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:	D3.1.3
Deliverable Type:	Report
Contractual Date of Delivery:	September 1, 2010
Actual Date of Delivery:	**August 27, 2010**
Title of Deliverable:	**Report on new methods for complex control problems (nonlinear, dynamic, constrained)**
Dissemination level:	Public
Workpackage contributing to the Deliverable:	WP3
WP Leader:	Wolfgang Marquardt
Partners:	RWTH, TUD, POLIMI, USE, UNC, SUPELEC, UWM
Authors:	H. Scheu, W. Marquardt, M.D. Doan, T. Keviczky, B. De Schutter, P.-D. Moroşan, R. Bourdais, D. Dumur, J. Buisson, D. Limon, J.M. Maestre, D. Muñoz de la Peña, E.F. Camacho, J. Espinosa, A. Marquez, J. Garcia, F. Valencia

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

We present different approaches for distributed model predictive control developed in the HD-MPC project. On the one hand we present a completely new approach for distributed nonlinear MPC, which is driven by a new coordination mechanism. The embedded method, called gradient-based distributed dynamic optimization (GBDDO), is based on linearized information of the overall system to coordinate the overall system.
Then we present a DMPC scheme based on Han's method. The presented distributed version of Han's method is an improved version, and it is successfully applied to in the control of a canal system, described by a linear system. The third method presented is based on a cooperative game. There, from a game theoretic point of view, at each time the agents are playing a cooperative game. The description is limited to a decomposition of the problem into two subproblems, and the method is applied to the well-known four tanks example. The fourth method is an extension to an existing cooperative DMPC scheme in order to track time-varying plant-wide target output signals. The method is also applied to the nonlinear four tanks system.
In order to cope with economical objective functions, we present a distributed MPC method based on Benders' decomposition. The method is successfully applied to the multi-source temperature control in buildings. Finally we present an infinite horizon model predictive control with targets and zone control. Hence, in contrast to the other methods presented, it is a hierarchical control approach and combines an economic stage (real time optimization) with infinite horizon MPC.

Chapter 1

Overview

In the beginning of the project, we have reviewed the existing methods for hierarchical and distributed model predictive control, as reported in [51, 55, 56]. There we have seen the lack of methods for nonlinear hierarchical and distributed MPC as well as robust methods for HD-MPC, which is the basis for our ongoing research. First results related to robust MPC are reported in [57]. Thus, in this report we will not focus on robust methods. In addition, we report the development of new coordination mechanisms [41], which are used for hierarchical and distributed model predictive control, but which are not included in this report in order to avoid a duplication.

Chapter 2 presents a new method for distributed MPC, in particular nonlinear DMPC. The gradient-based distributed dynamic optimization features a new coordination mechanism. The different controllers are coordinated by means of an inclusion of linear information of the overall process and overall optimality. A strength of this new method is its fast convergence. The method is applied to a four tanks system and compared to centralized, decentralized and dual-decomposition-based methods.

Chapter 3 an improved version of a distributed version of Han's method is presented, that can be used for distributed model predictive control (DMPC) of dynamically coupled linear systems, under coupling constraints. Some DMPC problems of water networks can be cast into this type. The method is applied to a canal system. The simulation results show that the modifications lead to faster convergence of the method.

Chapter 4 reviews the distributed MPC based on a cooperative game. There, from a game theoretic point of view, at each time the agents are playing a cooperative game. Each of the agent proposes values for the decision variables. In the last step, the decision variables are chosen by consensus. Because the controller chooses among a small finite number of different modes of operation, the resulting input trajectories are not smooth. This control scheme is specially designed for only two controllers, because the number of possible modes increases in a combinatorial way with the number of controllers.

In Chapter 5 an existing DMPC method [46] is extended, such that it is capable to track time-dependent output variables. The way this controller handle the tracking problem is characterized by considering an artificial steady state and input as decision variables, penalizing the deviation of the predicted trajectory with the artificial steady conditions, adding a quadratic offset-cost function to penalize the deviation between the artificial and the target equilibrium point, and considering an extended terminal constraint.

In contrast, the distributed MPC presented in Chapter 6 does not consider a tracking or regulation problem, but a linear cost function. Hence, economic objectives can be defined. The proposed method is based on Benders' decomposition. The method can handle both local and global linear constraints but it is particular effective when the number of local constraints is significantly greater than the number of global constraints. The method is successfully applied to the multi-source temperature control in buildings.

Finally, a hierarchical MPC approach is given in Chapter 7. In that scheme, input targets are calculated on an upper control layer based on an economic objective function, while on the lower layer an infinite horizon MPC is implemented. The scheme features a zone control, i.e. the outputs of the system are kept within a predefined zone.

Chapter 2

Nonlinear distributed dynamic optimization based on first order sensitivities

The results of this chapter have been developed by Holger Scheu, Jan Busch and Wolfgang Marquardt (AVT - Process Systems Engineering, RWTH Aachen University). It is a short version of the paper [54], presented at ACC2010. Furthermore, an extensive analysis of the method has been derived and submitted to Journal of Process Control.

Abstract

A method for the distributed optimization of dynamic nonlinear systems is presented. The method is based on partial goal-interaction operators [35]. Partial goal-interaction operators provide gradient information of non-local objective functions. Hence, these operators are used to modify the objective functions of infimal optimization problems in order to take non-local information into account and to achieve an optimum for the overall objective, i.e. the objective of the entire process. However, that optimum is achieved by a decentralized but cooperative optimization, while communication between the different infimal optimization units is limited. An important part of the method is the decentralized calculation of sensitivities. The method is applied to a nonlinear differential-algebraic simple-toy system and compared to the dual-optimization method [25] as well as to the solutions of fully centralized and fully decentralized optimizations.

2.1 Introduction

Decentralized control and optimization methods are rapidly gaining interest in today's research. This development is driven by many reasons:

- Better computational performance is anticipated [1];
- communication in distributed systems may be limited;
- reliability and maintainability could be increased compared to a centralized solution [43];
- and completely new applications are considered.

These new applications are positioned mainly in the field of autonomous vehicles [7, 29, 33], such as air crafts or satellites. In process system engineering, optimization based control methods such as linear and nonlinear model predictive control or dynamic real-time optimization are the methods of choice. These systems differ to the above-mentioned multi-agent systems mainly in the fact, that different subsystems usually interact with each other. Though, normally today these subsystems are controlled independently by decentralized methods, which do not take into account the interaction of the subsystems. Performance, optimality, reliability and maintainability are key drivers for research in distributed cooperative and coordinated control methods.

Already in the beginning of the 1970s, Mesarovic et al. [35] presented a fundamental monograph on distributed and hierarchical systems. They formulated some very general principles on how a coordinated control structure can be implemented, namely the 'Interaction Balance Principle' and the 'Interaction Prediction Principle' [36]. In 1970, Lasdon published another important monograph [26]: The dual optimization method is presented, which has gained a lot of interest in research of hierarchical and distributed optimization.

There exist already various survey papers on the topic of hierarchical and distributed control literature. In 1973, Smith et al. [61] published an introductory overview on the topic of hierarchical systems theory. In 1975, Singh et al. presented a review on practical hierarchical control methods for interconnected systems. In 1977, Mahmoud presented a very comprehensive overview [32]. While the main subject of that article covers multilevel optimization techniques, it also covers the early progress in multilevel systems identification as well as the application to water resource systems. In 1978, Sandell et al. presented another survey [50], which covered the topics of model simplifications, stability analysis of interconnected systems and decentralized control methods. In a recent article Rawlings and Stewart [47] summarize the present status of research in the field of coordinated optimization-based control, as well as opportunities and challenges for future research. A lot of different control topologies can be considered in hierarchical and distributed control, which are summarized by Scattolini in a recent review paper [52].

The main research focus in hierarchical and distributed model predictive control is attached to systems of linear ordinary differential equations (ODEs). Wakasa et al. [64] apply the dual decomposition method of Lasdon to linear time-invariant systems, where the dual problem is solved using a subgradient optimization algorithm. Necoara et al. [40] propose a dual decomposition based method, called the proximal center method, which can be applied for the optimization of linear time-invariant systems with convex objective functions. Scherer et al. [53] implemented a distributed optimization method for discrete-time LTI systems for the regulation of a distillation column.
In the following we will focus on distributed dynamic optimization methods for nonlinear systems, which is a basis for nonlinear DMPC and nonlinear dynamic-real time optimization. On the one hand the dual optimization method is shortly reviewed and on the other hand a new method is presented, which is based on partial goal-interaction operators [35]. This gradient-based distributed dynamic optimization (GBDDO) method belongs to the class of goal-coordination methods. The infimal objective functions are modified using information of the whole process, in order to achieve optimality for the overall process. The remainder of this chapter is organized as follows: Section 2.2 states the nonlinear distributed dynamic optimization problem considered. Section 2.3 presents both optimization methods considered. In section 2.4 a case study is performed. The methods are also compared to a completely decentralized as well as a fully centralized solution. Finally the content is summarized in Section 2.5.

2.2 Problem Formulation

We consider N subsystems, which are described by the nonlinear differential-algebraic equations

$$M_i \dot{x}_i = f_i(t, x_i, z_i, u_i, m_i), \quad x_i(0) = x_{i,0} \tag{2.1a}$$

$$0 = g_i(t, x_i, z_i, u_i, m_i), \tag{2.1b}$$

of differential index 1, for $i = 1, \dots, N$. t is the time, M_i is the constant mass matrix, x_i and $x_{i,0}$ are the differential state vector and its initial condition, z_i is the algebraic state vector, u_i is the local input vector and m_i are the interaction variables of the subsystem i. The interaction variables are those variables, which depend on other subsystems, i.e.

$$m = \begin{bmatrix} m_1 & \dots & m_N \end{bmatrix}^T = H\,[x, z, u]^T, \tag{2.2}$$

with a constant matrix H. The interaction variables depend only on the variables of other subsystems, i.e. m_i is not a function of x_i, y_i or u_i, $i = 1, \dots, N$. There may be some constraints for the state vectors x_i and z_i as well the input vector u_i, i.e.

$$lb_i \leq \begin{bmatrix} x_i \\ z_i \\ u_i \end{bmatrix} \leq ub_i, \quad i = 1, \dots, N, \tag{2.3}$$

where lb_i and ub_i are the lower and upper bounds respectively. Finally the infimal dynamic optimization problems can be formulated as

$$\min_{u_i} \Phi_i(t, x_i, z_i, u_i, m_i), \quad i = 1, \dots, N \tag{2.4}$$

with respect to equations (2.1) - (2.3). As optimality for the overall process shall be achieved in cooperative optimization methods, the overall optimization problem

$$\min_{u} \Phi = \sum_{i=1}^{N} \Phi_i(t, x_i, z_i, u_i, m_i), \tag{2.5a}$$

$$\text{s.t. } M_i \dot{x}_i = f_i(t, x_i, z_i, u_i, m_i), \quad x_i(0) = x_{i,0}, \tag{2.5b}$$

$$0 = g_i(t, x_i, z_i, u_i, m_i), \tag{2.5c}$$

$$lb_i \leq \begin{bmatrix} x_i \\ z_i \\ u_i \end{bmatrix} \leq ub_i, \tag{2.5d}$$

$$m_i = H_i\,[x, z, u]^T, \quad i = 1, \dots, N, \tag{2.5e}$$

has to be solved. It is important to note, that the objective function has to be separable, i.e. additive as formulated in (2.5a). As that dynamic optimization problem is not to be solved by centralized methods, we are looking for methods to solve this dynamic optimization problem by decomposing it into infimal subproblems. Thereby solving the infimal optimization problems (2.4) does in general not lead to the optimum of the overall problem (2.5).

For this purpose, the infimal optimization problems (2.4) can be adapted in different ways: Two different methods are stated in the following section.

2.3 Optimization methods for distributed systems

Mesarovic et al. [35] state that coordinated optimization can be achieved in two different ways: On the one hand the model can be adapted; on the other hand the goal, i.e. the objective junctions Φ_i, can be adapted. Here, we will focus upon the latter methods, which are referred to as goal coordination [35] methods. Within goal coordination methods the infimal objective functions Φ_i of the infimal optimization problems (2.4) are modified, i.e.

$$\tilde{\Phi}_i = \tilde{\Phi}_i(\Phi_i, \xi_i), \tag{2.6}$$

where ξ_i is an additional input variable to the adapted infimal objective function $\tilde{\Phi}_i$.

In the following, the main idea of the dual optimization method [25, 26] will be reviewed, which has gained a lot of attention. Subsequently, a new method is presented, however that method is based on the rather old notion of 'partial goal-interaction operators' [35].

Both methods are iterative methods as described in the following: In a first step, the infimal optimization problems are solved. Then, part of the local information is spread to other optimization units: either directly or via a coordinator. Using the new information, the optimization in the previous step is restarted until some convergence criterion is achieved. Hence, a series of infimal optimization problems have to be solved in order to get an optimum for the overall problem.

2.3.1 Dual optimization

We briefly review Lasdon's dual optimization method, for a full description of the method we refer to [25, 26].

We consider the dual problem of problem (2.5), which is

$$\max_{\lambda} \varphi(\lambda) \tag{2.7a}$$

$$\text{w.r.t. equations (2.1) and (2.3)} \tag{2.7b}$$

where

$$\varphi(\lambda) = \min_{u,m} \Phi^{\text{dual}}(t, x, z, u, m, \lambda), \tag{2.7c}$$

with

$$\begin{aligned}\Phi^{\text{dual}} &= \Phi(t, x, z, u, m) + \lambda^T \cdot h(x, z, u, m)\\ &= \sum_{i=1}^{N} \Phi_i(t, x_i, z_i, u_i, m_i) + \lambda^T \cdot \sum_{i=1}^{N} h_i(x_i, z_i, u_i, m_i).\end{aligned} \tag{2.7d}$$

Φ^{dual} is the Lagrangian function and the vector λ contains the Lagrange multipliers. Moreover, not only the local inputs u_i but also the interaction variables m_i are degrees of freedom. The coupling constraints (2.2) are reformulated as

$$0 = h(x, z, u, m) = m - H\,[x, z, u]^T \tag{2.8}$$

with

$$h(x, z, u, m) = \sum_{i=1}^{N} h_i(x_i, z_i, u_i, m_i) \quad . \tag{2.9}$$

Now it is possible to decompose the overall optimization problem into the N local (infimal) subproblems

$$\min_{u_i,m_i} \Phi_i^{\text{dual}} = \Phi_i(t,x_i,z_i,u_i,m_i) + \lambda^T h_i(x_i,z_i,u_i,m_i) \tag{2.10a}$$

$$\text{s.t. equations (2.1) and (2.3),} \tag{2.10b}$$

which are called primal problems.

The remaining unknowns are the Lagrange multipliers λ. Based on an initial guess λ^0, the dual problem (2.7) is solved, to obtain these multipliers. For this purpose, at first, the primal optimization problems (2.10) have to be solved. The Lagrange multipliers can then be easily updated [60], as the gradient of φ can be directly calculated from equation (2.7d), i.e.

$$\frac{\partial\phi(\lambda)}{\partial\lambda} = \sum_{i=1}^{N} h_i(x_i,z_i,u_i,m_i) = h \quad . \tag{2.11}$$

Thereby h is the error in the interaction constraints. The Lagrange multipliers λ can then be improved using a steepest ascent method [60], i.e.

$$\lambda^{\iota+1} = \lambda^{\iota} + \gamma^{\iota} h^{\iota}, \tag{2.12}$$

where ι is the iteration index, and γ^l is the step length at iteration l.

The Lagrange multipliers can be interpreted as prices, which are adjusted by the coordinator, such that all coupling constraints are fulfilled and an optimum of the overall problem is achieved. Furthermore the dual optimization method can be interpreted as in implementation of the Interaction Balance Principle [35].

2.3.2 Gradient-based distributed dynamic optimization

Mesarovic et al. [35] have proposed another modification of infimal objective functions, namely the so called 'interaction operators'. In the following we shortly review the 'partial goal-interaction operator', and then we discuss how it can be applied in distributed dynamic optimization.

The ij-th 'partial goal-interaction operator' describes the effect of the i-th local input u_i on the overall objective Φ via the j-th interaction input m_j. The operator $\Gamma_{ij}(\tilde{u})$ is defined at a given point $\tilde{u}$ as a mapping $\Gamma_{ij}(\tilde{u}) : U_i \to V$ [35]. Thereby U_i is the domain of the control input u_i of subsystem i and V is the domain of the objective function Φ, i.e., $u_i \in U_i$ and $\Phi \in V$.

Furthermore, Mesarovic et al. [35] define the 'linearized partial goal-interaction operator'. These linearized partial goal-interaction operators can be applied to modify infimal objective functions Φ_i, in order to consider overall process information in infimal optimizations. The modification leads to the new infimal objective functions

$$\Phi_i^{\text{GBDDO}} = \Phi_i + \left[\sum_{\substack{j=1 \\ j\neq i}}^{N} \left. \frac{\mathrm{d}\Phi_j}{\mathrm{d}u_i} \right|_{\tilde{u}} \right] (u_i - \tilde{u}_i). \tag{2.13}$$

Another interpretation of this new objective function is the following: The overall objective function Φ is assumed to be additive as in equation (2.5a). For the infimal optimizations the overall objective is considered, though all nonlocal summands are simplified to linear terms. A key step for the application of the method, is to calculate the sensitivities $\frac{\mathrm{d}\Phi_j}{\mathrm{d}u_i}$, which will be explained in the following part of this section. These sensitivities will in the following be referred to as interaction sensitivities.

