Distributed and Localized Model Predictive Control. Part II: Theoretical Guarantees

Carmen Amo Alonso, Jing Shuang (Lisa) Li, Nikolai Matni, and James Anderson[*]

January 3, 2023

Abstract

Engineered cyberphysical systems are growing increasingly large and complex. These systems require scalable controllers that robustly satisfy state and input constraints in the presence of additive noise – such controllers should also be accompanied by theoretical guarantees on feasibility and stability. In our companion paper, we introduced Distributed and Localized Model Predictive Control (DLMPC) for large-scale linear systems; DLMPC is a scalable *closed-loop* MPC scheme in which subsystems need only exchange local information in order to synthesize and implement local controllers. In this paper, we provide recursive feasibility and asymptotic stability guarantees for DLMPC. We leverage the System Level Synthesis framework to express the maximal positive robust invariant set for the closed-loop system and its corresponding Lyapunov function, both in terms of the closed-loop system responses. We use the invariant set as the terminal set for DLMPC, and show that this guarantees feasibility with minimal conservatism. We use the Lyapunov function as the terminal cost, and show that this guarantees stability. We provide fully distributed and localized algorithms to compute the terminal set offline, and also provide necessary additions to the online DLMPC algorithm to accommodate coupled terminal constraint and cost. In all algorithms, only local information exchanges are necessary, and computational complexity is independent of the global system size – we demonstrate this analytically and experimentally. This is the first distributed MPC approach that provides minimally conservative yet fully distributed guarantees for recursive feasibility and asymptotic stability, for both nominal and robust settings.

1 Introduction

Model Predictive Control (MPC) enjoys widespread success across diverse applications. Ensuring recursive feasibility and asymptoptic stability for MPC is a well-studied topic in the centralized setting [1], and sufficient conditions based on terminal sets and cost functions have been established [2]. Porting these ideas to distributed systems is a challenging task, both theoretically and computationally. High computational demand, limited and local communication, and coupling among subsystems prevents the use of techniques from the centralized setting. Thus, efforts have been made to develop theoretical guarantees for distributed MPC.

Prior work: The majority of distributed MPC approaches rely on the use of distributed terminal costs and terminal sets to provide theoretical guarantees. In order to obtain structure in the terminal cost, standard methods rely on Lyapunov stability results, often combined with relaxation techniques to make them

---

[*]C. Amo Alonso and J.S. Li are with the Computing and Mathematical Sciences Department at California Institute of Technology, Pasadena, CA, 91106, USA. N. Matni is with the Department of Electrical and Systems Engineering at the University of Pennsylvania, Philadelphia, PA, 19104, USA. J. Anderson is with the Department of Electrical Engineering and the Data Science Institute at Columbia University, New York, NY, 10027, USA. `{camoalon, jsli}@caltech.edu`, `nmatni@seas.upenn.edu`, `james.anderson@columbia.edu`

amenable to distributed settings [3–5]. For terminal sets, proposed distributed methods are often limited by the coupling among subsystems – this often leads to small terminal sets that result in too conservative solutions (see for example [6, 7] and references therein). In order to overcome these issues, several approaches have recently been proposed to synthesize structured terminal sets with adaptive properties, i.e. local terminal sets defined as the sub-level set of a structured Lyapunov function, which change at each iteration in order to avoid unnecessary conservatism [8–13]. These approaches successfully design structured robust positive invariant sets and reduce conservatism; however, they require online updates of the terminal set at each MPC iteration, which increase the controller's overall computational complexity and communication overhead. Moreover, the imposed structure unavoidably leads to a possibly small approximation of the maximal control invariant set (the least conservative option for a terminal set). To move away from structured sets and costs, a data-driven approach was recently developed in [14], where locally collected data are used to construct local control invariant sets and costs that provide guarantees. However, this approach is mainly limited to iterative control tasks, and conservatism of the terminal cost and set only reduces asymptotically as the system collects data. Online computation and refinements of the terminal set are also key to this approach.

Given the state of the art, our goal is to design a distributed MPC approach with *distributed* and *minimally conservative* feasibility and stability guarantees. We seek a distributed MPC algorithm with (i) a maximal positive invariant terminal set, and (ii) a fully distributed and scalable *offline* algorithm to compute this set. This will allow us to use the associated Lyapunov function of the terminal set as the terminal cost, and requires no explicit a priori structural assumptions. No method satisfying these requirement currently exists in the literature.

Contributions: We provide theoretical guarantees for the Distributed Localized MPC (DLMPC) for linear time-invariant systems approach presented in our companion paper [15]. We show that the *maximal* positive invariant set of the closed-loop system can be expressed in terms of the closed-loop system responses as defined in the System Level Synthesis (SLS) framework [16, 17]. We show that when the closed-loop system is localized, the set is naturally structured without requiring additional assumptions. We also show that this set can be used to provide recursive feasibility guarantees when used as the terminal set of the system, and stability guarantees when combined with its associated global Lyapunov function [18]. We provide a fully distributed and localized offline algorithm for computation of the terminal set – this algorithm requires only local information exchange between subsystems. We also provide necessary additions to the original DLMPC algorithm to accommodate coupled terminal constraint and cost. In particular, this can be done by using a nested Alternating Direction Method of Multipliers (ADMM)-based consensus algorithm. In the resulting implementation, each sub-controller first solves for its local portion of the terminal set, offline, then solves a local online MPC problem. Throughout all algorithms, only local information exchanges within some local neighborhood are necessary, and computational complexity is independent of the global system size. The presented approach applies to the nominal case as well as additive polytopic or locally norm-bounded disturbances. This approach is the first to compute without approximation the maximal positive invariant set and its associated global Lyapunov function in a fully distributed and localized manner. Through numerical experiments, we validate these results and further confirm the minimal conservatism introduced by this method.

Paper structure: In §II we present the problem formulation and briefly summarize essential concepts from our companion paper [15]. In §III, we formulate the maximal robust positive invariant set and its associated Lyapunov function in closed-loop coordinates, and use this to provide recursive feasibility and stability guarantees for DLMPC; we also discuss convergence guarantees for DLMPC. In §IV we provide an offline algorithm to distributedly and locally compute the terminal set, and a modified online DLMPC algorithm that accommodates the terminal set and cost, including any local coupling. In §V, we present a numerical study and we end in §VI with conclusions and directions of future work.

Notation: Lower-case and upper-case Latin and Greek letters such as x and A denote vectors and

matrices respectively, although lower-case letters might also be used for scalars or functions (the distinction will be apparent from the context). Bracketed indices denote time-step of the real system, i.e., the system input is $u(t)$ at time t, not to be confused with x_t which denotes the predicted state x at time t. Superscripted variables, e.g. x^k, correspond to the value of x at the kth iteration of a given algorithm. 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 $\mathfrak{c}$ denote a subset of $\mathbb{Z}^+$, e.g. $\mathfrak{c} = \{1, ..., n\} \subset \mathbb{Z}^+$. Boldface lower and upper case letters such as $\mathbf{x}$ and $\mathbf{K}$ denote finite horizon signals and block lower triangular (causal) operators, respectively:

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

where each x_i is an n-dimensional vector, and each $K_i[j]$ is a matrix of compatible dimension representing the value of K at the jth time-step computed at time i. $\mathbf{K}(\mathfrak{r}, \mathfrak{c})$ denotes the submatrix of $\mathbf{K}$ composed of the rows and columns specified by $\mathfrak{r}$ and $\mathfrak{c}$ respectively. We denote the block columns of $\mathbf{K}$ by $\mathbf{K}\{1\}$,...,$\mathbf{K}\{T\}$, i.e. $\mathbf{K}\{1\} := [K_0[0]^\mathsf{T} \ \dots \ K_T[T]^\mathsf{T}]^\mathsf{T}$, and we use : to indicate the range of columns, i.e. $\mathbf{K}\{2 : T\}$ contains the block columns from the second to the last. For compactness, we also define $Z_{AB} := \begin{bmatrix} I - Z\hat{A} & -Z\hat{B} \end{bmatrix}$ where $\hat{A} := \text{blkdiag}(A, ..., A)$ and $\hat{B} := \text{blkdiag}(B, ..., B, 0)$ for the dynamics matrices A and B, and Z is the block-downshift matrix.

2 Problem Formulation

We begin with a brief summary of the DLMPC formulation, approach, and algorithm (for details, refer to our companion paper [15]), then formally introduce the problem of providing theoretical guarantees – recursive feasibility and asymptotic stability – in the DLMPC scheme. We then introduce the open question that we resolve in this paper.

