Distributed and Localized Model-Predictive Control—Part I: Synthesis and Implementation

Carmen Amo Alonso, Jing Shuang Li, James Anderson, and Nikolai Matni

Abstract—The increasing presence of large-scale distributed systems highlights the need for scalable control strategies where only local communication is required. Moreover, in safety-critical systems, it is imperative that such control strategies handle constraints in the presence of disturbances. In response to this need, in this article, we present the distributed and localized model-predictive control (DLMPC) algorithm for large-scale linear systems. DLMPC is a distributed closed-loop model-predictive control (MPC) scheme, wherein only local state and model information needs to be exchanged between subsystems for the computation and implementation of control actions. We use the system-level synthesis framework to reformulate the centralized MPC problem and show that this allows us to naturally impose localized communication constraints between subcontrollers. The structure of the resulting problem can be exploited to develop an alternating-directional-method-of-multipliers-based algorithm that allows for a distributed and localized computation of closed-loop control policies. We demonstrate that the computational complexity of the subproblems solved by each subsystem in DLMPC is independent of the size of the global system. DLMPC is the first MPC algorithm that allows for the scalable computation and implementation of distributed closed-loop control policies and deals with additive disturbances. In our companion paper, we show that this approach enjoys recursive feasibility and asymptotic stability.

Index Terms—Closed loop systems, control system synthesis, decentralized control, networked control systems, optimal control.

I. INTRODUCTION

Model-Predictive control (MPC) has seen widespread success across many applications. However, the need to control increasingly large-scale, distributed, and networked systems has limited its applicability. Large-scale distributed systems are often impossible to control with a centralized controller, and even when such a centralized controller can be implemented, the high computational demand of MPC renders it impractical. Thus, efforts have been made to develop distributed MPC (DMPC) algorithms, wherein subcontrollers solve a local optimization problem and potentially coordinate with other subcontrollers in the network.

Prior work: The majority of DMPC research has focused on the cooperative setting, where subcontrollers exchange state and control action information in order to coordinate their behavior so as to optimize a global objective, typically through distributed optimization [2], [3], [4], [5], [6], [7], [8], [9]. Most of these approaches rely on nominal open-loop approaches, and while nominal MPC enjoys some intrinsic robustness [10], the resulting closed loop can be destabilized by an arbitrary small disturbance [11]. Thus, to maintain robustness in the presence of additive disturbances, closed-loop policies are desirable.

Two main closed-loop MPC approaches are used. The first approach, which we use here, is to compute dynamic structured closed-loop policies using suitable parameterizations. This strategy was introduced by Goulart et al. [12]. More recent methods exploit quadratic invariance [13] and the Youla parameterization [14]. These methods allow distributed closed-loop control policies to be synthesized via convex optimization; however, the resulting optimization problem lacks structure and is not amenable to distributed optimization techniques. Thus, these methods do not scale to large systems. Similarly, recent works exploit the system-level synthesis (SLS) parameterization to design robust MPC controllers [15]; however, it is unclear how these can be applied in the distributed setting. The alternative approach is to directly extend centralized methods (i.e., constraint tightening, tube MPC) instead of relying on a convex parameterization of distributed feedback policies. Contrary to the first approach, these methods are computationally efficient, but they require precomputed stabilizing controllers and often rely on strong assumptions, such as the existence of a static structured stabilizing controller (as in [16]), which can be NP-hard to compute [17], or decoupled subsystems (as in [18]).
Overall, though many closed-loop formulations exist, they rely on strong assumptions and/or are unsuitable for distributed computation. We seek a closed-loop DMPC algorithm that (1) computes structured feedback policies via convex optimization and (2) can be solved at scale via distributed optimization. To the best of our knowledge, no such algorithm exists.

**Contributions:** In this article, we address this gap and present the distributed localized MPC (DLMPC) algorithm for linear time-invariant (LTI) systems, which allows for the distributed computation of structured feedback policies with recursive feasibility and asymptotic stability guarantees. We leverage the SLS framework [19] to define a novel parameterization of distributed closed-loop MPC policies such that the resulting synthesis problem is both convex and structured, allowing for the natural use of distributed optimization techniques. Thanks to the nature of the SLS parameterization, this approach deals with disturbances in a straightforward manner with no additional assumptions. We show that by exploiting the sparsity of the underlying distributed system and resulting closed-loop system, as well as the separability properties [19] of commonly used objective functions and constraints, we are able to distribute the computation via the alternating direction method of multipliers (ADMM) [20], thus allowing the online computation of closed-loop MPC policies to be carried out in a scalable localized manner. Our results apply to the nominal case (as presented in [1]) as well as the robust case and provide a unifying algorithm that applies to all the cases and does not rely on simplifying approximations. In the resulting implementation, each subcontroller solves a low-dimensional optimization problem requiring only local communication of state and model information. Through numerical experiments, we validate these results and confirm that the complexity of the subproblems solved at each subsystem scales as $O(1)$ relative to the state dimension of the full system for both the nominal and robust cases.

The rest of this article is organized as follows. In Section II, we present the problem formulation. In Section III, we introduce the SLS framework and show how to recast MPC as an optimization problem over system responses in the nominal and robust cases. We also highlight the differences between SLS and the disturbance-based feedback method proposed in [12]. In Section IV, we define and analyze the DLMPC algorithm. In Section V, we present a numerical experiment. Finally, Section VI concludes this article. In our companion paper [21], we provide theoretical guarantees for convergence, recursive feasibility, and asymptotic stability of this approach and extend the algorithm to deal with coupling through cost and constraints.

**Notations:** Lowercase and uppercase letters, such as $x$ and $A$, denote vectors and matrices, respectively, although lowercase letters might also be used for scalars or functions (the distinction will be apparent from the context). Bracketed indices denote the time step of the real system, i.e., the system input is $u(t)$ at time $t$, not to be confused with $x_t$ that denotes the predicted state $x$ at time $t$. Superscripted variables, e.g., $x^k$, correspond to the value of $x$ at the $k$th iteration of a given algorithm. $\| \cdot \|$ denotes the Frobenius norm. Square bracket notation, i.e., $[x]_i$, denotes the components of $x$ corresponding to subsystem $i$. Calligraphic letters, such as $\mathcal{S}$ denote sets, and lowercase script letters, such as $c$, denote a subset of $\mathbb{Z}^+$, e.g., $c = \{1, \ldots, n\}$. Boldface lower- and uppercase letters, such as $x$ and $K$, denote finite-horizon signals and block lower triangular (causal) operators, respectively:

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_T \end{bmatrix}, \quad K = \begin{bmatrix} K_0[0] \\ K_1[1] & K_1[0] \\ \vdots & \vdots & \ddots & \vdots \\ K_T[T] & \ldots & K_T[1] & K_T[0] \end{bmatrix}$$

where each $x_i$ is an $n$-dimensional vector, and each $K_{i}[j]$ represents the value of matrix $K$ at the $j$th time step computed at time $i$. $K(t, c)$ denotes the submatrix of $K$ composed of $r$ rows and $c$ columns, respectively. We denote the block columns of $K$ by $K\{1\}, \ldots, K\{T\}$, i.e., $K\{i\} := [K_0[0]^T \ldots K_T[T]^T]^T$, and we use "::" to indicate the range of columns, i.e., $K\{2 : T\}$ contains the block columns from the second to the last.

