A Distributed Active Set Method for Model Predictive Control

Gösta Stomberg * **Alexander Engelmann** * **Timm Faulwasser** *

* *Institute for Energy Systems, Energy Efficiency and Energy Economics, Faculty of Electrical Engineering and Information Technology, TU Dortmund University, 44227 Dortmund, Germany (e-mail: {goesta.stomberg,alexander.engelmann}@tu-dortmund.de, timm.faulwasser@ieee.org)*

Abstract: This paper presents a novel distributed active set method for model predictive control of linear systems. The method combines a primal active set strategy with a decentralized conjugate gradient method to solve convex quadratic programs. An advantage of the proposed method compared to existing distributed model predictive algorithms is the primal feasibility of the iterates. Numerical results show that the proposed method can compete with the alternating direction method of multipliers in terms of communication requirements for a chain of masses example.

Keywords: distributed control, model predictive control, active set methods, conjugate gradient methods, ADMM

1. INTRODUCTION

In many control applications such as process control, distributed model predictive control (DMPC) is promising due to the absence of central coordination and a limited information exchange. DMPC schemes are often realized by solving optimal control problems (OCPs)—respectively, appropriate discretizations thereof—by means of distributed optimization algorithms. A challenge is that many distributed optimization algorithms guarantee consensus constraint satisfaction only asymptotically. This implies that per sampling instant usually many distributed optimization iterations have to be executed in order to ensure closed-loop properties such as stability and recursive feasibility (Kögel and Findeisen, 2012; Mota et al., 2012; Pu et al., 2014; Rostami et al., 2017). To overcome this limitation and to enable early termination of the optimization, constraint-tightening approaches have been proposed in Doan et al. (2011); Giselsson and Rantzer (2014). Wang and Ong (2017) extend these approaches to state constraints. However, constraint tightening potentially leads to a loss in control performance, or to a reduced domain of attraction resulting from the tightened constraint set.

A second line of research enables decomposition via coordinate descent methods, where each subsystem solves its own OCP while keeping the influence of the neighboring systems fixed. Such methods guarantee feasible iterates (Necoara, 2012). A Jacobi-type method has been proposed by Stewart et al. (2010), where coupling variables are averaged between subsystems and a linear rate of convergence has been shown by Groß and Stursberg (2013). This approach was extended to event-based communication in Groß and Stursberg (2016). A condition for verifying convergence a posteriori is presented in (Dang Doan et al., 2017), where the authors also provide convergence guarantees for the special case of chain-linked systems. However, a general shortcoming of Jacobi-type iterations is that—although they decrease the objective in every iteration—convergence to a minimizer is often not guaranteed. Moreover, for fast convergence they require carefully chosen weights. For a comprehensive overview on DMPC approaches we refer to Müller and Allgöwer (2017).

This paper aims to overcome the conservatism introduced by constraint tightening. We introduce a novel distributed active set method (ASM) that builds upon a decentralized conjugate gradient (DCG) method. The method converges to the *exact solution* of an OCP in a finite number of iterations and thus avoids constraint tightening. Moreover, the proposed method has the advantage that the iterates remain feasible after a feasible initial point has been computed. This makes early termination possible without losing recursive feasibility guarantees. Furthermore, our method is tuning-free, has a practically super-linear convergence rate and numerical examples indicate a low communication footprint.

The paper is structured as follows. Section 2 formulates the DMPC scheme for the control of a linear system network. Section 3 presents the proposed distributed ASM for solving the arising QPs in the DMPC scheme. Section 4 presents numerical results on an example chain of masses system where our method is compared to ADMM. Appendix A briefly discusses the ADMM variant we use as a benchmark.

2. PROBLEM STATEMENT

We consider a network of linear time-invariant systems $i \in \mathcal{M} = \{1, \dots, M\}$ with dynamics

$$x_i^+ = A_{ii}x_i + B_i u_i + \sum_{j \in \mathcal{M}_i^{\text{in}}} A_{ij}x_j, \quad x_i(0) = x_{i,0}. \tag{1}$$

where $x_i \in \mathbb{R}^{n_i}$ and $u_i \in \mathcal{U}_i \subset \mathbb{R}^{m_i}$ denote the state and input of agent i. We use a subscript $(\cdot)_i$ throughout the paper to mark a variable's association to agent i. The matrices $A_{ij} \in \mathbb{R}^{n_i \times n_j}$ create state couplings between neighboring agents. We distinguish between in-neighbors $\mathcal{M}_i^{\text{in}} = \{j \in \mathcal{M} \mid A_{ij} \neq 0\}$ and out-neighbors $\mathcal{M}_i^{\text{out}} = \{j \in \mathcal{M} \mid A_{ji} \neq 0\}$ and we denote their union by $\mathcal{M}_i \doteq \mathcal{M}_i^{\text{in}} \cup \mathcal{M}_i^{\text{out}}$. The dynamics (1) can be rewritten as

$$x_i^+ = A_{ii}x_i + B_i u_i + \sum_{j \in \mathcal{M}_i^{\text{in}}} A_{ij} v_{ji}, \quad x_i(0) = x_{i,0} \tag{2}$$

where $v_{ji} \doteq x_j$ are state copies. We gather the state copies of agent i's in-neighbors in

$$v_i \doteq \text{col}_{j \in \mathcal{M}_i^{\text{in}}}(v_{ji}) \in \mathbb{R}^{v_i}.$$

