaussian.
		Others	$K = 50$
Robot Arm	Adam	Input	6
	$1e$-3	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 512, 512. tanh activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 512, 512. tanh activation. Gaussian.
		Others	$K = 15$
CMU Human	Adam	Input	50
	$1e$-3	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 512, 512, 512. tanh activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 512, 512, 512. tanh activation. Gaussian.
		Others	$K = 15$

A.1.4 *Model Architectures Chapter 5.2*

Table A.8: Model architectures. FC refers to fully-connected layers. Conv2D and Conv2DT denote two-dimensional convolution layer and transposed two-dimensional convolution layer, respectively.

Dataset	Optimiser	Implementation Details	
CMU Human	Adam	Input	50
	1*e*-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation. Gaussian.
		$q_\Phi(\mathbf{1}\|\mathbf{z})$	FC 256, 256, 256, 256, ReLU activation.
		$p_\Theta(\mathbf{z}\|\mathbf{1})$	FC 256, 256, 256, 256, ReLU activation.
		Others	$\kappa = 0.03$, $\nu = 1$, $K = 32$, $\eta = 8000$
MNIST	Adam	Input	784 (flattened 28×28)
	1*e*-4	Latents	2
		$q_\phi(\mathbf{z}\|\mathbf{x})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\theta(\mathbf{x}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation. Bernoulli.
		$q_\Phi(\mathbf{1}\|\mathbf{z})$	FC 256, 256, 256, 256. ReLU activation.
		$p_\Theta(\mathbf{z}\|\mathbf{1})$	FC 256, 256, 256, 256. ReLU activation.
		Others	$\kappa = 0.245$, $\nu = 1$, $K = 16$, $\eta = 8000$
MOT16	Adam	Input	64×64×3
	3*e*-5	Latents	128
		$q_\phi(\mathbf{z}\|\mathbf{x})$	VGG16 (Simonyan and Zisserman, 2015)
		$p_\theta(\mathbf{x}\|\mathbf{z})$	Conv2DT+Conv2D 256, 128, 64, 32, 16. ReLU activation. Gaussian.
		$q_\Phi(\mathbf{1}\|\mathbf{z})$	FC 512, 512. ReLU activation.
		$p_\Theta(\mathbf{z}\|\mathbf{1})$	FC 512, 512. ReLU activation.
		Others	$\kappa = 0.8$, $\nu = 1$, $K = 8$, $\eta = 300$ or 3000

BIBLIOGRAPHY

Abadi, Martin et al. (2016). 'TensorFlow: A System for Large-Scale Machine Learning'. In: *Symposium on Operating Systems Design and Implementation (OSDI)*. USENIX Association, pp. 265–283 (cit. on p. 12).

Alemi, Alexander A., Ben Poole, Ian Fischer, Joshua V. Dillon, Rif A. Saurous and Kevin Murphy (2018). 'Fixing a Broken ELBO'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 80. PMLR, pp. 159–168 (cit. on pp. 1, 27, 53).

Altman, Naomi S. (1992). 'An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression'. In: *The American Statistician* 46.3, pp. 175–185 (cit. on p. 87).

Aubry, Mathieu, Daniel Maturana, Alexei A. Efros, Bryan C. Russell and Josef Sivic (2014). 'Seeing 3D Chairs: Exemplar Part-Based 2D-3D Alignment Using a Large Dataset of CAD Models'. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, pp. 3762–3769 (cit. on p. 47).

Beal, Matthew J. (2003). 'Variational algorithms for approximate Bayesian inference'. PhD thesis. University College London (cit. on p. 5).

Becker-Ehmck, Philip, Maximilian Karl, Jan Peters and Patrick van der Smagt (2020). 'Learning to Fly via Deep Model-Based Reinforcement Learning'. In: *CoRR* abs/2003.08876 (cit. on p. 51).

Bengio, Yoshua, Aaron Courville and Pascal Vincent (2013). 'Representation Learning: A Review and New Perspectives'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1798–1828 (cit. on p. 67).

