Distributed control design with local model information and guaranteed stability

Frederik Deroo * **Martin Meinel** ** **Michael Ulbrich** ** **Sandra Hirche** *

* *Institute for Information-Oriented Control, Technische Universität München, Barer Str. 21, D-80333 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,* meinel@ma.tum.de, mulbrich@ma.tum.de

Abstract: Most results on distributed control design of large-scale interconnected systems assume a central designer with global model knowledge. The wish for privacy of subsystem model data raises the desire to find control design methods to determine an optimal control law without centralized model knowledge, i.e. in a distributed fashion. In this paper we present a distributed control design method with guaranteed stability to minimize an infinite horizon LQ cost functional. The introduction of adjoint states allows to iteratively optimize the feedback matrix using a gradient descent method in a distributed way, based on a finite horizon formulation. Inspired by ideas on stabilizing model predictive control, a terminal cost term is used, which gives a bound on the infinite horizon cost functional and ensures stability. A method is presented to determine that term in a distributed fashion. The results are validated using numerical experiments.

Keywords: Distributed control, Large-scale systems, Distributed optimization, Predictive Control

1. INTRODUCTION

Due to new technological challenges, like increasing system sizes, or sharper efficiency requirements, the control of large-scale interconnected dynamical systems has become an important research direction. Practical applications include various distribution systems, like water, power and gas, or transportation systems. Motivated by the advances in communication technologies, distributed control has emerged as a crucial tool for large-scale systems. Important results on distributed control, e.g. [Rotkowitz and Lall 2006, Langbort et al. 2004], consider the situation where a central designer with global model knowledge designs a structured control law and then passes the subcontrollers to the subsystems. In some situations, however, this centralized design approach might not be feasible. An issue that is becoming more and more important is privacy. Picture, for instance, a situation with economical competitors in a physically interconnected system, e.g. an electrical power grid. In such a scenario the individual subsystems naturally prefer to keep model data to themselves as much as possible, and are not willing to share their model and other information with a central designer. Furthermore, a global, overall model of a large-scale system becomes difficult to handle from a computational point of view, and centralized methods are likely to scale worse than distributed design methods [Martensson and Rantzer 2012b]. Hence, distributed design and analysis methods are required.

Therefore, from now on we will consider the problem of *control design with limited model information* if no centralized entity with global model knowledge exists. So far, there are only a few results available for the analysis and design of distributed control systems with limited model information. The authors in [Farokhi et al. 2012] investigate the best achievable performance under structural constraints compared to a unstructured control law for fully actuated discrete-time linear systems. They give the optimal control law for the case that the subsystems only know their own model, and they state bounds on the achievable performance for more general cases. In [Alam et al. 2011] the authors present an approach for the specific system structure of line graphs with an application to vehicle platoons and describe a suboptimal control design method where each system only knows its predecessor. A different approach to the distributed control design with limited model information is presented in [Martensson and Rantzer 2012a, Deroo et al. 2012] which was extended to singular systems in [Deroo et al. 2013]. Their approach is based on the assumption that neighboring, i.e. physically coupled systems, are allowed and willing to communicate. Based on the introduction of adjoint states, and using simulated trajectories of the system, a linear state feedback matrix is iteratively improved using a distributed gradient descent method to minimize a finite-horizon LQR cost functional. However, stability is not guaranteed from these results.

In this paper we present an approach to determine a stabilizing feedback control law where each subsystem uses model knowledge and trajectory information only from neighboring subsystems. The approach to achieve stability is inspired by results from the model predictive control (MPC) literature which consider stability in combination with finite-horizon cost functionals [Chen and Allgöwer 1998, Bitmead et al. 1990]. The main challenge involves the determination of a suitable terminal cost term that ensures stability. While this is usually done in a centralized fashion, we present a method to determine it

⋆ The work is supported in parts by the German Research Foundation (DFG) within the Priority Program SPP 1305 "Control Theory of Digitally Networked Dynamical Systems" and the TUM Munich School of Engineering.

in a distributed fashion by distributedly solving a structured LMI condition [Deroo et al. 2014]. Afterwards, the gradient descent method from [Deroo et al. 2012] is used to determine the actual feedback matrix. It should be stressed that there are numerous results in the literature that solve structured optimal control problems without the restriction to using only neighborhood model knowledge, or approaches that repeatedly design an open-loop input trajectory in an MPC-setting, see e.g. the survey articles [Necoara et al. 2011, Scattolini 2009]. The limitation on model knowledge and the design of a stabilizing static feedback law, however, make the problem harder, and represent the main contribution of this paper.

The remainder of the paper is organized as follows. In Section 2, the problem formulation is presented. Section 3 shows the necessary steps to distributedly determine an optimal control law with guaranteed stability. A numerical example is given in Section 4, and the paper concludes with a summary in Section 5.

Notation. A zero matrix of dimensions $m \times n$ is denoted by $0_{m\times n}$, an $n \times n$-identity matrix is written as $I_{n\times n}$. The term $A \bullet B$ denotes the Frobenius inner product of two matrices A and B, i.e. trace(AB^T). The set of symmetric $n \times n$ matrices is denoted by $\mathcal{S}^n$. The set of positive definite matrices is denoted by $\mathcal{S}^n_{++}$, positive semi-definite matrices by $\mathcal{S}^n_{+}$. The fact that a matrix $X \in \mathcal{S}^n_{+}$ or $X \in \mathcal{S}^n_{++}$ is also written as $X \succeq 0$ or $X \succ 0$, respectively. The column-wise vectorization of a matrix $A \in \mathbb{R}^{m\times n}$ is denoted by vec$(A) \in \mathbb{R}^{mn}$

2. PROBLEM FORMULATION

In this paper we consider an interconnected system consisting of N linear time-invariant subsystems. The dynamics of subsystem i are

$$\dot{x}_i(t) = A_{ii}x_i(t) + \sum_{\substack{j=1\\ j\neq i}}^{N} A_{ij}x_j(t) + B_i u_i(t), x_i(0) = x_{i,0}, \quad (1)$$

where $x_i \in \mathbb{R}^{n_i}$ is the state, $u_i \in \mathbb{R}^{m_i}$ is the input, $A_{ij} \in \mathbb{R}^{n_i\times n_j}$ and $B_i \in \mathbb{R}^{n_i\times m_i}$. The overall system is a concatenation of the subsystem states and is compactly written as

$$\dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0, \quad (2)$$

where $x \in \mathbb{R}^n, u \in \mathbb{R}^m, n = \sum_{i=1}^N n_i, m = \sum_{i=1}^N m_i$.