First we notice two different facts:

1. The interaction sensitivities $\frac{\mathrm{d}\Phi_j}{\mathrm{d}u_i}$ contain information of two different subsystems. Hence, one important issue is how to calculate these sensitivities in a distributed method.
2. The input vector u_i is a function of time, e.g. an infinite-dimensional vector. In order to solve the problem, these functions have to be described by discrete parameters.

We first deal with the second fact: It is a common approach to explicitly discretize the input variables u_i, e.g. by some B-splines representation [12, 58], i.e.

$$u_{i,k} = \sum_{l=1}^{n_{u_{i,k}}} \hat{u}_{i,k,l} \cdot \phi_l^{(\mu)}(t), \tag{2.14}$$

where i is the index of the subsystem, k is the index of the input signal and l is the index of the base function $\phi_l^{(\mu)}$ and the corresponding coefficient $\hat{u}_{i,k,l}$. μ is the order of the B-splines.

Now we come back to the first fact: In order to calculate the interaction sensitivity, it is written as a product of local sensitivities as follows:

$$\frac{\mathrm{d}\Phi_j}{\mathrm{d}\hat{u}_{i,k,l}} = \frac{\mathrm{d}\Phi_j}{\mathrm{d}m_j} \cdot \frac{\mathrm{d}m_j}{\mathrm{d}\hat{u}_{i,k,l}} \quad . \tag{2.15}$$

It can again be noticed, that the interaction variables m_j are functions of time. Thus, in order to calculate the first factor $\frac{\mathrm{d}\Phi_j}{\mathrm{d}m_j}$, which we call local input sensitivity, we again apply a discretization by B-splines:

$$m_{j,o} = \sum_{p=1}^{n_{m_{j,o}}} \hat{m}_{j,o,p} \cdot \phi_p^{(\mu)}(t) \tag{2.16}$$

Here, j is the index of the subsystem, o is the index of the signal and p is the index of the base function $\phi_p^{(\mu)}$ and the corresponding coefficient $\hat{m}_{j,o,p}$. Then, we get

$$\frac{\mathrm{d}\Phi_j}{\mathrm{d}\hat{u}_{i,k,l}} = \sum_{o=1}^{n_{m_j}} \sum_{p=1}^{n_{m_{j,o}}} \left[\frac{\mathrm{d}\Phi_j}{\mathrm{d}\hat{m}_{j,o,p}} \cdot \frac{\mathrm{d}\hat{m}_{j,o,p}}{\mathrm{d}\hat{u}_{i,k,l}} \right] \quad . \tag{2.17}$$

n_{m_j} is the dimension of the interaction variable m_j while $n_{m_{j,o}}$ is the number of parameters for the discretization of $m_{j,o}$. The term $\frac{\mathrm{d}\hat{m}_{j,o,p}}{\mathrm{d}\hat{u}_{i,k,l}}$ can be written as

$$\frac{\mathrm{d}\hat{m}_{j,o,p}}{\mathrm{d}\hat{u}_{i,k,l}} = \frac{\partial \hat{m}_{j,o,p}}{\partial y_{i,\diamond,l}}^T \frac{\mathrm{d}y_{i,\diamond,l}}{\mathrm{d}\hat{u}_{i,k,l}} + \frac{\partial \hat{m}_{j,o,p}}{\partial \hat{u}_{i,k,l}}, \tag{2.18}$$

where $\frac{\partial \hat{m}_{j,o,p}}{\partial y_i}$ and $\frac{\partial \hat{m}_{j,o,p}}{\partial \hat{u}_{i,k,l}}$ can be derived directly from the coupling constraints (2.2). $y_{i,\diamond,l}$ summarizes the parameters of the state variables x_i and z_i for the corresponding base function $\phi_p^{(\mu)}$. The derivatives $\frac{\mathrm{d}y_i}{\mathrm{d}\hat{u}_{i,k,l}}$ will be referred to as local output sensitivities. Finally, the required gradient information can be written as follows:

$$\frac{\mathrm{d}\Phi_j}{\mathrm{d}\hat{u}_{i,k,l}} = \sum_{o=1}^{n_{m_j}} \sum_{p=1}^{n_{m_{j,o}}} \left[\frac{\mathrm{d}\Phi_j}{\mathrm{d}\hat{m}_{j,o,p}} \left(\frac{\partial \hat{m}_{j,o,p}}{\partial y_{i,\diamond,l}}^T \frac{\mathrm{d}y_{i,\diamond,l}}{\mathrm{d}\hat{u}_{i,k,l}} + \frac{\partial \hat{m}_{j,o,p}}{\partial \hat{u}_{i,k,l}} \right) \right] \quad . \tag{2.19}$$

These gradients have to be calculated once, after the infimal optimizations are finished. Then the infimal objective functions (2.13) are updated using the new gradient information and the optimization is started again.

2.4 Optimization example

The methods discussed in the preceeding section have been implemented for a simple nonlinear, differential-algebraic system, namely a 4-tank system, which is depicted in Fig. 2.1.

The mathematical model describes the heights x_q as well as the flow rates z_q of the four tanks ($q = 1,\dots,4$) using mass balances and Toricelli's law. The heights x_q are differential variables while the flow rates z_q are algebraic variables. The system is decomposed into two subsystems: tanks 1 and 2 on the one hand and tank 3 and 4 on the other hand. This decomposition results in the two coupled nonlinear DAE subsystems

$$
\begin{aligned}
\dot{x}_1 &= A_1^{-1}(u_1 - z_1)\,, && (2.20a)\\
\dot{x}_2 &= A_2^{-1}(\alpha z_1 + (1-\beta)m_1 - z_2)\,, && (2.20b)\\
x_1(0) &= x_{1,0}, \quad x_2(0) = x_{2,0}, && (2.20c)\\
0 &= z_1 - a_1\sqrt{2gx_1}, && (2.20d)\\
0 &= z_2 - a_2\sqrt{2gx_2} && (2.20e)
\end{aligned}
$$

and

$$
\begin{aligned}
\dot{x}_3 &= A_3^{-1}(u_2 - z_3)\,, && (2.21a)\\
\dot{x}_4 &= A_4^{-1}(\beta z_3 + (1-\alpha)m_2 - z_4)\,, && (2.21b)\\
x_3(0) &= x_{3,0}, \quad x_4(0) = x_{4,0}, && (2.21c)\\
0 &= z_3 - a_3\sqrt{2gx_3}, && (2.21d)\\
0 &= z_4 - a_4\sqrt{2gx_4} && (2.21e)
\end{aligned}
$$

with controlled input flow rates u_i and coupling constraints

$$
h = \begin{bmatrix} m_1 - z_3 \\ m_2 - z_1 \end{bmatrix} = \begin{bmatrix} m_1 \\ -z_1 \end{bmatrix} + \begin{bmatrix} -z_3 \\ m_2 \end{bmatrix}. \qquad (2.22)
$$

Additionally there are the following input and path constraints:

$$
\begin{aligned}
0 &\le u \le 0.6\cdot 10^{-3}, && (2.23a)\\
0 &\le z, && (2.23b)\\
0 &\le x \le 0.3 \quad . && (2.23c)
\end{aligned}
$$

The overall objective function Φ is defined as

$$
\Phi = \underbrace{\int_0^{t_f} (x_2 - x_2^{\text{des}})^2\,dt}_{=\Phi_1} + \underbrace{\int_0^{t_f} (x_4 - x_4^{\text{des}})^2\,dt}_{=\Phi_2}. \qquad (2.24)
$$

There, x_2^{des} and x_4^{des} are setpoints for the lower tanks of the system.

For the solution of the dynamic optimization problems we applied a full discretization and solved the problems simultaneously using the Matlab nonlinear constrained optimization solver `fmincon`. The optimization horizon has been set to 25 seconds with a sampling time of 1 second.

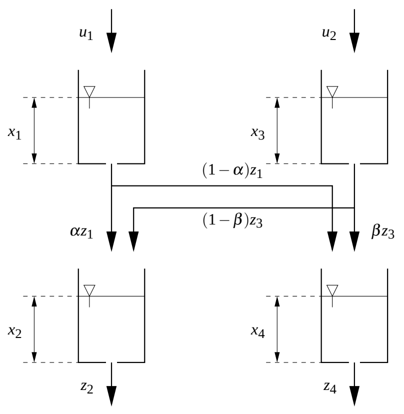


Figure 2.1: Considered 4 tank system

For comparison, on the one hand a centralized optimization of the problem has been implemented, which serves as the reference for all methods. On the other hand, a completely decentralized optimization has been realized, which neglects all interaction between subsystem 1 and subsystem 2, i.e. the interaction variables m_1 and m_2 are assumed to be zero.

Results of the offline optimization studies are exemplarily displayed in Figs. 2.2-2.4, which contain the trajectories of the heights x_2, x_4. The coupling between the subsystems has been varied with $[\alpha,\beta] = [1,1]$ (completely decoupled), $[\alpha,\beta] = [0.9,0.8]$ (weakly coupled), and $[\alpha,\beta] = [0.7,0.6]$ (strongly coupled). As one might expect, the decentralized optimization method, which neglects all interaction within the system, does not lead to reasonable results, except for the case of the decoupled subsystems. The corresponding trajectories differ substantially from those of the optimal 'reference' solution.

Although the system seems quite simple and nonlinearities appear to be quite weak, the dual-optimization method did not converge in these studies. Figs. 2.3 and 2.4 contain the optimization results for 50 iterations and a constant step size γ. Similar results have been achieved for different step sizes γ and for far more iterations. In all cases, the dual-optimization method was not able to adjust the Lagrange multipliers, such that all coupling constraints (2.22) could be fulfilled.

Finally, the gradient based distributed dynamic optimization (GBDDO) method solves the optimization problem of the 4 tank system very well: The trajectories of the GBDDO fit those of the 'reference' solution. Thereby convergence is quite fast: The figures contain the trajectories after only four iterations of the method.

2.5 Conclusions and future works

A distributed optimization method, the GBDDO method, which is based on the notion of partial goal-interaction operators, has been presented. This method uses gradient information of the overall objective function in order to modify the infimal objective functions for an efficient optimization of distributed dynamic systems.

A key part of the method is the decentralized computation of the interaction sensitivities, i.e. those sensitivities, which describe the dependence of an infimal objective function on a nonlocal input variable. These sensitivities are derived by a product of local sensitivities: in particular the local input sensitivities and the local output sensitivities. However, there are some drawbacks for GBDDO: If a continuous-time system is considered, gradient information of the interaction variables m_i has to

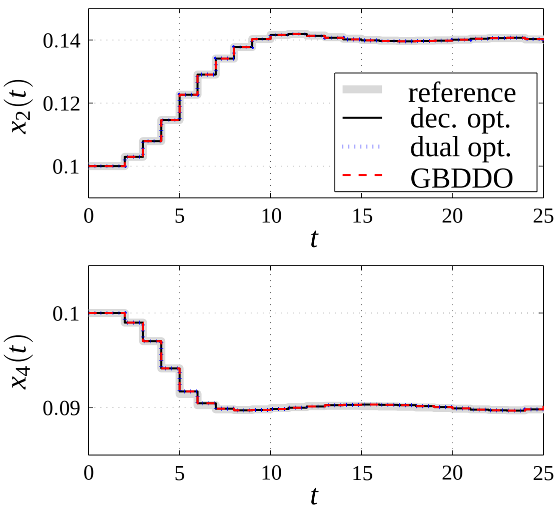


Figure 2.2: Trajectories of the heights x_2 and x_4 for the 4 different implemented methods for $[\alpha,\beta] = [1,1]$: centralized optimization (reference), decentralized optimization (dec. opt.), dual optimization (dual opt.), and gradient-based distributed optimization (GBDDO); 1 iteration

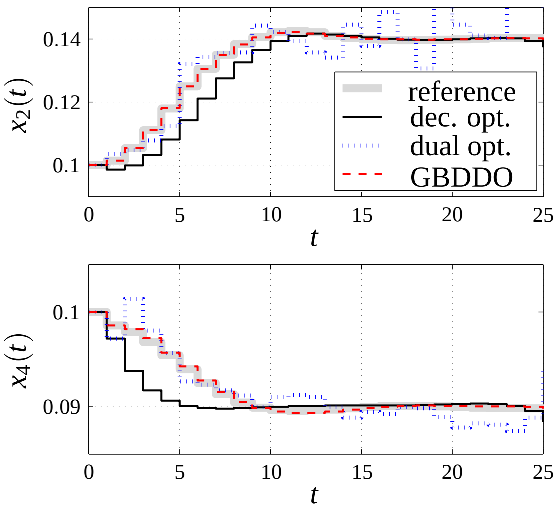


Figure 2.3: Trajectories of the heights x_2 and x_4 for the 4 different implemented methods for $[\alpha,\beta] = [0.9,0.8]$: centralized optimization (reference), decentralized optimization (dec. opt.), dual optimization (dual opt.) after 50 iterations, and gradient-based distributed optimization (GBDDO) after 4 iterations

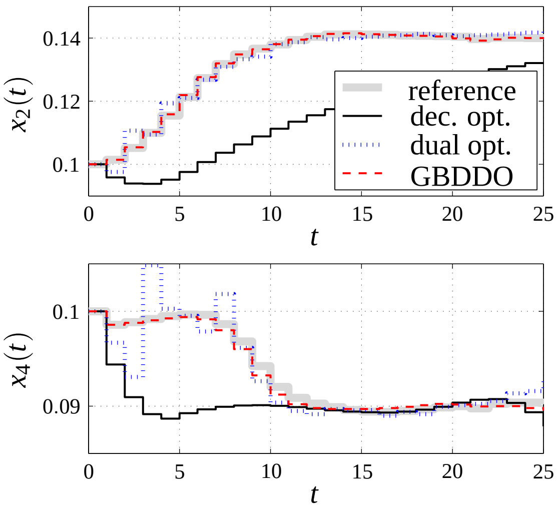


Figure 2.4: Trajectories of the heights x_2 and x_4 for the 4 different implemented methods for $[\alpha, \beta] = [0.7, 0.6]$: centralized optimization (reference), decentralized optimization (dec. opt.), dual optimization (dual opt.) after 50 iterations, and gradient-based distributed optimization (GBDDO) after 4 iterations

be approximated by discretization. As, for large-scale systems, the calculation of sensitivities is a demanding task of dynamic optimization, it is to be decided whether to choose a coarse discretization, which results in an unprecise sensitivity information, or a fine discretization, that leads to an additional computational burden.

The GBDDO method has been successfully applied to a simple nonlinear differential-algebraic system, namely a 4-tank system, where convergence has been achieved with only few iterations.

Future work will concentrate on the implementation of the GBDDO method for continuous-time systems and on improving the computation of the interaction sensitivities. In order to verify the results presented above, the application to a real application problem with stronger nonlinearities will be considered.

Chapter 3

An improved distributed version of Han's method for distributed MPC of canal systems

The work of this chapter has been developed by Minh Dang Doan, Tamás Keviczky and Bart De Schutter, and it has been published in [14].

Abstract

Recently, we have introduced a distributed version of Han's method that can be used for distributed model predictive control (DMPC) of dynamically coupled linear systems, under coupling constraints [13]. Some DMPC problems of water networks can be cast into this type. In this paper, we propose an improved version of this method and apply it to a canal system. The simulation results show that the modifications lead to faster convergence of the method, thus making it more practical in control of water networks.

3.1 Introduction

Optimization techniques have played a fundamental role in designing automatic control systems for most part of the past half century. This dependence is even more obvious in today's wide-spread use of online optimization-based control methods, such as Model Predictive Control (MPC) [30, 46]. The ability to express important process constraints and characterize comprehensive economic objective functions has made MPC the industry standard for controlling large-scale systems ranging from chemical processes to basic infrastructure.

For control of large-scale networked systems, *centralized* MPC may be considered impractical, inflexible, and unsuitable due to information exchange requirements and computational aspects. The subsystems in the network may belong to different authorities that prevent sending all necessary information to one processing center. Moreover, the optimization problem yielded by centralized MPC can be excessively large for real-time computation. In order to deal with these limitations, *distributed model predictive control* (DMPC) has been proposed for control of such large-scale systems, by decomposing the overall system into small subsystems [9, 22, 47]. The subsystems then employ distinct

MPC controllers that only solve local optimization problems, use local information from neighboring subsystems, and collaborate to achieve globally attractive solutions.

Approaches to DMPC design differ from each other in the problem setup. For systems with decoupled dynamics, [16] proposed a DMPC scheme focusing on multiple vehicles with coupled cost functions, and utilizing predicted trajectories of the neighbors in each subsystem's optimization. A DMPC scheme with a sufficient stability test for dynamically decoupled systems was proposed by [24], in which each subsystem optimizes also over the behaviors of its neighbors. [48] 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 [63] proposed a distributed MPC scheme, based on a Jacobi algorithm that deals with the primal problem, using a convex combination of new and old solutions. Other research related to the DMPC field is reported by [3, 4, 10, 15, 21, 27, 34, 40]. A recent survey on DMPC can be found in [52].

Recently, we have developed a distributed version of Han's parallel method for convex optimization [13]. The method aims to define local controllers for dynamically coupled subsystems, which share coupling constraints and minimize a separable objective function. Relying on a decomposition of the dual optimization problem such that local problems have analytical solutions, the algorithm has an iterative update procedure which converges asymptotically to the global optimizer of the primal problem. At each iteration, the controllers exchange information with other "neighboring" subsystems, with which they are "connected" in terms of dynamics or constraint coupling.