Bengio, Yoshua, Pascal Lamblin, Dan Popovici and Hugo Larochelle (2007). 'Greedy Layer-Wise Training of Deep Networks'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 19. MIT Press, pp. 153–160 (cit. on p. 34).

Berenson, Dmitry, Siddhartha S. Srinivasa, Dave Ferguson and James J. Kuffner (2009). 'Manipulation Planning on Constraint Manifolds'. In: *International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 625–632 (cit. on p. 100).

Berger, James et al. (2006). 'The Case for Objective Bayesian Analysis'. In: *Bayesian Analysis* 1.3, pp. 385–402 (cit. on p. 15).

Bewley, Alex, ZongYuan Ge, Lionel Ott, Fabio Tozeto Ramos and Ben Upcroft (2016). 'Simple Online and Realtime Tracking'. In: *International Conference on Image Processing (ICIP)*. IEEE, pp. 3464–3468 (cit. on pp. 87, 117).

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning*. Springer (cit. on p. 3).

Bishop, Christopher M., Markus Svensén and Christopher KI. Williams (1997). 'Magnification Factors for the SOM and GTM Algorithms'. In: *Proceedings Workshop on Self-Organizing Maps* (cit. on p. 94).

Bitzer, Sebastian, Ioannis Havoutis and Sethu Vijayakumar (2008). 'Synthesising Novel Movements Through Latent Space Modulation of Scalable Control Policies'. In: *International Conference on Simulation of Adaptive Behavior (SAB)*. Springer, pp. 199–209 (cit. on p. 101).

Blei, David M., Alp Kucukelbir and Jon D. McAuliffe (2017). 'Variational inference: A review for statisticians'. In: *Journal of the American Statistical Association* 112.518, pp. 859–877 (cit. on p. 5).

Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz and Samy Bengio (2016). 'Generating Sentences from a Continuous Space'. In: *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*. ACL, pp. 10–21 (cit. on pp. 1, 13, 25, 27, 53).

Boyd, Stephen, Stephen P. Boyd and Lieven Vandenberghe (2004). *Convex oOptimization*. Cambridge University Press (cit. on p. 27).

Buja, Andreas, Trevor Hastie and Robert Tibshirani (1989). 'Linear Smoothers and Additive Models'. In: *The Annals of Statistics* 17.2, pp. 453–510 (cit. on p. 93).

Burda, Yuri, Roger B. Grosse and Ruslan Salakhutdinov (2016). 'Importance Weighted Autoencoders'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on pp. 14, 32, 42, 45, 91).

Caruso, C. and F. Quarta (1998). 'Interpolation Methods Comparison'. In: *Computers & Mathematics with Applications* 35.12, pp. 109–126 (cit. on p. 87).

Cayton, Lawrence (2005). 'Algorithms for Manifold Learning'. In: *Univ. of California at San Diego Tech. Rep* 12.1, pp. 1–17 (cit. on p. 36).

Chen, Nutan, Justin Bayer, Sebastian Urban and Patrick van der Smagt (2015). 'Efficient Movement Representation by Embedding Dynamic Movement Primitives in Deep Autoencoders'. In: *International conference on humanoid robots (Humanoids)*. IEEE, pp. 434–440 (cit. on pp. 43, 101).

Chen, Nutan, Francesco Ferroni, Alexej Klushyn, Alexandros Paraschos, Justin Bayer and Patrick van der Smagt (2019). 'Fast Approximate Geodesics for Deep Generative Models'. In: *International Conference on Artificial Neural Networks (ICANN)*. Vol. 11728. Springer, pp. 554–566 (cit. on pp. vi, 36, 103, 114).

Chen, Nutan, Alexej Klushyn, Francesco Ferroni, Justin Bayer and Patrick van der Smagt (2020). 'Learning Flat Latent Manifolds With VAEs'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 119. PMLR, pp. 1587–1596 (cit. on pp. vi, 103).