The interconnection structure of the overall system (2) is represented by a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The vertex set $\mathcal{V}$ represents the set of subsystems $\{1, \ldots, N\}$, and an edge $(j, i) \in \mathcal{E}$ iff the block $A_{ij} \neq 0_{n_i\times n_j}$. Hence, iff subsystem i is influenced by subsystem j there is an edge $(j, i) \in \mathcal{E}$. We define the following subsets of the graph: $\mathcal{N}_{\text{out},i} = \{j|(i, j) \in \mathcal{E}\}$ are the nodes j influenced by i, $\mathcal{N}_{\text{in},i} = \{j|(j, i) \in \mathcal{E}\}$ are the nodes j that influence node i and we define the set of neighboring nodes as the union of both as

$$\mathcal{N}_i = \{j|(i, j) \in \mathcal{E} \vee (j, i) \in \mathcal{E}\} = \mathcal{N}_{\text{in},i} \cup \mathcal{N}_{\text{out},i}. \quad (3)$$

In this paper, we design a stabilizing distributed feedback matrix K_{dist} and therefore consider the following optimization problem:

$$\min_{x,u} J_\infty(t, x, u) = \int_t^\infty x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\mathrm{d}\tau. \quad (4a)$$

$$\text{s.t. } \dot{x}(\tau) = Ax(\tau) + Bu(\tau) \quad (4b)$$

$$u(\tau) = -K_{\text{dist}}x(\tau) \quad (4c)$$

$$K_{\text{dist}} \text{ is stabilizing} \quad (4d)$$

The constraint (4d) is already implicitly included in (4a)-(4c) because (4a) only has a finite value for a stable closed-loop system, but we state it for clarity.

Before we proceed we make several assumptions.

Assumption 1. System (2) has a sparsity structure, i.e. no subsystem is connected to every other subsystem.

Assumption 2. The weighting matrix $Q \in \mathcal{S}^n_{+}$ has at most the same neighborhood block sparsity structure as A, i.e. $Q_{ij} = 0$ whenever $(i, j) \notin \mathcal{N}_i$. Note that this is the maximal allowed structure for Q meaning that Q can contain more zero blocks than A. The weighting matrix $R \in \mathcal{S}^n_{++}$ is block-diagonal. The block sizes result from the subsystem dimensions.

Assumption 3. The feedback matrix K_{dist} is constrained to have a sparse, distributed structure such that communication between subsystems is only allowed among neighbors, i.e. $K_{\text{dist},ij} = 0$ whenever $j \notin \mathcal{N}_i$.

Assumption 4. We require that also during the design phase, agents have only access to model and trajectory information from their neighborhood, i.e. agent i has no information about agents that are not part of the set $\mathcal{N}_i$.

Assumption 4 has the consequence that both the design and the implementation of the control law $u = -K_{\text{dist}}x$ is distributed.

3. CONTROL DESIGN WITH NEIGHBORHOOD INFORMATION

In this section, we describe how we design a distributed control law in a distributed fashion using only neighborhood information. Afterwards, we explain how stability is guaranteed using a terminal penalty term in the cost functional and how to determine this penalty term in a distributed way.

3.1 Adjoint states and gradient descent direction

In this section, we briefly describe how the gradient of the cost functional with respect to the individual entries of the feedback matrix can be computed using neighborhood information. This is in parts a recapitulation of results from [Deroo et al. 2012; 2013, Martensson and Rantzer 2012a]. For more details, we refer the reader to these papers.

We consider the interconnected LTI-system (2) and the finite horizon LQR cost functional

$$J(t, x, u) = x^T(t + t_f)Sx(t + t_f) + \int_t^{t+t_f} x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\mathrm{d}\tau. \quad (5)$$

Assumption 5. $S \in \mathcal{S}^n_{+}$ satisfies the same sparsity structure restrictions as Q from Assumption 2, i.e. $S_{ij} = 0$ whenever $(i, j) \notin \mathcal{N}_i$.

Furthermore, we define $A_K = A - BK_{\text{dist}}$. The relationship between J_∞ from (4a) and J from (5) and why we consider the finite horizon case for now will become clear soon as we will see that an appropriate choice for S bounds J_∞.

We formulate the Lagrangian of the problem with objective (5) and constraints (2) given by

$$\begin{aligned} L(t,x,u) =& x^T(t+t_f)Sx(t+t_f) + \int_t^{t+t_f} x^T(\tau)Qx(\tau) \\ &+ u^T(\tau)Ru(\tau) + \lambda^T(\tau)(\dot{x}(\tau) - A_K x(\tau))\mathrm{d}\tau \\ &+ \mu(x(t) - x_t) \end{aligned} \tag{6}$$

where x_t is the initial condition of the state x for the time interval $[t, t+t_f]$. Based on (6), we derive the dynamics of the adjoint states $\lambda(\tau)$ by requiring $\frac{\partial L}{\partial x} = 0$ as

$$\begin{aligned} \dot{\lambda}(\tau) &= -A_K^T\lambda(\tau) + 2(Q + K_{\text{dist}}^T R K_{\text{dist}})x(\tau), \\ \lambda(t+t_f) &= -2Sx(t+t_f),\ \mu = -\lambda(t),\ \tau \in [t, t+t_f]. \end{aligned} \tag{7}$$

Since μ is not necessary in the following, we disregard it but it gives us the justification that $\lambda(t)$ is free while $\lambda(t+t_f)$ is fixed. Given Assumptions 2 and 5, the dynamics of the adjoint states (7) have the same neighborhood structure as the original system (2).

Using these adjoint states it is possible to formulate the gradient of the cost functional J with respect to the feedback matrix K.

Proposition 1. The gradient of the cost functional (5) with respect to the control law blocks $K_{\text{dist},ij}$ is given by

$$(\nabla_{K_{\text{dist}}} J)_{ij} = \int_t^{t+t_f} -2R_i u_i x_j^T + B_i^T \lambda_i x_j^T \,\mathrm{d}\tau \tag{8}$$

Proof. See [Deroo et al. 2012].

Using the proposed gradient descent direction, the following algorithm is used to find a (possibly sub-)optimal control law. The suboptimality may result from the fact that the problem is non-convex.

Algorithm 1.