**II. Problem Formulation**

Consider a discrete-time LTI system with dynamics

$$x(t+1) = Ax(t) + Bu(t) + w(t) \quad (1)$$

where $x(t) \in \mathbb{R}^n$ is the state, $u(t) \in \mathbb{R}^p$ is the control input, and $w(t) \in \mathcal{W} \subset \mathbb{R}^m$ is an exogenous disturbance. The system is composed of $N$ interconnected subsystems (each having one or more states), so the state, control, and disturbance inputs can be suitably partitioned as $[x]_i, [u]_i$, and $[w]_i$ for each subsystem $i$, consequently inducing a compatible block structure $[A]_{ij}, [B]_{ij}$ in the system matrices $(A, B)$. We model the interconnection topology of the system as a time-invariant unweighted directed graph $\mathcal{G}(A, B)(\mathcal{E}, \mathcal{V})$, where each subsystem $i$ is identified with a vertex $v_i \in \mathcal{V}$ and an edge $e_{ij} \in \mathcal{E}$ exists whenever $[A]_{ij} \neq 0$ or $[B]_{ij} \neq 0$.

We study the case where the control input is a model-predictive controller optimizing a nominal objective and is subject to constraints on the state and the input. As is standard, at each time step $\tau$, the controller solves an optimal control problem over a finite prediction horizon of length $T$ using the current state as the initial condition

$$\min_{x_t:u_t:\gamma_t} \sum_{t=0}^{T-1} f_t(x_t, u_t) + f_T(x_T) \quad (2)$$

where $f_t(\cdot, \cdot)$ and $f_T(\cdot)$ are closed, proper, and convex and $\gamma_{\cdot} \cdot \cdot$ is a measurable function of its arguments. In the nominal (i.e., noiseless) case, $\mathcal{X}_t$ and $\mathcal{U}_t$ are closed convex sets containing the origin. When noise is present, we consider polytopic sets $\mathcal{X}_t := \{x : H_{x,t}x \leq h_{x,t}\}$ and $\mathcal{U}_t := \{u : H_{u,t}u \leq h_{u,t}\}$, where $H_{x,t}, H_{u,t}$ and $h_{x,t}, h_{u,t}$ are matrices and vectors of compatible size, respectively, and $\mathcal{W}_t$ is a norm-bounded or polytopic set.

Our goal is to define an algorithm that allows us to solve the MPC problem (2) in a distributed manner while respecting local communication constraints. To achieve this goal, we impose that
information exchange—as defined by the graph \( G(A,B) (V, E) \)—
is localized to a subset of neighboring subcontrollers. In particular, we use the notion of a \( d \)-local information exchange constraint [19], which restricts subcontrollers to exchange their state and control actions with neighbors at most \( d \)-hops away, as measured by the communication topology \( G(A,B) \). This notion is captured as follows.

**Definition 1:** For a graph \( G(V, E) \), the \( d \)-outgoing set of subsystem \( i \) is \( \text{out}_i(d) \) := \( \{ v_j \mid \text{dist}(v_i \rightarrow v_j) \leq d \in \mathbb{N} \} \). The \( d \)-incoming set of subsystem \( i \) is \( \text{in}_i(d) \) := \( \{ v_j \mid \text{dist}(v_j \rightarrow v_i) \leq d \in \mathbb{N} \} \). Note that \( v_i \in \text{out}_i(d) \cap \text{in}_i(d) \) for all \( d \geq 0 \), and \( \text{dist}(v_i \rightarrow v_j) \) denotes the distance between \( v_i \) and \( v_j \), i.e., the number of edges in the shortest path connecting subsystems \( i \) and \( j \).

Hence, we can enforce a \( d \)-local information exchange constraint on the MPC problem (2)—where the size of the local neighborhood \( d \) is a design parameter—by imposing that each subcontroller policy respects

\[
[u_t]_i = \gamma_t([x_{t+1,i}]_{j \in \text{in}_i(d)}, [u_{kt-1}]_j \in \text{in}_i(d), [A]_{j,k} \in \text{in}_i(d), [B]_{j,k} \in \text{in}_i(d))
\]

for all \( t = 0, \ldots, T \) and \( i = 1, \ldots, N \), where \( \gamma_t \) is a measurable function of its arguments. This means that the closed-loop control policy at subcontroller \( i \) can be computed using only states, control actions, and system models collected from \( d \)-hop incoming neighbors of subsystem \( i \) in the communication topology \( G(A,B) \). Given such an interconnection topology, suitable structural compatibility assumptions between the cost function and state, input, and information exchange constraints are necessary for both the synthesis and implementation of a localized control action at each subsystem.

**Assumption 1:** In formulation (2), objective function \( f_t \) is such that \( f_t(x, u) = \sum_{i=1}^{N} f_t([x], [u_j]) \) with \( j \in \text{in}_i(d) \) for local functions \( f_t \), and state constraint sets \( X_t \) are such that \( x \in X_t \) if and only if \( [x] \in \text{in}_i(d) \in X_t \) \( \forall i \) and \( t \in \{0, \ldots, T \} \) for local sets \( X_t \), and \( \text{idem} \) for \( u_0, u_t \).

Assumption 1 imposes that whenever two subsystems are coupled through either the constraints or the objective function, they must be within the \( d \)-local regions—\( d \)-incoming and \( d \)-outgoing sets—of one another. This is a natural assumption for large networks where couplings between subsystems tend to occur at a local scale. We will show that under these conditions, DLMPC allows for localized synthesis and implementation of a control action at each subsystem by imposing appropriate \( d \)-local structural constraints on the closed-loop system responses of the system. For the rest of this article, we focus on developing a distributed and localized algorithmic solution and defer the design of a terminal cost and set that provides theoretical guarantees to our companion paper [21].

III. **LOCALIZED MPC VIA SLS**

We introduce the SLS framework [19] and justify its utility in MPC problems. We show how SLS naturally allows for locality constraints [19] to be imposed on the system responses and corresponding controller implementation and discuss how state and input constraints can be imposed in the presence of disturbances by extending the results from [22], leading to the formulation of the distributed and localized MPC problem.

A. **Time-Domain SLS**

The following is adapted from [19, Sec. 2]. Consider the dynamics of system (1) and let \( u_t \) be a causal linear time-varying state-feedback controller, i.e., \( u_t = K_t(x_{t,0}, x_{t,1}, \ldots, x_{t,i}) \) where \( K_t \) is some linear map to be designed.\(^1\) Let \( Z \) be the block-downshift matrix,\(^2\) and define \( \hat{A} := \text{blkdiag}(A, \ldots, A) \) and \( \hat{B} := \text{blkdiag}(B, \ldots, B, 0) \). Using the signal (bold) notation, we can compactly write the closed-loop behavior of system (1) under the feedback law \( u = Kx \), over the horizon \( t = 0, \ldots, T \), which can be entirely characterized by the system responses \( \Phi_x \) and \( \Phi_u \)

\[
x = (I - Z(\hat{A} + \hat{B}K))^{-1}w =: \Phi_xw
\]

\[
u = K(I - Z(\hat{A} + \hat{B}K))^{-1}w =: \Phi_uw.
\]

Here, \( x, u, \) and \( w \) are the finite-horizon signals corresponding to state, control input, and disturbance, respectively. By convention, we define the disturbance to contain the initial condition, i.e., \( w = [x_0^T, u_0^T, \ldots, w_{T-1}^T]^T \).

The approach taken by SLS is to directly parameterize and optimize over the set of achievable system responses \( \{ \Phi_x, \Phi_u \} \) from the exogenous disturbance \( w \) to the state \( x \) and the control input \( u \), respectively.

**Theorem 1** (see [19, Th. 2.1]): For the system (1) evolving under the state-feedback policy \( u = Kx \), where \( K \) is block-lower-triangular, the following are true.

1) The affine subspace

\[
Z_{AB} \Phi := \left[ I - Z\hat{A} - Z\hat{B} \right] \begin{bmatrix} \Phi_x \\ \Phi_u \end{bmatrix} = I
\]

with lower triangular \( \{ \Phi_x, \Phi_u \} \) parameters all the possible system responses (4).

2) For any block lower-triangular matrices \( \{ \Phi_x, \Phi_u \} \) satisfying (5), the controller \( K = \Phi_u \Phi_x^{-1} \) achieves the desired response (4).