In this paper, we present an improved distributed version of Han's parallel algorithm for a class of convex optimization problems [13,20] and show that it is applicable for DMPC of water networks. The improvements are illustrated in a simulation of the new DMPC scheme for a 4-reach canal system. The paper is organized as follows. The problem setup is described in Section 3.2. In Section 3.3, we summarize the original Han's method and the distributed version, followed by the new modified distributed version to speed up the convergence of the algorithm. The simulation results in Section 3.4 illustrate the properties of the DMPC scheme for the example setup of the 4-reach canal. Section 3.5 concludes the paper and indicates some directions for future research.

3.2 Problem setup

3.2.1 The canal system

In this paper we illustrate the application of the novel DMPC approach to the control of a system of irrigation canals. Irrigation canals are large systems, consisting of many interacting components, and spanning vast geographical areas. For the most safe and efficient 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 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 3.1. 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 master controller [59]. We will use the new DMPC method to design the master controllers.

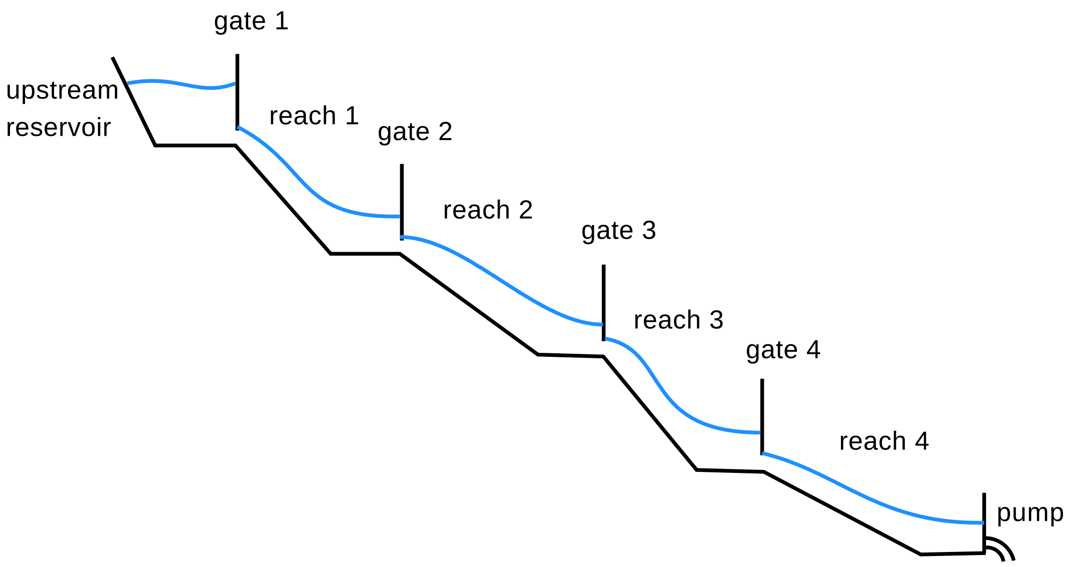


Figure 3.1: The example canal system

3.2.2 Modeling the canal

Subsystem modeling

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 4th subsystem has one more controller, corresponding to the pump at its downstream end.

We first use a simplified model for each subsystem as illustrated in Figure 3.2, and then obtain an overall model by connecting 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:

$$h^i_{k+1} - h^i_k = \frac{T_s}{A^i_s}\left[\left(Q^i_{\text{in}}\right)_k - \left(Q^i_{\text{out}}\right)_k\right] \tag{3.1}$$

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 (zero level is set at the autonomous steady state), A_s is the water surface (volume of reservoir = $h \cdot A_s$), Q_{in} and Q_{out} are in-flow and 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 $Q^4_{\text{out}} = p^4$.

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

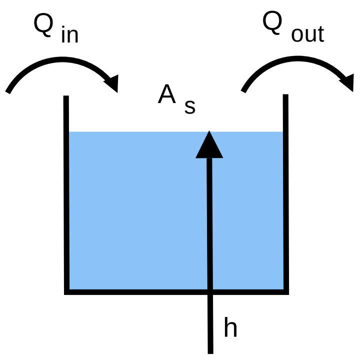


Figure 3.2: Model of a reach

model of the form:

$$x_{k+1}^{i} = \sum_{j=1,\cdots,4} A^{ij} x_k^j + B^{ij} u_k^j, \tag{3.2}$$

with the state-space matrices:

$$\begin{aligned}
A^{ii} &= \;1\;, \quad i = 1,\cdots,4; \quad A^{ij} = \;0\;, \quad i \neq j \\
B^{ii} &= \;T_\mathrm{s}/A_\mathrm{s}^i\;, \quad i = 1,2,3; \quad B^{44} = \begin{bmatrix} T_\mathrm{s}/A_\mathrm{s}^4 & -T_\mathrm{s}/A_\mathrm{s}^4 \end{bmatrix} \\
B^{i(i+1)} &= \;-T_\mathrm{s}/A_\mathrm{s}^i\;, \quad i = 1,2; \quad B^{34} = \begin{bmatrix} -T_\mathrm{s}/A_\mathrm{s}^4 & 0 \end{bmatrix} \\
B^{ij} &= 0, \quad j \notin \{i, i+1\}
\end{aligned}$$

Centralized MPC problem

The centralized MPC problem makes use of a quadratic cost function:

$$J = \sum_{i=1}^{4} \sum_{k=0}^{N-1} \left(\left(u_k^i\right)^T R_i u_k^i + \left(x_{k+1}^i\right)^T Q_i x_{k+1}^i \right) \tag{3.3}$$

in which N is the prediction horizon, and $\{Q_i, R_i\}_{i=1,\cdots,4}$ are given positive definite weights. It is easy to verify that this cost function can be rewritten as $J = \mathbf{x}^T H \mathbf{x}$ where H is a block-diagonal, positive definite matrix.

The constraints of the optimization problem include dynamical constraints (i.e. the model equations), initial state constraint, terminal constraint $x_N^i = 0, i = 1,\cdots,4$, and local state and input constraints:

$$|u_k^i| \leq u_{\max}^i, \qquad |x_k^i| \leq x_{\max}^i$$

Following the method of [13], the optimization problem to be solved by the centralized MPC controller at each sampling interval can be represented in a compact form as

$$\begin{aligned}
\min_{\mathbf{x}} \quad & \mathbf{x}^T H \mathbf{x} \\
\text{s.t.} \quad & a_l^T \mathbf{x} = b_l, \quad l = 1,\dots,n_\mathrm{eq} \\
& a_l^T \mathbf{x} \leq b_l, \quad l = n_\mathrm{eq}+1,\dots,s
\end{aligned} \tag{3.4}$$

with $s = n_\mathrm{eq} + n_\mathrm{ineq}$, where n_eq and n_ineq are the number of scalar equality and inequality constraints, respectively.

3.3 Distributed Model Predictive Control method

The optimization problem (3.4) will be solved by a distributed algorithm that is based on Han's parallel method for convex programs [20]. In the following we will give a summary of Han's method for convex quadratic programs, and then describe the distributed version of Han's method for quadratic programs in the form (3.4). Then we will proceed by describing the modified distributed version, which is the main contribution in this paper.

3.3.1 Han's parallel method for convex quadratic programs

The original Han's method considers general convex optimization problems where the constraint is an intersection of many convex sets. The algorithm is based on Fenchel's duality to perform a dual decomposition, and iteratively projects the dual variables onto local constraint sets. The sum of dual variables can be shown to converge to the minimizer of the dual problem [20]. A simplified version of Han's method for the quadratic optimization problem (3.4) is summarized in Algorithm 3.3.1.

Assumption 3.3.1 ***Han's method for convex programs***

Choose parameter α big enough[1]*. For $p = 1, 2, \ldots$:*

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

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

2) *Assign $\mathbf{y}_l^{(p)} = \mathbf{y}_l^{(p-1)} + (1/\alpha)\left(\mathbf{z}_l^{(p)} - \mathbf{x}^{(p-1)}\right)$*

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

4) *Compute: $\mathbf{x}^{(p)} = H^{-1}\mathbf{y}^{(p)}$*

In this representation, each vector $\mathbf{y}_l$ is a dual variable corresponding to l^{th} constraint. For problem (3.4), Han's method was proved to converge to the global optimum if the cost function is strongly convex, or equivalently if H is positive definite [20]. An interesting property of this method is that the number of parallel processes is equal to the number of constraints (as opposed to other dual decomposition methods where the number of parallel processes often equals the number of variables).

3.3.2 Distributed version of Han's method

Han's algorithm involves calculation of the global variables, therefore a global coordination method is required. A distributed version of Han's method was proposed by [13], and it makes use of the explicit solutions in Step 1 of Algorithm 1, and exploits the structure of (3.4) to decompose the computations, hence avoiding global communications.

[1] [20] recommended $\alpha = \alpha_0 \triangleq s/\rho$, where s is the number of constraints and ρ is one half of the smallest eigenvalue of H.

The main idea behind the distributed version of Han's method is illustrated in Figures 3.3 and 3.4, 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 illustrate 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 only communicates with the other subsystems of which the variables are necessary for computing the updates of its local variables.

The distributed version of Han's method was proved to achieve the same convergence property as the original method of Han [13].

3.3.3 Modifications of Han's method to speed up convergence

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 (3.4). Therefore, we need to modify the method to achieve better convergence rate.

In this paper, we present 2 modifications of the distributed version of Han's method:

- Scaling of the step sizes related to dual variables by using different α_l values for the update of each dual variable l 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.

We will use the same notations as in [13, Section VI], which are briefly summarized below:

- L_i: the set of indices of constraints that subsystem i is responsible for updating their dual variables throughout the algorithm.
- $\mathcal{N}^i$: the *neighborhood* of subsystem i, consisting of i itself and other subsystems that have direct dynamical or constraint couplings with subsystem i.
- $L_{\mathcal{N}^i}$: the set of indices of constraints within responsibility of all subsystems in $\mathcal{N}^i$.
- $\mathbf{x}^{(p)|i}$: the *self image* of the global variable vector $x^{(p)}$ made by subsystem i; this vector has the same size as $\mathbf{x}^{(p)}$, containing all variables of subsystem i at the right positions, and zeros for the other entries.
- $\mathbf{x}^{(p)|\mathcal{N}^i}$: the *neighborhood image* of $x^{(p)}$ made by subsystem i, using variables of all subsystems inside $\mathcal{N}^i$ at the right positions, and zeros for the other entries.
- $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$: the *assumed neighborhood image* $x^{(p)}$ made by subsystem i. The difference between $\mathbf{x}^{(p)|\mathcal{N}^i}_{\text{assumed}}$ and $\mathbf{x}^{(p)|\mathcal{N}^i}$ is that only the values of variables belonging to subsystem i are correct, while for the variables of other neighboring subsystems $j \in \mathcal{N}^i \setminus \{i\}$, the values could be different from the real ones.
- $\mathfrak{I}^i$: index matrix of subsystem i; it is the mask for the global variable $\mathbf{x}$ such that only variables of subsystem i are kept, i.e. $\mathbf{x}^{(p)|i} = \mathfrak{I}^i \mathbf{x}^{(p)}$.

We present the improved distributed version of Han's method in the following algorithm:

Assumption 3.3.2 ***Improved distributed algorithm for the MPC optimization problem***

Pre-computed 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 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 this section.*
- *For each $l \in L_i$: $\overline{c_l} = \frac{-1}{a_l^T a_l} H^{-1} a_l$. We can see that $\overline{c_l}$ 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.*

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. 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 idea of using previously predicted states and inputs for initialization is a popular technique in MPC [46]. 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.

The current state is plugged into the MPC problem, then we get an optimization problem of the form (3.4)*. This problem will be solved in a distributed way by the following iterative procedure.*

Distributed iterative procedure to solve the optimization problem:

Initialize with $p = 0$. Each subsystem i communicates with the neighbors $j \in \mathcal{N}^i$ to get $\mathbf{x}^{(0)|j}$, then constructs $\mathbf{x}^{(0)|\mathcal{N}^i} = \sum_{j \in \mathcal{N}^i} \mathbf{x}^{(0)|j}$. Subsystem i computes its local dual variable $\mathbf{y}^{(0)|\mathcal{N}^i} = H\mathbf{x}^{(0)|\mathcal{N}^i}$, and then computes initial intermediate variables:

$$\gamma_l^{(0)} = \max\{a_l^T (\mathbf{x}^{(0)|\mathcal{N}^i} - \mathbf{y}^{(0)|\mathcal{N}^i}) - b_l, 0\}, \quad l \in L_i$$

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

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

   Each controller i communicates with its neighbors $j \in \mathcal{N}^i$ to get updated values of their variables, contained in $\mathbf{x}^{(p-1)|j}$. Vice versa, i also sends its updated variables in $\mathbf{x}^{(p-1)|i}$ to its neighbors as requested.

   After getting information from the neighbors, controller i constructs the neighborhood image *$\mathbf{x}^{(p-1)|\mathcal{N}^i}$ as:*

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

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

   In this step, the local controllers update γ_l corresponding to each constraint l under their responsibility. More specifically, each local controller i updates γ_l for each $l \in L_i$ in the following manner:

- *If constraint l is an equality constraint ($l \in \{1,\dots,n_{\text{eq}}\}$), then* $\gamma_l^{(p)} = a_l^T \boldsymbol{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l$.
- *If constraint l is an inequality constraint ($l \in \{n_{\text{eq}}+1,\dots,s\}$), then* $\gamma_l^{(p)} = \max\{a_l^T \boldsymbol{x}^{(p-1)|\mathcal{N}^i} + \gamma_l^{(p-1)} - b_l, 0\}$.

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

Each local controller i communicates with its neighbors to get updated $\gamma_l^{(p)}$ values that the neighbors just computed in Step 2).

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

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 *of* $\boldsymbol{x}$*:*

$$\boldsymbol{x}_{assumed}^{(p)|\mathcal{N}^i} = \sum_{l \in L_{\mathcal{N}^i}} \frac{1}{\alpha_l} \gamma_l^{(p)} \overline{c}_l \tag{3.5}$$

Then controller i selects the values of its variables in $\boldsymbol{x}_{assumed}^{(p)|\mathcal{N}^i}$ to construct the new self image*:*

$$\boldsymbol{x}^{(p)|i} = \mathfrak{J}^i \boldsymbol{x}_{assumed}^{(p)|\mathcal{N}^i} \tag{3.6}$$

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

After updating their variables, each local controller checks the local termination criteria. When all local controllers have converged[2]*, the algorithm stops and the local control actions are implemented, otherwise the controllers proceed to Step 1) to start a new iteration.*

Implement MPC input:

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.

Method to choose the scaling vector:

In the modified version of distributed Han's method, a good choice of the scaling vector helps to dramatically improve the convergence speed. 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 "slower convergent" dual variables. In our simulation, we rely on the Hessian to find the scaling vector. Specifically, for a subsystem i whose variables have the average weight $\overline{h}_i$ (e.g. average of entries related to i's states and inputs in the diagonal of the Hessian), we choose the scale factor $\left(k_\alpha\right)_l = 1/\overline{h}_i$, with all $l \in L_i$. We also multiply the scaling vector k_α with a factor $\theta < 0$ for enlarging the step sizes of all dual variables; this θ is tuned in the first MPC step.

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 reuse the same scaling vector.

[2]Checking the termination criteria in a distributed fashion requires a dedicated logic scheme, the description of which is omitted for brevity.

3.4 Simulation results and discussion

DMPC methods are applied to the regulation problem of the simulated canal system of Section 3.2, which has a perturbed initial state. We use distributed Han's method with and without the modifications described in Section 3.3.3 for the same setup, and compare the results. Figure 3.5 shows that the distributed Han's method with modifications achieves better convergence rate, allowing the distributed optimization to converge within an acceptable number of iterations. A simulation of closed-loop MPC is performed for 20 sample steps. Figure 3.6 shows that the distributed solutions converge to the centralized solutions in every sample step.

Although the new scheme is verified by this simulation, there are still several theoretical issues that need to be addressed:

Firstly, there is no convergence proof for the modified distributed version of Han's method yet. We observe that in setups that are more complicated, the method to choose scaling vector proposed in this simulation does not always work well (sometimes after several sample steps, the algorithm does not converge in the next sample steps). Note that with this method we aim to solve the dual problem, therefore the primal iterate would be infeasible unless the algorithm converges.

Secondly, in the MPC formulation we keep both inputs and states as variables of the centralized optimization problem. This formulation is advantageous in distributed MPC because the Hessian will have a diagonal structure, and the *neighborhood* of each subsystem will only contain 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. In future research, we will also study MPC formulations in which all states are eliminated, so that the centralized optimization only has inequality constraints. Such formulation would allow stopping the algorithm in a finite number of steps, and the final iterate could be feasible (although it may be suboptimal).

Another problem is that the proposed method is for *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 for other decomposition methods that can handle more general problems, e.g. convex problems with linear or decoupled nonlinear constraints.

Last but not least, the MPC formulation in this paper employs the terminal constraint $x_N = 0$, which is conservative since it reduces the domain of attraction of MPC. An improvement could be made by replacing this constraint with less restrictive conditions (e.g. terminal constraint set and terminal controller). However, there is still no distributed scheme to construct the terminal constraint set and the terminal controller (and also the terminal penalty matrix that is solution of the Riccati equation), other than assuming them to be completely decoupled.