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

$$\begin{aligned} (\nabla_{K_{\text{dist}}} J)_{ij} =& \frac{1}{n}\left(\sum_{l=1}^{n} \int_t^{t+t_f} -2R_i u_{i,l} x_{j,l}^T \right. \\ &\left. + B_i^T \lambda_{i,l} x_{j,l}^T \mathrm{d}\tau \right) \end{aligned} \tag{9}$$

(4) For each neighboring agent j, update

$$K_{\text{dist},ij}^{(k+1)} = K_{\text{dist},ij}^{(k)} - \gamma_k (\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}. \tag{10}$$

with a Barzilai-Borwein step length γ_k which satisfies the Armijo rule [Deroo et al. 2012].
(5) If all $||(\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}|| < \epsilon$, or if a different stopping criterion is satisfied, stop. Otherwise, increase k and go back to 1.

As the initializing feedback K_{dist}^0 every choice is possible that satisfies the allowed structure of the control law. An obvious choice would be the zero matrix of appropriate size.

The attentive reader will have noticed that Algorithm 1 does not use the gradient formula from (8). The modified formula in the algorithm serves an averaging purpose to get rid of the dependence on the state initial condition x_0 used in the trajectory simulations. Naturally, this increases the design effort (linearly) but generally leads to better results [Deroo et al. 2012].

Remark 1. In order for the subsystems to be able to simulate the $x-$trajectories described by (1) and the λ-trajectories described by (7), the subsystems need to know their respective rows and columns of the matrix A_K. That means that they need to know how they are influenced by their neighbors, and in turn how they influence their neighbors. In addition, the nodes need to exchange the simulated trajectory data during the algorithm. Because the approach is based on simulated trajectories, the physical neighborhood topology described by the undirected version of graph $\mathcal{G}$ is essentially the minimal required information exchange topology in order for Algorithm 1 to work. Given state trajectories from other subsystems that are not direct neighbors, each subsystem could compute additional entries of the feedback matrix up to the full matrix thus possibly sacrificing privacy for improved performance.

So far we have designed a control law for a finite horizon cost functional. In the next subsection we will see that an appropriate choice of the terminal cost weighting matrix S allows us to find a control law that gives an upper bound on the infinite horizon cost functional J_∞ and thus an upper bound on the optimization problem (4).

3.2 Guaranteed stability

In this subsection, we describe how we use the terminal cost weighting matrix S to guarantee stability of the solution of Algorithm 1 by applying reasoning from MPC methods.

The principle of MPC is that an optimal control problem over a finite time-horizon t_f is solved, using recent state measurements to obtain an optimal open-loop input trajectory $u^*(\cdot)$. Then only the first part of the input trajectory is applied to the system for the duration Δt and is then re-optimized. In [Chen and Allgöwer 1998] it is shown that using MPC, guaranteed stability is achieved when the optimization problem contains a terminal cost term of a specific form. The idea is to determine the terminal cost term in such a way that it gives a bound on the infinite horizon cost functional. Therefore, we combine the method from the previous subsection with the terminal cost idea from MPC in order to obtain a stabilizing distributed control law obtained in a distributed fashion without centralized model information. Note that since we do not consider any constraints on the input or state, the optimization problem (4) is always feasible. We adopt the notation from the MPC literature that $\bar{x}(\tau; x(t), t)$ is the prediction of the state trajectory at time τ using state information from time t, and $\bar{x}^*(\tau; x(t), t, t+t_f)$ is the optimal predicted state trajectory at time τ using state information from time t with optimization horizon t_f.

To achieve stability, we propose a two step algorithm inspired by the results in [Chen and Allgöwer 1998] but instead of a classical centralized viewpoint, we put emphasis on our distributed setting for large-scale systems.

In the first step, the subsystems try to design a decentralized (block-diagonal) control law K_{dec} which stabilizes the system, using only local model data. To ensure that the system can be stabilized using a decentralized feedback law we have to make the following assumption.

Assumption 6. System (2) does not contain any decentralized fixed modes.

In a next step the agents determine a solution (P, δ) to the linear matrix inequality

$$A_{K_{\text{dec}}}^T \underbrace{\text{diag}(P^1, \ldots, P^N)}_{P} + \text{diag}(P^1, \ldots, P^N) A_{K_{\text{dec}}} + \gamma\delta \underbrace{(Q + K_{\text{dec}}^T R K_{\text{dec}})}_{Q_{K_{\text{dec}}}} \preceq 0, \quad (11)$$

where $A_{K_{\text{dec}}} = A - BK_{\text{dec}}$ and $\gamma > 0$ is a pre-specified constant.

Remark 2. Note that we restrict P and K_{dec} to be block-diagonal. This limitation introduces a restriction on the solution space and hence might involve conservativeness. If P is not block-diagonal, then $PA_{K_{\text{dec}}}$ has a different structure than the original system, if K_{dec} is not block-diagonal, the term $K_{\text{dec}}^T R K_{\text{dec}}$ would violate the pattern. The sparsity structure is not strictly necessary to find a solution to (11), because without it the LMI just takes the form of the centralized case in [Chen and Allgöwer 1998]. The structure is, however, the key to the distributed solution of (11) which is shown in the next subsection. If one is able find a sparsity pattern for P and K_{dec} that is not block-diagonal but still retains the neighborhood structure of the original system, these choices still allow the distributed solution.

If the solution of the LMI is a positive definite P and positive δ, the decentralized control law is stabilizing, otherwise it needs to be redesigned to achieve stability. The solution is used in the following lemma.

Lemma 1. Given a solution (P, δ) to the LMI (11), a stabilizing decentralized feedback $u(\tau) = -K_{\text{dec}} x(\tau)$ and by setting $S = \frac{P}{\gamma\delta}$, the infinite horizon cost functional $J_\infty(t, x, u)$ is bounded from above by the terminal cost term as

$$x(t_1)^T S x(t_1) \geq \int_{t_1}^{\infty} x^T(\tau) Q x(\tau) + u^T(\tau) R u(\tau) \mathrm{d}\tau. \quad (12)$$

Proof. By differentiating $x^T S x$ along a trajectory of system (2) and using (11), we get

$$\dot{V} = \frac{\mathrm{d}}{\mathrm{d}t} x^T S x = x^T (A_{K_{\text{dec}}}^T S + S A_{K_{\text{dec}}}) x \leq -x^T Q_{K_{\text{dec}}} x < 0. \quad (13)$$

Integrating both sides from t_1 to ∞ and knowing that K_{dec} is stabilizing gives the result from (12).

After having obtained the terminal cost term, in a second step the systems design a distributed control law K_{dist} using Algorithm 1 where $S = \frac{P}{\delta\gamma}$ is used in the terminal cost term. Next, we show that with this feedback control law the optimal value function is non-increasing.

