Survey

Gösta Stomberg, Alexander Engelmann and Timm Faulwasser*

A compendium of optimization algorithms for distributed linear-quadratic MPC

Ein Optimierungskompendium für die verteilte linear-quadratische modellprädiktive Regelung

https://doi.org/10.1515/auto-2021-0112
Received August 9, 2021; accepted January 4, 2022

Abstract: Model Predictive Control (MPC) for {networked, cyber-physical, multi-agent} systems requires numerical methods to solve optimal control problems while meeting communication and real-time requirements. This paper presents an introduction on six distributed optimization algorithms and compares their properties in the context of distributed MPC for linear systems with convex quadratic objectives and polytopic constraints. In particular, dual decomposition, the alternating direction method of multipliers, a distributed active set method, an essentially decentralized interior point method, and Jacobi iterations are discussed. Numerical examples illustrate the challenges, the prospect, and the limits of distributed MPC with inexact solutions.

Keywords: model predictive control, distributed optimization, ADMM, dual decomposition, active set methods, interior point methods, Jacobi iterations, conjugate gradients, fast gradients

Zusammenfassung: Die modellprädiktive Regelung vernetzter cyber-physischer Systeme erfordert numerische Verfahren, die Optimalsteuerungsprobleme in Echtzeit und unter Kommunikationseinschränkungen lösen. Dieser Beitrag vergleicht sechs Algorithmen für die verteilte prädiktive Regelung linearer Systeme mit konvexquadratischer Zielfunktion und affinen Restriktionen hinsichtlich ihrer Konvergenzeigenschaften und ihrer Kommunikationsanforderungen. Es werden zwei Varianten der dualen Dekomposition, die Alternating Direction Method of Multipliers, eine verteilte Aktive-Mengen-Strategie, ein verteiltes Innere-Punkte-Verfahren und das Jakobi-Verfahren behandelt. Numerische Beispiele illustrieren das Potential, aber auch die Herausforderungen und Grenzen von verteilter prädiktiver Regelung mittels inexakter Lösung der Optimalsteuerungsprobleme.

Schlagwörter: Modellprädiktive Regelung, verteilte Optimalsteuerung, dezentrale Optimierung

1 Introduction

The formulation and the efficient numerical solution of Optimal Control Problems (OCP) is key for a variety of tasks in systems and control. In particular, research on control of {networked, cyber-physical, multi-agent} systems has led to manifold results on Distributed MPC (DMPC). One prominent approach to DMPC is to solve the OCPs in the MPC loop via *iterative* schemes in which agents communicate multiple times per MPC step [27, 32]. Further DMPC approaches (though not covered here) include sequential DMPC schemes and schemes where the physical coupling between agents is viewed as disturbance [27]. In general, different aspects of iterative DMPC are considered in the literature:

(i) The formulation of separable OCPs with local terminal constraints and costs [8, 10, 25].
(ii) The design of tailored numerical algorithms, specifically the Jacobi iterations [11, 36, 39], Dual Decomposition (DD) [9, 19], the Alternating Direction Method of Multipliers (ADMM) [3, 6, 9, 31], and a recently proposed distributed Active Set Method (ASM) [37].
(iii) The stability analysis of the closed loop under inexact optimization [3, 18, 25].

The third item is important since distributed optimization algorithms often exhibit slow asymptotic convergence and hence solving the OCPs to full accuracy might be pro-

***Corresponding author: Timm Faulwasser,** Institute for Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany, e-mail: timm.faulwasser@ieee.org
Gösta Stomberg, Alexander Engelmann, Institute for Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany, e-mails: goesta.stomberg@tu-dortmund.de, alexander.engelmann@tu-dortmund.de

hibitive under strict communication requirements. Closed-loop stability under inexact optimization is analysed in [18, 25] for DD and in [3] for ADMM. The recent overview [5] discusses ADMM and a basic variant of DD for DMPC without numerical case studies.

In the present paper, we focus exclusively on deterministic optimization methods, i. e., we do not touch upon any heuristic schemes. Moreover, we assume perfect communication and we refer to [41] for an analysis including communication imperfection. Our compendium covers the Jacobi iterations, DD, a combination of DD with a Fast Gradient Method (DD-FGM), ADMM, ASM, and a recently proposed essentially decentralized Interior Point method (d-IP) [16]. Moreover, we illustrate the performance of the above algorithms on a numerical case study from formation control and on an adversarial example highlighting challenges in DMPC when using inexact solutions.

The paper is structured as follows: Section 2 provides the problem formulation as a partially separable quadratic program, Section 3 recalls optimization methods applicable to DMPC, Section 4 compares properties of the methods, and Section 5 applies the methods on two numerical examples.

Notation: The set $\mathbb{I}_{[0,N]}$ denotes the integers from 0 to N. I_n is the identity matrix in $\mathbb{R}^{n\times n}$ and $\otimes$ is the Kronecker product. The largest singular value of a matrix is written as $\sigma_{\max}$. Given a matrix A, $[A]_p$ denotes the pth row of A. Likewise, $[a]_p$ is the pth component of vector a. The concatenation of vectors into a column vector is $\mathrm{col}(\cdot)$. $A = [A_{ij}]$ is the block matrix with entries A_{ij} at block position (i,j).

2 Problem statement

Consider the discrete-time OCP

$$\min_{\boldsymbol{x},\boldsymbol{u}} \frac{1}{2}\sum_{i\in\mathcal{S}} J_i(\boldsymbol{x},\boldsymbol{u}) \tag{1a}$$

$$\text{s. t. for all } i\in\mathcal{S}:$$

$$x_i^{k+1} = \sum_{j\in\mathcal{S}} A_{ij}x_j^k + \sum_{j\in\mathcal{S}} B_{ij}u_j^k, \quad \forall k\in\mathbb{I}_{[0,N-1]} \tag{1b}$$

$$x_i^0 = x_{i,0} \tag{1c}$$

$$(x_i^k, u_i^k)\in\mathbb{X}_i\times\mathbb{U}_i, \quad \forall k\in\mathbb{I}_{[0,N-1]} \tag{1d}$$

$$x_i^N\in\mathbb{X}_i^{\mathrm{f}}, \tag{1e}$$

where

$$J_i(\boldsymbol{x},\boldsymbol{u}) = \sum_{j\in\mathcal{S}}\left(x_i^{N\top}P_{ij}x_j^N + \sum_{k=0}^{N-1} x_i^{k\top}Q_{ij}x_j^k + u_i^{k\top}R_{ij}u_j^k\right).$$

Here, $\mathcal{S} = \{1,\dots,|\mathcal{S}|\}$ is a set of linear time-invariant subsystems. The variables $x_i\in\mathbb{X}_i\subseteq\mathbb{R}^{n_i}$, $u_i\in\mathbb{U}_i\subseteq\mathbb{R}^{m_i}$ denote the states and inputs of subsystem $i\in\mathcal{S}$ with corresponding constraints and terminal constraints $\mathbb{X}_i^{\mathrm{f}}$. Moreover, $\boldsymbol{x}_i \doteq \mathrm{col}_{k\in\mathbb{I}_{[0,N]}}(x_i^k)$, $\boldsymbol{u}_i \doteq \mathrm{col}_{k\in\mathbb{I}_{[0,N-1]}}(u_i^k)$ are shorthand for the predicted trajectories over the prediction horizon N. The matrices $A_{ij}\in\mathbb{R}^{n_i\times n_j}$ and $B_{ij}\in\mathbb{R}^{n_i\times m_j}$ describe the physical coupling between subsystems $(i,j)\in\mathcal{S}\times\mathcal{S}$ and the matrices $Q_{ij}\in\mathbb{R}^{n_i\times n_j}$, $R_{ij}\in\mathbb{R}^{n_i\times m_j}$, and $P_{ij}\in\mathbb{R}^{n_i\times n_j}$ describe cost coupling.

The overall dynamics are

$$x^{k+1} = Ax^k + Bu^k, \quad x^0 = \mathrm{col}_{i\in\mathcal{S}}(x_{i,0}), \tag{2}$$

where $x\doteq\mathrm{col}_{i\in\mathcal{S}}(x_i)\in\mathbb{R}^n$, $u\doteq\mathrm{col}_{i\in\mathcal{S}}(u_i)\in\mathbb{R}^m$, $A=[A_{ij}]\in\mathbb{R}^{n\times n}$, and $B=[B_{ij}]\in\mathbb{R}^{m\times m}$.

The DMPC schemes discussed subsequently execute three steps per control interval: (i) Each subsystem measures or estimates its current state $x_{i,0}$; (ii) the subsystems solve OCP (1) cooperatively to obtain u_i^0; and (iii), each subsystem applies u_i^0.

Closed-loop stability can be guaranteed by the design of $\mathbb{X}_i^{\mathrm{f}}$ and P_{ij}. Most approaches in the literature require the following assumptions to hold.

Assumption 1 (Requirements for OCP (1)).

(a) The overall system (A,B) from (2) is stabilizable.
(b) The matrices $Q=[Q_{ij}]\in\mathbb{R}^{n\times n}$, $P=[P_{ij}]\in\mathbb{R}^{n\times n}$ and $R=[R_{ij}]\in\mathbb{R}^{m\times m}$ are positive definite.
(c) The constraint sets $\mathbb{X}_i$ and $\mathbb{U}_i$ are convex and closed polytopes that contain the origin in their interior.
(d) The sets $\mathbb{X}_i^{\mathrm{f}}$ are closed and convex polytopes.
(e) For all $x^k\in\mathbb{X}^{\mathrm{f}}$, there exists $u^k\in\mathbb{U}$, such that $x^{k+1}\in\mathbb{X}^{\mathrm{f}}$ and

$$V^{\mathrm{f}}(x^{k+1}) - V^{\mathrm{f}}(x^k) \le -\frac{1}{2}x^{k\top}Qx^k - \frac{1}{2}u^{k\top}Ru^k,$$

where $V^{\mathrm{f}}(x) = \frac{1}{2}x^\top Px$.
(f) OCP (1) is feasible at $t=0$.

