Distributed controller design for a class of sparse singular systems with privacy constraints

Frederik Deroo * **Michael Ulbrich** ** **Sandra Hirche** *
Brian D. O. Anderson ***

* *Institute for Information-Oriented Control, Technische Universität München, D-80290 München, Germany,* fred.deroo@tum.de, hirche@tum.de
** *Chair of Mathematical Optimization, Department of Mathematics, Technische Universität München, Boltzmannstr. 3, D-85747 Garching b. München, Germany,* mulbrich@ma.tum.de
*** *The Australian National University and National ICT Australia, Canberra ACT 2600 Australia.* brian.anderson@anu.edu.au

Abstract: In the current research on distributed control of interconnected large-scale dynamical systems an often neglected issue is the desire to ensure privacy of subsystems. This gives motivation for the presented distributed controller design method which requires communication and the exchange of model data only with direct neighbors. Thus, no global system knowledge is required. An important property of many large-scale systems is the presence of algebraic conservation constraints, for example in terms of energy or mass flow. Therefore, the presented controller design takes these constraints explicitly into account while preserving the sparsity structure of the distributed system necessary for a distributed design. The computation is based on the simulation of the system states and of adjoint states. The control objective is represented by the finite horizon linear quadratic cost functional.

Keywords: Distributed control, Singular systems, Large-scale systems, Descriptor systems

1. INTRODUCTION

Large-scale dynamical systems have moved into the focus of current research in the control theory community. Applications that are typically cited include power systems, water distribution or transportation systems. Centralized control mechanisms are not applicable to these systems for obvious reasons. The first ideas to deal with these systems involve decentralized control (see Siljak (1978)) where no communication between subsystems is allowed. Recent research tries to relax this restriction by allowing communication between some – but not all – subsystems which is then called distributed control. Distributed control attempts to achieve better performance and robustness than decentralized control while keeping the communication complexity low when compared to centralized control.

Most available results on distributed controller synthesis have in common that they require centralized global knowledge about the system model, see for example Langbort et al. (2004), Shah and Parrilo (2010) and Vamsi and Elia (2010). This assumption might not be valid in large-scale systems because of privacy issues, meaning that the subsystems are not willing to supply their model data to every other subsystem or to some centralized entity. In the power system, for example, the introduction of smart-appliances and small power generation units on the consumer level introduce more available model data into the overall system. This raises privacy issues over who has access to this additional model data, see e.g. Cavoukian et al. (2010), and a distributed controller design can help to ensure consumer privacy in this context. Other reasons for a distributed design are the large system size, or sometimes computational limitations. Therefore, it is desirable that the computation of a controller for a large-scale system is done distributedly without global system knowledge and without the need for every subsystem to globally share all of its information. Results in this direction can be found in Deroo et al. (2012) and Martensson and Rantzer (2012). Related research is also presented in Farokhi et al. (2013) where the authors investigate the difference in performance between a controller with limited model information and one with full information.

An often neglected point in the distributed control literature is that most application examples of large-scale interconnected dynamical systems are subject to some kind of conservation constraint. For example, the power system needs to satisfy the Kirchhoff laws. Water or other distribution systems also need to satisfy mass, energy or current balances. These conservation laws or other algebraic constraints lead to a differential-algebraic form for the system equation. The systems are then called singular or descriptor systems. For an introduction to singular systems, see Lewis (1986). One method to control this system class would be to substitute the solution of the algebraic constraints into the dynamic equations and then proceed with standard control design methods. However, this approach destroys the sparsity structure of the distributed system. This sparsity structure, however, is necessary when distributed design methods are required, e.g. in large-scale systems.

⋆ The work is supported by the German Research Foundation (DFG) within the Priority Program SPP 1305 "Control Theory of Digitally Networked Dynamical Systems". The work of B. D. O. Anderson is supported by NICTA, which is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program, and the Australian Research Councils Discovery Project DP- 110100538.

Important results regarding LQ-control of singular systems are given in Cobb (1983) and Bender and Laub (1987). Decentralized control of singular systems is treated in e.g. Chang and Davison (2001), but to the best of the authors' knowledge, there are no results on distributed control with a structured feedback matrix (the structure reflecting subsystem interaction), let alone controller design without centralized model knowledge.

The main contribution of this paper is a gradient descent method to iteratively and distributedly determine an optimal state feedback controller for singular systems. The algebraic part of the system equations are taken into account explicitly during the controller design and the algebraic variables are also used for feedback. The controller is computed using simulated state trajectories and trajectories of adjoint states and does not need any global system model knowledge but only communication with direct neighbors. This ensures that it does not have a complexity determined by the global system and ensures a degree of privacy of the subsystems. The gradient descent method uses a distributedly computed Barzilai-Borwein step size and checks the Armijo rule distributedly to guarantee convergence. The performance of the presented approach is evaluated using numerical simulations. The main difference to our earlier works, see Deroo et al. (2012), are the additional algebraic system constraint and a different derivation method to achieve the result. The addition of the algebraic constraint makes the presented distributed controller design applicable to a larger system class, and enables an application to the important practical example of the power system.

Research related to the presented paper is done in Zargham et al. (2012) where a distributed Armijo step size is computed based on N-hop neighborhood information and minimum consensus, while in this paper a Barzilai-Borwein step size is used and the Armijo rule is only verified distributedly. A control problem similar to the presented one is treated in distributed MPC such as in Giselsson and Rantzer (2010), but the presented approach here differs in that a time-invariant feedback matrix is determined before its application instead of a recomputation of an open-loop input trajectory. The work in Chen and Baras (2006) can also be seen as related to the presented work, because they present a scalable distributed control law for network optimization under constraints.

The remainder of the paper is organized as follows. In Section 2, the problem formulation is presented. Section 3 shows the algorithm to determine a distributed controller for singular systems. A numerical example is given in Section 4, and the paper concludes with a summary in Section 5.

Notation.

Given a matrix $A \in \mathbb{R}^{m\times n}$ with columns a_i, we can give a vectorized version of the matrix by associating the vector

$$\text{vec}(A) = \left[a_1^T, \cdots, a_n^T\right]^T \in \mathbb{R}^{nm\times 1}.$$

The scalar product of two vectors $a, b \in \mathbb{R}^n$ is denoted by $\langle a, b\rangle$. A (block-)diagonal combination of n matrices M_i is denoted by $\text{diag}(M_1, ..., M_n)$. The term $A \bullet B$ denotes the Frobenius inner product of two matrices A,B: $\text{trace}(AB^T)$.

