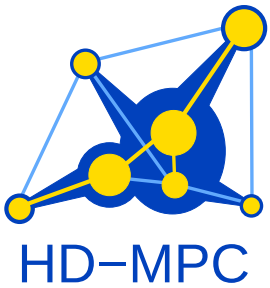
SEVENTH FRAMEWORK PROGRAMME
THEME – ICT
[Information and Communication Technologies]

Contract Number:	223854
Project Title:	Hierarchical and Distributed Model Predictive Control of Large-Scale Systems
Project Acronym:	HD-MPC



Deliverable Number:	D3.3.3
Deliverable Type:	Report
Contractual Date of Delivery:	March 1, 2011
Actual Date of Delivery:	**March 1, 2011**
Title of Deliverable:	**Report on extensive assessment of the developed coordination mechanisms, including case studies**
Dissemination level:	Public
Workpackage contributing to the Deliverable:	WP3
WP Leader:	Wolfgang Marquardt
Partners:	RWTH, TUD, POLIMI, UMC, SUPELEC
Author(s):	H. Scheu, W. Marquardt, M. Farina, R. Scattolini, M.D. Doan, T. Keviczky, B. De Schutter

Table of contents

Project co-ordinator

Name: Bart De Schutter
Address: Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 Delft, The Netherlands
Phone Number: +31-15-2785113
Fax Number: +31-15-2786679
E-mail: b.deschutter@tudelft.nl

Project web site: `http://www.ict-hd-mpc.eu`

Executive Summary

This report presents three coordination mechanisms for distributed MPC with neighbor-to-neighbor communications, i.e. there is no need for a global coordinator.
In the first case, a dual-decomposition scheme based on Fenchel's duality is employed to solve the centralized optimization problem in a distributed way. The distributed method is able to achieve the optimal solution as if it were obtained using a corresponding centralized algorithm. In order to deal with the slow convergence rate, which is a standard issue of dual decomposition approaches, we also introduce an improved algorithm that uses heterogeneous step sizes together with warm-starting, and that can still be implemented using local communications.
The second approach is based on a new sensitivity-driven coordination. The Quadratic Program (QP) resulting for the optimal control problem of the MPC is solved in a distributed way. In each local QP, sensitivity information of neighboring subsystems is included. By this means, the method provides optimality as a centralized solution. In contrast to dual-decomposition approaches, the method can provide fast convergence, provided a proper decomposition is given.
Finally, the third approach is based on a non-iterative scheme with neighbor-to-neighbor communication among the subsystems where partial (local) structural information is needed. The main rationale behind that approach is to transmit among the neighbors the future reference trajectories and to interpret the difference between these trajectories and the true ones as disturbances to be rejected by a proper robust MPC method. Therefore it is not necessary for each subsystem to know the dynamical models governing the trajectories of the other subsystems and the transmission of information is limited.

Chapter 1

Synopsis of the report

1.1 Introduction

Hierarchical and distributed model predictive control relies strongly on the hierarchical or distributed optimization method (see the reports of WP 4 for the related research results) and in particular on the coordination mechanisms used. The coordination mechanism defines how the different controllers in the hierarchical or distributed control topology interact by communication, cooperation or coordination.

This report presents three coordination mechanisms for single layer distributed model predictive control developed in the HD-MPC project:

1. A distributed coordination scheme based on Fenchel's duality,
2. a distributed coordination scheme based on a sensitivity-driven coordination, and
3. a coordination scheme with low communication requirements based on robust MPC methods.

The notion of these coordination mechanisms differs in many ways. In the following three sections, the idea of the methods is briefly explained. Then, this chapter contains general conclusions on the new coordination mechanisms for DMPC. Full descriptions of the research results can be found in Chapters 2 – 4.

1.2 An iterative scheme for distributed model predictive control using Fenchel's duality

In [1] (see also Chapter 2), we present a cooperative distributed MPC approach using neighbor-to-neighbor communications only, i.e. there is no need for a global coordinator. Considering the MPC problem of interacting discrete-time linear subsystems, a dual-decomposition scheme based on Fenchel's duality is employed to solve the centralized optimization problem in a distributed way. The distributed method is able to achieve the optimal solution as if it were obtained using a corresponding centralized algorithm. The method is further improved to achieve faster convergence speed. We also demonstrate the application of our methods in a simulated water network control problem, and discussed the open issues of the proposed scheme.

Consider a plant consisting of M dynamically coupled LTI subsystems. The dynamics of each subsystem are assumed linear and to be influenced directly by only a *small* number of other subsystems.

The centralized MPC problem, named $(\mathscr{P})$, can be cast in a compact form as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T H \mathbf{x} \\ \text{s.t.} \quad & a_l^T \mathbf{x} = b_l, \quad l = 1, \ldots, n_{\text{eq}} \\ & a_l^T \mathbf{x} \leq b_l, \quad l = n_{\text{eq}} + 1, \ldots, s \end{aligned} \tag{1.1}$$

where H is a block-diagonal and positive definite matrix, $s = n_{\text{eq}} + n_{\text{ineq}}$.

1.2.1 Solution approach

Han's parallel method for convex programs

Problem $(\mathscr{P})$ can be solved using Han's parallel algorithm [2], which is a method to decompose the Fenchel's dual problem [3] of a convex optimization problem. Han's method essentially uses iterative projection of the dual variables onto the local constraint sets, this helps splitting the computation into s parallel subproblems, where s is the number of constraints. However, it requires a *global update scheme* and the parallel problems operate with the full-sized decision vector. Implementing the scheme in a DMPC system, where the goal is to reduce the size of local computations and to rely on local communication between subsystems only, is not straightforward. Our main approach is to exploit the structure of the problem $(\mathscr{P})$, resulting in a distributed algorithm that does not require global communications.

Distributed version of Han's method for the MPC problem

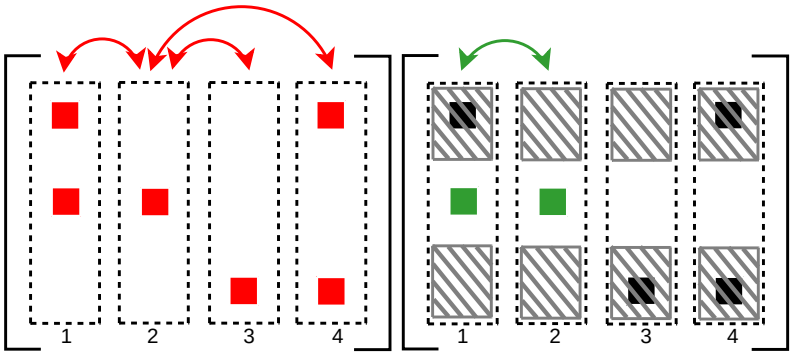
The main idea behind the distributed version of Han's method is illustrated in Figures 1.1(a) and 1.1(b), with a simple system consisting of 4 subsystems and the coupling matrix that shows how subsystems are coupled via their variables (boxes on the same row indicate the variables that are coupled in one constraint). In Han's method using global variables, a subsystem has to communicate with all other subsystems in order to compute the updates of the global variables. For the distributed version of Han's method, each subsystem i only communicates with the other subsystems of which the variables are necessary for computing the updates of its local variables, i.e., the subsystems in its neighborhood.

1.2.2 Main results

In [1], we show that both the proposed distributed algorithm and the original Han's method generate the same updates. This allows us to implement a DMPC scheme using the distributed algorithm, which only need neighbor-to-neighbor communications. We also show that feasibility, and stability properties of the DMPC scheme are achieved upon convergence of the iteration at every time step.

One typical drawback of dual decomposition-based approaches is the slow convergence rate, which is also a valid issue of Han's method and the distributed variation. In order to use this algorithm in MPC, we need to speed up the convergence of the iteration so that problem $(\mathscr{P})$ can be solved within each sampling period. We present an improved distributed version of Han's method that employs warm-starting and acceleration using heterogeneous step sizes. Its efficiency is demonstrated in the example of irrigation canal control, which is described in detail in Chapter 2.

To conclude, we summarize open issues of using Han's method and other dual decomposition methods for MPC, including the topics of distributed formulation, convergence rate, primal feasibility, and stability of MPC. These issues are used to recommend future research directions.



(a) (b)

Figure 1.1: Illustration of communication links with (a) the centralized coordination version and (b) the distributed coordination version of Han's algorithm for an example 4-subsystem problem. In (a), an update for a global variable requires the 2nd subsystem to communicate with all the others. In (b), the 2nd subsystem only cares about its local variable, therefore it should only communicate with the 1st subsystem.

1.3 Sensitivity-driven coordination

The idea of sensitivity-driven coordination has first been studied in [4] in the context of nonlinear optimal control, where the method has been applied to control a four-tanks system. Due to the promising results, we have continued research on this method as reported in [5] and [6] in the context of linear quadratic constrained optimal control.

The main idea is to transcribe the continuous-time (see [5]) or discrete-time (see [6]) optimal control problem into constrained parametric optimization problem. For general problems, this results in an NLP

$$\min_{p} \sum_{i=1}^{N} \Phi_i(p), \tag{1.2}$$

$$\text{s.t. } c_i(p) \geq 0, \quad \forall i, \tag{1.3}$$

with objective function $\Phi = \sum_{i=1}^{N} \Phi_i(p)$ and the set of constraint functions $c_i(p), \forall i$. Here, p defines the parameters of the NLP, which are divided into N subvectors p_i, and i indicates the subsystem. For the linear quadratic optimal control problem a QP is derived. The QP (or in general, the NLP) is then solved in a distributed manner. The subproblems are coordinated driven by sensitivities of the objective function and the constraint function. Hence, the decomposition results in the set of NLPs [5]

$$\min_{p_i} \Phi_i^*(p) \tag{1.4}$$

$$\text{s.t. } c_i(p) \geq 0, \tag{1.5}$$

with objective functions

$$\Phi_i^* = \Phi_i(p) + \left[\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\partial \Phi_j}{\partial p_i}\bigg|_{p^{[k]}}^{T} - \lambda_j^{[k]\,T} \frac{\partial c_j}{\partial p_i}\bigg|_{p^{[k]}} \right] (p_i - p_i^{[k]}), \tag{1.6}$$

which are solved iteratively. Under certain assumptions, the solution of the distributed algorithm will converge to solution of the original optimization problem [5, 6].

The sensitivity-driven coordination-mechanism has been assessed on several case studies, e.g.

- a simulated nonlinear distributed chemical plant for the alkylation of benzene [5],
- a real implementation of Johansson's quadruple tank benchmark [7], where it has been compared to a set of other predictive controllers [8], and
- an synthetic unstable process [6].

Exemplarily, the results of the last contribution can be found in Chapter 3. The resulting distributed model predictive control scheme is referred to as sensitivity-driven distributed MPC (or S-DMPC for short). S-DMPC is an optimal controller, i.e. it reproduces the same trajectories as an centralized MPC. The main challenges of future work are to guarantee stability and convergence of the proposed scheme, as suggested in [5].

1.4 Distributed non-cooperative MPC

A new state feedback distributed control algorithm, based on model predictive control (MPC), is here proposed: Distributed Predictive Control (henceforth called DPC) [9]. It is based on a non iterative scheme with neighbor-to-neighbor (i.e., partially connected) communication among the subsystems where partial (local) structural information are needed, and is deeply inspired to the robust state feedback MPC approach first introduced in [10].
The main rationale behind DPC is to transmit among the neighbors the future reference trajectories and to interpret the difference between these trajectories and the true ones as disturbances to be rejected by a proper robust MPC method. Therefore in DPC it is not necessary for each subsystem to know the dynamical models governing the trajectories of the other subsystems and the transmission of information is limited; moreover joint constraints between the subsystems could be included, so that DPC can also be used for control of independent systems with coupling constraints. Finally, convergence results can be established.

An off-line design phase must be carried out in order to apply the DPC algorithm:
1) Define a decentralized control law (i.e., the auxiliary control law) which, at the same time, (a) stabilizes the local subsystems when neglecting the interconnections, (b) stabilizes the overall large scale system, (c) has a Lyapunov function which basically corresponds to a weighted sum of local Lyapunov functions. The above-mentioned issues can be addressed using a number of well-established results, worked out in the past in the field of decentralized control. For instance, one can rely on milestone results on *connective stability* [11], vector Lyapunov functions and the so-called "weighted sum approach" for proving connective stability [12]. More recently, problems (a) and (b) have been successfully addressed in [13], where a small gain condition for large-scale (nonlinear) systems has been derived.
2) Set the stage and final cost functions.
3) Define the proper sets to constrain the state and input trajectories using set-theoretic considerations.
4) For each subsystem $i = 1, \ldots, M$, define an initial reference state trajectory.

Once the cost functions and the constraining sets are properly defined, the minimization problems to be solved on-line correspond to low-order MPC problems, defining local subsystem's inputs. Note that the reference trajectory, for each subsystem, is incrementally defined.

Concluding, DPC enjoys the following properties:

Classes of distributed systems. DPC has been designed for controlling a wide class of large scale systems. Specifically, it can be used for dynamically coupled subsystems, as well as for subsystems with coupled constraints on the state variables.

Local knowledge. Each subsystem is required to know and store only the model governing the dynamics of its local state variables, and how the outputs of the neighboring subsystems influence them.

Information transmission. DPC requires neighbor-to-neighbor communication among subsystems. Furthermore, the data transmitted to each subsystem correspond only to the value of the local reference state variable N-steps ahead (which is a n_i dimensional variable). Finally, information must be transmitted and received with non-iterative communication protocol i.e, once within a sampling time, and no negotiation is required among subsystems.

Scalability. The DPC algorithm is scalable: as the number of subsystems grows (while, for instance, the average number of neighbors for each subsystem remains constant), the information required to be stored, processed and transmitted by each subsystem not linked to the new subsystems remains constant.

Guaranteed properties. The convergence of the overall system controlled with DPC is guaranteed under suitable assumptions, and both local and global constraints on state and input variables are handled.

1.5 General conclusions

We have introduced three coordination methods for distributed MPC, which are all based on neighbor-to-neighbor communication. While the scheme based on Fenchel's duality and the sensitivity-based coordination reproduce the optimality of a centralized controller, the non-cooperative DPC scheme features low-communication requirements and does not rely on an iteration of the solution, which may be advantageous. While S-DMPC features fast convergence properties, extensions are necessary, e.g. to guarantee closed-loop stability. For DMPC on Fenchel's duality, speed of convergence could be improved, however there is still a desire to speed up convergence. Furthermore, there is a desire to implement less conservative stability property than the zero-terminal constraint. The non-cooperative DPC approach provides some conservatism for a caused by its robustness approach. However, as a consequence DPC features several advantages over the other schemes (and also compared to many others in literature), i.e. it is a non-iterative approach, and only local information is required for each controller. However, there are also still important issues to solve in future, i.e. to provide appropriate sets for the control errors in order to measure the uncertainties of the coupling variables.

Chapter 2

An iterative scheme for distributed model predictive control using Fenchel's duality

This chapter is based on a paper that has been accepted for publication in the Journal of Process Control, Special Issue for Hierarchical and Distributed Model Predictive Control, with particular citation: M.D. Doan, et al., An iterative scheme for distributed model predictive control using Fenchel's duality, J. Process Control (2011), doi: 10.101/j.jprocont.2010.12.009.

2.1 Introduction

Nowadays, Model Predictive Control (MPC) is widely used for controlling industrial processes [14], and it also has been studied thoroughly by the scientific community [15, 16, 17]. MPC can naturally handle operational constraints and, moreover, it is designed for multi-input multi-output systems, both of which contributed to the popularity of MPC. Another advantage of MPC is that it relies on optimization techniques to solve the control problem. Hence, improvements in optimization techniques can help to broaden the applications of MPC for more complex problems.

When considering a control problem for a large-scale networked system (such as complex manufacturing or infrastructure processes), using MPC in a centralized fashion may be considered impractical and unsuitable due to the computational burden and the requirement of global communications across the network. It is also inflexible against changes of network structure and the limitation of information exchange between different authorities who might be in control of a local subsystem. In order to deal with these limitations, Distributed MPC (DMPC) has been proposed for control of such large-scale systems, by decomposing the overall system into small subsystems [18, 19]. The subsystems then employ distinct MPC controllers that only solve local control problems, use local information from neighboring subsystems, and collaborate to achieve globally attractive solutions.

DMPC is an emerging topic for scientific research. The open issues of DMPC have recently been discussed in [20, 21]. Several DMPC methods were proposed for different problem setups. For systems with decoupled dynamics, a DMPC scheme for multiple vehicles with coupled cost functions was proposed in [22], utilizing predicted trajectories of the neighbors in each subsystem's optimization. A DMPC scheme with a sufficient stability test for dynamically decoupled systems was presented in [23], in which each subsystem optimizes also over the behaviors of its neighbors. In [24], Richards and How proposed a robust DMPC method for decoupled systems with coupled constraints, based on constraint tightening and a serial solution approach. For systems with coupled dynamics and decoupled constraints, a DMPC scheme has been developed in [25] based on a Jacobi algorithm that deals

with the primal problem, using a convex combination of new and old solutions. In [26], the neighboring subsystem states are treated as bounded contracting disturbances, and each subsystem solves a min-max problem. A partitioning-based algorithm was proposed in [27, 28], with sufficient conditions for the a posteriori stability analysis. In [29], Li et al. proposed an algorithm with stability conditions in which subproblems are solved in parallel in order to get a Nash equilibrium. Several DMPC algorithms based on decomposing of the global optimization problems were proposed in [30, 31, 32]. Other recent work on applications of DMPC is reported in [33, 34, 35].

In this paper, we present a decomposition scheme based on Han's parallel method [36, 2], aiming to solve the centralized optimization problem of MPC in a distributed way. This approach results in two distributed algorithms that are applicable to DMPC of large-scale industrial processes. The main ideas of our algorithms are to find a distributed update method that is equivalent to Han's method (which relies on global communications), and to improve the convergence speed of the algorithm [37]. We will demonstrate the application of our methods in a simulated water network control problem. The open issues of the proposed scheme will be discussed to formulate future research directions.

The paper is organized as follows. The MPC problem is formulated and the underlying optimization problem is stated in Section 2.2. In Section 2.3, we summarize Han's parallel method for convex programs [2] as the starting point for our approach. In Section 2.4, we present two distributed MPC schemes that exploit the structure of the optimization problem for local communications. The first DMPC scheme uses a distributed iterative algorithm that we prove to be equivalent to Han's algorithm. As a consequence of this equivalence, the proposed DMPC scheme achieves the global optimum upon convergence and thus inherits feasibility and stability properties from its centralized MPC counterpart. The second DMPC scheme is an improved algorithm that aims to speed up the convergence of the distributed approach. In Section 2.5, we illustrate the application of the new DMPC schemes in an example system involving irrigation canals. In Section 2.6, we discuss the open issues of Han's method and other dual decomposition techniques for DMPC that motivate directions for future research. Section 2.7 concludes the paper.

2.2 MPC problem formulation

2.2.1 Subsystems and their neighborhood

Consider a plant consisting of M dynamically coupled subsystems. The dynamics of each subsystem are assumed linear and to be influenced directly by only a *small* number of other subsystems. Moreover, each subsystem i is assumed to have local linear coupled constraints involving only variables from a small number of other subsystems.

Based on the couplings, we define the 'neighborhood' of subsystem i, denoted as $\mathscr{N}^i$, as the set including i and the indices of subsystems that have either a direct dynamical coupling or a constraint coupling with subsystem i.

2.2.2 Coupled subsystem model

We assume that each subsystem can be represented by a discrete-time, linear time-invariant model of the form[1]:

$$x_{k+1}^i = \sum_{j \in \mathcal{N}^i} A^{ij} x_k^j + B^{ij} u_k^j, \tag{2.1}$$

where $x_k^i \in \mathbb{R}^{n^i}$ and $u_k^i \in \mathbb{R}^{m^i}$ are the states and control inputs of the i-th subsystem at time step k, respectively.

2.2.3 Linear coupled constraints

Each subsystem i is assumed to have local linear coupled constraints involving only variables within its neighborhood $\mathcal{N}^i$. Within one prediction period, all constraints that subsystem i is involved in can be written in the following form

$$\sum_{j \in \mathcal{N}^i} \sum_{k=0}^{N-1} D_k^{ij} x_k^j + E_k^{ij} u_k^j = c_{\text{eq}} \tag{2.2}$$

$$\sum_{j \in \mathcal{N}^i} \sum_{k=0}^{N-1} \bar{D}_k^{ij} x_k^j + \bar{E}_k^{ij} u_k^j \leq \bar{c}_{\text{ineq}} \tag{2.3}$$

in which N is the prediction horizon, c_{eq} and $\bar{c}_{\text{ineq}}$ are column vectors, and D_k^{ij}, E_k^{ij}, $\bar{D}_k^{ij}$, and $\bar{E}_k^{ij}$ are matrices with appropriate dimensions.

2.2.4 First MPC problem

We will formulate the centralized MPC problem for systems of the form (2.1) using a terminal point constraint approach that imposes constraints to zero out all terminal states. Under the conditions that a feasible solution of the centralized MPC problem exists, and that the point with zero states and inputs is in the relative interior of the constraint set, this MPC scheme ensures feasibility and stability, as shown in [16] and [38]. However, the algorithm proposed in this paper will also work with any other centralized MPC approach that does not require a terminal point constraint, provided that the subsystems have local stabilizing terminal controllers. We will further assume without loss of generality that the initial time is zero.

The optimization variable of the centralized MPC problem is constructed as a stacked vector of predicted subsystem control inputs *and* states over the prediction horizon:

$$\begin{aligned} \mathbf{x} = \Big[&\left(u_0^1\right)^T, \ldots, \left(u_0^M\right)^T, \ldots, \left(u_{N-1}^1\right)^T, \ldots, \left(u_{N-1}^M\right)^T, \\ &\left(x_1^1\right)^T, \ldots, \left(x_1^M\right)^T, \ldots, \left(x_N^1\right)^T, \ldots, \left(x_N^M\right)^T \Big]^T \end{aligned} \tag{2.4}$$

Recall that n^i and m^i denote the numbers of states and inputs of subsystem i. The number of optimization variables for the centralized problem is thus:

$$n_{\mathbf{x}} = N \sum_{i=1}^{M} m^i + N \sum_{i=1}^{M} n^i \tag{2.5}$$

[1] This system description is chosen for simplicity of exposition and our framework can be easily extended to consider output signals with appropriate observability assumptions.

The cost function of the centralized MPC problem is assumed to be decoupled and convex quadratic:

$$J = \sum_{i=1}^{M} \sum_{k=0}^{N-1} \left(\left(u_k^i\right)^T R_i u_k^i + \left(x_{k+1}^i\right)^T Q_i x_{k+1}^i \right) \tag{2.6}$$

with positive definite weights R_i, Q_i. This cost function can be rewritten using the decision variable $\mathbf{x}$ as

$$J = \mathbf{x}^T H \mathbf{x} \tag{2.7}$$

in which the Hessian H is an appropriate block-diagonal, positive definite matrix.

Remark 2.2.1 *The positive definiteness assumption on Q_i and R_i and the choice of the centralized variable as described in* (2.4) *without eliminating state variables will help to compute the inverse of the Hessian easily, by allowing simple inversion of each block on the diagonal of the Hessian.*

The centralized MPC problem, named $(\mathscr{P})$, is defined as: the minimization of (2.6), subject to (2.1) for $i = 1,\ldots,M, k = 0,\ldots,N-1$, (2.2) and (2.3) for $i = 1,\ldots,M$, as well as $x_N^i = 0$ for $i = 1,\ldots,M$.