Setup: Consider a discrete-time linear time invariant (LTI) system

$$
x(t + 1) = Ax(t) + Bu(t) + w(t), \tag{1}
$$

where $x(t) \in \mathbb{R}^n$ is the state, $u(t) \in \mathbb{R}^p$ is the control input, and $w(t) \in \mathcal{W} \subset \mathbb{R}^n$ is an exogenous disturbance at time t. System (1) can be interpreted as N interconnected subsystems. Each subsystem is equipped with a sub-controller. We model the interconnection topology as an 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 impose that information exchange between sub-controllers – as defined by the graph $\mathcal{G}_{(A,B)}$ – is confined to a subset of neighboring sub-controllers. We use the d-local communication constraints [19] to formalize this idea. Each subsystem i:

- Receives information from its *d-incoming set* $\mathbf{in}_i(d) := \{v_j \mid \mathbf{dist}(v_j \to v_i) \leq d \in \mathbb{N}\}$, and

- Sends information to its *d-outgoing set* $\mathbf{out}_i(d) := \{v_j \mid \mathbf{dist}(v_i \to v_j) \leq d \in \mathbb{N}\}$.

3

We use a model predictive controller to determine the control input; at time step τ, the controller solves:

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

$$x_0 = x(\tau), \ x_{t+1} = Ax_t + Bu_t + w_t,$$

$$\text{s.t.} \quad x_T \in \mathcal{X}_T, \ x_t \in \mathcal{X}_t, \ u_t \in \mathcal{U}_t \ \forall w_t \in \mathcal{W}_t,$$

$$u_t = \gamma_t(x_{0:t}, u_{0:t-1}), \ t = 0, ..., T-1.$$

To provide tractability, $f_t(\cdot, \cdot)$ and $f_T(\cdot)$ are assumed to be closed, proper, and convex, and $\gamma_t(\cdot)$ is a measurable function of its arguments. The sets $\mathcal{X}_t$ and $\mathcal{U}_t$ are assumed to be closed and convex sets containing the origin for all t. For simplicity, we will consider constant state and input constraint sets $\mathcal{X}$ and $\mathcal{U}$.

Note that (2) does not include the d-local communication constraints. It is not possible to introduce such constraints in a convex manner in the classical MPC formulation (2). However, the DLMPC formulation allows to incorporate locality constraints into the MPC formulation in a straightforward manner with the only requirement that the MPC formulation be compatible with the locality constraints.(see our companion paper [15] for details). Hence, we assume that if two subsystems are coupled through either the constraints or cost, then the two subsystems must be in the d-incoming and d-outgoing set from one another:

Assumption 1. *Given an MPC problem* (2) *over system* (1)*, the objective function f is such that $f(x, u) = \sum f^i([x]_{\mathbf{in}_i(d)}, [u]_{\mathbf{in}_i(d)})$ for f^i local functions; and the constraint sets are such that $x \in \mathcal{X} = \mathcal{X}^1 \cap ... \cap \mathcal{X}^N$, where $x \in \mathcal{X}$ if and only if $[x]_{\mathbf{in}_i(d)} \in \mathcal{X}^i$ for all i, and idem for $\mathcal{U}$ for the local sets $\mathcal{X}^i$ and $\mathcal{U}^i$.*

Approach: In [15], we use the SLS framework to reformulate the MPC problem (2) into the DLMPC problem. This allows for distributed and localized synthesis and implementation, i.e., each subsystem requires only local information to synthesize its local sub-controller and determine the local control action. This is made possible by imposing appropriate d-local structural constraints $\mathcal{L}_d$ on the *closed-loop system responses* of the system $\mathbf{\Phi}_x$ and $\mathbf{\Phi}_u$, which become the decision variables of the MPC problem.

The DLMPC subroutine over time horizon T at time τ is as follows:

$$\min_{\mathbf{\Phi}} \quad f(\mathbf{\Phi}\{1\}x_0) + f_T(\mathbf{\Phi}\{1\}x_0) \tag{3}$$

$$Z_{AB}\mathbf{\Phi} = I, \ x_0 = x(\tau), \ \mathbf{\Phi} \in \mathcal{L}_d,$$

$$\text{s.t.} \quad \mathbf{\Phi}\mathbf{w} \in \mathcal{P}, \ \mathbf{\Phi}_{x,T}\mathbf{w} \in \mathcal{X}_T, \ \forall \mathbf{w} \in \mathcal{W},$$

where $\mathbf{\Phi} := \begin{bmatrix} \mathbf{\Phi}_x^\intercal & \mathbf{\Phi}_u^\intercal \end{bmatrix}^\intercal$, $f_t(\cdot, \cdot)$ and $f_T(\cdot)$ are closed, proper, and convex cost functions, and $\mathcal{P}$ is defined so that $\mathbf{\Phi}\mathbf{w} \in \mathcal{P}$ if and only if $\mathbf{x} \in \mathcal{X}$, and $\mathbf{u} \in \mathcal{U}$. By convention, we define the disturbance to contain the initial condition, i.e., $\mathbf{w} = [x_0^\intercal \ w_0^\intercal \ ... \ w_{T-1}^\intercal] =: [x_0^\intercal \ \boldsymbol{\delta}^\intercal]$ and $\mathcal{W}$ is defined over $\boldsymbol{\delta}$ so that it does not restrict x_0. In the nominal case, i.e., $w_t = 0 \ \forall t$, $\mathcal{P}$ and $\mathcal{X}_T$ are closed and convex sets containing the origin. When noise is present, we restrict ourselves to polytopic sets only: $\mathcal{P} := \{[\mathbf{x}^\intercal \ \mathbf{u}^\intercal]^\intercal : H[\mathbf{x}^\intercal \ \mathbf{u}^\intercal]^\intercal \le h\}$, and consider different options for set $\mathcal{W}$:

- Polytopic set: $\boldsymbol{\delta} \in \{\boldsymbol{\delta} : G\boldsymbol{\delta} \le g\}$.

- Locally norm-bounded: $[\boldsymbol{\delta}]_i \in \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_p \le \sigma\} \ \forall i = 1, ..., N$ and $p \ge 1$.

Although (3) solves a distributed control problem, the optimization problem (3) is itself centralized. To solve (3) in a distributed and localized manner, we perform a variable duplication in order to apply the

ADMM distributed optimization technique [20]:

$$\min_{\tilde{\mathbf{\Phi}}, \tilde{\mathbf{\Psi}}} \quad f(M_1 \tilde{\mathbf{\Phi}} \{1\} x_0) \tag{4}$$

$$\text{s.t.} \quad Z_{AB} M_2 \tilde{\mathbf{\Psi}} = I, \, x_0 = x(\tau), \, \tilde{\mathbf{\Phi}}, \tilde{\mathbf{\Psi}} \in \mathcal{L}_d,$$

$$\tilde{\mathbf{\Phi}} x_0 \in \tilde{\mathcal{P}}, \, \tilde{\mathbf{\Phi}} = \tilde{H} \tilde{\mathbf{\Psi}}$$

where variables $\tilde{\mathbf{\Phi}}$ and $\tilde{\mathbf{\Psi}}$ are extensions of the system response $\mathbf{\Phi}$ to account for the robust case and M_1, M_2, $\tilde{H}$ are auxiliary matrices.[1] We apply ADMM to solve (4) in a distributed and localized manner, as shown in Algorithm 1:

Algorithm 1 Subsystem i DLMPC implementation

1: Measure local state $[x(\tau)]_i$ and exchange with neighbors in $\mathbf{out}_i(d)$. Set $k \leftarrow 0$.
2: Share the measurement with neighbors in $\mathbf{out}_i(d)$.
3: Solve for local rows of $\tilde{\mathbf{\Phi}}^{k+1}$ via (11a) in [15].
4: Exchange rows of $\mathbf{\Phi}$ with d-local neighbors.
5: Solve for local columns of $\tilde{\mathbf{\Psi}}^{k+1}$ via (11b) in [15].
6: Exchange rows of $\mathbf{\Phi}$ with d-local neighbors.
7: Perform the multiplier update step via (11c) in [15].
8: **if** ADMM has converged**:**
       Apply $[u_0]_i = [\Phi_{u,0}[0]]_i [x_0]_i$. Return to step 1.
   else:
       Set $k \leftarrow k + 1$. Return to step 3.

Problem statement: The DLMPC approach (4) introduced in [21, 22] lacks feasibility, stability, and convergence guarantees – our goal is to provide these. Also, (4) does not explicitly consider a terminal set $\mathcal{X}_T$ and terminal cost f_T – we want to leverage these to provide the aforementioned feasibility and stability guarantees. Additionally, Algorithm 1 was developed under the assumption that subsystems are not coupled. We want to augment the algorithm to accommodate coupling as per Assumption 1. In the remainder of this paper, we address all of these problems; we provide theoretical guarantees by selecting an appropriate terminal set and cost, and present distributed and localized algorithms that perform the necessary computations and accommodate coupling.