2. PROBLEM FORMULATION

In this paper, we consider an interconnected large-scale linear time-invariant system consisting of N subsystems subject to linear equality constraints. More specifically, we assume that the subsystems are not coupled dynamically but only through the constraints. The dynamics of the ith agent can thus be written as a semi-explicit DAE of the form

$$\dot{x}_i = A_{xx,i}x_i + A_{xy,i}y_i + B_{x,i}u_i \tag{1a}$$

$$0 = \sum_{j=1}^{N} A_{yx,ij}x_j + \sum_{j=1}^{N} A_{yy,ij}y_j + B_{y,i}u_i, \tag{1b}$$

where $x_i \in \mathbb{R}^{n_{x_i}}$, $y_i \in \mathbb{R}^{n_{y_i}}$, $u_i \in \mathbb{R}^{m_i}$, $A_{xx,i} \in \mathbb{R}^{n_{x_i}\times n_{x_i}}$, $A_{xy,i} \in \mathbb{R}^{n_{x_i}\times n_{y_i}}$, $A_{yx,ij} \in \mathbb{R}^{n_{y_i}\times n_{x_j}}$, $A_{yy,ij} \in \mathbb{R}^{n_{y_i}\times n_{y_j}}$, $B_{x,i} \in \mathbb{R}^{n_{x_i}\times m_i}$ and $B_{x,i} \in \mathbb{R}^{n_{y_i}\times m_i}$. More compactly, this is

$$\dot{x} = A_{xx}x + A_{xy}y + B_x u,\ x(0) = x_0. \tag{2a}$$

$$0 = A_{yx}x + A_{yy}y + B_y u. \tag{2b}$$

Here, we denote by $x \in \mathbb{R}^{n_x}$ the differential variables, by $y \in \mathbb{R}^{n_y}$ the algebraic variables and by $u \in \mathbb{R}^m$ the input to the system. The dimensions satisfy $n_x = \sum_{k=1}^{N} n_{x_k}$, $n_y = \sum_{k=1}^{N} n_{y_k}$, $n = n_x + n_y$ and $m = \sum_{k=1}^{N} m_k$. Note that $y(0)$ is not specified but it is determined through the algebraic constraints. This is necessary to ensure that the whole initial condition of the system is admissible and does not cause any discontinuous behavior. Furthermore, we assume that $A_{xx} = \text{diag}(A_{xx,i})$, $A_{xy} = \text{diag}(A_{xy,i})$, i.e. the systems are not coupled through the dynamic equations.

We make the following assumptions about A_{yx} and A_{yy}.

Assumption 1.

- A_{yx} and A_{yy} are sparse matrices that signify which subsystems are coupled through the algebraic constraints.
- A_{yy} is invertible.
- A_{yy} is row diagonally dominant.

The first assumption is reasonable for large-scale systems where every agent has only few neighbors relative to the total number of agents. The second assumption ensures that there is a unique solution to the algebraic constraint, and it makes the system a DAE of index 1, where the index is the number of derivations of parts of the system equations with respect to time necessary to obtain an ODE. The third assumption is necessary to guarantee that the system can be simulated in a distributed fashion using a Jacobi algorithm, as will be seen in Section 3.2. If the third assumption is not met, there are alternative methods to solve the algebraic part, e.g. BiCGstab as presented in Van der Vorst (1992). This method can also be completely distributed (see e.g. Yang and Brent (2002)), when scalar products are computed using a consensus scheme. These methods require, however, a higher communication effort.

A typical real world example satisfying these assumptions is (a linearized version of) the power system (see Iavernaro and La Scala (1998)), where the dynamic states are the states of each generator (frequency, mechanical power, etc.) and the algebraic states are voltages, currents or electrical power flows.

Lemma 1. (from Bender and Laub (1987)) If the matrix A_{yy} in system (2) is invertible, the system has no impulsive modes and the pencil $(sE - A)$ is regular, where

$$E = \begin{bmatrix} I^{n_x\times n_x} & 0 \\ 0 & 0 \end{bmatrix},\ A = \begin{bmatrix} A_{xx} & A_{xy} \\ A_{yx} & A_{yy} \end{bmatrix}.$$

By Lemma 1 and Assumption 1, the treated system class treated has the important characteristic of no discontinuous behavior.

Instead of directly treating (2), one could plug the solution of (2b) into (2a) to obtain an ODE. This yields

$$\dot{x} = (A_{xx} - A_{xy}A_{yy}^{-1}A_{yx})x + (B_x - A_{xy}A_{yy}^{-1}B_y)u.$$

However, this system description lacks the sparsity structure of the original system. The structure is important to enable distributed control design methods with local information exchange. Distributed design is helpful for this system class when privacy of the subsystems is an issue. Furthermore, a controller designed for the system in this reduced form does not explicitly incorporate feedback information of the static states. Therefore, we work with the system in its original form given in (2).

The neighborhood structure will be determined by the coupling through the algebraic constraints and is thus derived from the sparsity structure of the matrices A_{yx} and A_{yy}. This means that if systems are coupled in constraints, they are considered to be neighbors. In order to define the set of neighbors of a subsystem i, we consider the directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ associated with the matrices A_{yx} and A_{yy}. The vertex set $\mathcal{V}$ is given by the set of subsystems $\mathcal{V} = \{1, ..., N\}$, and the edge set $\mathcal{E}$ contains the edge $(j, i) \in \mathcal{E}$ iff $A_{yx,ij} \neq 0 \vee A_{yy,ij} \neq 0$. This means an edge $(j, i) \in \mathcal{E}$ iff subsystem i is influenced directly by the states of agent j.

We define the set of neighboring nodes to node i as

$$\mathcal{N}_i = \{j | (i, j) \in \mathcal{E} \text{ or } (j, i) \in \mathcal{E}\}.$$

Additionally, we define the set of nodes j influenced by node i as $\mathcal{N}_{\text{out},i} = \{j | (i, j) \in \mathcal{E}\}$ and the set of nodes j influencing node i as $\mathcal{N}_{\text{in},i} = \{j | (j, i) \in \mathcal{E}\}$.