The SLS framework relies on part 1 of Theorem 1 to reformulate optimal control problems as a search over system responses \( \Phi \) lying in subspace (5), rather than an optimization problem over states and inputs \( \{x, u\} \). Using parameterization (4), we reformulate the MPC subroutine (2) in terms of the system responses as

\[
\min_{\Phi} f(\Phi(1)x_0)
\]

s.t. \( Z_{AB} \Phi = I, \ x_0 = x(\tau), \ \Phi w \in \mathcal{P} \forall w \in \mathcal{W} \)

where the polytope \( \mathcal{W} := \otimes_{t=0}^{T-1} \mathcal{W}_t \), and the polytope \( \mathcal{P} \) is defined so that \( \Phi w \in \mathcal{P} \) if and only if \( x_t \in X_t \) and \( u_t \in U_t \) for all \( t = 0, \ldots, T - 1 \). Notice that \( \mathcal{W} \) is defined so that it does not

\(^1\)Our assumption of a linear policy is without loss of generality, as an affine control policy \( u_t = K_t(x_{t,0} + v) \) can always be written as a linear policy acting on the homogenized state \( \bar{x} = [x; 1] \).

\(^2\)A matrix with identity matrices along its first block subdiagonal and zeros elsewhere.
restrict \( x_0 \). The objective function \( f \) is defined consistent with the objective function of problem (2). Note that Assumption 1 directly applies to the objective function and constraint set of the SLS reformulation (6). We emphasize that \( \Phi \{1\} x_0 \) appears in the objective function as it corresponds to the nominal (noise-free) state and input responses.

The equivalence between the MPC SLS problem (6) and the original MPC problem (2) stems from the well-known fact—restated in terms of the SLS parameterization—that linear time-varying controllers are as expressive as nonlinear controllers over a finite horizon, given a fixed initial condition and noise realization. In fact, for a fixed initial condition and noise realization \( w \), any control sequence \( u(w) := [u_0, \ldots, u_{T-1}]^T \) can be achieved by a suitable choice of feedback matrix \( K(w) \) such that \( u(w) = K(w)x \) (that such a matrix always exists follows from a dimension counting argument). As this control action can be achieved by a linear time-varying controller \( K(w) \). Theorem 1 states that there exists a corresponding achievable system response pair \( \{\Phi_x, \Phi_u\} \) such that \( u(w) = \Phi_u w \). Thus, the SLS reformulation introduces no conservatism relative to open-loop MPC. We discuss the closed-loop setting at the end of this section and show that the disturbance-based parameterization [12] is a special case of ours.

**Why use SLS for DMPC:** In the centralized setting, where both the system matrices \((A, B)\) and the system responses \( \{\Phi_x, \Phi_u\} \) are dense without information constraints, the SLS parameterized problem (6) is slightly more computationally costly than the original MPC problem (2), as there are now \( n(n+p)T \) decision variables, as opposed to \((n+p)T\) decision variables. However, under suitable localized structural assumptions on \( f_t \) and constraint sets \( X_1 \) and \( U_1 \), lifting to this higher dimensional parameterization makes the problem decomposable; it allows us to take advantage of the structure of the underlying system. This allows for not only the convex synthesis of a distributed closed-loop control policy (as is similarly done in [12] and [14]), but also for the solution of this convex synthesis problem to be computed using distributed optimization.

This latter feature is one of the main contributions of this article. In particular, we show that the resulting number of optimization variables in the local subproblems solved at each subsystem scales as \( O(d^2T) \), where \( d \) is the size of the neighborhood for each subsystem as per Definition 1 (usually \( d \ll n \)) and \( T \) is the time horizon; complexity is independent of the global system size \( n \). To the best of our knowledge, this is the first distributed closed-loop MPC algorithm with such properties.

### B. Locality in SLS

Here, we illustrate how to enforce the information sharing constraint (3) in the SLS framework and how localized system responses result in a localized controller implementation.

A key advantage of using the SLS framework is that the system responses not only parameterize the closed-loop map but also provide a controller realization. In particular, the controller achieving the system responses (4) can be implemented as

\[
\begin{align*}
    u &= \Phi_u \hat{w}, \quad \hat{x} = (I - \Phi_x)\hat{w}, \quad \hat{w} = x - \hat{x}
\end{align*}
\]

where \( \hat{x} \) is the nominal state trajectory, and \( \hat{w} = Zw \) is a delayed reconstruction of the disturbance. The advantage of this controller implementation, as opposed to \( u = \Phi_u \hat{x} \), is that any structure imposed on the system response \( \{\Phi_u, \Phi_x\} \) translates directly to structure on the controller implementation (7). This is particularly relevant for imposing locality constraints, and we will show how locality in system responses translates into the locality of the controller implementation.

We begin by defining the notion of \( d \)-localized system responses, which follows naturally from the notion of \( d \)-local information exchange constraints (3). They consist of system responses with suitable sparsity patterns such that the information exchange needed between subsystems to implement the controller realization (7) is limited to \( d \)-hop incoming and outgoing neighbors.

**Definition 2:** Let \( [\Phi_x]_{ij} \) be the submatrix of system response \( \Phi_x \) describing the map from disturbance \( [w]_j \) to the state \( [x]_i \) of subsystem \( i \). The map \( \Phi_x \) is \( d \)-localized if and only if for every subsystem \( i \), \( [\Phi_x]_{ij} = 0 \forall i \notin \text{out}(d) \). The definition for \( d \)-localized \( \Phi_u \) is analogous but with disturbance to control action \( [u]_i \).

When the system responses are \( d \)-localized, so is the controller implementation (7). In particular, by enforcing \( d \)-localized structure on \( \Phi_x \), only a local subset \( [\hat{w}]_{j \in \text{in}(d)} \) of \( \hat{w} \) is necessary for subsystem \( i \) to compute its local disturbance estimate \( [\hat{w}]_i \), which ultimately means that only local communication is required to reconstruct the relevant disturbances for each subsystem. Similarly, if the \( d \)-localized structure is imposed on \( \Phi_u \), then only a local subset \( [\hat{w}]_{j \in \text{in}(d)} \) of the estimated disturbances \( \hat{w} \) is needed for each subsystem to compute its control action \( [u]_i \). Hence, each subsystem only needs to collect information from its \( d \)-incoming set to implement the control law (7), and it only needs to share information with its \( d \)-outgoing set to allow for other subsystems to implement their respective control laws. Furthermore, such locality constraints are enforced via an affine subspace constraint in the SLS formulation (6).

**Definition 3:** A subspace \( L_d \subseteq L_d \) enforces a \( d \)-locality constraint if \( (\Phi_x, \Phi_u) \in L_d \) implies that \( \Phi_x \) is \( d \)-localized and \( \Phi_u \) is \( (d+1) \)-localized.\(^3\) A system \((A, B)\) is then \( d \)-localizable if the intersection of \( L_d \) with the affine space of achievable system responses (5) is nonempty.

**Remark 1:** Although \( d \)-locality constraints are always convex subspace constraints, not all systems are \( d \)-localizable. The locality diameter \( d \) can be viewed as a design parameter, and for the rest of this article, we assume that there exists a \( d \ll n \) such that the system \((A, B)\) to be controlled is \( d \)-localizable. Notice that the parameter \( d \) is tuned independently of the horizon \( T \) and captures how “far” in the interconnection topology a disturbance striking a subsystem is allowed to spread—as described in detail in [19], localized control is a spatiotemporal generalization of deadbeat control.