The numerical methods of Section 3 require that each subsystem communicates bidirectionally with all neighboring—i. e., interacting—subsystems. Hence, the neighbors of subsystem i are given by $\mathcal{N}_i \doteq \mathcal{N}_i^{\mathrm{in}}\cup\mathcal{N}_i^{\mathrm{out}}$, where

$$\mathcal{N}_i^{\mathrm{in}} \doteq \left\{j\in\mathcal{S} \mid \bigcup \mathcal{X}_{ij}\neq 0, \mathcal{X}\in\{A,B,Q,R,P\}\right\}$$

$$\mathcal{N}_i^{\mathrm{out}} \doteq \left\{j\in\mathcal{S} \mid \bigcup \mathcal{X}_{ji}\neq 0, \mathcal{X}\in\{A,B,Q,R,P\}\right\}.$$

Formulation of (1) as a partially separable problem

To apply distributed optimization methods, we reformulate OCP (1) as partially separable Quadratic Program (QP).

To this end, we introduce trajectory copies

$$v_{ji}^k = \mathrm{col}(x_j^k, u_j^k), \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{N}_i^{\mathrm{in}}, \ \forall k \in \mathbb{I}_{[0,N-1]}. \tag{3}$$

This allows to write (1b) as

$$x_i^{k+1} = A_{ii}x_i^k + B_{ii}u_i^k + \sum_{j\in\mathcal{N}_i^{\mathrm{in}}} [A_{ij}, B_{ij}]v_{ji}^k \quad \forall i \in \mathcal{S}.$$

The copies $v_i^k \doteq \mathrm{col}_{j\in\mathcal{N}_i^{\mathrm{in}}}(v_{ji}^k)$ are additional decision variables of subsystem i. We write OCP (1) as

$$\min_{z_i, i\in\mathcal{S}} \quad \sum_{i\in\mathcal{S}} \frac{1}{2} z_i^\top H_i z_i + g_i^\top z_i \tag{4a}$$

$$\text{s.t.} \quad C_i^{\mathcal{E}} z_i = b_i^{\mathcal{E}} \quad \forall i \in \mathcal{S}, \tag{4b}$$

$$C_i^{\mathcal{I}} z_i \le b_i^{\mathcal{I}} \quad \forall i \in \mathcal{S}, \tag{4c}$$

$$\sum_{i\in\mathcal{S}} E_i z_i = b, \tag{4d}$$

where $z_i \doteq \mathrm{col}(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{v}_i) \in \mathbb{R}^{n_{z,i}}$, $\boldsymbol{v}_i = \mathrm{col}_{j\in\mathcal{N}_i^{\mathrm{in}}}(\boldsymbol{v}_{ji})$, and $\boldsymbol{v}_{ji} \doteq \mathrm{col}_{k\in\mathbb{I}_{[0,N]}}(v_{ji}^k)$. The matrices H_i are composed of the weight matrices Q_{ij}, R_{ij}, and P_{ij}. The equality constraints (4b) with $C_i^{\mathcal{E}} \in \mathbb{R}^{n_{g,i}\times n_{z,i}}$ include the dynamics (1b). The inequalities (4c) with $C_i^{\mathcal{I}} \in \mathbb{R}^{n_{h,i}\times n_{z,i}}$ describe the state and input constraints (1d). Depending on the type of terminal constraint, (1e) is part of (4b) or (4c). The parameters g_i and b are zero for OCP (1). The constraints (4d) couple original and copied variables via (3). Due to the coupling of original and copied variables, the matrices $E_i \in \mathbb{R}^{n_c\times n_{z,i}}$, where n_c denotes the number of coupling constraints, exhibit a sparse structure. This sparsity is crucial for solving the OCP in distributed fashion and it is exploited by all optimization methods covered in this article. We remark that, in principle, OCP (4) also allows for coupled state and input constraints in addition to the coupled dynamics and objectives.

For the sake of brevity, we rewrite QP (4) as

$$\min_{z_i\in\mathbb{Z}_i, i\in\mathcal{S}} \quad \sum_{i\in\mathcal{S}} \phi_i(z_i) \tag{5a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{S}} E_i z_i = b \quad | \ \lambda, \tag{5b}$$

where $\mathbb{Z}_i$ denotes the feasible set of subsystem i

$$\mathbb{Z}_i \doteq \{z_i \in \mathbb{R}^{n_{z,i}} | C_i^{\mathcal{E}} z_i = b_i^{\mathcal{E}}, \ C_i^{\mathcal{I}} z_i \le b_i^{\mathcal{I}}\},$$

$\phi_i(z_i) = \frac{1}{2} z_i^\top H_i z_i + g_i^\top z_i$ is the local objective. The notation in (5b) highlights that λ is the Lagrange multiplier to (5b).

Example 1 (Problem formulation). To illustrate the decoupling of dynamics via trajectory copies, consider the three subsystems

$$\begin{aligned} x_1^{k+1} &= x_1^k + u_1^k, \quad x_1^0 = x_{1,0}, \\ x_2^{k+1} &= x_1^k + x_2^k, \quad x_2^0 = x_{2,0}, \\ x_3^{k+1} &= x_2^k + x_3^k, \quad x_3^0 = x_{3,0}. \end{aligned}$$

Defining the copies $v_2 \doteq x_1$ and $v_3 \doteq x_2$ yields

$$\begin{aligned} x_1^{k+1} &= x_1^k + u_1^k, \quad x_1^0 = x_{1,0}, \\ x_2^{k+1} &= v_2^k + x_2^k, \quad x_2^0 = x_{2,0}, \\ x_3^{k+1} &= v_3^k + x_3^k, \quad x_3^0 = x_{3,0}. \end{aligned}$$

For $N = 1$, the decision variables in (4) are $z_1^\top = [x_1^0 \ \ x_1^1 \ \ u_1^0]$, $z_2^\top = [x_2^0 \ \ x_2^1 \ \ v_2^0]$, and $z_3^\top = [x_3^0 \ \ x_3^1 \ \ v_3^0]$. The coupling is given by

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} z_1 + \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} z_2 + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} z_3 = 0.$$

Remark 1 (Decentralized and distributed optimization). Throughout the paper we distinguish distributed, decentralized, and essentially decentralized algorithms. We call a method decentralized, if the subsystems communicate only between neighbors. We refer to a method as distributed, if in addition to neighbor-to-neighbor communication it requires a limited amount of centralized coordination, computation, and communication, e. g., to solve a centralized coordination optimization problem. We denote a method as essentially decentralized, if there is no central computation but limited global communication—i. e., communication between all agents—, which is small compared to the communication between neighbors.

Remark 2 (Stability under inexact optimization). To limit communication and computation, optimization algorithms are often terminated early leading to a suboptimal control input. Closed-loop stability for suboptimal control inputs can be guaranteed via appropriate terminal ingredients P_i and $\mathbb{X}_i^{\mathrm{f}}$, i. e., feasibility of the overall system implies stability [33]. For instance, one can choose a zero terminal constraint $\mathbb{X}_i^{\mathrm{f}} = \{0\}$ or design local terminal sets $\mathbb{X}_i^{\mathrm{f}}$ [8]. Feasibility of suboptimal solutions can be due to the optimization method (e. g., feasibility of all iterates) or it can be enforced via tightened constraint sets. Dual optimization methods such as DD and ADMM, however, achieve primal feasibility only asymptotically. For these methods stability under inexact optimization can be guaranteed by appropriate stopping criteria [3, 18, 25].

In case one resorts to the straight-forward choice $\mathbb{X}_i^{\mathrm{f}} = \{0\}$ (e. g., in Section 5) it is necessary to require that the aggregated system (2), i. e., the pair (A, B), is controllable.

3 Algorithms for DMPC

3.1 Dual decomposition

The Lagrangian of (5) is defined as

$$L(z,\lambda) = \left(\sum_{i\in\mathcal{S}} \phi_i(z_i) + \lambda^\top E_i z_i\right) - \lambda^\top b.$$

The dual problem to (5) is then given by

$$\max_{\lambda} q(\lambda), \tag{6}$$

where $q(\lambda) = \min_{z_i\in\mathbb{Z}_i} L(z,\lambda)$ is called the dual function. Dual decomposition applies gradient-based methods to (6). By Assumption 1, the objectives $\phi_i(z_i)$ are strongly convex with convexity parameter μ, and the local constraint sets $\mathbb{Z}_i$ are closed and convex. This ensures continuous differentiability of $q(\lambda)$ and hence

$$\nabla q(\lambda) = \left(\sum_{i\in\mathcal{S}} E_i z_i^\star\right) - b,$$

where $z_i^\star = \operatorname{argmin}_{z_i\in\mathbb{Z}_i} \phi_i(z_i) + \lambda^\top E_i z_i$, see [1, Prop. 6.1.1]. Applying the gradient method with step size c to (6) yields

$$\lambda^{l+1} = \lambda^l + c\nabla q(\lambda^l).$$

Convergence to the optimum is then guaranteed for any step size $c \in (0, 2\mu/\|E\|_2^2)$, where

$$E = [E_1, \cdots, E_{|\mathcal{S}|}],$$

see [1, Prop. 1.2.3]. The iterations read

$$z_i^l = \operatorname*{argmin}_{z_i\in\mathbb{Z}_i} \phi_i(z_i) + \lambda^{l\top} E_i z_i \tag{7a}$$

$$\lambda^{l+1} = \lambda^l + c\left(\sum_{i\in\mathcal{S}} E_i z_i^l - b\right). \tag{7b}$$

DD appears to be a distributed method due to update (7b). However, one can easily obtain a decentralized version as follows. The iterations (7) imply

$$z_i^{l+1} = \operatorname*{argmin}_{z_i\in\mathbb{Z}_i} \phi_i(z_i) + \left(\lambda^{l\top} + c\sum_{j\in\mathcal{S}} z_j^{l\top} E_j^\top - b^\top\right) E_i z_i.$$

If $j \notin \mathcal{N}_i \cup \{i\}$, then $E_j^\top E_i = 0$ and hence

$$z_i^{l+1} = \operatorname*{argmin}_{z_i\in\mathbb{Z}_i} \phi_i(z_i) + \left(\lambda^{l\top} + c\sum_{j\in\mathcal{N}_i\cup\{i\}} z_j^{l\top} E_j^\top - b^\top\right) E_i z_i.$$

Next, we introduce a local variable $\lambda_i \in \mathbb{R}^{n_c}$ for each subsystem such that

$$\lambda^\top E_i = \lambda_i^\top E_i, \quad \forall i \in \mathcal{S}. \tag{8}$$

Algorithm 1: Decentralized DD [5].