The goal is to design a structurally constrained constant linear feedback law $u(t) = -[K_x\, K_y][x^T(t)\, y^T(t)]^T = -K[x^T(t)\, y^T(t)]^T$ minimizing the following cost functional

$$J(x, u) = \int_0^T x^T(t)Q_x x(t) + y^T(t)Q_y y(t) + u^T(t)Ru(t)\mathrm{d}t, \tag{3}$$

By Assumption 1 and Lemma 1, the system has no impulsive modes, thus guaranteeing that the cost functional (3) exists. The symmetric, weighting matrices $Q \in \mathbb{R}^{n\times n}$ (positive semidefinite) and $R \in \mathbb{R}^{m\times m}$ (positive definite) are assumed to be block-diagonal with the block-dimension corresponding to the respective subsystem size. Under this assumption, the cost functional is separable for each agent as

$$J(x, u) = \sum_{i=1}^{N} \int_0^T x_i^T(t)Q_{x,i}x_i(t) + y_i^T(t)Q_{y,i}y_i(t) + u_i^T(t)R_i u_i(t)\mathrm{d}t.$$

As a result the optimization problem to find the optimal feedback is not coupled in the cost but only in the constraint to satisfy the underlying dynamics. Further, we require the following assumption to be satisfied.

Assumption 2. The feedback matrix K is constrained to have a distributed structure where communication between subsystems is only allowed among neighbors, so the blocks $K_{x,ij} \neq 0$ and $K_{y,ij} \neq 0$ only if $j \in \mathcal{N}_i$.

Note that the optimization problem is in general not convex with respect to the controller parameters

3. CONTROL SYNTHESIS

In this section, we present our main result. The approach follows an optimal control approach using the Lagrange function of the optimization problem. From this, we first derive the adjoint states for the considered class of singular systems, which themselves also constitute a singular system. The adjoint states are used to take the dynamic and algebraic constraints into account. It is then shown that with the adjoint states, the problem can be distributed among the subsystems, and with the adjoint states we also determine a gradient descent direction with respect to the feedback matrix to minimize the cost functional (3). Afterwards, we explain in detail how everything can be computed distributedly. In the last subsection, we treat the the problem of the dependency of the approach with respect to the initial condition of the state x_0.

3.1 Adjoint states and gradient descent direction

The distributed computation of the gradient is based on the simulation of state trajectories. In addition, adjoint states are needed which we first derive. We will use the following abbreviations: $A_{K,xx} = (A_{xx} - B_xK_x), A_{K,xy} = (A_{xy} - B_xK_y), A_{K,yx} = (A_{yx} - B_yK_x), A_{K,yy} = (A_{yy} - B_yK_y)$.

The Lagrange function for the problem is written as

$$\begin{aligned} L = \int_0^T \Big(& x^TQ_xx + y^TQ_yy + x^TK_x^TRK_xx \\ & + y^TK_y^TRK_yy + 2y^TK_y^TRK_xx \\ & + \lambda_x^T(\dot{x} - A_{K,xx}x - A_{K,xy}y) \\ & + \lambda_y^T(A_{K,yx}x + A_{K,yy}y)\Big)\mathrm{d}t \\ & + \mu^T(x_0 - x(0)). \end{aligned} \tag{4}$$

The adjoint equations are obtained by requiring that $\frac{\partial L}{\partial x} = 0$ and $\frac{\partial L}{\partial y} = 0$. To get the first adjoint state we determine $\langle \frac{\partial L}{\partial x}, v\rangle$ with v representing a small variation of x as

$$\begin{aligned} \langle \frac{\partial L}{\partial x}, v\rangle = & \int_0^T 2v^TQ_xx + 2v^TK_x^TRK_xx + 2v^TK_x^TRK_yy \\ & + \lambda_x^T(\dot{v} - A_{K,xx}v) + \lambda_y^T(A_{K,yx}v)\mathrm{d}t \\ & - \mu^Tv(0) \\ = & \int_0^T v^T(2(Q_x + K_x^TRK_x)x + 2K_x^TRK_yy - \dot{\lambda}_x \\ & - A_{K,xx}^T\lambda_x + A_{K,yx}^T\lambda_y)\mathrm{d}t - \mu^Tv(0) + [\lambda_x^Tv]_0^T. \end{aligned}$$

By varying v, we get the first adjoint equation as

$$\begin{aligned} \dot{\lambda}_x = & -A_{K,xx}^T\lambda_x + A_{K,yx}^T\lambda_y + 2(Q_x + K_x^TRK_x)x \\ & + 2K_x^TRK_yy, \quad \lambda_x(T) = 0, \\ \mu = & -\lambda_x(0). \end{aligned} \tag{5a}$$

Since μ is not necessary in the following, we disregard it but it gives us the justification that $\lambda_x(0)$ is free while $\lambda_x(T)$ is fixed to 0. For the second adjoint state we then determine $\langle \frac{\partial L}{\partial y}, w\rangle$ as

$$\begin{aligned} \langle \frac{\partial L}{\partial y}, w\rangle = & \int_0^T 2w^TQ_yy + 2w^TK_y^TRK_yy + 2w^TK_y^TRK_xx \\ & - \lambda_x^TA_{K,xy}w + \lambda_y^TA_{K,yy}w\mathrm{d}t \\ = & \int_0^T w^T(2Q_yy + 2K_y^TRK_yy + 2K_y^TRK_xx \\ & - A_{K,xy}^T\lambda_x + A_{K,yy}^T\lambda_y)\mathrm{d}t. \end{aligned}$$

Analogously, by varying w, the second adjoint equation is obtained which constitutes the algebraic component of the adjoint system:

$$0 = - A_{K,xy}^T \lambda_x + A_{K,yy}^T \lambda_y + 2(Q_y + K_y^T R K_y) y + 2 K_y^T R K_x x. \tag{5b}$$

Proposition 2. The gradients of the cost functional with respect to the controller blocks $K_{x_{ij}}$ and $K_{y_{ij}}$ are given by

$$(\nabla_{K_x} J)_{ij} = \int_0^T -2R_i u_i x_j^T + (B_{x,i}^T \lambda_{x_i} - B_{y,i}^T \lambda_{y_i}) x_j^T \mathrm{d}t \tag{6a}$$

$$(\nabla_{K_y} J)_{ij} = \int_0^T -2R_i u_i y_j^T + (B_{x,i}^T \lambda_{x_i} - B_{y,i}^T \lambda_{y_i}) y_j^T \mathrm{d}t \tag{6b}$$