3.5 Conclusions

The modified distributed version of Han's method has an improved convergence rate, thus it is more suitable for DMPC of large-scale water networks. Future research will involve finding a way to construct the scaling vector of the modified distributed version of Han's method together with a theoretical proof of the convergence. We will also investigate different distributed optimization methods using dual decomposition techniques to address nonlinear MPC with more general optimization problem. Another direction is to find distributed MPC schemes for suboptimal MPC.

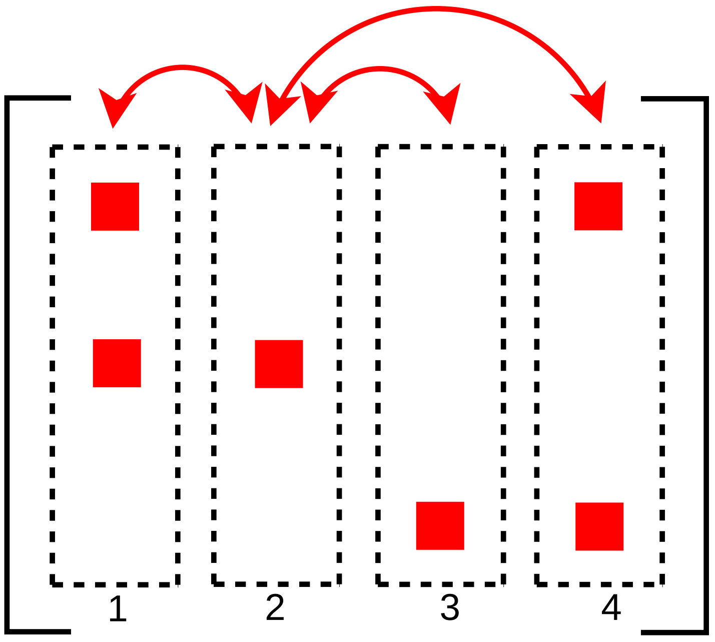


Figure 3.3: Communication links of the 2nd subsystem in the *centralized* coordination version of Han's algorithm for an example 4-subsystem problem. An update for a global variable requires the 2nd subsystem to communicate with all the others.

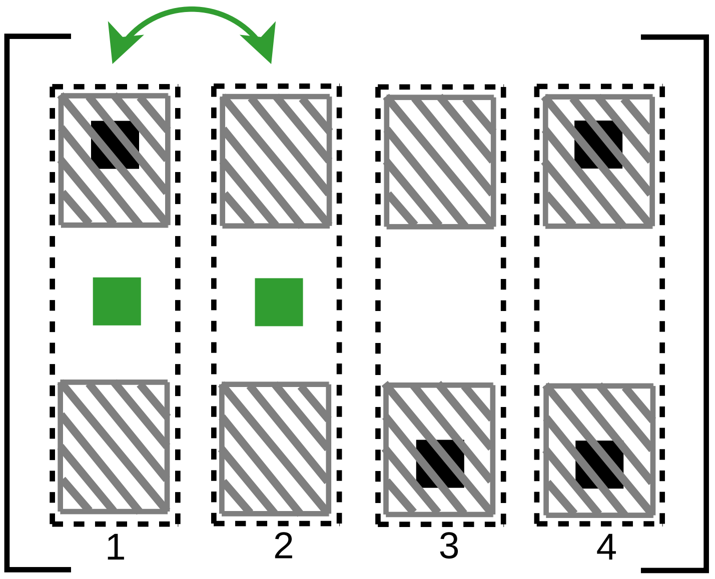


Figure 3.4: Communication link of the 2nd subsystem in the *distributed* coordination version of Han's algorithm for an example 4-subsystem problem. The 2nd subsystem only cares about its local variable, therefore it does not need to communicate with the others that do not couple with it.

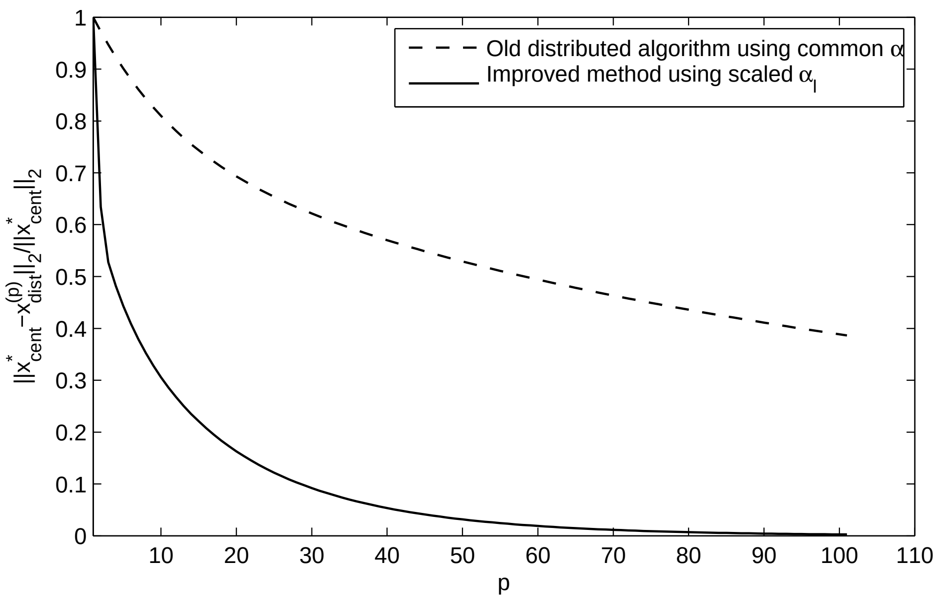


Figure 3.5: Comparison of convergence rates of the former and the new distributed versions of Han's method in the first sampling time (*k*=1)

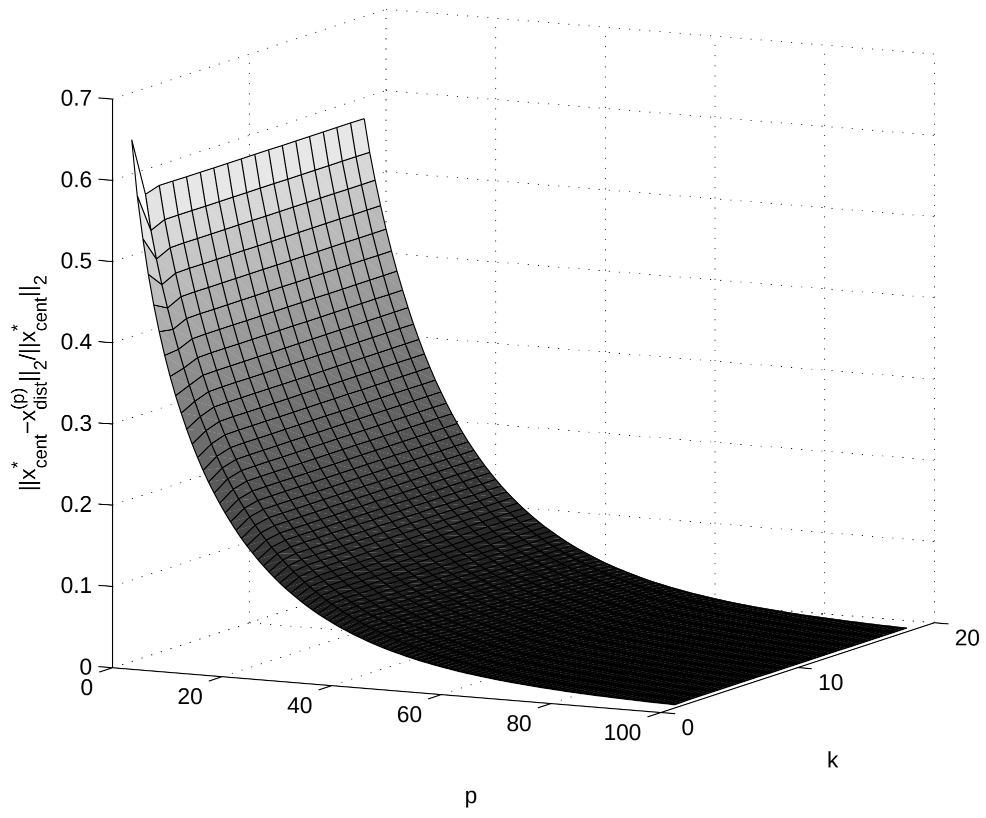


Figure 3.6: Normalized norm of difference between the centralized and the distributed solutions versus the iteration step *p* and sample step *k*

Chapter 4

Distributed MPC based on a cooperative game

In this chapter we present the distributed MPC based on a cooperative game scheme presented in [31]. This control scheme considers the following class of distributed linear systems in which two subsystems coupled with the neighbor subsystem through the inputs are defined

$$\begin{aligned} x_1(t+1) &= A_1x_1(t)+B_{11}u_1(t)+B_{12}u_2(t) \\ x_2(t+1) &= A_2x_2(t)+B_{21}u_1(t)+B_{22}u_2(t) \end{aligned} \tag{4.1}$$

where $x_i \in \mathbb{R}^{n_i},\ i=1,2$ are the states of each subsystem and $u_i \in \mathbb{R}^{m_i},\ i=1,2$ are the different inputs.

The control objective is to regulate the system to the origin while guaranteeing that a given set of state and input constraints are satisfied. The proposed distributed scheme assumes that for each subsystem, there is an agent that has access to the model and the state of that subsystem. The agents do not have any knowledge of the dynamics of their neighbor, but can communicate freely among them in order to reach an agreement. The proposed strategy is based on negotiation between the agents on behalf of a global performance index. At each sampling time, agents make proposals to improve an initial feasible solution on behalf of their local cost function, state and model. These proposals are accepted if the global cost improves the corresponding to the current solution. The trajectories chose are denoted as U_1^d and U_2^d.

To this end, the MPC controllers to minimize the sum of two local performance indexes J_1 and J_2 that depend on the future evolution of both states and inputs. Each agent solves a sequence of reduced dimension optimization problems to determine the future input trajectories U_1 and U_2 based on the model of its subsystem. We summarize next, the DMPC algorithm proposed in [31]:

1. At time step t, each agent i receives its corresponding partial state measurement $x_i(t)$.

2. Both agents communicate. Agent 1 sends $K_1x_{1,N}$ and agent 2 sends $K_2x_{2,N}$, where $x_{1,N}$ is the N-steps ahead predicted state obtained from the current state applying $U_1^d(t-1), U_2^d(t-1)$ shifted one time step. This information is used to generate the shifted trajectories $U_i^s(t)$, which is the initial solution.

3. Each agent i minimizes J_i assuming that the neighbor keeps applying the shifted optimal trajectory evaluated at the previous time step $U_{ni}^s(t)$. The optimal solution is denoted $U_i^*(t)$.

4. Each agent i minimizes J_i optimizing the neighbor input assuming that it applies the shifted input trajectory U_{ni}^s. Solving this optimization problem, agent i defines an input trajectory denoted $U_{ni}^w(t)$ for its neighbor that optimizes its local cost function J_i.

Table 4.1: Cost function table used for the decision making.

	$U_2^s(t)$	$U_2^*(t)$	$U_2^w(t)$
$U_1^s(t)$	$J_1(x_1(t),U_1^s(t),U_2^s(t))$ $+J_2(x_2(t),U_2^s(t),U_1^s(t))$	$J_1(x_1(t),U_1^s(t),U_2^*(t))$ $+J_2(x_2(t),U_2^*(t),U_1^s(t))$	$J_1(x_1(t),U_1^s(t),U_2^w(t))$ $+J_2(x_2(t),U_2^w(t),U_1^s(t))$
$U_1^*(t)$	$J_1(x_1(t),U_1^*(t),U_2^s(t))$ $+J_2(x_2(t),U_2^s(t),U_1^*(t))$	$J_1(x_1(t),U_1^*(t),U_2^*(t))$ $+J_2(x_2(t),U_2^*(t),U_1^*(t))$	$J_1(x_1(t),U_1^*(t),U_2^w(t))$ $+J_2(x_2(t),U_2^w(t),U_1^*(t))$
$U_1^w(t)$	$J_1(x_1(t),U_1^w(t),U_2^s(t))$ $+J_2(x_2(t),U_2^s(t),U_1^w(t))$	$J_1(x_1(t),U_1^w(t),U_2^*(t))$ $+J_2(x_2(t),U_2^*(t),U_1^w(t))$	$J_1(x_1(t),U_1^w(t),U_2^w(t))$ $+J_2(x_2(t),U_2^w(t),U_1^w(t))$

5. Both agents communicate. Agent 1 sends $U_1^*(t)$ and $U_2^w(t)$ to agent 2 and receives $U_2^*(t)$ and $U_1^w(t)$.

6. Each agent evaluates the local cost function J_i for each the nine different possible combination of input trajectories; that is $U_1 \in \{U_1^s(t), U_1^w(t), U_1^*(t)\}$ and $U_2 \in \{U_2^s(t), U_2^w(t), U_2^*(t)\}$.

7. Both agents communicate and share the information of the value of local cost function for each possible combination of input trajectories. In this step, both agents receive enough information to take a cooperative decision.

8. Each agent applies the input trajectory that minimizes $J = J_1 + J_2$. Because both agents have access to the same information after the second communication cycle, both agents choose the same optimal input sets $U_1^d(t), U_2^d(t)$.

9. The first input of each optimal sequence is applied and the procedure is repeated the next sampling time.

From a game theory point of view, at each time step both agents are playing a cooperative game. This game can be synthesized in strategic form by a three by three matrix. Each row represents one of the three possible decisions of agent 1, and each column represents one of the three possible decisions of agent 2. The cells contain the sum of the cost functions of both agents for a particular choice of future inputs. At each time step, the option that yields a lower global cost is chosen. Note that both agents share this information, so they both choose the same option. The nine possibilities are shown in table 4.1.

The proposals made are suboptimal because each agent has an incomplete view of the system and they propose the best solutions from their point of view. The proposed algorithm has low communication and computational burdens and provides a feasible solution to the centralized problem. In addition, sufficient conditions that guarantee practical stability of the closed-loop system as well as an optimization based procedure to design the controller so that these conditions are satisfied are provided, see [31] for more details.

4.1 Design procedure

The proposed benchmark is based on a nonlinear model and consists of several reference steps of the levels of tanks 1 and 2. In order to test the proposed DMPC scheme a discrete time linear model around the equilibrium point h_0, q_0 (which corresponds to the first reference) has been obtained linearizing the nonlinear model of the quadruple tank process with a sampling time of 5s. The state and input variables of the linearized model are defined as follows

$$x_1 = \begin{bmatrix} h_1 - h_{10} \\ h_3 - h_{30} \end{bmatrix}, u_1 = \begin{bmatrix} q_a - q_{a0} \end{bmatrix}, x_2 = \begin{bmatrix} h_2 - h_{20} \\ h_4 - h_{40} \end{bmatrix}, u_2 = \begin{bmatrix} q_b - q_{b0} \end{bmatrix}$$

The objective of the MPC controllers is to minimize a performance index that depends on the future evolution of both states and inputs based on the following local cost functions

$$\begin{aligned} J_1(x_1, U_1, U_2) &= \sum_{k=1}^{N} (x_{1,k} - x_{1r})^T Q_1 (x_{1,k} - x_{1r}) + \sum_{k=0}^{N-1} (u_{1,k} - u_{1r})^T R_1 (u_{1,k} - u_{1r}) \\ J_2(x_2, U_2, U_1) &= \sum_{k=1}^{N} (x_{2,k} - x_{2r})^T Q_2 (x_{2,k} - x_{2r}) + \sum_{k=0}^{N-1} (u_{2,k} - u_{2r})^T R_2 (u_{2,k} - u_{2r}) \end{aligned}$$

where $N = 5$, $x_{i,k}$ and $u_{i,k}$ are the k-steps ahead predicted states and inputs of agent i respectively. The variables $x_{i,r}$ and $u_{i,r}$ are the target state and input obtained from the difference between the equilibrium point and the reference levels and flows. To determine these values, the nonlinear model has been used to obtain the levels of h_3, h_4 and the corresponding equilibrium flows q_a, q_b that guarantee that the references are an equilibrium point of the system. This implies that it has been done in a centralized manner. The agents receive the appropriate references as inputs. In this point we have to remark the fact that when the reference is switched from one working point to another one it is necessary to reset the value of U_s to a feasible solution. This is necessary in order to guarantee a decreasing cost and the stability. Note that for this particular benchmark, no terminal region has been taken into account.

The weighting matrices were chosen to minimize the benchmark objective function, that is, $Q_1 = Q_2 = I$, $R_1 = R_2 = 0.01$. The local controller gains for each agent were $K_1 = [0.17\ 0.21]$ and $K_2 = [-0.16\ -0.14]$. These gains were designed with LMI techniques in order to stabilize both subsystems independently while assuring the stability of the centralized system. Following the procedure detailed previously it is possible to calculate a distributed invariant set corresponding to this gain. The role of these gains is important because the option in the game that allows to guarantee closed-loop stability is constructed shifting the last decided control action; that is, the first element is dropped after it is applied in the system and a term evaluated with these gains is added at the end of the horizon control vector, see [31] for more details.

The proposed distributed MPC controller only needs three communication steps in order to obtain a cooperative solution to the centralized optimization problem, has low communication and computational burdens and provides a feasible solution to the centralized problem. The simulation and experimental results show that the distributed scheme is able to control the system. Note that in this case, because the control input is decided by consensus, the pairing does not affect the performance of the distributed control scheme if the states are grouped correctly.

Chapter 5

Cooperative Distributed MPC for tracking

The work of this chapter has been developed by Daniel Limon at University of Seville.