\(^3\)Notice that we are imposing \( \Phi_u \) to be \((d+1)\)-localized because in order to localize the effects of a disturbance within the region of size \( d \), the “boundary” controllers at distance \( d+1 \) must take action (for more details, the reader is referred to [19]).
While it was not possible to incorporate locality (3) into the classical MPC formulation (2) in a convex and computationally tractable manner, it is straightforward to do so via the affine constraint \((\Phi_x, \Phi_u) \in \mathcal{L}_d\) as per Definition 2, with the only requirement that some mild compatibility assumptions as per Assumption 1 between the cost functions, state and input constraints, and \(d\)-local information exchange constraints are satisfied. In the rest of this section, we exploit this fact when tackling robust state and input constraints, which, in turn, results in a problem structure that can be solved using distributed optimization. The resulting control policies are computed using only local information.

C. State and Input Constraints in SLS

Here, we extend the method to deal with robust state and input constraints. We emphasize that the resulting SLS reformulation retains the locality structure of the original problem, which is key to enabling a distributed solution to the problem.

As previously stated, in the noisy case, we restrict ourselves to polytopic constraints, such that

\[
P = \{(x^T u^T)^T : H(x^T u^T)^T \leq h\}
\]

where \(H := \text{blkdiag}(H_{x,1}, \ldots, H_{x,T}, H_{u,1}, \ldots, H_{u,T-1})\) and \(h := (h_{x,1}, \ldots, h_{x,T}, h_{u,1}, \ldots, h_{u,T-1})\). We assume that \(H\) and \(h\) are known. We consider two structures for the vector of the noise, which, to distinguish from the initial condition \(x_0\), we denote by \(\delta\), i.e., \(w = [x_0^T \delta^T]^T\).

1) Locally Norm-Bounded Disturbance: Definition 4: \(\mathcal{W}_\sigma\) is a separable local norm-bounded set if for every signal \(\delta \in \mathcal{W}_\sigma\), \(\|\delta\|_p \leq \sigma\) \(\forall\) \(i\) and \(p \in \mathbb{Z}_{\geq 1}\).

Remark 2: Note that \(\delta \in \mathcal{W}_\sigma\) if and only if \(\|\delta\|_\infty \leq \sigma\).  

Lemma 1: Let the noise signal belong to a separable local norm-bounded set \(\mathcal{W}_\sigma\). Then, problem (6) with additional localization constraints has the following convex reformulation:

\[
\min_{\Phi} \quad f(\Phi(1)x_0) \\
\text{subject to} \quad \sum_{i=1}^{T} \phi_i \Phi(1)[x_i \in \mathcal{W}_\sigma] + \sum_{j} \sigma \|e_j\|_T \|H \Phi(2:T)\|_2 \|e_j\|_\infty \leq \|h\|_i \quad \forall i
\]

where \(e_j\) are vectors of the standard basis and \(\|\cdot\|_\infty\) is the dual norm of \(\|\cdot\|_p\).

Proof: The proof follows from a simple duality argument on the robust polytopic constraint \(\Phi w \in P \forall w \in \mathcal{W}\). In particular, \(\Phi\) has to satisfy

\[
H \Phi(1)x_0 + \max_{\|\delta\|_p \leq \sigma} H \Phi(2:T) \delta \leq h
\]

where the maximization over \(\delta\) and the inequality are element-wise. Since \(\Phi\) is localized and the disturbance set is separable and local, this can be equivalently written as

\[
[H \Phi(1)](i)[x_0] + \max_{\|\delta\|_p \leq \sigma} [H \Phi(2:T)](i) \|\delta\|_\infty \leq \|h\|_i \quad \forall i
\]

And by definition of the dual norm, the second term on the left-hand side gives

\[
\sum_{j} \sigma \|e_j\|_T [H \Phi(2:T)](i) \|e_j\|_\infty \leq \|h\|_i \quad \forall i.
\]

Remark 3: Notice that the argument of Lemma 1 can be extended to any setting for which a closed-form expression for \(\sum_{j} \sigma \|e_j\|_T [H \Phi(2:T)](i) \|e_j\|_\infty \leq \|h\|_i \)

Moreover, since \(H\) is localized block-diagonal, and \(\Phi\) is localized, this constraint can be enforced using only local information.

2) Polytopic Disturbance: Lemma 2: Let the noise signal belong to a polytope, i.e., \(\delta \in \{\delta : G\delta \geq \delta\}\), where

\[
G := \text{blkdiag}(G_1, \ldots, G_T)\text{ and } g := (g_1, \ldots, g_T)
\]

and each of the \(\{G_1 \cdots G_T\}\) block-diagonal with structure compatible with subsystemwise decomposition of \(w\). Then, problem (6) with \(\Phi \in \mathcal{L}_d\) is reformulated as

\[
\min_{\Phi, \Xi \geq 0} \quad f(\Phi(1)x_0) \\
\text{subject to} \quad \sum_{i=1}^{T} \phi_i \Phi(1)[x_i \in \mathcal{W}_\sigma] + \sum_{j} \sigma \|e_j\|_T \|H \Phi(2:T)\|_2 \|e_j\|_\infty \leq \|h\|_i \quad \forall i
\]

where \(\Xi \geq 0\) satisfies \(\Xi g \leq \delta, H \Phi(2:T) = \Xi G\)

Proof: The reformulation of the robust polytopic constraint follows from duality. In particular, \(\Phi\) has to satisfy

\[
H \Phi(1)x_0 + \max_{\|G\delta\|_\infty \leq \|g\|} H \Phi(2:T) \delta \leq \|h\|_i.
\]

We study this constraint row-wise and focus on the second term on the left-hand side, which can be seen as a linear program (LP) in \(\delta\) for each row of \(H\). Since strong duality holds, we have that each of the LPs can equivalently be replaced by its corresponding dual problem. In particular, we can solve for the \(k\)th row of the second term on the left-hand side as

\[
\min_{\Xi(k,:) \geq 0} \quad \Xi(k,:)g \\
\text{subject to} \quad H(k,:) \Phi(2:T) = \Xi(k,:)G
\]

where the matrix operator \(\Xi\) results from stacking the dual variables from the row-wise dual LPs. Hence, the robust polytopic constraint can be replaced by

\[
\Xi(k,:)g \leq \|h\|_i, \quad H(k,:) \Phi(2:T) = \Xi G
\]

where \(\Xi \geq 0\) (satisfied componentwise) becomes a decision variable of the MPC problem. Furthermore, the constraint \(H(k,:) \Phi(2:T) = \Xi G\) allows for a sparse structure on \(\Xi\). In particular, \(\Xi G\) has to have the same sparsity as \(H(k,:) \Phi(2:T)\). When \(G\) is block-diagonal, it immediately follows that if \(\sum_{j} H(i,k)\Phi(2:T)(k,j) = 0\), then \(\Xi(i,j) = 0\). Hence, \(\Xi\) lies in \(\mathcal{L}_{dh}\) and, together with the dual reformulation of the robust polytopic constraint \(G\delta \geq g\), gives rise to problem (9).
**Remark 4:** By Assumption 1, the subspace $L_{d_{ht}}$ contains matrices with sparsity such that subsystems at most 2d-hops away are coupled.

These results allow us to solve the DLMPC problem (6) using standard convex optimization methods and further preserve the locality structure of the original problem under the given assumptions. Imposing $d$-local structure on the system responses, coupled with an assumption of compatible $d$-local structure on the objective functions and constraints of the MPC problem (2), leads to a structured SLS MPC optimization problem (6). This structural compatibility in all optimization variables, cost functions, and constraints is the key feature that we exploit in Section IV to apply distributed optimization techniques to scalably and exactly solve problem (6).