We can rewrite problem $(\mathscr{P})$ in a compact form as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T H \mathbf{x} \\ \text{s.t.} \quad & a_l^T \mathbf{x} = b_l, \quad l = 1,\ldots,n_{\text{eq}} \\ & a_l^T \mathbf{x} \leq b_l, \quad l = n_{\text{eq}}+1,\ldots,s \end{aligned} \tag{2.8}$$

with $s = n_{\text{eq}} + n_{\text{ineq}}$. The algorithms to be described in the next sections will focus on how to solve this optimization problem.

2.3 Han's parallel method for convex programs

Han's algorithm [2] is a method to decompose the Fenchel's dual problem [3]. Fenchel's duality theorem aims at minimizing a difference $f(\mathbf{x}) - g(\mathbf{x})$, where f is a convex function and g is a concave function. A special case of this problem is minimizing f over a constraint set C, where g is a penalty function for violating the constraint. In Han's problem, the set C is the intersection of local constraint sets, and the dual variables are iteratively projected onto the local constraint sets. As a consequence, the sum of the dual variables converges to the minimizer of the Fenchel's dual problem [2]. In this section, we summarize the main elements of Han's parallel method, followed by a simplified version for the case of definite quadratic programming.

2.3.1 Han's algorithm for general convex problems

The class of optimization problems tackled by Han's algorithm is the following:

$$\begin{aligned} \min_{\mathbf{x}} \quad & q(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in C \triangleq C_1 \cap \cdots \cap C_s \end{aligned} \tag{2.9}$$

where $C_1, \cdots, C_s$ are closed convex sets and $C \neq \varnothing$, and where $q(\mathbf{x})$ is uniformly convex[2] and differentiable on $\mathbb{R}^{n_\mathbf{x}}$.

A problem of type (2.9) can be solved by Han's algorithm. In the following algorithm we will describe Han's method, which is an iterative procedure. We use p as iteration counter of the algorithm, and the superscript (p) for variables that are computed at iteration p.

Algorithm 2.3.1 ***Han's algorithm for convex programs***

Let α be a sufficiently large number[3] and define $\boldsymbol{y}^{(0)} = \boldsymbol{y}_1^{(0)} = \cdots = \boldsymbol{y}_s^{(0)} = 0$, with $\boldsymbol{y}^{(0)}, \boldsymbol{y}_l^{(0)} \in \mathbb{R}^{n_\mathbf{x}}, l = 1, \ldots, s$, and $\boldsymbol{x}^{(0)} = \nabla q^* \left(\boldsymbol{y}^{(0)}\right)$ with q^* being the conjugate function[4] of q. For $p = 1, 2, \ldots$, we perform the following computations:

1) For $l = 1, \ldots, s$, find $\boldsymbol{z}_l^{(p)}$ that solves

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \frac{1}{2} \left\| \boldsymbol{z} + \alpha \boldsymbol{y}_l^{(p-1)} - \boldsymbol{x}^{(p-1)} \right\|_2^2 \\ s.t. \quad & \boldsymbol{z} \in C_l \end{aligned} \tag{2.10}$$

2) Assign

$$\boldsymbol{y}_l^{(p)} = \boldsymbol{y}_l^{(p-1)} + (1/\alpha) \left(\boldsymbol{z}_l^{(p)} - \boldsymbol{x}^{(p-1)} \right) \tag{2.11}$$

3) Set $\boldsymbol{y}^{(p)} = \boldsymbol{y}_1^{(p)} + \cdots + \boldsymbol{y}_s^{(p)}$

4) Compute

$$\boldsymbol{x}^{(p)} = \nabla q^* \left(\boldsymbol{y}^{(p)} \right) \tag{2.12}$$

In [2], Han and Lou also showed that Algorithm 2.3.1 converges to the global optimum if the conditions on q and C mentioned after (2.9) are satisfied.

Remark 2.3.2 *Han's method essentially solves the dual problem of* (2.9)*, so that $\boldsymbol{y}^{(p)}$ converges to the solution of the Fenchel's dual problem:*

$$\min_{\boldsymbol{y}} \left(q^*(\boldsymbol{y}) - \delta^*(\boldsymbol{y}|C) \right) \tag{2.13}$$

in which $\delta(\boldsymbol{x}|C)$ is the indicator function, which is 0 if $\boldsymbol{x} \in C$ and ∞ otherwise. The conjugate function of $\delta(\boldsymbol{x}|C)$ is $\delta^*(\boldsymbol{y}|C) = \sup_{\boldsymbol{x} \in C} \boldsymbol{y}^T \boldsymbol{x}$. According to Fenchel's duality theorem [3], the minimum of the convex problem $f(\boldsymbol{x}) - g(\boldsymbol{x})$, where f is a convex function on $\mathbb{R}^{n_\mathbf{x}}$ and g is a concave function on $\mathbb{R}^{n_\mathbf{x}}$, equals the maximum of the concave problem $g^*(\boldsymbol{y}) - f^*(\boldsymbol{y})$, or equivalently the minimum of $f^*(\boldsymbol{y}) - g^*(\boldsymbol{y})$. In this situation $f \equiv q$ and $g \equiv \delta$. A value $\boldsymbol{y}^{(\bar{p})}$ achieved when Algorithm 2.3.1 converges is an optimizer of (2.13)*, hence $\boldsymbol{x}^{(\bar{p})} = \nabla q^* \left(\boldsymbol{y}^{(\bar{p})} \right)$ is the solution of* (2.9)*.*

---

[2] A function $q(\mathbf{x})$ is uniformly convex (or strongly convex) on a set S if there is a constant $\rho > 0$ such that for any $\mathbf{x}_1,\ \mathbf{x}_2 \in S$ and for any $\lambda \in (0,1)$:

$$q(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda q(\mathbf{x}_1) + (1-\lambda) q(\mathbf{x}_2) - \rho \lambda (1-\lambda) \|\mathbf{x}_1 - \mathbf{x}_2\|^2.$$

[3] α is a design parameter that has to be sufficiently large. With $\alpha \geq s/\rho$ Han's method will converge [2]. For positive definite QPs we can choose ρ as one half of the smallest eigenvalue of the Hessian matrix.

[4] The conjugate function of a function $q(\mathbf{x}),\ \mathbf{x} \in \mathbb{R}^{n_\mathbf{x}}$ is defined by: $q^*(\mathbf{y}) = \sup_{\mathbf{x} \in \mathbb{R}^{n_\mathbf{x}}} \left(\mathbf{y}^T \mathbf{x} - q(\mathbf{x}) \right)$. The conjugate function q^* is always convex [39].

2.3.2 Han's algorithm for definite quadratic programs

In case the optimization problem has a positive definite cost function and linear constraints as in (2.9), the optimization problem (2.10) and the derivative of conjugate function (2.12) have analytical solutions, and then Han's method becomes simpler. In the following we show how the analytical solutions of (2.10) and (2.12) can be obtained when applying Algorithm 2.3.1 to the problem (2.8).

Remark 2.3.3 *The result of simplifying Han's method in this section is slightly different from the original one described in [2], so as to correct the minor mistakes we found in that paper.*

As in (2.9), each constraint $\mathbf{x} \in C_l$ is implicitly expressed by a scalar linear equality or inequality constraint. So (2.10) takes one of the following two forms:

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{z} + \alpha \mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\|_2^2 \\ \text{s.t.} \quad & a_l^T \mathbf{z} = b_l \end{aligned} \tag{2.14}$$

or

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{z} + \alpha \mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\|_2^2 \\ \text{s.t.} \quad & a_l^T \mathbf{z} \leq b_l \end{aligned} \tag{2.15}$$

Let us first consider (2.15):

- If $a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) \leq b_l$, then $\mathbf{z}_l^{(p)} = \mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}$ is the solution of (2.15). Substituting this $\mathbf{z}_l^{(p)}$ into (2.11), leads to the following update of $\mathbf{y}_l^{(p)}$:

$$\begin{aligned} \mathbf{y}_l^{(p)} &= \mathbf{y}_l^{(p-1)} + (1/\alpha)\left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)} - \mathbf{x}^{(p-1)}\right) \\ \Rightarrow \mathbf{y}_l^{(p)} &= 0 \end{aligned} \tag{2.16}$$

- If $a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) > b_l$, then the constraint is active. The optimization problem (2.15) aims to find the point in the half-space $a_l^T \mathbf{z} \leq b_l$ that minimizes its distance to the point $\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}$ (which is outside that half-space). The solution is the projection of the point $\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}$ on the hyperplane $a_l^T \mathbf{z} = b_l$, which is given by the following formula:

$$\mathbf{z}_l^{(p)} = \mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)} - \frac{a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l\right) - b_l}{a_l^T a_l} a_l \tag{2.17}$$

Substituting this $\mathbf{z}_l^{(p)}$ into (2.11), leads to:

$$\begin{aligned} \mathbf{y}_l^{(p)} &= \mathbf{y}_l^{(p-1)} + \frac{1}{\alpha}\left(-\alpha \mathbf{y}_l^{(p-1)} - \frac{a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) - b_l}{a_l^T a_l} a_l\right) \\ &= -\frac{a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) - b_l}{\alpha a_l^T a_l} a_l \end{aligned} \tag{2.18}$$

Then defining $\gamma_l^{(p)} = a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) - b_l$ yields

$$\mathbf{y}_l^{(p)} = -\frac{\gamma_l^{(p)}}{\alpha a_l^T a_l} a_l \tag{2.19}$$

If we define

$$\gamma_l^{(p)} = \max\{a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) - b_l, 0\} \tag{2.20}$$

then we can use the update formula (2.19) for both cases.

Similarly, for the minimization under equality constraint (2.14), we define

$$\gamma_l^{(p)} = a_l^T \left(\mathbf{x}^{(p-1)} - \alpha \mathbf{y}_l^{(p-1)}\right) - b_l \tag{2.21}$$

and the update formula (2.19) gives the result of (2.11).

Now we consider step 4) of Algorithm 2.3.1. As shown in [39], the function $q(\mathbf{x}) = \mathbf{x}^T H \mathbf{x}$ with H being a positive definite matrix, is uniformly convex on $\mathbb{R}^{n_\mathbf{x}}$ and has the conjugate function:

$$q^*(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T H^{-1} \mathbf{y} \tag{2.22}$$

$$\Rightarrow \nabla q^*(\mathbf{y}) = H^{-1}\mathbf{y} \tag{2.23}$$

Consequently, in Han's algorithm for the definite quadratic program (2.8), it is not necessary to compute $\mathbf{z}^{(p)}$, and $\mathbf{y}^{(p)}$ can be eliminated using (2.19). We are now ready to describe the simplified Han's algorithm for problem (2.8), with the choice $\alpha = s/\rho$ (cf. footnote 3).

Algorithm 2.3.4 ***Han's algorithm for definite quadratic programs***

For each $l = 1, \ldots, s$, compute

$$c_l = \frac{-1}{\alpha a_l^T a_l} H^{-1} a_l \tag{2.24}$$

Initialize $\gamma_1^{(0)} = \cdots = \gamma_s^{(0)} = 0$ and $\boldsymbol{x}^{(0)} = 0$. For $p = 1, 2, \ldots$, perform the following computations:

1) *For each l corresponding to an equality constraint ($l = 1, \ldots, n_{\text{eq}}$), compute $\gamma_l^{(p)} = a_l^T \boldsymbol{x}^{(p-1)} + \gamma_l^{(p-1)} - b_l$.*
   For each l corresponding to an inequality constraint ($l = n_{\text{eq}}+1, \ldots, s$), compute $\gamma_l^{(p)} = \max\{a_l^T \boldsymbol{x}^{(p-1)} + \gamma_l^{(p-1)} - b_l, 0\}$;

2) *Set*

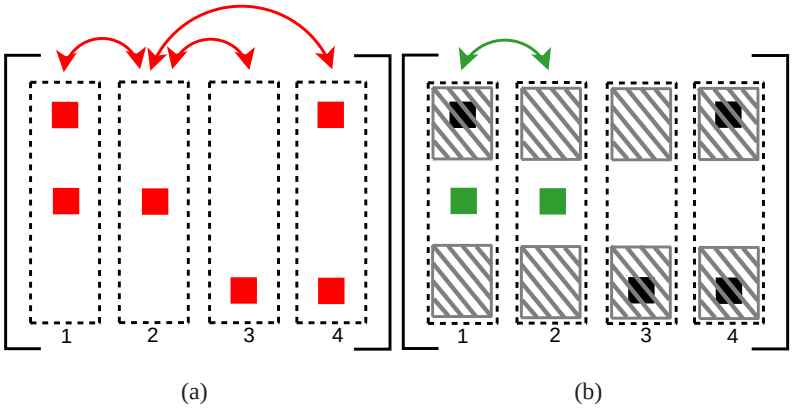
$$\boldsymbol{x}^{(p)} = \sum_{l=1}^{s} \gamma_l^{(p)} c_l \tag{2.25}$$

Note that Han's method splits up the computation into s parallel subproblems, where s is the number of constraints. However, although Algorithm 2.3.4 is simpler than the original form in Algorithm 2.3.1, it still requires a *global update scheme* and the parallel problems still operate with the full-sized decision vector. Implementing the scheme in a DMPC system, where the goal is to reduce the size of local computations and to rely on local communication between subsystems only, is not straightforward. In the following section, we will exploit the structure of the problem (2.8), resulting in a distributed algorithm that does not require global communications.

2.4 Distributed version of Han's method for the MPC problem

2.4.1 Distributed version of Han's method with common step size

The main idea behind the distributed version of Han's method is illustrated in Figures 2.1(a) and 2.1(b), with a simple system consisting of 4 subsystems and the coupling matrix that shows how subsystems are coupled via their variables (boxes on the same row indicate the variables that are coupled in one constraint). In Han's method using global variables, a subsystem has to communicate with all other subsystems in order to compute the updates of the global variables. For the distributed version of Han's method, each subsystem i only communicates with the other subsystems of which the variables are necessary for computing the updates of its local variables, i.e., the subsystems in its neighborhood $\mathcal{N}^i$.



(a) (b)

Figure 2.1: Illustration of communication links with (a) the centralized coordination version and (b) the distributed coordination version of Han's algorithm for an example 4-subsystem problem. In (a), an update for a global variable requires the 2nd subsystem to communicate with all the others. In (b), the 2nd subsystem only cares about its local variable, therefore it should only communicate with the 1st subsystem.

For the algorithm presented in this section, we use M local controllers attached to M subsystems. Each controller i then computes $\gamma_l^{(p)}$ with regards to a small set of constraints indexed by $l \in L_i$, where L_i is a set of indices[5] of several constraints that involve subsystem i. Subsequently, it performs a local update for its own variables, such that the parallel local update scheme will be equivalent to the global update scheme in Algorithm 2.3.4.

Initialization of the algorithm

Store invariant parameters

The parameter α is chosen as in Algorithm 2.3.4 and stored in the memory of all local controllers.

We also compute s invariant values c_l as in (2.24), in which each c_l corresponds to one constraint of (2.8). Note that H is block-diagonal, H^{-1} can be computed easily by inverting each block of H and has the same block structure as H. Hence c_l is as sparse as the corresponding a_l. We can see that c_l can be computed locally by a local controller with a priori knowledge of the parameter a_l and the weighting blocks on the diagonal of H that correspond to the non-zero elements of a_l.

[5]The choice of L_i will be described in the next section.

We assume that each local controller i knows its local dynamics, and the input and state weights of its neighbors in the cost function. Then each local controller i can compute the c_l values associated with its dynamic equality constraints.

Assign responsibility of each local controller

Each local controller is in charge of updating the variables of its subsystem. Moreover, we also assign to each local controller the responsibility of updating *some* intermediate variables that relate to several equality or inequality constraints in which its subsystem's states or inputs appear. The control designer has to assign each of the s scalar constraints to one of the M local controllers[6] such that the following requirements are satisfied:

- Each constraint is taken care of by one and only one local controller (even for a coupled constraint, there will be only one controller that is responsible).
- A local controller can only be in charge of constraints that involve its own variables.

Let L_i denote the set of indices l that local controller i is in charge of[7]. We also define $L_{\mathcal{N}^i}$ as the set of indices l corresponding to the constraints that are taken care of by subsystem i or by any neighbor of i:

$$L_{\mathcal{N}^i} = \bigcup_{j \in \mathcal{N}^i} L_j \tag{2.26}$$

If a local controller i is in charge of the constraints indexed by $l \in L_i$, then it computes locally c_l using (2.24) and exchanges these values with its neighbors. Then each local controller i stores $\{c_l\}_{l \in L_{\mathcal{N}^i}}$ in its memory throughout the optimization process.

Iterative procedure

The distributed algorithm consists of an iterative procedure running within each sampling interval. At each iteration, four steps are executed: two steps are communications between each local controller and its direct neighbors, and two are computation steps that are performed locally by the controllers in parallel. Since feasibility is only guaranteed upon convergence of Han's algorithm, we assume that the sampling time used is large enough such that the algorithm can converge within one sampling interval. This assumption will be used in Proposition 2.4.7, and its restrictiveness will be discussed in Section 2.6.

In this algorithm description, p is used to denote the iteration step. Values of variables obtained at iteration p are denoted with superscript (p).

Definition 2.4.1 (Index matrix of subsystems) *In order to present the algorithm compactly, we introduce the index matrix of subsystems: each subsystem i is assigned a square diagonal matrix $\mathcal{I}^i \in \mathbb{R}^{n_x \times n_x}$, with an entry on the diagonal being 1 if it corresponds to the position of a variable of subsystem i in the vector $\mathbf{x}$, and 0 otherwise. In short, $\mathcal{I}^i$ is a selection matrix such that the multiplication $\mathcal{I}^i \mathbf{x}$ only retains the variables $u_0^i, \ldots, u_{N-1}^i, x_1^i, \ldots, x_N^i$ of subsystem i in its nonzero entries.*

[6] Note that s, the total number of constraints, is often much larger than M.

[7] Note that this partitioning is not unique and has to be created according to a procedure that is performed in the initialization phase.

From Definition 2.4.1 it follows that:

$$\sum_{i=1}^{M} \mathfrak{I}^i = I \tag{2.27}$$

Definition 2.4.2 (Self-image) *We denote with $\mathbf{x}^{(p)|i} \in \mathbb{R}^{n_x}$ the vector that has the same size as $\mathbf{x}$, containing $u_0^{i,(p)}, \ldots, u_{N-1}^{i,(p)}, x_1^{i,(p)}, \ldots, x_N^{i,(p)}$ (i.e. the values of i's variables computed at iteration p) at the right positions, and zeros for the other entries. This vector is called the self-image of $\mathbf{x}^{(p)}$ made by subsystem i.*

Using the index matrix notation, the relation between $\mathbf{x}^{(p)|i}$ and $\mathbf{x}^{(p)}$ is:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)} \tag{2.28}$$

Definition 2.4.3 (Neighborhood image) *Extending the concept of self-image, we denote with $\mathbf{x}^{(p)|\mathcal{N}^i}$ the neighborhood image of subsystem i made from $\mathbf{x}$. At step p of the iteration, subsystem i constructs $\mathbf{x}^{(p)|\mathcal{N}^i}$ by putting the values of its neighbors' variables and its own variables into the right positions, and filling in zeros for the remaining slots of $\mathbf{x}$. The neighborhood image $\mathbf{x}^{(p)|\mathcal{N}^i}$ satisfies the following relations:*

$$\mathbf{x}^{(p)|\mathcal{N}^i} = \sum_{j \in \mathcal{N}^i} \mathbf{x}^{(p)|j} \tag{2.29}$$

$$\mathbf{x}^{(p)|\mathcal{N}^i} = \left(\sum_{j \in \mathcal{N}^i} \mathfrak{I}^j \right) \mathbf{x}^{(p)} \tag{2.30}$$

By definition, we also have the following relation between the self-image and the neighborhood image made by the same subsystem:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)|\mathcal{N}^i} \tag{2.31}$$

Using the notation described above, we now describe the subtasks that each controller will use in the distributed algorithm.

- **Communications with the neighbors**

  Each controller i communicates only with its neighbors $j \in \mathcal{N}^i$ to get updated values of their variables and sends its updated variables to them. The data that each subsystem i transmits to its neighbor $j \in \mathcal{N}^i$ consists of the self-image $\mathbf{x}^{(p)|i}$ and the intermediate variables $\gamma_l^{(p)}, l \in L_i$, which are maintained locally by subsystem i.

- **Update intermediate variables γ_l**

  When the local controller i updates γ_l corresponding to each constraint $l \in L_i$ under its responsibility, it computes in the following manner:

  – If constraint l is an equality constraint ($l \in \{1, \ldots, n_{\text{eq}}\}$), then

$$\gamma_l^{(p)} = a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l \tag{2.32}$$

– If constraint l is an inequality constraint ($l \in \{n_{\text{eq}}+1,\dots,s\}$), then

$$\gamma_l^{(p)} = \max\{a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l, 0\} \tag{2.33}$$

- **Update main variables**

  Local controller i uses all $\gamma_l^{(p)}$ values that it has (by communications and those computed by itself) to compute an 'assumed neighborhood image' $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$. Note that $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$ has the same structure as the neighborhood image $\mathbf{x}^{(p-1)|\mathcal{N}^i}$. However, it is not the exact update of the neighborhood image. Indeed, $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$ is used only for constructing the new self-image by selecting the variables of subsystem i in $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$:

$$\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i} \tag{2.34}$$

  which contains $u_0^{i,(p)},\dots,u_{N-1}^{i,(p)},x_1^{i,(p)},\dots,x_N^{i,(p)}$.

- **Check the local termination criteria**

  For each local controller, there are local termination criteria. The local termination criteria also aim to keep a subsystem informed when other subsystems terminate. Hence when one set of local termination criteria is satisfied, the termination criteria for all subsystems are also satisfied. Each controller checks the local termination criteria using local communications only[8]. When all local controllers have converged, the algorithm stops and the local control actions are implemented.

In the following, we will describe the new method using the distributed algorithm.

Algorithm 2.4.4 ***Distributed algorithm for definite quadratic programs***

Initialize with $p=0$, $u_k^{i,(0)}=0, x_{k+1}^{i,(0)}=0, \forall i,k=0,\dots,N-1$ *(this means* $\mathbf{x}^{(0)|i}=0,\forall i$*, implying* $\mathbf{x}^{(0)}=0$*), and* $\gamma_l^{(0)}=0, l=1,\dots,s$

Next, for $p=1,2,\dots$*, the following steps are executed:*

1) ***Communications to get the updated main variables***

   Each controller i gets updated values of $\mathbf{x}^{(p-1)|j}$ *from its neighbors* $j\in\mathcal{N}^i$*, where only non-zero elements need to be transmitted*[9].

   Then controller i constructs the neighborhood image $\mathbf{x}^{(p-1)|\mathcal{N}^i}$ *using formula* (2.29).

2) ***Update intermediate variables*** γ_l ***in parallel***

   Each local controller i updates γ_l *for each* $l\in L_i$*, using* (2.32) *or* (2.33).

3) ***Communications to get the updated intermediate variables***

   Each local controller i gets $\gamma_l^{(p)}, l\in L_{\mathcal{N}^i}$ *that are updated by controllers in the neighborhood of i.*

[8] Checking the termination criteria in a distributed fashion requires a dedicated logic scheme, several schemes were described in [40, Chapter 8].