Initialization: λ^0, $\{\lambda_i^0\}_{i\in\mathcal{S}}$ satisfying (8), $l_{\max}$
while $l < l_{\max}$ **do**
  $z_i^l = \operatorname*{argmin}_{z_i\in\mathbb{Z}_i} \phi_i(z_i) + \lambda_i^{l\top} E_i z_i$
  Receive $[z_j^l]_p$ satisfying (9) $\forall j \in \mathcal{N}_i$.
  $\lambda_i^{l+1} = \lambda_i^l + c\left(\sum_{j\in\mathcal{N}_i\cup\{i\}} E_j z_j^l - b\right)$.

If $j \in \mathcal{N}_i \cup \{i\}$, then $E_j^\top E_i$ is sparse and only the components

$$[z_j^l]_p \; \forall p \in \{\mathbb{I}_{[0,n_{z,j}]} \mid [E_j^\top E_i]_p \neq 0\} \tag{9}$$

of subsystem j are required to compute λ_i^{l+1}. With this we finally obtain DD in decentralized form as summarized in Algorithm 1. If (5) is obtained by copying variables as described in Section 2, then (8) implies

$$[\lambda_i]_p = [\lambda]_p \quad \forall p \in \{\mathbb{I}_{[0,n_c]} \mid [E_i]_p \neq 0\}, \quad \forall i \in \mathcal{S}.$$

3.2 Dual decomposition with fast gradients

Instead of the gradient method, we can also apply a fast gradient method to solve the dual problem (6). This gives

$$\begin{aligned}
\lambda^{l+1} &= \bar{\lambda} + \frac{1}{L}\nabla q(\bar{\lambda}) \\
\alpha^{l+1} &= \frac{\alpha^l}{2}\left(\sqrt{(\alpha^l)^2 + 4} - \alpha^l\right) \\
\beta^l &= \frac{\alpha^l(1-\alpha^l)}{(\alpha^l)^2 + \alpha^{l+1}} \\
\bar{\lambda}^{l+1} &= \lambda^{l+1} + \beta^l(\lambda^{l+1} - \lambda^l),
\end{aligned}$$

where L is a Lipschitz constant of $\nabla q(\lambda)$ [9, 30] and β is the step size. To obtain fast convergence, we choose the smallest Lipschitz constant $L^\star$ of $\nabla q(\lambda)$, i. e.,

$$L^\star = \sigma_{\max}(EH^{-1/2})^2,$$

where $H = \operatorname{blkdiag}(H_1, \cdots, H_{|\mathcal{S}|})$, see [30]. Again, we introduce local versions $\lambda_i \in \mathbb{R}^{n_c}$ and use the sparsity of E_i to arrive at the decentralized version in Algorithm 2.

3.3 Alternating direction method of multipliers

Distributed ADMM requires a different problem formulation for which we introduce the auxiliary decision variable

Algorithm 2: DD with fast gradients [9].

Initialization: λ^0, $\{\lambda_i^0\}_{i\in\mathcal{S}}$ satisfying (8), $\alpha^0 = \frac{\sqrt{5}-1}{2}$, $\bar{\lambda}_i^0 = \lambda_i^0\ \forall i \in \mathcal{S}$, $l_{\max}$
while $l < l_{\max}$ **do**
  $z_i^l = \underset{z_i\in\mathbb{Z}_i}{\operatorname{argmin}}\ \phi_i(z_i) + \bar{\lambda}_i^{l\top} E_i z_i$
  Receive $[z_j^l]_p$ satisfying (9) $\forall j \in \mathcal{N}_i$.
  $\lambda_i^{l+1} = \bar{\lambda}_i^l + \frac{1}{L}\Big(\sum_{j\in\mathcal{N}_i\cup\{i\}} E_j z_j - b\Big)$
  $\alpha^{l+1} = \frac{\alpha^l}{2}(\sqrt{(\alpha^l)^2+4} - \alpha^l)$
  $\beta^l = \alpha^l(1-\alpha^l)/((\alpha^l)^2 + \alpha^{l+1})$
  $\bar{\lambda}_i^{l+1} = \lambda_i^{l+1} + \beta^l(\lambda_i^{l+1} - \lambda_i^l)$.

Algorithm 3: Decentralized ADMM [31].

Initialization: v^0, $\{v_i^0\}_{i\in\mathcal{S}}$ satisfying (12), $\{\bar{z}_i^0\}_{i\in\mathcal{S}}$, $l_{\max}$
while $l < l_{\max}$ **do**
  $z_i^{l+1} = \underset{z_i\in\mathbb{Z}_i}{\operatorname{argmin}}\ L_{\rho,i}(z_i, \bar{z}_i^l, v_i^l)$
  Receive copies $\boldsymbol{v}_{ji}^{l+1}$ from $j \in \mathcal{N}_i^{\text{out}}$.
  $\operatorname{col}(\bar{\boldsymbol{x}}_i^{l+1}, \bar{\boldsymbol{u}}_i^{l+1}) = \frac{1}{|\mathcal{N}_i^{\text{out}}|+1}(\operatorname{col}(\boldsymbol{x}_i^{l+1}, \boldsymbol{u}_i^{l+1}) + \sum_{j\in\mathcal{N}_i^{\text{out}}} \boldsymbol{v}_{ji}^{l+1})$
  Send $\operatorname{col}(\bar{\boldsymbol{x}}_i^{l+1}, \bar{\boldsymbol{u}}_i^{l+1})$ to $j \in \mathcal{N}_i^{\text{out}}$.
  $\bar{\boldsymbol{v}}_{ji}^{l+1} = \operatorname{col}(\bar{\boldsymbol{x}}_j^{l+1}, \bar{\boldsymbol{u}}_j^{l+1})$ for all $j \in \mathcal{N}_i^{\text{in}}$
  $\bar{\boldsymbol{v}}_i^{l+1} = \operatorname{col}_{j\in\mathcal{N}_i^{\text{in}}}(\bar{\boldsymbol{v}}_{ji}^{l+1})$
  $\bar{z}_i^{l+1} = \operatorname{col}(\bar{\boldsymbol{x}}_i^{l+1}, \bar{\boldsymbol{u}}_i^{l+1}, \bar{\boldsymbol{v}}_i^{l+1})$
  $v_i^{l+1} = v_i^l + \rho(z_i^{l+1} - \bar{z}_i^{l+1})$.

$\bar{z}_i \in \mathbb{R}^{n_{z,i}}$ for each subsystem. We assume that $b = 0$ and reformulate QP (5) as

$$\min_{z_i\in\mathbb{Z}_i, \bar{z}_i, i\in\mathcal{S}} \quad \sum_{i\in\mathcal{S}} \phi_i(z_i) \tag{10a}$$

$$\text{s.t.} \quad z_i - \bar{z}_i = 0, \quad \forall i \in \mathcal{S} \quad | \ v_i \tag{10b}$$

$$\sum_{i\in\mathcal{S}} E_i \bar{z}_i = 0, \tag{10c}$$

where $v_i \in \mathbb{R}^{n_{z,i}}$ are Lagrange multipliers associated to (10b). The augmented Lagrangian of (10) is given by

$$\begin{aligned} L_\rho(z, \bar{z}, v) &= \sum_{i\in\mathcal{S}} L_{\rho,i}(z_i, \bar{z}_i, v_i) \\ &= \sum_{i\in\mathcal{S}} \phi_i(z_i) + v_i^\top(z_i - \bar{z}_i) + \frac{\rho}{2}\|z_i - \bar{z}_i\|_2^2, \end{aligned}$$

where $\rho \in \mathbb{R}^+$ is called penalty parameter. Let $\bar{z} = \operatorname{col}_{i\in\mathcal{S}}(\bar{z}_i)$ and let $\mathbb{E} = \{\bar{z} \in \mathbb{R}^{n_z} \mid \sum_{i\in\mathcal{S}} E_i \bar{z}_i = 0\}$ denote the feasible set of the coupling constraints (10c). ADMM updates the decision variables z_i and $\bar{z}_i$ in alternating fashion with iterations

$$z_i^{l+1} = \underset{z_i\in\mathbb{Z}_i}{\operatorname{argmin}}\ L_{\rho,i}(z_i, \bar{z}_i^l, v_i^l), \tag{11a}$$

$$\bar{z}^{l+1} = \underset{\bar{z}\in\mathbb{E}}{\operatorname{argmin}} \sum_{i\in\mathcal{S}} L_{\rho,i}(z_i^{l+1}, \bar{z}_i, v_i^l), \tag{11b}$$

$$v_i^{l+1} = v_i^l + \rho(z_i^{l+1} - \bar{z}_i^{l+1}). \tag{11c}$$

At each iteration, the variables z_i satisfy the local constraints and the variables $\bar{z}_i$ satisfy the coupling constraints. Upon convergence, $z_i = \bar{z}_i$ such that all z_i are primal feasible, i. e., they satisfy (10b).

From the iterations (11) it is not yet clear why ADMM is a decentralized method as the update (11b) seemingly requires global coordination to satisfy the coupling constraints. However, for QP (10), the update can be replaced by an averaging step under suitable conditions on E_i [4, p. 55]. These conditions are intrinsically satisfied if (4) is constructed using the procedure via copied trajectories as described in Example 1. To decentralize ADMM, one solves the corresponding KKT system analytically $\bar{z}^{l+1} = M(z^{l+1} + v/\rho)$, where $M = (I_{n_z} - E^\top(EE^\top)^{-1}E)$, $z = \operatorname{col}_{i\in\mathcal{S}}(z_i)$, and $v = \operatorname{col}_{i\in\mathcal{S}}(v_i)$. Then, each row $[E]_k$ has precisely two non-zero elements: one element is +1 and corresponds to the original variable, and the other element is –1 and corresponds to the copy. Multiplication by M is an averaging step. If the Lagrange multipliers satisfy

$$Mv = 0, \tag{12}$$

then the update of element $[\bar{z}]_p$ is given by the average of the original and copied variables that represent the same physical variable as $[\bar{z}]_p$. Since v_i^{l+1} satisfies (12) for all $l \geq 1$ by (11) [4, p. 55], we only have to enforce (12) upon initialization if we wish to perform (11b) via the described averaging step. Furthermore, one may warm-start ADMM with the Lagrange multipliers obtained at the previous MPC step. Algorithm 3 summarizes the decentralized ADMM variant.