Assumption 1. (Input Constraints). The sets $\mathcal{U}_i$ are compact convex polytopes and contain the origin in their interior. □

Remark 1. (State Constraints). For the sake of simplicity, we consider only input constraints in the present paper. Our approach can be extended to include state constraints as well as terminal constraints. □

Let $x \doteq \text{col}_{i \in \mathcal{M}}(x_i) \in \mathbb{R}^n$ denote the overall system state and $u \doteq \text{col}_{i \in \mathcal{M}}(u_i) \in \mathbb{R}^m$ the overall input. The overall system dynamics are $x^+ = Ax + Bu$, where $A \in \mathbb{R}^{n \times n}$ is a block matrix with entries A_{ij} and $B = \text{diag}_{i \in \mathcal{M}}(B_i) \in \mathbb{R}^{n \times m}$.

Assumption 2. (Controllability). The overall system (A, B) is controllable. □

2.1 MPC Formulation

Let $\boldsymbol{x}_i$, $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ denote the state, input and copy sequences $\boldsymbol{x}_i^\top \doteq [x_i^{0\top}, \dots, x_i^{N-1\top}]$, $\boldsymbol{u}_i^\top \doteq [u_i^{0\top}, \dots, u_i^{N-1\top}]$ and $\boldsymbol{v}_i^\top \doteq [v_i^{0\top}, \dots, v_i^{N-1\top}]$ for agent i with horizon N. The OCP reads

$$\min_{\boldsymbol{x}_i, x_i^N, \boldsymbol{u}_i, \boldsymbol{v}_i} \quad \sum_{i \in \mathcal{M}} J_i(\boldsymbol{x}_i, x_i^N, \boldsymbol{u}_i) \tag{3a}$$

subject to for all $i \in \mathcal{M}$:

$$x_i^+ = A_{ii}x_i + B_i u_i + \sum_{j \in \mathcal{M}_i^{\text{in}}} A_{ij} v_{ji}, \; x_i^0 = x_{i,0}, \tag{3b}$$

$$u_i \in \mathcal{U}_i, \tag{3c}$$

$$v_{ji} = x_j \quad \forall j \in \mathcal{M}_i^{\text{in}}, \tag{3d}$$

where the objective for agent $i \in \mathcal{M}$ is given by

$$J_i(\boldsymbol{x}_i, x_i^N, \boldsymbol{u}_i) \doteq \frac{1}{2} x_i^{N\top} P_i x_i^N + \frac{1}{2} \sum_{k=1}^{N-1} (x_i^{k\top} Q_i x_i^k + u_i^{k\top} R_i u_i^k).$$

Assumption 3. The weighting matrices Q_i and R_i are positive definite. The matrices P_i are positive semi-definite. □

Given measured or observed states $x_{i,0}$, the MPC scheme solves this OCP for the optimal input sequences $\boldsymbol{u}_i^*$ and then applies the first part of the input sequence $u_i^{0,*}$ as a control input to each agent. This MPC scheme asymptotically stabilizes the origin given Assumptions 1–3 for sufficiently long horizons N on a set of initial conditions $\mathbb{X}_0$. The set $\mathbb{X}_0$ on which closed-loop stability can be guaranteed depends on the interplay of the input constraints $\mathcal{U}$ and the stability of A, cf. (Boccia et al., 2014).

2.2 Distributed OCP Formulation

We next aggregate the state, input and copy trajectories into decision variables for each agent into $z_i^\top \doteq [\boldsymbol{x}_i^\top, x_i^{N\top}, \boldsymbol{u}_i^\top, \boldsymbol{v}_i^\top] \in \mathbb{R}^{N_{zi}}$ where $\boldsymbol{v}_i^\top \doteq [v_i^{0\top}, \dots, v_i^{N-1\top}] \in \mathbb{R}^{Nv_i}$ and $n_{zi} \doteq (N+1)n_i + N(m_i + v_i)$. The decision variables z_i are not typeset bold to simplify the notation even though they contain trajectories. With this we rewrite OCP (3) as the partially separable quadratic program (QP)

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

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

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

$$\sum_{i \in \mathcal{M}} C_i^c z_i = 0. \tag{4d}$$

The Hessian matrix is given by

$$H_i \doteq \text{diag}(\tilde{\boldsymbol{Q}}_i, P_i, \boldsymbol{R}_i, [\tilde{\boldsymbol{Q}}_j]_{j \in \mathcal{M}_i^{\text{in}}}) \in \mathbb{R}^{n_{zi} \times n_{zi}}, \tag{5}$$

where $\tilde{\boldsymbol{Q}}_i \doteq 1/(|\mathcal{M}_i^{\text{out}}| + 1) \cdot \text{diag}(Q_i, \dots, Q_i) \in \mathbb{R}^{Nn_i \times Nn_i}$ and $\boldsymbol{R}_i \doteq \text{diag}(R_i, \dots, R_i) \in \mathbb{R}^{Nm_i \times Nm_i}$. This choice of $\tilde{\boldsymbol{Q}}_i$ spreads the cost associated with state x_i evenly among agent i and its outgoing neighbors. The equality constraints (4b) with $C_i^{\mathcal{E}} \in \mathbb{R}^{n_{hi} \times n_{zi}}$ include the initial condition and dynamics (2); the inequality constraints (4c) include the input constraints (3c); and the coupling constraints (4d) with $C_i^c \in \mathbb{R}^{n_c \times n_{zi}}$ include the coupling between states and copies (3d). We propose to solve OCP (4) with the following distributed ASM.