Why previous methods are not amenable to distributed solutions: While previous methods [12], [14] allow for similar structural constraints to be imposed on the controller realization through the use of either disturbance feedback or Youla parameterizations (subject to quadratic invariance [13] conditions), the resulting synthesis problems do not enjoy the structure needed for distributed optimization techniques to be effective. We focus on the method defined in [12], as a similar argument applies to the synthesis problem in [14]. Intuitively, the disturbance-based feedback parameterization of [12] only parameterizes the closed-loop map $\Phi_a$ from $w \rightarrow u$ and leaves the state $x$ as a free variable. Hence, regardless as to what structure is imposed on the objective functions, constraints, and the map $\Phi_a$, the resulting optimization problem is strongly and globally coupled because the state variable $x$ is always dense. This can be made explicit by noticing that the disturbance feedback parameterization of [12] can be recovered from the SLS parameterization of Theorem 1, and we present a formal derivation in the Appendixes. A similar coupling arises in the Youla-based parameterization suggested in [14]. This limits their usefulness to smaller scale examples where centralized computation of policies is feasible. In contrast, by explicitly parameterizing the additional system response $\Phi_x$ from $w \rightarrow x$, we can naturally enforce the structure needed for distributed optimization techniques to be fruitfully applied.

**IV. DISTRIBUTED AND LOCALIZED MPC BASED ON THE ADMM**

We now use the previous results to reformulate the DLMPC problem (6) in a way that is amenable to distributed optimization techniques and show that the ADMM [20] can be used to find a solution in a distributed manner. We exploit the separability (see Assumption 1), locality constraints, and the notion of row/columnwise separability (to be defined next), to solve each of the local subproblems in parallel and with $d$-local information only. In what follows, we restrict Assumption 1 to the case where only dynamical coupling is considered. In our companion work [21], we extend these results to all the cases considered in Assumption 1, i.e., constraints and objective functions that introduce $d$-localized couplings. Hence, all cost function and constraints have structure

$$f(x, u) = \sum_{i=1}^{N} f^i([x]_i, [u]_i),$$

where $[x]]_i \in \mathcal{P}^i$ if and only if $[x]_i \in \mathcal{P}^i$.

By the definition of the SLS system responses (4), we can equivalently write these conditions in terms of $\Phi$ as

$$f(\Phi(1)x_0) = \sum_{i=1}^{N} f^i(\Phi(1)(\tau_i,:)x_0),$$

and

$$\Phi \in \mathcal{P} \text{ if and only if } \Phi(\tau_i,:)x_0 \in \mathcal{P}^i \forall i$$

where $\tau_i$ is the set of rows in $\Phi$ corresponding to subsystem $i$, i.e., those that parameterize $[x]_i$ and $[u]_i$. These separability features are formalized as follows.

**Definition 5:** Given the partition $\{\tau_1, \ldots, \tau_k\}$, a functional/set is row-wise separable if:

1) for a functional $g_i, g_i(\Phi) = \sum_{i=1}^{k} g_i(\Phi(\tau_i,:))$ for some functions $g_i$ for $i = 1, \ldots, k$;

2) for a set $\mathcal{P}, \Phi \in \mathcal{P}$ if and only if $\Phi(\tau_i,:)x_0 \in \mathcal{P}^i \forall i$ for some sets $\mathcal{P}_i$ for $i = 1, \ldots, k$.

An analogous definition exists for columnwise separable functionals and sets [19], where the partition $\{\epsilon_1, \ldots, \epsilon_k\}$ entails the columns of $\Phi$, i.e., $\Phi(\epsilon_i, c_i)$.

When the objective function and all the constraints of an optimization problem are separable with respect to a partition of cardinality $k$, then the optimization trivially decomposes into $k$ independent subproblems. However, this is not the case for the localized DLMPC problem (6), since some elements are row-wise separable, while others are columnwise separable. To make it amenable to a distributed solution, we propose the following reformulation, which applies to the noise-free case as well as to both noisy cases (locally norm-bounded and polytopic) considered in the previous section:

$$\begin{align*}
\min_{\Phi} & \quad f(M_1 \tilde{\Phi}(1)x_0) \\
\text{s.t.} & \quad Z_{AB}M_2 \tilde{\Psi} = I, \quad x_0 = x(\tau), \quad \tilde{\Phi}, \tilde{\Psi} \in L_d \\
& \quad \tilde{\Phi} \in \tilde{\mathcal{P}}, \quad \tilde{\Psi} = \tilde{H} \tilde{\Psi}
\end{align*}$$

(10)

where:

1) In the noise-free case

$$\tilde{\Phi} := \Phi(1), \quad \tilde{\Psi} := \Phi(1), \quad M_1 = M_2 = \tilde{H} = : I,$$

$$\tilde{\mathcal{P}} = \{\Phi(1): \Phi(1)x_0 \in \mathcal{P}\}.$$

2) In the noisy case

$$M_1 := \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \quad M_2 := \begin{bmatrix} 0 & I \\ 0 & H \end{bmatrix}.$$
a) When noise is locally bounded
\[ \Phi := \begin{bmatrix} \Phi (1) & 0 \\ \Omega (1) & \Omega (2) : T \end{bmatrix}, \]
\[ \Psi := \begin{bmatrix} \Psi (1) & 0 \\ \Psi (1) & \Psi (2) : T \end{bmatrix}, \]
\[ \tilde{P} := \{ \Omega : [\Omega (1)]_{x_0} \}, \]
\[ + \sum_j \sigma \| \epsilon_j \|_2 [\Omega (2) : T]_{x_j} \leq [h]_{i \forall i}. \]

b) When noise is polytopic
\[ \Phi := \begin{bmatrix} \Phi (1) & 0 \\ \Omega (1) & \Xi G \end{bmatrix}, \]
\[ \Psi := \begin{bmatrix} \Psi (1) & 0 \\ \Psi (1) & \Psi (2) : T \end{bmatrix}, \]
\[ \tilde{P} := \{ \Omega, \Xi : [\Omega (1)]_{x_0} + \Xi g \leq h, \Xi \geq 0 \}. \]

The matrices \( \Phi \) and \( \Omega \) are simply duplicates of \( \Phi \) and \( \Xi \) is the dual variable as introduced in Lemma 2. The advantage of introducing these variables and creating the augmented variables \( \Phi \) and \( \Psi \) is that all the components of problem (10) involving \( \Phi \) are row-wise separable, and all the components involving \( \Psi \) are columnwise separable. We can easily separate these two computations via the ADMM using the relationship between both variables \( \bar{H} \Phi = \Psi \). Furthermore, we take advantage of the structure of these subproblems and separate them with respect to a row and column partition induced by the subsystemwise partitions of the state and control inputs, \( [x]_i \) and \( [u]_i \), for each subsystem \( i \). Each of these rows and column subproblems resulting from the distribution across subsystems can be solved independently and in parallel, where each subsystem solves for its corresponding row and column partition. Moreover, since locality constraints are imposed, the decision variables \( \Phi \) and \( \Psi \) have a sparse structure. This means that the length of the rows and columns that a subsystem solves for is much smaller than the length of the rows and columns of \( \Phi \). For instance, when considering the columnwise subproblem evaluated at subsystem \( i \), the \( j \)th row of the \( j \)th column partitions of \( \Phi_x \) and \( \Phi_u \) are nonzero only if \( j \in \cup_{k \in \text{out}(d)} \mathcal{R}_k \) and \( j \in \cup_{k \in \text{out}(d+1)} \mathcal{R}_k \), respectively.\(^5\) Thus, the subproblems that subsystem \( i \) solves for are
\[ [\Phi]_{n+1} = \left\{ \arg \min_{\Phi} f([M]_i \Phi_i [x]_i) + \frac{\rho}{2} g_{R}(\Phi, \Psi, \Lambda) \right\} \]
\[ \text{s.t. } [\Phi]_{i} \in \tilde{P} \cap \mathcal{L}_d, [x]_{i} = [x(t)]_{i} \]
(11a)
\[ [\Psi]_{n+1} = \left\{ \arg \min_{\Psi} g_{B}(\Psi, \Phi, \Lambda) \right\} \]
\[ \text{s.t. } [Z]_{AB} [M]_i [\Psi]_{i} = [I]_{i} \]
(11b)
\[ [\Lambda]_{n+1} = [\Phi]_{n+1} - [\bar{H}]_{i} [\Psi]_{n+1} + [\Lambda]_{n} \]
(11c)

\(^5\)An algorithm to find the relevant components for each subsystem rows and columns can be found in [19, Appendix A].