3.4 Distributed active set method

Next, we present a distributed active set method from [37]. In QP (4) the active set of agent i is

$$\mathcal{A}(z_i) \doteq \{j \in \{1, \dots, n_{h,i}\} \mid [C_i^{\mathcal{I}}]_j z_i = [b_i^{\mathcal{I}}]_j\}.$$

Starting from a feasible initialization z_i^0, the agents take steps $z_i^{l+1} = z_i^l + \alpha^l \Delta z_i^l$, where the step size α^l is the same for

Algorithm 4: Distributed ASM [37].

1 **Initialization:** $\mathcal{A}(z_i^0), z_i^0, \varepsilon$
2 Local condensing for all $i \in \mathcal{S}$: compute Z_i, Y_i, $w_i = (C_i^l Y_i)^{-1} d_i$, $\bar{H}_i = Z_i^\top H_i Z_i$, $\bar{f}_i = Z_i^\top f_i^l + Z_i^\top H_i Y_i w_i$, $\bar{E}_i = E_i Z_i$, $S_i = \bar{E}_i \bar{H}_i^{-1} \bar{E}_i^\top$, and $s_i = E_i Y_i w_i - \bar{E}_i \bar{H}_i^{-1} \bar{f}_i$.
3 Solve $\sum_{i\in\mathcal{S}} S_i \lambda = \sum_{i\in\mathcal{S}} s_i$ with d-CG.
4 Local direction for all $i \in \mathcal{S}$: $\Delta z_i^l = Z_i \bar{H}_i^{-1}(-\bar{f}_i - \bar{E}_i^\top \lambda) + w_i$.
5 **if** $\|\Delta z_i^l\| < \varepsilon \ \forall i \in \mathcal{S}$ **then**
6 $y_i^l = (C_i^l C_i^{l\top})^{-1} C_i^l (-f_i^l - E_i^\top \lambda) \ \forall i \in \mathcal{S}$.
7 **if** $[y_i^{\mathcal{I}}]_p \geq 0, \forall p \in \mathcal{A}(z_i^l), \forall i \in \mathcal{S}$ **then**
8 **return** $z_i^l \ \forall i \in \mathcal{S}$.
9 **else**
10 $[i^\star, p^\star] = \underset{i\in\mathcal{S},\, p\in\mathcal{A}(z_i^l)}{\operatorname{argmin}} [y_i^{\mathcal{I}}]_p$.
11 Remove constraint $p^\star$ from $\mathcal{A}(z_{i^\star}^l)$.
12 Update $C_{i^\star}^l$.
13 **go to** 2.
14 **else**
15 Local step size: α_i^l according to (14) $\forall i \in \mathcal{S}$.
16 Global step size: $\alpha^l = \min_{i\in\mathcal{S}} \alpha_i^l$, $i^\star = \operatorname{argmin}_{i\in\mathcal{S}} \alpha_i^l$.
17 Add blocking constraint to $\mathcal{A}(z_{i^\star}^l)$.
18 Update $C_{i^\star}^l$.
19 Local step: $z_i^{l+1} = z_i^l + \alpha^l \Delta z_i^l$ for all $i \in \mathcal{S}$.
20 **go to** 2.

Algorithm 5: d-CG [13].

1 **Initialization:** λ^0, $r^0 = p^0 = s - S\lambda^0$, ε
2 $r_i^0 = I_{\mathcal{C}(i)} r^0$, $p_i^0 = I_{\mathcal{C}(i)} p^0$
3 $\eta_i^0 = r_i^{0\top} \Lambda_i^{-1} r_i^0$
4 $\eta^0 = \sum_{i\in\mathcal{S}} \eta_i^0$
5 **while** $\|r_i^l\|_\infty > \varepsilon \quad \forall i \in \mathcal{S}$ **do**
6 $\sigma_i^l = p_i^{l\top} \hat{S}_i p_i^l$
7 $\sigma^l = \sum_{i\in\mathcal{S}} \sigma_i^l$
8 $\lambda_i^{l+1} = \lambda_i^l + \frac{\eta^l}{\sigma^l} p_i^l$
9 $r_i^{l+1} = r_i^l - \frac{\eta^l}{\sigma^l} \sum_{j\in\mathcal{N}_i\cup i} I_{ij} \hat{S}_j p_j^l$
10 $\eta_i^{l+1} = r_i^{l+1\top} \Lambda_i^{-1} r_i^{l+1}$
11 $\eta^{l+1} = \sum_{i\in\mathcal{S}} \eta_i^{l+1}$
12 $p_i^{l+1} = r_i^{l+1} + \frac{\eta^{l+1}}{\eta^l} p_i^l$.

all agents. The directions Δz_i^l are obtained solving

$$\min_{\Delta z_i, i\in\mathcal{S}} \quad \sum_{i\in\mathcal{S}} \frac{1}{2} \Delta z_i^\top H_i \Delta z_i + f_i^{l\top} \Delta z_i \tag{13a}$$

$$\text{s.t.} \quad C_i^l \Delta z_i = d_i, \quad \forall i \in \mathcal{S} \qquad | \ \gamma_i \tag{13b}$$

$$\sum_{i\in\mathcal{S}} E_i \Delta z_i = 0, \qquad | \ \lambda \tag{13c}$$

where $C_i^l \doteq \begin{bmatrix} C_i^{\mathcal{E}\top} & (\operatorname{col}_{j\in\mathcal{A}(z_i^l)}([C_i^{\mathcal{I}}]_j))^\top \end{bmatrix}^\top$, $f_i^l \doteq H_i z_i^l + g_i$, $d_i = 0$, and γ_i and λ are Lagrange multipliers.

Let Z_i be a matrix where the columns of Z_i form a basis of the nullspace of C_i^l and let Y_i be a matrix where the columns of Y_i form a basis of the range space of C_i^l. Algorithm 4 summarizes ASM.

Step 2 in Algorithm 4 first eliminates (13b), i. e., condenses (13), via the null space basis Z_i and then computes the Schur complement S_i and s_i. Step 3—inspired by the bi-level distributed algorithm from [15]—is the key idea of Algorithm 4: solve (13) with an essentially decentralized version of the Conjugate Gradient method (d-CG) [13]. The step directions Δz_i are obtained via back substitution in Step 4. In Step 7 each agent checks for dual feasibility, if the step directions satisfy $\|\Delta z_i^l\| < \varepsilon$ for a chosen threshold ε. The method stops if dual feasibility is achieved. Else, the active inequality constraint corresponding to the most negative Lagrange multiplier among all agents is removed from the active set (Step 11) and new step directions Δz_i are computed. If $\|\Delta z_i^l\| \geq \varepsilon$ for some agents, then in Step 15 every agent computes the largest step size $\alpha_i^l \in (0, 1]$ which retains primal feasibility [29]

$$\alpha_i^l \doteq \min \left\{ 1, \min_{j\notin\mathcal{A}(z_i^l), [c_i^{\mathcal{I}}]_j \Delta z_i^l < 0} \frac{[b_i^{\mathcal{I}}]_j - [c_i^{\mathcal{I}}]_j z_i^l}{[c_i^{\mathcal{I}}]_j \Delta z_i^l} \right\}. \tag{14}$$

If $\alpha^l < 1$, the respective blocking constraint is added to the blocking agents' active set in Step 17.

To complete the presentation of ASM, we recall d-CG as the key component in Step 3 of Algorithm 4.

Essentially decentralized conjugate gradients

Consider the system of linear equations $S\lambda = s$, where $S \doteq \sum_{i\in\mathcal{S}} S_i \in \mathbb{R}^{n_c \times n_c}$ is positive definite, $s \doteq \sum_{i\in\mathcal{S}} s_i$ and where the matrices S_i from Step 2 in Algorithm 4 inherit the sparse structure from the coupling matrices E_i. The essentially decentralized conjugate gradient method solves this system of equations iteratively in at most n_c steps [13]. To exploit the sparsity in S_i, define the set of constraints assigned to subsystem $i \in \mathcal{S}$ by $\mathcal{C}(i) = \{j \in \{1, \dots n_c\} \mid [E_i]_j \neq 0\}$. Moreover, introduce matrices that map from global to local constraints $I_{\mathcal{C}(i)} \doteq \operatorname{col}_{j\in\mathcal{C}(i)}(e_j^\top) \in \mathbb{R}^{n_{c,i} \times n_c}$, where $e_j \in \mathbb{R}^{n_c}$ is the jth unit vector. Algorithm 5 summarizes d-CG, with $\Lambda \doteq \sum_{i\in\mathcal{S}} I_{\mathcal{C}(i)}^\top I_{\mathcal{C}(i)}$, $\Lambda_i \doteq I_{\mathcal{C}(i)} \Lambda I_{\mathcal{C}(i)}^\top$, $\hat{S}_i \doteq I_{\mathcal{C}(i)} S_i I_{\mathcal{C}(i)}^\top$ and $I_{ij} \doteq I_{\mathcal{C}(i)} I_{\mathcal{C}(j)}^\top$. d-CG iteratevly updates the residual r_i, the

step direction p_i, and the stepsizes σ and η. d-CG requires two types of communication: Step 9 requires neighbor-to-neighbor communication of equal amount as DD and ADMM. Steps 3, 7, and 11 require the summation of the scalars σ_i and η_i over the entire network rendering d-CG an essentially decentralized algorithm. For a convergence proof and further details on d-CG, see [13].