5.1 Problem statement

Consider a system described by a linear invariant discrete time model

$$\begin{aligned} x^+ &= Ax+Bu \\ y &= Cx+Du \end{aligned} \tag{5.1}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the current control vector, $y \in \mathbb{R}^p$ is the controlled output and x^+ is the successor state. The solution of this system for a given sequence of control inputs $\mathbf{u}$ and initial state x is denoted as $x(j) = \phi(j,x,\mathbf{u})$ where $x = \phi(0,x,\mathbf{u})$. The state of the system and the control input applied at sampling time k are denoted as $x(k)$ and $u(k)$ respectively. The system is subject to hard constraints on state and control:

$$x(k) \in X, \quad u(k) \in U \tag{5.2}$$

for all $k \geq 0$. $X \subset \mathbb{R}^n$ and $U \subset \mathbb{R}^m$ are compact convex polyhedra containing the origin in its interior. It is assumed that the following hypothesis hold.

Assumption 1 *The pair (A,B) is stabilizable and the state is measured at each sampling time.*

In this work, a decentralized control framework is considered. Thus, it is assumed that system (5.1) can be partitioned in M subsystems of the form [46]:

$$\begin{aligned} x_i^+ &= A_i x_i + \sum_{j=1}^{M} \bar{B}_{ij} u_j \\ y_i &= C_i x_i + \sum_{j=1}^{M} \bar{D}_{ij} u_j \end{aligned} \tag{5.3}$$

where $x_i \in \mathbb{R}^{n_i}$, $u_j \in \mathbb{R}^{m_j}$, $y_i \in \mathbb{R}_i^p$, $A_i \in \mathbb{R}^{n_i \times n_i}$ and $B_{ij} \in \mathbb{R}^{n_i \times m_j}$.

For the sake of simplicity of the exposition, the results will be presented for the case of two players game. In this case, the plant can be represented in the form:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^+ = \begin{bmatrix} A_1 & \\ & A_2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \bar{B}_{11} \\ \bar{B}_{21} \end{bmatrix} u_1 + \begin{bmatrix} \bar{B}_{12} \\ \bar{B}_{22} \end{bmatrix} u_2$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 & \\ & C_2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \bar{D}_{11} \\ \bar{D}_{21} \end{bmatrix} u_1 + \begin{bmatrix} \bar{D}_{12} \\ \bar{D}_{22} \end{bmatrix} u_2$$

The steady state, input and output of the plant (x_s, u_s, y_s) are such that (5.1) is fulfilled, i.e. $x_s = Ax_s + Bu_s$ and $y_s = Cx_s + Du_s$. Under assumption 1, the set of steady states and inputs of system (5.1) is a m-dimensional linear subspace of $\mathbb{R}^{n+m}$ [5] given by

$$(x_s, u_s) = M_\theta \theta \tag{5.4}$$

$$y_s = N_\theta \theta \tag{5.5}$$

where M_θ is a full column rank such that $[A-I, B]M_\theta = 0$ and $N_\theta = [CD]M_\theta$. Therefore, every pair of steady state and input $(x_s, u_s) \in \mathbb{R}^{n+m}$ is characterized by a given parameter $\theta \in \mathbb{R}^m$. For the partitioned model of the plant this can be rewritten as

$$(x_{s,i}, u_{s,i}) = M_{\theta,i}\ \theta$$

The problem we consider is the design of a distributed MPC controller $u_i = \kappa_i(x, y_t)$ to track a (possible time-varying) plant-wide target output y_t, such that the subsystems are steered (as close as possible) to the target while fulfilling the constraints.

5.2 Cooperative MPC

Among the existing solutions for the decentralized predictive control problem, we focus our attention on the cooperative game [46]. In this case, the two players share a common objective, which can be considered as the overall plant objective.

$$V(x_1, x_2, y_t; \mathbf{u}_1, \mathbf{u}_2) = \sum_{k=0}^{N-1} \|x(k) - x_t\|_Q^2 + \|u(k) - u_t\|_R^2 + \|x(N) - x_t\|_P^2$$

where (x_t, u_t, y_t) defines the state, input and output of the target.

In cooperative decentralized MPC, each i-th agent calculates its corresponding input u_i by solving an iterative decentralized optimization problem. At the sampling time k, the solution of the agent i at the iteration p will be denoted as $\mathbf{v}_i(k)^{[p]}$. The optimization problem to be solved by i-th agent at iteration $p+1$ and at the state (x_1, x_2) is the following:

$$\mathbf{v}_i^o = \arg\min_{\mathbf{u}_i} V(x_1, x_2, y_t; \mathbf{v}_1, \mathbf{v}_2) \tag{5.6}$$

$$s.t. \quad (5.3) \text{ with } x(0) = (x_1, x_2), u_j(i) = v_j(i) \tag{5.7}$$

$$\mathbf{v}_j = \mathbf{v}_j^{[p]}, \quad j \in \mathbb{I}_{1,2} \setminus \{i\}, \tag{5.8}$$

$$\mathbf{v}_i \in \mathcal{U}_i, \tag{5.9}$$

$$S^u x(N) = 0 \tag{5.10}$$

where it is assumed that the agent knows the whole state of the plant and the solution of the decentralized controller at the last iteration p. Denoting the optimal solution of this problem as $\mathbf{v}_i^o$, then the solution at the current iteration $p+1$ will be given by

$$\mathbf{v}_1^{[p+1]} = w_1\mathbf{v}_1^o + w_2\mathbf{v}_1^{[p]} \tag{5.11}$$

$$\mathbf{v}_2^{[p+1]} = w_1\mathbf{v}_2^{[p]} + w_2\mathbf{v}_2^o \tag{5.12}$$

After a certain number of iterations, the each controller provides the solution denoted as $\mathbf{v}_i^*$.

It has been demonstrated that this decentralized approach ensures recursive feasibility, optimality and asymptotic stability under mild assumptions. See [46, Chapter 6] for a more detailed exposition.

As in the centralized case, when the target changes the decentralized controller may fail to track the new target due to the loss of feasibility. In [17, 28] a novel formulation of the MPC for tracking is presented. The way this controller handle the tracking problem is characterized by (i) considering an artificial steady state and input as decision variables, (ii) penalizing the deviation of the predicted trajectory with the artificial steady conditions, (iii) adding a quadratic offset-cost function to penalize the deviation between the artificial and the target equilibrium point and (iv) considering an extended terminal constraint. In this work, this controller is extended to the case of a cooperative distributed MPC formulation.

5.3 Cooperative MPC for tracking

As in the centralized case, an artificial equilibrium point (x_s, u_s, y_s) (represented by the corresponding parameter θ) is added as decision variable and the following modified cost function is considered:

$$V_t(x_1,x_2,y_t;\mathbf{u}_1,\mathbf{u}_2,\theta) = \sum_{k=0}^{N-1} \|x(k)-x_s\|_Q^2 + \|u(k)-u_s\|_R^2 + \|x(N)-x_s\|_P^2 + V_O(y_s-y_t)$$

where $V_O(y_s-y_t)$ is a convex function which penalizes the deviation of the artificial output to the target. Typically this is chosen as a norm of this distance [17].

At each sampling time k and iteration $p+1$, each subsystem $i \in \mathbb{I}_{[1,2]}$ solves the following optimization problem:

$$(\mathbf{u}_i^o, \theta_i^o) = \arg\min_{\mathbf{u}_i,\theta} V(x_1,x_2,y_t;\mathbf{u}_1,\mathbf{u}_2,\theta) \tag{5.13}$$

$$s.t. \quad (5.3) \text{ with } x(0) = (x_1,x_2), \tag{5.14}$$

$$\mathbf{u}_j = \mathbf{u}_j^{[p]}, \quad j \in \mathbb{I}_{1,2} \setminus \{i\}, \tag{5.15}$$

$$\mathbf{u}_i \in \mathcal{U}_i, \tag{5.16}$$

$$(x(N),\theta) \in \Omega_{t,K} \tag{5.17}$$

The solution of the $p+1$-iteration is given by

$$\mathbf{u}_1^{[p+1]} = w_1\mathbf{u}_1^o + w_2\mathbf{u}_1^{[p]} \tag{5.18}$$

$$\mathbf{u}_2^{[p+1]} = w_1\mathbf{u}_2^{[p]} + w_2\mathbf{u}_2^o \tag{5.19}$$

$$\theta^{[p+1]} = w_1\theta_1^o + w_2\theta_2^o \tag{5.20}$$

The solutions to this problem are given by:

$$\mathbf{u}_1^*(x_1(k),x_2(k),y_{t,1},\mathbf{u}_2(k)) \quad \mathbf{u}_2^*(x_1(k),x_2(k),y_{t,2},\mathbf{u}_1(k))$$

As in [46], given the current iteration $(\mathbf{u}_1(k), \mathbf{u}_2(k))$ and $(\theta_1(k), \theta_2(k))$, the next iteration is given by:

$$\begin{aligned}(\mathbf{u}_1(k+1), \mathbf{u}_2(k+1)) &= w_1(\mathbf{u}_1^*(x_1(k), x_2(k), y_{t,1}, \mathbf{u}_2(k)), \mathbf{u}_2(k)) \\ &\quad + w_2(\mathbf{u}_1(k), \mathbf{u}_2^*(x_1(k), x_2(k), y_{t,2}, \mathbf{u}_1(k))) \\ &\quad w_1 + w_2 = 1 \quad w_1, w_2 > 0 \end{aligned} \tag{5.21}$$

Remark 1 (Target selection) *At each iteration, each agent solves the global tracking problem, by finding a global θ. This means that each agent has to minimize an offset cost function w.r.t. the overall system. The other ingredients of the MPC for tracking, the invariant set for tracking, is hence calculates as the invariant set for tracking of the centralized problem [28].*

Remark 2 (Stability) *A stability proof has not been obtained yet, and it is one of the works in progress.*

5.4 Example: Application to the 4 tanks system

The presented controller has been tested in simulation on a 4 tanks system model.

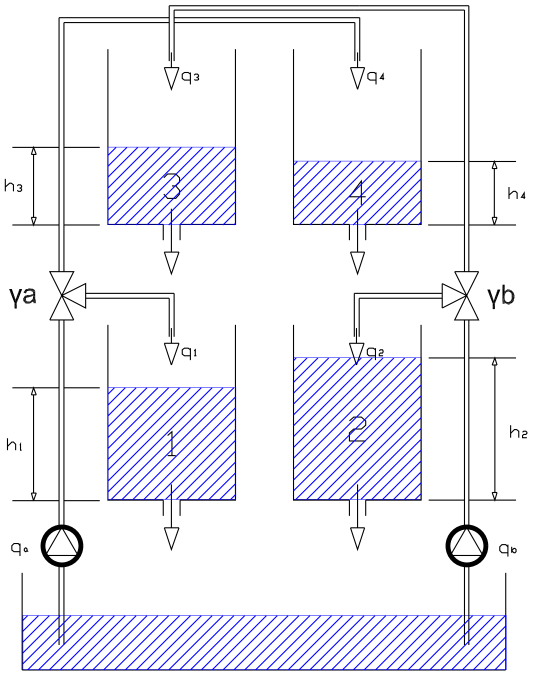
5.4.1 Distributed model

The four tanks plant [23] is a multivariable laboratory plant of interconnected tanks with nonlinear dynamics and subject to state and input constraints. A scheme of this plant is presented in Figure 5.1(a). A real experimental plant developed at the University of Seville [5] is presented in Figure 5.1(b).

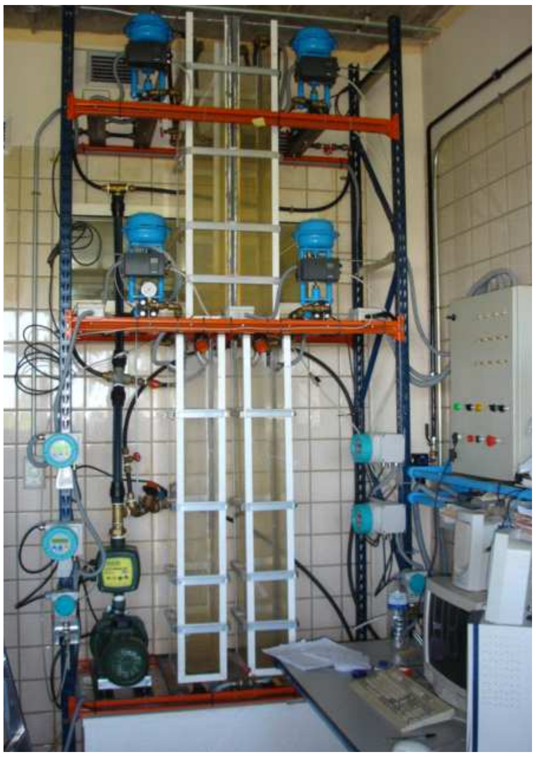
A state space continuous time model of the quadruple tank process system ([23]) can be derived from first principles as follows

$$\begin{aligned}\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_a}{A_1}q_a \\ \frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_b}{A_2}q_b \\ \frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_b)}{A_3}q_b \\ \frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_a)}{A_4}q_a \end{aligned} \tag{5.22}$$

The plant parameters, estimated on the real plant are shown in the following table:



(a) Scheme of the 4 tank process.



(b) The real plant.

Figure 5.1: The 4 tanks process.

	Value	Unit	Description
H_{1max}	1.36	m	Maximum level of the tank 1
H_{2max}	1.36	m	Maximum level of the tank 2
H_{3max}	1.30	m	Maximum level of the tank 3
H_{4max}	1.30	m	Maximum level of the tank 4
H_{min}	0.3	m	Minimum level in all cases
Q_{1max}	2.8	m^3/h	Maximal inflow of tank 1
Q_{2max}	2.45	m^3/h	Maximal inflow of tank 2
Q_{3max}	2.3	m^3/h	Maximal inflow of tank 3
Q_{4max}	2.4	m^3/h	Maximal inflow of tank 4
Q_{min}	0	m^3/h	Minimal inflow in all cases
Q_a^0	1.6429	m^3/h	Equilibrium flow ($Q_1 + Q_4$)
Q_b^0	2.0000	m^3/h	Equilibrium flow ($Q_2 + Q_3$)
a_1	1.341e-4	m^2	Discharge constant of tank 1
a_2	1.533e-4	m^2	Discharge constant of tank 2
a_3	9.322e-5	m^2	Discharge constant of tank 3
a_4	9.061e-5	m^2	Discharge constant of tank 4
A	0.06	m^2	Cross-section of all tanks
γ_a	0.3		Parameter of the 3-ways valve
γ_b	0.4		Parameter of the 3-ways valve
h_1^0	0.627	m	Equilibrium level of tank 1
h_2^0	0.636	m	Equilibrium level of tank 2
h_3^0	0.652	m	Equilibrium level of tank 3
h_4^0	0.633	m	Equilibrium level of tank 4

The minimum level of the tanks has been taken greater than zero to prevent eddy effects in the discharge of the tank. One important property of this plant is that the dynamics present multivariable transmission zeros which can be located in the right hand side of the s plane for some operating conditions. Hence, the values of γ_a and γ_b have been chosen in order to obtain a system with non-minimum phase multivariable zeros.

Linearizing the model at an operating point given by h_i^0 and defining the deviation variables $x_i = h_i - h_i^o$ and $u_j = q_j - q_j^o$ where $j = a, b$ and $i = 1, \cdots, 4$ we have that:

$$\begin{aligned} \frac{dx}{dt} &= \begin{bmatrix} \frac{-1}{\tau_1} & 0 & \frac{A_3}{A_1\tau_3} & 0 \\ 0 & \frac{-1}{\tau_2} & 0 & \frac{A_4}{A_2\tau_4} \\ 0 & 0 & \frac{-1}{\tau_3} & 0 \\ 0 & 0 & 0 & \frac{-1}{\tau_4} \end{bmatrix} x + \begin{bmatrix} \frac{\gamma_a}{A_1} & 0 \\ 0 & \frac{\gamma_b}{A_2} \\ 0 & \frac{(1-\gamma_b)}{A_3} \\ \frac{(1-\gamma_a)}{A_4} & 0 \end{bmatrix} u. \\ y &= x \end{aligned}$$

where $\tau_i = \frac{A_i}{a_i}\sqrt{\frac{2h_i^0}{g}} \geq 0$, $i = 1, \cdots, 4$, are the time constants of each tank. This model has been discretized using the zero-order hold method with a sampling time of 5 seconds.

In order to test de cooperative distributed MPC for tracking presented in the paper, the linear model has been partitioned in two subsystems in such a way that the two subsystems are interconnected through the inputs. The two subsystems model are the following:

$$\begin{aligned} \frac{dx_1}{dt} &= \begin{bmatrix} \frac{-1}{\tau_1} & \frac{A_3}{A_1\tau_3} \\ 0 & \frac{-1}{\tau_3} \end{bmatrix} x_1 + \begin{bmatrix} \frac{\gamma_a}{A_1} \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ \frac{(1-\gamma_b)}{A_3} \end{bmatrix} u_2. \\ \frac{dx_2}{dt} &= \begin{bmatrix} \frac{-1}{\tau_2} & \frac{A_4}{A_2\tau_4} \\ 0 & \frac{-1}{\tau_4} \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ \frac{(1-\gamma_a)}{A_4} \end{bmatrix} u_1 + \begin{bmatrix} \frac{\gamma_b}{A_2} \\ 0 \end{bmatrix} u_2. \end{aligned}$$

where $x_1 = (h_1, h_3)$, $x_2 = (h_2, h_4)$, $u_1 = q_a$ and $u_2 = q_b$.