Lemma 2. For $\tau \in (t, t + \Delta t]$ the optimal value function satisfies

$$J^*(x(\tau), \tau, \tau + t_f) \leq J^*(x(t), t, t + t_f) - \int_t^{\tau} x^T(s) Q x(s) + u^T(s) R u(s) \mathrm{d}s. \quad (14)$$

Proof. At time t, the optimal feedback input $\bar{u}^*(\cdot; x(t), t, t + t_f)$ is computed as $\bar{u}^* = -K_{\text{dist}} \bar{x}^*$ and we additionally have the optimal predicted state trajectory $\bar{x}^*(\cdot; x(t), t, t + t_f)$ on $[t, t + t_f]$. Then, the value of the optimal value function is

$$\begin{aligned} &J^*(x(t), t, t + t_f) \\ &= \int_t^{t+t_f} \bar{x}^{*^T}(s; x(t), t, t + t_f) Q \bar{x}^*(s; x(t), t, t + t_f) \\ &+ \bar{u}^{*^T}(s; x(t), t, t + t_f) R \bar{u}^*(s; x(t), t, t + t_f) \mathrm{d}s \\ &+ \bar{x}^{*^T}(t + t_f; x(t), t, t + t_f) S \bar{x}^*(t + t_f; x(t), t, t + t_f). \end{aligned} \quad (15)$$

For τ in the current control time interval $(t, t + \Delta t]$, the control input is determined through K_{dist} and the state trajectory is identical to the predicted trajectory, i.e. $x(s) = \bar{x}^*(s; x(t), t, t + t_f)$ for any $s \in [t, \tau]$. Assuming the suboptimal feedback

$$u(t) = \begin{cases} -K_{\text{dist}} x(t), & 0 \leq t \leq t_f \\ -K_{\text{dec}} x(t), & t > t_f. \end{cases}$$

is applied to the system, the generated trajectories are the same as the ones from the optimization time t, except for the shifted part in the time interval $[t + t_f, \tau + t_f]$, so we have that $\bar{x}(s; x(\tau), \tau) = \bar{x}^*(s; x(t), t, t + t_f)$ for all $s \in [\tau, t + t_f]$.

In order to determine the value of the cost function for $\tau \in (t, t + \Delta t]$ we have to compute the cost generated in the new part of the optimization interval, namely $[t + t_f, \tau + t_f]$. With the chosen input we know that in that time interval (13) is satisfied. Integrating (13) in the interval of interest $[t, +t_f, \tau + t_f]$ gives

$$\begin{aligned} &\bar{x}(\tau + t_f; x(\tau), \tau)^T S \bar{x}(\tau + t_f; x(\tau), \tau) \\ &+ \int_{t+t_f}^{\tau+t_f} \bar{x}(s; x(\tau), \tau)^T Q_{K_{\text{dec}}} \bar{x}(s; x(\tau), \tau) \mathrm{d}s \\ &\leq \bar{x}^*(t + t_f; x(t), t, t + t_f)^T S \bar{x}^*(t + t_f; x(t), t, t + t_f). \end{aligned}$$

Using that, we can bound the value of the cost functional for $\tau \in (t, t + \Delta t]$ as

$$\begin{aligned} &\bar{J}(x(\tau), \tau, \tau + t_f) \leq \\ &\int_{\tau}^{t+t_f} \bar{x}^*(s; x(t), t, t + t_f)^T Q_{K_{\text{dist}}} \bar{x}^*(s; x(t), t, t + t_f) \mathrm{d}s \\ &+ \bar{x}^*(t + t_f; x(t), t, t + t_f)^T S \bar{x}^*(t + t_f; x(t), t, t + t_f), \end{aligned}$$

where $Q_{K_{\text{dist}}} = Q + K_{\text{dist}}^T R K_{\text{dist}}$. Combining this with (15) and knowing that J^* is optimal, we get for $\tau \in (t, t + \Delta t]$

$$\begin{aligned} J^*(x(\tau), \tau, \tau + t_f) &\leq \bar{J}(x(\tau), \tau, \tau + t_f) \\ &\leq J^*(x(t), t, t + t_f) \\ &\quad - \int_t^{\tau} x(s)^T Q_{K_{\text{dist}}} x(s) \mathrm{d}s. \end{aligned} \quad (16)$$

With $Q \succeq 0$ and $R \succ 0$, this means that the value function is always non-increasing.

Note that the proof follows [Chen and Allgöwer 1998] closely but is adapted to the feedback input of this paper.

With that result we finally formulate the following theorem.

Theorem 1. Given Assumptions 2, 3, 5 and 6, the closed-loop of system (2) with an MPC implementation of the input $u(t) = -K_{\text{dist}} x(t)$ resulting from Algorithm 1 with $S = \frac{P}{\gamma\delta}$ from (11) is asymptotically stable.

Proof. We define the function $V(x) = J^*(x, t, t + t_f)$. The function has the properties:

- $V(0) = 0, V(x) > 0$ for $x \neq 0$,
- along the trajectory of the closed-loop system, there is for $0 \leq t_1 < t_2 \leq \infty$

$$V(x(t_2)) - V(x(t_1)) \leq -\int_{t_1}^{t_2} x^T(t) Q x(t) \mathrm{d}t, \quad (17)$$

where Lemma 2 is used. We now show asymptotic stability, i.e. for every $\epsilon > 0$, there is $\eta(\epsilon) > 0$ such that $||x(0)|| < \eta(\epsilon)$ implies $||x(t)|| < \epsilon$ for all $t \geq 0$ [Khalil 2002]. From (17), we get

$$V(x(\infty)) \leq V(x(0)) - \int_0^\infty x^T(t)Qx(t)\mathrm{d}t.$$

Because $V(x(\infty)) \geq 0$ and $V(x(0)) \leq \beta > 0$, the integral must exist and it is bounded. Clearly the closed loop system given by (2) with feedback $u = -K_{\text{dist}}x$ is uniformly continuous. Hence, we can apply Barbalat's lemma and conclude that $||x(t)|| \to 0$ as $t \to 0$, which implies asymptotic stability.

Again, note that the proof follows [Chen and Allgöwer 1998] but is adapted to our needs.

We have thus proven stability of an MPC-implementation of the feedback control law $u = -K_{\text{dist}}x$. Because the design of the feedback law with Algorithm 1 is independent of current state measurements no repeated re-optimization of K_{dist} is necessary for an MPC-implementation, and we can continuously apply the feedback law for all times. Note that this result resembles the centralized, full-information control case [Bitmead et al. 1990] where the optimal MPC control law corresponds to a static state-feedback. We would like to stress again that the presented control scheme is not technically a model predictive method, but a static feedback law. However, the reasoning from the MPC literature is required to prove stability.