3.5 Essentially decentralized interior point method

Next, we summarize the essentially decentralized interior point method (d-IP) from [16]. We reformulate the inequality constraints (4c) by a logarithmic barrier function, which yields

$$\min_{z_i, w_i, i\in\mathcal{S}} \sum_{i\in\mathcal{S}} \phi_i(z_i) - \mathbb{1}^\top \delta \ln(w_i) \tag{15a}$$

$$\text{s.t.} \quad C_i^{\mathcal{E}} z_i = b_i^{\mathcal{E}}, \quad \forall i \in \mathcal{S} \quad | \ \gamma_i \tag{15b}$$

$$C_i^{\mathcal{I}} z_i + w_i = b_i^{\mathcal{I}}, \quad \forall i \in \mathcal{S} \quad | \ \mu_i \tag{15c}$$

$$w_i \geq 0, \quad \forall i \in \mathcal{S} \tag{15d}$$

$$\sum_{i\in\mathcal{S}} E_i z_i = b \quad | \ \lambda \tag{15e}$$

with $\mathbb{1} = (1, \dots, 1)^\top \in \mathbb{R}^{n_{h,i}}$, slack variables $w_i \in \mathbb{R}^{n_{h,i}}$, and Lagrange multipliers γ_i, μ_i, and λ. Problem (15) is solved for a decreasing sequence of the barrier parameter $\delta > 0$. In the limit $\delta \to 0$, the problems (4) and (15) are equivalent. Here, we consider a specific variant of interior point methods that applies *one* Newton step to (15) and then updates δ [7].

A Newton step for (15) reads

$$\begin{bmatrix} \nabla F_1^\delta & 0 & \dots & \tilde{E}_1^\top \\ 0 & \nabla F_2^\delta & \dots & \tilde{E}_2^\top \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{E}_1 & \tilde{E}_2 & \dots & 0 \end{bmatrix} \begin{bmatrix} \Delta p_1 \\ \Delta p_2 \\ \vdots \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -F_1^\delta \\ -F_2^\delta \\ \vdots \\ b - \sum_{i\in\mathcal{S}} E_i z_i \end{bmatrix} \tag{16}$$

with $\Delta p_i = (\Delta z_i, \Delta w_i, \Delta\gamma_i, \Delta\mu_i)$. We have

$$F_i^\delta(p_i, \lambda) = \begin{bmatrix} H_i z_i + g_i + C_i^{\mathcal{E}\top}\gamma_i + C_i^{\mathcal{I}\top}\mu_i + E_i^\top\lambda \\ -\delta W_i^{-1}\mathbb{1} + \mu_i \\ C_i^{\mathcal{E}} z_i - b_i^{\mathcal{E}} \\ C_i^{\mathcal{I}} z_i + w_i - b_i^{\mathcal{I}} \end{bmatrix},$$

$W_i = \operatorname{diag}(w_i)$,

$$\nabla F_i^\delta = \begin{bmatrix} H_i & 0 & C_i^{\mathcal{E}\top} & C_i^{\mathcal{I}\top} \\ 0 & W_i^{-1}M_i & 0 & I \\ C_i^{\mathcal{E}} & 0 & 0 & 0 \\ C_i^{\mathcal{I}} & I & 0 & 0 \end{bmatrix},$$

$M_i = \operatorname{diag}(\mu_i)$, and $\tilde{E}_i = [E_i \quad 0 \quad 0 \quad 0]$. In case ∇F_i^δ is invertible, we obtain from (16)

$$\sum_{i\in\mathcal{S}} S_i \Delta\lambda = \sum_{i\in\mathcal{S}} s_i, \tag{17}$$

where

$$S_i \doteq \tilde{E}_i (\nabla F_i^\delta)^{-1} \tilde{E}_i^\top, \tag{18a}$$

$$s_i \doteq \tilde{E}_i (\nabla F_i^\delta)^{-1} F_i^\delta + E_i z_i - \frac{1}{|\mathcal{S}|} b. \tag{18b}$$

To obtain an essentially decentralized algorithm, we solve (17) with d-CG from Subsection 3.5 [13]. Similar to the ASM, d-IP can be understood as a bi-level distributed algorithm [15]. Notice that early termination of d-CG allows to reduce communication. Consider the residual $r_\lambda^l = S^l \Delta\lambda_\lambda^l - s^l$ of (17). Convergence of d-IP can be ensured by specifying the d-CG stopping criterion $\|r_\lambda^l\|_\infty \leq c_1 (\delta^l)^\eta$, with parameters $c_1 > 0$ and $\eta > 1$. The barrier parameter δ is updated via a rule based on the complementarity condition

$$\delta^{l+1} = \max_{i\in\mathcal{S}} \delta_i^{l+1} \quad \text{with } \delta_i^{l+1} = \theta \left(\frac{v_i^{l\top} \mu_i^l}{n_{h,i}} \right)^{1+\gamma} \tag{19}$$

for parameters $\gamma > 0$ and θ close to 1. We ensure that the Newton step does not leave the domain of the barrier function $\ln(\cdot)$ via the *fraction-to-the-boundary rule* for the primal and dual step sizes α^p and α^d,

$$\alpha^{p,l} = \min_{i\in\mathcal{S}} \alpha_i^{p,l}, \qquad \alpha^{d,l} = \min_{i\in\mathcal{S}} \alpha_i^{d,l},$$

where

$$\alpha_i^{p,l} = \min\left(\tau \min_{\Delta[w_i^l]_n < 0} \left(-\frac{[w_i^l]_n}{\Delta[w_i^l]_n} \right), 1 \right), \tag{20a}$$

$$\alpha_i^{d,l} = \min\left(\tau \min_{\Delta[\mu_i^l]_n < 0} \left(-\frac{[\mu_i^l]_n}{\Delta[\mu_i^l]_n} \right), 1 \right), \tag{20b}$$

and $\tau = 1 - (\delta^l)^\beta$ with $\beta > \gamma$ [29].

Summing up, the updates for all primal and dual variables are given by

$$z_i^{l+1} = z_i^l + \alpha^p \Delta z_i^l, \qquad \mu_i^{l+1} = \mu_i^l + \alpha^d \Delta\mu_i^l, \tag{21a}$$

$$w_i^{l+1} = w_i^l + \alpha^p \Delta w_i^l, \qquad \gamma_i^{l+1} = \gamma_i^l + \alpha^d \Delta\gamma_i^l. \tag{21b}$$

The overall d-IP method is stated in Algorithm 6, where ε is a user-specified threshold, $F^\delta(p^l)$ is the right hand side of (16), and $p_i = (z_i, w_i, \gamma_i, \mu_i)$.

Algorithm 6: d-IP [16].

Initialization: $\{p_i^0\}_{i\in\mathcal{S}}$, $\mu^0, \lambda^0, \delta^0$, ε
while $\|F^\delta(p^l)\|_\infty > \varepsilon$ **do**
  Local condensing: (S_i^l, s_i^l) via (18a).
  Solve $\sum_{i\in\mathcal{S}} S_i\lambda = \sum_{i\in\mathcal{S}} s_i$ with d-CG.
  Local step size: $(\delta_i^l, \alpha_i^{p,l}, \alpha_i^{d,l})$ via (19), (20).
  Global step size: $(\delta^l, \alpha^{p,l}, \alpha^{d,l})$.
  Local: $\Delta p_i = (\nabla F_i^\delta)^{-1}(F_i^\delta - \tilde{E}_i^\top \Delta\lambda)$
  Local update: p_i^{l+1} via (21).

Algorithm 7: Jacobi iterations [11].

Initialization: z^0 feasible, $\{\omega_i\}_{i\in\mathcal{S}}$ satisfying (24), $l_{\max}$
while $l < l_{\max}$ **do**
  $z_i^\star = \operatorname{argmin}_{z_i} \phi(z_i, z_{-i}^l)$
  Merge $z_i^\star$ and z_{-i}^l into z_i^l.
  Convex combination: $z^{l+1} = \sum_{i\in\mathcal{S}} \omega_i z_i^{l+1}$.
  Distribute z^{l+1} among all agents and form z_{-i}^{l+1}.

3.6 Jacobi iterations

Next, we summarize an approach based on the parallel Jacobi iterations from [2, Chapter 3]. First DMPC schemes based on Jacobi iterations were reported in [38, 39] and later in [35, 36]. The overlapping variant considered here is from [11].

Consider the centralized OCP (1) with $\mathbb{X}_i^\mathrm{f} = 0$ for all $i \in \mathcal{S}$. Subsequently and in contrast to the other methods, we differ from the given notation in (4). Let $z = \mathrm{col}(\boldsymbol{x}, \boldsymbol{u})$ denote the state and input trajectories over the horizon for the overall system (2). That is, in contrast to (4), the variable z *now does not contain copies* of state or input trajectories. We rewrite OCP (1) as

$$\min_z \frac{1}{2} z^\top H z + g^\top z \tag{22a}$$
$$\text{s.t.} \quad C^\mathcal{E} z = b^\mathcal{E}, \tag{22b}$$
$$C^\mathcal{I} z \le b^\mathcal{I}. \tag{22c}$$

In Jacobi iterations, each agent minimizes (22) with respect to its individual decision variables and subsequently coordinates with its neighbors. The assignment of decision variables to agents is a design choice and influences the performance of the method. Here, we choose an overlapping decomposition as in [11], where each agent is assigned the subset z_i of the decision variables z that contains its own state and input trajectories as well as the state and input trajectories of its directly interacting neighbors: $z_i \doteq \mathrm{col}(\boldsymbol{x}_j, \boldsymbol{u}_j)_{j\in\mathcal{N}_i\cup\{i\}}$. We stack all other elements of z in z_{-i}. The problem for agent i then reads

$$\min_{z_i} \phi(z_i, z_{-i}) \tag{23a}$$
$$\text{s.t.} \quad C_i^\mathcal{E} z_i + C_{-i}^\mathcal{E} z_{-i} = b_i^\mathcal{E}, \tag{23b}$$
$$C_i^\mathcal{I} z_i + C_{-i}^\mathcal{I} z_{-i} \le b_i^\mathcal{I}, \tag{23c}$$

where (23b), (23c) form the non-zero rows of (22b) and (22c) for the components of z_i, and $\phi(z_i, z_{-i}) = 1/2 z^\top H z + g^\top z$. Starting from a feasible guess z^0, the subsystems complete the iterations given in Algorithm 7, where the weights ω_i are designed such that

$$\omega_i \ge 0, \quad \sum_{i\in\mathcal{S}} \omega_i = 1. \tag{24}$$

Step 3 of Algorithm 7 can be executed in parallel by each subsystem. Step 4 merges the trajectories $z_i^\star$ of all subsystems $j \in \mathcal{N}_i \cup \{i\}$ with the trajectories z_{-i}^l of all subsystems $j \notin \mathcal{N}_i \cup \{i\}$ to obtain the ith subsystem's guess for the trajectories of the entire network z_i^l. The Jacobi iterations can be decentralized, if each subsystem performs the summation in Step 5 for its own trajectories $\mathrm{col}(\boldsymbol{x}_i, \boldsymbol{u}_i)$ and then sends the result to its neighbors.