Chen, Nutan, Alexej Klushyn, Alexandros Paraschos, Djalel Benbouzid and Patrick van der Smagt (2018). 'Active Learning Based on Data Uncertainty and Model Sensitivity'. In: *International Conference on*

Intelligent Robots and Systems (IROS). IEEE, pp. 1547–1554 (cit. on pp. vi, 103).

Chen, Xi et al. (2017). 'Variational Lossy Autoencoder'. In: *International Conference on Learning Representations, (ICLR)*. OpenReview.net (cit. on pp. 1, 15, 25).

Chen*, Nutan, Alexej Klushyn*, Richard Kurle*, Xueyan Jiang, Justin Bayer and Patrick van der Smagt (2017). 'Metrics for Deep Generative Models Based on Learned Skills'. In: *Advances in Neural Information Processing Systems (NeurIPS), Workshop on Acting and Interacting in the Real World: Challenges in Robot Learning* (cit. on p. vi).

– (2018). 'Metrics for Deep Generative Models'. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 84. PMLR, pp. 1540–1550 (cit. on pp. v, 88).

Child, Rewon (2020). 'Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images'. In: *CoRR* abs/2011.10650 (cit. on p. 25).

Cho, Kyunghyun et al. (2014). 'Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation'. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 1724–1734 (cit. on p. 24).

Cremer, Chris, Quaid Morris and David Duvenaud (2017). 'Reinterpreting Importance-Weighted Autoencoders'. In: *International Conference on Learning Representations (ICLR), Workshop Track Proceedings*. OpenReview.net (cit. on pp. 14, 40, 91).

De Casteljau, Paul de Faget (1986). *Shape Mathematics and CAD*. Kogan Page (cit. on p. 94).

Fraccaro, Marco, Simon Kamronn, Ulrich Paquet and Ole Winther (2017). 'A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. Curran Associates, Inc., pp. 3601–3610 (cit. on pp. 54, 57, 60, 62, 69).

Gómez-Bombarelli, Rafael et al. (2018). 'Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules'. In: *ACS central science* 4.2, pp. 268–276 (cit. on p. 25).

Goodfellow, Ian, Yoshua Bengio, Aaron Courville and Yoshua Bengio (2016). *Deep Learning*. MIT Press (cit. on pp. 9, 24).

Hafner, Danijar et al. (2019). 'Learning Latent Dynamics for Planning from Pixels'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 97. PMLR, pp. 2555–2565 (cit. on pp. 51, 57, 67, 69).

Higgins, Irina et al. (2017). 'beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on pp. 1, 25, 27, 28, 33, 56, 67, 107).

Hochreiter, Sepp and Jürgen Schmidhuber (1997). 'Long Short-Term Memory'. In: *Neural Computation* 9.8, pp. 1735–1780 (cit. on p. 24).

Hu, Zhiting, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov and Eric P. Xing (2017). 'Toward Controlled Generation of Text'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 70. PMLR, pp. 1587–1596 (cit. on p. 25).

Jiang, Xueyan (2014). 'Integrating Prior Knowledge Into Factorization Approaches for Relational Learning'. PhD thesis. Ludwig Maximilian University of Munich (cit. on p. 93).

Kalman, Rudolph E. et al. (1960). 'A New Approach to Linear Filtering and Prediction Problems'. In: *Journal of Basic Engineering* 82.1, pp. 35–45 (cit. on p. 20).

Karl, Maximilian, Maximilian Soelch, Justin Bayer and Patrick van der Smagt (2017). 'Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on pp. 52, 57–59, 61, 69, 70, 72).

Karl, Maximilian, Maximilian Soelch, Philip Becker-Ehmck, Djalel Benbouzid, Patrick van der Smagt and Justin Bayer (2019). 'Unsupervised Real-Time Control Through Variational Empowerment'. In: *International Symposium on Robotics Research* (cit. on pp. 58, 59).

Kingma, Diederik P., Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever and Max Welling (2016). 'Improved Variational Inference with Inverse Autoregressive Flow'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. Curran Associates, Inc., pp. 4736–4744 (cit. on pp. 1, 15, 25).