3 Feasibility and Stability Guarantees

Theoretical guarantees for the DLMPC problem (3) are now derived. First, we describe a maximal positive invariant set using an SLS-style parametrization; recursive feasibility for DLMPC is guaranteed by using this set as the terminal set. We also use this set to construct a terminal cost to guarantee asymptotic stability for the nominal setting and input-to-state stability (ISS) for the robust setting. Convergence results from the ADMM literature are used to establish convergence guarantees.

3.1 Feasibility guarantees

Recursive feasibility guarantees for the DLMPC problem (3) are given by the following lemma:

---

[1]Their actual definition depends on whether the system experiences no disturbances, locally bounded disturbances, or polytopic disturbances; they are defined in §IV of [15].

Lemma 1. *Let the terminal set $\mathcal{X}_T$ for the DLMPC problem* (3) *be of the form*

$$\mathcal{X}_T := \{x_0 \in \mathbb{R}^n : \ \mathbf{\Phi} \begin{bmatrix} x_0^\mathsf{T} & \boldsymbol{\delta}^\mathsf{T} \end{bmatrix}^\mathsf{T} \in \mathcal{P} \ \forall \boldsymbol{\delta} \in \mathcal{W}\}, \tag{5}$$

for some $\mathbf{\Phi}$ satisfying $Z_{AB}\mathbf{\Phi} = I$. Then recursive feasibility is guaranteed for the DLMPC problem (3). *Moreover, $\mathcal{X}_T$ is the maximal robust positive invariant set for the closed-loop described by $\mathbf{\Phi}$.*

Proof. First, we show that $\mathcal{X}_T$ is the maximal robust positive invariant set. By Algorithm 10.4 in [1], the set

$$\mathcal{S} := \bigcap_{k=0}^{\infty} \mathcal{S}_k, \quad \text{with} \quad \mathcal{S}_0 = \mathcal{X},$$
$$\mathcal{S}_k = \{x \in \mathbb{R}^n : \ Ax + Bu + w \in \mathcal{S}_{k-1} \ \forall w \in \mathcal{W}\}$$

is the maximal robust positive invariant set for the closed-loop system (1) with some fixed input $u \in \mathcal{U}$. We show by induction that $\mathcal{S}_k$ can also be written as

$$\mathcal{S}_k = \{x_0 \in \mathbb{R}^n : \ x_k \in \mathcal{X} \ \forall w_{0:k-1} \in \mathcal{W}\} \tag{6}$$

for some sequence $u_{0:k-1} \in \mathcal{U}$. The base case at $k = 1$ is trivially true, since $\mathcal{S}_0 = \mathcal{X}$. For the inductive step, assume that equation (6) holds at k. Then, at $k = 1$,

$$\mathcal{S}_{k+1} = \{x \in \mathbb{R}^n : \ Ax + Bu + w \in \mathcal{S}_k \ \forall w \in \mathcal{W}\}$$
$$= \{x_0 \in \mathbb{R}^n : \ x := Ax_0 + Bu + w \ s.t.$$
$$A^{k+1}x + \sum_{j=0}^{k} A^j(Bu_j + w) \in \mathcal{X} \ \forall w \in \mathcal{W}\}$$
$$= \{x_0 \in \mathbb{R}^n : \ x_{k+1} \in \mathcal{X}, \ \forall w_{0:k} \in \mathcal{W}\}$$

for some sequence of inputs $u_{0:k} \in \mathcal{U}$. This implies that $\mathcal{S}$ can be written as

$$\mathcal{S} = \{x_0 \in \mathbb{R}^n : \ \mathbf{x} \in \mathcal{X} \ \forall w \in \mathcal{W}\} \text{ for some } \mathbf{u} \in \mathcal{U}.$$

From the definition of $\mathbf{\Phi}$, this implies directly that

$$\mathcal{S} = \{x_0 \in \mathbb{R}^n : \ \mathbf{\Phi} \begin{bmatrix} x_0^\mathsf{T} & \boldsymbol{\delta}^\mathsf{T} \end{bmatrix} \in \mathcal{P} \ \forall \boldsymbol{\delta} \in \mathcal{W}\} := \mathcal{X}_T,$$

for some $\mathbf{\Phi}$ satisfying $Z_{AB}\mathbf{\Phi} = I$. This constraint automatically enforces that the resulting closed-loop is feasible, i.e. that $\mathbf{u} = \mathbf{\Phi}_u \begin{bmatrix} x_0^\mathsf{T} & \boldsymbol{\delta}^\mathsf{T} \end{bmatrix}$ exists; furthermore, all inputs $\mathbf{u}$ can be captured by this closed-loop parametrization since it parametrizes a linear time-varying controller over a finite time horizon [17]. Hence, $\mathcal{X}_T$ is the maximal robust positive invariant set for the closed-loop system as defined by $\mathbf{\Phi}$.

Next, we show that imposing $\mathcal{X}_T$ as the terminal set of the DLMPC problem (3) guarantees recursive feasibility. Notice that by definition, $\mathcal{X}_T$ is not only a robust positive invariant set but also a robust *control* invariant set. We can directly apply the proofs from Theorem 12.1 in [1] to the robust setting – recursive feasibility is guaranteed if the terminal set is control invariant, which $\mathcal{X}_T$ is. $\qquad\square$

Remark 1. *Recursive feasibility is guaranteed by a* control *invariant $\mathcal{X}_T$. In the ideal case, we want $\mathcal{X}_T$ to be the maximal robust control invariant set, in order to minimize conservatism introduced by $\mathcal{X}_T$ in* (3). *However, this set generally lacks sparsity, violates Assumption 1, and is not amenable for inclusion in our distributed and localized algorithm. For this reason, we use the maximal robust* positive *invariant set in Lemma 1. Here, the choice of $\mathbf{\Phi}$ is critical in determining how much conservatism $\mathcal{X}_T$ will introduce.*

As suggested in §12 of [1], we choose $\mathbf{\Phi}$ corresponding to the unconstrained closed-loop system. In the interests of distributed synthesis and implementation, we additionally enforce $\mathbf{\Phi}$ to have localized structure. We discuss how this $\mathbf{\Phi}$ is computed in §4, and demonstrate that the resulting terminal set $\mathcal{X}_T$ introduces no conservatism in §5.

By Assumption 1, constraint sets are localized, i.e. $\mathcal{X} = \mathcal{X}^1 \cap ... \cap \mathcal{X}^N$, where $x \in \mathcal{X}$ if and only if $[x]_{\mathbf{in}_i(d)} \in \mathcal{X}^i$ for all i (and idem for $\mathcal{U}$). Moreover, we can use the constraint $\mathcal{L}_d$ to enforce that the system response $\mathbf{\Phi}$ is localized. This implies that the set $\mathcal{X}_T$ is also localized:

$$\mathcal{X} = \mathcal{X}_T^1 \cap ... \cap \mathcal{X}_T^N,$$

where

$$\mathcal{X}_T^i = \{[x_0]_{\mathbf{in}_i(d)} \in \mathbb{R}^{[n]_i} : [\mathbf{\Phi}]_{\mathbf{in}_i(d)} \begin{bmatrix} x_0^\mathsf{T} & \boldsymbol{\delta}^\mathsf{T} \end{bmatrix}_{\mathbf{in}_i(2d)} \in \mathcal{P}^i \ \forall [\boldsymbol{\delta}]_{\mathbf{in}_i(2d)} \in \mathcal{W}^{\mathbf{in}_i(d)}\}$$

and $x \in \mathcal{X}_T$ if and only if $[x]_{\mathbf{in}_i(d)} \in \mathcal{X}_T^i$ for all i. We will show in §4 that this allows for a localized and distributed computation of the terminal set $\mathcal{X}_T$.

3.2   Stability guarantees

Stability guarantees for the DLMPC problem (3) are given by the following lemma:

Lemma 2. *Consider system* (1) *subject to the MPC law* (3), *where:*

1. *The cost f is continuous and positive definite.*

2. *The set $\mathcal{P}$ contains the origin and is closed.*

3. *$\mathcal{X}_T$ is defined by* (5).

4. *$f_T(x) = \inf\{\eta \geq 0 : x \in \eta\mathcal{X}_T\}$.*

Then, in the nominal setting, the origin is asymptotically stable with domain of attraction $\mathcal{X}$, and in the robust setting, $\mathcal{X}_T$ is input-to-state stable with domain of attraction $\mathcal{X}$.