3.3 Distributed computation of terminal cost term

In the previous subsection we have shown how we determine a stabilizing feedback control law $u = -K_{\text{dist}}x$ using a terminal cost term which involves the solution (P, δ) of (11). Because Algorithm 1 runs using only information from neighbors it would be undesirable if more than neighborhood information would be necessary to determine a solution of (11). Hence, in this subsection we apply a method developed in [Deroo et al. 2014] to find (P, δ) using only neighborhood information.

Because of the block-diagonal structures of P and K_{dec}, the problem of solving (11) clearly has a sparsity structure related to the structure of the matrix A. We denote by $\bar{A}$ a matrix where all zero entries of blocks A_{ij} with $j \in \mathcal{N}_i$ are set to arbitrary nonzero values. Then the structure of (11) is given by the binarization of the symmetric part of $\bar{A}$, denoted by $A^{\text{sym}} = \text{Bin}(|\bar{A}| + |\bar{A}|^T)$, where $|\bar{A}|$ denotes the componentwise norm of $\bar{A}$. The structure is also described by the undirected graph $\mathcal{G}_u = (\mathcal{V}_u, \mathcal{E}_u)$, where $(j, i) \in \mathcal{E}_u$ iff $A^{\text{sym}}_{ij} \neq 0$. Note that unlike in the graph $\mathcal{G}$, here an edge signifies the presence of an individual entry in the matrix instead of a matrix block. In [Kim et al. 2011] the authors show how to decompose an LMI-condition using the so-called range-space conversion method. The method, however, works only when the sparsity structure of the problem corresponds to a chordal graph. Hence, we have to make the following assumption.

Assumption 7. $\mathcal{G}_u$ is a chordal graph.

A chordal graph is defined to be a graph where every induced cycle has length less than 4. In other words, every cycle of length ≥ 4 has a chord, i.e. an edge joining non-consecutive vertices of the cycle [Gross and Yellen 2004]. Examples of graphs that are chordal graphs include line and star graphs as well as trees. If $\mathcal{G}_u$ does not satisfy this assumption and is not a chordal graph by itself, local communication with neighbors is not sufficient for the presented method of solving (11). However, it is possible to extend the communication topology of the network such that Assumption 7 is satisfied.

The principle approach here is to formulate an optimization problem that includes (11) as a constraint, to decompose the constraint [Kim et al. 2011] and then to apply distributed optimization methods [Meinel et al. 2014]. It turns out that for the solution of the optimization problem only neighborhood information is necessary.

First, we formulate the following optimization problem:

$$\min_{\delta \in \mathbb{R}, P^l \in \mathcal{S}^{n_l}} \; -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2} \parallel P^l \parallel_F^2 \tag{18a}$$

$$\text{s.t. } (11), \tag{18b}$$

$$P^l - \delta I_{n_l} \succeq 0 \text{ for } l = 1, \ldots, N, \tag{18c}$$

where the convexity parameter σ_δ and σ_{P^l} have to be chosen a priori. It can be shown that (11) has a feasible solution if and only if the optimal objective function value of problem (18) is negative (which then also implies $\delta > 0$). For details, we refer to [Deroo et al. 2014].

In order to decompose the LMI (11), it is necessary to express it in a basis of $\mathcal{S}^n$. Let $\mathcal{N} = \{1, \ldots, n\}$ and denote by E_{ij} the $n \times n$ symmetric matrix whose components (i, j) and (j, i) are 1 and all others are 0. Note that the set $\{E_{ij} : (i, j) \in \mathcal{N} \times \mathcal{N},\ i \leq j\}$ forms a basis of $\mathcal{S}^n$. Also, we define the indices of the lth subsystem block as

$$\mathcal{I}_l = \left\{\sum_{i=1}^{l-1} n_i + 1, \ldots, \sum_{i=1}^{l} n_i\right\} \times \left\{\sum_{i=1}^{l-1} n_i + 1, \ldots, \sum_{i=1}^{l} n_i\right\},$$

where $l = 1, \ldots, N$.

Next, for $l = 1, \ldots, N$ and $(i, j) \in \mathcal{I}_l$ we define

$$F^0 = -\gamma Q_{K_{\text{dec}}},$$

$$F^l_{ij} = \begin{cases} \frac{1}{2}\left(-A^T_{K_{\text{dec}}}E_{ij} - E_{ij}A_{K_{\text{dec}}}\right) & \text{if } i < j, \\ \frac{1}{2}\left(-A^T_{K_{\text{dec}}}E_{ji} - E_{ji}A_{K_{\text{dec}}}\right) & \text{if } i > j, \\ -A^T_{K_{\text{dec}}}E_{ij} - E_{ij}A_{K_{\text{dec}}} & \text{if } i = j. \end{cases}$$

With $i_l = i - \sum_{s=1}^{l-1} n_s$ it follows that (11) is equivalent to

$$F(P, \delta) := F^0\delta + \sum_{l=1}^{N} \sum_{(i,j)\in\mathcal{I}_l} F^l_{ij} P^l_{i_l j_l} \succeq 0,$$

and it follows that problem (18) is equivalent to

$$\min_{\delta \in \mathbb{R}, P^l \in \mathcal{S}^{n_l}} \; -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2} \parallel P^l \parallel_F^2 \tag{19a}$$

$$\text{s.t. } F(P, \delta) \succeq 0, \tag{19b}$$

$$P^l - \delta I_{n_l} \succeq 0 \text{ for } l = 1, \ldots, N. \tag{19c}$$

Definition 1. We define [Kim et al. 2011]:

$$\mathcal{S}^C = \{X \in \mathcal{S}^n : X_{ij} = 0 \text{ if } (i, j) \notin C \times C\}\, \forall C \subseteq \mathcal{N},$$

$$\mathcal{S}^C_+ = \left\{X \in \mathcal{S}^C : X \succeq 0\right\} \forall C \subseteq \mathcal{N},$$

$$J(C) = \{(i, j) \in C \times C : 1 \leq i \leq j \leq n\}\, \forall C \subseteq \mathcal{N}.$$

The cliques C of the graph are subsets where each node is adjacent to all the other nodes of the subset. Given the maximal cliques (cliques that are no subsets of other cliques) $C_1, \ldots, C_p$ of $\mathcal{G}_u$, define

$$J = \bigcup_{s=1}^{p} J(C_s),$$
$$\Gamma(i,j) = \{s : i \in C_s,\ j \in C_s\}\ \forall (i,j) \in J.$$