3. DISTRIBUTED ACTIVE SET METHOD

Active set methods (ASMs) are well-known for solving inequality-constrained convex QPs. In particular, primal ASMs produce primal feasible iterates (Nocedal and Wright, 2006), which is advantageous for MPC. We next present a distributed primal ASM for solving OCP (4).

3.1 Distributed Active Set Method

Let $c_i^{j\mathcal{I}}$ denote the j^{th} row of $C_i^{\mathcal{I}}$ and $b_i^{j\mathcal{I}}$ the j^{th} element of $b_i^{\mathcal{I}}$. The active set of agent i is given by

$$\mathcal{A}(z_i) \doteq \{j \mid c_i^{j\mathcal{I}} z_i = b_i^{j\mathcal{I}}\}.$$

The proposed method first chooses an initial active set $\mathcal{A}(z_i^0)$ and a corresponding iterate z_i^0 for each agent. Afterwards, the method takes steps $z_i^{n+1} \doteq z_i^n + \alpha^n \Delta z_i^n$ until it has converged. The step directions Δz_i^n are obtained by solving the equality constrained QP

$$\min_{\Delta z_i} \quad \sum_{i \in \mathcal{M}} \frac{1}{2} (\Delta z_i^\top H_i \Delta z_i + g_i^{n\top} \Delta z_i) \tag{6a}$$

$$\text{s.t.} \quad C_i^n \Delta z_i = d_i \quad \mid \gamma_i \quad \forall i \in \mathcal{M}, \tag{6b}$$

$$\sum_{i \in \mathcal{M}} C_i^c \Delta z_i = 0 \quad \mid \lambda_C \tag{6c}$$

where $C_i^n \doteq \begin{bmatrix} C_i^{\mathcal{E}} \\ \text{col}_{j \in \mathcal{A}(z_i)}[c_i^{j\mathcal{I}}] \end{bmatrix}$, $g_i^n \doteq H_i z_i^n$ and $d_i = 0$. Here, γ_i and λ_C are the Lagrange multipliers associated with the constraints (6b) and (6c). Then, each agent

computes the largest step length $\alpha_i^n \in (0,1]$ such that primal feasibility is maintained (Nocedal and Wright, 2006):

$$\alpha_i^n \doteq \min\left\{1, \min_{j\notin\mathcal{A}(z_i^n), c_i^{j\mathcal{I}}p_i^n<0} \frac{b_i^{j\mathcal{I}} - c_i^{j\mathcal{I}} z_i^n}{c_i^{j\mathcal{I}} p_i^n}\right\}. \quad (7)$$

The step length $\alpha^n = \min\{\alpha_1^n, \dots, \alpha_M^n\}$ is then chosen for all agents to obtain z_i^{n+1}. This choice ensures that z_i^{n+1} remains feasible for all agents. If $\alpha < 1$, then an inactive constraint of one agent is blocking and the constraint is added to the respective agent's active set. This process is repeated until $\|\Delta z_i^n\| < \varepsilon \quad \forall i \in \mathcal{M}$ where ε is small. Then, each agent checks for dual feasibility

$$\gamma_i^{j\mathcal{I}} \geq 0, \quad \forall j \in \mathcal{A}(z_i^n).$$

If dual feasibility is attained for all agents, then z^n is returned as solution to problem (6). Else the constraint corresponding to the smallest Lagrange multiplier among all agents is removed from the respective agent's active set. Algorithm 1 summarizes the ASM and it further contains the condensing and DCG steps that we subsequently describe in Subsections 3.2 and 3.3.

Remark 2. (ASM initialization). Different ways of initializing ASMs with $\mathcal{A}(z_i^0)$ and z_i^0 have been reported in the literature. Ferreau et al. (2008) present an ASM that is initialized on a homotopy path between the current and the previous sample. Klaučo et al. (2019) propose to warm-start ASMs with classification methods from supervised learning. Here, we initialize with the optimal active set from the previous MPC iteration as a warm-start or with $\mathcal{A}(z_i^0) = \emptyset$ if no previous MPC iteration is available. We then find z_i^0 by solving a QP similar to (4) but with (4c) replaced by the active set. If the obtained z_i^0 violates an inequality constraint, then this constraint is added to the active set. This is repeated until a feasible initialization is obtained. □

Remark 3. (Early termination). The active set method produces primal feasible iterates that satisfy (4b)–(4d). It can therefore be terminated early and still guarantee stability for MPC schemes with terminal constraints where feasibility implies stability (Scokaert et al., 1999). □