Proof. It suffices to show that these conditions immediately imply satisfaction of the necessary assumptions in [23], together with the additional sufficient conditions of Theorem 4.2 in [24]. These results state that if

(i) f and f_T are continuous and positive definite,

(ii) $\mathcal{X}$, $\mathcal{U}$, $\mathcal{X}_T$ contain the origin and are closed,

(iii) $\mathcal{X}_T$ is control invariant, and

(iv) $\min_{u \in \mathcal{U}} f(x, u) + f_T(Ax + Bu) - f_T(x) \leq 0 \ \forall x \in \mathcal{X}_T$,

then the desired stability guarantees hold.

Condition *1)* implies satisfaction of the part of (i) that concerns f.

Condition *2)* implies satisfaction of (ii). If $\mathcal{P}$ contains the origin and is closed, by definition this implies that $\mathcal{X}$ and $\mathcal{U}$ also contain the origin and are closed. Also, since $\mathcal{X}_T$ is defined in terms of $\mathcal{P}$ in (5), $\mathcal{X}_T$ also contains the origin and is closed.

Condition *3)* implies satisfaction of (iii) by virtue of Lemma 1.

Condition *4)* implies that f_T is a Lyapunov function on $\{x \in \mathbb{R}^n : 1 \leq f_T(x)\} \supseteq \mathcal{X}_T$ since it is the Minkowski functional of the terminal set $\mathcal{X}_T$ (see Theorem 3.3. in [18]). Therefore, the condition stated in

(iv) is automatically satisfied for all $x \in \mathcal{X}_T$. Moreover, f_T is necessarily positive definite, so the part of (i) that concerns f_T is satisfied as well.

Therefore, by virtue of the results in [23] and [24], we guarantee asymptotic stability of the origin in the nominal setting and ISS of $\mathcal{X}_T$ in the robust setting, both with domain of attraction $\mathcal{X}$.

$\square$

For any cost f and constraint $\mathcal{P}$ that satisfy conditions *1)* and *2)* of Lemma 2, we can choose an appropriate terminal set $\mathcal{X}_T$ and terminal cost f_T as per *3)* and *4)* to satisfy the lemma. This guarantees stability for the DLMPC problem (3). However – as stated, f_T does not satisfy Assumption 1, and therefore it is not localized. In particular, f_T can be written as:

$$f_T(x) = \inf_{\eta} \{\eta \geq 0 : [x]_{\mathbf{in}_i(d)} \in \eta \mathcal{X}_T^i \ \forall i\}, \tag{7}$$

which cannot be written as a sum of local functions. Nonetheless, this scalar objective function admits a distributed and localized implementation – we can add it in the DLMPC algorithm using the ADMM-based consensus technique described in §4.

3.3 Convergence guarantees

Algorithm 1 relies on ADMM. We can guarantee convergence of the overall algorithm by leveraging the ADMM convergence result from [20].

Lemma 3. *In Algorithm 1, the residue, objective function, and dual variable converge as $k \rightarrow \infty$ i.e.*

$$\tilde{H}\tilde{\boldsymbol{\Phi}}^k - \tilde{\boldsymbol{\Psi}}^k \rightarrow 0, \ f(\tilde{\boldsymbol{\Phi}}^k x_0) \rightarrow f(\tilde{\boldsymbol{\Phi}}^* x_0), \ \tilde{\boldsymbol{\Psi}}^k \rightarrow \tilde{\boldsymbol{\Psi}}^*,$$

where $\star$ indicates optimal value.

Proof. Algorithm 1 is the result of applying ADMM to the DLMPC problem (3), then exploiting the separability and structure of resulting sub-problems to achieve distributed and localized implementation, as presented in [15]. Thus, to prove convergence, we only need to show that the underlying ADMM algorithm converges. By the ADMM convergence result in [20], the desired convergence of the residue, objective function, and dual variable are guaranteed if

1. *The extended-real-value functional for the algorithm is closed, proper, and convex, and*

2. *The unaugmented Lagrangian for the algorithm has a saddle point.*

We first show *1)*. The extended-real-value functional $h(\tilde{\boldsymbol{\Phi}})$ is defined for this algorithm as

$$h(\tilde{\boldsymbol{\Phi}}) = \begin{cases} f(M_1\tilde{\boldsymbol{\Phi}}\{1\}x_0) & \text{if } Z_{AB}M_2\tilde{H}^{\dagger}\tilde{\boldsymbol{\Phi}} = I, \\ & \tilde{\boldsymbol{\Phi}} \in \mathcal{L}_d, \tilde{\boldsymbol{\Phi}}x_0 \in \tilde{\mathcal{P}}, \\ \infty & \text{otherwise.} \end{cases}$$

where $\tilde{H}^{\dagger}$ is the left inverse of $\tilde{H}$ from (4); $\tilde{H}$ has full column rank. When formulating the DLMPC problem (3) as an ADMM problem, we perform variable duplication to obtain problem (4). We can write (4) in terms of $h(\tilde{\boldsymbol{\Phi}})$ with the constraint $\tilde{\boldsymbol{\Phi}} = \tilde{H}\tilde{\boldsymbol{\Psi}}$:

$$\min_{\tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}}} \ h(\tilde{\boldsymbol{\Phi}}) \ \text{s.t.} \ \ \tilde{\boldsymbol{\Phi}} = \tilde{H}\tilde{\boldsymbol{\Psi}}.$$

By assumption, $f(M_1 \tilde{\mathbf{\Phi}} x_0)$ is closed, proper, and convex, and $\tilde{\mathcal{P}}$ is a closed and convex set. The remaining constraints $Z_{AB}\tilde{\mathbf{\Phi}} = I$ and $\tilde{\mathbf{\Phi}} \in \mathcal{L}_d$ are also closed and convex. Hence, $h(\tilde{\mathbf{\Phi}})$ is closed, proper, and convex.

We now show *2)*. This condition is equivalent to showing that strong duality holds [25]. Since problem (3) is assumed to have a feasible solution in the relative interior of $\mathcal{P}$ by means of Lemma 1 (given that the first iteration is feasible), Slater's condition is automatically satisfied, and therefore the unaugmented Lagrangian of the problem has a saddle point.

Both conditions of the ADMM convergence result from [20] are satisfied – Algorithm 1 converges in residue, objective function, and dual variable, as desired.

$\square$

4 Feasible and Stable DLMPC

We incorporate theoretical results from §3 into the DLMPC computation. We provide an algorithm to compute the terminal set $\mathcal{X}_T$. We also provide a distributed and localized computation for the terminal cost function f_T. The terminal set and cost generally introduce local coupling among subsystems; this minimizes conservatism but requires an extension to Algorithm 1 to accommodate coupling. All algorithms provided are distributed and localized, with computational complexity that is independent of the global system size.

4.1 Offline synthesis of the terminal set $\mathcal{X}_T$

Our first result is to provide an offline algorithm to compute terminal set $\mathcal{X}_T$ from (5) in a distributed and localized manner. As discussed in §3, we compute the maximal robust positive invariant set for the unconstrained localized closed-loop system. We use SLS-based techniques to obtain a localized closed-loop map $\mathbf{\Phi}$.

Our algorithm is based on Algorithm 10.4 from [1]. The advantage of implementing this algorithm in terms of the localized closed-loop map $\mathbf{\Phi}$ is twofold: i) we can work with locally-bounded and polytopic disturbance sets $\mathcal{W}$ by leveraging Lemmas 1 and 2 in [15], and ii) the resulting robust invariant set is automatically localized given the localized structure of the closed-loop map.

We start by finding a localized closed-loop map $\mathbf{\Phi}$ for system (1). To do this, we need to solve

$$\min_{\mathbf{\Phi}} f(\mathbf{\Phi}) \quad \text{s.t. } Z_{AB}\mathbf{\Phi} = I, \ \mathbf{\Phi} \in \mathcal{L}_d. \tag{8}$$

This is an SLS problem with a separable structure for most standard cost functions f [17]. The separable structure admits localized and distributed computation. Even when separability is not apparent, a relaxation can often be found that allows for distributed computation (see for example [26, 27]). The infinite-horizon solution for quadratic cost is presented in [28]. For other costs, computing an infinite horizon solution to (8) remains an open question – in these cases, we can use finite impulse response SLS with sufficiently long time horizon.

Once a localized closed-map $\mathbf{\Phi}$ has been found, we can compute the associated maximal positive invariant set. In the robust case, we leverage results from Lemmas 1 and 2 in our companion paper [15], which use duality arguments to tackle specific formulations of $\mathcal{W}$. We denote σ as the upper bound of $\|[\delta]_i\|_*$ for all i, where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|_p$. Also, Each e_j is the jth vector in the standard basis.