Proof. The gradients are determined from the Lagrange function. For the feedback matrix for the dynamic states we get

$$\begin{aligned}
\langle \frac{\partial L}{\partial K_{x,ij}}, M_{x,ij} \rangle = \int_0^T & 2 x_j^T M_{x,ij}^T R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} K_{x,ik} x_k \\
& + 2 x_j^T M_{x,ij}^T R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} K_{y,ik} y_k \\
& + x_j^T M_{x,ij}^T B_{x,i}^T \lambda_{x_i} - x_j^T M_{x,ij}^T B_{y,i}^T \lambda_{y_i} \mathrm{d}t \\
= \int_0^T & 2 R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} (K_{x,ik} x_k + K_{y,ik} y_k) x_j^T \\
& + (B_{x,i}^T \lambda_{x_i} - B_{y,i}^T \lambda_{y_i}) x_j^T \mathrm{d}t \bullet M_{x,ij},
\end{aligned} \tag{7}$$

where $M_{x,ij}$ is a variation in $K_{x,ij}$. Then, we use the fact that $\sum_{k \in \mathcal{N}_{\mathrm{in},i}} (K_{x,ik} x_k + K_{y,ik} y_k) = -u_i$ to achieve the result. Similarly, for the feedback matrix for the static states we obtain

$$\begin{aligned}
\langle \frac{\partial L}{\partial K_{y,ij}}, M_{y,ij} \rangle = \int_0^T & 2 y_j^T M_{y,ij}^T R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} K_{y,ik} y_k \\
& + 2 y_j^T M_{y,ij}^T R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} K_{x,ik} x_k \\
& + y_j^T M_{y,ij}^T B_{x,i}^T \lambda_{x_i} - y_j^T M_{y,ij}^T B_{y,i}^T \lambda_{y_i} \mathrm{d}t \\
= \int_0^T & 2 R_i \sum_{k \in \mathcal{N}_{\mathrm{in},i}} (K_{y,ik} y_k + K_{x,ik} x_k) y_j^T \\
& + (B_{x,i}^T \lambda_{x_i} - B_{y,i}^T \lambda_{y_i}) y_j^T \mathrm{d}t \bullet M_{y,ij}.
\end{aligned} \tag{8}$$

With similar reasoning, we get the given result from the proposition which concludes the proof.

Using the proposed gradient descent direction, the following algorithm can be used to find a locally optimal controller.

Algorithm 1. (1) Simulate the states $x_i(t), y_i(t)$ of system (1) for the finite horizon T.
(2) Simulate the adjoint states $\lambda_{x_i}(t), \lambda_{y_i}(t)$ for the same finite horizon T in the backwards direction according to Eqs. (5).
(3) Every agent calculates the respective entries of the gradient by the formulas given in Proposition 2.
(4) For each neighboring agent j, update

$$\begin{aligned}
K_{x,ij}^{(k+1)} &= K_{x,ij}^{(k)} - \gamma_k (\nabla_{K_x} J)_{ij}^{(k)}, \\
K_{y,ij}^{(k+1)} &= K_{y,ij}^{(k)} - \gamma_k (\nabla_{K_y} J)_{ij}^{(k)}.
\end{aligned}$$

with a suitable scalar step length γ_k, independent of i, j.
(5) If all $||(\nabla_K J)_{ij}^{(k)}|| < \epsilon$, or if a different stopping criterion is satisfied, stop. Otherwise, increase k and go back to 1.

As the initializing feedbacks K_x^0 and K_y^0 every choice is possible that satisfies the allowed structure of the controller. An obvious choice would be the zero matrix of appropriate size.

A possible choice for a step length that shows good performance is given in Section 3.3.

Remark 1. Note that unless the resulting controller has entries such that rows or columns of $(A_{yy} - B_y K_y)$ become linearly dependent, thus making the matrix non-invertible, the controller does not cause any impulsive modes and the pencil $(sE - (A - BK))$ will also be regular.

Remark 2. We should stress that the approach finds a controller for systems that inherently satisfy the given equality constraints which are part of the system dynamics of semi-explicit singular systems by themselves. The problem treated in this paper is not directly related to controller design for ODE systems that ensures satisfaction of equality constraints which we would want to impose on the system. Whether the presented approach can be used to find controllers that guarantee constraint satisfaction for designed constraints, not inherent to the dynamics, remains to be seen in future work.

As can be expected, the communication effort of the distributed design is higher than for a centralized design. The reason for this is the requirement to simulate the system trajectories which requires neighborhood information. But it should be noted that the design can be done entirely offline. During the process, only a state measurement exchange between neighboring nodes is necessary to compute the control input using the feedback K.

3.2 Distributed computation

In this section, we show how it is possible to simulate the states and adjoint states using only local and neighborhood information and thence how to compute the gradient. One method to simulate semi-explicit singular systems as considered in this paper is to solve the algebraic part and to plug in the solution into the dynamic part.

The closed loop of the dynamic state (1a) is written as

$$\dot{x}_i = A_{xx,i} x_i - B_{x,i} \sum_{j \in \mathcal{N}_{\mathrm{in},i}} (K_{x,ij} x_j + K_{y,ij} y_j).$$

It is easy to see that each agent i can update its dynamic state by communicating state information with its neighbors, i.e. knowing x_j and y_j for all $j \in \mathcal{N}_{\mathrm{in},i}$. As for the algebraic part, we can rewrite (1b) as

$$- \sum_{j \in \mathcal{N}_{\mathrm{in},i}} A_{K,yy,ij} y_j = \sum_{j \in \mathcal{N}_{\mathrm{in},i}} A_{K,yx,ij} x_j.$$

Using a simple Jacobi algorithm (see Bertsekas and Tsitsiklis (1989)), this system of linear equations can be solved for y_j using only information from the neighbors $\mathcal{N}_{\mathrm{in},i}$ following the iteration

$$y_i = -A_{K,yy,ii}^{-1} \big[\sum_{j \in \mathcal{N}_{\mathrm{in},i}} A_{K,yy,ij} y_j + \sum_{j \in \mathcal{N}_{\mathrm{in},i}} A_{K,yx,ij} x_j \big].$$

Similar investigations can be done for the adjoint state. The dynamic part is component-wise rewritten as

$$\begin{aligned}
\dot{\lambda}_{x_i} = & - \sum_{j \in \mathcal{N}_{\mathrm{out},i}} A_{K,xx,ji}^T \lambda_{x_j} - 2 \sum_{j \in \mathcal{N}_{\mathrm{out},i}} K_{x,ji}^T R_j u_j. \\
& + \sum_{j \in \mathcal{N}_{\mathrm{out},i}} A_{K,yx,ji}^T \lambda_{y_j} + 2 Q_{x_i} x_i
\end{aligned}$$

Similarly for the static co-state we obtain

$$0 = -\sum_{j\in\mathcal{N}_{\text{out},i}} A_{K,xy,ji}^T \lambda_{x_j} - 2\sum_{j\in\mathcal{N}_{\text{out},i}} K_{y,ji}^T R_j u_j. + \sum_{j\in\mathcal{N}_{\text{out},i}} A_{K,yy,ji}^T \lambda_{y_j} + 2Q_{y_i} y_i$$

This can also be solved for $\lambda_{y,j}$ using a Jacobi algorithm using only information (states, inputs, system model) from neighbors.