[9] Since $\mathbf{x}^{(p-1)|i}$ only has a few non-zero elements, which are $u_0^{i,(p-1)},\dots,u_{N-1}^{i,(p-1)},x_1^{i,(p-1)},\dots,x_N^{i,(p-1)}$, only these values need to be transmitted by controller i to reduce communications.

4) ***Update main variables in parallel***

Each local controller i computes an assumed neighborhood image *of* $\boldsymbol{x}$*:*

$$\boldsymbol{x}_{assumed}^{(p)|\mathcal{N}^i} = \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \tag{2.35}$$

Then controller i constructs the new self-image, *using* (2.34).

5) ***Check the local termination criteria in parallel***

Each local controller checks the local termination criteria. If local termination criteria are satisfied, the algorithm stops, otherwise go to step 1) to start a new iteration.

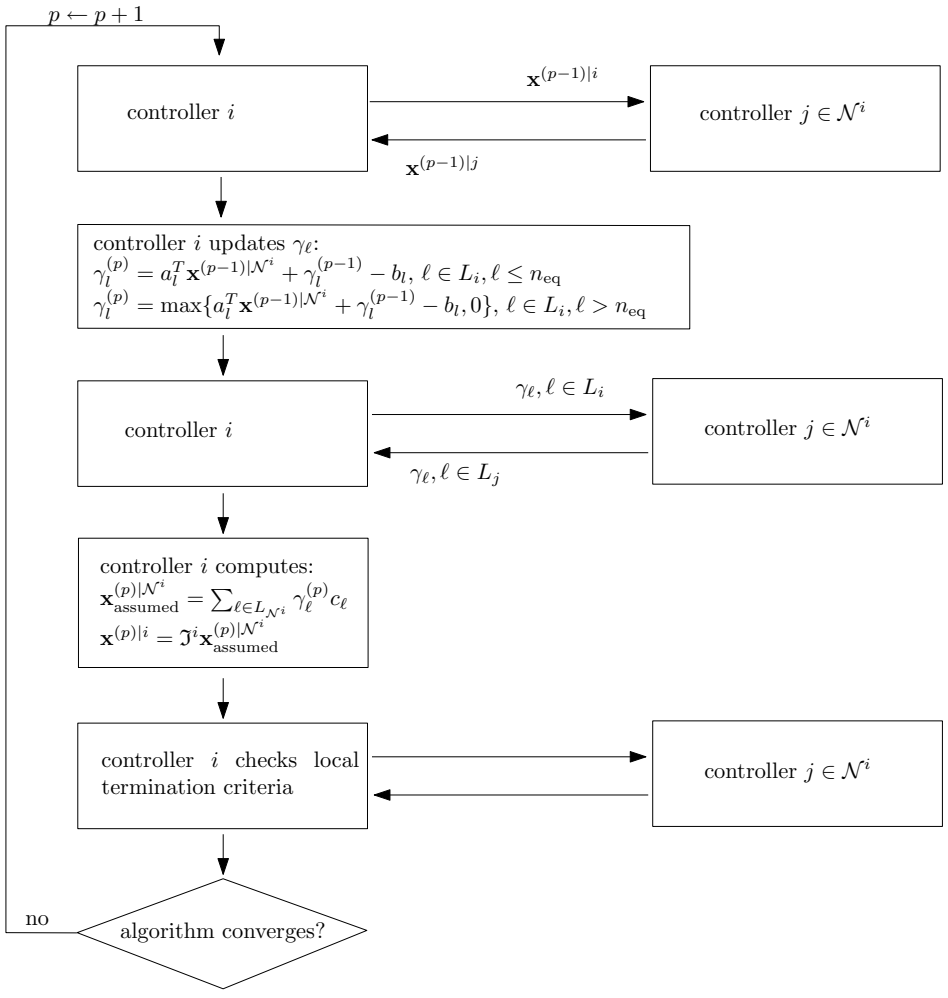


Figure 2.2: Computation and communication flow-chart of controller *i* in each iteration of Algorithm 2.4.4. Controller *i* only needs to communicate with its neighbor $j \in \mathcal{N}^i$.

In Algorithm 2.4.4, the activities of one local controller can be demonstrated by the diagram in Figure 2.2. The diagram clearly shows that in the distributed algorithm, each local controller *i* only

communicates with its neighbors $j \in \mathcal{N}^i$, enabling implementation of the method in a distributed setting. The properties of the distributed algorithm will be discussed in the following subsections.

Proof of equivalence to Han's algorithm using a global update scheme

In Algorithm 2.3.4, at step 2), the centralized variable $\mathbf{x}^{(p)}$ is updated via a global update scheme. In Algorithm 2.4.4, by the local update scheme we obtain $\mathbf{x}^{(p)|i}$ for $i = 1,\dots,M$. The equivalence of these two algorithms is stated in the following proposition:

Proposition 2.4.5 *Applying Algorithms 2.3.4 and 2.4.4 to the same problem* (2.8) *with the same parameter α, at any iteration p, the following statements hold:*

a) $\gamma_l^{(p)}$ *are the same in Algorithms 2.3.4 and 2.4.4, for all* $l \in \{1,\dots,s\}$.

b) $\boldsymbol{x}^{(p)} = \sum_{i=1}^{M} \boldsymbol{x}^{(p)|i}$, *in which* $\boldsymbol{x}^{(p)}$ *is calculated in Algorithm 2.3.4 while* $\boldsymbol{x}^{(p)|i}, i = 1,\dots,M$ *are calculated in Algorithm 2.4.4.*

Hence, Algorithm 2.3.4 and Algorithm 2.4.4 are equivalent.

Proof: The proposition will be proved by induction.

It is clear that properties a) and b) hold for $p = 0$.

Now consider iteration p, and assume that the properties a) and b) hold for all iterations before iteration p.

First, we prove property a). For any l and i such that $l \in L_i$, we have:

$$\begin{aligned} a_l^T \mathbf{x}^{(p-1)} &= a_l^T \sum_{j=1}^{M} \mathfrak{I}^j \mathbf{x}^{(p-1)|j} \\ &= a_l^T \left(\sum_{j \in \mathcal{N}^i} \mathfrak{I}^j \mathbf{x}^{(p-1)|j} + \sum_{j \notin \mathcal{N}^i} \mathfrak{I}^j \mathbf{x}^{(p-1)|j} \right) \end{aligned} \tag{2.36}$$

Due to the definition of *neighborhood*, a subsystem outside $\mathcal{N}^i$ does not have any coupled constraints with subsystem i. Therefore, $a_l^T \sum_{j \notin \mathcal{N}^i} \mathfrak{I}^j \mathbf{x}^{(p-1)|j} = 0$, which - in combination with (2.29) - leads to:

$$a_l^T \mathbf{x}^{(p-1)} = a_l^T \sum_{j \in \mathcal{N}^i} \mathfrak{I}^j \mathbf{x}^{(p-1)|j} = a_l^T \mathbf{x}^{(p-1)|\mathcal{N}^i} \tag{2.37}$$

Equation (2.37) then guarantees that $\gamma_l^{(p)}$ computed at step 1) of Algorithm 2.3.4 and at step 2) of Algorithm 2.4.4 are the same.

Now consider property b), where the main argument is the following: The same set of $\gamma_l^{(p)}$ and c_l are used for updating i's variables in $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$ (at step 4 of Algorithm 2.4.4) and in $\mathbf{x}^{(p)}$ (at step 2 of Algorithm 2.3.4). Thus each vector of local update $\mathbf{x}^{(p)|i}$, which contains values of i's variables selected from $\mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i}$, is a part of the centralized update $\mathbf{x}^{(p)}$.

More specifically, we can express the formula of $\mathbf{x}^{(p)|i}$ computed in Algorithm 2.4.4 as

$$\begin{aligned} \mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}_{\text{assumed}}^{(p)|\mathcal{N}^i} &= \mathfrak{I}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \\ \Rightarrow \sum_{i=1}^{M} \mathbf{x}^{(p)|i} &= \sum_{i=1}^{M} \mathfrak{I}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \end{aligned} \tag{2.38}$$

Note that in the following equations, $\mathbf{x}^{(p)}$ refers to the update of the decision variable computed by (2.25) in Algorithm 2.3.4, which we can express as

$$\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathfrak{I}^i \mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathfrak{I}^i \sum_{l=1}^{s} \gamma_l^{(p)} c_l \tag{2.39}$$

in which the first equality is due to the relation (2.27), the second equality is from (2.25).

Recall that c_l has the same structure as a_l, and if $l \notin L_{\mathcal{N}^i}$ then a_l and c_l do not have any non-zero values at the positions associated with variables of subsystem i. Therefore

$$\mathfrak{I}^i \sum_{l=1}^{s} \gamma_l^{(p)} c_l = \mathfrak{I}^i \left(\sum_{l \notin L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l + \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \right) = \mathfrak{I}^i \sum_{l \in L_{\mathcal{N}^i}} \gamma_l^{(p)} c_l \tag{2.40}$$

This equality shows that (2.39) and (2.38) are equivalent, thus proving the equality in property b): $\mathbf{x}^{(p)} = \sum_{i=1}^{M} \mathbf{x}^{(p)|i}$. □

The equivalence of Algorithms 2.3.4 and 2.4.4 implies that problem (2.8) can be solved using Algorithm 2.4.4. This allows us to implement a DMPC scheme using Algorithm 2.4.4 that does not need global communications.

Properties of the distributed MPC controller

Convergence, feasibility, and stability properties of the DMPC scheme using Algorithm 2.4.4 are established by the following propositions:

Proposition 2.4.6 *Assume that* $(\mathscr{P})$ *has a feasible solution. Then Algorithm 2.4.4 asymptotically converges to the centralized solution of* $(\mathscr{P})$ *at each sampling step.*

Proof: In [2] it is shown that Han's method is guaranteed to converge to the centralized solution of the convex quadratic program (2.8) under the conditions that $q(\mathbf{x})$ is uniformly convex and differentiable on $\mathbb{R}^{n_\mathbf{x}}$ and (2.8) has a feasible solution. Due to the positive definiteness of Q_i and R_i, and the assumption that $(\mathscr{P})$ has a feasible solution, such conditions hold for the quadratic problem (2.8). Moreover, Algorithm 2.4.4 is equivalent to Han's method for the problem (2.8). Hence, the distributed scheme in Algorithm 2.4.4 converges to the centralized solution of (2.8), which is the same as $(\mathscr{P})$. □

Proposition 2.4.7 *Assume that at every sampling step, Algorithm 2.4.4 asymptotically converges. Then the DMPC scheme is recursively feasible and stable.*

Proof: By letting Algorithm 2.4.4 converge at every sampling step, the centralized solution of $(\mathscr{P})$ is obtained. Recursive feasibility and stability is guaranteed as a consequence of centralized MPC with a terminal point constraint, as shown in [16] and [38]. □

It is also worth to address the conservativeness of the MPC formulation using the terminal point constraint $x_N = 0$, which would reduce the domain of attraction of MPC. However, this issue is not related to Han's method. In fact, the distributed Han's method is able to handle optimization problems of other MPC formulations, given that the cost function has a sparse coupling structure. Note that finding other MPC formulations with a sparse coupling structure is not straightforward, we will discuss this problem in Section 2.6.

2.4.2 Distributed version of Han's method with scaled step size

A disadvantage of Han's method (and its distributed version) is the slow convergence rate, due to the fact that it is essentially a projection method to solve the dual problem of (2.8). Moreover, Han's (distributed) method uses zeros as the initial guess, which prevents warm starting of the algorithm by choosing an initial guess that is close to the optimizer. Therefore, we need to modify the method to achieve a better convergence rate.

In this section, we present two modifications of the distributed version of Han's method:

- Scaling of the step sizes related to dual variables by using heterogeneous α_l for the update of each l-th dual variable instead of the same α for all dual variables.
- Use of nonzero initial guesses, which allows taking the current MPC solution as the start for the next sample step.

Note that the modified distributed algorithm is then not equivalent to the centralized algorithm anymore. There is no convergence proof for the modified distributed algorithm yet; this will be discussed in Section 2.6.

In order to implement the above modifications, the improved distributed version of Han's method is initialized similarly to the distributed algorithm in Section 2.4.1, except for the following procedures:

1. **Pre-computed invariant parameters**

   Each subsystem i computes and stores the following parameters throughout the control scheme:

   - For each $l \in L_i$: $\alpha_l = \left(k_\alpha\right)_l \alpha_0$, where k_α is the scaling vector. α_l acts as local step size regarding the l-th dual variable, and therefore k_α should be chosen such that the convergence rates of all s dual variables are improved. The method to choose k_α will be discussed in Remark 2.4.9.
   - For each $l \in L_i$: $\bar{c_l} = \frac{-1}{a_l^T a_l} H^{-1} a_l$. We can see that $\bar{c_l}$ can be computed locally by a local controller with a priori knowledge of the parameter a_l and the weighting blocks on the diagonal of H that correspond to the non-zero elements of a_l.

2. **MPC step**

   At the beginning of the MPC step, the current states of all subsystems are measured. The sequences of predicted states and inputs generated in the previous MPC step are shifted forward one step, then we add zero states and zero inputs to the end of the shifted sequences. The new sequences are then used as the initial guess for solving the optimization problem in the current MPC step[10]. The initial guess for each subsystem can be defined locally. For subsystem i, denote the initial guess as $\mathbf{x}^{(0)|i}$. At the first MPC step, we have $\mathbf{x}^{(0)|i} = 0, \forall i$.

   The current state is plugged into the MPC problem, then we get an optimization problem of the form (2.8). This problem will be solved by the following modified distributed algorithm of Han's method.

[10] The idea of using previously predicted states and inputs for initialization is a popular technique in MPC [17]. Especially with Han's method, whose convergence rate is slow, an initial guess that is close to the optimal solution will be very helpful to reduce the number of iterations.

Algorithm 2.4.8 ***Improved distributed algorithm for the MPC optimization problem***

Initialize with $p = 0$. Each subsystem i uses the initial guess as $\boldsymbol{x}^{(0)|i}$.
Next, for $p = 1, 2, \ldots$, the following steps are executed:

1) *See step 1 of Algorithm 2.4.4.*

2) *See step 2 of Algorithm 2.4.4, except that for $p = 1$, each subsystem i computes the initial intermediate variables by*[11]*:*

$$\gamma_l^{(1)} = a_l^T \left(\boldsymbol{x}^{(0)|\mathcal{N}^i} - \frac{\alpha_l}{s} H \boldsymbol{x}^{(0)|\mathcal{N}^i} \right) - b_l, \quad l \in L_i, l \leq n_{\text{eq}} \tag{2.41}$$

$$\gamma_l^{(1)} = \max \left\{ a_l^T \left(\boldsymbol{x}^{(0)|\mathcal{N}^i} - \frac{\alpha_l}{s} H \boldsymbol{x}^{(0)|\mathcal{N}^i} \right) - b_l, 0 \right\}, \quad l \in L_i, l > n_{\text{eq}} \tag{2.42}$$

3) *See step 3 of Algorithm 2.4.4.*

4) *See step 4 of Algorithm 2.4.4 but with a different formula to update the assumed neighborhood image for each i:*

$$\boldsymbol{x}_{assumed}^{(p)|\mathcal{N}^i} = \sum_{l \in L_{\mathcal{N}^i}} \frac{1}{\alpha_l} \gamma_l^{(p)} \bar{c}_l \tag{2.43}$$

5) *See step 5 of Algorithm 2.4.4.*

When the iterative procedure finishes, each subsystem applies the first input $u_0^{i,(p)}$, then waits for the next state measurement to start a new MPC step.

Remark 2.4.9 *The main improvement of Algorithm 2.4.8 over Algorithm 2.4.4 is the improved convergence speed, which heavily depends on a good choice of the scaling vector k_α. We have observed that the convergence rate of some dual variables under the responsibility of a subsystem i will affect the convergence rate of dual variables under the responsibility of its neighbors in $\mathcal{N}^i$. Therefore the choice of scaling vector should focus on improving the convergence rate of dual variables that appear to converge more slowly. In our case, we rely on the Hessian to find the scaling vector. Specifically, for each subsystem i, let $\bar{h}_i$ denote the average weight of its variables (i.e. average of entries related to i's states and inputs in the diagonal of the Hessian). We then choose the scaling factor $\left(k_\alpha\right)_l = 1/\bar{h}_i$, for all $l \in L_i$. We also multiply the scaling vector k_α with a factor $\theta \in (0,1)$ for enlarging the step sizes of all dual variables. In the first MPC step, we start tuning with $\theta \simeq 1$ and gradually reduce θ until it causes the algorithm to diverge, then we stop and choose the smallest θ such that the algorithm still converges.*

The choice of the scaling vector depends on the structure of the centralized optimization problem, thus we only need to choose it once in the first MPC step. Then for the next MPC steps, we can re-use the same scaling vector.

The efficiency of Algorithm 2.4.8 will be demonstrated in the example of irrigation canal control, which is presented in the next section.

[11] The intermediate variables are constructed following the formulas (2.20)–(2.21) with replacing the common α by α_l for each $l \in \{1, \ldots, s\}$, where we implicitly use $\mathbf{y}_l^{(0)} = \frac{1}{s}\mathbf{y}^{(0)}, \forall l \in \{1, \ldots, s\}$ and $\mathbf{y}^{(0)} = H\mathbf{x}^{(0)}$. Also note that since a_l only involves neighboring subsystems and H is block-diagonal, the computation $a_l^T \left(\mathbf{x}^{(0)} - \frac{\alpha_l}{s} H \mathbf{x}^{(0)} \right)$ only uses values from neighboring subsystems, similarly to the argument for (2.37).

2.5 Application of Han's method for distributed MPC in canal systems

2.5.1 The example canal system

The novel DMPC approach is applicable to a wide range of large-scale systems which could be modeled in the LTI form as described in Section 2.2. In this section, we demonstrate its application in an example control problem, where the objective is to regulate the water flows in a system of irrigation canals. Irrigation canals are large-scale systems, consisting of many interacting components, and spanning vast geographical areas. For the most efficient and safe operation of these canals, maintaining the levels of the water flows close to pre-specified reference values is crucial, both under normal operating conditions as well as in extreme situations. Manipulation of the water flows in irrigation canals is typically done using devices such as pumps and gates.

The example irrigation canal to be considered is a 4-reach canal system as illustrated in Figure 2.3. In this system, water flows from an upstream reservoir through the reaches, under the control of 4 gates and a pump at the end of the canal system that discharges water.

The control design is based on the master-slave control paradigm, in which the master controllers compute the flows through the gates, while each slave controller uses the local control actuators to guarantee the flow set by the master controller [41]. We will use the new DMPC method to design the master controllers.

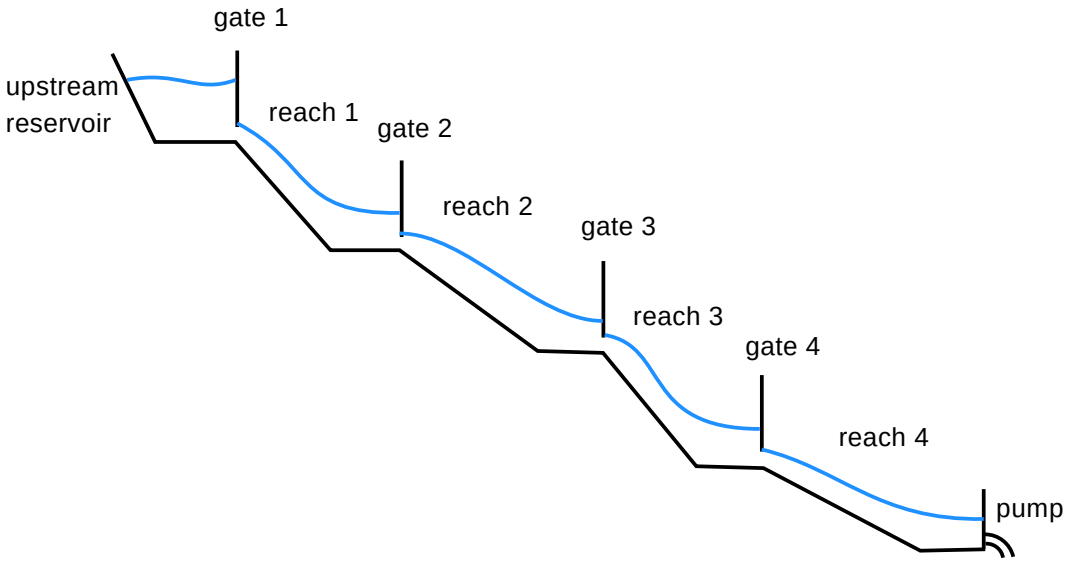


Figure 2.3: The example canal system

2.5.2 Modeling the canal

The canal system is divided into 4 subsystems, each of which corresponds to a reach and also includes the local controller at the upstream gate of the reach. The 4^{th} subsystem has one more controller, corresponding to the pump at its downstream end.

We use a simplified model for each subsystem as illustrated in Figure 2.4, and then obtain the overall model by connecting the subsystem models. A subsystem is approximately modeled by a reservoir with upstream in-flow and downstream out-flow.

The discrete-time model of reach i is represented by:

$$h^i_{k+1} - h^i_k = \frac{T_s}{A^i_s}\left[\left(Q^i_{\text{in}}\right)_k - \left(Q^i_{\text{out}}\right)_k\right] \tag{2.44}$$

where superscript i represents the subsystem index, subscript k is for the time index, T_s is the sampling time, h is the downstream water level of the reach, A_s is the water surface (i.e. the volume of reservoir $= h \cdot A_s$), Q_{in} and Q_{out} are the in-flow and the out-flow of the canal which are measured at the upstream and downstream ends, respectively. Denote the flow passing i^{th} gate by q^i, and the flow passing the pump by p^4. Due to the mass conservation law, we have $Q^i_{\text{out}} = Q^{i+1}_{\text{in}} = q^{i+1}$, for $i = 1,2,3$, and $Q^4_{\text{out}} = p^4$.

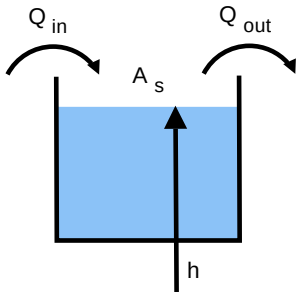


Figure 2.4: Model of a reach

In order to derive local dynamics, we choose input and state vectors of subsystem i as

$$x^i_k = h^i_k$$
$$u^i_k = \begin{cases} q^i_k & , \quad i = 1,2,3 \\ \begin{bmatrix} q^i_k \\ p^i_k \end{bmatrix} & , \quad i = 4 \end{cases}$$

The dynamics of each subsystem can be represented by a discrete-time, linear time-invariant model of the form (2.1) with the state-space matrices:

$$A^{ii} = \ 1 \ , \quad i = 1,\ldots,4; \quad A^{ij} = \ 0 \ , \quad i \neq j$$
$$B^{ii} = \ T_s/A^i_s \ , \quad i = 1,2,3; \quad B^{44} = \begin{bmatrix} T_s/A^4_s & -T_s/A^4_s \end{bmatrix}$$
$$B^{i(i+1)} = \ -T_s/A^i_s \ , \quad i = 1,2; \quad B^{34} = \begin{bmatrix} -T_s/A^4_s & 0 \end{bmatrix}$$
$$B^{ij} = 0\,, \quad i = 1,2,3, \quad j \notin \{i, i+1\}.$$

2.5.3 Simulation results

DMPC methods are applied to the regulation problem of the simulated canal system described in previous subsections using sampling time $T_s = 240s$, with a perturbed initial state. We use the distributed Han's method with and without the modifications described in Section 2.4, and compare the results. Figure 2.5 shows the convergence of the distributed solutions to the centralized solution for the problem. Starting from the same initial guess in the first MPC step, i.e. all variables are initialized with zeros, the distributed algorithm with modifications achieves a better convergence rate, allowing the distributed optimization to converge within an acceptable number of iterations. Similar results were also achieved for the next MPC steps, when we simulate the closed-loop MPC and let the distributed solutions converge to the centralized solution at every step, with maximally 100 iterations per step.