- Nominal (i.e. no disturbance):

$$\mathcal{S} := \{x_0 \in \mathbb{R}^n : \ \mathbf{\Phi}\{1\}x_0 \in \mathcal{P}\}.$$

9

- Locally bounded disturbance:

$$\mathcal{S} := \{x_0 \in \mathbb{R}^n : [H]_i [\mathbf{\Phi}\{1\}]_i [x_0]_i + \sum_j \sigma \left\| e_j^\mathsf{T} [H]_i [\mathbf{\Phi}\{2:T\}]_i \right\|_* \le [h]_i \, \forall i\}.$$

- Polytopic disturbance:

$$\mathcal{S} := \{x_0 \in \mathbb{R}^n : H\mathbf{\Phi}\{1\}x_0 + \Xi g \le h, \text{where } \Xi(j,:) = \min_{\Xi_j \ge 0} \Xi_j g \text{ s.t. } H(j,:)\mathbf{\Phi}\{2:T\} = \Xi_j G \, \forall j\}.$$

As per Lemma 1, we calculate $\mathcal{S}$ by iteratively computing $\mathcal{S}_{k+1}$ from $\mathcal{S}_k$.[2] Assume $\mathcal{S}_k$ can be written as

$$\mathcal{S}_k = \{x_0 \in \mathbb{R}^n : \hat{H}x \le \hat{h}\}.$$

Then, we can write $\mathcal{S}_{k+1}$ as

$$\mathcal{S}_{k+1} = \{x_0 \in \mathbb{R}^n : \Phi_{x,1}[1]x_0 \in \mathcal{F}(\mathcal{S}_k), \Phi_{u,0}[0]x_0 \in \mathcal{U}\}, \tag{9}$$

where $\mathcal{F}(\mathcal{S}_k) := \mathcal{S}_k$ in the nominal case. In the case of locally bounded disturbance,

$$\mathcal{F}(\mathcal{S}_k) := \{x \in \mathbb{R}^n : [\hat{H}]_i [x]_i + \sum_j \sigma \left\| e_j^\mathsf{T} [\hat{H}]_i \right\|_* \le [\hat{h}]_i \, \forall i\},$$

and for polytopic disturbance,

$$\mathcal{F}(\mathcal{S}_k) := \{x \in \mathbb{R}^n : \hat{H}x + \Xi g \le \hat{h}, \text{where } \Xi(j,:) = \min_{\Xi_j \ge 0} \Xi_j g \text{ s.t. } \hat{H}(j,:) = \Xi_j G \, \forall j\}.$$

These formulations use the simplifying fact that $\Phi_{x,0}[0] = I$ for all feasible closed-loop dynamics. Also, notice that by using $\mathcal{S}_k$ to calculate $\mathcal{S}_{k+1}$, the only elements of $\mathbf{\Phi}$ that we require are $\Phi_{x,1}[1]$ and $\Phi_{u,0}[0]$.

The conditions stated in (9) are row- and column-wise separable in $\mathbf{\Phi}$ (and Ξ)[3]; this allows us to calculate $\mathcal{S}_k$ in a distributed way. Furthermore, $\mathcal{F}(\mathcal{S}_k)$ and $\mathcal{U}$ are localizable by Assumption 1, in both nominal and robust settings. Since $\mathbf{\Phi}$ is also localized, we can exploit the structure of $\mathbf{\Phi}$ to and rewrite S_{k+1} as the intersection of local sets, i.e. $S_{k+1} = S_{k+1}^1 \cap \cdots \cap S_{k+1}^N$, where

$$\mathcal{S}_{k+1}^i = \{[x_0]_{\mathbf{in}_i(d)} \in \mathbb{R}^{[n]_i} : [\Phi_{x,1}[1]]_{\mathbf{in}_i(d)} [x_0]_{\mathbf{in}_i(d)} \in \mathcal{F}(\mathcal{S}_k^{\mathbf{in}_i(d)}), [\Phi_{x,1}[1]]_{\mathbf{in}_i(d)} [x_0]_{\mathbf{in}_i(d)} \in \mathcal{U}^{\mathbf{in}_i(d)}\} \tag{10}$$

We now present Algorithm 2 to compute the terminal set $\mathcal{X}_T := \mathcal{S}$ in a distributed and local manner. Each subsystem i computes its own local terminal set $\mathcal{S}^i$ using only local information exchange. This algorithm is inspired by Algorithm 10.4 in [1].

If state and input constraints $\mathcal{X}$ and $\mathcal{U}$ do not induce coupling (as assumed in Algorithm 1), the resulting terminal set $\mathcal{X}_T$ will be at most d-localized. If $\mathcal{X}$ and $\mathcal{U}$ do induce d-local coupling, the terminal set will be at most $2d$-localized since – in the presence of coupling – Alg. 2 requieres communication with not only local patch neighbors, but also neighbors of those neighbors. Convergence is guaranteed for system (1) if it is stable when $u = 0$, $w = 0$, and when the constraint and disturbance sets $\mathcal{X}$, $\mathcal{U}$, $\mathcal{W}$ are bounded and contain the origin, as per [29].

---

[2]For simplicity of presentation, we write out $\mathcal{S}$ only for finite-horizon $\mathbf{\Phi}$; the proposed algorithm to synthesize $\mathcal{S}$ works for infinite-horizon $\mathbf{\Phi}$ as well.

[3]A formal definition or row and column-wise separability is given in [15].

Algorithm 2 Subsystem i terminal set computation
---
  input: $[\Phi_{x,1}[1]]_{\mathbf{in}_i(d)}, [\Phi_{u,0}[0]]_{\mathbf{in}_i(d)}, \mathcal{U}^{\mathbf{in}_i(d)}$
    for polytopic noise, also $[G]_{\mathbf{in}_i(d)}, [g]_{\mathbf{in}_i(d)}$
1: $\mathcal{S}_0^i \leftarrow \mathcal{X}^i, k \leftarrow -1$
2: **repeat:**
3:     $k \leftarrow k+1$
4:     Share $\mathcal{S}_k^i$ with $\mathbf{out}_i(d)$.
         Receive $\mathcal{S}_k^j$ from all $j \in \mathbf{in}_i(d)$.
5:     Compute $\mathcal{S}_{k+1}^i$ via (10).
6:     $\mathcal{S}_{k+1}^i \leftarrow \mathcal{S}_k^i \cap \mathcal{S}_{k+1}^i$
7: **until:** $\mathcal{S}_{k+1}^i = \mathcal{S}_k^i$ for all i
8: $\mathcal{X}_T^i \leftarrow \mathcal{S}_k^i$
  output: $\mathcal{X}_T^i$

4.2 Online synthesis of DLMPC

DLMPC Algorithm 1 was developed under the assumption that no coupling is introduced through cost or constraints. We now extend this algorithm to accommodate local coupling, which is allowed as per Assumption 1. This allows us to incorporate the terminal set and cost introduced in the previous sections, both of which generally induce local coupling. We will use notation that assumes all constraints and costs (including the terminal set) are d-localized. In the case that the terminal set is $2d$-localized, subsystems will need to exchange information with neighbors up to $2d$-hops away; simply replace { $\mathbf{out}_i(d)$, $\mathbf{in}_i(d)$ } with { $\mathbf{out}_i(2d)$, $\mathbf{in}_i(2d)$ } wherever they appear in Algorithms 1, 2 and 3.

Appending the terminal set (5) and terminal cost (7) to the DLMPC problem (4), gives:

$$\min_{\tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}}, \eta} \qquad f(M_1 \tilde{\boldsymbol{\Phi}}\{1\}x_0) + \eta \tag{11}$$
$$\text{s.t.} \qquad Z_{AB}M_2\tilde{\boldsymbol{\Psi}} = I, x_0 = x(\tau),\ \tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}} \in \mathcal{L}_d,$$
$$\tilde{\boldsymbol{\Phi}}x_0 \in \tilde{\mathcal{P}},\ \tilde{H}\tilde{\boldsymbol{\Phi}} = \tilde{\boldsymbol{\Psi}},$$
$$M_1 \tilde{\boldsymbol{\Phi}}_T\{1\}x_0 \in \eta\mathcal{X}_T, 0 \le \eta \le 1,$$

where $\tilde{\boldsymbol{\Phi}}_T$ represents block rows of $\tilde{\boldsymbol{\Phi}}$ that correspond to time horizon T – e.g., in the nominal setting, the last block row of $\boldsymbol{\Phi}_x$.