As for the gradient, the formulation in Proposition 2 makes it obvious that the gradients can be computed locally if agent i can communicate with agent j, and since the controller structure is restricted to neighborhood information, this is possible.

3.3 Step size selection

For the step size γ_k, we suggest a Barzilai-Borwein (BB) step size which is given by Barzilai and Borwein (1988)

$$\gamma_k = \frac{\langle \Delta\text{vec}(K), \Delta\text{vec}(K)\rangle}{\langle \Delta\text{vec}(K), \Delta\text{vec}(\nabla_K J)\rangle} \tag{9}$$

where $\Delta\text{vec}(K) = \text{vec}(K^{(k)}) - \text{vec}(K^{(k-1)})$ and where $\Delta\text{vec}(\nabla_K J) = \text{vec}((\nabla_K J)^{(k)}) - \text{vec}((\nabla_K J)^{(k-1)})$. In order to guarantee convergence of the gradient method to a stationary point, the step size γ_k needs to satisfy the so-called Armijo rule (Bertsekas, 1999, Section 1.2) which is stated as follows

$$J(K^{(k)} + \gamma_k s^{(k)}) - J(K^{(k)}) \leq \alpha\gamma_k \text{vec}(\nabla_K J(K^{(k)}))^T s^{(k)} \tag{10}$$

where $\alpha \in (0,1)$, γ_k is initially the BB step size and where in our case $s^{(k)} = -\text{vec}(\nabla_{K^{(k)}} J(K^{(k)}))$. It can be shown that this condition is always satisfied for sufficiently small γ_k. The BB step size can be computed distributedly in two steps. In the first step, each agent computes a local estimate of the numerator and denominator of the BB-step size γ_k using its own entries of the feedback and gradient matrix. In a second step, a distributed consensus algorithm will lead to the mean of all local estimates which then gives the value of (9). Furthermore, the Armijo rule can also be tested in a completely distributed fashion. We refer to Deroo et al. (2012) for more details on the distribution of the step size computations.

3.4 Averaged initial condition

It can be seen that the result of Algorithm 1 depends not only on the initial condition K_0 related to the decision variables of the optimization, but also on the state initial condition x_0 chosen for the design process. Usually, during the controller design, the initial condition of the process is not known so we would like a controller which performs well for every initial condition. Additionally, we do not know how to systematically pick one specific initial condition.

Furthermore, since the approach is model-free (at least from a global point of view), we require enough excitation of the system such that the agents can extract model information from the trajectories. If only one arbitrarily picked initial condition is used, it might happen that some agents do not receive information from the whole system, especially if mostly far away nodes are excited with the initial condition. Of course, the controller resulting from one specific initial condition will work well if the design initial condition coincides with the process initial condition but this is usually not the case.

For these reasons, we would like to overcome the dependency with respect to the state initial condition and propose an averaging process over the initial condition. Therefore we make the following assumption about the initial condition of the state. Note that this assumption is only related to the x_0 used in the trajectory simulations used for the gradient computation leading to the gains, and not to the actual online process. Therefore, the original problem is not changed.

Assumption 3. The initial condition x_0 is a random variable, uniformly distributed on the surface of the n-dimensional unit sphere with expected value $\text{E}[x_0 x_0^T] = \frac{1}{n} I$, where I is the identity matrix.

The cost functional (3) has to be changed to the following

$$J(x,u) = \text{E}\left[\int_0^T x^T(t)Q_x x(t) + y^T(t)Q_y y(t) + u^T(t)Ru(t)\text{d}t\right], \tag{11}$$

where E represents the expected value with respect to the initial condition x_0. Thus, the cost becomes independent of x_0. Using the solutions for $x(t), y(t), \lambda_x(t), \lambda_y(T)$, it can be shown that all terms in the gradients (6) are linear in the term $x_0 x_0^T$. To do that, we need to show that all of the trajectories depend linearly on x_0. For $x(t)$ and $y(t)$, this is obvious. The solution for $\lambda_x(t)$ can be given as

$$\lambda_x(t) = 2\int_t^\infty e^{-A_\lambda(\tau - t)} A_x x(\tau)\text{d}\tau,$$

where

$$\begin{aligned} A_\lambda &= \Big[- A_{K,xx}^T + A_{K,yx}^T A_{K,yy}^{-T} A_{K,xy}^T\Big], \\ A_x &= \Big[A_{K,yx}^T A_{K,yy}^{-T}[-2K_y^T R K_x \\ &\quad + 2(Q_y + K_y^T R K_y) A_{K,yy}^{-1} A_{K,yx}] + 2(Q_x + K_x^T R K_x) \\ &\quad - 2K_x^T R K_x A_{K,yy}^{-1} A_{K,yx}\Big]. \end{aligned}$$

Because of the linear dependence of $x(\tau)$ on x_0, it follows that $\lambda_x(t)$ also depends linearly on x_0, and thus so does $\lambda_y(t)$. Hence, all products in the gradients are linear in $x_0 x_0^T$.

Because of the linearity and in order to average over unit initial conditions, we define $x_m(t), y_m(t), \lambda_{x,m}(t), \lambda_{y,m}(t)$ as the simulated trajectories based on the initial condition e_m, where e_m is the m-th unit base vector (zeros everywhere except the m-th entry).

Then similarly to Deroo et al. (2012), the algorithm to compute the gradient is adapted to the following.