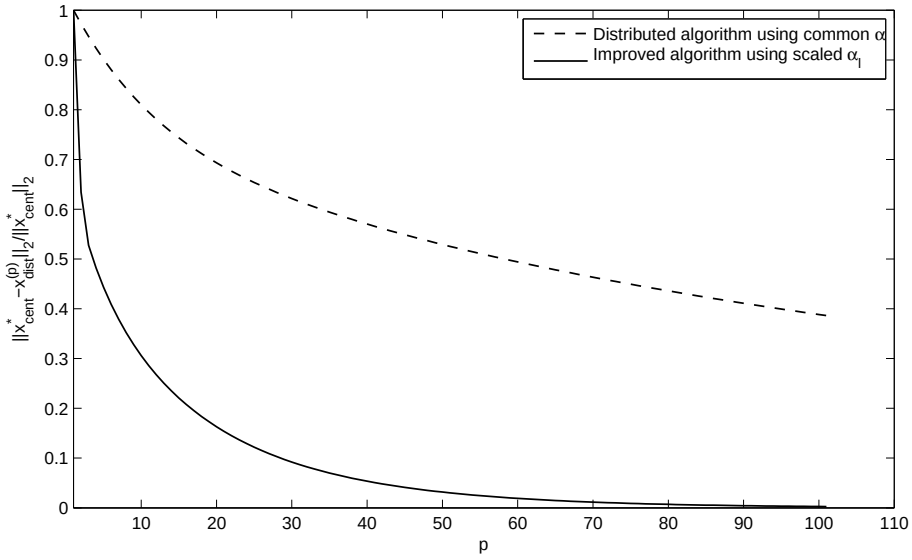


Figure 2.5: Comparison of convergence rates of two distributed versions of Han's method for the first sampling time step (k=1)

2.6 Discussion and outlook on future research

Two distributed versions of Han's method have been described in Section 2.4, followed by a short demonstration of their usage in Section 2.5. Although these algorithms help to implement Han's method in a distributed setting for MPC, there are still some theoretical issues that need to be addressed.

Firstly, the proposed distributed algorithms deal with quadratic programs only. Although many MPC problems for linear time-invariant systems are formulated as quadratic programs, there are other variants that use different objective functions, and nonlinear MPC would also yield more complicated optimization problems than quadratic programs. With such problems, we might not be able to implement Han's parallel method in a distributed fashion. This issue motivates the research on other decomposition methods that can handle more general problems, e.g. convex problems with linear or decoupled nonlinear constraints.

As noted in Section 2.4.1, the MPC formulation in this paper employs the terminal constraint $x_N = 0$, which is a conservative approach. In case we want to use less conservative MPC, e.g. MPC with a terminal constraint set and a terminal controller, we need to find a separable terminal penalty function and local terminal constraint sets. However, to the authors' best knowledge, there is still no distributed scheme available to construct local terminal constraint sets and local terminal controllers (and also the terminal penalty matrix that is solution of the Riccati equation), other than assuming them to be completely decoupled. Therefore, although distributed Han's method can also be applied to any uniformly convex QP with sparse coupling structure, it requires further research on MPC formulations that have such optimization problems.

In general, Han's method has a slow convergence rate due to its iterative projection nature, which is inherited by Algorithm 2.4.4. Since the feasibility and stability properties are derived upon convergence of the algorithm within each sampling step, we need to speed up the convergence of this method.

The distributed version of Han's method with scaling can improve the convergence rate significantly, as illustrated in Section 2.5. However, its proof of convergence is still lacking. We observe that in setups that are more complicated, the proposed method to choose the scaling vector does not always work well (sometimes after several sample steps, the algorithm does not converge anymore). Due to the requirement not to have global communications, it is difficult to adjust the scaling vector during the iteration to reach convergence. Therefore speeding up Han's method while providing a proof for convergence is still an open issue, and we may use a coordinator at a higher level of hierarchy that has global communication capabilities to tackle this issue.

Another issue is due to the formulation of the optimization problem for MPC, where we keep both inputs and states as variables of the centralized optimization problem and do not eliminate the states using the dynamic model equations. This formulation is advantageous in distributed MPC because the Hessian will then keep a block diagonal structure, and the neighborhood of each subsystem will only contains its direct neighbors (the neighborhood would be greatly extended if we eliminate the states in the optimization problem). However, using states as variables requires considering the dynamical equations as equality constraints of the optimization problem, and the existence of equality constraints typically requires an exact solution in order to guarantee feasibility. Since Han's method converges asymptotically, we may not be able to get the exact optimal multipliers in real-time, and then the corresponding primal iterates would not be guaranteed to be feasible. In general, most dual decomposition methods do not provide primal feasible solutions before reaching the dual optimal solutions, so this feasibility issue also applies to other dual decomposition methods.

In future research, we will also study dual decomposition methods that can provide primal feasible solutions in a finite number of iterations. In order to tackle the convex problem, we intend to make use of the subgradient schemes proposed in [42] and [43], which extend the traditional primal recovery schemes for linear programs [44, 45]. With this approach, the standard proof for MPC stability, which is based on optimality, will not be obtained. Therefore, we need to prove stability of suboptimal MPC, which can be based on the theorems proposed in [46], i.e. showing the reduction of the cost function (acting as a Lyapunov function) associated with the feasible solution. We intend to use the bounds of suboptimality of the subgradient iterations to show the decreasing property of the cost function. Finding such bounds that are suitable for proving suboptimal MPC stability is still an open question.

2.7 Conclusions

A decomposition approach based on Fenchel's duality and Han's parallel method has been developed in this paper, resulting in two distributed algorithms that are applicable to DMPC. The first distributed algorithm generates updates that are equivalent with those computed globally by Han's method for definite quadratic problems, and therefore it has the same convergence property as Han's method. Moreover, feasibility and stability of DMPC are achieved upon convergence of the iterations. The second distributed algorithm aims to improve the convergence speed by using scaled step sizes and nonzero initial guess. The new methods have been applied to an example irrigation canal network, demonstrating their applicability for water network and other large-scale networked systems. We have also summarized open issues of using Han's method and other dual decomposition methods for MPC, including the topics of distributed formulation, convergence rate, primal feasibility, and stability of MPC. These issues were used to recommend future research directions.

Chapter 3

Distributed model predictive control driven by simultaneous derivation of prices and resources

This chapter is part of the research accepted for publication at the 2011 IFAC World Congress, Milan, Italy, with particular citation: Holger Scheu and Wolfgang Marquardt, Distributed Model-Predictive Control Driven by Simultaneous Derivation of Prices and Resources, Proceedings of the 2011 IFAC World Congress, Milan, Italy.

3.1 Introduction

Chemical processes, among other processes, are usually described by dynamic multi-input multi-output (MIMO) models. Industrial plants are typically operated by decentralized control technology, e.g. single-input single-output (SISO) PID loops, enhanced by supervisory optimal controllers. Optimal control methods such as linear or nonlinear model predictive control (MPC) have become important technologies, as they ensure an optimal operation of the plant while maintaining operational constraints and while, in contrast to the SISO PID controllers, considering the MIMO behavior of the system. The industrial state of the art on model predictive control technology is outlined in the paper of [47].

Recently, distributed control methods have become an important area in control research. These methods are expected to provide better computational performance [48] compared to decentralized controllers and to remove possible communication bottlenecks. Furthermore, compared to a centralized solution, reliability and maintainability could be increased [49]. Finally, completely new applications are addressed requiring decentralized control [50, 51, 52]. A process plant usually consists of different subsystems, namely the process units, which exchange material, energy or signals as illustrated in Figure 3.1. On a coarser scale, there are multiple plants on a production site or in a supply chain, which again interact also by the exchange of material, energy and signals. Hence, process plants and production sites are represented by large-scale dynamical models composed by a set of coupled smaller models. Similar topologies can be found e.g. in water or power systems.

These systems are either controlled by a single centralized controller or independently by decentralized controllers. The decentralized controllers typically neglect the interactions between the subsystems. However, nominal stability, feasibility, optimality, reliability and maintainability are desired properties for the implemented control system. Distributed model predictive control (DMPC)

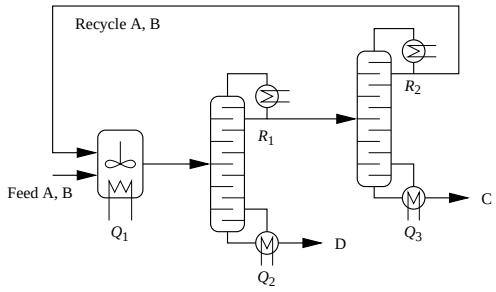


Figure 3.1: Example for a distributed process with reaction and separation units.

methods are expected to contribute towards this aim as they combine the optimality properties of centralized predictive controllers and the modularity and flexibility of decentralized controllers. In addition they are expected to cope with large-scale systems.

Already in the early 1970s, the basis of distributed control, and in particular, distributed optimal control, has been established [53]. Two main decomposition methods, which are the basis for DMPC, have been proposed, namely primal decomposition [54, 55] and dual decomposition [56]. Various early survey papers exist on distributed control, i.e. [57], [58], and [12]. The present status of research in the field of distributed optimal control is summarized by [20]. Different control architectures have been reviewed by [21] recently.

Focus in recent research of DMPC is on linear systems. [59] consider SISO linear time-invariant systems with output coupling and apply dual decomposition. The dual problem is solved using a subgradient optimization algorithm in this work. [60] propose a proximal center-based dual decomposition method. The aim of the method is to smooth the Lagrangian function resulting in a smooth convex objective function, while the primal problems involve strongly convex objective functions. A DMPC method based on primal decomposition, called feasible cooperation-based control, is proposed by [61]. The full objective function is included in all controllers to ensure coordination and a feasible solution is generated in each iteration of the method. [62] consider linear step-response models and formulate a DMPC formulation based on resource allocation, i.e. a primal decomposition approach, which is successfully applied to a fluid catalytic cracking process. A strength of the method is its fast convergence.

Results related to nonlinear DMPC are limited. [63] propose a cooperative distributed model predictive control method for nonlinear systems. Communication and cooperation is restricted to neighboring subsystems. Lyapunov-based DMPC is proposed by [64]. In order to ensure stability the optimal control problem embedded into the DMPC includes an additional dynamic equation ensuring a decrease of a Lyapunov function. [65] proposes a distributed MPC method and applies the method to a system of coupled oscillators. [66] propose a stabilizing non-cooperative decentralized MPC scheme for nonlinear discrete-time systems.

In the following, we will focus on a new sensitivity-driven distributed model predictive control (S-DMPC) for discrete-time systems, which is based on a novel distributed dynamic optimization method [4]. In order to achieve optimality for the overall process, the subsystems' objective functions are modified using information of the whole process. The modification of the objective functions incorporates a linearization of neighboring objective and constraint functions in the adapted subproblems. Hence, the coordination in the distributed method is based on first order sensitivities. In this paper we focus on linear discrete-time systems, while [5] considered continuous-time systems. A convergence analysis is provided for discrete-time systems. The method is applied to an unstable dis-

tributed system to illustrate its capabilities. In the case study, fast convergence can be observed, while computing time is reduced compared to a centralized controller.

The remainder of this paper is organized as follows: In Section 3.2, we state the discrete-time optimal control problem to be solved. Section 3.3 presents the distributed model predictive control method and a convergence analysis of the method. In Section 3.4, the S-DMPC method is applied to a simulated process. Finally, we conclude the paper with a summary and an outlook in Section 3.5.

3.2 Problem formulation

We consider the distributed discrete-time linear time-invariant system

$$x(k+1) = A\,x(k) + B\,u(k), \quad x(0) = x_0. \tag{3.1a}$$

N denotes the number of subsystems, $A = [A_{ij}]_{i,j=1,\dots,N}$, the system matrix, and $B = [B_{ij}]_{i,j=1,\dots,N}$, the input matrix. $x(k) \in \mathbb{R}^n$ is the state vector with $x = (x_1,\dots,x_N)$[1] and $x_i(k) \in \mathbb{R}^{n_i}$, $u(k) \in \mathbb{R}^m$ is the input vector with $u = (u_1,\dots,u_N)$ and $u_i(k) \in \mathbb{R}^{m_i}$. $x_0 = (x_{10},\dots,x_{N0})$ refers to the initial condition, while k refers to the time index with $t = t_0 + k\,\Delta t$ and sampling time Δt. The linear quadratic control problem reads as

$$\min_{x,u} \frac{1}{2} \sum_{k=k'}^{k'+K-1} \left(\|x(k)\|_Q^2 + \|u(k)\|_R^2 \right) + \|x(h)\|_P^2, \tag{3.2a}$$

$$\text{s.t. } x(k+1) = Ax(k) + Bu(k), \quad k = k',\dots,k'+K-1, \tag{3.2b}$$

$$x(k') = x_{k'}, \tag{3.2c}$$

with $Q = [Q_{ij}]_{i,j=1,\dots,N}$, $R = [R_{ij}]_{i,j=1,\dots,N}$, $P = [P_{ij}]_{i,j=1,\dots,N}$, on a moving horizon with $k' = 0,1,2,\dots$ without consideration of measurement noise. k' denotes the initial time sample of the moving horizon problem formulation. The submatrices $Q_{ij} \in \mathbb{R}^{n_i \times n_j}$, $R_{ij} \in \mathbb{R}^{m_i \times m_j}$, and $P_{ij} \in \mathbb{R}^{n_i \times n_j}$ are positive definite weighting matrices, $K \cdot \Delta t$ is the prediction and control horizon with K time samples. While inequality constraints for input variables u and state variables x are not considered in this work, the theory presented can be extended as described in [5].

The optimal control problem for the individual subsystems i is coupled through the dynamics as well as the objective function. For distributed control, the optimal control problem (3.2) is decomposed into N subproblems such that it can be solved by the corresponding N controllers of the distributed MPC. Thus, the optimal control problem (3.2) is first reformulated as

$$\min_{\boldsymbol{x},\boldsymbol{u}} \sum_{i=1}^{N} \Phi_i \tag{3.3a}$$

$$\text{s.t. } \Phi_i = \sum_{j=1}^{N} \frac{1}{2} \left(\boldsymbol{x}_i^T \, \mathscr{Q}_{ij} \, \boldsymbol{x}_j + \boldsymbol{u}_i^T \, \mathscr{R}_{ij} \, \boldsymbol{u}_j \right), \tag{3.3b}$$

$$0 = c_i \tag{3.3c}$$

$$c_i = -I\boldsymbol{x}_i + \sum_{j=1}^{N} \left(\boldsymbol{A}_{ij}\boldsymbol{x}_j + \boldsymbol{B}_{ij}\boldsymbol{u}_j + X_{ijk'} \right), \tag{3.3d}$$

$$\forall i \in \{1,\dots,N\},\ k' = 0,1,2,\dots\ ,$$

[1] The brackets $(v_1 \dots v_N)$ indicate $[v_1^T \dots v_N^T]^T$.

with

$$
\begin{aligned}
\boldsymbol{x}_i &= (x_i(k'+1),\ldots,x_i(k'+K)),\\
\boldsymbol{u}_i &= (u_i(k'),\ldots,u_i(k'+K-1)),\\
\mathscr{Q}_{ij} &= \mathrm{diag}(Q_{ij},\ldots,Q_{ij},P_{ij}),\\
\mathscr{R}_{ij} &= \mathrm{diag}(R_{ij},\ldots,R_{ij}),\\
X_{ijk'} &= (A_{ij}x_j(k'),0,\ldots,0),\\
\boldsymbol{A}_{ij} &= \begin{bmatrix} 0 & & & \\ A_{ij} & \ddots & & \\ & \ddots & \ddots & \\ & & A_{ij} & 0 \end{bmatrix}, \boldsymbol{B}_{ij} = \begin{bmatrix} B_{ij} & & & \\ & \ddots & & \\ & & \ddots & \\ & & & B_{ij} \end{bmatrix}.
\end{aligned}
$$

The objective function is separated such that the separable parts are assigned to the unique corresponding subsystems. The coupled parts are equally assigned to both of the corresponding subsystems, here. The variables in the constraint functions c_i are separated with respect to their indices, i.e.

$$
\begin{aligned}
c_i = \left[\boldsymbol{A}_{ii} - I, \quad \boldsymbol{B}_{ii}\right](\boldsymbol{x}_i,\boldsymbol{u}_i) + \sum_{j=1}^{N} X_{ijk'}\\
+ \sum_{\substack{j=1\\ j\neq i}}^{N} \left[\boldsymbol{A}_{ij}, \quad \boldsymbol{B}_{ij}\right](\boldsymbol{x}_j,\boldsymbol{u}_j).
\end{aligned} \tag{3.4}
$$

We finally introduce $\boldsymbol{z}_i = (\boldsymbol{x}_i,\boldsymbol{u}_i)$, $\boldsymbol{z} = (\boldsymbol{z}_1,\ldots,\boldsymbol{z}_N)$, $\mathscr{T}_{ij} = \begin{bmatrix} \mathscr{Q}_{ij} & 0 \\ 0 & \mathscr{R}_{ij} \end{bmatrix}$, $\mathscr{A}_{ij} = \left[\boldsymbol{A}_{ij}, \quad \boldsymbol{B}_{ij}\right]$, and $\mathscr{I}_i = \left[I_i, 0\right]$ to obtain the quadratic program (QP)

$$
\begin{aligned}
&\min_{\boldsymbol{z}} \sum_{i=1}^{N} \Phi_i && \text{(3.5a)}\\
&\text{s.t. } \Phi_i = \frac{1}{2}\sum_{j=1}^{N} \boldsymbol{z}_i^T \ \mathscr{T}_{ij}\, \boldsymbol{z}_j, && \text{(3.5b)}\\
&\qquad 0 = c_i && \text{(3.5c)}\\
&\qquad c_i = (\mathscr{A}_{ii} - \mathscr{I}_i)\boldsymbol{z}_i + \sum_{j=1}^{N} X_{ijk'} + \sum_{\substack{j=1\\ j\neq i}}^{N} \mathscr{A}_{ij}\boldsymbol{z}_j, && \text{(3.5d)}\\
&\qquad \forall i \in \{1,\ldots,N\}.
\end{aligned}
$$

The QP derived is coupled by the objective as well as the constraint functions.

3.3 Distributed model predictive control

We suggest to coordinate the distributed optimal control embedded in the QP (3.5) using the adapted objective functions [5]

$$\Phi_i^* = \Phi_i + \left[\sum_{\substack{j=1 \\ j\neq i}}^{N} \left. \frac{\partial \Phi_j}{\partial \boldsymbol{z}_i} \right|_{\boldsymbol{z}^{[l]}} - \left. \frac{\partial c_j}{\partial \boldsymbol{z}_i} \right|_{\boldsymbol{z}^{[l]}} \lambda_j^{[l]} \right] (\boldsymbol{z}_i^{[l+1]} - \boldsymbol{z}_i^{[l]}), \tag{3.6}$$

which includes the local objective function Φ as well as linear information of the full optimal control problem provided by first order sensitivities. Thus, the method is referred to as Sensitivity-Driven Distributed Model Predictive Control (S-DMPC) for discrete-time. There, the upper index $^{[l]}$ refers to the l-th iteration of the corresponding variable.

The QP (3.5) is replaced by a set of QP with minimizer

$$\boldsymbol{z}_j^{[l+1]} = \arg\min_{\boldsymbol{z}_i} \ \Phi_i^* \tag{3.7a}$$

$$\begin{aligned} \text{s.t. } \Phi_i^* = {} & \frac{1}{2} \boldsymbol{z}_i^T \ \mathscr{T}_{ii} \ \boldsymbol{z}_i + \frac{1}{2} \sum_{\substack{j=1 \\ j\neq i}}^{N} \boldsymbol{z}_i^T \ \mathscr{T}_{ij} \ \boldsymbol{z}_j^{[l]} \\ & + \left[\sum_{\substack{j=1 \\ j\neq i}}^{N} \left. \frac{\partial \Phi_j}{\partial \boldsymbol{z}_i} \right|_{\boldsymbol{z}^{[l]}} - \left. \frac{\partial c_j}{\partial \boldsymbol{z}_i} \right|_{\boldsymbol{z}^{[l]}} \lambda_j^{[l]} \right] (\boldsymbol{z}_i - \boldsymbol{z}_i^{[l]}), \end{aligned} \tag{3.7b}$$

$$0 = (\mathscr{A}_{ii} - \mathscr{I}_i)\boldsymbol{z}_i + \sum_{j=1}^{N} X_{ijk'} + \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathscr{A}_{ij} \boldsymbol{z}_j^{[l]}, \tag{3.7c}$$

$$\forall i \in \{1, \ldots, N\}.$$

QP (3.7) has the same minimizer as QP (3.5) [5]. The proof is done by comparing the necessary conditions of optimality (NCO), the so-called Karush-Kuhn-Tucker conditions. In this distributed QP formulation prices (given by the Lagrange multipliers λ) and resources (given by primal variables $\boldsymbol{z}$) are derived simultaneously, while for the most common approach in distributed MPC, namely optimization based on dual decomposition [56], the prices and resources are sequentially calculated on different hierarchical layers. Furthermore, the S-DMPC presented is able to cope with non separable cost functions Φ_i due to the inclusion of the corresponding sensitivities in each of the subsystems cost functions.