Then the LMI (19b) can be decomposed in the following way [Kim et al. 2011])

$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s - E_{ij} \bullet F(P,\delta) = 0$$
$$\text{for } (i,j) \in J \text{ and } Y^s \in \mathcal{S}_+^{C_s} \text{ for } s = 1,\dots,p.$$

With this it follows that problem (19) is equivalent to

$$\min_{\delta \in \mathbb{R}, P^l \in \mathcal{S}^{n_l}} -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2} \parallel P^l \parallel_F^2 \quad (20a)$$
$$\text{s.t. } \delta I_{n_l} - P^l \preceq 0 \text{ for } l = 1,\dots,N, \quad (20b)$$
$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s - E_{ij} \bullet F(P,\delta) = 0 \text{ for } (i,j) \in J, \quad (20c)$$
$$Y^s \in \mathcal{S}_+^{C_s} \text{ for } s = 1,\dots,p. \quad (20d)$$

To solve problem (20) in parallel we apply the distributed proximal center algorithm (DPCA) [Meinel et al. 2014, Necoara and Suykens 2008]. This optimization method uses dual decomposition to distribute the computation. As the objective function of (20) is not strictly convex, the differentiability of the gradient of the dual objective function cannot be guaranteed and thus a strictly convex regularization of the dual function would be necessary in order to apply the DPCA. To remedy this drawback we consider the following optimization problem

$$\min_{\delta \in \mathbb{R}, P^l \in \mathcal{S}^{n_l}} -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2}\|P^l\|_F^2 + \sum_{s=1}^{p} \frac{\sigma_{Y^s}}{2}\|Y^s\|_F^2 \quad (21a)$$
$$\text{s.t.} \quad \delta I_{n_l} - P^l \preceq 0 \text{ for } l = 1,\dots,N, \quad (21b)$$
$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s - E_{ij} \bullet F(P,\delta) = 0 \text{ for } (i,j) \in J, \quad (21c)$$
$$Y^s \in \mathcal{S}_+^{C_s} \text{ for } s = 1,\dots,p. \quad (21d)$$

As before it can be shown that (11) has a feasible solution if and only if the optimal objective function value of problem (21) is negative [Deroo et al. 2014].

In order to formulate the dual problem of problem (21) consider the corresponding Lagrangian with dual multipliers Λ and M

$$\begin{aligned}
&\mathcal{L}(\delta, P, Y, \Lambda, M) \\
&= -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2}\|P^l\|_F^2 + \sum_{s=1}^{p} \frac{\sigma_{Y^s}}{2}\|Y^s\|_F^2 \\
&\quad + \sum_{l=1}^{N} M^l \bullet \left(\delta I_{n_l} - P^l\right) \\
&\quad + \sum_{(i,j) \in J} \Lambda_{ij} \left(E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s - E_{ij} \bullet F(P,\delta) \right) \\
&= -x_\delta \delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \left(-X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2}\|P^l\|_F^2 \right) \\
&\quad + \sum_{s=1}^{p} \left(-X_Y^s \bullet Y^s + \frac{\sigma_{Y^s}}{2}\|Y^s\|_F^2 \right).
\end{aligned}$$

where

$$\begin{aligned}
x_\delta &= \sum_{(i,j) \in J} \Lambda_{ij} E_{ij} \bullet F^0 + 1 - \sum_{l=1}^{N} M^l \bullet I_{n_l}, \\
X_Y^s &= \sum_{(i,j) \in J(C_s)} -\Lambda_{ij} E_{ij}, \\
X_{P_{i_l j_l}}^l &= \sum_{(a,b) \in J} \Lambda_{ab} E_{ab} \bullet F_{ij}^l + M_{i_l j_l}^l \text{ for } (i,j) \in \mathcal{I}_l.
\end{aligned}$$

As the Lagrangian $\mathcal{L}$ is separable in δ, $P^1,\dots,P^N$, and $Y^1,\dots,Y^p$, the corresponding dual objective function

$$\begin{aligned}
f(\Lambda, M) = &\min_{\delta \in \mathbb{R}} \left\{ -x_\delta \delta + \frac{\sigma_\delta}{2}\delta^2 \right\} + \\
&\sum_{l=1}^{N} \min_{P^l \in \mathcal{S}^{n_l}} \left\{ -X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2} \parallel P^l \parallel_F^2 \right\} + \\
&\sum_{s=1}^{p} \min_{Y^s \in S_+^{C_s}} \left\{ -X_Y^s \bullet Y^s + \frac{\sigma_{Y^s}}{2} \parallel Y^s \parallel_F^2 \right\},
\end{aligned}$$

can be evaluated in parallel and is continuously differentiable due to the uniqueness of the solutions $\delta(\Lambda, M)$, $P^l(\Lambda, M)$, and $Y^s(\Lambda, M)$. Furthermore its gradient, given by

$$\begin{aligned}
\nabla_{\Lambda_{ij}} f(\Lambda, M) &= E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s(\Lambda, M) \\
&- E_{ij} \bullet \left(F^0 \delta(\Lambda, M) + \sum_{l=1}^{N} \sum_{(i,j) \in \mathcal{I}_l} F_{ij}^l P_{i_l j_l}^l(\Lambda, M) \right), \\
\nabla_{M^l} f(\Lambda, M) &= \delta(\Lambda, M) I_{n_l} - P^l(\Lambda, M)
\end{aligned}$$

for $(i,j) \in J$ and $l = 1,\dots,N$, is Lipschitz continuous with Lipschitz constant

$$\begin{aligned}
L = &\sum_{s=1}^{p} \|E_{C_s}\|^2 / \sigma_{W^s} + \sum_{l=1}^{N_C} \left(\|\hat{F}^l\|^2 + 1 \right) / \sigma_{P^l} + \\
&\sum_{(i,j) \in J} \left(E_{ij} \bullet F^0 \right)^2 / \sigma_\delta + n / \sigma_\delta,
\end{aligned}$$

where $E_{C_s} \in \mathbb{R}^{|J(C_s)| \times n^2}$ is the matrix that contains the rows $E_{ij}(:)^T$ for $(i,j) \in J(C_s)$ and $\hat{F}^l \in \mathbb{R}^{(|J|) \times n_l^2}$ is the matrix that contains rows $(E_{ab} \bullet F_{i^1 j^1}^l, \dots, E_{ab} \bullet F_{i^{|\mathcal{I}_l|} j^{|\mathcal{I}_l|}}^l)$ for $(a,b) \in J$ [Deroo et al. 2014].