4 Comparison and discussion

Now we compare the algorithms in terms of convergence properties, assumptions, and implications for DMPC. The main insights are summarized in Table 1.

4.1 Dual decomposition

Assumptions, convergence, feasibility, and termination

Differentiability of the dual function can be ensured if the primal objective is strongly convex and the feasible set is compact [1, Prop. 6.1.1]. In this case, the root convergence rate in the dual variables is sublinear ($\mathcal{O}(1/k)$) for DD and $\mathcal{O}(1/k^2)$ for DD-FGM. The iterates z_i^l satisfy the local constraints, but feasibility with respect to the coupling constraints is only achieved asymptotically. The methods may be terminated, if the coupling constraint violation $\nabla q(\lambda)$ is sufficiently small.

Closed-loop stability

The satisfaction of the local constraints at any iteration can be exploited if the agents are coupled only via the cost

Table 1: Comparison of distributed optimization methods for MPC.

Method	Category	Assumptions on QP (5)	Proven convergence rate	Feasible iterates	Known guarantees for non-convex OCPs	Further reading
DD	decentralized	strongly convex	sublinear $\mathcal{O}(1/k)$	no	no	[5, 25]
DD-FGM	decentralized	strongly convex	sublinear $\mathcal{O}(1/k^2)$	no	no	[9, 18, 19, 30]
ADMM	decentralized	convex	sublinear $\mathcal{O}(1/k)$	no	special cases only	[4, 17, 21]
	decentralized	strongly convex	linear	no		[3, 6, 24, 31]
ASM	ess. decentralized	strictly convex	finite-time	yes	no	[37]
d-IP	ess. decentralized	strongly convex	superlinear	no	yes	[16]
Jacobi	decentralized	convex	–	yes	no	[11, 20, 35, 36, 38, 39]

in OCP (1) and not via the dynamics or the state or input constraints. In this case, if P_{ij} and $\mathbb{X}_i^{\mathrm{f}}$ are designed such that feasibility with respect to $\mathbb{Z}_i$ implies stability, then the methods may be terminated after any iteration and yield a stabilizing input. However, if the coupling also occurs in the dynamics or state/input constraints, additional measures have to be taken as the coupling constraints are only satisfied asymptotically. For DD with proximal gradient methods, the following results exist. A stopping criterion to guarantee stability despite infeasibility is given in [18]. An a priori upper bound on the required iterations to guarantee stability when using polytopic terminal constraints is presented in [25].

Warm-starting and communication

DD and DD-FGM can be warm-started with λ_i^l obtained in the last MPC step to accelerate convergence. The methods are decentralized and require to communicate $2n_c$ floats in total per iteration.

4.2 ADMM

Assumptions, convergence, feasibility, and termination

ADMM requires less restrictive assumptions than DD. For general convex problems—i. e., neither Lipschitz continuity nor strong convexity is required—ADMM is guaranteed to converge at a sublinear rate ($\mathcal{O}(1/k)$) for *all* values of $\rho > 0$ [21, Thm. 4.1]. If the problem is strongly convex, linear convergence can be shown [17, Cor. 2]. Convergence guarantees of ADMM for special classes of non-convex problems are given in [40], but guarantees for OCPs with nonlinear dynamics are yet unavailable. As for DD, the iterates z_i^l satisfy the local constraints at any iteration, but the coupling constraints are only satisfied asymptotically. The method may be terminated if both the primal residuals $r_i^l = \|z_i^l - \bar{z}_i^l\|_\infty$ and the dual residuals $s_i^l = \rho\|z_i^l - z_i^{l-1}\|_\infty$ are sufficiently small, see [3, 4].

Closed-loop stability

Feasibility with respect to the local constraints can be exploited if the coupling occurs only in the cost function similar to DD. For general coupling setups, a stopping criterion to guarantee closed-loop stability under inexact optimization with ADMM is derived in [3].

Warm-starting and communication

ADMM can be warm-started with the solutions $\bar{z}_i^l$ and y_i^l obtained in the last MPC step. Executing the update (11b) via the averaging step as in Algorithm 3 renders ADMM a decentralized method. This is well-known in the context of distributed MPC [3, 9, 31]. This decentralized ADMM implementation requires to communicate $2n_c$ floats in total per iteration.

4.3 Distributed ASM

Assumptions, convergence, feasibility, and termination

To perform the condensing in Step 2 of Algorithm 4, ASM requires that $\bar{H}_i$ is invertible. To apply d-CG, the coefficient matrix S must be positive definite, which can be ensured by choosing H_i positive definite [37]. Assuming no cycling occurs, ASM terminates in a finite number of iterations. Notably, ASM is feasible-side convergent as all iterates z_i^l satisfy the local and coupling constraints. This can be exploited to guarantee closed-loop stability under early truncation [33].

Feasible initialization

The initial iterates z_i^0 must satisfy all constraints. In general, this is difficult to ensure by distributed computation. However, if the OCP contains only input box constraints, one can compute a feasible initialization for the ASM as follows: First, suppose that no input constraints are active. Then, solve (13) with $d_i = b_i^{\mathcal{E}}$ and thus obtain a guess for z_i^0. If input constraints are violated, then add them to the active set and repeat until z_i^0 is feasible.

Warm-starting and communication
The method can be warm-started with the solution z_i^l from the last MPC step and the corresponding active set. Unlike DD and ADMM, the active set method requires global communication in Steps 10 and 16 and to run d-CG in Step 3. d-CG communicates two floats globally and $2n_c$ floats from on neighbor-to-neighbor per iteration.

4.4 Essentially decentralized IP

Assumptions, convergence, feasibility, and termination
Due to its Newton-type nature, d-IP notably exhibits superlinear local convergence for non-convex problems under standard regularity assumptions [16, Thm. 2].[1] By replacing the barrier parameter update rule with a more aggressive one, even quadratic convergence rates are possible [7]. The iterates z_i^l satisfy the local and coupling constraints asymptotically. It is straightforward to see that the evaluation of the termination criterion in Algorithm 6 can be realized via additional neighbor-to-neighbor communication efforts. In practice, one can also limit the number of iterations or terminate d-IP early if the progress stalls. The decentralized evaluation of the termination criterion, however, warrants further investigation.

Warm-starting and communication
There is no efficient warm-starting procedure for d-IP available yet. As for ASM, global communication of scalars is required in Step 6 of Algorithm 6 and in d-CG as the inner algorithm. d-CG requires to communicate $2n_c$ floats in total per iteration between neighbors.

4.5 Jacobi iterations

Assumptions, convergence, feasibility, and termination
If OCP (1) is convex, Jacobi iterations converge to a fixed point [11]. However, convergence to a stationary point or to a minimizer is not guaranteed. The iterates z_i^l satisfy all constraints. A possible termination condition is that the iterates are sufficiently close to a fixed point, i. e., $\|z^{l+1} - z^l\| \leq \varepsilon$.

Warm-starting and communication
Jacobi iterations require a feasible initialization z_i^0 for each agent. If there is no plant-model mismatch and there are no disturbances, then the method can be initialized with the solution from the previous MPC step. However, this is in general not the case in practice and renders the application difficult. In total, each iteration requires to communicate $2n_c$ floats between neighbors.

4.6 Methods not discussed

Due to space limitations this compendium has not touched upon several methods which have seen application in the context of distributed optimization and DMPC. That is, we have not included block-coordinate descent schemes [28], the Augmented Lagrangian Alternating Direction Inexact Newton (ALADIN) method [22], and its communication efficient bi-level distributed variant [15]. While ALADIN is originally tailored to non-convex settings, it has seen first applications to convex DMPC problems [23]; an open-source implementation of ALADIN is available [14]. We remark that bi-level ALADIN [15] already included an early variant of d-CG, a later variant of which is used in the ASM and d-IP schemes [16, 37]. Besides ADMM, there exist further proximal methods suited for distributed optimization and DMPC, see [26, 34].

5 Numerical examples

This section serves two purposes: (i) We illustrate the prospect of early termination in the different algorithms considering a robot formation control problem, and (ii) we present an adversarial example which highlights the potential risks of early termination. For both examples, we design a stabilizing centralized MPC scheme with a zero terminal constraint, we formulate the OCP as a strongly convex QP, and we investigate the effects of early termination on the closed-loop system. For numerical results of DD and ADMM for randomly generated QPs we refer to [9, 18].

Simulation setup
In the following, we are interested in comparing the inner (i. e., d-CG) iterations for the bi-level algorithms ASM and d-IP. We cold-start all methods at MPC iteration $k = 0$ except for the Jacobi iterations, where we solve a modification of (22) with $H = 0$ and $g = 0$ at $k = 0$ with a centralized QP solver. We warm-start the methods for MPC iterations $k \geq 1$ with the solution obtained in the last step shifted appropriately. We furthermore tune the methods as follows and choose the parameters that yield the fastest convergence for the OCP at

1 In [16] we show Q-superlinear convergence in the outer iterations. This corresponds to R-superlinear convergence in the inner iterations if the inner algorithm terminates after a finite number of iterations, which is the case for d-CG.

$t = 0$. For DD, we choose $c \in [0.1 \cdot 2\mu/\|E\|_2^2, 0.99 \cdot 2\mu/\|E\|_2^2]$. For DD-FGM, we set $L = \|EH^{-0.5}\|_2^2$ as in [30]. For ADMM, we tune $\rho \in [10^{-3}, 10^3]$. For d-IP, we fix $(c_1, \gamma, \beta, \eta, \varepsilon) = (1, 1.01, 2, 1.01, 10^{-7})$ and tune $\theta \in [0.1, 0.9]$ and $\delta^0 \in [0.1, 1]$. For JI, we adjust $\omega_i \in [0.1, 0.9]$. For ASM, we set $\varepsilon = 10^{-6}$ and $r < 10^{-8}$ for d-CG. Moreover, we initialize ASM as described in Subsection 4.3 and we remark that this initialization is not guaranteed to succeed as the examples also include state and terminal constraints. Furthermore, there is no plant-model mismatch such that any loss in performance is only due to inexact optimization.