Algorithm 1 Distributed ASM

Initialization: $\mathcal{A}(z_i^0)$ **and** z_i^0**.**
Repeat until convergence:
  Condense QP (6) into (8).
  Solve (8) with DCG for λ_C^n.
  Obtain Δz_i via backsubstitution (12) for all $i \in \mathcal{M}$.
  If $\|\Delta z_i\| < \varepsilon \quad \forall i \in \mathcal{M}$:
    $\gamma_i^n = (C_i^n C_i^{n\top})^{-1} C_i^n(-g_i - C_i^{c\top}\lambda_C^n) \; \forall i \in \mathcal{M}$.
    If $\gamma_i^{j\mathcal{I}} \geq 0 \quad \forall j \in \mathcal{A}(z_i^n), \quad \forall i \in \mathcal{M}$:
      Return z_i^n **and terminate**.
    Else:
      Find the smallest $\gamma_i^{j\mathcal{I}}$ among all $i \in \mathcal{M}$.
      Remove the respective constraint j from $\mathcal{A}(z_i^n)$.
  Else:
    Compute α_i^n locally according to (7) for all $i \in \mathcal{M}$.
    Determine agent with the smallest α_i and set $\alpha = \alpha_i$.
    $\mathcal{A}(z_i^n) \leftarrow \mathcal{A}(z_i^n) \cup \{j\}$ where j is the blocking constraint.
    Local step: $z_i^{n+1} \leftarrow z_i^n + \alpha^n \Delta z_i^n$ for all $i \in \mathcal{M}$.
  End If

3.2 Condensing the QP

The core idea of the distributed ASM is to solve QP (6) with a decentralized conjugate gradient method (DCG) from (Engelmann and Faulwasser, 2021). To apply DCG, we condense and rewrite QP (6) as

$$\left(\sum_{i\in\mathcal{M}} S_i\right)\lambda_C = \sum_{i\in\mathcal{M}} s_i, \quad (8)$$

where S_i and s_i are yet to be defined. In particular, $\sum_{i\in\mathcal{M}} S_i$ is positive definite and the matrices S_i have favorable sparsity properties that motivate DCG.

To arrive at (8), we first apply the nullpace method (Nocedal and Wright, 2006) to eliminate the equality constraints $C_i \in \mathbb{R}^{n_{hi}\times n_{zi}}$. Let

$$\Delta z_i = Z_i v_i + Y_i w_i, \quad (9)$$

where the columns of $Z_i \in \mathbb{R}^{n_{zi}\times(n_{zi}-n_{hi})}$ form a nullspace of C_i and $Y_i \in \mathbb{R}^{n_{zi}\times n_{hi}}$ is chosen such that $[Z_i \; Y_i]$ is invertible. This choice of Y_i together with $C_i Z_i = 0$ result in $C_i Y_i$ being nonsingular and inserting into the equality constraints yields $w_i = (C_i Y_i)^{-1} d_i$. We insert (9) into (6) and obtain

$$\min_{v_i} \quad \sum_{i\in\mathcal{M}} \left(\frac{1}{2} v_i^\top \bar{H}_i v_i + \bar{g}_i^\top v_i\right) \quad (10a)$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{M}} (\bar{C}_i^c v_i + b_i) = 0 \quad | \; \lambda_C \quad (10b)$$

where $\bar{H}_i \doteq Z_i^\top H_i Z_i$, $\bar{g}_i \doteq Z_i^\top g_i + Z_i^\top H_i Y_i w_i$, $\bar{C}_i^c \doteq C_i^c Z_i$ and $b_i \doteq C_i^c Y_i w_i$. The KKT conditions of (10) read

$$\begin{bmatrix} \bar{H}_1 & & & \bar{C}_1^{c\top} \\ & \ddots & & \vdots \\ & & \bar{H}_M & \bar{C}_M^{c\top} \\ \bar{C}_1^c & \dots & \bar{C}_M^c & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_M \\ \lambda_C \end{bmatrix} = \begin{bmatrix} -\bar{g}_1 \\ \vdots \\ -\bar{g}_M \\ -\sum_i b_i \end{bmatrix}. \quad (11)$$

Assumption 4. The matrices $\bar{H}_i$ are positive definite. □

Utilizing that $\bar{H}_i$ is positive definite for all $i \in \mathcal{M}$, we solve (11) for $v_i = \bar{H}_i^{-1}(-\bar{g}_i - \bar{C}_i^{c\top}\lambda_C)$. Inserting back into the KKT conditions yields (8) where $S_i \doteq \bar{C}_i^c \bar{H}_i^{-1} \bar{C}_i^{c\top}$ and $s_i \doteq b_i - \bar{C}_i^c \bar{H}_i^{-1} \bar{g}_i$. The QP (6) has hence been reduced to a set of n_c linear equations. The solution to QP (6) can be obtained via backsubstitution:

$$\Delta z_i = Z_i \bar{H}_i^{-1}(-\bar{g}_i - \bar{C}_i^{c\top}\lambda_C) + Y_i (C_i Y_i)^{-1} d_i. \quad (12)$$

Remark 4. The above derivation allows for $d_i \neq 0$ in (6b). This is needed to perform the ASM initialization with DCG as described in Remark 2. Once the ASM initialization is complete, we use DCG to find Δz_i and in this case $d_i = 0$. □