Remark 2. *In the robust setting, we incorporate the terminal constraint into $\tilde{\mathcal{P}}$ to ensure robust satisfaction of the terminal constraint. However, $\mathcal{X}_T$ still appears as nominal constraint in order to integrate the definition of the terminal cost (7) into the DLMPC formulation (3).*

In the original formulation (4), we assume no coupling; as a result, all expressions involving $\tilde{\boldsymbol{\Phi}}$ are row-separable, and all expressions involving $\tilde{\boldsymbol{\Psi}}$ are column-separable. If we allow local coupling as per Assumption 1, we lose row-separability; row-separability is also lost in the new formulation (11) due to local coupling in the terminal set. This is because without coupling, $[x]_i$ and $[u]_i$ (which correspond to a fixed set of rows in $\boldsymbol{\Phi}$), can only appear in f^i and $\mathcal{P}^i$; thus, each row of $\boldsymbol{\Phi}$ (and $\tilde{\boldsymbol{\Phi}}$) is solved by exactly one subsystem in step 3 of Algorithm 1. With coupling, $[x]_i$ and $[u]_i$ can appear in f^j and $\mathcal{P}^j$ for any $j \in \mathbf{in}_i(d)$; now, each row of $\boldsymbol{\Phi}$ is solved for by multiple local subsystems at once, and row-separability is lost. This only affects step 3 of Algorithm 1 (i.e. the row-wise problem); other steps remain unchanged. We write out

the row-wise problem corresponding to (11):

$$\min_{\tilde{\boldsymbol{\Phi}}} \quad f(M_1\tilde{\boldsymbol{\Phi}}\{1\}x_0) + \eta + g(\tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}}^k, \boldsymbol{\Lambda}^k) \tag{12}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{\Phi}} \in \tilde{\mathcal{P}} \cap \mathcal{L}_d, \ M_1\tilde{\boldsymbol{\Phi}}_T\{1\}x_0 \in \eta\mathcal{X}_T,$$
$$0 \le \eta \le 1, \ x_0 = x(\tau),$$

where $g(\tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}}, \boldsymbol{\Lambda}) = \frac{\varrho}{2}\left\|\tilde{\boldsymbol{\Phi}} - \tilde{H}\tilde{\boldsymbol{\Psi}} + \boldsymbol{\Lambda}\right\|_F^2$, and all other relevant variables and sets are defined as per equation (10) in our companion paper [15].

Note that if $\mathcal{P}$ induces coupled linear inequality constraints, row-separability is maintained. Though $[x]_i$ and $[u]_i$ appear in multiple $\mathcal{P}_j$, this corresponds to distinct rows of $\boldsymbol{\Omega}$, each solved by one subsystem, as opposed to one row of $\boldsymbol{\Phi}$ that is solved for by multiple subsystems. A similar idea applies if coupling is induced by some quadratic cost, i.e. $f(\boldsymbol{\Phi}x_0) = \|C\boldsymbol{\Phi}x_0\|_F^2$ for some matrix C which induces coupling. In this case, we can modify $\tilde{H}$ such that we enforce $\boldsymbol{\Phi} = C\boldsymbol{\Psi}$, and rewrite the cost as $\|\boldsymbol{\Phi}x_0\|_F^2$; now, each row of $\boldsymbol{\Phi}$ is solved by exactly one subsystem, and row-separability is maintained.[4] However, this technique does not apply to coupling in the general case; nor does it apply to the coupling induced by the terminal cost – it is generally true that each row of $\tilde{\boldsymbol{\Phi}}$ will be need to be solved for by several subsystems. We introduce a new vector variable

$$\mathbf{X} := M_1\tilde{\boldsymbol{\Phi}}\{1\}x_0.$$

This variable facilitates consensus between subsystems who share the same row(s) of $\tilde{\boldsymbol{\Phi}}$. Each subsystem solves for components $[\mathbf{X}]_{\mathbf{in}_i(d)}$, and comes to a consensus with its neighboring subsystems on the value of these components. In the interest of efficiency, we directly enforce consensus on elements of $\mathbf{X}$ instead of enforcing consensus on rows of $\tilde{\boldsymbol{\Phi}}$. We introduce a similar variable for terminal cost η; we define vector $\boldsymbol{\eta} := [\eta_1, \ldots, \eta_N]$. Subsystem i solves for η_i, which is its own copy of η. It then comes to a consensus on this value with its neighbors, i.e. η_j has the same value $\forall j \in \mathbf{in}_i(d)$. Assuming that $\mathcal{G}_{(A,B)}$ is connected, this guarantees that η_i has the same value $\forall i \in \{1\ldots N\}$, i.e., all subsystems agree on the value of η. We combine these two consensus-facilitating variables into the augmented vector variable

$$\tilde{\mathbf{X}} := \begin{bmatrix}\mathbf{X}^\mathsf{T} & \boldsymbol{\eta}^\mathsf{T}\end{bmatrix}^\mathsf{T}.$$

With this setup, we follow a variable duplication strategy and apply ADMM for consensus [30]. In particular, we duplicate variables so each subsystem has *its own copy* of the components of $\tilde{\mathbf{X}}$. Problem (12) becomes:

$$\min_{\tilde{\boldsymbol{\Phi}}, \tilde{\mathbf{X}}, \tilde{\mathbf{Y}}} \quad \tilde{f}(\tilde{\mathbf{X}}) + g(\tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\Psi}}^k, \boldsymbol{\Lambda}^k) \tag{13}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{\Phi}}, \tilde{\mathbf{X}} \in \mathcal{Q}, \ \tilde{\mathbf{X}} \in \tilde{\mathcal{X}}_T, \ \tilde{\boldsymbol{\Phi}} \in \mathcal{L}_d, \ x_0 = x(\tau),$$
$$[\tilde{\mathbf{X}}]_i = [M_1\tilde{\boldsymbol{\Phi}}]_{i_r}[x_0]_i, \ [\tilde{\mathbf{X}}]_j = [\tilde{\mathbf{Y}}]_j \ \forall j \in \mathbf{in}_i(d) \ \forall i,$$

where we define $\tilde{f}$, $\tilde{\mathcal{X}}_T$ and $\mathcal{Q}$ as follows:

- $\tilde{f}(\tilde{\mathbf{X}}) := f(\mathbf{X}) + \frac{1}{N}\sum_{i=1}^N \eta_i$,

- $\tilde{\mathbf{X}} \in \tilde{\mathcal{X}}_T$ if and only if $M_1\tilde{\boldsymbol{\Phi}}_T\{1\}x_0 \in \eta_i\mathcal{X}_T \ \forall i$,

- $\tilde{\boldsymbol{\Phi}}, \tilde{\mathbf{X}} \in \mathcal{Q}$ if and only if $\mathbf{0} \le \boldsymbol{\eta} \le \mathbf{1}$ **and** $\{\tilde{\boldsymbol{\Phi}} \in \tilde{\mathcal{P}}$ if $\tilde{\mathcal{P}}$ induces linear inequalities **or** $\tilde{\mathbf{X}} \in \mathcal{P}$ otherwise$\}$.[5]

---

[4]We would also need to use $[\Psi_{u,0}[0]]_i$ for the algorithm output instead of $[\Phi_{u,0}[0]]_i$.

[5]By the assumptions in [15], $\mathcal{P}$ (and therefore $\tilde{\mathcal{P}}$) must induce linear inequalities in the robust setting. The only case where $\tilde{\mathcal{P}}$