Algorithm 2.
(1) Simulate the states $x_{i,m}(t), y_{i,m}(t)$ of System (1) for a finite horizon T for every initial condition e_m with $m = 1, ..., n$.
(2) Simulate the adjoint states $\lambda_{x_i,m}(t), \lambda_{y_i,m}(t)$ for the same finite horizon T in the backwards direction.
(3) Every agent calculates the respective entries of the gradients by

$$(\nabla_{K_x} J)_{ij} = \frac{1}{n}\left(\sum_{m=1}^{n} \int_0^T -2R_i u_{i,m} x_{j,m}^T + (B_{x,i}^T \lambda_{x_i,m} - B_{y,i}^T \lambda_{y_i,m}) x_{j,m}^T dt \right) \quad (12)$$

$$(\nabla_{K_y} J)_{ij} = \frac{1}{n}\left(\sum_{m=1}^{n} \int_0^T -2R_i u_{i,m} y_{j,m}^T + (B_{x,i}^T \lambda_{x_i,m} - B_{y,i}^T \lambda_{y_i,m}) y_{j,m}^T dt \right) \quad (13)$$

(4) For each neighboring agent j, update
$$K_{x,ij}^{(k+1)} = K_{x,ij}^{(k)} - \gamma_k (\nabla_{K_x} J)_{ij}^{(k)},$$
$$K_{y,ij}^{(k+1)} = K_{y,ij}^{(k)} - \gamma_k (\nabla_{K_y} J)_{ij}^{(k)}.$$
with a step length γ_k according to formula (9) that satisfies the Armijo rule (10).

(5) If all $||(\nabla_K J)_{ij}^{(k)}|| < \epsilon$, or if a different stopping criterion is satisfied, stop. Otherwise, increase k and go back to 1.

In order to use this algorithm, the agents need to have some global knowledge about the system (the total number of dynamic states n_x) because they need to know how often to run the simulation. Further, a protocol needs to determine which unit base vector is used at what time as the initial condition. However, both requirements are quite easy to satisfy. It has to be noted that this approach does not help to overcome the non-convexity of the optimization problem with respect to the decision variables (feedback parameters).

3.5 Discrete event-based trajectory simulation

4. NUMERICAL EXAMPLE

In this section, we present several numerical investigations to illustrate the contributions of this paper. First, we give a short illustrative example. Second, we compare the resulting controllers to a centralized approach. Third, the advantages of using the averaged initial condition are shown. Last, we show a practical example of a power system.

4.1 Illustrative example

To illustrate the approach to the reader, we treat an example system with 4 subsystems, each having only one dynamic state, one static state and one input.

The corresponding graph $\mathcal{G}$ is shown in Figure 1. We also show the allowed communications which are the undirected version of the same graph. From this we get that both feedback matrices K_x and K_y have the following form for all the iterations (except iteration 1 where the feedback matrices are zero)

$$K_{x/y} = \begin{bmatrix} * & 0 & * & * \\ 0 & * & 0 & * \\ * & 0 & * & 0 \\ * & * & 0 & * \end{bmatrix},$$

reflecting the desired communication structure. The nonzero entries are optimized by the distributed algorithm and every agent only manipulates its own row using information from its neighbors, e.g. subsystem 1 makes changes to K_{11}, K_{13} and K_{14}. In Figure 2, we also show the cost resulting from using the feedback matrices in every iteration and we see that the cost is monotonically decreasing, as guaranteed by the Armijo rule.

4.2 Comparison between sparse and non-sparse controller

In order to evaluate the performance of the presented distributed controller, we compare the performance of the resulting controller of Algorithm 2 with a prescribed sparsity structure with the resulting controller without the structure (K is a full matrix). As test systems, we randomly construct singular systems of the form given in Eq. (2). We use systems with $N = 4$ subsystems, each having a random number of dynamic (between 1 and 3) and static states (1 or 2) and each having 1 input. We do the comparison for 100 randomly created systems for which a stabilizing feedback is obtained with the gradient method. On average, the systems have a total number of 12 states (dynamic and static combined). The weighting matrices Q and R are set to be identity matrices of appropriate sizes. We choose a finite horizon of $T = 5$. With these parameters, the gradient method needs 27 iterations on average to converge. Convergence is assumed to be achieved in these simulations when the largest element of the gradient is smaller than 10^{-3}. It turns out that the average cost difference is only 0.41% in favor of the non-sparse controller. The reason for that is that the optimal controller obtained without demanded sparsity structure still results in a sparse controller. It should be mentioned that the same is true for the feedback matrices resulting from the centralized infinite horizon result from Bender and Laub (1987). Naturally, it is to be expected that the difference increases with increasing system size. Nevertheless, only the presented approach is able to combine the goal of an optimal controller with the desire to secure privacy of the subsystems.

4.3 Averaged initial condition

In this section, we investigate the difference between Algorithm 1 and Algorithm 2. To do that, we compute the controller matrices with both algorithms and then compare the cost caused by each controller. We use two different scenarios for the comparison: First, we use a different initial condition for the cost simulations than for the controller computations in Algorithm 1. Second, we use the same initial condition that was used in the design. We do this test for 100 randomly generated systems which were created the same way as in the previous numerical example. In this case, on average, the total

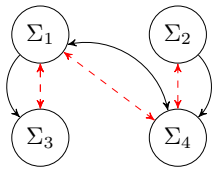

Fig. 1. Graph for the example system. Physical coupling: black, solid. Control communication: red, dashed

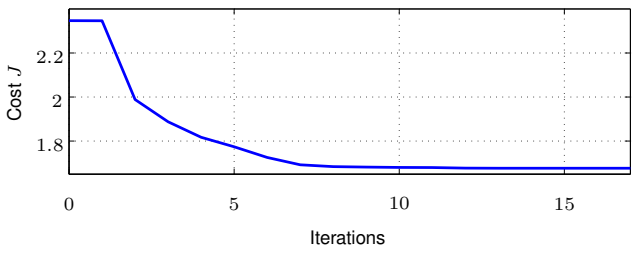

Fig. 2. Cost evolution over the iterations of the algorithm

number of states is also 12. The simulation horizon is $T = 1$. As for the simulation effort, the Jacobi algorithm for typical systems of the used system size needs on average 16 iterations to converge. When using a different initial condition for the cost simulations, the resulting controllers with the averaged initial condition produce a cost that is on average 10.72% smaller than the cost with one arbitrary initial condition, displaying the positive effect of the averaging process. In addition, according to these simulations, Algorithm 2 only needs 20 iterations on average to converge, while Algorithm 1 needs 42. Granted, the individual iterations of the averaging process are more costly, but combined with the cost improvement, Algorithm 2 should be the preferred option in this scenario.