**Algorithm 1:** Subsystem \( i \) DLMPC Implementation.

**Input:** \( \epsilon_p, \epsilon_d, \rho > 0 \).

1. Measure local state \( [x(t)]_i, k \leftarrow 0 \).
2. Share the measurement with neighbors in \( \text{out}(d) \). Receive the corresponding \( [x(t)]_j \) from \( \text{in}(d) \) and build \( [x_0]_i \).
3. Solve optimization problem (11a).
4. Share \( [\Phi]_{n+1} \) with \( \text{out}(d) \). Receive the corresponding \( [\Phi]_{n+1} \) from \( \text{in}(d) \) and build \( [\Phi]_{n+1} \).
5. Solve problem (11b) via the closed form (12).
6. Share \( [\Psi]_{n+1} \) with \( \text{out}(d) \). Receive the corresponding \( [\Psi]_{n+1} \) from \( \text{in}(d) \) and build \( [\Psi]_{n+1} \).
7. Perform the multiplier update step (11c).
8. if \( \| [\Phi]_{n+1} - [\bar{H}]_{i} [\Psi]_{n+1} \|_F \leq \epsilon_p \) and \( \| [\Lambda]_{n+1} \|_F \leq \epsilon_d \) apply control action \( [u]_i = [\Phi_{i,0}]_{i} [x_0]_i \), and return to step 1.
   else:
   Set \( k \leftarrow k + 1 \), return to step 3.

where the scalar \( \rho \) is the ADMM multiplier, operator \( \Lambda \) is the dual variable associated with the ADMM, and
\[ g_{R}(\Phi, \Psi, \Lambda) = \| [\Phi]_{i} - [\bar{H}]_{i} [\Psi]_{i} + [\Lambda]_{i} \|_F^2. \]

**Computational complexity of Algorithm 1:** The complexity of the algorithm is determined by update steps 3, 5, and 7. In

\(^6\)Predicted control actions can be computed as \( [u]_i = [\Phi_{i,t}]_{0} [x_0]_i \) for \( t \in [1, T - 1] \).
particular, steps 5 and 7 can be directly solved in closed form, reducing their evaluation to the multiplication of matrices of dimension $O(d^2 T^2)$ in the noisy case and $O(d^2 T)$ in the noise-free case. In general, step 3 requires an optimization solver, where each local iterate subproblem is over $O(d^2 T^2)$ optimization variables in the noisy case, and $O(d^2 T)$ in the noise-free case, subject to $O(d T)$ constraints. In certain cases, step 3 can also be computed in closed form if a proximal operator exists for the formulation [20]. This is true, for instance, if step 3 reduces to a quadratic convex cost function subject to affine equality constraints, in which case complexity reduces to $O(dP T)$ since piecewise closed-form solutions can be computed [23]. Notice that the complexity of the subproblems is largely dependent on whether noise is considered. The reason for this is that in the noise-free case, only the first block column of $\Phi$ is considered, whereas in the presence of noise, all $T$ block columns must be computed. Regardless of this, the use of locality constraints leads to a significant computational saving when $d < N$. The communication complexity—as determined by steps 2, 4, and 6—is limited to the local exchange of information between $d$-local neighbors.

V. SIMULATION EXPERIMENTS

We now apply the DLMPC algorithm to a power-system-inspired example. After introducing the simplified model, we present simulations\(^7\) under different noise realizations and validate the algorithm correctness and optimal performance by comparing to a centralized algorithm, as well as its ability to achieve constraint satisfaction in the presence of noise. We further demonstrate the scalability of the proposed method by verifying different network and problem parameters (locality, network size, and time horizon) and, in particular, show that runtime stays steady as the network size increases.

A. System Model

We begin with a 2-D square mesh, where we randomly determine whether each node connects to each of its neighbors with a 40% probability. The expected number of edges is $0.8 * n * (n - 1)$. Each node represents a two-state subsystem that follows linearized and discretized swing dynamics

$$
\begin{bmatrix}
\dot{\theta}(t + 1) \\
\dot{\omega}(t + 1)
\end{bmatrix}_i = \sum_{j \in \text{in}_i(1)} [A]_{ij} \begin{bmatrix}
\theta(t) \\
\omega(t)
\end{bmatrix}_j + [B]_i [u]_i + [w]_i,
$$

where $[\theta]_i$, $[\dot{\theta}]_i$, and $[u]_i$ are the phase angle deviation, frequency deviation, and control action of the controllable load of bus $i$.

The dynamic matrices are

$$
[A]_{ij} = \begin{bmatrix} 1 & \Delta t \frac{\Delta t}{m_i} \\ -\frac{k_i}{m_i} \Delta t & 1 - \frac{d_i}{m_i} \Delta t \end{bmatrix},
[B]_{ij} = \begin{bmatrix} 0 & 0 \\ \frac{k_{ij}}{m_i} \Delta t & 0 \end{bmatrix}
$$

and $[B]_i = [0 \ 1]^T$ for all $i$. Parameters in bus $i$: $m_i^{-1}$ (inertia inverse), $d_i$ (damping), and $k_{ij}$ (coupling) are randomly generated and uniformly distributed between $[0, 2]$, $[0.5, 1]$, and $[1, 1.5]$, respectively. The discretization step is $\Delta t = 0.2$, and $k_i := \sum_{j \in \text{in}_i(1)} k_{ij}$.

In simulations, we optimize a quadratic cost $f_i(x_t, u_t) = x_t^T \Phi x_t + u_t^T u_t$ for all $t$ and start with a randomly generated initial condition. We study three noise scenarios: noise-free, polytopic noise, and locally bounded noise. Noise follows a uniform distribution, and in the locally bounded case, it is scaled appropriately to meet the local bounds within each $d$-local neighborhood (see code for details). The baseline parameter values are $d = 3$, $T = 5$, and $N = 16 (4 \times 4$ grid).

B. Optimal Performance

We observe the trajectories of the closed-loop system when using DLMPC for the different noise realizations. We compare these results with the solution to the corresponding centralized MPC problem (using CVX [24]). For a randomly chosen initial condition and network topology, we plot the evolution of the two states of subsystem 4 under the different noise conditions. We use the following constraints for all $i$ and all $t$: $[\theta]_i \in [-0.3, 0.3]$ in the noiseless case, $[\theta]_i \in [-10, 10]$ and $[\omega]_i \in [-0.5, 1.5]$ for locally bounded noise, and $[\theta]_i \in [-3.2, 3.2]$ for polytopic noise.

Results from simulations are summarized in Fig. 1. In all the cases, the centralized solution coincides with the solution achieved by the DLMPC algorithm, validating the optimality of the proposed algorithm. It is worth noting that in the absence of noise, both nominal and robust DLMPC yield the same result, illustrating that the robust formulation of DLMPC is a generalization of the nominal case, but the former is more computationally efficient. In the noisy cases, nominal DLMPC violates state bounds, illustrating the need for a robust approach. In general, robust DLMPC can introduce conservatism, as it anticipates worst-case disturbances; this is more apparent for locally bounded noise than for polytopic noise. This is because there always exists a worst-case polytopic noise realization, but there does not always exist a worst-case locally bounded noise realization. Different $d$-local neighborhoods overlap and are sometimes strict subsets of other neighborhoods; though the algorithm assumes a worst-case local disturbance at each neighborhood, the worst-case disturbance at some node $i$ may be different for each $d$-local neighborhood containing $i$. Thus, there may be no noise realization that is worst case for every $d$-local neighborhood simultaneously. Overall, locally bounded noise formulations inherently introduce some conservatism, as they anticipate a worst case that is generally mathematically impossible.