The overall control objective is to control the level of tanks 1 and 2 while fulfilling the constraints on the levels and on the inputs.

5.4.2 Simulations

The controller has been tested in a simulation with four changes of reference. The starting points for agent 1 and 2 are $y_1 = 0.65$ and $y_2 = 0.65$ respectively. The references used for agent 1 are $Ref_1 = (0.65, 0.5, 0.8, 1.25, 0.3, 0.65)$. The references used for agent 2 are $Ref_2 = (0.65, 0.8, 0.5, 1.25, 0.3, 0.65)$. The controllers' setups are the followings:

Agent 1 $Q_1 = 100I_2$, $R_1 = I_1$, N=3, $\rho_1 = 0.5$, $w_1 = 0.5$.

Agent 2 $Q_2 = 100I_2$, $R_2 = I_1$, N=3, $\rho_2 = 0.5$, $w_2 = 0.5$.

The results of the simulation are plotted in Figures 5.2 and 5.3. In figure 5.2 the levels of tank 1 and tank 2 are plotted. The evolutions of the systems are plotted in solid lines, while the reference and the artificial references are plotted respectively in dotted and dashed lines. The optimal setpoints for the centralized system are plotted in dashed-dotted lines. See how the controller always steers the system to the optimal setpoint of the centralized control. In Figure 5.3 the control actions, which are the flows from the pumps, are plotted.

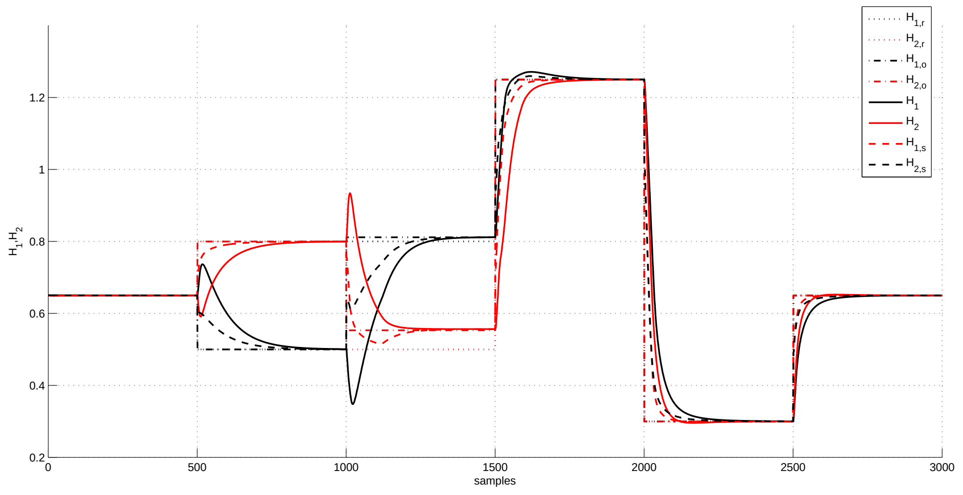


Figure 5.2: Time evolution of the levels.

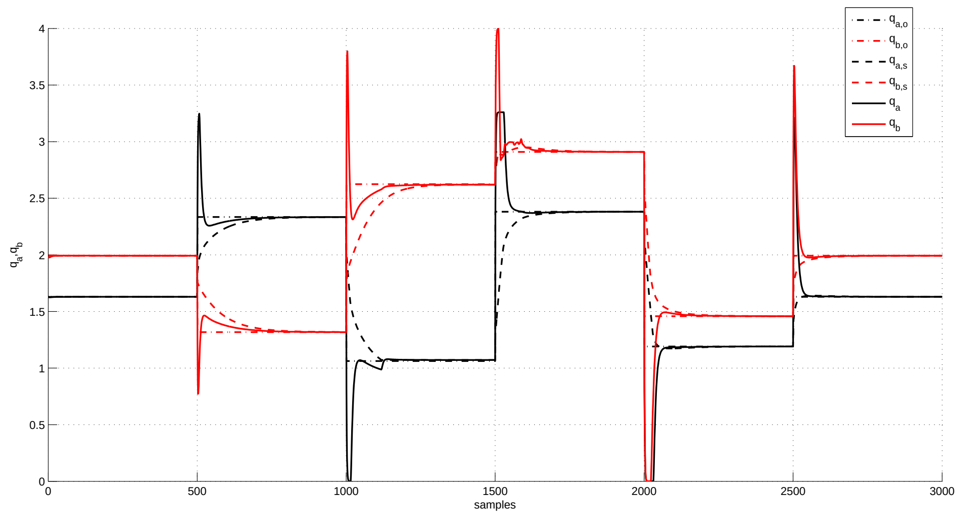


Figure 5.3: Time evolution of the flows.

Chapter 6

Distributed MPC based on Benders' decomposition

The research in this chapter has been developed by P.-D. Moroşan, R. Bourdais, D. Dumur and J. Buisson (SUPELEC).

Abstract

Even if most of the MPC formulations are based on a quadratic cost function, many economical objectives of the controlled processes are expressed in a linear form. For large scale systems, even the linear programming problem can become prohibitive from the computational point of view. In this work we propose a distributed MPC algorithm based on Benders' decomposition, in order to reduce the computational demand of the online optimization by using a network of parallel computing agents. The method can handle both local and global linear constraints but it is particular effective when the number of local constraints is significantly greater than the number of global constraints.

6.1 Introduction

During the last two decades a growing interest has been granted to model predictive control (MPC) due to its ability to handle constraints in an optimal control environment. In MPC, the control input is calculated by solving an optimal control problem (minimization of a cost function) over a given horizon. Only the first element of the open-loop command sequence is applied to the system. At the next instant, a new optimization is performed based on current measurements. The predictive control has been successfully used in many and varied applications [8, 37, 44].

Traditionally, the model predictive cost function has been formulated as a quadratic criterion. A part of the popularity of this type of criterion is due to their mathematical properties: convexity, differentiability, ..., without forgetting about the equivalent linear control law obtained using an unconstrained quadratic cost function and a linear prediction model, which has become a strong opponent of PID controller.

The operation cost of the processes has become important, and nowadays the optimization of the energy consumption has become an important control objective. Usually, the cost is linear dependent on the energy flow. Many control problems can be formulated as minimizing the energy cost while maintaining some parameters (inputs, states or outputs) within some predefined bounds, which leads to a linear programming (LP) problem. Using the linear cost function in MPC is not a novel idea [45]. Even if a linear criterion should be more attractive than a quadratic one, the lack of the analytic solution and the non-smoothness of the objective function makes linear MPC formulations very rarely in the literature. This work presents a distributed method for a class of linear MPC problems for large scale systems.

A practical drawback of the predictive control technique is the computational cost. As a consequence, the action area of MPC has been limited to linear are relatively slow systems. The necessity to solve the optimization problem in real time is especially troublesome for large scale processes. This is why many works has been focused on distributed model predictive control (DMPC) algorithms, where the objective is to decrease the computational demand by a parallelization of the online optimization using multiple local agents. Many DMPC strategies have been proposed in the last ten years [52]. In this research field we can see two axes. The first one focuses on achieving the centralized optimum through the distributed control structure, also called coordination-based approach. Convergence and stability conditions were formulated in a constrained [62] and unconstrained [65] quadratic criterion environment. Usually, the necessary number of iterations needed to achieve the optimum with a certain error cannot be reached within a sample time. For these situations, but not only, different communication-based DMPC strategies have been proposed. Minimizing the local cost function by each agent and iterating the optimization and communication procedure, the control system reaches a Nash equilibrium [27]. In [2], the authors propose a method for the interconnection model type between subsystem in order to optimize a performance index. A DMPC approach based on a cooperative game with three communication cycles per sample time was proposed in [31]. A communication-based distributed strategy for regulating the temperature in a multi-zone building was proposed and compared with the centralized and the decentralized approach in [38]. In order to provide a better rejection of high frequency perturbations, a dynamic prediction horizon DMPC law was presented in [39]. In this paper we propose a distributed predictive control architecture, based on Benders' decomposition technique [6].

6.2 Problem formulation

6.2.1 Benders' decomposition

Benders' decomposition, also known as the dual of Dantzig-Wolfe decomposition [11], uses the block-angular structure of the constraint matrix (see Fig.6.1) in order to parallelize the computation of a linear optimization problem. This decomposition method splits a single large-scale linear programming problem into several independent problems which are coordinated by a single master problem (MP). The optimal solution of the original large-scale problem can be shown to be identical to the solution obtained after a finite number of iterations, solving sequentially the master problem and the subproblems [6], as we will detail in Section 3.

6.2.2 Linear criterion MPC

With either l_1 or l_∞ criterion, we may transform the optimal control problem to a linear program by introducing slack variables. Usually, for large scale systems, the global cost function can be written as the sum of the local objectives:

$$J = \sum_{i=1}^{s} J_i = \sum_{i=1}^{s} \boldsymbol{c}_i'^T \boldsymbol{u}_i',$$

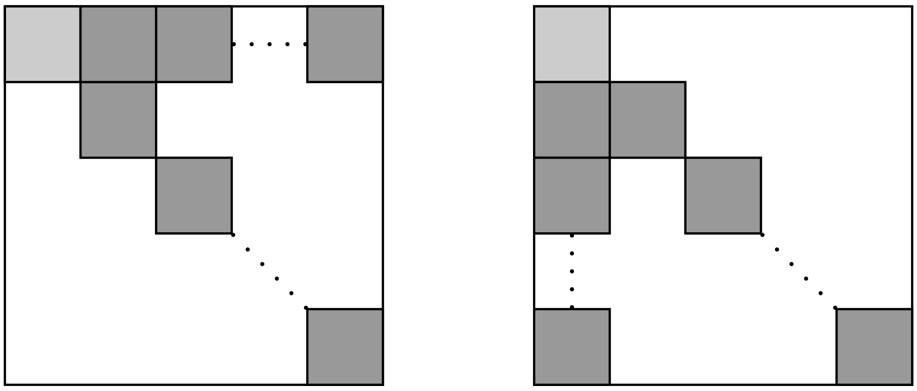


Figure 6.1: Primal block-angular matrix structure (left, favors the Dantzig-Wolfe decomposition method) and dual block-angular matrix structure (right, favors the Benders' decomposition method)

where s is the number of subsystems, $\boldsymbol{u}'_i$ represents the future sequence of the control inputs of subsystem i and eventually the necessary slack variables due to the transformation of the original criterion.

The two block-angular structured matrices presented in Fig.6.1 can be often obtained in the case of large scale systems. The primal block-angular structure corresponds to local independent constraints on each system (the block-diagonal elements) and a global (coupled) constraint, represented by the first block-row of the constraint matrix. The dual block-angular structure can correspond to a case where all the subsystems share a common input, and they have local constraints on states or outputs. Note that if the first block-row, in the case of the primal block-angular constraint matrix (the first block-column, in the case of the primal block-angular constraint matrix) has only the first block element non-zero, then the constraint matrix will have a block-diagonal structure, and the solution of the global problem can be easily computed as the solutions of each local subproblem.

6.3 Decomposition method synthesis

Even if any linear programming problem can be solved applying this decomposition technique, the method is recommended for structured linear programs. For the sake of simplicity, in this section we present the Benders' decomposition for a linear programming problem having a dual block-angular structured constraint matrix, as:

$$\begin{array}{rcccccccc}
\min\limits_{\boldsymbol{u}'_c,\boldsymbol{u}'_i,\forall i\in S} & \boldsymbol{c}'^T_c\boldsymbol{u}'_c & + & \boldsymbol{c}'^T_1\boldsymbol{u}'_1 & \cdots & + & \boldsymbol{c}'^T_s\boldsymbol{u}'_s & & \\
\text{subject to} & \boldsymbol{D}\boldsymbol{u}'_c & & & & & & = & \boldsymbol{g} \\
 & \boldsymbol{E}_1\boldsymbol{u}'_c & + & \boldsymbol{F}_1\boldsymbol{u}'_1 & & & & = & \boldsymbol{h}_1 \\
 & \vdots & & & & \ddots & & & \vdots \\
 & \boldsymbol{E}_s\boldsymbol{u}'_c & & & & + & \boldsymbol{F}_s\boldsymbol{u}'_s & = & \boldsymbol{h}_s \\
 & \boldsymbol{u}'_c & , & \boldsymbol{u}'_1 & \cdots & , & \boldsymbol{u}'_s & \geq & \boldsymbol{0},
\end{array} \tag{6.1}$$

In (6.1), the optimization variable $\boldsymbol{u}'_c$, also called complicating variable, prevents obtaining the optimal solution by solving each subproblem independently. But, for a fixed value of this complicating variable, we know that solving independently the subproblems leads to the global optimum. This is the main idea of this iterative decomposition technique, where at each iteration l the master problem optimal solution $\boldsymbol{u}'^l_c$ tends to the optimal value $\boldsymbol{u}'^*_c$. In order to write explicitly the master problem and the subproblems, we will firstly rewrite the linear programming problem (6.1) as:

$$VAL = \min_{\boldsymbol{u}'_c} \boldsymbol{c}'^T_c \boldsymbol{u}'_c + \underbrace{\sum_{i=1}^{s} z_i(\boldsymbol{u}'_c)}_{z},$$

subject to

$$\boldsymbol{D}\boldsymbol{u}'_c = \boldsymbol{g},\ \boldsymbol{u}'_c \geq \boldsymbol{0},$$

where

$$z_i(\boldsymbol{u}'_c) = \min_{\boldsymbol{u}'_i} \boldsymbol{c}'^T_i \boldsymbol{u}'_i, \forall i \in S, \tag{6.2a}$$

subject to

$$\boldsymbol{F}_{ii}\boldsymbol{u}'_i = \boldsymbol{h}_i - \boldsymbol{E}_i\boldsymbol{u}'_c, \tag{6.2b}$$

$$\boldsymbol{u}'_i \geq \boldsymbol{0}. \tag{6.2c}$$

We call (6.2) the subproblem i, once the complicating variable $\boldsymbol{u}'_c$ has been chosen. Applying the duality, $z_i(\boldsymbol{u}'_c)$ can also be computed through the dual of (6.2), defined as:

$$z_i(\boldsymbol{u}'_c) = \max_{\boldsymbol{p}_i} \boldsymbol{p}_i^T(\boldsymbol{h}_i - \boldsymbol{E}_i\boldsymbol{u}'_c), \tag{6.3a}$$

subject to

$$\boldsymbol{F}_{ii}^T\boldsymbol{p}_i \leq \boldsymbol{c}'_i. \tag{6.3b}$$

The reason of using the dual subproblem is that the polyhedron $\mathcal{D}_i = \left\{\boldsymbol{p}_i \mid \boldsymbol{F}_{ii}^T\boldsymbol{p}_i \leq \boldsymbol{c}'_i\right\}$ that defines the feasible region of (6.3) is independent of $\boldsymbol{u}'_c$. The solution of (6.3) is an extreme point of $\mathcal{D}_i$ due to the fact that (6.3) is always feasible and bounded, which is a consequence of the feasibility and boundedness of the primal problem, by its definition. Using an algorithm to solve exactly the dual subproblems (6.3), it will return one of the extreme points $\boldsymbol{p}_i^k$, $k = 1...I_i$ of the feasible region $\mathcal{D}_i$ and the subproblem objective function can be expressed as:

$$z_i(\boldsymbol{u}'_c) = \overline{\boldsymbol{p}}_i^T(\boldsymbol{h}_i - \boldsymbol{E}_i\boldsymbol{u}'_c) = \max_{k=1,\dots,I_i} (\boldsymbol{p}_i^k)^T(\boldsymbol{h}_i - \boldsymbol{E}_i\boldsymbol{u}'_c).$$

Now we are able to write the MP at iteration l, knowing the solutions of all dual subproblems at every previous iteration $l_p < l$:

$$\min_{\boldsymbol{u}'_c, z} \boldsymbol{c}'^T_c \boldsymbol{u}'_c + z, \tag{6.4a}$$

subject to

$$\boldsymbol{D}\boldsymbol{u}'_c = \boldsymbol{g},\ \boldsymbol{u}'_c \geq \boldsymbol{0},\ z \geq 0, \tag{6.4b}$$

$$\sum_{i=1}^{s} (\overline{\boldsymbol{p}}_i^{l_p})^T \boldsymbol{E}_i \boldsymbol{u}'_c + z \geq \sum_{i=1}^{s} (\overline{\boldsymbol{p}}_i^{l_p})^T \boldsymbol{h}_i,\ \forall l_p = 1...l-1, \tag{6.4c}$$

where $\overline{\boldsymbol{p}}_i^{l_p}$ denotes the solution of the dual subproblem i computed at iteration l_p.

The algorithm that solves (6.1) using the Benders' decomposition is iterative. Each iteration will have two main steps, consisting of solving the master problem and then solving (in parallel) the subproblems using the current value of the complicating variable. We add the new constraint (6.4c) to the master problem (also called Benders' cut) and a new iteration can begin. Note that over the iterations, the constraint matrix dimensions of subproblems remains constant while the constraint matrix of MP increases (by one line) after each iteration. The algorithm reaches the optimal solution when $z^l = \sum_{i=1}^{s} z_i^l$. In practice at each iteration l an upper and a lower bound of the optimum are computed as:

$$VAL_{up}^l = \boldsymbol{c}_c'^T \boldsymbol{u}_c'^l + \sum_{i=1}^{s} \underbrace{\boldsymbol{c}_i'^T \boldsymbol{u}_i'^l}_{z_i^l},\; VAL_{low}^l = \boldsymbol{c}_c'^T \boldsymbol{u}_c'^l + z^l \tag{6.5}$$

and the stop condition will be $VAL_{up}^l - VAL_{low}^l \leq \varepsilon$. Algorithm 1 summarizes the decomposition method.