3.3 The Decentralized Conjugate Gradient Method

Next, we recall the decentralized conjugate gradient (DCG) method from (Engelmann and Faulwasser, 2021) to solve the positive definite linear system of equations (8) in decentralized fashion. Consider the system of equations

$$S\lambda_C = s,$$

where $S \doteq \sum_i S_i$ is positive definite and $s \doteq \sum_i s_i$. The centralized conjugate gradient method (Nocedal and Wright, 2006) solves this system iteratively for λ_C until the

residual at iteration n defined as $r^n \doteq s - S\lambda_C^n$ vanishes. The iterations read

$$\alpha^n = \frac{r^{n\top} r^n}{p^{n\top} S p^n}, \tag{13a}$$
$$\lambda_C^{n+1} = \lambda_C^n + \alpha^n p^n, \tag{13b}$$
$$r^{n+1} = r^n - \alpha^n S p^n, \tag{13c}$$
$$\beta^n = \frac{r^{n+1\top} r^{n+1}}{r^{n\top} r^n}, \tag{13d}$$
$$p^{n+1} = r^{n+1} + \beta^n p^n, \tag{13e}$$

with the initialization $r^0 = p^0 = s - S\lambda_C^0$. The idea of DCG is to introduce local versions of the CG variables λ_C, r and p and to decompose the CG updates into local updates. Since DCG is a reformulation of the centralized CG method, DCG is guaranteed to solve the QP (6) in a finite number of iterations and exhibits a practically superlinear convergence rate.

We exploit sparsity properties of the S_i matrices that result from zero rows in C_i^c. The rows of C_i^c couple variables between two agents: the states of one agent and copies of those states in a neighboring agent. Each row in C_i^c represents a coupling constraint and the row is non-zero if and only if agent i is coupled via the associated constraint. Observe that zero rows of C_i^c lead to zero rows and columns in S_i, because $S_i = C_i^c Z_i \bar{H}_i Z_i^\top C_i^{c\top}$. Hence, only those rows and columns of S_i are non-zero that correspond to the coupling constraints that couple agent i to its neighbors $\mathcal{M}_i$. Let $c_i^{j,c}$ denote the j^{th} row of C_i^c. We let $\mathcal{C}(i) \doteq \{j \,|\, c_i^{j,c} \neq 0\}$ denote the consensus constraints that couple agent i to its neighbors. We next introduce matrices $I_{\mathcal{C}(i)} \in \mathbb{R}^{|\mathcal{C}(i)| \times n_c}$ that map from global CG variables to local variables. The matrices $I_{\mathcal{C}(i)}$ are obtained by taking the identity matrix $I \in \mathbb{R}^{n_c \times n_c}$ and by subsequently eliminating all rows that do not belong to any constraint in $\mathcal{C}(i)$. With this we introduce the local variables

$$\lambda_{C,i} \doteq I_{\mathcal{C}(i)} \lambda_C,$$
$$r_i \doteq I_{\mathcal{C}(i)} r,$$
$$p_i \doteq I_{\mathcal{C}(i)} p.$$

Let $\Lambda \doteq \sum_{i \in \mathcal{M}} I_{\mathcal{C}(i)}^\top I_{\mathcal{C}(i)}$ and $\Lambda_i \doteq I_{\mathcal{C}(i)} \Lambda I_{\mathcal{C}(i)}^\top$. We rewrite (13a) as $\alpha^n = \frac{\eta^n}{\sigma^n}$ where $\eta^n \doteq r^{n\top} r^n$ and $\sigma^n \doteq p^{n\top} S p^n$. This can be decomposed into

$$\eta^n = \sum_{i \in \mathcal{M}} \eta_i^n, \quad \eta_i^n \doteq r_i^{n\top} \Lambda_i^{-1} r_i^n,$$
$$\sigma^n = \sum_{i \in \mathcal{M}} \sigma_i^n, \quad \sigma_i^n \doteq p_i^{n\top} \hat{S}_i p_i^n,$$

where $\hat{S}_i \doteq I_{\mathcal{C}(i)} S_i I_{\mathcal{C}(i)}^\top$. The computation of α^n therefore requires the local computation of η_i and σ_i for each agent and two subsequent scalar global sums. We next decentralize the updates of the Lagrange multiplier λ_C and the step direction p. We multiply (13b) and (13e) by $I_{\mathcal{C}(i)}$ from the left and get

$$\lambda_{C,i}^{n+1} = \lambda_{C,i}^n + \frac{\eta^n}{\sigma^n} p_i^n,$$
$$p_i^{n+1} = r_i^{n+1} + \beta^n p_i^n,$$

where $\beta^n = \frac{\eta^{n+1}}{\eta^n}$ as before. Note that both updates can be performed locally. In the last step, we decompose the residual update

$$r_i^{n+1} = r_i^n - \frac{\eta^n}{\sigma^n} \sum_{j \in \mathcal{M}(i) \cup i} I_{ij} \hat{S}_j p_j^n,$$

where $I_{ij} \doteq I_{\mathcal{C}(i)} I_{\mathcal{C}(j)}^\top$. This requires local communication among neighboring agents. We have now decomposed all CG updates and can summarize DCG in Algorithm 2. The initialization of the residual and step direction must satisfy $r^0 = p^0 = s - S\lambda_C^0$ where λ_C^0 can be chosen. This can be decomposed into

$$r_i^0 = p_i^0 = \sum_{j \in \mathcal{M}(i) \cup i} I_{ij} I_{C(j)} s_j - \sum_{j \in \mathcal{M}(i) \cup i} I_{ij} \hat{S}_j \lambda_{C,j}^0$$

and hence requires neighbor-to-neighbor communication.