5.1 Formation control of mobile robots

We consider four mobile robots that move in the (x, y)-plane with dynamics

$$x_i^{k+1} = x_i^k + Tu_i, \quad x_i^0 = x_{i,0}, \quad \forall i \in \{1, \dots, 4\},$$

where $x_i \in \mathbb{R}^2$ is the position, $u_i \in \mathbb{R}^2$ is the velocity and $T = 0.2$ is the sampling interval of the controller. The velocity is limited to $\mathrm{col}(-2, -2) \le u_i \le \mathrm{col}(2, 2)$. The task is to steer the robots from an initial formation $x_0 \in \mathbb{R}^8$ into a desired formation $x_d \in \mathbb{R}^8$. The centralized OCP has the objective $\frac{1}{2}\sum_{k=0}^{N}(x^k - x_d)^\top Q(x^k - x_d) + (u^k - u_d)^\top R(u^k - u_d)$, horizon length $N = 20$, and a zero terminal constraint $\mathbb{X}^\mathrm{f} = \{x_d\}$.[2] We choose $R = I_8$ and follow the approach from [12] to design $Q = C_p^\top D C_p$ with

$$C_p = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix} \otimes I_2$$

$$D = \mathrm{diag}(0.1, 10, 10, 10, 10, 10) \otimes I_2.$$

Since the coupling occurs only in the objective and not in the constraints and as we employ a terminal constraint, DD and ADMM can be terminated at any iteration and still yield a stabilizing controller. Figure 1 shows the convergence of the respective methods to the minimizer $k = 0$, where we tuned ADMM by choosing $\rho = 1$. Compared to ASM and d-IP, ADMM converges slowly in z, but is quick to achieve a modest accuracy in u^0. Figure 2 illustrates how suboptimal inputs can achieve good control performance on this example. The results for DD are omitted in Figure 2 because of the slow convergence observed in Figure 1.

2 For $x_d \neq 0$ this problem can be written in form (1) by an appropriate state transformation. Alternatively, the OCP can directly be cast in form (4) with $g_i \neq 0$.

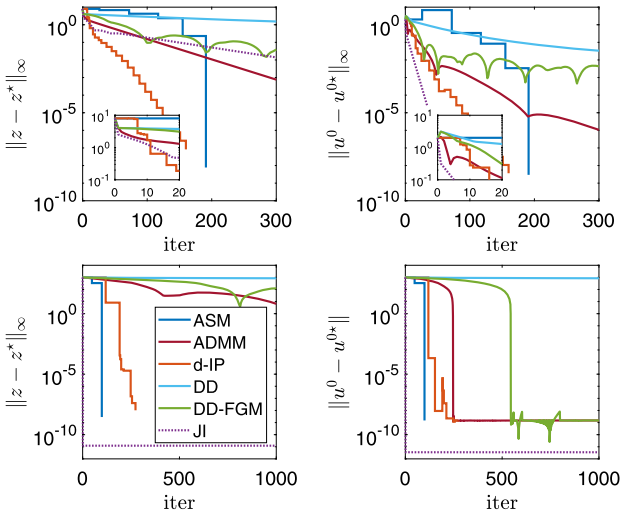


Figure 1: Convergence to the OCP minimizer at $t = 0$. Top: robot example. Bottom: adversarial example.

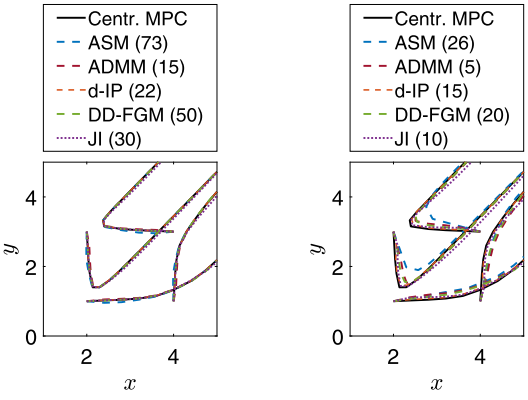


Figure 2: Robot example: closed-loop trajectories in the (x, y)-plane and corresponding maximum number of iterations per MPC step in parentheses. Left: performance similar to centralized MPC. Right: performance worse than centralized MPC.

5.2 An adversarial example

Consider three first-order systems

$$\begin{aligned} \dot{x}_1(t) &= u_1(t), & x_1(0) &= 1 \\ \dot{x}_2(t) &= x_1(t) + 4x_2(t), & x_2(0) &= 2 \\ \dot{x}_3(t) &= x_2(t) + 4x_3(t), & x_3(0) &= 3 \end{aligned}$$

with constraints $-1000 \le u_1 \le 1000$ and $-200 \le x_i \le 200$. This example is challenging as agents two and three are not stabilizable individually even though the aggregated overall system is controllable. We choose the sampling interval $T = 0.040$ and discretize the dynamics with exact zero-order hold. We design the centralized OCP with $N = 50$, $Q_i = 10$, $R = 1$, and a zero terminal constraint.

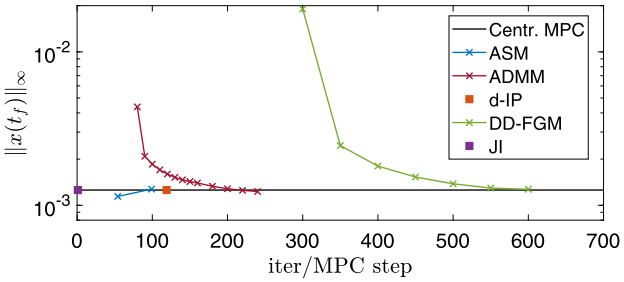


Figure 3: Adversarial example: closed-loop convergence versus maximum number of iterations per MPC step.

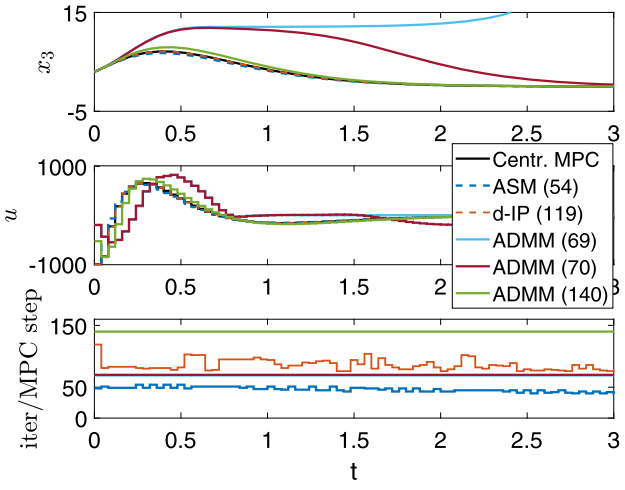


Figure 4: Adversarial example: closed-loop trajectories for ASM, d-IP and ADMM after early termination.

Figure 1 (bottom) shows the convergence to the unique minimizer at MPC iteration $k = 0$. Despite large residuals in the overall decision variable z, ADMM and DD-FGM converge to the optimal input $u^{0\star}$ after 250 and 530 iterations, respectively. In contrast, ASM attains the minimum in about 100 iterations, while d-IP gives very promising results in z for 250 iterations.

Figure 3 shows the distance to the origin of the centralized state vector after running the controller for $k_f = 125$ MPC steps (i. e., $t_f = 5$) versus the maximum number of iterations taken per MPC step. The plot thus illustrates how many iterations per MPC step a method requires to stabilize the system starting at the considered initial conditions. We remark that standard DD requires approximately 10^4 iterations per MPC step to achieve closed-loop convergence. It is hence omitted in the plot.

Figure 4 shows the closed-loop trajectories for a selected set of simulations using ADMM, d-IP, and ASM while limiting the number of iterations per MPC step. 70 ADMM iterations per MPC step suffice to achieve convergence, see Figure 3. However, the trajectory of x_3 in Figure 4 indicates that the performance is far worse than for ASM and d-IP which achieve almost centralized MPC performance.

This example allows a couple of interesting observations: Due to the lack of individual controllability of the subsystems, a slight violation of the consensus constraint (5b) leads to instability of the closed loop. As shown in Figure 4, ADMM applied with less than 70 iterations per MPC step does not stabilize. This stability loss occurs despite nominal convergence guarantees of the optimization methods and despite feasibility of the predicted state and input trajectories with respect to the local constraints (4b)–(4c). The instability is due to the infeasibility of the ADMM iterations with respect to the coupling constraints (4d).

Indeed, this adversarial example is constructed to highlight the lack of robustness which is induced by the combination of early truncation in DMPC and by a lack of local stabilizability/controllability properties. In essence, this emphasizes that local feasibility does not necessarily imply stability. Instead, local feasibility and coupling feasibility together—under appropriate assumptions on OCP (4a)—imply stability [33].

Finally, while the Jacobi iterations show very promising performance in view of Figure 3, the substantial pitfalls of this method are its need for feasible initialization (here with the centralized optimal solution) and by the lack of suitable optimality-upon-convergence properties. Indeed, for the considered example the Jacobi method converges to $z^\star$ in a single iteration as each agent solves the entire OCP in Step 3 of Algorithm 7.

This example illustrates that the promising results of [6]—which reports that already five ADMM iterations per MPC step can suffice to stabilize a set of three Van der Pol oscillators—might be affected if subsystems are not controllable. However, it also stands to reason that ADMM has a certain robustness. We remark that one unique advantage of ASM and d-IP over ADMM is that no optimization solver is needed for the agents. Indeed, embedded implementations might benefit from this feature.

6 Conclusions

This paper has reviewed six distributed or decentralized algorithms for distributed model predictive control of linear system subject to convex objectives and polytopic constraints. Specifically, we reviewed two variants of dual decomposition, ADMM, a distributed active set method, an

essentially decentralized interior point method, and Jacobi iterations. We compared the methods in terms of their convergence properties, communication effort, and structure. Simulation results for a formation control problem illustrate the convergence properties of the algorithms. Moreover, we presented an adversarial example to highlight the difficulties that can arise due to the path infeasibility (i. e., asymptotic convergence towards feasibility) of dual methods and to showcase the advantages of methods with faster asymptotic convergence. Benchmarking of the covered methods on further problems is subject to future work.