Kingma, Diederik P. and Max Welling (2014). 'Auto-Encoding Variational Bayes'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on pp. 1, 8, 11, 12, 25, 31).

– (2019). 'An Introduction to Variational Autoencoders'. In: *Foundations and Trends in Machine Learning* 12.4, pp. 307–392 (cit. on pp. 8, 11, 13).

Kitagawa, Genshiro (1987). 'Non-Gaussian State-Space Modeling of Nonstationary Time Series'. In: *Journal of the American Statistical Association* 82.400, pp. 1032–1041 (cit. on p. 19).

Klushyn, Alexej (2019). *Learning Hierarchical Priors in VAEs*. URL: https://argmax.ai/blog/vhp-vae/ (visited on 26/05/2021) (cit. on p. 13).

Klushyn, Alexej, Nutan Chen, Botond Cseke, Justin Bayer and Patrick van der Smagt (2019a). 'Increasing the Generalisation Capacity of Conditional VAEs'. In: *International Conference on Artificial Neural Networks (ICANN)*. Vol. 11728. Springer, pp. 779–791 (cit. on pp. vi, 122).

Klushyn, Alexej, Nutan Chen, Richard Kurle, Botond Cseke and Patrick van der Smagt (2019b). 'Learning Hierarchical Priors in VAEs'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. Curran Associates, Inc., pp. 2870–2879 (cit. on pp. v, 25).

Klushyn, Alexej, Richard Kurle, Maximilian Soelch, Botond Cseke and Patrick van der Smagt (2021). 'Latent Matters: Learning Deep State-Space Models'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34. Curran Associates, Inc. (cit. on pp. v, 51).

Krishnan, Rahul G., Uri Shalit and David A. Sontag (2015). 'Deep Kalman Filters'. In: *CoRR* abs/1511.05121 (cit. on pp. 23, 52, 54, 57, 69).

Kulis, Brian et al. (2013). 'Metric Learning: A Survey'. In: *Foundations and Trends in Machine Learning* 5.4, pp. 287–364 (cit. on p. 87).

Kurle, Richard, Botond Cseke, Alexej Klushyn, Patrick van der Smagt and Stephan Günnemann (2020). 'Continual Learning With Bayesian Neural Networks for Non-Stationary Data'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on p. vi).

Lake, Brenden M., Ruslan Salakhutdinov and Joshua B. Tenenbaum (2015). 'Human-Level Concept Learning Through Probabilistic Program Induction'. In: *Science* 350.6266, pp. 1332–1338 (cit. on p. 45).

Larochelle, Hugo and Iain Murray (2011). 'The Neural Autoregressive Distribution Estimator'. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 15. JMLR.org, pp. 29–37 (cit. on pp. 12, 45, 97, 114).

Lecun, Y., L. Bottou, Y. Bengio and P. Haffner (1998). 'Gradient-Based Learning Applied to Document Recognition'. In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on p. 45).

Lee, John M. (2006). *Riemannian Manifolds: An Introduction to Curvature*. Springer Science & Business Media (cit. on p. 105).

Levine, Sergey and Vladlen Koltun (2013). 'Guided Policy Search'. In: *Proceedings of the International Conference on Machine Learning, (ICML)*. Vol. 28. JMLR.org, pp. 1–9 (cit. on p. 51).

Li, Tao and Chris H. Q. Ding (2006). 'The Relationships Among Various Nonnegative Matrix Factorization Methods for Clustering'. In: *Proceedings of the International Conference on Data Mining (ICDM)*. IEEE Computer Society, pp. 362–371 (cit. on p. 87).

Milan, Anton, Laura Leal-Taixé, Ian D. Reid, Stefan Roth and Konrad Schindler (2016). 'MOT16: A Benchmark for Multi-Object Tracking'. In: *CoRR* abs/1603.00831 (cit. on p. 116).

Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press (cit. on pp. 3, 4, 11, 15).

Neal, Radford M. and Geoffrey E. Hinton (1998). 'A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants'. In: *Learning in Graphical Models*. Springer, pp. 355–368 (cit. on pp. 7, 18).