Algorithm 2 DCG

Initialization: λ_C^0 **and** $r^0 = p^0 = s - S\lambda_C^0$
Repeat until $\|r_i^n\| < \varepsilon \quad \forall i \in \mathcal{M}$ **:**

1. $\eta_i^n = r_i^{n\top} \Lambda_i^{-1} r_i^n, \quad \sigma_i^n = p_i^{n\top} \hat{S}_i p_i^n$
2. $\eta^n = \sum_{i \in \mathcal{M}} \eta_i^n, \quad \sigma^n = \sum_{i \in \mathcal{M}} \sigma_i^n$
3. $\lambda_{C,i}^{n+1} = \lambda_{C,i}^n + \frac{\eta^n}{\sigma^n} p_i^n$
4. $r_i^{n+1} = r_i^n - \frac{\eta^n}{\sigma^n} \sum_{j \in \mathcal{M}_i \cup i} I_{ij} \hat{S}_j p_j^n$
5. $p_i^{n+1} = r_i^{n+1} + \frac{\eta^{n+1}}{\eta^n} p_i^n$

3.4 Communication analysis

We rely on three measures to analyse the communication requirements of ASM/DCG: the number of floats that are sent to and from a central coordinator (global floats), the number of booleans that are sent to and from a central coordinator (global booleans), and the number of floats that are sent on a neighbor-to-neighbor basis (local floats). In each DCG iteration, each agent sends η_i^n and σ_i^n to a central coordinator. The coordinator then computes the sums σ^n and η^n and returns them to each agent. This totals in $4M$ global floats per DCG iteration. In addition, each agent sends a convergence flag to the coordinator and the coordinator returns a global convergence flag to each agent, which gives $2M$ global booleans per DCG iteration. In step 4. of the DCG iteration, agent j sends those elements of $\hat{S}_j^n p_j^n$ to neighbor i, that correspond to non-zero elements in I_{ij}. Hence, agents i and j exchange one float per coupled variable. This results in $2n_c$ local floats per DCG iteration. Within each ASM iteration, either the smallest Lagrange multiplier or the largest feasible step size have to be determined and convergence is checked. This requires $2M$ global floats and $2M$ global booleans. The communication footprint for the ASM, DCG and the ADMM variant explained in Appendix A is summarized in Table 1.

Table 1. Communication per iteration.

	global fl.	global bo.	local fl.
DCG	$4M$	$2M$	$2n_c$
ASM	$2M$	$2M$	0
ADMM	0	$2M$	$2n_c$

4. NUMERICAL RESULTS

We compare the presented ASM with an ADMM implementation (cf. Appendix A) on a chain of masses system (Conte et al., 2016). The baseline example consists of 10 masses with $m = 1\,\text{kg}$ that are coupled by springs with stiffness $k = 3\,\text{N/m}$ and dampers with coefficient $d = 3\,\text{Ns/m}$. Each mass is actuated by a force $-1\,\text{N} \leq u_i \leq 1\,\text{N}$. We do not consider constraints on the states $x_i(t) = [y_i(t)\ v_i(t)]^\top$, where $y_i(t)$ and $v_i(t)$ denote the position and velocity of mass i. The equations of motion are discretized using the Euler forward discretization with step size $T = 0.2\,\text{s}$. We choose the parameters of the MPC controller as $N = 12$, $Q_i = \text{diag}([10, 10])$, $P_i = 0$ and $R = 1$. The resulting OCP (4) has the dimension $n_z = 812$ with 260 equality constraints for the dynamics, 240 input constraints and 432 coupling constraints. We choose the following tolerances for the stopping criteria: $\|r_i\|_\infty < 10^{-7}$ for DCG and $\|\Delta z_i\|_\infty < 10^{-6}$ for ASM. We choose $\lambda^0_{C,i} = 0$ to initialize DCG and warm-start the ASM with the optimal active set from the previous MPC iteration. For ADMM, the stopping criteria $\|C^c_i(z_i - \bar{z}_i)\|_\infty \leq \varepsilon_r \min\{\max\{\|C^c_i z_i\|_\infty, \|C^c_i \bar{z}_i\|_\infty\}, 1\}$ and $\|\rho C^c_i(z^+_i - z_i)\|_\infty \leq \varepsilon_d \min\{\|\lambda_{C,i}\|_\infty, 1\}$ are used. Two different tolerance levels are chosen:

- ADMM1: $\varepsilon_r = 10^{-6}$ and $\varepsilon_d = 10^{-3}$
- ADMM2: $\varepsilon_r = 10^{-4}$ and $\varepsilon_d = 10^{-2}$.