Algorithm 1 DMPC procedure for multi-source multi-zone heating system based on Benders' decomposition

```
Require: Problem formulation (6.1), $\varepsilon, \boldsymbol{x}_i$, $\forall i \in S$
Ensure: Optimal control input sequences $\boldsymbol{u}_c'^*$ and $\boldsymbol{u}_i'^*$, $i \in S$
 1: Initialization: $l = 1$
 2: MPC$_c$ solves the MP (6.4), obtaining $\boldsymbol{u}_c'^l$ and $z^l$
 3: MPC$_c$ broadcasts $\boldsymbol{u}_c'^l$ to local controllers MPC$_i$, $\forall i \in S$
 4: All MPC$_i$ solve (in parallel) the subproblems (in both primal (6.2) and dual (6.3) forms) and send
    the results, $\boldsymbol{u}_i'^l$ and $\overline{\boldsymbol{p}}_i^l$, to MPC$_c$
 5: Compute the current criterion bounds (6.5)
 6: if $VAL_{up}^l - VAL_{low}^l \leq \varepsilon$ (and $l \leq l_{max}$) then
 7:     $\boldsymbol{u}_c'^* = \boldsymbol{u}_c'^l$ and $\boldsymbol{u}_i'^* = \boldsymbol{u}_i'^l$, $\forall i \in S$, Stop
 8: else
 9:     Update the constraints of the MP (by adding the new Benders' cut), $l = l + 1$ and Goto step 2
10: end if
```

The procedure presented in Algorithm 1 is executed at each sample time by the control structure. The central controller, MPC$_c$, acts like a coordinator for the local controllers, MPC$_i$. It also tests the stop condition of the algorithm after every iteration. The local predictive controllers solve their subproblems once they have received from the master the current value of the complicating variable.

6.4 Simulation

6.4.1 Multi-source temperature control in buildings

To show the performances of the proposed distributed MPC scheme we consider now the temperature control problem in multi-zone buildings. In order to reduce the energy costs, many buildings are equipped with several heating sources with different dynamics, gains and energy prices, an example is the use of a hot water based central heating and local electric convectors as a complementary heating source. This is the case that we will consider in the followings. Using a linear state space model describing the thermal behavior, we can write the local model of zone i as:

$$\begin{cases} \boldsymbol{x}_i(k+1) = \boldsymbol{A}_i \boldsymbol{x}_i(k) + \begin{bmatrix} \boldsymbol{B}_{i1} & \boldsymbol{B}_{i2} \end{bmatrix} \cdot \begin{bmatrix} u_i(k) \\ u_c(k) \end{bmatrix} \\ y_i(k) = \boldsymbol{C}_i \boldsymbol{x}_i(k), \end{cases} \tag{6.6}$$

where the vector $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$ is the local state, $u_i, u_c, y_i \in \mathbb{R}$ are the local and the shared input (electrical heating power and power input of the boiler) and the output (measured room temperature), respectively.

The control objective is to minimize the energy bill due to the indoor heating, which is usually a linear function of the consumed energy (our control inputs). The thermal comfort (defined here by an upper $\overline{w}_i(k+j)$ and a lower $\underline{w}_i(k+j)$ temperature bound) and the physical limitations of the process are the constraints of our optimization problem. The thermal comfort is defined only during the occupation periods (see Fig. 6.2). So denote $\boldsymbol{\delta}_i(k) = \left[\delta_i(k+1) \quad \cdots \quad \delta_i(k+N_2)\right]^T$ as the future occupation profile over the prediction horizon for the room i at time step k. Intuitively, each element of this vector is defined as:

$$\delta_i(k+j) = \begin{cases} 1, \, k+j \in \text{Occupation}_i \\ 0, \, k+j \in \text{Inoccupation}_i. \end{cases} \tag{6.7}$$

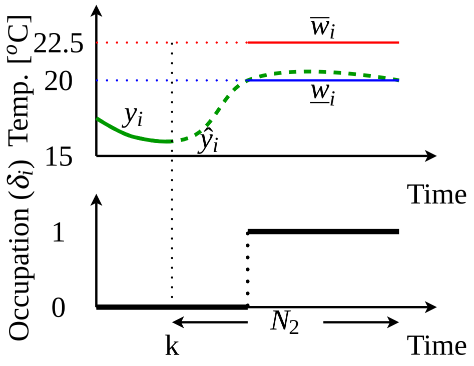


Figure 6.2: Occupation profile illustration

If, at a certain time step, one of the thermal comfort constraints cannot be satisfied, the global optimization problem becomes infeasible. In practice, to avoid this issue the comfort constraints are softened, we add a penalty, f_i, to the cost criterion if they are not satisfied. Then, the minimization problem becomes:

$$\min_{\boldsymbol{u}_c(k),\boldsymbol{u}_i(k),\forall i\in S} J(k) = \sum_{i\in S}\left(\boldsymbol{c}_i^T(k)\boldsymbol{u}_i(k) + \sum_{j=1}^{N_2} f_i(k+j)\right) + \boldsymbol{c}_c^T(k)\boldsymbol{u}_c(k), \tag{6.8a}$$

subject to

$$0 \le u_i(k+j) \le \bar{u}_i, \; \forall j = 0 \ldots N_u - 1, \; \forall i \in S, \tag{6.8b}$$

$$0 \le u_c(k+j) \le \bar{u}_c, \; \forall j = 0 \ldots N_u - 1, \tag{6.8c}$$

where the comfort penalty function, f_i, is defined as:

$$f_i(k+j) = \begin{cases} 0, \; \mu_i^{up}(k+j) \le 0 \text{ and } \mu_i^{low}(k+j) \le 0 \\ \lambda_i \mu_i^{up}(k+j), \; \mu_i^{up}(k+j) > 0 \\ \lambda_i \mu_i^{low}(k+j), \; \mu_i^{low}(k+j) > 0, \end{cases}$$

with

$$\begin{aligned} \mu_i^{low}(k+j) &= \delta_i(k+j)(\underline{w}_i(k+j) - \hat{y}_i(k+j|k)), \\ \mu_i^{up}(k+j) &= \delta_i(k+j)(\hat{y}_i(k+j|k) - \overline{w}_i(k+j)), \\ &\forall i \in S, \; \forall j = 1 \ldots N_2, \end{aligned}$$

$$\boldsymbol{c}_i(k) = \left[c_i(k) \quad \cdots \quad c_i(k+N_u-2) \quad \sum_{j=N_u-1}^{N_2-1} c_i(k+j) \right]^T,$$
$$\boldsymbol{c}_c(k) = \left[c_c(k) \quad \cdots \quad c_c(k+N_u-2) \quad \sum_{j=N_u-1}^{N_2-1} c_c(k+j) \right]^T.$$

where $c_i(k+j)$ and $c_c(k+j)$ are the energy prices for local and shared inputs, respectively, at time step $k+j$. The sum, defining the last element of the cost sequences, is due to the fact that the control input is considered constant out of the control horizon and equal to the value corresponding to the time step $k+N_u-1$. λ_i is the weighting factor that penalizes the criterion when the comfort constraints are not accomplished.

The linear programming problem (6.8) can be written in a standard form, with a dual block-diagonal constraint matrix, as in (6.1), with the following notations:

$$\boldsymbol{c}_i'^T = \left[\boldsymbol{c}_i^T \quad \mathbf{0}_{1,N_u} \quad \mathbf{0}_{1,N_2} \quad \mathbf{0}_{1,N_2} \quad \lambda_i \mathbf{1}_{1,N_2} \quad \lambda_i \mathbf{1}_{1,N_2} \right],$$
$$\boldsymbol{c}_c'^T = \left[\boldsymbol{c}_c^T \quad \mathbf{0}_{1,N_u} \right], \boldsymbol{D} = \left[\boldsymbol{I}_{N_u,N_u} \quad \boldsymbol{I}_{N_u,N_u} \right], \boldsymbol{g} = \left[\overline{\boldsymbol{u}}_c \right],$$
$$\boldsymbol{\Phi}_{ij} = \begin{bmatrix} \phi_{ij}^0 & 0 & \cdots & 0 \\ \phi_{ij}^1 & \phi_{ij}^0 & 0 & \cdots \\ \vdots & \cdots & \ddots & \vdots \\ \phi_{ij}^{N_2-1} & \cdots & \phi_{ij}^{N_2-N_u+1} & \sum_{k=0}^{N_2-N_u} \phi_{ij}^k \end{bmatrix},$$
$$\phi_{ij}^k = \mathbf{C}_i \mathbf{A}_i^k \mathbf{B}_{ij}, \boldsymbol{\Psi}_i = \left[(\mathbf{C}_i \mathbf{A}_i^1)^T \quad \cdots \quad (\mathbf{C}_i \mathbf{A}_i^{N_2})^T \right]^T,$$
$$\boldsymbol{E}_i = \begin{bmatrix} \mathbf{0}_{N_u,N_u} & \mathbf{0}_{N_u,N_u} \\ -\boldsymbol{\Delta}_i \boldsymbol{\Phi}_{i2} & \mathbf{0}_{N_o^i,N_u} \\ \boldsymbol{\Delta}_i \boldsymbol{\Phi}_{i2} & \mathbf{0}_{N_o^i,N_u} \end{bmatrix}, \boldsymbol{h}_i = \begin{bmatrix} \overline{\boldsymbol{u}}_i \\ -\boldsymbol{\Delta}_i \left(\underline{\boldsymbol{w}}_i - \boldsymbol{\Psi}_i \boldsymbol{x}_i \right) \\ \boldsymbol{\Delta}_i \left(\overline{\boldsymbol{w}}_i - \boldsymbol{\Psi}_i \boldsymbol{x}_i \right) \end{bmatrix},$$
$$\boldsymbol{F}_i = \begin{bmatrix} \boldsymbol{I}_{N_u,N_u} & \boldsymbol{I}_{N_u,N_u} & \mathbf{0}_{N_u,N_o^i} & \mathbf{0}_{N_u,N_o^i} & \mathbf{0}_{N_u,N_o^i} & \mathbf{0}_{N_u,N_o^i} \\ -\boldsymbol{\Delta}_i \boldsymbol{\Phi}_{i1} & \mathbf{0}_{N_o^i,N_u} & \boldsymbol{I}_{N_o^i,N_o^i} & \mathbf{0}_{N_o^i,N_o^i} & -\boldsymbol{I}_{N_o^i,N_o^i} & \mathbf{0}_{N_o^i,N_o^i} \\ \boldsymbol{\Delta}_i \boldsymbol{\Phi}_{i1} & \mathbf{0}_{N_o^i,N_u} & \mathbf{0}_{N_o^i,N_o^i} & \boldsymbol{I}_{N_o^i,N_o^i} & \mathbf{0}_{N_o^i,N_o^i} & -\boldsymbol{I}_{N_o^i,N_o^i} \end{bmatrix}.$$

The optimization variables $\boldsymbol{u}_c'$ and $\boldsymbol{u}_i'$ are obtained by adding the required number of slack variables to the $\boldsymbol{u}_c$ and $\boldsymbol{u}_i$, respectively.

6.4.2 Empirical study of efficiency

Since the distributed algorithm was implemented on a sequential machine, the computational time required by the decomposition algorithm to solve a linear programming problem is:

$$t_{seq} = \sum_{l=1}^{\bar{l}} \left(t_{MPC_c}(l) + \sum_{j \in S} t_{MPC_j}(l) \right),$$

where $\bar{l}$ is the number of Benders iterations needed to solve the problem, $t_{MPC_c}(l)$ is the time required by the central MPC to solve the master problem at iteration l, while $t_{MPC_j}(l)$ is the computational time to solve the subproblem j. The computational time using a distributed computing environment, ignoring the communication time required at each iteration, can be expressed as:

$$t_{distr} = \sum_{l=1}^{\bar{l}} \left(t_{MPC_c}(l) + \max_{j \in S} t_{MPC_j}(l) \right).$$

In order to study the complexity of the distributed algorithm compared to the centralized solver, we will use a very simple thermal model of a room and supposing (without loss of generality) that all the subsystems have the same model. The values of the model (6.6) matrices are:

$$\boldsymbol{A}_i = \begin{bmatrix} 0.9921 & 0 \\ 0 & 0.9931 \end{bmatrix}, \boldsymbol{B}_i = \begin{bmatrix} 0.2595 & 0 \\ 0 & 0.1376 \end{bmatrix}, \boldsymbol{C}_i^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The efficiency of the distributed algorithm will be measured regarding the equivalent distributed computational time and the number of iterations $\bar{l}$. The main parameters of the algorithm are: the number of subproblems (subsystems) s, the dimension of the subproblems $d_i = (N_u + 2N_o^i) \times (2N_u + 4N_o^i)$, where $N_o^i \in \{0, 1, \dots, N_2\}$ is the number of occupation time steps within the prediction period (i.e. the number of lines of $\boldsymbol{\Delta}_i$) and the tolerance ε. The dimension of the prediction horizon N_2 should be chosen sufficiently large in order to offer enough time to the heating system to increase the indoor temperature up to the desired setpoint in the worst situation (low initial temperatures). In the following simulation results, we used $N_2 = 30$. For the three scenarios presented below we considered five different cases (in each case we changed the initial state values of the subsystems) for each value of the variable parameter, in order to have more consistent statistical results.

Scenario 1

$s \in \{2^1, 2^2, \dots, 2^7\}$, $N_u = 5$, $N_o^i = 15$, $\varepsilon = 10^{-3}$. Fig. 6.3 shows a very good scaling behavior of the algorithm in a distributed computing environment, regarding the number of subproblems. In the mean time, for a small number of subsystems the centralized (Simplex) method offers better performances. Concerning the number of iterations, we observe a logarithmic dependence of $\bar{l}$ on s. This fact shows a good convergence speed of the algorithm and its slight dependence on the number of subproblems.

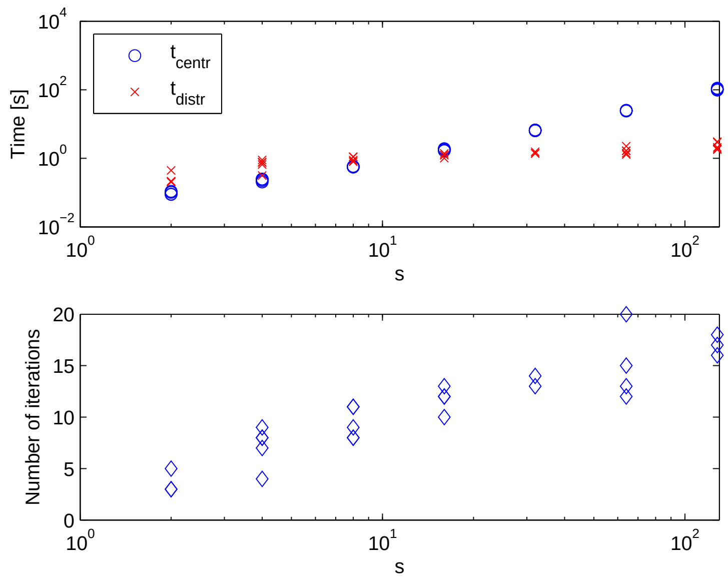


Figure 6.3: The influence of the number of subsystems, s

Scenario 2

$s = 50$, $N_u \in \{1, 5, 10, \ldots, 30\}$, $N_o^i \in \{1, 5, 10, \ldots, 30\}$, $\varepsilon = 10^{-3}$. Here we study the algorithm performances with respect to the subproblem sizes, d_i. The control horizon dimension, N_u, is a tuning parameter, while N_o^i depends on the occupation profiles. As Fig. 6.4 shows, N_o^i has a more important influence over the computational time than N_u, which is normal as N_o^i has a greater weight on the dimension of the subproblem.

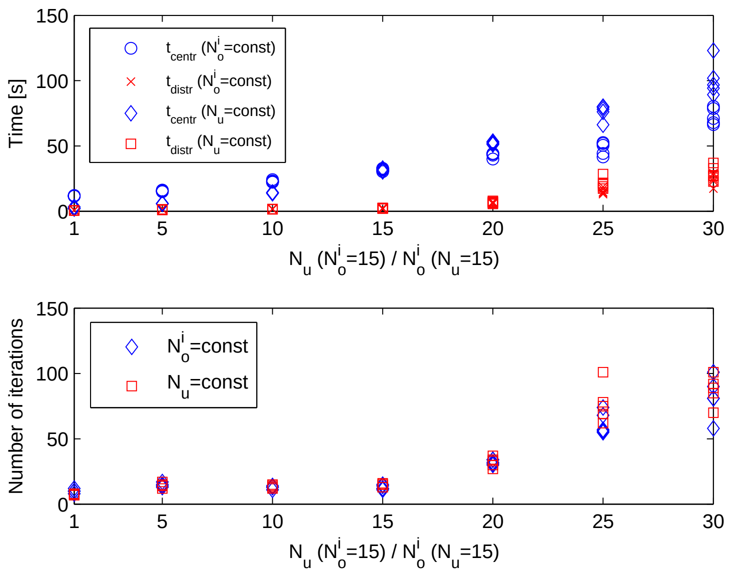


Figure 6.4: The influence of the control horizon, N_u, and of the occupation time steps, N_o^i

Scenario 3

$s = 50$, $N_u = 10$, $N_o^i = 15$, $\varepsilon \in \{10^{-6}, 10^{-5}, \ldots, 10^{-1}\}$. Fig. 6.5 shows that the tolerance ε influences very slightly the computational demand of the distributed algorithm. The number of Benders iterations has a logarithmic dependence on the tolerance which shows the exponential convergence of the method.