The structure of problem (13) allows us to solve it in a distributed and localized manner via ADMM-based consensus. In particular, each subsystem i solves:

$$\left\{\begin{array}{c} [\tilde{\boldsymbol{\Phi}}]_{i_r,}^{k+1,n+1} \\[2mm] [\tilde{\mathbf{X}}]_{\mathbf{in}_i(d)}^{n+1} \end{array}\right\} = \left\{\begin{array}{cc} \underset{[\tilde{\boldsymbol{\Phi}}]_{i_r},[\tilde{\mathbf{X}}]_{\mathbf{in}_i(d)}}{\operatorname{argmin}} & \tilde{g}_k^i(\tilde{\mathbf{X}},\tilde{\boldsymbol{\Phi}}) + \dfrac{\mu}{2} h^i(\tilde{\mathbf{X}},\tilde{\mathbf{Y}}^n,\tilde{\mathbf{Z}}^n) \\[3mm] & [\tilde{\boldsymbol{\Phi}}]_{r_i} \in \mathcal{Q}^i,\ [\tilde{\mathbf{X}}_T]_i \in \tilde{\mathcal{X}}_T^i \cap \mathcal{Q}^i, \\ \text{s.t.} & [\tilde{\mathbf{X}}]_i = [M_1\tilde{\boldsymbol{\Phi}}]_{r_i}[x_0]_i \end{array}\right\} \tag{14a}$$

$$[\tilde{\mathbf{Y}}]_i^{n+1} = \frac{h^i(\tilde{\mathbf{X}}^{n+1},\mathbf{0},\tilde{\mathbf{Z}}^n)}{|\mathbf{in}_i(d)|}, \tag{14b}$$

$$[\tilde{\mathbf{Z}}]_{ij}^{n+1} = [\tilde{\mathbf{Z}}]_{ij}^n + [\tilde{\mathbf{X}}]_i^{n+1} - [\tilde{\mathbf{Y}}]_j^{n+1}, \tag{14c}$$

where to simplify notation, we define

$$\tilde{g}_k^i(\tilde{\mathbf{X}},\tilde{\boldsymbol{\Phi}}) := \tilde{f}^i([\tilde{\mathbf{X}}]_{\mathbf{in}_i(d)}) + g\big([\boldsymbol{\Phi}]_{i_r} - [\boldsymbol{\Psi}]_{i_r}^k + [\boldsymbol{\Lambda}]_{i_r}^k\big),$$

$$h^i(\tilde{\mathbf{X}},\tilde{\mathbf{Y}}^n,\tilde{\mathbf{Z}}^n) := \sum_{j\in\mathbf{in}_i(d)} \big\|[\mathbf{X}]_j - [\mathbf{Y}]_i^n + [\mathbf{Z}]_{ij}^n\big\|_F^2,$$

Consensus iterations are denoted by n, outer-loop (i.e. Algorithm 1) iterations are denoted by k, and μ is the ADMM consensus parameter. Intuitively, $\tilde{g}_k^i$ represents the original objective from (12), and h^i represents the consensus objective.

The subroutine described by (14) allows us to accommodate local coupling induced by cost and constraint (including terminal), and can be implemented in a distributed and localized manner. As stated above, this subroutine solves the row-wise problem (12), corresponding to step 3 of Algorithm 1. Thus, in order to accommodate local coupling (including terminal set and cost), we need only to replace step 3 of Algorithm 1 with the subroutine defined by Algorithm 3 below. Convergence is guaranteed by a similar argument to Lemma 3.

Algorithm 3 Subsystem i implementation of step 3 in Algorithm 1 when subject to localized coupling

---

    input: tolerance parameters $\epsilon_x, \epsilon_z, \mu > 0$.
1: $n \leftarrow 0$.
2: Solve optimization problem (14a).
3: Share $[\mathbf{X}]_i^{n+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{X}]_j^{n+1}$ from $j \in \mathbf{in}_i(d)$.
4: Perform update (14b).
5: Share $[\mathbf{Y}]_i^{n+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{Y}]_j^{n+1}$ from $j \in \mathbf{in}_i(d)$.
6: Perform update (14c).
7: **if** $\big\|[\mathbf{X}]_i^{n+1} - [\mathbf{Z}]_i^{n+1}\big\|_F < \epsilon_x$
    and $\big\|[\mathbf{Z}]_i^{n+1} - [\mathbf{Z}]_i^n\big\|_F < \epsilon_z$ **:**
      Go to step 4 in Algorithm 1.
    else:
      Set $n \leftarrow n + 1$ and return to step 2.

Computational complexity of the algorithm: The presence of coupling induces an increase in computational complexity compared to the uncoupled scenario (i.e., Algorithm 1); however, the scalability properties from the uncoupled scenario still apply. Complexity in the current algorithm is determined by steps 2, 4, 6 of Algorithm 3 and steps 5 and 7 of Algorithm 1. Except for step 2 in Algorithm 3, all other steps

---

may induce something other than linear inequalities is in the nominal setting.

can be solved in closed form. Sub-problems solved in step 2 require $O(d^2T^2)$ optimization variables in the robust setting ($O(dT^2)$ in the nominal setting) and $O(d^2T)$ constraints. All other steps enjoy less complexity since their evaluation reduces to multiplication of matrices of dimension $O(d^2T^2)$ in the robust setting, and $O(d^2T)$ in the nominal setting. The difference in complexity between the nominal and robust settings is consistent with the uncoupled scenario. Compared to the uncoupled scenario, additional computation burden is incurred by the consensus subroutine in Algorithm 3, which increases the total number of iterations. The consensus subroutine also induces increased communication between subsystems, as it requires local information exchange. However, this exchange is limited to a d-local region, resulting in small consensus problems that converge quickly, as we illustrate empirically in §5. As with the original uncoupled algorithm, complexity of this new algorithm is determined by the size of the local neighborhood and does not increase with the size of the global network.

5 Simulation Experiments

Using examples, we demonstrate how adding terminal constraint and cost affects the performance of the DLMPC algorithm. We verify that introducing terminal constraint and cost indeed produces the desired feasibility and stability properties. We additionally empirically characterize the computational complexity of algorithms presented in previous sections and verify that complexity is independent of global network size for both offline and online algorithms. Code to replicate these experiments is available at `https://github.com/unstable-zeros/dl-mpc-sls`; this code makes use of the SLS toolbox [31], which includes ready-to-use MATLAB implementations of all algorithms presented in this paper and its companion paper [15].

5.1 System model

Simulations are performed on the system from our companion paper [15]; a two-dimensional square mesh, where each node represents a two-state subsystem that follows linearized and discretized swing dynamics

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

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

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

and $[B]_{ii} = \begin{bmatrix} 0 & 1 \end{bmatrix}^\mathsf{T}$ for all i.

Connectivity among nodes is determined at random; each node connects to each of its neighbors with a 40% probability. The expected number of edges is $0.8 * n * (n-1)$. The parameters in bus i: m_i^{-1} (inertia inverse), d_i (damping) and k_{ij} (coupling term) are randomly generated and uniformly distributed between $[0, 2]$, $[0.5, 1]$, and $[1, 1.5]$ respectively. We set the discretization step $\Delta t = 0.2$, and define $k_i := \sum_{j \in \mathbf{in}_i(1)} k_{ij}$.

We study both the nominal setting and robust setting with uniformly distributed polytopic noise. The baseline parameter values are $d = 3$, $T = 5$, $N = 16$ (4×4 grid). Unless otherwise specified, we start with a random-generated initial condition. We use a quadratic cost and polytopic constraints on both angle and frequency deviation, and impose upper and lower bounds.

14

5.2 Performance

The addition of the terminal set and cost to the DLMPC algorithm introduces minimal conservatism in both nominal and robust settings. We study the the DLMPC cost for varying values of the time horizons for three different cases: (i) without terminal set and cost (ii) with terminal set, (iii) with terminal set *and* terminal cost. Results are summarized in Fig. 1. Observe that the difference between the optimal cost across all three cases are negligible, indicating that our proposed terminal set and cost introduce no conservatism (while still providing the necessary theoretical guarantees).

Figure 1: Relative difference of the optimal cost obtained (i) with terminal set (pink) and (ii) with both terminal set and cost (yellow) compared to the optimal cost computed without terminal set and cost. Relative difference is obtained by taking the difference of the two costs and normalizing by the non-terminal-constrained cost. The difference obtained by adding the terminal set (indicated in pink) is on the order of 10^{-5} and is not visible on the plot.

Generally, the inclusion of the terminal set and cost introduce minimal change. In the vast majority of cases (e.g. all simulations from [15]), the DLMPC algorithm is feasible and stable even without a terminal set. This phenomenon has already been observed for the centralized case [1]. However, we also want to demonstrate how the terminal set and cost *can* make a difference – example subsystem trajectories for the three different cases are shown in Fig. 2 for the nominal case and in the presence of polytopic disturbances. For these simulations only, we use a smaller ($N = 5$), more unstable (m_i^{-1} between $[0, \ 16]$) system, extremely short time horizon ($T = 2$), and somewhat hand-crafted initial states and disturbances to obtain clearly visible differences between cases – without such instability, short time horizon, and hand-crafting, differences are generally tiny and not visible. In all cases, the centralized solution (computed via CVX [32]) coincides with the solution achieved by the DLMPC Algorithm 1, validating the optimality of the proposed algorithm. The effects of introducing terminal set and terminal cost are apparent and consistent with the theoretical results presented in this paper.

5.3 Computational complexity

Simulations results verify the scalability of the proposed methods. We measure runtime[6] while varying different network and problem parameters: locality d, network size N, and time horizon T.[7] We run 5

---

[6]In online simulations, runtime is measured after the first iteration, so that all iterations for which runtime is measured are warm-started.

[7]To increase network size, we vary the size of the grid over 4×4 (32 states), 6×6 (72 states), 8×8 (128 states), and 11×11 (242 states) grid sizes.