Five case studies that each simulate the closed-loop behavior for 25 MPC iterations starting at 30 random initial positions are conducted. For the baseline case study, the initial position and velocity for each agent are chosen from a uniform distribution in the invertvals $-1\,\text{m} \leq y_{i,0} \leq 1\,\text{m}$ and $-0.5\,\text{m/s} \leq v_{i,0} \leq 0.5\,\text{m/s}$. Figure 4 shows simulation results with the closed-loop trajectories of the agents and the number of floats per sample interval that is communicated locally from neighbor-to-neighbor in the network. The output and input trajectories of the fifth agent are displayed in black for better visualization and the trajectories of the remaining agents are shown in grey. Table 2 shows the communication in terms of floats and booleans sent to and from a central coordinator (global floats) as well as floats sent on a neighbor-to-neighbor basis (local floats). We note that this table and all further tables do not include data on the first MPC iteration for each initial position as these iterations cannot be warm-started. We further note that ASM/DCG achieves an accuracy of 10^{-7} for the closed loop state trajectories compared to the centralized MPC scheme whereas ADMM1 and ADMM2 achieve only accuracy levels of 10^{-5} and 10^{-4}, respectively. Table 3 shows the number of iterations required by each algorithm per sample interval. For ASM/DCG, we count outer iterations spent in Algorithm 1 (ASM) and inner iterations spent in Algorithm 2 (DCG) separately. The iterations split into a first part to obtain z^0_i and into a second part to update the active set. For the baseline case, most ASM/DCG iterations are spent finding a feasible z^0_i. Future work may improve the initialization procedure (cf. Remark 2). Four additional case studies were conducted in addition to the baseline case. A single parameter compared to the baseline case has been changed for each of the additional cases and the results are given in Table 4. The iterations reported for ASM/DCG are inner (i.e. DCG) iterations. The results show that ASM/DCG requires less iterations than ADMM1 for all analyzed scenarios.

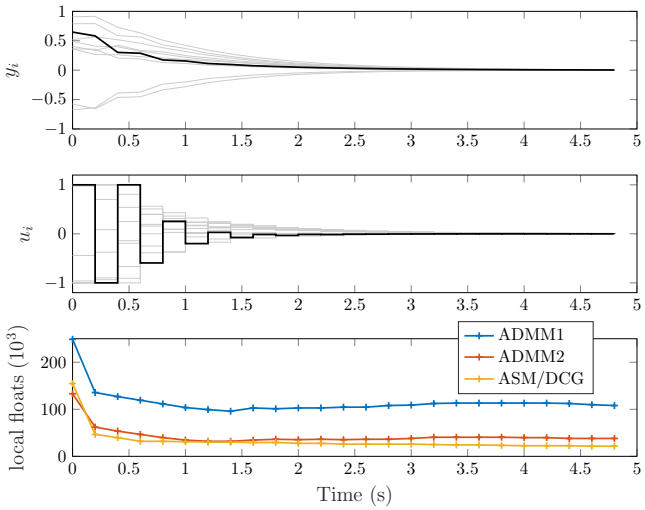

Fig. 1. Closed-loop trajectories and locally communicated floats per sampling instance.

Table 2. Communication footprint (baseline case).

	global floats		global bool.		local floats	
	mean	max	mean	max	mean	max
ASM/DCG	1.3k	3.9k	0.7k	2.1k	27k	88k
ADMM1	0	0	2.4k	3.7k	102k	160k
ADMM2	0	0	0.8k	1.6k	35k	68k

Table 3. Iterations (baseline case).

	total iter.		feas. guess		AS updating	
	mean	max	mean	max	mean	max
DCG	30	98	27	97	1	1
ASM	1	1	-	-	1	1
ADMM1	117	185	-	-	-	-
ADMM2	41	78	-	-	-	-

Table 4. Iterations and communication footprint (modified problem parameters).

	total iter.		feas. guess		local floats	
	mean	max	mean	max	mean	max
$\lvert v_{i,0}\rvert \leq 2$						
ASM/DCG	37	283	29	108	34k	251k
ADMM1	133	303	-	-	109k	262k
$N = 5$						
ASM/DCG	30	137	29	108	12k	52k
ADMM1	154	226	-	-	56k	82k
5 Masses						
ASM/DCG	26	68	25	67	11k	28k
ADMM1	106	171	-	-	41k	66k
20 Masses						
ASM/DCG	32	160	29	153	61k	301k
ADMM1	127	237	-	-	231k	433k

5. SUMMARY AND OUTLOOK

This paper has proposed a novel distributed active set method that can be used for the distributed MPC of linear systems. The method combines a usual active set approach

with a recently proposed decentralized conjugate gradient method to solve the arising equality constrained QPs. It thus is a distributed method. Our numerical case study shows a competitive communication footprint compared to an ADMM variant in terms of number of floats communicated on a neighbor-to-neighbor basis. Future work will include the extension to state and terminal constraints. In particular, the proposed scheme can be terminated early for MPC schemes involving terminal constraints and still guarantee stability because the scheme ensures primal feasibility. This aspect and improved initialization procedures may also reduce the communication requirement further.

REFERENCES

Boccia, A., Grüne, L., and Worthmann, K. (2014). Stability and feasibility of state constrained MPC without stabilizing terminal constraints. *Systems & Control Letters*, 72, 14–21.

Boyd, S., Parikh, N., and Chu, E. (2011). *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

Conte, C., Jones, C.N., Morari, M., and Zeilinger, M.N. (2016). Distributed synthesis and stability of cooperative distributed model predictive control for linear systems. *Automatica*, 69, 117–125.

Dang Doan, M., Diehl, M., Keviczky, T., and De Schutter, B. (2017). A Jacobi decomposition algorithm for distributed convex optimization in distributed model predictive control. *IFAC-PapersOnLine*, 50(1), 4905–4911.