Convergence of the iterative method is analyzed in the following. For this purpose, the NCO are stated for the decomposed QP (3.7). First, the Lagrangian function

$$\mathscr{L}_i = \Phi_i^*(\boldsymbol{z}) - \lambda_i c_i(\boldsymbol{z}) \tag{3.8}$$

is stated for each subsystem i. The NCO are

$$\begin{aligned} \frac{\partial \mathscr{L}_i}{\partial \boldsymbol{z}_i} = 0 = {} & \mathscr{T}_{ii} \boldsymbol{z}_i^{[l+1]} + \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathscr{T}_{ij} \boldsymbol{z}_j^{[l]} \\ & - \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathscr{A}_{ji}^T \lambda_j^{[l]} - (\mathscr{A}_{ii} - \mathscr{I}_i)^T \lambda_i^{[l+1]} \end{aligned} \tag{3.9}$$

and

$$\begin{aligned}\frac{\partial \mathscr{L}_i}{\partial \lambda_i} &= 0 = c_i(\boldsymbol{z}) \\ &= (\mathscr{A}_{ii} - \mathscr{I}_i)\boldsymbol{z}_i^{[l+1]} + \sum_{j=1}^{N} X_{ijk'} + \sum_{\substack{j=1 \\ j\neq i}}^{N} \mathscr{A}_{ji}\boldsymbol{z}_j^{[l]}. \end{aligned} \tag{3.10}$$

We define a mapping ζ_i with $(\boldsymbol{z}_i, \lambda_i) = \zeta_i(\boldsymbol{z}, \lambda)$, where $\lambda = (\lambda_1, \ldots, \lambda_N)$ is the aggregated vector of Lagrange multipliers. Aggregating equations (3.9) and (3.10) for all i results in the mapping $(\boldsymbol{z}, \lambda) = \zeta(\boldsymbol{z}, \lambda)$ given by

$$\begin{bmatrix} \boldsymbol{z}^{[l+1]} \\ \lambda^{[l+1]} \end{bmatrix} = \underbrace{\Xi_{\mathrm{d}}^{-1}\Xi_1 \begin{bmatrix} \boldsymbol{z}^{[l]} \\ \lambda^{[l]} \end{bmatrix} + \Xi_{\mathrm{d}}^{-1}\Xi_0}_{=\zeta(\boldsymbol{z}^{[l]}, \lambda^{[l]})} \tag{3.11a}$$

with

$$\Xi_{\mathrm{d}} = \begin{bmatrix} \mathscr{T}_{\mathrm{d}} & [\mathscr{I} - \mathscr{A}_{\mathrm{d}}]^T \\ [\mathscr{I} - \mathscr{A}_{\mathrm{d}}] & 0 \end{bmatrix}, \tag{3.11b}$$

$$\Xi_1 = \begin{bmatrix} (\mathscr{T}_{\mathrm{d}} - \mathscr{T}) & (\mathscr{A} - \mathscr{A}_{\mathrm{d}})^T \\ (\mathscr{A} - \mathscr{A}_{\mathrm{d}}) & 0 \end{bmatrix}, \tag{3.11c}$$

$$\Xi_0 = \begin{bmatrix} 0 \\ -X_0 \end{bmatrix}. \tag{3.11d}$$

As a result of the contraction mapping theorem [67], the method is convergent for

$$L = \|\Xi_{\mathrm{d}}^{-1}\Xi_1\| < 1. \tag{3.12}$$

The matrices are defined as follows:

$$\begin{aligned} \mathscr{A}_{\mathrm{d}} &= \mathrm{diag}(\mathscr{A}_{11}, \ldots, \mathscr{A}_{NN}), \\ \mathscr{I} &= \mathrm{diag}(\mathscr{I}_1, \ldots, \mathscr{I}_N), \\ \mathscr{T}_{\mathrm{d}} &= \mathrm{diag}(\mathscr{T}_{11}, \ldots, \mathscr{T}_{NN}), \\ \mathscr{A} &= (\mathscr{A}_{ij})_{i,j=1,\ldots,N}, \\ \mathscr{T} &= (\mathscr{T}_{ij})_{i,j=1,\ldots,N}, \\ X_0 &= (\boldsymbol{X}_{ijk'})_{i,j=1,\ldots,N}. \end{aligned}$$

An open question is a proper stopping criterion for the iterative method. We suggest one of the following three criteria: the change of the variables $\boldsymbol{z}$, i.e. $\|\boldsymbol{z}^{[l+1]} - \boldsymbol{z}^{[l]}\|/\|\boldsymbol{z}^{[l]}\| < \varepsilon_{\boldsymbol{z}}$; the change of the calculated objective functions, i.e. $\|\Phi^{[l+1]} - \Phi^{[l]}\|/\|\Phi^{[l]}\| < \varepsilon_\Phi$; or a fixed number J of total iterations. At each sampling time, the S-DMPC method needs a proper initialization of the variables $\boldsymbol{z}^{[0]}$ and $\lambda^{[0]}$. Here, we suggest to initialize these variables using the values of the last iterate of the previous sampling time, i.e. $\boldsymbol{z}^{[0]}|_k = \boldsymbol{z}^{[J]}|_{k-1}$ and $\lambda^{[0]}|_k = \lambda^{[J]}|_{k-1}$. Alternative initialization methods are the subject of future work.

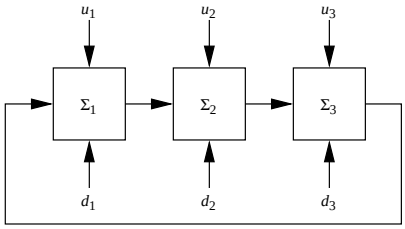


Figure 3.2: Illustration of the case study: Cascaded system with feedback and disturbances

3.4 Case study

For illustration of the method, we consider the discrete-time linear system

$$x(k+1) = A\,x(k) + B\,u(k) + D\,d(k)$$

with

$$A = \left[\begin{array}{c|c|c} A_{11} & 0 & A_0 \\ \hline A_0 & A_{22} & 0 \\ \hline 0 & A_0 & A_{33} \end{array}\right], \quad B = D = \left[\begin{array}{c|c|c} B_0 & 0 & 0 \\ \hline 0 & B_0 & 0 \\ \hline 0 & 0 & B_0 \end{array}\right],$$

$$A_{11} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0.125 & -0.75 & 2.5 \end{bmatrix}, \quad A_{22} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0.008 & -0.12 & 1.6 \end{bmatrix},$$

$$A_{33} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ -0.027 & -0.27 & 0.1 \end{bmatrix}, \; A_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.025 & 0 & 0 \end{bmatrix},$$

$$B_0 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T.$$

$x_1(k)$, $x_2(k)$, and $x_3(k) \in \mathbb{R}^3$ are the state vectors, $u_1(k)$, $u_2(k)$, and $u_3(k)$ the scalar inputs, and $d(k) \in \mathbb{R}^3$ an unknown disturbance vector. The topology of the system is depicted in Figure 3.2. The system consists of three cascaded subsystems Σ_1, Σ_2, and Σ_3, with a feedback representing a recycle. Hence, there is a strong coupling between the variables of one subsystem, but a weaker coupling between the different subsystems represented by the zero submatrices and the sparse submatrices A_0. The system considered is open-loop unstable. Its eigenvalues are depicted in Figure 3.3.

Table 3.1: Controller performance: Absolute performance Φ_{abs}, relative performance Φ_{rel} and average computing time $\bar{t}$ for the controllers considered.

Method	It.	Φ_{abs}	Φ_{rel} [%]	$\bar{t}$ [s]
Cen. MPC	–	1.94e4	–	0.112
Dec. MPC	–	1.34e5	589	3×0.026
S-DMPC	1	1.95e4	0.5	3×0.030
S-DMPC	2	1.94e4	0	3×0.059

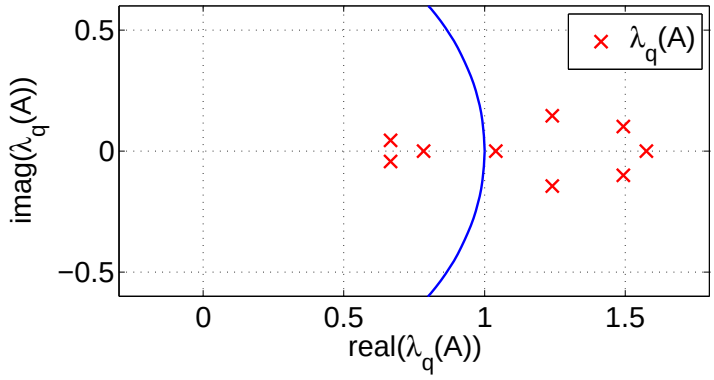


Figure 3.3: Eigenvalues of the system considered in the case study. The solid line indicates the stability bound.

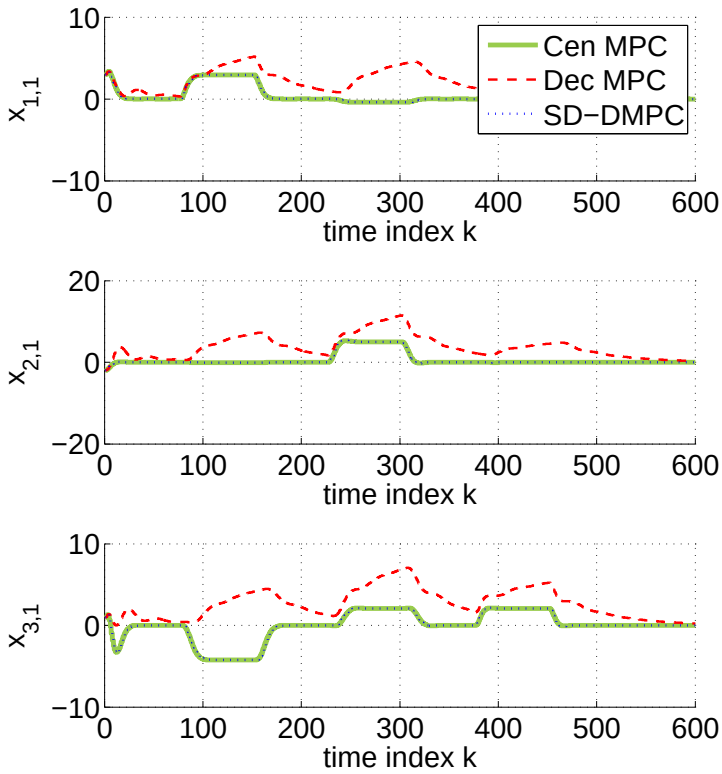


Figure 3.4: State trajectories

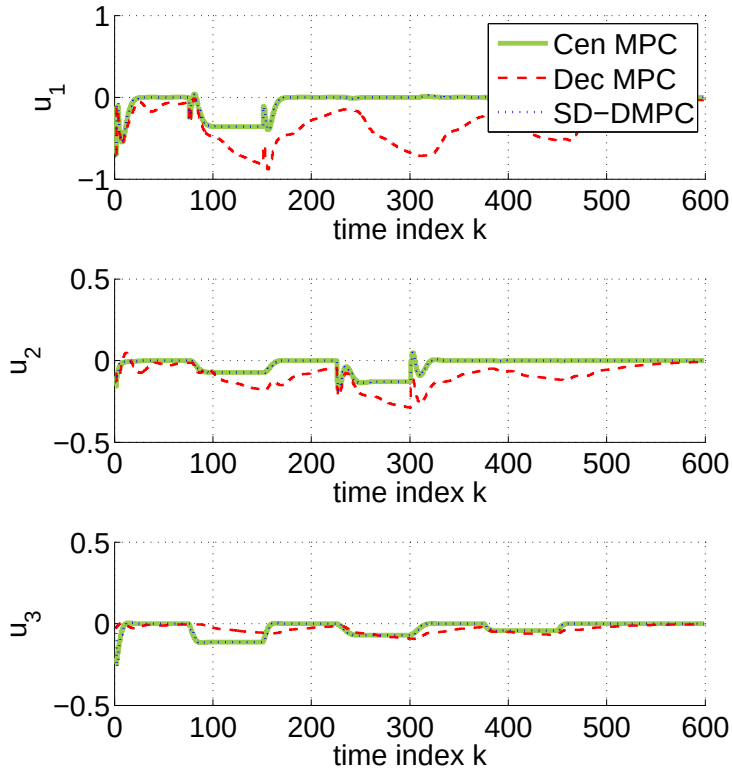


Figure 3.5: Input trajectories

The optimal control problem for this case study is defined as

$$\begin{aligned}\min_{x,u} \frac{1}{2} \sum_{k=k'}^{k'+K-1} \left(\|x(k)\|_Q^2 + \|u(k)\|_R^2\right) \\ \text{s.t. } x(k+1) = Ax(k) + Bu(k), \quad k = k', \ldots, k'+K-1, \\ x(k') = x_{k'},\end{aligned}$$

where $Q = \text{diag}(Q_{11}, Q_{22}, Q_{33})$, $R = \text{diag}(R_{11}, R_{22}, R_{33})$ with $Q_{ii} = I \in \mathbb{R}^3$, $i = 1,2,3$ and $R_{ii} = 1000$, $i = 1,2,3$ and $k' = 0, 1, \ldots, 599$. The aim is to stabilize the plant at the set point. The plant is simulated for $H = 600$ time samples. During the simulation, disturbances d_1, d_2, and d_3 with

$$d_1(k) = \begin{cases} 0.1, & \text{for } 75 \leq k \leq 150 \\ 0, & \text{else} \end{cases},$$

$$d_2(k) = \begin{cases} 0.1, & \text{for } 225 \leq k \leq 300 \\ 0, & \text{else} \end{cases},$$

$$d_3(k) = \begin{cases} 0.1, & \text{for } 375 \leq k \leq 450 \\ 0, & \text{else} \end{cases},$$

affect the system and are unknown to the controller. There is no estimation model included in the controllers. We assume state feedback in the case study.

We compare the distributed controller to a decentralized as well as a centralized controller in order to judge its performance. While the centralized controller has full knowledge on the system dynamics and solves QP (3.5), the decentralized controller as well as the distributed controller have only reduced knowledge of the dynamics of the corresponding subsystem. Hence, the DMPC solves the coupled QP (3.7), while the decentralized MPC neglects all interactions and solves the small-scale QP

$$\begin{aligned}\min_{\boldsymbol{z}_i} \frac{1}{2} \boldsymbol{z}_i^T \ \mathscr{T}_{ii} \ \boldsymbol{z}_i \\ \text{s.t. } 0 = \left[\boldsymbol{A}_{ii} - I, \quad \boldsymbol{B}_{ii}\right] \boldsymbol{z}_i + X_{iik'}, \\ \forall i \in 1,2,3, \quad k' = 0,1,2,\ldots,\end{aligned}$$

independently. For the controllers considered the control and prediction horizon is $K = 50$. For the S-DMPC method, a fixed number $J \in \{1,2\}$ of iterations is chosen.

First we analyze the mapping ζ for the system considered. The corresponding Lipschitz constant L is calculated as $L = 0.54 < 1$ (cf. equation (3.12)). Thus, the S-DMPC method is convergent and applicable for the case study.

Then, each of the three MPC schemes is implemented for the case study. Figure 3.4 shows representative trajectories for each of the subsystems, while the corresponding inputs are depicted in Figure 3.5. The decentralized MPC is hardly able to stabilize the distributed system. The simulated time horizon is not sufficient to achieve a steady-state for the closed-loop system using decentralized MPC. In contrast, for the centralized MPC (which is considered as the reference) and the S-DMPC, the system is stabilized at a new steady state almost immediately after each change in the disturbances. Figure 3.4 and 3.5 contain the results of S-DMPC with $J = 1$. A summary on the control performance is given in Table 3.1. It contains the absolute performance given by the objective function

$$\Phi_{\text{abs}} = \sum_{k=0}^{H-1} \left(\|x(k)\|_Q^2 + \|u(k)\|_R^2\right),$$

considering the complete simulation horizon and a relative performance

$$\Phi_{\text{rel}} = \frac{\Phi_{\text{abs}} - \Phi_{\text{abs,ref}}}{\Phi_{\text{abs,ref}}},$$

where $\Phi_{\text{abs,ref}}$ is the absolute performance of the reference method, i.e. the centralized MPC. While the decentralized MPC clearly is not able to compete with the centralized MPC, the distributed MPC scheme proposed is capable to achieve almost optimal results at a very low number of iterations. For $J = 1$, the loss of performance, compared to the reference solution is only 0.5%, while for $J = 2$ optimal results can be achieved. A key requirement for this low number of iterations is a proper initialization of the variables $\boldsymbol{z}$ and λ. Although, a simple initialization is chosen in this work, this is very effective. In addition, Table 3.1 provides the average computing time[2] $\bar{t}$ for each of the methods implemented. The average computing time can be reduced for the decentralized as well as the distributed MPC, in particular if the different controllers use different processors. Then, for $J = 2$ iterations, the average computing time $\bar{t} = 0.059s$ of SD-DMPC is reduced by 47% compared to the average computing time $\bar{t} = 0.112$ of the centralized MPC.

3.5 Conclusions

We have adopted the concept of Sensitivity-Driven Distributed Model Predictive Control (S-DMPC) to the control of discrete-time systems. The optimal control formulation includes a sum of non separable cost terms, i.e. the cost functions is non-additive. Though, the coordination mechanism can handle the non-separable cost function. Hence, e.g. arbitrary final costs can be included. A necessary condition of convergence is provided to prove the applicability of the method. Coordination is possible in case the internal coupling of each subsystem is strong compared to the coupling between the subsystems. The method has been successfully applied to a case study. An almost optimal control sequence can be achieved after only one iteration in this case, while a significant reduction of the computing time compared to a centralized MPC can be observed. While we did not consider any inequality constraints in this work, the method can be extended to consider inequality constraints for the inputs u and the state variables x. In addition the method can be extended to accelerate convergence. For details of both extensions we refer to [5].

A remaining task is to explicitly consider stability of the closed-loop system, while in the case study provided, stability is achieved by a sufficiently large prediction horizon. In addition, a major challenge is to extend the results presented to nonlinear systems. Finally, the method has to be implemented for large-scale systems.

[2] The controllers have been implemented using the standard Matlab QP solver `quadprog` on an Intel Core2 Quad Q6700 machine. Only one core has been assigned to Matlab.

Chapter 4

Distributed non-cooperative MPC with neighbor-to-neighbor communication

This research of this chapter has been developed by Marcello Farina and Riccardo Scattolini, Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan, Italy and will be presented at the 2011 IFAC World Congress [9].

4.1 Introduction

The majority of results in the field of model predictive control (MPC) have been developed under the assumption that the available data and information are gathered in single location and processed by a single computer. Unfortunately, many industrial problems cannot be solved in a centralized fashion, such as those arising when dealing with large scale systems [12].
Real-world examples of large scale systems are power networks [11, 68, 69] transport networks [12, 70, 71] and hydro power plants [72, 73], which are characterized by strongly interacting and spatially distributed subsystems, possibly with uncertainties, and which may suffer from limited computation capabilities and transmission load. In [74] the authors call for new ideas for dividing the centralized control synthesis problem into almost independent subproblems and for coping with limited computational capabilities and memory, as well as with uncertainties and perturbations. In [75] it is remarked that another important challenge is to reduce, as much as possible, the information exchange among the subsystems, in order to satisfy technological constraints and for economical reasons.
In the last years many distributed control structures have been developed based on MPC techniques. Specifically, completely decentralized architectures [76, 77], distributed schemes (see, e.g., [65, 78, 79, 25, 80, 81], just to mention some recent contributions) and coordinated control techniques for independent (dynamically uncoupled) systems [82, 83, 24, 84] have been proposed.
Focusing more specifically on distributed MPC schemes [21], they can be classified according to the topology of the transmission network (i.e., *fully connected* or *partially connected* networks), to the information exchange protocol needed (i.e., *non-iterative* or *iterative algorithms*) and to the type of cost function which is optimized (i.e., *cooperative* or *non-cooperative* algorithms).
For example, in [65] a non-iterative, non-cooperative distributed MPC technique is proposed for continuous-time systems based on neighbor-to-neighbor information exchange, where it is required that each subsystems knows only its own dynamic subsystem and how the neighboring subsystems states affect its dynamics. In [79] two distributed cooperative MPC algorithms for continuous-time systems are proposed, where each actuator is required to know the overall system model: specifically,

the authors propose a non-iterative sequential (partially connected) algorithm based on previous results presented in [78], and a novel iterative fully connected one.
In [25, 80, 81] a cooperative fully connected output-feedback MPC algorithm for discrete time systems is discussed, where only input constraints can be assigned and full knowledge on the system dynamics is required to all the subsystems. Interestingly, this algorithm guarantees stability both in its iterative and non-iterative formulation, while optimal performance (i.e., equivalent to the performances provided by a centralized MPC algorithm) are attained when an iterative transmission protocol is employed.
In this work we propose a non-iterative, non-cooperative MPC algorithm where a neighbor-to-neighbor (i.e., partially connected) communication network and partial (regional) structural information are needed. The rationale of the proposed technique is that, at each sampling time, each subsystem sends to its neighbor information about its future reference trajectory, and guarantees that the actual trajectory lies within a certain bound in the neighborhood of the reference one. Then, a robust MPC approach inspired by [10] provides a tool for the statement of the local optimization problems solved by each subsystem.
The proposed algorithm handles input and state constraints and, under mild assumptions on the existence of a suitable decentralized auxiliary control law, we prove convergence of the closed loop control system.
The highlights of the proposed approach are: (*i*) it is not necessary for each subsystem to know the dynamical models governing the trajectories of the other subsystems (not even the ones of the neighbors); (*ii*) the transmission of information is limited, in that each subsystem needs the reference trajectories only of the variables of one's neighbors which actually affect its dynamics (which is normally a narrow subset of the neighbors' variables); (*iii*) its rationale is very similar to the MPC algorithms presently employed in industry, where reference trajectories tailored on the dynamics of the system under control are used; (*iv*) the required transmission burden is even smaller in steady state conditions, when no transmission would be needed even in presence of small perturbations of the subsystems' trajectories, while the variables affecting the state variables lie within a given set.
Notation. We use the short-hand $\mathbf{v} = (v_1, \ldots, v_s)$ to denote a column vector with s (not necessarily scalar) components $v_1, \ldots, v_s$. The symbol $\oplus$ denotes the Minkowski sum, namely $C = A \oplus B$ if and only if $C = \{c : c = a + b, \text{ for all } a \in A, b \in B\}$. We also denote $\bigoplus_{i=1}^{M} A_i = A_1 \oplus \cdots \oplus A_M$. For a discrete-time signal s_t and $a, b \in \mathbb{N}$, $a \leq b$, we denote $(s_a, s_{a+1}, \ldots, s_b)$ with $s_{[a:b]}$. Finally, a continuous function $\alpha : \mathbb{R}_+ \to \mathbb{R}_+$ is a $\mathscr{K}_\infty$ function iff $\alpha(0) = 0$, it is strictly increasing and $\alpha(s) \to +\infty$ as $s \to +\infty$.

4.2 Partitioned systems

Consider a process which obeys to the linear dynamics

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \tag{4.1}$$

where $\mathbf{x}_t \in \mathbb{R}^n$ is the state vector and $\mathbf{u}_t \in \mathbb{R}^m$ is the input signal.
Let the system (4.1) be partitioned in M low order interconnected non overlapping subsystems, where a generic submodel has $x_t^{[i]} \in \mathbb{R}^{n_i}$ as state vector, i.e., $\mathbf{x}_t = (x_t^{[1]}, \ldots, x_t^{[M]})$ and $\sum_{i=1}^{M} n_i = n$. According to this decomposition, the state transition matrices $A_{11} \in \mathbb{R}^{n_1 \times n_1}, \ldots, A_{MM} \in \mathbb{R}^{n_M \times n_M}$ of the M subsystems are diagonal blocks of $\mathbf{A}$, whereas the non-diagonal blocks of $\mathbf{A}$ (i.e., A_{ij}, with $i \neq j$) define the coupling terms between subsystems.
The partition performed on the system induces a interconnected network of subsystems, which can be naturally described by means of a directed graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, where the nodes in $\mathscr{V}$ are the subsystems

and the edge (j,i) in the set $\mathscr{E} \subseteq \mathscr{V} \times \mathscr{V}$ models that the state of j affects the dynamics of subsystem i. In particular, $(j,i) \in \mathscr{E}$ if and only if $A_{ij} \neq 0$. We denote as $\mathscr{N}_i$ the set of neighbors of subsystem i (which excludes i) i.e., $\mathscr{N}_i = \{j | j \neq i \text{ and } (j,i) \in \mathscr{E}\}$.
Furthermore, we assume that the input $\mathbf{u}_t$ can be partitioned into a set of M input vectors $u_t^{[i]} \in \mathbb{R}^{m_i}$, with $i = 1,\ldots,M$, where we assume that $u_t^{[i]}$ directly affects only the state of the i-th subsystem $x_t^{[i]}$. This implies that $\mathbf{B}$ has a block diagonal structure $\mathbf{B} = \text{diag}(B_1,\ldots,B_M)$, where $B_i \in \mathbb{R}^{n_i \times m_i}$ for all $i = 1,\ldots,M$. It finally results that the i-th subprocess obeys to the linear dynamics

$$x_{t+1}^{[i]} = A_{ii} x_t^{[i]} + B_i u_t^{[i]} + \sum_{j \in \mathscr{N}_i} A_{ij} x_t^{[j]} \tag{4.2}$$

where $x_t^{[i]} \in \mathbb{X}_i \subseteq \mathbb{R}^{n_i}$ is the state vector and $u_t^{[i]} \in \mathbb{U}_i \subseteq \mathbb{R}^{m_i}$ is the input vector. The sets $\mathbb{X}_i$ and $\mathbb{U}_i$ are convex neighborhoods of the origin. Furthermore we define $\mathbb{X} = \prod_{i=1}^M \mathbb{X}_i \subseteq \mathbb{R}^n$ and $\mathbb{U} = \prod_{i=1}^M \mathbb{U}_i$, which are convex by convexity of $\mathbb{X}_i$ and $\mathbb{U}_i$, respectively, for $i = 1,\ldots,M$. When $\mathbb{X} = \mathbb{R}^n$ and $\mathbb{U} = \mathbb{R}^m$ we say that the system is unconstrained.