Having obtained the smooth dual function we apply the DPCA to maximize it in parallel.

Algorithm 2. (Distributed solution of (11)). For $k \geq 0$ do

(1) Given $M^{l,k}$ and components Λ_{ij}^k, the agents compute in parallel

$$\begin{aligned}
\delta^{k+1} &= \arg\min_{\delta \in \mathbb{R}} \left\{ -x_\delta \delta + \frac{\sigma_\delta}{2}\delta^2 \right\}, \\
P^{l,k+1} &= \arg\min_{P^l \in \mathcal{S}^{n_l}} \left\{ -X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2}\|P^l\|_F^2 \right\}, \\
Y^{s,k+1} &= \arg\min_{Y^s \in \mathcal{S}_+^{C_s}} \left\{ -X_Y^s \bullet Y^s + \frac{\sigma_{Y^s}}{2}\|Y^s\|_F^2 \right\},
\end{aligned}$$

for $l = 1,\dots,N$ and $s = 1,\dots,p$. Moreover, they send δ^{k+1}, $P^{l,k+1}$, and $Y^{s,k+1}$ to the agents that require them to update their dual iterate.

For $(i,j) \in J$ and $l = 1,\dots,N$, the agents do in parallel

(2) Given δ^{k+1}, $P^{l,k+1}$, and $Y^{s,k+1}$ compute

$$\nabla_{\Lambda_{ij}} f(\Lambda^k, M^k) = E_{ij} \bullet \sum_{s\in\Gamma(i,j)} Y^{s,k+1} - E_{ij} \bullet \left(F^0 \delta^{k+1} + \sum_{l=1}^{N} \sum_{(i,j)\in\mathcal{I}_l} F_{ij}^l P_{i_l j_l}^{l,k+1} \right),$$

$$\nabla_{M^l} f(\Lambda^k, M^k) = \delta^{k+1} I_{n_l} - P^{l,k+1}.$$

(3) Find

$$Y_{ij}^k = \arg\max_{Y_{ij}\in\mathbb{R}} \left\{ -\frac{L}{2}\left(Y_{ij} - \Lambda_{ij}^k\right)^2 + \nabla_{\Lambda_{ij}} f(\Lambda^k, M^k) Y_{ij} \right\},$$

$$H^{l,k} = \arg\max_{H^l\in\mathcal{S}_+^{n_l}} \left\{ -\frac{L}{2}\|H^l - M^{l,k}\|_F^2 + \nabla_{M^l} f(\Lambda^k, M^k) \bullet H^l \right\}.$$

(4) Find

$$Z_{ij}^k = \arg\max_{Z_{ij}\in\mathbb{R}} \left\{ -\frac{L}{2} Z_{ij}^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{\Lambda_{ij}} f(\Lambda^j, M^j) Z_{ij} \right\},$$

$$T^{l,k} = \arg\max_{T^l\in\mathcal{S}_+^{n_l}} \left\{ -\frac{L}{2}\|T^l\|_F^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{M^l} f(\Lambda^j, M^j) \bullet T_l \right\}.$$

(5) Set

$$\Lambda_{ij}^{k+1} = \frac{k+1}{k+3} Y_{ij}^k + \frac{2}{k+3} Z_{ij}^k, \tag{22}$$

$$M^{l,k+1} = \frac{k+1}{k+3} H^{l,k} + \frac{2}{k+3} T^{l,k}. \tag{23}$$

(6) Send Λ_{ij}^k and $M^{l,k}$ to the neighboring agents.

While some of the subproblems in Algorithm 2 may look challenging at first glance, there are actually small-scale closed form solutions for all of them. For instance, the solution for $Y^{s,k+1}$ in step 1 is determined as follows [Deroo et al. 2014]: Consider the spectral decomposition of the symmetric matrix X_Y^s given by

$$X_Y^s = Q\Sigma Q^T = (Q_+, Q_-) \begin{pmatrix} \Sigma_+ & 0 \\ 0 & \Sigma_- \end{pmatrix} \begin{pmatrix} Q_+^T \\ Q_-^T \end{pmatrix},$$

where Σ_+ contains the non-negative eigenvalues of X_Y^s. Then the optimal solution $Y^{s,k+1}$ is the projection on the positive semidefinite part of X_Y^s, i.e.

$$Y^{s,k+1} = \frac{Q_+ \Sigma_+ Q_+^T}{\sigma_{Y^s}}. \tag{24}$$

Identically, we get solutions for $H^{l,k}$ and $T^{l,k}$.

Note that Algorithm 2 only leads to an approximate solution because it is based on dual decomposition. However, the convergence of the above Algorithm follows with Theorem 3 in [Deroo et al. 2014] where it is shown that the number of necessary iterations to achieve a desired accuracy ε of the approximate solution of problem (21) can be computed in advance.

Remark 3. Note that several adaptations to the algorithm are possible. One is to implement the algorithm using event-based communication [Meinel et al. 2014].

Remark 4. Note that all steps in Algorithm 2 require only neighborhood information except the computation of δ. However, the variable δ can be computed distributedly through a consensus if the size of the system is known.

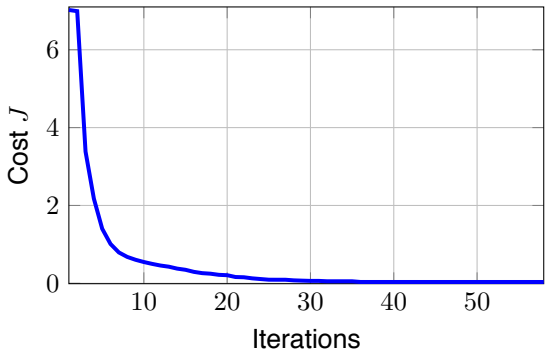
4. NUMERICAL INVESTIGATIONS

In this section, we demonstrate numerically that the feedback law obtained with Algorithm 1 where $S = \frac{P}{\gamma\delta}$ is obtained through (11) is stabilizing. For this purpose, we randomly create 100 unstable test systems, each with 20 subsystems, and every subsystem has two states. The systems are created in such a way such that the subsystems can stabilize the system with decentralized LQR control laws designed with only their own decoupled model ($Q = I_{n_i\times n_i}, R = 100 I_{m_i\times m_i}$). Afterwards, we determine solutions to (11) both with Yalmip [Löfberg 2004] (S_Y) and with Algorithm 2 (S_{dist}). Then, we determine distributed control laws K_{dist} with Algorithm 1 for four scenarios: (1) $t_f = 1$ and $S = S_Y$, (2) $t_f = 1$ and $S = S_{\text{dist}}$, (3) $t_f = 1$ and $S = 0_{n\times n}$, (4) $t_f = 10$ and $S = 0_{n\times n}$. The other weighting matrices are $Q = I_{n\times n}$ and $R = I_{m\times m}$. The first two scenarios are chosen to compare the results of the distributed algorithm 2 with the results from Yalmip. The third one is used to see if the additional terminal cost term even has an effect, while (4) investigates the influence of a longer optimization horizon on stability.