Doan, M.D., Keviczky, T., and De Schutter, B. (2011). A dual decomposition-based optimization method with guaranteed primal feasibility for hierarchical MPC problems. *IFAC Proceedings Volumes*, 44(1), 392–397.

Engelmann, A. and Faulwasser, T. (2021). Decentralized conjugate gradients with finite-step convergence. *Submitted*, arXiv:2102.12311.

Ferreau, H.J., Bock, H.G., and Diehl, M. (2008). An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 18(8), 816–830.

Giselsson, P. and Rantzer, A. (2014). On feasibility, stability and performance in distributed model predictive control. *IEEE Transactions on Automatic Control*, 59(4), 1031–1036.

Groß, D. and Stursberg, O. (2013). On the convergence rate of a Jacobi algorithm for cooperative distributed mpc. In *52nd IEEE Conference on Decision and Control*, 1508–1513.

Groß, D. and Stursberg, O. (2016). A cooperative distributed mpc algorithm with event-based communication and parallel optimization. *IEEE Transactions on Control of Network Systems*, 3(3), 275–285.

Klaučo, M., Kalúz, M., and Kvasnica, M. (2019). Machine learning-based warm starting of active set methods in embedded model predictive control. *Engineering Applications of Artificial Intelligence*, 77, 1–8.

Kögel, M. and Findeisen, R. (2012). Cooperative distributed mpc using the alternating direction multiplier method. *IFAC Proceedings Volumes*, 45(15), 445–450.

Mota, J.F.C., Xavier, J.M.F., Aguiar, P.M.Q., and Püschel, M. (2012). Distributed ADMM for model predictive control and congestion control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 5110–5115.

Müller, M.A. and Allgöwer, F. (2017). Economic and distributed model predictive control: Recent developments in optimization-based control. *SICE Journal of Control, Measurement, and System Integration*, 10(2), 39–52.

Necoara, I. (2012). Suboptimal distributed MPC based on a block-coordinate descent method with feasibility and stability guarantees. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 4480–4485.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer Science & Business Media.

Pu, Y., Zeilinger, M.N., and Jones, C.N. (2014). Inexact fast alternating minimization algorithm for distributed model predictive control. In *53rd IEEE Conference on Decision and Control*, 5915–5921.

Rostami, R., Costantini, G., and Görges, D. (2017). ADMM-based distributed model predictive control: Primal and dual approaches. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 6598–6603. IEEE.

Scokaert, P.O., Mayne, D.Q., and Rawlings, J.B. (1999). Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3), 648–654.

Stewart, B.T., Venkat, A.N., Rawlings, J.B., Wright, S.J., and Pannocchia, G. (2010). Cooperative distributed model predictive control. *Systems & Control Letters*, 59(8), 460–469.

Wang, Z. and Ong, C.J. (2017). Distributed model predictive control of linear discrete-time systems with local and global constraints. *Automatica*, 81, 184–195.

Appendix A. ADMM

We briefly comment on the ADMM variant that we use here for the comparison with ASM/DCG. We refer the reader to (Boyd et al., 2011) for an extensive overview of ADMM and to (Rostami et al., 2017) for the application of ADMM to distributed MPC. To apply ADMM, we introduce the trajectory average $\bar{\boldsymbol{x}}_i \in \mathbb{R}^{Nn_i}$ and the decision variable $\bar{z}_i^\top \doteq [\bar{\boldsymbol{x}}_i^\top, x_i^{N\top}, \boldsymbol{u}_i^\top, [\bar{\boldsymbol{x}}_j^\top]_{j\in\mathcal{M}_i^{\text{in}}}] \in \mathbb{R}^{n_{zi}}$, which satisfy the coupling constraints (4d) by design. ADMM alternately updates z_i, which satisfies (4b) and (4c), and $\bar{z}_i$ until z_i satisfies (4d) to a chosen accuracy. The $\bar{z}_i$ update requires local communication and convergence flags are sent to a global coordinator as for ASM/DCG.

Algorithm 3 ADMM

Initialization: $\lambda_{C,i}^0$ **and** $\bar{z}_i^0$

Repeat until convergence:

1. $z_i^{n+1} = \underset{z_i}{\operatorname{argmin}}\ z_i^\top H_i z_i + \lambda_{C,i}^\top C_i^c z_i + \frac{\rho}{2}\|C_i^c(z_i - \bar{z}_i^n)\|_2^2$
   $\text{s.t.}\quad C_i^{\mathcal{E}} z_i = b_i^{\mathcal{E}},\quad C_i^{\mathcal{I}} z_i \le b_i^{\mathcal{I}}$
2. Receive $[\boldsymbol{v_{ij}}]_{j\in\mathcal{M}_i^{\text{out}}}$.
3. Compute $\bar{\boldsymbol{x}}_i = \sum_{j\in\mathcal{M}_i^{\text{out}}}(\boldsymbol{x_i} + \boldsymbol{v_{ij}})/(2|\mathcal{M}_i^{\text{out}}|)$
4. Receive $[\bar{\boldsymbol{x}}_j]_{j\in\mathcal{M}_i^{\text{in}}}$ and form $\bar{z}_i$.
5. $\lambda_{C,i}^{n+1} = \lambda_{C,i}^n + \rho C_i^c(z_i^{n+1} - \bar{z}_i^{n+1})$