\(^7\)Code needed to replicate these experiments is available at https://github.com/unstable-zeros/dl-mpc-sls; this code makes use of the SLS toolbox at https://github.com/sls-caltech/sls-code, which includes ready-to-use MATLAB implementations of all the algorithms presented in this article and its companion paper [21].
Average runtime per DLMPC iteration with network size (left), to 1 s—where runtime does significantly increase for the robust cases and as such that acceptable per-
can lead to improved

d for the states of subsystem 4 under a nominal (green) and a robust (purple) controller. Red-dashed line indicates upper or lower bound, respectively. Both DLMPC and centralized MPC controllers yield the same result. In the absence of noise, both the nominal and robust controllers lead to the same trajectory, whereas in the presence of noise (both locally bounded and polytopic), the nominal controller leads to a violation of state bounds.

C. Computational Complexity

To assess the scalability of the algorithm, we measure runtime\(^8\) while varying different network and problem parameters: locality \(d\), network size \(N\),\(^9\) and time horizon \(T\). We run five different simulations for each of the parameter combinations, using different realizations of the randomly chosen parameters as to provide a consistent estimation of runtimes. In this case, constraints are \(\theta_i \in [-4, 4]\) and \(\omega_i \in [-20, 20]\) across all the scenarios.

We study the scalability of the DLMPC algorithm for each of the different computation strategies presented: noiseless, locally bounded noise, and polytopic noise. For the sake of comparison, we include an additional computation strategy based on explicit MPC that reduces the overhead in the noiseless case by replacing the optimization solver by a piecewise solution. Interested readers are referred to [23] for details. We observe the behavior for these four different strategies in Fig. 2, and we note that the computation strategy determines the order of magnitude of the runtime, ranging from \(10^{-3}\) to \(10^{-2}\) s in the noiseless case—depending on whether or not explicit solutions are used—to \(10^{-1}\) s in the locally-bounded case, and around 1–10 s when polytopic noise is considered. This difference is expected and is explained by the size of the decision variables across different scenarios, i.e., \(\Phi\) has a much larger dimension in the polytopic case than in the noiseless case. Despite this difference in order of magnitude, the trends observed are the same for each of the different scenarios and radically different from that observed when using a centralized solver—ranging from \(10^{-2}\) to 1 s—where runtime does significantly increase with the size of the network. In contrast, when using Algorithm 1, runtime barely increases with the size of the network, and the slight increase in runtime—likely due to the larger coordination needed—does not seem to be significant and appears to level off for sufficiently large networks. These observations are consistent with those of [25], where the same trend was noted. In contrast, runtime appears to increase with time horizon and more notably with locality region size. This is also expected, as according to our complexity analysis the number of variables in a DLMPC subproblem scales as \(O(d^2 T^2)\) for the robust cases and as \(O(d^2 T)\) for the nominal case. It is well known that a larger time horizon, while providing an improvement in performance, can be detrimental for computational overhead. The same is true for locality size, which is much smaller than the size of the network. Although a larger localized region \(d\) can lead to improved performance, as a broader set of subsystems can coordinate their actions directly, it also leads to an increase in computational complexity and communication overhead. Thus, by choosing the smallest localization parameter \(d\) such that acceptable performance is achieved, the designer can tradeoff between computational complexity and closed-loop performance. This further highlights the importance of exploiting the underlying structure of the dynamics, which allow us to enforce locality constraints on the system responses and, consequently, on the controller implementation.

VI. Conclusion

In this article, we defined and analyzed a closed-loop distributed and localized MPC algorithm. By leveraging the SLS framework, we were able to enforce information exchange constraints by imposing locality constraints on the system responses. We further showed that when locality is combined with mild assumptions on the separability structure of the objective functions and constraints of the problem, an ADMM-based
solution to the DLMPC subproblems can be implemented that requires only local information exchange and system models, making the approach suitable for large-scale distributed systems. This is the first DMPC algorithm that allows for the distributed synthesis of closed-loop policies. In our companion paper [21], we provide recursive feasibility and asymptotic stability guarantees, as well as scalable synthesis and implementations for terminal set and cost.

Given that the presented algorithm relies on multiple exchanges of information between the subsystems, how communication loss affects the closed-loop performance of the algorithm is an interesting question. Although a formal analysis is left as future research, the work done in [26] suggests that it would be possible to slightly modify the proposed ADMM-based scheme to make it robust to unreliable communication links. Moreover, only two types of noise were explored in this article. Results in polytopic containment [27] could enable a convex DLMPC reformulation of other noise scenarios. Finally, it is of interest to extend these results to information exchange topologies defined with both sparsity and delays—while the SLS framework naturally allows for delay to be imposed on the implementation structure of a distributed controller, it is less clear how to incorporate such constraints in a distributed optimization scheme.

**APPENDIX**

**A. SLS-Based MPC and Disturbance-Based Feedback Parameterizations**

**Proposition 1:** The disturbance-based parameterization \((M, v)\) defined in [12, Sec. 4] as \(u = Mw + v\) is a special case of the SLS parameterization (4).

**Proof:** We start with the statement from Theorem 1 and in particular affine constraint (5). Multiplying this constraint by \(w\) on the right, we obtain

\[
\begin{bmatrix}
I - Z\hat{A} - Z\hat{B}
\end{bmatrix}
\begin{bmatrix}
\Phi_x \\
\Phi_u
\end{bmatrix}w = w
\]

where by definition of \(\Phi_x, x := \Phi_x w\). Hence, by [19, Th. 2.1], the SLS MPC subproblem

\[
\begin{align*}
\min_{x, \Phi_x} & \quad f(x, \Phi_x, w) \\
\text{s.t.} & \quad (I - Z\hat{A})x = Z\hat{B}\Phi_x w + w, \quad x_0 = x(t) \in X, \quad \Phi_x w \in U \quad \forall w \in W
\end{align*}
\]

is equivalent to problem (2) when restricted to solving over linear time-varying feedback policies. Now, if we consider nominal (disturbance free) cost, i.e., \(w = [x_0^T, 0, \ldots, 0]^T\), the problem becomes

\[
\begin{align*}
\min_{x, \Phi_x} & \quad f(x, \Phi_x, 1, x_0) \\
\text{s.t.} & \quad (I - Z\hat{A})x = Z\hat{B}\Phi_x w + w, \quad x_0 = x(t) \in X, \quad \Phi_x w \in U \quad \forall w \in W.
\end{align*}
\]

Notice that by setting \(M = \Phi_u\) and \(v = w\), we recover the optimization problem over disturbance feedback policies \(u = Mw + v\) suggested in [12, Sec. 4].

An equivalent derivation arises in the Youla-based parameterization suggested in [14].

**B. Closed-Form Solutions for ADMM Iterates**

**Lemma 3:** Let \(z^*(M, v, P, q) := \arg\min_z ||Mz - v||^2_F\) s.t. \(Pz = q\). Then

\[
\begin{bmatrix}
\mu^* \\
\mu^*
\end{bmatrix} = \begin{bmatrix}
MAB & PT \\
0 & P
\end{bmatrix}^\dagger \begin{bmatrix}
MABv \\
q
\end{bmatrix}
\]

where \(\dagger\) denotes pseudoinverse and is the optimal solution and \(\mu^*\) is the corresponding optimal Lagrange multiplier.

**Proof:** The proof follows from applying the Karush–Kuhn–Tucker conditions to the optimization problem. By the stationarity condition, \(MABz^* - MBAv + PT\mu^* = 0\), where \(z^*\) is the solution to the optimization problem and \(\mu^*\) is the optimal Lagrange multiplier vector. From the primal feasibility condition, \(Pz^* = q\). Hence, \(z^*\) and \(\mu^*\) are computed as the solution to this system of two equations. \(\square\)

**ACKNOWLEDGMENT**

This article contains full proofs of all results, expands the cases considered in the robust setting, and provides a general formulation for the DLMPC framework. It also briefly presents and appropriately references additional work on explicit solutions.