Our aim is to design, for each of the subsystems, an algorithm for computing an input sequence $u_t^{[i]}$ based on the state $x_t^{[i]}$ and some information which is transmitted by i-th neighboring subsystems, which guarantees closed loop asymptotic convergence of the state of the large scale system (4.1), the minimization of a given local cost function and constraint satisfaction. Specifically, we assume that each subsystem has a reference trajectory $\tilde{x}_t^{[i]}$ which is transmitted to the subsystems which have i as neighbor, and which is incrementally defined (as better specified in the following). We also assume that one can guarantee that, for all $t \geq 0$, the real local state trajectory $x_t^{[i]}$ lies in a specified time-invariant neighborhood of $\tilde{x}_t^{[i]}$ i.e, $x^{[i]} - \tilde{x}_t^{[i]} \in \mathscr{E}_i$, where $0 \in \mathscr{E}_i$.
Letting $w_t^{[i]} = \sum_{j \in \mathscr{N}_i} A_{ij}(x_t^{[j]} - \tilde{x}_t^{[j]})$, the i-th system model (4.2) can be written as follows

$$x_{t+1}^{[i]} = A_{ii} x_t^{[i]} + B_i u_t^{[i]} + \sum_{j \in \mathscr{N}_i} A_{ij} \tilde{x}_t^{[j]} + w_t^{[i]} \tag{4.3}$$

where the term $w_t^{[i]} \in \mathbb{W}_i = \bigoplus_{j \in \mathscr{N}_i} A_{ij} \mathscr{E}_j$ represents a bounded disturbance affecting equation (4.3) and $\sum_{j \in \mathscr{N}_i} A_{ij} \tilde{x}_t^{[j]}$ can be considered as a known input. Provided that, for all $i = 1,\ldots,M$, the constraint $x^{[i]} - \tilde{x}_t^{[i]} \in \mathscr{E}_i$ is satisfied for all $t \geq 0$, we can cast the problem of designing a distributed MPC control law as the problem of designing a robust control law for the subsystem (4.3), for all $i = 1,\ldots,M$.
To this aim, we rely on the robust MPC algorithm presented in [10] for constrained linear systems with bounded disturbances for the statement of the local MPC sub-problems (which will be denoted i-MPC problems). The two main advantages of this approach are that no burdensome min-max problem is required to be solved on-line, and that it naturally provides the reference trajectory $\tilde{x}_t^{[i]}$, which is one of the key points of the algorithm presented in this paper (see Section 4.3 for details).

4.3 The distributed MPC algorithm

As a preliminary step to the statement of the local i-MPC problem, we define the i-th subsystem nominal model associated to equation (4.3)

$$\hat{x}_{t+1}^{[i]} = A_{ii} \hat{x}_t^{[i]} + B_i \hat{u}_t^{[i]} + \sum_{j \in \mathscr{N}_i} A_{ij} \tilde{x}_t^{[j]} \tag{4.4}$$

The control law for the real i-th subsystem (4.3) will be assigned, for all $t \geq 0$, according to

$$u_t^{[i]} = \hat{u}_t^{[i]} + K_i^{aux}(x_t^{[i]} - \hat{x}_t^{[i]}) \tag{4.5}$$

where K_i^{aux} is a suitable control gain. If we define $\varepsilon_t^{[i]} = x_t^{[i]} - \hat{x}_t^{[i]}$ we obtain, from (4.3) and (4.5)

$$\varepsilon_{t+1}^{[i]} = (A_{ii} + B_i K_i^{aux})\varepsilon_t^{[i]} + w_t^{[i]} \tag{4.6}$$

where $w_t^{[i]} \in \mathbb{W}_i$. Since $\mathbb{W}_i$ is bounded, if $(A_{ii} + B_i K_i^{aux})$ is Schur, then there exists a robust positively invariant (RPI) set Z_i for (4.6) such that, for all $\varepsilon_t^{[i]} \in Z_i$, then $\varepsilon_{t+1}^{[i]} \in Z_i$. A method for computing polytopic, robust positively invariant, outer approximations of the minimal robust positively invariant set is proposed in [85]. From (4.6) it follows that, if $u_k^{[i]}$ is computed as in (4.5) for all $k \geq t$, then

$$x_t^{[i]} - \hat{x}_t^{[i]} \in Z_i \tag{4.7}$$

implies that $x_k^{[i]} - \hat{x}_k^{[i]} \in Z_i$ for all $k \geq t$.
Now write

$$x_t^{[i]} - \tilde{x}_t^{[i]} = (x_t^{[i]} - \hat{x}_t^{[i]}) + (\hat{x}_t^{[i]} - \tilde{x}_t^{[i]})$$

and define the set E_i for all $i = 1, \ldots, M$ as a set containing the origin and satisfying $E_i \oplus Z_i \subseteq \mathscr{E}_i$. Since, in view of (4.7), $x_k^{[i]} - \hat{x}_k^{[i]} \in Z_i$ for all $k \geq t$, if we also satisfy the constraint

$$\hat{x}_k^{[i]} - \tilde{x}_k^{[i]} \in E_i \tag{4.8}$$

for all $k \geq t$, then $x_k^{[i]} - \tilde{x}_k^{[i]} \in \mathscr{E}_i$ for all $k \geq t$ as required.
Now we are in the position to state the local minimization problem for all subsystems at instant t. Given the future reference trajectory of i and its neighbors $\tilde{x}_k^{[j]}$, $k = t, \ldots, t+N-1$, $j \in \mathscr{N}_i \cup \{i\}$, the i-MPC problem consists in the following

$$\min_{\hat{x}_t^{[i]}, \hat{u}_{[t:t+N-1]}^{[i]}} \quad V_i^N(\hat{x}_t^{[i]}, \hat{u}_{[t:t+N-1]}^{[i]}) \tag{4.9}$$

subject to the dynamic constraints (4.4), the static constraints (4.7), (4.8),

$$\hat{x}_k^{[i]} \in \hat{\mathbb{X}}_i \tag{4.10}$$

$$\hat{u}_k^{[i]} \in \hat{\mathbb{U}}_i \tag{4.11}$$

where $\hat{\mathbb{X}}_i \oplus Z_i \subseteq \mathbb{X}_i$ and $\hat{\mathbb{U}}_i \oplus K Z_i \subseteq \mathbb{U}_i$, and the terminal constraint

$$\hat{x}_{t+N}^{[i]} \in \hat{\mathbb{X}}_i^F \tag{4.12}$$

where $\hat{\mathbb{X}}_i^F$ is the i-th nominal subsystem terminal set, whose properties will be specified in the following.
The cost function $V_i^N(\hat{x}_t^{[i]}, \hat{u}_{[t:t+N-1]}^{[i]})$ is

$$V_i^N(\hat{x}_t^{[i]}, \hat{u}_{[t:t+N-1]}^{[i]}) = \sum_{k=t}^{t+N-1} l_i(\hat{x}_k^{[i]}, \hat{u}_k^{[i]}) + V_i^F(\hat{x}_{t+N}^{[i]}) \tag{4.13}$$

where $l_i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}_+$ is the stage cost and $V_i^F : \mathbb{R}^{n_i} \to \mathbb{R}_+$ is the final cost. From now on, we assume that l_i is defined in such a way that $l_i(0,0) = 0$ and that there exists, for all $i = 1, \ldots, M$, a $\mathscr{K}_\infty$ function α and a matrix R_i satisfying $\text{rank}([B_i^T \; R_i^T]^T) = m_i$ such that $l_i(x^{[i]}, u^{[i]}) \geq \alpha(\|(x^{[i]}, R_i u^{[i]})\|)$ for all $x^{[i]} \in \mathbb{R}^{n_i}$, $u^{[i]} \in \mathbb{R}^{m_i}$. As in [10], in the stated optimization problem minimization is performed

with respect both to the nominal system initial state $\hat{x}_t^{[i]}$ and to the nominal input trajectory $\hat{u}_{[t:t+N-1]}^{[i]}$. Letting the pair $\hat{x}_{t/t}^{[i]}, \hat{u}_{[t:t+N-1]/t}^{[i]}$ be the solution to the *i*-MPC problem (4.9) at time t, and according to the receding horizon paradigm, we assign the input to the nominal system (4.4), at time t, as $\hat{u}_{t/t}^{[i]}$. According to (4.5), the input to the real system (4.2), at instant t, is

$$u_t^{[i]} = \hat{u}_{t/t}^{[i]} + K_i^{aux}(x_t^{[i]} - \hat{x}_{t/t}^{[i]}) \tag{4.14}$$

Furthermore, let us define as $\hat{x}_{k/t}^{[i]}$ the trajectory stemming from $\hat{x}_{t/t}^{[i]}$ and $\hat{u}_{[t:t+N-1]/t}^{[i]}$, in view of equation (4.4). The value of the reference state variable $\tilde{x}_{t+N}^{[i]}$ is set to

$$\tilde{x}_{t+N}^{[i]} = \hat{x}_{t+N/t}^{[i]} \tag{4.15}$$

We stress that we do not define, at each instant t, a new reference trajectory $\tilde{x}_k^{[i]}$, $k = t+1,\dots,t+N$, but we append the value $\tilde{x}_{t+N}^{[i]}$ to the reference trajectory which has been already defined for $k \le t+N-1$. A definition is required to define the set of admissible initial conditions $\mathbf{x}_0 = (x_0^{[1]},\dots,x_0^{[M]})$ and initial reference trajectories $\tilde{x}_{[0:N-1]}^{[j]}$, for all $j = 1\dots,M$.

Definition 4.3.1 *Writing* $\mathbf{x} = (x^{[1]},\dots,x^{[M]})$*, we denote the feasibility region* $\mathbb{X}^N$ *for all the i-MPC problems as the set*

$$\begin{aligned}\mathbb{X}^N := \quad & \{\mathbf{x}: \text{ if } x_0^{[i]} = x^i \text{ for all } i = 1,\dots,M \\ & \text{then } \exists(\tilde{x}_{[0:N-1]}^{[1]},\dots,\tilde{x}_{[0:N-1]}^{[M]}), (\hat{x}_{0/0}^{[1]},\dots,\hat{x}_{0/0}^{[M]}), \\ & (\hat{u}_{[0:N-1]}^{[1]},\dots,\hat{u}_{[0:N-1]}^{[M]}) \text{ such that (4.2), (4.7), (4.8),} \\ & \text{(4.10)-(4.12) are satisfied for all } i = 1,\dots,M\}\end{aligned}$$

We also denote, for each $\mathbf{x} \in \mathbb{X}^N$*, the region of acceptable initial reference trajectories as*

$$\begin{aligned}\tilde{\mathbb{X}}_{\mathbf{x}} := \quad & \{(\tilde{x}_{[0:N-1]}^{[1]},\dots,\tilde{x}_{[0:N-1]}^{[M]}): \text{ if } x_0^{[i]} = x^i \text{ for all } i = 1,\dots,M \\ & \text{then } \exists(\hat{x}_{0/0}^{[1]},\dots,\hat{x}_{0/0}^{[M]}), (\hat{u}_{[0:N-1]}^{[1]},\dots,\hat{u}_{[0:N-1]}^{[M]}) \text{ such that} \\ & \text{(4.2), (4.7), (4.8), (4.10)-(4.12) are satisfied for all } i = 1,\dots,M\}\end{aligned}$$

4.4 Convergence results

The following assumptions are needed to state the main results of the paper.

Assumption 4.4.1 *The matrix* $A_{ii} + B_i K_i^{aux}$ *is Schur, for all* $i = 1,\dots,M$.

Assumption 4.4.2 *Letting* $\mathbf{K}^{aux} = diag(K_1^{aux},\dots,K_M^{aux})$, $\hat{\mathbb{X}} = \prod_{i=1}^M \hat{\mathbb{X}}_i$, $\hat{\mathbb{U}} = \prod_{i=1}^M \hat{\mathbb{U}}_i$ *and* $\hat{\mathbb{X}}^F = \prod_{i=1}^M \hat{\mathbb{X}}_i^F$, *it holds that:*

(i) *The matrix* $\mathbf{A} + \mathbf{B}\mathbf{K}^{aux}$ *is Schur;*

(ii) $\hat{\mathbb{X}}^F \subseteq \hat{\mathbb{X}}$ *is an invariant set for* $\hat{\mathbf{x}}^+ = (\mathbf{A} + \mathbf{B}\mathbf{K}^{aux})\hat{\mathbf{x}}$*;*

(iii) $\hat{\mathbf{u}} = \mathbf{K}^{aux}\hat{\mathbf{x}} \in \hat{\mathbb{U}}$ *for any* $\hat{\mathbf{x}} \in \hat{\mathbb{X}}^F$*;*

(iv) for all $\hat{\mathbf{x}} \in \hat{\mathbb{X}}^F$ *and, for a given constant* $\kappa > 0$

$$\mathbf{V}^F\left(\hat{\mathbf{x}}^+\right) - \mathbf{V}^F\left(\hat{\mathbf{x}}\right) \leq -(1+\kappa)\mathbf{l}(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \tag{4.16}$$

where $\mathbf{V}^F(\hat{\mathbf{x}}) = \sum_{i=1}^M V_i^F(\hat{x}^{[i]})$ *and* $\mathbf{l}(\hat{\mathbf{x}}, \hat{\mathbf{u}}) = \sum_{i=1}^M l_i(\hat{x}^{[i]}, \hat{u}^{[i]})$.

Assumption 4.4.3 *Given the sets* $\mathscr{E}_i$, *where* $0 \in \mathscr{E}_i$ *for all* $i = 1\ldots,M$, *and the RPI sets* Z_i *for equations* (4.6), *there exists a real positive constant* $\bar{\rho}_E > 0$ *such that* $Z_i \oplus \mathscr{B}_{\bar{\rho}_E}(0) \subseteq \mathscr{E}_i$ *for all* $i = 1,\ldots,M$, *where* $\mathscr{B}_{\bar{\rho}_E}(0)$ *is a ball of radius* $\bar{\rho}_E > 0$ *centered at the origin.*

Now we are in the position to state the main result.

Theorem 1 *Let Assumptions 4.4.1-4.4.3 be satisfied and let* E_i *be a neighborhood of the origin satisfying* $E_i \oplus Z_i \subseteq \mathscr{E}_i$. *Then the trajectory* $\mathbf{x}_t$, *starting from any initial condition* $\mathbf{x}_0 \in \mathbb{X}^N$, *asymptotically converges to the origin, provided that the initial reference trajectories are in* $\hat{\mathbb{X}}_{\mathbf{x}_0}$.

Proof 1 *See Appendix 4.7*

Notice that the fulfillment of Assumptions 4.4.1, 4.4.2 requires the design of a decentralized auxiliary control law which, at the same time, (a) stabilizes the local subsystems when neglecting the interconnections, (b) stabilizes the overall large scale system, (c) has a Lyapunov function which basically corresponds to a weighted sum of local Lyapunov functions.
The above-mentioned issues can be addressed using a number of well-established results, worked out in the past in the field of decentralized control. For instance, one can rely on milestone results on *connective stability* [11], vector Lyapunov functions and the so-called "weighted sum approach" for proving connective stability [12]. More recently, problems (a) and (b) have been successfully addressed in [13], where a small gain condition for large-scale (nonlinear) systems has been derived. Similar concepts can be used to provide conditions for the validity of Assumption 4.4.3.

4.5 Example

Consider a fourth-order system with two input variables. The dynamics of the system is described by (4.4), where

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}$$

and

$$A_{11} = A_{22} = \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}, A_{12} = -A_{21} = \begin{bmatrix} 0.1 & 0.1 \\ 0 & 0.3 \end{bmatrix}, B_1 = B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The following constraints are set to the input signals: $-2.5 \leq u_k^{[1]} \leq 2.5$ and $-4 \leq u_k^{[2]} \leq 4$. We define, in (4.13), $l_i(\hat{x}_k^{[i]}, \hat{u}_k^{[i]}) = \frac{1}{2}\|\hat{x}_k^{[i]}\|_{Q_i}^2$ and $V_i^F(\hat{x}_{t+N}^{[i]}) = \frac{1}{2}\|\hat{x}_{t+N}^{[i]}\|_{P_i}^2$, where $P_1 = P_2 = \text{diag}(1,3)$ and $Q_1 = Q_2 = \text{diag}(0.4593, 0.4593)$. Setting $K_1 = K_2 = \begin{bmatrix} 1 & -2 \end{bmatrix}$, we verify Assumptions 4.4.1 and 4.4.2. In the simulations, we set $N = 4$ and the reference trajectories are initialized by simulating the subsystems controlled using the auxiliary control law, where coupling terms are neglected, that is $\tilde{x}_k^{[i]} = (A_{ii} + B_i K_i)^k x_0^{[i]}$, for $k = 0,\ldots,t+N-1$. A choice of the sets Z_i, E_i and $\mathbb{W}_i$, $i = 1,2$, consistent with Assumption 4.4.3, is shown in Fig. 4.1 (grey ellipsoids), where the black

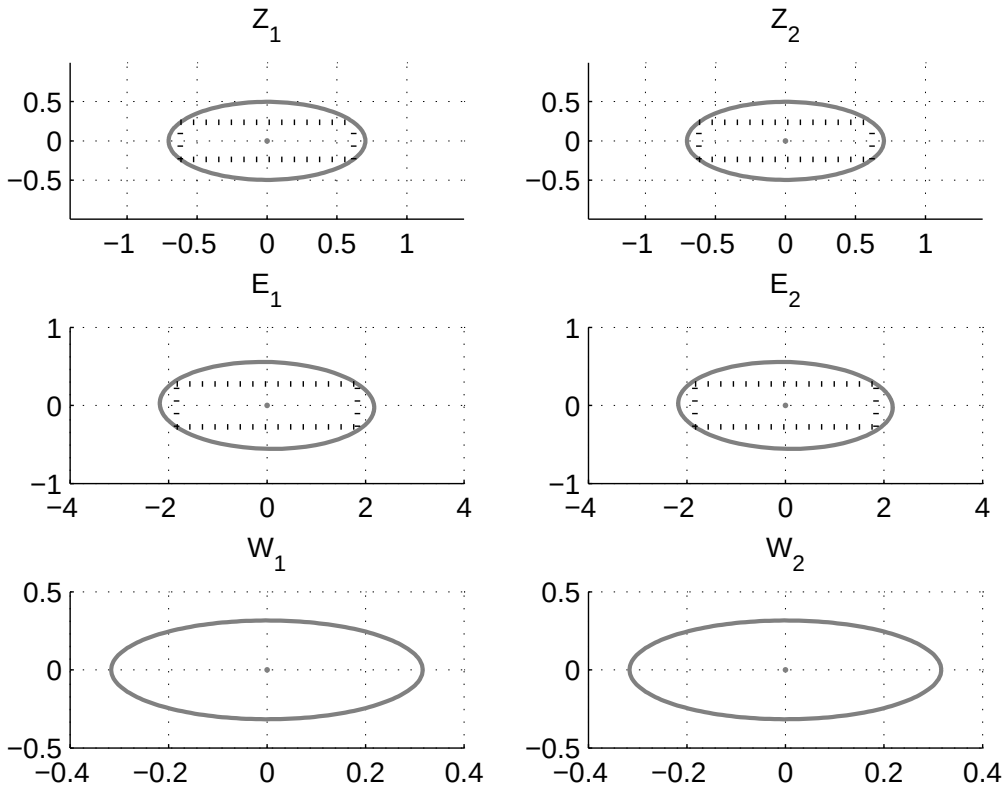


Figure 4.1: Sets Z_i, E_i and $\mathbb{W}_i$, $i = 1,2$, chosen in the given example. The black dotted lines represent the real constraints exerted in the example.

dotted lines represent the real constraints exerted, for simplicity, while solving the constrained optimization problems i-MPC, $i = 1,2$.

In Fig. 4.2 the plots of the optimal input trajectories obtained with the distributed MPC algorithm (dMPC) are shown and compared with those obtained with a centralized MPC (cMPC). Notably, at time $t = 0$ the constraint on $\hat{u}_t^{[1]}$ is active, while it is apparent that the constraint on the real input variable $u_t^{[1]}$ is far from being violated. This clearly shows that the robustness argument used to define the distributed MPC leads to a level of conservativeness in the solution of the problem which is directly proportional to the dimension of the uncertainty sets.

In Fig. 4.3 we compare the optimal trajectories obtained with the proposed scheme with the ones obtained using a centralized MPC controller. These results show that, in the specific case considered here, the performance degradation is not significant.