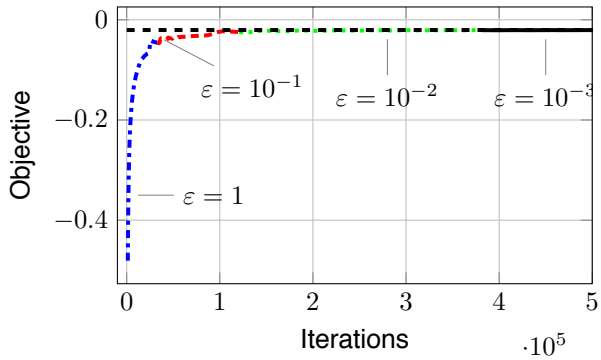
The results are summarized as follows: For scenarios (1), (2) and (4) all 100 systems are stabilized with the distributed control law. For scenario (3) only 46 systems, so less than half, are stabilized without the terminal cost term. This demonstrates that the terminal cost term leads to a stabilizing control law. Note that the terminal cost term leads to an increase of iterations in Algorithm 1. The results are summarized in Table 1. The principle behavior of Algorithm 1 is shown in Figure 1a where we plot the average of the achieved cost over the iterations.

We also see that if the horizon t_f is sufficiently long and no S is used, we can achieve stability as well. However, the comparison of scenarios (4) and (3) – in both cases $S = 0$ – shows that it is difficult and in no way obvious without global knowledge which horizon is "long enough" to guarantee stability. The tuning of the horizon to achieve stability without giving guarantees can then turn into a "playing of games" [Bitmead et al. 1990] and makes clear why the presented approach involving the terminal cost term is preferable.

Additionally as a comparison of the results of Yalmip with the distributed Algorithm 2 we compare the costs of the resulting control laws obtained with S_Y and S_{dist} and the maximum relative difference is less than 0.25%. This indicates that the distributed algorithm gives results that are very close to the centralized results even though only neighborhood information is used, and demonstrates the applicability of the distributed approach. To illustrate the behavior Algorithm 2, the interesting end part of a typical cost evolution for different values of the accuracy parameter ε is shown in Figure 1b.



(a) Average cost evolution for distributed gradient descent in Algorithm 1 for 100 systems



(b) Cost evolution of Algorithm 2 for different values of accuracy parameter ε. Optimal value in black (dashed) is -0.0204

Fig. 1. Illustration of behaviors of Algorithms 1 and 2.

5. CONCLUSION

In this paper we present a distributed control design method with guaranteed stability that only requires information exchange with neighboring subsystems. Stability is achieved by introducing a terminal penalty term into the finite-horizon LQ cost functional. A method is shown how to determine the penalty term in a distributed fashion using distributed optimization methods. Subsequently, the feedback matrix is computed using an iterative gradient-descent method. Because the only global information that has to be known is the system size, privacy of the subsystems is maintained in the sense that subsystems need to share their dynamic model only with a limited number of agents. The effectiveness of the approach is validated through numerical investigations.

Table 1. Comparison of the approach involving the new terminal cost term and without the new terminal cost term

	with S	$S = 0_{n\times n}$
# stabilized systems out of 100	100	46
Average # iterations for Algorithm 1	15.5	9.8

REFERENCES

Alam, A.A., Gattami, A., and Johansson, K.H. (2011). Suboptimal decentralized controller design for chain structures: Applications to vehicle formations. In *Proc. 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 6894–6900. IEEE.

Bitmead, R.R., Gevers, M., and Wertz, V. (1990). Adaptive Optimal Control: The Thinking Man's GPC.

Chen, H. and Allgöwer, F. (1998). A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10), 1205–1217.

Deroo, F., Meinel, M., Ulbrich, M., and Hirche, S. (2014). Distributed stability tests for large-scale systems with limited model information. *IEEE Transactions on Control of Network Systems (Submitted)*. URL `www-m1.ma.tum.de/foswiki/pub/M1/Lehrstuhl/MartinMeinel/DMUH_stability_final.pdf`.

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*.

Deroo, F., Ulbrich, M., Hirche, S., and Anderson, B.D.O. (2013). Distributed controller design for a class of sparse singular systems with privacy constraints. In *4th IFAC Workshop on Dist. Estimation and Cont. in Netw. Sys. (NecSys)*.

Farokhi, F., Langbort, C., and Johansson, K.H. (2012). Optimal structured static state-feedback control design with limited model information for fully-actuated systems. *Automatica*, 49(2), 326 – 337.

Gross, J.L. and Yellen, J. (2004). *Handbook of Graph Theory*. CRC press.

Khalil, H.K. (2002). *Nonlinear Systems*, volume 3. Prentice hall.

Kim, S., Kojima, M., Mevissen, M., and Yamashita, M. (2011). Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. *Mathematical Programming*, 129(1), 33–68.

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.

Löfberg, J. (2004). Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*. URL `http://users.isy.liu.se/johanl/yalmip`.

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

Martensson, K. and Rantzer, A. (2012b). Synthesis of structured controllers for large-scale systems. *IEEE Transactions on Automatic Control (Submitted)*.

Meinel, M., Ulbrich, M., and Albrecht, S. (2014). A class of distributed optimization methods with event-triggered communication. *Computational Optimization and Applications*, 57(3), 517–553.

Necoara, I., Nedelcu, V., and Dumitrache, I. (2011). Parallel and distributed optimization methods for estimation and control in networks. *Journal of Process Control*, 21(5), 756–766.

Necoara, I. and Suykens, J. (2008). Application of a smoothing technique to decomposition in convex optimization. *IEEE Transactions on Automatic Control*, 53(11), 2674–2679.

Rotkowitz, M. and Lall, S. (2006). A characterization of convex problems in decentralized Control. *IEEE Transactions on Automatic Control*, 51(2), 274–286.

Scattolini, R. (2009). Architectures for distributed and hierarchical model predictive control – a review. *Journal of Process Control*, 19(5), 723–731.