References

1. Bertsekas, D.P. 1999. *Nonlinear Programming*. Athena Scientific, Belmont.
2. Bertsekas, D.P. and Tsitsiklis, J.N. 1989. *Parallel and Distributed Computation: Numerical Methods, vol. 23*. Prentice Hall, Englewood Cliffs, NJ.
3. Bestler, A. and K. Graichen. 2019. Distributed model predictive control for continuous-time conlinear systems based on suboptimal ADMM. *Optimal Control Appl. Methods* 40(1): 1–23.
4. Boyd, S., N. Parikh, E. Chu, B. Peleato and J. Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.* 3(1): 1–122.
5. Braun, P. and L. Grüne. 2018. Verteilte Optimierung: Anwendungen in der Modellprädiktiven Regelung *Automatisierungstechnik* 66(11): 939–949.
6. Burk, D., A. Völz and K. Graichen. 2021. A modular framework for distributed model predictive control of nonlinear continuous-time systems (GRAMPC-D), *Optim. Eng.* 1–25.
7. Byrd, R.H., G. Liu and J. Nocedal. 1997. On the local behavior of an interior point method for nonlinear programming. *Numer. Anal.* 37–56.
8. Conte, C., C.N. Jones, M. Morari and M.N. Zeilinger. 2016. Distributed synthesis and stability of cooperative distributed model predictive control for linear systems. *Automatica* 69: 117–125.
9. Conte, C., T. Summers, M.N. Zeilinger, M. Morari and C.N. Jones. 2012. Computational aspects of distributed optimization in model predictive control. In: *Proc. 51st IEEE Conference on Decision and Control*, pp. 6819–6824.
10. Costantini, G., R. Rostami and D. Görges. 2018. Distributed linear quadratic regulator for the synthesis of a separable terminal cost for distributed model predictive control. In: *Proc. 57th IEEE Conference on Decision and Control*, pp. 5170–5175.
11. Doan, M.D., M. Diehl, T. Keviczky and B. De Schutter. 2017. A jacobi decomposition algorithm for distributed convex optimization in distributed model predictive control. *IFAC-PapersOnLine* 50(1): 4905–4911.
12. Ebel, H. and P. Eberhard. 2021. A comparative look at two formation control approaches based on optimization and algebraic graph theory. *Robot. Auton. Syst.* 136: 103686.
13. Engelmann, A. and Faulwasser, T., Decentralized conjugate gradients with finite-step convergence. (2021). arXiv:2102.12311.
14. Engelmann, A., Y. Jiang, H. Benner, R. Ou, B. Houska and T. Faulwasser. 2021. ALADIN-α—an open-source MATLAB toolbox for distributed non-convex optimization. *Optimal Control Appl. Methods* 1–19.
15. Engelmann, A., Y. Jiang, B. Houska and T. Faulwasser. 2020. Decomposition of nonconvex optimization via bi-level distributed ALADIN. *IEEE Trans. Control Netw. Syst.* 7(4): 1848–1858.
16. Engelmann, A., G. Stomberg and T. Faulwasser. 2021. Toward decentralized interior point methods for control. In: *Proc. 60th IEEE Conference on Decision and Control*.
17. Giselsson, P. and S. Boyd. 2016. Linear convergence and metric selection for douglas-rachford splitting and ADMM *IEEE Trans. Automat. Control* 62(2): 532–544.
18. Giselsson, P. and A. Rantzer. 2014. On feasibility, stability and performance in distributed model predictive control. *IEEE Trans. Automat. Control* 59(4): 1031–1036.
19. Giselsson, P., M.D. Doan, T. Keviczky, B.D. Schutter and A. Rantzer. 2013. Accelerated gradient methods and dual decomposition in distributed model predictive control. *Automatica* 49(3): 829–833.
20. Groß, D. and O. Stursberg. 2013. On the convergence rate of a jacobi algorithm for cooperative distributed MPC. In: *Proc. 52nd IEEE Conference on Decision and Control*, pp. 1508–1513.
21. He, B. and X. Yuan. 2012. On the O(1/n) convergence rate of the Douglas–Rachford alternating direction method. *SIAM J. Numer. Anal.* 50(2): 700–709.
22. Houska, B., J. Frasch and M. Diehl. 2016. An augmented lagrangian based algorithm for distributed nonconvex optimization. *SIAM J. Optim.* 26(2): 1101–1127.
23. Jiang, Y., P. Sauerteig, B. Houska and K. Worthmann. 2020. Distributed optimization using ALADIN for MPC in smart grids. *IEEE Trans. Control Syst. Technol.*.
24. Kögel, M. and R. Findeisen. 2012. Cooperative distributed MPC using the alternating direction multiplier method. *IFAC Proc. Vol.* 45(15): 445–450.
25. Köhler, J., M.A. Müller and F. Allgöwer. 2019. Distributed model predictive control—recursive feasibility under inexact dual optimization. *Automatica* 102: 1–9.
26. Latafat, P., N.M. Freris and P. Patrinos. 2019. A new randomized block-coordinate primal-dual proximal algorithm for distributed optimization. *IEEE Trans. Automat. Control* 64(10): 4050–4065.
27. Müller, M.A. and F. Allgöwer. 2017. Economic and distributed model predictive control: recent developments in optimization-based control. *SICE J. Control Meas. Syst. Integr.* 10(2): 39–52.
28. Necoara, I. and D. Clipici. 2013. Efficient parallel coordinate descent algorithm for convex optimization problems with separable constraints: application to distributed MPC. *J. Process Control* 23(3): 243–253.
29. Nocedal, J. and S. Wright. 2006. *Numerical Optimization*. Springer Science & Business Media, New York.
30. Richter, S., M. Morari and C.N. Jones. 2011. Towards computational complexity certification for constrained MPC based on Lagrange relaxation and the fast gradient method. In: *Proc. 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 5223–5229.

31. Rostami, R., G. Costantini and D. Görges. 2017. ADMM-based distributed model predictive control: primal and dual approaches. In: *Proc. 56th IEEE Conference on Decision and Control*, pp. 6598–6603.
32. Scattolini, R. 2009. Architectures for distributed and hierarchical model predictive control – a review. *J. Process Control* 19(5): 723–731.
33. Scokaert, P., D. Mayne and J. Rawlings. 1999. Suboptimal model predictive control (feasibility implies stability). *IEEE Trans. Automat. Control* 44(3): 648–654.
34. Stathopoulos, G., H. Shukla, A. Szucs, Y. Pu and C.N. Jones. 2016. Operator splitting methods in control. *Found. Trends Syst. Control* 3(3): 249–362.
35. Stewart, B.T., S.J. Wright and J.B. Rawlings. 2011. Cooperative distributed model predictive control for nonlinear systems. *J. Process Control* 21(5): 698–704.
36. Stewart, B.T., A.N. Venkat, J.B. Rawlings, S.J. Wright and G. Pannocchia. 2010. Cooperative distributed model predictive control. *Systems Control Lett.* 59(8): 460–469.
37. Stomberg, G., A. Engelmann and T. Faulwasser. 2021. A distributed active set method for model predictive control. *IFAC-PapersOnLine* 54(3): 263–268.
38. Venkat, A.N., I.A. Hiskens, J.B. Rawlings and S.J. Wright. 2008. Distributed MPC strategies with application to power system automatic generation control. *IEEE Trans. Control Syst. Technol.* 16(6): 1192–1206.
39. Venkat, A.N., J.B. Rawlings and S.J. Wright. 2005. Stability and optimality of distributed model predictive control. In: *Proc. 44th IEEE Conference on Decision and Control*, pp. 6680–6685.
40. Wang, Y., W. Yin and J. Zeng. 2019. Global convergence of ADMM in nonconvex nonsmooth optimization. *J. Sci. Comput.* 78(1): 29–63.
41. Xiao, G. and F. Liu. 2020. Observer-based cooperative distributed fault-tolerant model predictive control with imperfect network communication and asynchronous measurements. *Internat. J. Robust Nonlinear Control* 30(12): 4531–4549.

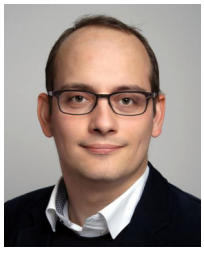
Bionotes

Gösta Stomberg
Institute for Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany
goesta.stomberg@tu-dortmund.de

Gösta Stomberg received the M. Sc. degree in mechanical and computational engineering from TU Darmstadt, Germany, in 2019. Since 2020, he has been a Ph. D. student at the Institute for Energy Systems, Energy Efficiency and Energy Economics at TU Dortmund University, Germany. His research interests include distributed optimization and model predictive control.

Alexander Engelmann
Institute for Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany
alexander.engelmann@tu-dortmund.de

Alexander Engelmann received the M. Sc. degree in electrical engineering and information technology in 2016, and the Ph. D. degree in informatics in 2020, both from the Karlsruhe Institute of Technology, Karlsruhe, Germany. Currently, he is a postdoc in the optimization and control group at the Institute for Energy Systems, Energy Efficiency and Energy Economics at TU Dortmund University, Germany. His research focuses on distributed optimization and optimal control for power and energy systems.



Timm Faulwasser
Institute for Energy Systems, Energy Efficiency and Energy Economics, TU Dortmund University, 44227 Dortmund, Germany
timm.faulwasser@ieee.org

Timm Faulwasser has studied engineering cybernetics at the University of Stuttgart. From 2008 until 2012 he was a member of the International Max Planck Research School for Analysis, Design and Optimization in Chemical and Biochemical Process Engineering Magdeburg; he obtained his Ph. D. from the Department of Electrical Engineering and Information Technology, Otto-von-Guericke University Magdeburg, Germany in 2012. After postdocs at École Polytechnique Fédérale de Lausanne, Switzerland, and at Karlsruhe Institute of Technology, Germany, he joined the Department of Electrical Engineering and Information Technology at TU Dortmund University, Germany in 2019. Currently, he serves as associate editor for IEEE Transactions on Automatic Control, IEEE CSS Letters, European Journal of Control, and Mathematics of Control Systems and Signals. His research interests are optimization and control of uncertain nonlinear systems and cyber-physical networks with applications in energy, mechatronics, process control, climate economics, and beyond.