4.6 Conclusions

In this paper we have proposed a novel non-iterative, non-cooperative distributed MPC algorithm. Under mild assumptions on the existence of a suitable decentralized auxiliary control law, convergence of the closed loop control system can be guaranteed. As it is discussed, according to this approach to

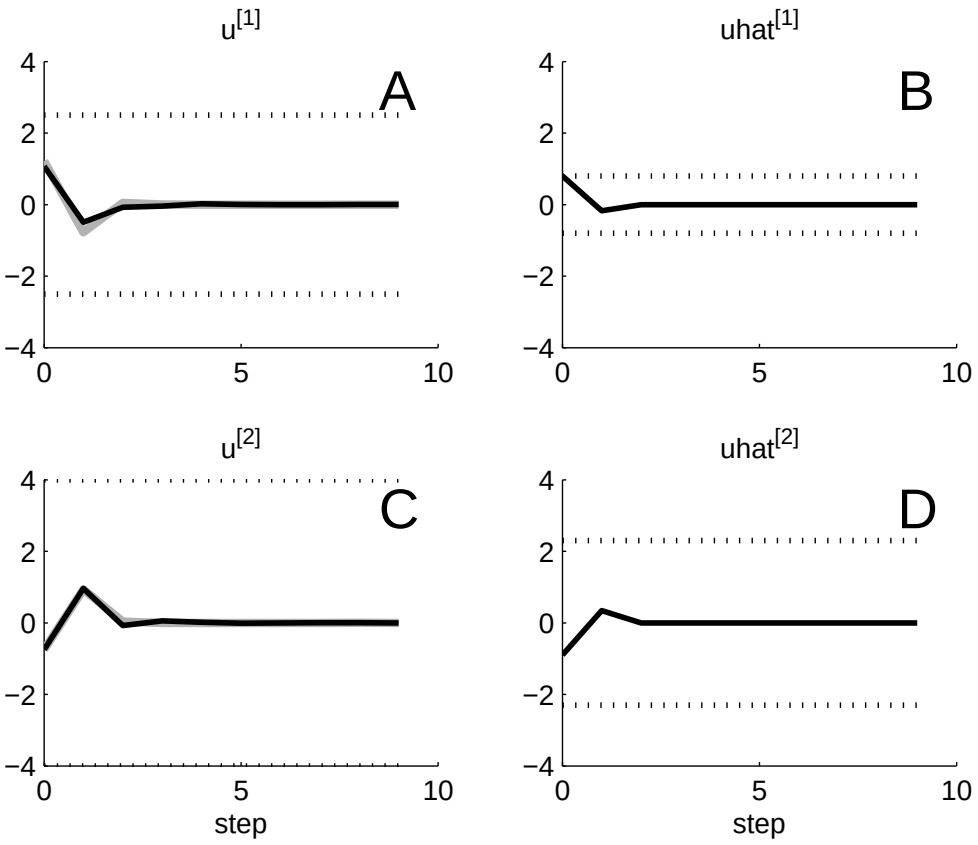


Figure 4.2: Input trajectories. Panel A: $u_t^{[1]}$ (black solid line), input obtained with cMPC (grey solid line), thresholds for $u_t^{[1]}$ (black dotted lines). Panel B: $\hat{u}_t^{[1]}$ (black solid line), thresholds for $\hat{u}_{t/t}^{[1]}$ (black dotted lines). Panel C: $u_t^{[2]}$ (black solid line), inputs obtained with cMPC (grey solid line), thresholds for $u_t^{[2]}$ (black dotted lines). Panel B: $\hat{u}_{t/t}^{[2]}$ (black solid line), thresholds for $\hat{u}_t^{[2]}$ (black dotted lines).

distributed control, no information on other subsystem's dynamics is required to the subsystems, and the data received by system i is a subset of $\tilde{x}_{t+N}^{[j]}$, only if j is a neighbor of i.
Although this paper establishes the main algorithm and convergence results, much work has still to be devoted to the problem of enhancing its applicability. The following issues are of paramount importance: (a) to give simple conditions on the system under control and to provide constructing criteria in order to fulfill Assumptions 4.4.1-4.4.3; (b) to give criteria to chose sets Z_i and E_i; (c) to give criteria for optimal choices of the initial reference trajectories $\tilde{x}_{[0:N-1]}^{[i]}$.
Since the proposed method strongly relies on robustness concepts, a further improvement to the proposed scheme is envisaged, coping with uncertainties on the model of how the state variables of a subsystem affect the dynamics of the neighbors.

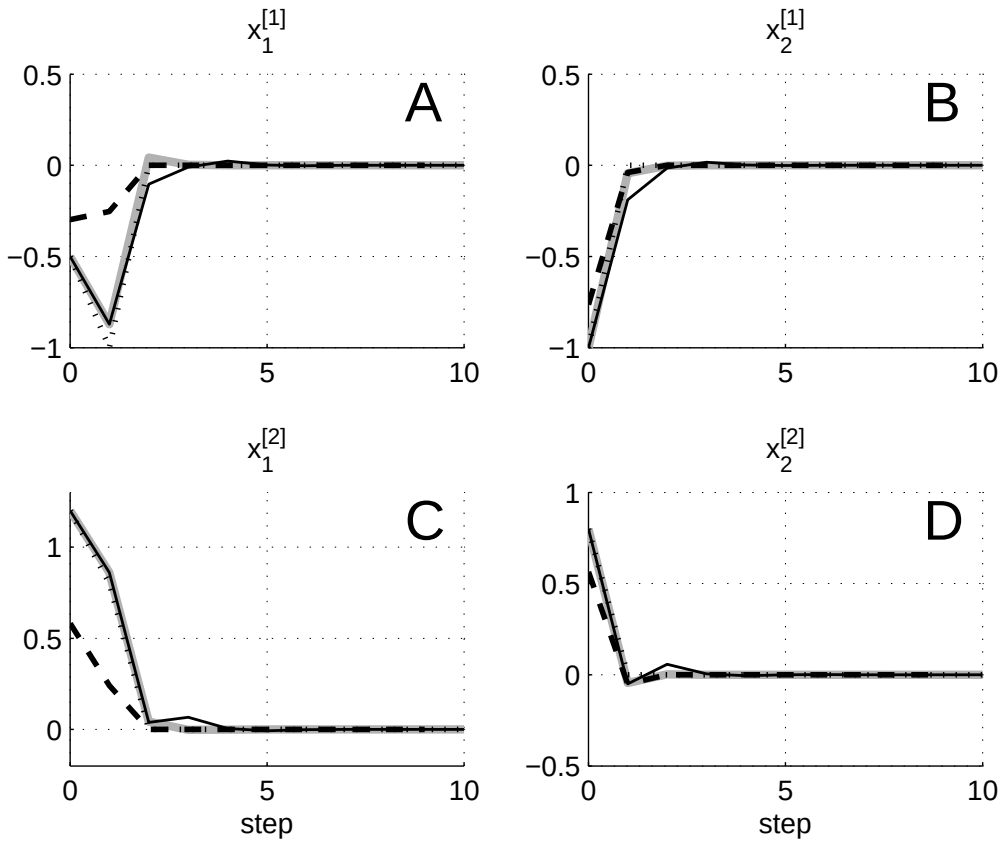


Figure 4.3: Controlled state variables with dMPC (black solid lines) and with cMPC (grey solid lines), $\tilde{x}_t^{[i]}$ (dotted lines) and $\hat{x}_{t/t}^{[i]}$ (dashed lines). Panel A: first element of $x_t^{[1]}$. Panel B: second element of $x_t^{[1]}$. Panel C: first element of $x_t^{[2]}$. Panel D: second element of $x_t^{[2]}$.

4.7 Appendix: Proof of Theorem 1

4.7.1 The collective problem

Define the collective vectors $\hat{\mathbf{x}}_t = (\hat{x}_t^{[1]},\ldots,\hat{x}_t^{[M]})$, $\tilde{\mathbf{x}}_t = (\tilde{x}_t^{[1]},\ldots,\tilde{x}_t^{[M]})$, $\hat{\mathbf{u}}_t = (\hat{u}_t^{[1]},\ldots,\hat{u}_t^{[M]})$, $\mathbf{w}_t = (w_t^{[1]},\ldots,w_t^{[M]})$ and $\boldsymbol{\varepsilon}_t = (\varepsilon_t^{[1]},\ldots,\varepsilon_t^{[M]})$. Furthermore, define the matrices $\mathbf{A}^* = \text{diag}(A_{11},\ldots,A_{MM})$ and $\tilde{\mathbf{A}} = \mathbf{A} - \mathbf{A}^*$. Collectively, write equations (4.3) and (4.4) as

$$\mathbf{x}_{t+1} = \mathbf{A}^*\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \tilde{\mathbf{A}}\tilde{\mathbf{x}}_t + \mathbf{w}_t \tag{4.17}$$

$$\hat{\mathbf{x}}_{t+1} = \mathbf{A}^*\hat{\mathbf{x}}_t + \mathbf{B}\hat{\mathbf{u}}_t + \tilde{\mathbf{A}}\tilde{\mathbf{x}}_t \tag{4.18}$$

In view of (4.5), $\mathbf{u}_t = \hat{\mathbf{u}}_t + \mathbf{K}^{aux}(\mathbf{x}_t - \hat{\mathbf{x}}_t)$, and collectively write (4.6) as

$$\boldsymbol{\varepsilon}_{t+1} = (\mathbf{A}^* + \mathbf{B}\mathbf{K}^{aux})\boldsymbol{\varepsilon}_t + \mathbf{w}_t \tag{4.19}$$

Since each i-MPC problem depends upon local variables (note, in fact, that the coupling terms $\tilde{x}_k^{[i]}$ are fixed for all $k = t, \ldots, t+N-1$), minimizing (4.9) for all $i = 1, \ldots, M$ is equivalent to minimize

$$\mathbf{V}^{N*}(\mathbf{x}_t) = \min_{\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{[t:t+N-1]}} \mathbf{V}^N(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{[t:t+N-1]}) \tag{4.20}$$

subject to the dynamic constraints (4.18), the static constraints

$$\mathbf{x}_t - \hat{\mathbf{x}}_t \in \mathbb{Z} = \prod_{i=1}^{M} Z_i \tag{4.21a}$$

$$\hat{\mathbf{x}}_k - \tilde{\mathbf{x}}_k \in \mathbb{E} = \prod_{i=1}^{M} E_i \tag{4.21b}$$

$$\hat{\mathbf{x}}_k \in \hat{\mathbb{X}} \tag{4.21c}$$

$$\hat{\mathbf{u}}_k \in \hat{\mathbb{U}} \tag{4.21d}$$

and the terminal constraint

$$\hat{\mathbf{x}}_{t+N} \in \hat{\mathbb{X}}^F \tag{4.22}$$

Here, the cost function $\mathbf{V}^N$ is defined as

$$\mathbf{V}^N(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{[t:t+N-1]}) = \sum_{k=t}^{t+N-1} \mathbf{l}(\hat{\mathbf{x}}_k, \hat{\mathbf{u}}_k) + \mathbf{V}^F(\hat{\mathbf{x}}_{t+N})$$

We also define

$$\mathbf{V}^{N,0}(\hat{\mathbf{x}}_t) = \min_{\hat{\mathbf{u}}_{[t:t+N-1]}} \mathbf{V}^N(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{[t:t+N-1]}) \tag{4.23}$$

subject to the dynamic constraints (4.18) and the static constraints (4.21b)-(4.22).

4.7.2 Feasibility

From definition 4.3.1, it collectively holds that

$$\begin{aligned} \mathbb{X}^N = \quad & \{\mathbf{x} : \text{ if } \mathbf{x}_0 = \mathbf{x} \text{ then } \exists \tilde{\mathbf{x}}_{[0:N-1]}, \hat{\mathbf{x}}_{0/0}, \hat{\mathbf{u}}_{[0,N-1]} \\ & \text{such that (4.18), (4.21) and (4.22) are satisfied}\} \end{aligned}$$

and that, for each point of the feasibility set $\mathbf{x} \in \mathbb{X}^N$

$$\begin{aligned} \tilde{\mathbb{X}}_{\mathbf{x}} := \quad & \{\tilde{\mathbf{x}}_{[0:N-1]} : \text{ if } \mathbf{x}_0 = \mathbf{x} \text{ then } \exists \hat{\mathbf{x}}_{0/0}, \hat{\mathbf{u}}_{[0,N-1]} \\ & \text{such that (4.18), (4.21) and (4.22) are satisfied}\} \end{aligned}$$

We also define $\hat{\mathbb{X}}^N = \mathbb{X}^N \ominus \mathbb{Z}$.
Assume that, at instant t, $\mathbf{x}_t \in \mathbb{X}^N$ and that $\tilde{\mathbf{x}}_{[t:t+N-1]} \in \tilde{\mathbb{X}}_{\mathbf{x}_t}$. The optimal nominal input and state sequences obtained by minimizing the collective MPC problem are $\hat{\mathbf{u}}_{[t:t+N-1]/t} = \{\hat{\mathbf{u}}_{t/t}, \ldots, \hat{\mathbf{u}}_{t+N-1/t}\}$ and $\hat{\mathbf{x}}_{[t:t+N]/t} = \{\hat{\mathbf{x}}_{t/t}, \ldots, \hat{\mathbf{x}}_{t+N/t}\}$, respectively. Finally, recall that it is set $\tilde{\mathbf{x}}_{t+N} = \hat{\mathbf{x}}_{t+N/t}$.
We define $\hat{\mathbf{u}}^{aux}_{t+N/t} = \mathbf{K}^{aux}\hat{\mathbf{x}}_{t+N/t}$ and we compute $\hat{\mathbf{x}}^{aux}_{t+N+1/t}$ according to (4.18) from $\hat{\mathbf{x}}_{t+N/t}$ where $\hat{\mathbf{u}}_{t+N} = \hat{\mathbf{u}}^{aux}_{t+N/t}$. We obtain

$$\hat{\mathbf{x}}^{aux}_{t+N+1/t} = \mathbf{A}^*\hat{\mathbf{x}}_{t+N/t} + \mathbf{B}\hat{\mathbf{u}}^{aux}_{t+N/t} + \tilde{\mathbf{A}}\tilde{\mathbf{x}}_{t+N}$$

since $\tilde{\mathbf{x}}_{t+N} = \hat{\mathbf{x}}_{t+N/t}$, the latter is equivalent to

$$\hat{\mathbf{x}}^{aux}_{t+N+1/t} = (\mathbf{A} + \mathbf{B}\mathbf{K}^{aux})\hat{\mathbf{x}}_{t+N/t}$$

Note that, in view of constraint (4.22) and Assumption 4.4.2, $\hat{\mathbf{u}}^{aux}_{t+N/t} \in \hat{\mathbb{U}}$ and $\hat{\mathbf{x}}^{aux}_{t+N+1/t} \in \hat{\mathbb{X}}^F$. Therefore, they satisfy constraints (4.21c), (4.21d) and (4.22). Also, according to Assumption 4.4.2, (4.16) holds.
We also define the input sequence

$$\bar{\mathbf{u}}_{[t+1:t+N]/t} = \{\hat{\mathbf{u}}_{t+1/t}, \ldots, \hat{\mathbf{u}}_{t+N-1/t}, \hat{\mathbf{u}}^{aux}_{t+N/t}\}$$

and the state sequence stemming from the initial condition $\hat{\mathbf{x}}_{t+1/t}$ and the input sequence $\bar{\mathbf{u}}_{[t+1:t+N]/t}$ i.e.,

$$\bar{\mathbf{x}}_{[t+1:t+N+1]/t} = \{\hat{\mathbf{x}}_{t+1/t}, \ldots, \hat{\mathbf{x}}_{t+N/t}, \hat{\mathbf{x}}^{aux}_{t+N+1/t}\}$$

In view of the feasibility of the *i*-MPC problem at time t, we have that $\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1/t} \in \mathbb{Z}$ and $\hat{\mathbf{x}}_{k/t} - \tilde{\mathbf{x}}_k \in \prod_{i=1}^{M} \mathbb{E}$ for all $k = t+1, \ldots, t+N-1$. Note also that $\hat{\mathbf{x}}_{t+N/t} - \tilde{\mathbf{x}}_{t+N} = 0 \in \mathbb{E}$ by (4.15). Therefore, we can conclude that the state and the input sequences $\bar{\mathbf{x}}_{[t+1:t+N+1]/t}$ and $\bar{\mathbf{u}}_{[t+1:t+N]/t}$ are feasible at $t+1$, since constraints (4.21) and (4.22) are satisfied. This proves that $\mathbf{x}_t \in \mathbb{X}^N$ and $\tilde{\mathbf{x}}_{[t:t+N-1]} \in \tilde{\mathbb{X}}_{\mathbf{x}_t}$ implies that $\mathbf{x}_{t+1} \in \mathbb{X}^N$ and $\tilde{\mathbf{x}}_{[t+1:t+N]} \in \tilde{\mathbb{X}}_{\mathbf{x}_{t+1}}$.

4.7.3 Convergence of the optimal cost function

By optimality

$$\mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t}) \leq \mathbf{V}^{N}(\hat{\mathbf{x}}_{t+1/t}, \bar{\mathbf{u}}_{[t+1:t+N]/t})$$

where

$$\mathbf{V}^{N}(\hat{\mathbf{x}}_{t+1/t}, \bar{\mathbf{u}}_{[t+1:t+N]/t}) = \sum_{k=t+1}^{t+N} \mathbf{l}(\hat{\mathbf{x}}_{k/t}, \hat{\mathbf{u}}_{k/t}) + \mathbf{V}^{F}\left(\hat{\mathbf{x}}^{aux}_{t+N+1/t}\right) \tag{4.24}$$

where it is set $\hat{\mathbf{u}}_{t+N/t} = \hat{\mathbf{u}}^{aux}_{t+N/t}$. Therefore we compute that

$$\begin{aligned}\mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t}) - \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t/t}) \leq -\mathbf{l}(\hat{\mathbf{x}}_{t/t}, \hat{\mathbf{u}}_{t/t}) + \mathbf{l}(\hat{\mathbf{x}}_{t+N/t}, \hat{\mathbf{u}}^{aux}_{t+N/t}) + \\ + \mathbf{V}^{F}\left(\hat{\mathbf{x}}^{aux}_{t+N+1/t}\right) - \mathbf{V}^{F}\left(\hat{\mathbf{x}}_{t+N/t}\right)\end{aligned} \tag{4.25}$$

In view of (4.16)

$$\begin{aligned}\mathbf{V}^{F}\left(\hat{\mathbf{x}}^{aux}_{t+N+1/t}\right) \quad & -\mathbf{V}^{F}\left(\hat{\mathbf{x}}_{t+N/t}\right) + \mathbf{l}(\hat{\mathbf{x}}_{t+N/t}, \hat{\mathbf{u}}^{aux}_{t+N/t}) \leq \\ & -\kappa\mathbf{l}(\hat{\mathbf{x}}_{t+N/t}, \hat{\mathbf{u}}^{aux}_{t+N/t})\end{aligned}$$

and so, from (4.25), it follows that

$$\mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t}) \leq \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t/t}) - \mathbf{l}(\hat{\mathbf{x}}_{t/t}, \hat{\mathbf{u}}_{t/t}) - \kappa\mathbf{l}(\hat{\mathbf{x}}_{t+N/t}, \hat{\mathbf{u}}^{aux}_{t+N/t}) \tag{4.26}$$

Recall the definition of l_i and of matrix R_i, for all $i = 1, \ldots, M$, and define $\mathbf{R} = \text{diag}(R_1, \ldots, R_M)$. Then, there exists a $\mathscr{K}_\infty$ function α_L such that $\mathbf{l}(\mathbf{x}, \mathbf{u}) \geq \alpha_L(\|(\mathbf{x}, \mathbf{R}\mathbf{u})\|)$ for all $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$. This implies that $\mathbf{l}(\mathbf{x}, \mathbf{u}) \geq \alpha_L(\|\mathbf{x}\|)$ for all $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$. Therefore

$$\mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t}) \leq \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t/t}) - \alpha_L(\|\hat{\mathbf{x}}_{t/t}\|) - \kappa\alpha_L(\|\tilde{\mathbf{x}}_{t+N}\|) \tag{4.27}$$

for all $\hat{\mathbf{x}}_{t/t} \in \hat{\mathbb{X}}^N$ and for all feasible sequences $\tilde{\mathbf{x}}_k$, $k = t, \dots, t+N-1$.

Now we analyze the properties of the cost function $\mathbf{V}^{N*}(\mathbf{x}_t)$ defined in (4.20). First, note that, by definition of $\hat{\mathbf{x}}_{t/t}$, we have that $\mathbf{V}^{N*}(\mathbf{x}_t) = \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t/t})$. By optimality, we have that

$$\mathbf{V}^{N*}(\mathbf{x}_{t+1}) = \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t+1}) \leq \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t+1/t})$$

Considering (4.27), we obtain that

$$\mathbf{V}^{N*}(\mathbf{x}_{t+1}) \leq \mathbf{V}^{N*}(\mathbf{x}_t) - \alpha_L(\|\hat{\mathbf{x}}_{t/t}\|) - \kappa\alpha_L(\|\tilde{\mathbf{x}}_{t+N}\|) \tag{4.28}$$

for all $\mathbf{x}_t \in \mathbb{X}^N$ and for all sequences $\tilde{\mathbf{x}}_{[t:t+N-1]} \in \tilde{\mathbf{X}}_{\mathbf{x}_t}$. This proves that $\|\hat{\mathbf{x}}_{t/t}\| \to 0$ and $\|\tilde{\mathbf{x}}_t\| \to 0$ as $t \to +\infty$.

4.7.4 Convergence of the trajectories

Define a positive real number δ_F in such a way that, if $\|\hat{\mathbf{x}}_k\| < \delta_F$, $k = t, \dots, t+N$, and if $\|\hat{\mathbf{u}}_k\| < \delta_F$, $k = t, \dots, t+N-1$, then constraints (4.21b)-(4.22) are satisfied.
Defining a sequence $\bar{\mathbf{x}}_{k/t}$, $k = t, \dots, t+N$, stemming from the initial condition $\bar{\mathbf{x}}_{t/t} = \hat{\mathbf{x}}_{t/t}$, whose dynamics obeys to (4.18), and where the input $\hat{\mathbf{u}}_k = \bar{\mathbf{u}}_{k/t} = \mathbf{K}^{aux}\bar{\mathbf{x}}_{k/t}$, for all $k = t, \dots, t+N-1$, then there exists a positive real number $\delta_x < \delta_F$ such that, if $\|\hat{\mathbf{x}}_{t/t}\| < \delta_x$ and $\|\tilde{\mathbf{x}}_k\| < \delta_x$ for $k = t, \dots, t+N-1$, then $\|\bar{\mathbf{x}}_{k/t}\| < \delta_F$, $k = t, \dots, t+N$, and $\|\bar{\mathbf{u}}_{k/t}\| < \delta_F$, $k = t, \dots, t+N-1$. In fact, denoting $\mathbf{F} = \mathbf{A}^* + \mathbf{B}\mathbf{K}^{aux}$, we solve (4.18) and we obtain that, for $i \geq 1$

$$\bar{\mathbf{x}}_{t+i/t} = \mathbf{F}^i\hat{\mathbf{x}}_{t/t} + \sum_{j=0}^{i-1} \mathbf{F}^j\tilde{\mathbf{A}}\tilde{\mathbf{x}}_{t+i-j-1} \tag{4.29}$$

and therefore

$$\begin{array}{lcl} \|\bar{\mathbf{x}}_{t/t}\| & = & \|\hat{\mathbf{x}}_{t/t}\| < \delta_x < \delta_F \\ \|\bar{\mathbf{x}}_{t+i/t}\| & < & \max_{i=1,\dots,N} \|\mathbf{F}^i + \sum_{j=0}^{i-1}\mathbf{F}^j\tilde{\mathbf{A}}\|\delta_x \\ \|\bar{\mathbf{u}}_{k/t}\| & \leq & \|\mathbf{K}^{aux}\|\|\bar{\mathbf{x}}_{k/t}\| \end{array}$$

Therefore, for a suitable δ_x, if $\|\hat{\mathbf{x}}_{t/t}\| < \delta_x$ and $\|\tilde{\mathbf{x}}_k\| < \delta_x$, $k = t, \dots, t+N-1$, are verified at time t, then the trajectories $\bar{\mathbf{x}}_{k/t}$, $k = t, \dots, t+N$ and $\bar{\mathbf{u}}_{k/t}$, $k = t, \dots, t+N-1$ are feasible (since also $\hat{\mathbf{x}}_{t/t}$ satisfies (4.21a) for the feasibility of the i-MPC problem at time t).
Since $\|\hat{\mathbf{x}}_{t/t}\| \to 0$ and $\|\tilde{\mathbf{x}}_t\| \to 0$ as $t \to +\infty$, there exists $\bar{t} > 0$ such that $\|\hat{\mathbf{x}}_{t/t}\| < \delta_x$ and $\|\tilde{\mathbf{x}}_t\| < \delta_x$ for all $t \geq \bar{t}$, which makes the trajectories $\bar{\mathbf{x}}_{k/t}$, $k = t, \dots, t+N$, and $\bar{\mathbf{u}}_{k/t}$, $k = t, \dots, t+N-1$, feasible for all $t \geq \bar{t}$. By optimality, if $t \geq \bar{t}$

$$\mathbf{V}^{N*}(\mathbf{x}_t) = \mathbf{V}^{N,0}(\hat{\mathbf{x}}_{t/t}) \leq \sum_{k=t}^{t+N-1} \mathbf{l}(\bar{\mathbf{x}}_{k/t}, \bar{\mathbf{u}}_{k/t}) + \mathbf{V}^F\left(\bar{\mathbf{x}}_{t+N/t}\right) \tag{4.30}$$

Recall (4.16). Since $\mathbf{V}^F \geq 0$ by definition, one has that

$$\mathbf{l}(\bar{\mathbf{x}}_{k/t}, \bar{\mathbf{u}}_{k/t}) \leq \tfrac{1}{1+\kappa}\mathbf{V}^F\left(\bar{\mathbf{x}}_{k/t}\right) \leq \mathbf{V}^F\left(\bar{\mathbf{x}}_{k/t}\right)$$

since $\kappa > 0$. Therefore, from (4.30), we have that

$$\mathbf{V}^{N*}(\mathbf{x}_t) \leq \sum_{k=t}^{t+N} \mathbf{V}^F\left(\bar{\mathbf{x}}_{k/t}\right) \tag{4.31}$$

From (4.29) and (4.31), we obtain that, for all $t \geq \bar{t}$, there exists a $\mathscr{K}_\infty$ function β such that

$$\mathbf{V}^{N*}(\mathbf{x}_t) \leq \beta(\|(\hat{\mathbf{x}}_{t/t}, \tilde{\mathbf{x}}_{[t:t+N-1]})\|) \tag{4.32}$$

For this it follows that $\mathbf{V}^{N*}(\mathbf{x}_t) \to 0$ as $t \to +\infty$.