Papamakarios, George, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed and Balaji Lakshminarayanan (2021). 'Normalizing Flows for Probabilistic Modeling and Inference'. In: *Journal of Machine Learning Research* 22.57, pp. 1–64 (cit. on p. 15).

Paszke, Adam et al. (2019). 'PyTorch: An Imperative Style, High-Performance Deep Learning Library'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. Curran Associates, Inc., pp. 8024–8035 (cit. on p. 12).

Paysan, Pascal, Reinhard Knothe, Brian Amberg, Sami Romdhani and Thomas Vetter (2009). 'A 3D Face Model for Pose and Illumination Invariant Face Recognition'. In: *International Conference on Advanced Video and Signal Based Surveillance, (AVSS)*. IEEE Computer Society, pp. 296–301 (cit. on p. 47).

Rangapuram, Syama Sundar, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang and Tim Januschowski (2018). 'Deep State Space Models for Time Series Forecasting'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 31. Curran Associates, Inc., pp. 7796–7805 (cit. on p. 51).

Rauch, Herbert E., F. Tung and Charlotte T. Striebel (1965). 'Maximum likelihood estimates of linear dynamic systems'. In: *AIAA journal* 3.8, pp. 1445–1450 (cit. on p. 20).

Razavi, Ali, Aäron van den Oord and Oriol Vinyals (2019). 'Generating Diverse High-Fidelity Images with VQ-VAE-2'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. Curran Associates, Inc., pp. 14837–14847 (cit. on p. 25).

Rezende, Danilo Jimenez and Shakir Mohamed (2015). 'Variational Inference With Normalizing Flows'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 37. PMLR, pp. 1530–1538 (cit. on pp. 1, 15, 25, 36).

Rezende, Danilo Jimenez, Shakir Mohamed and Daan Wierstra (2014). 'Stochastic Backpropagation and Approximate Inference in Deep Generative Models'. In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 32. PMLR, pp. 1278–1286 (cit. on pp. 1, 8, 11, 25).

Rezende, Danilo Jimenez and Fabio Viola (2018). 'Taming VAEs'. In: *CoRR* abs/1810.00597 (cit. on pp. 1, 25, 27, 28).

Riesselman, Adam J., John B. Ingraham and Debora S. Marks (2018). 'Deep Generative Models of Genetic Variation Capture the Effects of Mutations'. In: *Nature Methods* 15, pp. 816–822 (cit. on p. 25).

Rifai, Salah, Yann N. Dauphin, Pascal Vincent, Yoshua Bengio and Xavier Muller (2011a). 'The Manifold Tangent Classifier'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 24. Curran Associates, Inc., pp. 2294–2302 (cit. on p. 36).

Rifai, Salah et al. (2011b). 'Higher Order Contractive Auto-Encoder'. In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*. Vol. 6912. Springer, pp. 645–660 (cit. on p. 105).

Ristani, Ergys, Francesco Solera, Roger S. Zou, Rita Cucchiara and Carlo Tomasi (2016). 'Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking'. In: *European Conference on*

Computer Vision (ECCV). Vol. 9914. Springer, pp. 17–35 (cit. on p. 118).

Rumelhart, David E., Geoffrey E. Hinton and Ronald J. Williams (1986). 'Learning Representations by Back-Propagating Errors'. In: *Nature* 323.6088, pp. 533–536 (cit. on p. 10).

Rybkin, Oleh, Kostas Daniilidis and Sergey Levine (2020). 'Simple and Effective VAE Training with Calibrated Decoders'. In: *CoRR* abs/2006.13202 (cit. on p. 31).

Salinas, David, Valentin Flunkert, Jan Gasthaus and Tim Januschowski (2020). 'DeepAR: Probabilistic Forecasting With Autoregressive Recurrent Networks'. In: *International Journal of Forecasting* 36.3, pp. 1181–1191 (cit. on p. 51).

Särkkä, Simo (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press (cit. on pp. 3, 20, 23).