Figure 2: For both nominal cases, upper and lower bounds for all states are 1 and -1 respectively; indicated by red dashed lines in select plots. **Top (nominal)**: evolution of the states of subsystem 4 under a DLMPC controller without terminal set and cost (green), with terminal set (pink), and with both terminal set and cost (yellow). Without a terminal set, the algorithm becomes infeasible after $t = 2$. With a terminal set, the algorithm remains feasible. Notice the difference in ω_4 for the trajectories at $t = 2$. The addition of the terminal cost introduces little change. **Middle (nominal)**: evolution of the states of subsystem 3 under different initial conditions. Infeasibility is encountered at $t = 15$; here, the difference between the infeasible and feasible trajectories is not visible. Note the oscillations in ω_3; adding a terminal set itself enables feasibility but gives large oscillations – additionally including the terminal cost results in much smaller, decaying oscillations. **Bottom (polytopic)**: upper and lower bounds for θ states are -4 and 4, respectively; bounds for ω states are -20 and 20. Bounds are indicated by red dashed lines in select plots. Without a terminal set, the algorithm becomes infeasible after $t = 13$; with a terminal set, feasibility is maintained. Notice the difference in trajectories with and without terminal set just before $t = 13$. The addition of the terminal cost introduces little change.

16

different simulations for each of the parameter combinations, using different realizations of the randomly chosen parameters to provide consistent runtime estimates.

First, we study the scalability of the offline Algorithm 2 to compute the terminal set; results are shown in Fig. 3. Consistent with theoretical analyses in §4, runtime does not increase with the size of the network; rather, it increases with the size of the neighborhood. As expected, computations for the robust set take slightly longer than for the nominal set, since the variables in the robust setting have greater dimension. Overall, synthesis times for both nominal and robust settings are extremely low, especially when a small locality size is used.

Figure 3: Average runtime of Algorithm 2 with network size (left) and locality parameter (right). The lines are the mean values and the shaded areas show the values within one standard deviation. Since computations are parallelized across subsystems, runtime is measured on a subsystem, normalized per state, and averaged after the algorithm computation is finished.

We also study how scalability of the DLMPC algorithm is affected when we impose a terminal set and terminal cost, and use Algorithm 3 to handle coupling. Results are shown in Fig. 4; this figure was generated using the same systems and parameters as Fig. 2 from our companion paper [15], allowing for direct comparison of online runtimes. The addition of the terminal set/cost slightly increases runtime, as expected. In the nominal case, runtime is increased from about 10^{-3}s to 10^{-1}s. In the case of polytopic disturbances, runtime is increased from about $1 - 10$s to 10s. Scalability is maintained; runtime barely increases with the size of the network. Overall, simulations indicate that the introduction of a terminal set and cost preserve scalability, minimally impact computational overhead and performance, and provide the desired guarantees.

Figure 4: Average runtime per DLMPC iteration with network size when terminal set and terminal cost are imposed. The lines are the mean values and the shaded areas show the values within one standard deviation. Since computations are parallelized across subsystems, runtime is measured on a subsystem, normalized per state, and averaged out after the MPC controller is finished.

6 Conclusion

In this paper we provide theoretical guarantees for the *closed-loop* DLMPC approach, in both nominal and robust settings. In particular, we ensure recursive feasibility and stability by incorporating a terminal set and terminal cost. We also give guarantees for convergence of the algorithm. For the terminal set, we choose the maximal robust positive invariant set, which can be expressed compactly in the SLS parametrization. We introduce an algorithm to scalably compute this terminal set. We also provide the requisite modifications to the online DLMPC algorithm to accommodate local coupling induced by the terminal set and cost, as well as general coupling induced by process cost and constraints. All algorithms require only local information exchange, and enjoy computational complexity that is independent of the global system size. The results presented in this paper are the first to provide a distributed and localized computation of the maximal robust positive invariant control set and Lyapunov function of a large-scale system.

Acknowledgements

References

[1] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[2] D. Q. Mayne, M. M. Seron, and S. V. Rakovic, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, pp. 219 – 224, 2005.

[3] C. Langbort, C. R.S., and D. R., "Distributed control design for systems interconnected over an arbitrary graph," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1502 – 1519, 2004.

[4] A. Jokic and M. Lazar, "On decentralized stabilization of discrete-time nonlinear systems," in *Proc. IEEE ACC*, 2009, pp. 5777–5782.

[5] A. I. Zecevic and D. D. Siljak, *Control of Complex Systems*. New York: Commun. Control Eng., Springer, 1988.

[6] B. T. Stewart, A. Venkat, J. Rawlings, S. Wright, and G. Pannocchia, "Cooperative distributed model predictive control," *Syst. Control Lett.*, vol. 59, no. 8, pp. 460 – 469, 2010.

[7] J. M. Maestre, D. Muñoz de la Peña, E. F. Camacho, and T. Alamo, "Distributed model predictive control based on agent negotiation," *J. Process Control*, vol. 21, no. 5, pp. 685 – 697, 2011.

[8] 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, Jul 2016.

[9] P. A. Trodden and J. M. Maestre, "Distributed predictive control with minimization of mutual disturbances," *Automatica*, vol. 77, pp. 31 – 43, 2017.

[10] G. Darivianakis, A. Eichler, and J. Lygeros, "Distributed model predictive control for linear systems with adaptive terminal sets," *IEEE Trans. Autom. Control*, vol. 65, no. 3, pp. 1044 – 1056, 2020.

[11] A. Aboudonia, A. Eichler, and J. Lygeros, "Distributed model predictive control with asymmetric adaptive terminal sets for the regulation of large-scale systems," 2020. [Online]. Available: https://arxiv.org/abs/2005.04077

[12] S. Muntwiler, K. P. Wabersich, A. Carron, and M. N. Zeilinger, "Distributed model predictive safety certification for learning-based control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5258 – 5265, 2020.

[13] Y. Wang and C. Manzie, "Robust distributed model predictive control of linear systems: analysis and synthesis," 2021. [Online]. Available: https://arxiv.org/abs/2005.04006

[14] 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 CDC*, 2020, pp. 4366–4373.

[15] C. Amo Alonso, J. S. Li, N. Matni, and J. Anderson, "Distributed and localized model predictive control. Part I: Synthesis and implementation," 2021. [Online]. Available: https://arxiv.org/abs/2110.07010

[16] Y.-S. Wang, N. Matni, and J. C. Doyle, "A system-level approach to controller synthesis," *IEEE Trans. Autom. Control*, vol. 64, no. 10, pp. 4079–4093, 2019.

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

[18] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[19] Y. Wang, N. Matni, and J. C. Doyle, "Separable and localized system-level synthesis for large-scale systems," *IEEE Trans. Autom. Control*, vol. 63, no. 12, pp. 4234–4249, Dec. 2018.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[21] C. Amo Alonso and N. Matni, "Distributed and localized closed-loop model predictive control via System Level Synthesis," in *Proc. IEEE CDC*, 2020, pp. 5598–5605.

[22] C. Amo Alonso, J.S. Li, N. Matni, and J. Anderson, "Robust distributed and localized model predictive control," 2021. [Online]. Available: https://arxiv.org/abs/2103.14171

[23] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[24] J. Löfberg, "Minimax approaches to robust modelpredictive control," *PhD thesis, Department of ElectricalEngineering, Linköping University, Sweden*, 2003.

[25] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004.

[26] C. Amo Alonso, D. Ho, and J. Maestre, "Distributed linear quadratic regulator robust to communication dropouts," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 3072 – 3078, 2020.

[27] H. Wang and J. Anderson, "Large-scale system identification using a randomized SVD," 2021. [Online]. Available: https://arxiv.org/abs/2109.02703

[28] J. Yu, Y.-S. Wang, and J. Anderson, "Localized and distributed $\mathcal{H}_2$ state feedback control," 2020. [Online]. Available: https://arxiv.org/abs/2010.02440

[29] E. Gilbert and K. Tan, "Linear systems with state and control constraints: the theory and applications of maximal output admissible sets," *IEEE Trans. Autom. Control*, vol. 36, no. 9, pp. 1008 –1020, 1991.

[30] G. Costantini, R. Rostami, and D. Gorges, "Decomposition Methods for Distributed Quadratic Programming with Application to Distributed Model Predictive Control," in *IEEE Proc. Annu. Allerton Conf. Commun., Control, Comput.*, 2018, pp. 943 – 950.

[31] J. S. Li, "SLS-MATLAB: Matlab toolbox for system level synthesis," 2019. [Online]. Available: https://github.com/sls-caltech/sls-code

[32] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.