In the second case, where we use the same initial condition for the cost calculation that the cost with the controller of Algorithm 1 is 0.22% lower than the cost with the controller of Algorithm 2 which can be considered to be negligible. This illustrates the Algorithm 2 achieves optimality independently of the initial condition of the process.

4.4 Small power system

As a practical example, we apply the method to a small power system with $n_{\text{total}} = 8$ buses, where we have $n_{\text{g}} = 5$ generator nodes and 3 load nodes. We consider the following example of a power system with 8 nodes (see Liu et al. (2011)). The interconnection graph of the generators and loads is shown in Figure 3. Each of the 5 generators follows the differential equation given in Sauer and Pai (1998)

$$\begin{bmatrix} \dot{\delta}_i \\ \dot{\omega}_i \\ \dot{P}_{m,i} \\ \dot{a}_i \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{D_i}{J_i} & \frac{1}{J_i} & 0 \\ 0 & 0 & -\frac{1}{T_{u,i}} & \frac{1}{T_{u,i}} \\ 0 & -\frac{1}{T_{g,i}} & 0 & -\frac{R_i}{T_{g,i}} \end{bmatrix}}_{A_{xx,i}} + \underbrace{\begin{bmatrix} 0 \\ -\frac{1}{J_i} \\ 0 \\ 0 \end{bmatrix}}_{A_{xy,i}} E_{0,i} I_{q,i} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{T_{g,i}} \end{bmatrix}}_{B_{x,i}} u, \tag{14}$$

where δ_i is the phase angle of the generator, ω_i is the angle velocity, $P_{m,i}$ is the mechanical power, a_i is the valve position, D_i is the damping coefficient, J_i is the inertia constant, $T_{u,i}$ is a time constant representing the delay between the control valves and the turbine nozzles, $T_{g,i}$ is the time constant of the valve servomotor, R_i is the permanent speed droop of the turbine (see Kiani and Annaswamy (2012)). Each generator thus has the dynamic states $x_i = [\delta_i, \omega_i, P_{m,i}, a_i]^T$ and x of the total system is the stacked vector of all x_i.

Each generator node has the four algebraic variables $y_{G,i} = [V_i, \theta_i, I_{d,i}, I_{q,i}]$, each load node has only two algebraic states $y_{L,i} = [V_i, \theta_i]$, where V_i is the bus voltage magnitude, θ_i is the bus voltage angle, $I_{d,i}$ is the d-axis current and $I_{q,i}$ is the q-axis current. All algebraic variables need to satisfy the following algebraic constraints given in Sauer and Pai (1998)

$$0 = V_i e^{j\theta_i} + (R_{s,i} + jX_{d,i})(I_{d,i} + jI_{q,i}) e^{j\delta_i - \frac{\pi}{2}} - E_{0,i} e^{j\delta_i},\ i = 1, ..., n_{\text{g}} \tag{15}$$

$$0 = V_i e^{j\theta_i} (I_{d,i} - jI_{q,i}) e^{-j(\delta_i - \frac{\pi}{2})} + P_{L,i}(V_i) + jQ_{L,i}(V_i) - \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})},\ i = 1, ..., n_{\text{g}} \tag{16}$$

$$0 = P_{L,i}(V_i) + jQ_{L,i}(V_i) - \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})},\ i = n_{\text{g}} + 1, ..., n_{\text{total}}, \tag{17}$$

where $R_{s,i}$ and $X_{d,i}$ are internal resistors and impedances of the generators, Y is the magnitude of the admittance matrix of the network and α is the corresponding angle of the admittance matrix. The loads in this system are assumed to have a linear dependence with respect to the bus voltage, i.e. $P_{L,i}(V_i) = k_{p,i} V_i$ and $Q_{L,i}(V_i) = k_{q,i} V_i$. The equations (15)-(17) can be split into real and imaginary parts and then linearized around operating points $\delta_{i,0}, V_{i,0}, \theta_{i,0}, I_{d,i,0}, I_{q,i,0}$. The operating points result from a loadflow calculation. These linear equations can then be written in the form $0 = A_{yx} x + A_{yy} y$, where A_{yx} is block-diagonal, because the equations for node i only depend on x_i in the form of δ_i. A_{yy} has sparsity structure resembling that of the admittance matrix Y. The algebraic equations have no input so B_y is zero in this case.

The dynamical part can also be written in a form $\dot{x} = A_{xx} x + A_{xy} y + B_x u$, where A_{xx} and A_{xy} are block-diagonal with $A_{xx,i}$ and $A_{xy,i}$ on the respective diagonal. B_x is also block-diagonal with $B_{x,i}$ on the diagonal. For the considered system size, we have $x \in \mathbb{R}^{20}$ and $y \in \mathbb{R}^{26}$.

Thus, we obtain a singular system of the considered system class. We apply Algorithm 2 to the system with a time horizon $T = 10$s. The weighting matrices are $Q = I^{46\times 46}$ and $R = I^{5\times 5}$ with appropriate units. The algorithm stops after 60 iterations when the change in the gradient is considered small (less than $\epsilon = 0.02$). During these iterations, the nodes only communicate with their neighbors to simulate the trajectories which requires (considerable) information exchange but realizes privacy.

If we take a closer look at the interconnection graph of the system, we observe that none of the generators (the dynamic parts of the system) are directly coupled. This means that during the design phase, the generators do not communicate directly with each other but only with the load nodes. In the process phase K_x is for this reason essentially a decentralized controller and no dynamic states will be communicated between the systems. However, the nodes 6-8 which only have static states are coupled to the generator nodes and they can be used. Thus, K_y is not decentralized but its coupling structure is based on the structure of the admittance matrix. In summary, K_x and K_y