**REFERENCES**

[1] C. Amo Alonso and N. Matni, “Distributed and localized closed-loop model predictive control via system level synthesis,” in Proc. IEEE 59th Conf. Decis. Control, 2020, pp. 5598–5605.

[2] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright, “Distributed MPC strategies with application to power system automatic generation control,” IEEE Trans. Control Syst. Technol., vol. 16, no. 6, pp. 1192–1206, Nov. 2008.

[3] Y. Zheng, S. Li, and H. Qiu, “Networked coordination-based distributed model predictive control for large-scale system,” IEEE Trans. Control Syst. Technol., vol. 21, no. 3, pp. 991–998, May 2013.

[4] P. Giselsson, M. D. Doan, T. Kezic, B. D. Schutter, and A. Rantzer, “Accelerated gradient methods and dual decomposition in distributed model predictive control,” Automatica, vol. 49, no. 5, pp. 829–833, 2013.

[5] C. Conte, C. N. Jones, M. Morari, and M. N. Zeilinger, “Distributed synthesis and stability of cooperative distributed model predictive control for linear systems,” Automatica, vol. 69, pp. 117–125, 2016.

[6] R. E. Jalal and B. P. Rasmussen, “Limited-communication distributed model predictive control for coupled and constrained subsystems,” IEEE Trans. Control Syst. Technol., vol. 25, no. 5, pp. 1807–1815, Sep. 2017.

[7] Z. Wang and C.-J. Ong, “Distributed MPC of constrained linear systems with online decoupling of the terminal constraint,” in Proc. IEEE Amer. Control Conf., 2015, pp. 2942–2947.

[8] A. N. Venkat, J. B. Rawlings, and S. J. Wright, “Stability and optimality of distributed model predictive control,” in Proc. IEEE 44th Conf. Decis. Control, 2005, pp. 6680–6685.

[9] Y. R. Sturz, E. L. Zhu, U. Rosolia, K. H. Johansson, and F. Borrelli, “Distributed learning model predictive control for linear systems,” in Proc. IEEE Conf. Decis. Control, 2020, pp. 4366–4373.

[10] D. Limon Martueado, T. Alamo, and E. F. Camacho, “Stability analysis of systems with bounded additive uncertainties based on invariant sets: Stability and feasibility of MPC,” in Proc. IEEE Amer. Control Conf., 2002, pp. 364–369.

[11] G. Grimm, M. J. Messina, S. E. Tuna, and A. R. Teel, “Examples when nonlinear model predictive control is nonrobust,” Automatica, vol. 40, no. 10, pp. 1729–1738, 2004.

[12] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, “Optimization over state feedback policies for robust control with constraints,” Automatica, vol. 42, no. 4, pp. 523–533, 2006.
[13] M. Rotkowitz and S. Lall, “A characterization of convex problems in decentralized control,” IEEE Trans. Autom. Control, vol. 51, no. 2, pp. 274–286, Feb. 2006.

[14] L. Furieri and M. Kamgarpour, “Robust control of constrained systems given an information structure,” in Proc. IEEE Conf. Decis. Control, 2017, pp. 3481–3486.

[15] J. Sieber, S. Bennani, and M. N. Zeilinger, “A system level approach to tube-based model predictive control,” IEEE Control Syst. Lett., vol. 6, pp. 776–781, 2022, doi: 10.1109/LCSYS.2021.3086190.

[16] C. Conte, M. N. Zeilinger, M. Morari, and C. N. Jones, “Robust distributed model predictive control of linear systems,” in Proc. IEEE Eur. Control Conf., 2013, pp. 2764–2769.

[17] V. Blondel and J. N. Tsitsiklis, “NP-hardness of some linear control design problems,” SIAM J. Control Optim., vol. 35, no. 6, pp. 2118–2127, 1997.

[18] A. Richards and J. P. How, “Robust distributed model predictive control,” Int. J. Control, vol. 80, no. 9, pp. 1517–1531, 2007.

[19] J. Anderson, J. C. Doyle, S. H. Low, and N. Matni, “System level synthesis,” Annu. Rev. Control, vol. 47, pp. 364–393, 2019.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Found. Trends Mach. Learn., vol. 3, no. 1, pp. 1–122, 2010.

[21] C. Alonso Alonso, J. S. Li, J. Anderson, and N. Matni, “Distributed and localized model predictive control—Part II: Theoretical guarantees,” 2022, arXiv:2203.00780.

[22] Y. Chen and J. Anderson, “System level synthesis with state and input constraints,” in Proc. IEEE Conf. Decis. Control, 2019, pp. 5258–5263.

[23] C. Alonso Alonso, N. Matni, and J. Anderson, “Explicit distributed and localized model predictive control via system level synthesis,” in Proc. IEEE Conf. Decis. Control, 2020, pp. 5606–5613.

[24] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” 2014. Available: http://cvxr.com/cvx

[25] C. Conte, T. Summers, M. N. Zeilinger, M. Morari, and C. N. Jones, “Computational aspects of distributed optimization in model predictive control,” in Proc. IEEE Conf. Decis. Control, 2012, pp. 6819–6824.

[26] Q. Li, B. Kailkhura, R. Goldhahn, P. Ray, and P. K. Varshney, “Robust decentralized learning using ADMM with unreliable agents,” IEEE Trans. Signal Process., vol. 70, pp. 2743–2757, 2022, doi: 10.1109/TSP.2022.3178655.

[27] S. Sadraddini and R. Tedrake, “Linear encodings for polytope containment problems,” in Proc. IEEE Conf. Decis. Control, 2019, pp. 4367–4372.

Carmen Amo Alonso received the B.Sc. degree in aerospace engineering from the Polytechnic University of Madrid, Madrid, Spain, in 2016, and the M.Sc. degree in space engineering in 2017 from the California Institute of Technology, Pasadena, CA, USA, where she is currently working toward the Ph.D. degree in control and dynamical systems.

She is also an AI4Science Amazon/Caltech Fellow. Her research interests include distributed optimal control, convex optimization, and parallel programming.

Jing Shuang Li received the B.A.Sc. degree in engineering science from the University of Toronto, Toronto, ON, Canada, in 2018. She is currently working toward the Ph.D. degree in control and dynamical systems with the California Institute of Technology, Pasadena, CA, USA. Her Ph.D. research was supported by a postgraduate scholarship from the Natural Sciences and Engineering Research Council of Canada. Her research interests include distributed control and its applications to biological modeling.

James Anderson received the D.Phil. (Ph.D.) degree in engineering science from the University of Oxford, Oxford, U.K., in 2012. He is currently an Assistant Professor with the Department of Electrical Engineering, Columbia University, New York, NY, USA, where he is also a member of the Data Science Institute. From 2016 to 2019, he was a Senior Postdoctoral Scholar with the Computing + Mathematical Sciences Department, California Institute of Technology, Pasadena, CA, USA. Prior to this, he was a Junior Research Fellow with the University of Oxford. His research interests include optimal and distributed control, convex optimization, and scientific computing with big data.

Nikolai Matni received the Ph.D. degree in control and dynamical systems from the California Institute of Technology (Caltech), Pasadena, CA, USA, in 2016. He is currently an Assistant Professor with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA. Prior to this, he was a Postdoctoral Scholar with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA. He was also a Postdoctoral Scholar with the Computing + Mathematical Sciences Department, Caltech.

Dr. Matni was a recipient of the National Science Foundation CAREER Award in 2021, a Google Research Scholar Award in 2021, and the IEEE CSS George S. Axelby Award in 2021.