Recall that $\hat{\mathbf{x}}_{k/t}$ is generated according to (4.18), stemming from the initial condition $\hat{\mathbf{x}}_{t/t}$ and inputs $\hat{\mathbf{u}}_{k/t}$. One can write the solution to (4.18) as $\hat{\mathbf{x}}_{t+i/t} = \mathbf{v}_{t+i/t} + \mathscr{B}_i \mathbf{U}_t$, where

$$\mathbf{v}_{t+i/t} = (\mathbf{A}^*)^i \hat{\mathbf{x}}_{t/t} + \sum_{j=0}^{i-1} (\mathbf{A}^*)^j \tilde{\mathbf{A}} \tilde{\mathbf{x}}_{t+i-j-1},$$

$$\mathscr{B}_i = \begin{bmatrix} (\mathbf{A}^*)^{i-1}\mathbf{B} & \dots & \mathbf{B} & 0 & \dots & 0 \end{bmatrix}$$

if $i = 1, \dots, N$, $\mathbf{U}_t = (\hat{\mathbf{u}}_{t/t}, \dots, \hat{\mathbf{u}}_{t+N-1/t})$. Note that, since $\|\hat{\mathbf{x}}_{t/t}\| \to 0$ and $\|\tilde{\mathbf{x}}_t\| \to 0$ as $t \to +\infty$, also $\|\mathbf{v}_{k/t}\| \to 0$ as $t \to +\infty$ for all $k = t+1, \dots, t+N$. We also denote $\mathbf{v}_{t/t} = \hat{\mathbf{x}}_{t/t}$ and $\mathscr{B}_0 = 0_{n \times Nm}$.
Now, consider again the function $\mathbf{V}^{N*}(\mathbf{x}_t)$:

$$\mathbf{V}^{N*}(\mathbf{x}_t) = \sum_{k=t}^{t+N-1} \mathbf{l}(\mathbf{v}_{k/t} + \mathscr{B}_{k-t}\mathbf{U}_t, \hat{\mathbf{u}}_{k/t}) + \mathbf{V}^F\left(\mathbf{v}_{t+N/t} + \mathscr{B}_N \mathbf{U}_t\right) \tag{4.33}$$

From the definition of l_i it follows that $\mathbf{l}(\mathbf{x}_k, \mathbf{u}_k) \geq \alpha_L(\|(\mathbf{x}_k, \mathbf{R}\mathbf{u}_k)\|)$, and so

$$\begin{array}{rl} 0 \leq & \sum_{k=t}^{t+N-1} \alpha_L(\|(\mathbf{v}_{k/t} + \mathscr{B}_{k-t}\mathbf{U}_t, \mathbf{R}\hat{\mathbf{u}}_{k/t})\|) \\ & + \mathbf{V}^F\left(\mathbf{v}_{t+N/t} + \mathscr{B}_N \mathbf{U}_t\right) \leq \mathbf{V}^{N*}(\mathbf{x}_t) \end{array}$$

Since it is proved that $\mathbf{V}^{N*}(\mathbf{x}_t) \to 0$ as $t \to +\infty$, it follows that, for all $k = t, \dots, t+N-1$

$$\alpha_L(\|(\mathbf{v}_{k/t} + \mathscr{B}_{k-t}\mathbf{U}_t, \mathbf{R}\hat{\mathbf{u}}_{k/t})\|) \to 0$$

and $\mathbf{V}^F\left(\mathbf{v}_{t+N/t} + \mathscr{B}_N \mathbf{U}_t\right) \to 0$ as $t \to +\infty$. This implies that:

$$\mathbb{B}\mathbf{U}_t + \mathbf{V}_t \to 0 \tag{4.34}$$

as $t \to \infty$, where

$$\mathbb{B} = \begin{bmatrix} \mathscr{B}_0 \\ \vdots \\ \mathscr{B}_N \\ \text{diag}(\mathbf{R}, \dots, \mathbf{R}) \end{bmatrix}$$

and $\mathbf{V}_t = (v_{t/t}, \dots, v_{t+N/t}, 0, \dots, 0)$. It is readily seen that, in view of the triangular structure of

$$\begin{bmatrix} \mathscr{B}_1 \\ \vdots \\ \mathscr{B}_N \end{bmatrix}$$

and since, by definition of R_i, $i = 1, \dots, M$

$$\text{rank}\left(\begin{bmatrix} \mathbf{B} \\ \mathbf{R} \end{bmatrix}\right) = m$$

then $\text{rank}(\mathbb{B}) = Nm$. Since $\mathbf{V}_t \to 0$ as $t \to +\infty$, from (4.34) it follows that $\mathbf{U}_t \to 0$ as $t \to +\infty$. Therefore $\hat{\mathbf{u}}_{t/t} \to 0$ as $t \to +\infty$.

Finally, recall that the state $\mathbf{x}_t$ evolves according to the equation

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A} + \mathbf{B}\left[\hat{\mathbf{u}}_{t/t} + \mathbf{K}^{aux}(\mathbf{x}_t - \hat{\mathbf{x}}_{t/t})\right] \\ &= (\mathbf{A} + \mathbf{B}\mathbf{K}^{aux})\,\mathbf{x}_t + \mathbf{B}\left(\hat{\mathbf{u}}_{t/t} - \mathbf{K}^{aux}\hat{\mathbf{x}}_{t/t}\right) \end{aligned}$$

By asymptotic convergence to zero of the nominal state and input signals $\hat{\mathbf{x}}_{t/t}$ and $\hat{\mathbf{u}}_{t/t}$ respectively, we obtain that
$\mathbf{B}\left(\hat{\mathbf{u}}_{t/t} - \mathbf{K}^{aux}\hat{\mathbf{x}}_{t/t}\right)$ is an asymptotically vanishing term. Since also $(\mathbf{A} + \mathbf{B}\mathbf{K}^{aux})$ is Schur by Assumption 4.4.2, we obtain that $\mathbf{x}_t \to 0$ as $t \to +\infty$. This concludes the proof of Theorem 1.

Bibliography

[1] M. D. Doan, T. Keviczky, and B. De Schutter, "An iterative scheme for distributed model predictive control using fenchel's duality," *Journal of Process Control*, 2011.

[2] S.-P. Han and G. Lou, "A parallel algorithm for a class of convex programs," *SIAM Journal on Control and Optimization*, vol. 26, no. 2, pp. 345–355, Mar. 1988.

[3] R. T. Rockafellar, *Convex Analysis*. Princeton, NJ: Princeton University Press, 1970.

[4] H. Scheu, J. Busch, and W. Marquardt, "Nonlinear distributed dynamic optimization based on first order sensitivities," in *Proceedings of the American Control Conference 2010*, Baltimore, Maryland, USA, June 30–July 02 2010, pp. 1574–1579.

[5] H. Scheu and W. Marquardt, "Sensitivity-based coordination in distributed model predictive control," *Accepted for Journal of Process Control*, 2011.

[6] ——, "Distributed model-predictive control driven by simultaneous derivation of prices and resources," in *accepted for the proceedings of the 2011 IFAC World Confgress*, Milan, Italy, September 2011.

[7] K. H. Johansson, "The quadruple-tank process," *IEEE Transactions on Control Systems Technology*, vol. 8, pp. 456 – 465, 2000.

[8] I. Alvarado, D. Limon, D. M. de la Pena, J. Maestre, M. Ridao, H. Scheu, W. Marquardt, R. Negenborn, B. D. Schutter, F. Valencia, and J. Espinosa, "A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark," *Submitted to Journal of Process Control*, 2011.

[9] M. Farina and R. Scattolini, "Distributed non-cooperative MPC with neighbor-to-neighbor communication," in *18th IFAC World Congress*, Milan, Italy, 2011.

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

[11] D. D. Šiljac, *Large-scale dynamic systems. Stability and structure.* North Holland, 1978.

[12] N. Sandell Jr, P. Varaiya, M. Athans, and M. Safonov, "Survey of decentralized control methods for large scale systems," *IEEE Trans. on Automatic Control*, vol. 23, no. 2, pp. 108 – 128, Apr 1978.

[13] S. Dashkovskiy, B. S. Rüffer, and F. Wirth, "An ISS small gain theorem for iss general networks," *Mathematics of Control, Signals, and Systems*, vol. 19, no. 2, pp. 93–122, 2007.

[14] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, July 2003. [Online]. Available: http://www.sciencedirect.com/science/article/B6V2H-47BX35T-1/2/1e355f78abeb6d9ee76d726330e7ca54

[15] J. M. Maciejowski, *Predictive Control with Constraints*. Harlow, England: Prentice Hall, 2002.

[16] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 7, pp. 789–814, June 2000.

[17] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, WI: Nob Hill Publishing, 2009.

[18] D. Jia and B. Krogh, "Distributed model predictive control," in *American Control Conference*, vol. 4, 2001, pp. 2767–2772.

[19] E. Camponogara, D. Jia, B. Krogh, and S. Talukdar, "Distributed model predictive control," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 44–52, Feb. 2002.

[20] J. B. Rawlings and B. T. Stewart, "Coordinating multiple optimization-based controllers: New opportunities and challenges," *Journal of Process Control*, vol. 18, no. 9, pp. 839–845, Oct. 2008.

[21] R. Scattolini, "Architectures for distributed and hierarchical model predictive control - a review," *Journal of Process Control*, vol. 19, pp. 723–731, 2009.

[22] W. B. Dunbar and R. M. Murray, "Distributed receding horizon control for multi-vehicle formation stabilization," *Automatica*, vol. 42, no. 4, pp. 549–558, Apr. 2006.

[23] T. Keviczky, F. Borrelli, and G. J. Balas, "Decentralized receding horizon control for large scale dynamically decoupled systems," *Automatica*, vol. 42, no. 12, pp. 2105–2115, Dec. 2006.

[24] A. Richards and J. How, "Robust distributed model predictive control," *International Journal of Control*, vol. 80, no. 9, pp. 1517–1531, Sept. 2007.

[25] A. N. Venkat, I. Hiskens, J. B. Rawlings, and S. J. Wright, "Distributed MPC strategies with application to power system automatic generation control," *IEEE Trans. on Automatic Control*, vol. 16, no. 6, pp. 1192 – 1206, 2008.

[26] D. Jia and B. Krogh, "Min-max feedback model predictive control for distributed control with communication," in *American Control Conference*, vol. 6, 2002, pp. 4507–4512.

[27] A. Alessio and A. Bemporad, "Decentralized model predictive control of constrained linear systems," in *European Control Conference*, 2007, pp. 2813–2818.

[28] ——, "Stability conditions for decentralized model predictive control under packet drop communication," in *American Control Conference*, 2008, pp. 3577–3582.

[29] S. Li, Y. Zhang, and Q. Zhu, "Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem," *Information Sciences*, vol. 170, no. 2-4, pp. 329–349, Feb. 2005.

[30] E. Camponogara and S. Talukdar, "Distributed model predictive control: Synchronous and asynchronous computation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 37, no. 5, pp. 732–745, Sept. 2007.

[31] I. Necoara, D. Doan, and J. Suykens, "Application of the proximal center decomposition method to distributed model predictive control," in *IEEE Conference on Decision and Control*, Dec. 2008, pp. 2900–2905.

[32] I. Necoara and J. Suykens, "Application of a smoothing technique to decomposition in convex optimization," *IEEE Trans. on Automatic Control*, vol. 53, no. 11, pp. 2674–2679, Dec. 2008.

[33] M. Mercangoz and F. J. Doyle III, "Distributed model predictive control of an experimental four-tank system," *Journal of Process Control*, vol. 17, no. 3, pp. 297–308, Mar. 2007.

[34] R. R. Negenborn, P. J. van Overloop, T. Keviczky, and B. De Schutter, "Distributed model predictive control for irrigation canals," *Networks and Heterogeneous Media*, vol. 4, no. 2, pp. 359–380, June 2009.

[35] M. Arnold, R. R. Negenborn, G. Andersson, and B. De Schutter, "Distributed predictive control for energy hub coordination in coupled electricity and gas networks," in *Intelligent Infrastructures*, R. R. Negenborn, Z. Lukszo, and H. Hellendoorn, Eds. Dordrecht, The Netherlands: Springer, 2010, pp. 235–273.

[36] D. Doan, T. Keviczky, I. Necoara, M. Diehl, and B. De Schutter, "A distributed version of Han's method for DMPC using local communications only," *Control Engineering and Applied Informatics*, vol. 11, no. 3, pp. 6–15, Sept. 2009.

[37] M. D. Doan, T. Keviczky, and B. De Schutter, "An improved distributed version of Han's method for distributed MPC of canal systems," in *12th symposium on Large Scale Systems: Theory and Applications*, July 2010.

[38] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback control laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations," *Journal of Optimization Theory and Applications*, vol. 57, no. 2, pp. 265–293, May 1988.

[39] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, MA: Cambridge University Press, 2004.

[40] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[41] J. Schuurmans, A. Hof, S. Dijkstra, O. H. Bosgra, and R. Brouwer, "Simple water level controller for irrigation and drainage canals," *Journal of Irrigation and Drainage Engineering*, vol. 125, no. 4, pp. 189–195, July 1999.

[42] A. Nedic and A. Ozdaglar, "Approximate primal solutions and rate analysis for dual subgradient methods," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1757–1780, Nov. 2009.

[43] Y. Nesterov, "Primal-dual subgradient methods for convex problems," *Mathematical Programming*, vol. 120, no. 1, pp. 221–259, Aug. 2009.

[44] T. Larsson and Z. Liu, "A Lagrangean relaxation scheme for structured linear programs with application to multicommodity network flows," *Optimization*, vol. 40, pp. 247–284, 1997.

[45] H. Sherali and G. Choi, "Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs," *Operations Research Letters*, vol. 19, no. 3, pp. 105–113, Sept. 1996.

[46] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. on Automatic Control*, vol. 44, no. 3, pp. 648–654, Mar. 1999.

[47] S. Qin and T. Badgwell, "An overview of nonlinear model predictive control applications," in *Nonlinear Model Predictive Control*, F. Allgower and A. Zheng, Eds. Birkhauser Berlin, 2000, pp. 369–392.

[48] W. Al-Gherwi, H. Budman, and A. Elkamel, "An online algorithm for robust distributed model predictive control," in *Proceedings of ADCHEM 2009 (International Symposium on Advanced Control of Chemical Processes)*, vol. 7, no. 1, Istanbul, Turkey, July 2009.

[49] G. Pannocchia, S. J. Wright, B. T. Stewart, and J. B. Rawlings, "Efficient cooperative distributed MPC using partial enumeration," in *Proceedings of ADCHEM 2009 (International Symposium on Advanced Control of Chemical Processes)*, Istanbul, Turkey, July 2009.

[50] F. Borrelli, T. Keviczky, K. Fregene, and G. J. Balas, "Decentralized Receding Horizon Control of Cooperative Vehicle Formations," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, December 2005, pp. 3955–3960.

[51] G. Lionis and K. J. Kyriakopoulos, "Approximate control of formations of multiagent systems," in *Proceedings of the 44th Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, December 2005, pp. 4958–4963.

[52] P. Massioni and M. Verhaegen, "Distributed Control of Vehicle Formations: a Decomposition Approach," in *Proceedings of the 47th IEEE Conference on Decision and Control 2008*, Cancun, Mexico, December 2008, pp. 2906–2912.

[53] M. D. Mesarovic, D. Macko, and Y. Takahara, *Theory of Hierarchical, Multilevel, Systems*, R. Bellman, Ed. Academic Press, 1970.

[54] A. M. Geoffrion, "Primal resource-directive approach for optimizing nonlinear decomposable systems," *Operations Research*, vol. 18, no. 3, pp. 375–403, May–June 1970.

[55] G. J. Silverman, "Primal decomposition of mathematical programs by resource allocation: I – basic theory and a direction-finding procedure," *Operations Research*, vol. 20, no. 1, pp. 58–74, January–February 1972.

[56] L. S. Lasdon, *Optimization Theory for Large Systems*. Macmillan Series for Operations Research, 1970.

[57] M. G. Singh, S. A. W. Drew, and J. F. Coales, "Comparisons of practical hierarchical control methods for interconnected dynamical systems," *Automatica*, vol. 11, pp. 331–350, 1975.

[58] M. S. Mahmoud, "Multilevel systems control and applications: A survey," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-7, no. 3, pp. 125–143, March 1977.

[59] Y. Wakasa, M. Arakawa, K. Tanaka, and T. Akashi, "Decentralized model predictive control via dual decomposition," in *Proceedings of the 47th IEEE Conference on Decision and Control 2008*, Cancun, Mexico, December 2008, pp. 381–386.

[60] I. Necoara and J. A. Suykens, "A proximal center-based decomposition method for multi-agent convex optimization," in *Proceedings of the 47th IEEE Conference on Decision and Control 2008*, Cancun, Mexico, December 2008, pp. 3077–3082.

[61] A. N. Venkat, J. B. Rawlings, and S. J. Wright., "Plant-wide optimal control with decentralized MPC," in *Proceedings of DYCOPS*, Boston, MA, USA, July 2004.

[62] N. I. Marcos, J. F. Forbes, and M. Guay, "Coordination of distributed model predictive controllers for constrained dynamic processes," in *Proceedings of ADCHEM (International Symposium on Advanced Control of Chemical Processes)*, Istanbul, Turkey, July 2009.

[63] S. Talukdar, D. Jia, P. Hines, and B. H. Krogh, "Distributed model predictive control for the mitigation of cascading failure," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Sevilla, Spain, December 2005, pp. 4440–4445.

[64] J. Liu, X. Chen, D. Muñoz de la Peña, and P. D. Christofides, "Sequential and iterative architectures for distributed model predictive control of nonlinear process systems," *AIChE Journal*, vol. 56, no. 8, pp. 2137–2149, 2010.

[65] W. Dunbar, "Distributed receding horizon control of dynamically coupled nonlinear systems," *IEEE Trans. on Automatic Control*, vol. 52, no. 7, pp. 1249–1263, July 2007.

[66] L. Magni and R. Scattolini, "Stabilizing decentralized model predictive control of nonlinear systems," *Automatica*, vol. 42, pp. 1231–1236, 2006.

[67] H. K. Khalil, *Nonlinear systems - Third edition*. Prentice Hall, 2000.

[68] R. Negenborn, A. Beccuti, T. Demiray, S. Leirens, G. Damm, B. De Schutter, and M. Morari, "Supervisory hybrid model predictive control for voltage stability of power networks," in *Proc. of the IEEE American Control Conference*, New York City, USA, 2008, pp. 5444–5449.

[69] M. Arnold, R. Negenborn, G. Andersson, and B. De Schutter, "Multi-area predictive control for combined electricity and natural gas systems," *Proceedings of the European Control Conference*, pp. 1408–1413, 2009.

[70] L. Baskar, B. De Schutter, and H. Hellendoorn, "Hierarchical traffic control and management with intelligent vehicles," *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 834–839, 2007.

[71] ——, "Model-based predictive traffic control for intelligent vehicles: dynamic speed limits and dynamic lane allocation," *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 174–179, 2007.

[72] R. Negenborn, A. Sahin, Z. Lukszo, B. De Schutter, and M. Morari, "A non-iterative cascaded predictive control approach for control of irrigation canals," *Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics, San Antonio, Texas*, pp. 3652–3657, 2009.

[73] R. Negenborn, P.-J. van Overloop, and B. De Schutter, "Coordinated distributed model predictive reach control of irrigation canals," *Proceedings of the European Control Conference*, pp. 1420–1425, 2009.

[74] L. Bakule, "Decentralized control: an overview," *Annual Reviews in Control*, vol. 32, pp. 87–98, 2008.

[75] D. Šiljac and A. Zecevic, "Control of large scale systems: beyond decentralized feedback," *Annual Reviews in Control*, vol. 29, pp. 169–179, 2005.

[76] D. M. Raimondo, L. Magni, and R. Scattolini, "Decentralized MPC of nonlinear systems: an input-to-state stability approach," *International Journal of Robust and Nonlinear control*, vol. 17, pp. 1651 – 1667, 2006.

[77] D. Barcelli and A. Bemporad, "Decentralized model predictive control of dynamically-coupled linear systems: Tracking under packet loss," *Proc. 1st IFAC Workshop on Estimation and Control of Networked Systems, Venice, Italy*, 2009.

[78] J. Liu, D. Muñoz de la Peña, and P. Christofides, "Distributed model predictive control of nonlinear process systems," *AIChE J.*, vol. 55, no. 9, pp. 1171–1184, 2009.

[79] J. Liu, X. Chen, D. Muñoz de la Peña, and P. Christofides, "Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. part i: theory." *Proceedings of American Control Conference*, pp. 3148 – 3155, 2010.

[80] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, "Cooperative distributed model predictive control," *System & Control Letters*, vol. 59, pp. 460–469, 2010.

[81] B. T. Stewart, J. B. Rawlings, and S. J. Wright, "Hierarchical cooperative distributed model predictive control," *In Proc. of the American Control Conference, Baltimore, Maryland, June 2010*, 2010.

[82] W. Dunbar and R. Murray, "Distributed receding horizon control for multi-vehicle formation stabilization," *Automatica*, vol. 42, pp. 549–558, 2006.

[83] G. Ferrari-Trecate, L. Galbusera, M. Marciandi, and R. Scattolini, "Model predictive control schemes for consensus in multi-agent systems with single- and double-integrator dynamics," *IEEE Trans. on Automatic Control*, vol. 54, no. 11, pp. 2560 – 2572, 2009.

[84] P. Trodden and A. Richards, "Distributed model predictive control of linear systems with persistent disturbance," *International Journal of Control*, vol. 83, no. 8, pp. 1653 – 1663, 2010.

[85] S. V. Raković, E. C. Kerrigan, K. I. Kouramas, and D. Mayne, "Invariant approximations of the minimal robust positively invariant set," *IEEE Trans. Aut. Cont*, vol. 50, no. 3, pp. 406 – 410, 2005.