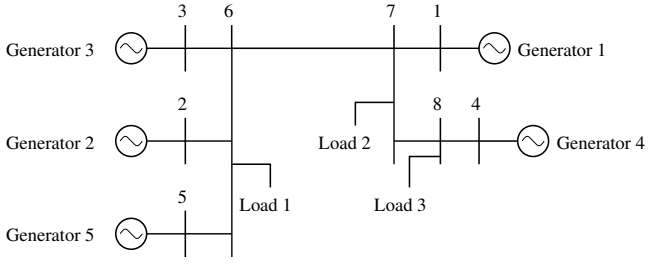

Fig. 3. 8 bus power system with 5 generators and 3 load nodes

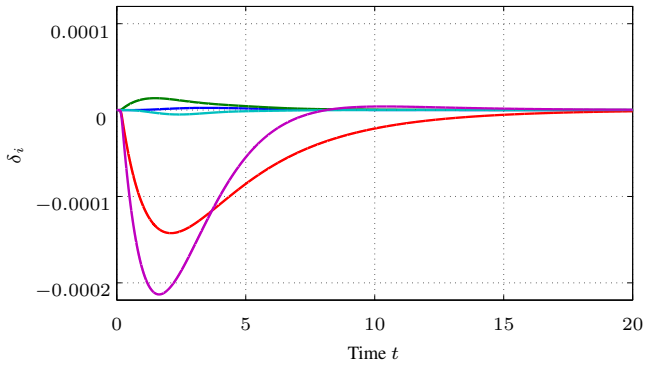

Fig. 4. Evolution of the phase angles δ_i of the 5 generators

have the following structures

$$K_x = \begin{bmatrix} * & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & * \end{bmatrix}, K_y = \begin{bmatrix} * & 0 & 0 & 0 & 0 & 0 & * & 0 \\ 0 & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & * & * & 0 & 0 \end{bmatrix}$$

In summary, for this example system, the main difference to classical decentralized control which is often used in power systems, is that this controller also makes use of local static variables (voltages, currents) and additionally static variables from adjacent load nodes.

For power systems, it is of interest, how the system reacts to a disturbance. For this reason, we simulate a load increase in one of the loads by offsetting the corresponding bus voltage between the time 0.1 and 0.2. The simulated state trajectories of the phase angles are shown in Figure 4. It can be seen that the controller from the presented algorithm handles the disturbance well and stabilizes the system to the equilibrium.

5. CONCLUDING REMARKS

In this paper, we present a distributed gradient descent method to compute a distributed linear controller for semi-explicit index 1 singular systems, i.e. systems with linear equality constraints, present in applications like the power distribution system. The approach guarantees privacy in the sense that dynamic models are shared with only a limited number of agents. The gradient method is guaranteed to converge using a Barzilai-Borwein step size and a check of the Armijo rule. The calculation of the descent direction uses simulated trajectories of the system states and of adjoint states. Averaging over the state initial condition achieves independence of the specific initial condition. The effectiveness of the approach is shown through several numerical simulations.

REFERENCES

Barzilai, J. and Borwein, J.M. (1988). Two-point step size gradient methods. *IMAJ of Numerical Anal.*, 8(1), 141–148.

Bender, D. and Laub, A. (1987). The linear-quadratic optimal regulator for descriptor systems. *IEEE Trans. on Automatic Cont.*, 32(8), 672–688.

Bertsekas, D.P. (1999). *Nonlinear Programming*. Athena Scientific, Belmont, MA.

Bertsekas, D.P. and Tsitsiklis, J.N. (1989). *Parallel and distributed computation*. Prentice Hall.

Cavoukian, A., Polonetsky, J., and Wolf, C. (2010). Smartprivacy for the smart grid. *Identity in the Information Society*, 3(2), 275–294.

Chang, T.N. and Davison, E.J. (2001). Decentralized control of descriptor systems. *IEEE Trans. on Automatic Cont.*, 46(10), 1589–1595.

Chen, H. and Baras, J.S. (2006). Scalable and distributed control laws for network flow optimization. In *Int. Symp. Math. Theory Netw. Sys., Kyoto, Japan*, 42–49.

Cobb, D. (1983). Descriptor variable systems and optimal state regulation. *IEEE Trans. on Automatic Cont.*, 28(5), 601–611.

Deroo, F., Ulbrich, M., Anderson, B.D.O., and Hirche, S. (2012). Accelerated iterative Distributed Controller Synthesis with a Barzilai-Borwein Step Size. In *51st IEEE Conf. on Decision and Control*.

Farokhi, F., Langbort, C., and Johansson, K.H. (2013). Optimal structured static state-feedback control design with limited model information for fully-actuated systems. *Automatica*, 49(2), 326 – 337.

Giselsson, P. and Rantzer, A. (2010). Distributed model predictive control with suboptimality and stability guarantees. In *49th IEEE Conf. on Decision and Control*, 7272–7277.

Iavernaro, F. and La Scala, M. (1998). Boundary values methods for time-domain simulation of power system dynamic behavior. *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, 45(1), 50–63.

Kiani, A. and Annaswamy, A. (2012). Distributed hierarchical control for renewable energy integration in a Smart Grid. In *IEEE PES Innovative Smart Grid Technologies*, 1–8.

Langbort, C., Chandra, R., and D'Andrea, R. (2004). Distributed control design for systems interconnected over an arbitrary graph. *IEEE Trans. on Automatic Cont.*, 49(9), 1502–1519.

Lewis, F.L. (1986). A survey of linear singular systems. *Circuits, Systems, and Signal Processing*, (I), 3–36.

Liu, J., Gusrialdi, A., Hirche, S., and Monti, A. (2011). Joint controller-communication topology design for distributed wide-area damping control of power systems. In *18th IFAC World Congress*.

Martensson, K. and Rantzer, A. (2012). A scalable method for continuous-time distributed control synthesis. In *American Control Conf.*, 6308–6313.

Sauer, P.W. and Pai, M.A. (1998). *Power system dynamics and stability*. Prentice Hall.

Shah, P. and Parrilo, P.A. (2010). H2-optimal decentralized control over posets: A state space solution for state-feedback. In *49th IEEE Conf. on Decision and Control*.

Siljak, D.D. (1978). *Large-Scale Dynamic Systems: Stability and Structure*. North-Holland.

Vamsi, A.S.M. and Elia, N. (2010). Design of distributed controllers realizable over arbitrary directed networks. In *49th IEEE Conf. on Decision and Control*, 4795–4800.

Van der Vorst, H.A. (1992). Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2), 631–644.

Yang, L. and Brent, R.P. (2002). The improved bicgstab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures. In *5th Int. Conf. on Algorithms and Architectures for Parallel Proc.*, 324–328.

Zargham, M., Ribeiro, A., and Jadbabaie, A. (2012). A distributed line search for network optimization. In *American Control Conf.*, 472–477.