Schölkopf, Bernhard, Alexander J. Smola and Klaus-Robert Müller (1997). 'Kernel Principal Component Analysis'. In: *International Conference on Artificial Neural Networks (ICANN)*. Vol. 1327. Springer, pp. 583–588 (cit. on p. 87).

Schön, Thomas B, Adrian Wills and Brett Ninness (2011). 'System Identification of Nonlinear State-Space Models'. In: *Automatica* 47.1, pp. 39–49 (cit. on p. 19).

Shannon, Claude E. (1948). 'A Mathematical Theory of Communication'. In: *The Bell System Technical Journal* 27.3, pp. 379–423 (cit. on p. 47).

Simonyan, Karen and Andrew Zisserman (2015). 'Very Deep Convolutional Networks for Large-Scale Image Recognition'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on p. 128).

Sønderby, Casper Kaae, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby and Ole Winther (2016). 'Ladder Variational Autoencoders'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. Curran Associates, Inc., pp. 3738–3746 (cit. on pp. 14, 27, 33, 37, 47, 56, 107).

Tassa, Yuval et al. (2018). 'DeepMind Control Suite'. In: *CoRR* abs/1801.00690 (cit. on p. 69).

Taylor, Graham W., Geoffrey E. Hinton and Sam Roweis (2007). 'Modeling Human Motion Using Binary Latent Variables'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 19. MIT Press, pp. 1345–1352 (cit. on p. 101).

Tomczak, Jakub and Max Welling (2018). 'VAE With a VampPrior'. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 84. PMLR, pp. 1214–1223 (cit. on pp. 1, 15, 25, 32, 43, 45, 55).

Tosi, Alessandra, Søren Hauberg, Alfredo Vellido and Neil D. Lawrence (2014). 'Metrics for Probabilistic Geometries'. In: *Proceedings of the*

Conference on Uncertainty in Artificial Intelligence (UAI). AUAI Press, pp. 800–808 (cit. on pp. 87, 89).

Wang, Yixin, Andrew C. Miller and David M. Blei (2019). 'Comment: Variational Autoencoders as Empirical Bayes'. In: *Statistical Science* 34.2, pp. 229–233 (cit. on p. 15).

Watter, Manuel, Jost Tobias Springenberg, Joschka Boedecker and Martin A. Riedmiller (2015). 'Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28. Curran Associates, Inc., pp. 2746–2754 (cit. on pp. 52, 61, 70).

Weinberger, Kilian Q., John Blitzer and Lawrence K. Saul (2006). 'Distance Metric Learning for Large Margin Nearest Neighbor Classification'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 18. MIT Press, pp. 1473–1480 (cit. on p. 87).

Wojke, Nicolai and Alex Bewley (2018). 'Deep Cosine Metric Learning for Person Re-identification'. In: *Winter Conference on Applications of Computer Vision (WACV)*. IEEE Computer Society, pp. 748–756 (cit. on p. 117).

Wojke, Nicolai, Alex Bewley and Dietrich Paulus (2017). 'Simple Online and Realtime Tracking With a Deep Association Metric'. In: *International Conference on Image Processing (ICIP)*. IEEE, pp. 3645–3649 (cit. on pp. 87, 117).

Xiao, Han, Kashif Rasul and Roland Vollgraf (2017). 'Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms'. In: *CoRR* abs/1708.07747 (cit. on p. 45).

Xing, Eric P., Andrew Y. Ng, Michael I. Jordan and Stuart J. Russell (2002). 'Distance Metric Learning with Application to Clustering with Side-Information'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 15. MIT Press, pp. 505–512 (cit. on p. 87).

Yu, Fengwei, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi and Junjie Yan (2016). 'POI: Multiple Object Tracking with High Performance Detection and Appearance Feature'. In: *European Conference on Computer Vision (ECCV)*. Vol. 9914. Springer, pp. 36–42 (cit. on p. 117).

Zhang, Hongyi, Moustapha Cissé, Yann N. Dauphin and David Lopez-Paz (2018). 'Mixup: Beyond Empirical Risk Minimization'. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net (cit. on p. 106).