6.5 Conclusion

A distributed model predictive control strategy has been proposed for solving a class of LP large scale problems. The distributed control strategy is based on Benders' decomposition, which allows the decrease of the computational demand by using a network of local controllers, coordinated by a master controller. This decomposition method is very effective when the constraint matrix has a specific block structure (dual block-angular). The effective of the distributed algorithm, regarding the computational time, has been shown using a simple multi-source temperature control in buildings.

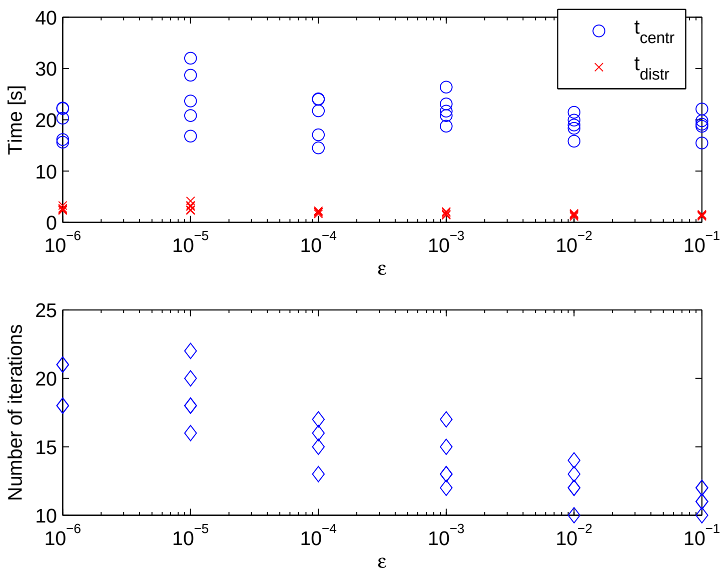


Figure 6.5: The influence of the stop condition tolerance, ε

Chapter 7

Infinite Horizon Model Predictive Control with targets and zone control

Abstract

This work deals with an Infinite Horizon MPC (IHMPC) developed for the zone control and input target. The IHMPC with zone control and input targets includes on the addition of an economic stage in order to compute the desired values of the manipulated variables based on an economic objective. The computed values of the manipulated variables on the economic stage are sent to the controller (IHMPC controllers are assumed in this work), and the objective of the controller is to drive the manipulated variables, to their desired values, keeping the outputs of the system within a predefined zone (range of values).

7.1 Infinite Horizon Model Predictive Control

Consider the nonlinear system given by

$$\begin{aligned} \dot{x}(t) &= f(x(t),u(t)) \\ y(t) &= g(x(t),u(t)) \end{aligned} \tag{7.1}$$

where $f(\cdot)$, $g(\cdot)$ are smooth C^1 functions, $x \in \Re^{n_x}$, $u \in \Re^{n_u}$, and $y \in \Re^{n_y}$ denote the states, the inputs, and the outputs of the dynamical system (7.1).
In order to design a model predictive controller for the system (7.1),a linear time invariant model (7.2) is considered. The construction of the linear time invariant model in the state-space representation can be conducted by linearizing the system at $[x^*,u^*]$:

$$\begin{aligned} x(k+1) &= E + A(x(k)-x^*) + B(u(k)-u^*) \\ y(k) &= F + C(x(k)-x^*) + D(u(k)-u^*) \end{aligned} \tag{7.2}$$

where

$$
\begin{aligned}
E &= f(x^*,u^*) \\
F &= g(x^*,u^*) \\
A &= \left.\frac{\partial f(x,u)}{\partial x}\right|_{x^*,u^*} \\
B &= \left.\frac{\partial f(x,u)}{\partial u}\right|_{x^*,u^*} \\
C &= \left.\frac{\partial g(x,u)}{\partial x}\right|_{x^*,u^*} \\
D &= \left.\frac{\partial g(x,u)}{\partial u}\right|_{x^*,u^*}
\end{aligned}
$$

Often, the output of the system $y(k)$ is independent of the inputs $u(k)$ (i.e., $D = 0$). In this work we also assume the same. However, the approach presented here can be easily extended to systems in which $D \neq 0$.

A modeling approach frequently adopted in model predictive controller (MPC) considers a discrete-time state -space model in incremental form [49],

$$
\begin{aligned}
\tilde{x}(k+1) &= \tilde{A}\tilde{x}(k) + \tilde{B}\Delta u(k) \\
y(k) &= \tilde{C}\tilde{x}(k)
\end{aligned}
\tag{7.3}
$$

where $\tilde{A}$, $\tilde{B}$, $\tilde{C}$, are matrices of the system in the so called "incremental form", and $\Delta u_k = u_k - u_{k-1}$ is the input increment.
The model (7.3) can be represented in Jordan canonical form as:

$$
\begin{aligned}
\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} &= \begin{bmatrix} P & 0 \\ 0 & I_{ny} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \Delta u(k) \\
y(k) &= \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}
\end{aligned}
\tag{7.4}
$$

where $x_1(k) = V_1\tilde{x}(k),\ \ x_2(k) = V_2\tilde{x}(k-1)$, V_1, V_2 are transformation matrices, P is a block diagonal matrix with components corresponding to the poles of the system, and I_{ny} is a identity matrix of size ny. In the state equation defined in (7.4), the state component $x_2(k)$ corresponds to the integrating poles produced by the incremental form of the model, and $x_1(k)$ corresponds to the system modes. For stable systems, it is easy to show that if the system approaches to the steady state, x_1 tends to zero. Based on (7.3), the cost function of the output-tracking problem for the infinite horizon MPC (IHMPC) can be defined as follows [49]:

$$
J_{k,\infty} = \sum_{j=1}^{\infty} e(k+j|k)^T Q e(k+j|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) \tag{7.5}
$$

where $e(k+j|k) = y(k+j|k) - r(j)$; $y(k+j|k)$ is the output prediction at time instant $k+j$ made at time k; $r(j)$ is the desired output value at time j, N_c is the control horizon, $Q \in \Re^{n_y \times n_y}$ and $R \in \Re^{n_u \times n_u}$

are positive definite weighting matrices. The controller based on the minimization of the above cost function corresponds to the IHMPC for the output-tracking case. Often, the infinite horizon controllers reduce to finite horizon controllers by defining a terminal state penalty $\overline{Q}$. For the cost defined in (7.5) such a terminal penalty is computed by the following Lyapunov equation :

$$\overline{Q} - P^T\overline{Q}P = P^T {C_1}^T QC_1P \tag{7.6}$$

Since an infinite horizon is used and the model defined in (7.4) has integrating modes, terminal constraints must be added. According to [49], these constraints can be written as follows:

$$\overline{C}_2x_2(k) - r + \overline{C}_2\tilde{B}_2\Delta u_k = 0 \tag{7.7}$$

where

$$\tilde{B}_2 = \left[\begin{array}{c} B_2, \cdots, B_2 \end{array}\right] \in \Re^{n_u \times N_c \cdot n_x}$$

$$\overline{C}_2 = \mathrm{diag}\,[C_2, \cdots, C_2] \in \Re^{N_c \cdot n_y \times N_c \cdot n_x}$$

With the terminal penalty $\overline{Q}$, the cost defined in (7.5) can be written as

$$J_{k,\infty} = \sum_{j=1}^{N_c-1} e(k+j|k)^T Qe(k+j|k) + x_1(k+N_c|k)^T\overline{Q}x_1(k+N_c|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R\Delta u(k+j|k) \tag{7.8}$$

Finally, the control optimization problem of the IHMPC can be formulated as:

$$\min_{\Delta u_k} J_{k,\infty} = \sum_{j=1}^{N_c} e(k+j|k)^T Qe(k+j|k) + x_1(k+N_c|k)^T\overline{Q}x_1(k+N_c|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R\Delta u(k+j|k) \tag{7.9}$$

subject to

$$\begin{gathered}
\left[\begin{array}{c} x_1(k+1) \\ x_2(k+1) \end{array}\right] = \left[\begin{array}{cc} P & 0 \\ 0 & I_{ny} \end{array}\right] \left[\begin{array}{c} x_1(k) \\ x_2(k) \end{array}\right] + \left[\begin{array}{c} B_1 \\ B_2 \end{array}\right] \Delta u(k) \\
y(k) = \left[\begin{array}{cc} C_1 & C_2 \end{array}\right] \left[\begin{array}{c} x_1(k) \\ x_2(k) \end{array}\right] \\
\overline{C}_2x_2(k) - r + \overline{C}_2\tilde{B}_2\Delta u_k = 0 \\
-\Delta u^{max} \leq \Delta u(k+j|k) \leq \Delta u^{max} \\
\Delta u(k+j|k) = 0 \;\; ; \;\; j \geq m \\
u^{min} \leq u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \leq u^{max}; \;\; j = 0, 1, ..., m-1
\end{gathered} \tag{7.10}$$

Using model equation (7.4) to represent the output prediction as a function of the future control actions and the current state, the control objective represented in (7.9) can be written as follows:

$$J_{k,\infty} = \tfrac{1}{2}\Delta u_k^T H \Delta u_k + c_f^T \Delta u_k + c \tag{7.11}$$

where

$$H = (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u})^T \overline{Q}_1 (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u}) + \overline{R} + F_{1u}^T \overline{Q}_2 F_{1u}$$

$$c_f = (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u})^T \overline{Q}_1 (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r}) + F_{1u}^T \overline{Q}_2 (\overline{C}_1 F_x x_1(k))$$

$$c = (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r})^T \overline{Q}_1 (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r}) + (\overline{C}_1 F_x x_1(k))^T \overline{Q}_2 (\overline{C}_1 F_x x_1(k))$$

$$\bar{I} = \begin{bmatrix} I_{ny} \\ \vdots \\ I_{ny} \end{bmatrix}, F_x = \begin{bmatrix} P \\ \vdots \\ P^{N_c} \end{bmatrix}, F_{1u} = \begin{bmatrix} B_1 & 0 & \cdots & 0 \\ PB_1 & B_1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ P^{N_c-1}B_1 & P^{N_c-2}B_1 & \dots & B_1 \end{bmatrix}$$

$$F_{2u} = \begin{bmatrix} B_2 & 0 & \cdots & 0 \\ B_2 & B_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ B_2 & B_2 & \dots & B_2 \end{bmatrix}, \Delta u_k = \begin{bmatrix} \Delta u(k|k) \\ \vdots \\ \Delta u(k+N_c-1|k) \end{bmatrix}, \bar{r} = \begin{bmatrix} r \\ \vdots \\ r \end{bmatrix}$$

$$\overline{C}_1 = \mathrm{diag}\,[C_1, \cdots, C_1], \overline{C}_2 = \mathrm{diag}\,[C_2, \cdots, C_2], \overline{Q}_1 = \mathrm{diag}\,[Q, \cdots, Q], \overline{Q}_2 = \mathrm{diag}\,[0, \cdots, Q_2]$$

Finally the control optimization problem can be formulated as:

$$\min_{\Delta u_k} J_{k,\infty} = \tfrac{1}{2} \Delta u_k^T H \Delta u_k + c_f^T \Delta u_k + c \tag{7.12}$$

subject to

$$\overline{C}_2 x_2(k) - r + \overline{C}_2 \tilde{B}_2 \Delta u_k = 0$$

$$-\Delta u^{max} \le \Delta u(k+j|k) \le \Delta u^{max}$$

$$\Delta u(k+j|k) = 0 \;\; ; \;\; j \ge m$$

$$u^{min} \le u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \le u^{max}; \;\; j = 0, 1, \dots, m-1$$

Note that for large changes on $x_2(k)$, or for large changes on $r = [r^T(1), \dots, r^T(N_c-1)]^T$, or if r corresponds to an unreachable steady state, the optimization problem defined through 7.9-7.10 may become infeasible because of a conflict between constraints. Consequently, the IHMPC as defined above cannot be implemented in practice.

7.2 Extended Infinite Horizon Model Predictive Control

The results presented in this section are based on [42].

In order to design an IHMPC which is implementable in practice, the objective function of infinite horizon MPC is re-defined as follows:

$$J_{k,\infty} = \sum_{j=1}^{\infty} (e(k+j|k) - \delta_k)^T Q (e(k+j|k) - \delta_k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k \tag{7.13}$$

where $\delta_k \in \Re^{n_y}$ is a vector of slack variables and $S \in \Re^{n_y \times n_y}$ is a assumed positive definite. Observe that each slack variable refers to a given controlled output. Weight matrix S should be selected such that controller pulls to zero the slacks or at least minimize them depending on the number of inputs, which are not constrained.

Analogously to the IHMPC, the extended infinite horizon controllers reduce to finite horizon controllers by defining a terminal state penalty $\overline{Q}$ that is obtained by solving (7.6). Hence, the control objective defined in (7.13) can be written as

$$J_{k,\infty} = \sum_{j=1}^{N_c} (e(k+j|k) - \delta_k)^T Q (e(k+j|k) - \delta_k) + x_1(k+N_c|k)^T \overline{Q} x_1(k+N_c|k)$$

$$+ \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k$$

Since the prediction model has integrating states, terminal constraints must be added. Such constraint can be written as follows:

$$\overline{C}_2 x_2(k) - r + \overline{C}_2 \tilde{B}_2 \Delta u_k - \delta_k = 0 \tag{7.14}$$

Finally, the control optimization problem for the extended IHMPC can be formulated as:

$$\min_{\Delta u_k, \delta_k} J_{k,\infty} = \sum_{j=1}^{N_c} (e(k+j|k) - \delta_k)^T Q (e(k+j|k) - \delta_k) + x_1(k+N_c|k)^T \overline{Q} x_1(k+N_c|k)$$

$$+ \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k \tag{7.15}$$

subject to

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} P & 0 \\ 0 & I_{ny} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \Delta u(k)$$

$$y(k) = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

$$\overline{C}_2 x_2(k) - r + \overline{C}_2 \tilde{B}_2 \Delta u_k - \delta_k = 0$$

$$-\Delta u^{max} \leq \Delta u(k+j|k) \leq \Delta u^{max}$$

$$\Delta u(k+j|k) = 0 \;\; ; \;\; j \geq m$$

$$u^{min} \leq u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \leq u^{max}; \;\; j = 0, 1, \ldots, m-1$$

Using model equation 7.4 to represent the output prediction as a function of the future control actions and as a function of the current state, the control objective represented in 7.9 can be written as

$$J_{k,\infty} = \tfrac{1}{2} \begin{bmatrix} \Delta u_k^T & \delta_k^T \end{bmatrix} H \begin{bmatrix} \Delta u_k \\ \delta_k \end{bmatrix} + c_f^T \begin{bmatrix} \Delta u_k \\ \delta_k \end{bmatrix} + c \tag{7.16}$$

where

$$H=\begin{bmatrix} (\overline{C}_1F_{1u}+\overline{C}_2F_{2u})^T\overline{Q}_1(\overline{C}_1F_{1u}+\overline{C}_2F_{2u})+\overline{R}+F_{1u}^T\overline{Q}_2F_{1u} & -(\overline{C}_1F_{1u}+\overline{C}_2F_{2u})^T\overline{Q}_1\overline{I} \\ -\overline{IQ}_1(\overline{C}_1F_{1u}+\overline{C}_2F_{2u}) & S+\overline{I}^T\overline{Q}_1+\overline{I}+Q \end{bmatrix}$$

$$c_f=\begin{bmatrix} (\overline{C}_1F_{1u}+\overline{C}_2F_{2u})^T\overline{Q}_1(\overline{C}_1F_xx_1(k)+\overline{C}_2\overline{I}x_2(k)-\overline{r})+F_{1u}^T\overline{Q}_2(\overline{C}_1F_xx_1(k)) \\ -\overline{IQ}_1(\overline{C}_1F_xx_1(k)+\overline{C}_2\overline{I}x_2(k)-\overline{r})-Q(\overline{C}_2\overline{I}x_2(k)-\overline{r}) \end{bmatrix}$$

$$c=(\overline{C}_1F_xx_1(k)+\overline{C}_2\overline{I}x_2(k)-\overline{r})^T\overline{Q}_1(\overline{C}_1F_xx_1(k)+\overline{C}_2\overline{I}x_2(k)-\overline{r})+(\overline{C}_1F_xx_1(k))^T\overline{Q}_2(\overline{C}_1F_xx_1(k))$$

Finally the control optimization problem becomes:

$$\min_{\Delta u_k,\delta_k} J_{k,\infty}=\frac{1}{2}\begin{bmatrix}\Delta u_k^T & \delta_k^T\end{bmatrix}H\begin{bmatrix}\Delta u_k \\ \delta_k\end{bmatrix}+c_f^T\begin{bmatrix}\Delta u_k \\ \delta_k\end{bmatrix}+c \tag{7.17}$$

subject to

$$\overline{C}_2x_2(k)-r+\overline{C}_2\tilde{B}_2\Delta u_k-\delta_k=0$$

$$-\Delta u^{max}\leq\Delta u(k+j|k)\leq\Delta u^{max}$$

$$\Delta u(k+j|k)=0\ ;\ \ j\geq m$$

$$u^{min}\leq u_{k-1}+\sum_{i=0}^{j}\Delta u_{k+i}\leq u^{max};\ \ j=0,1,\ldots,m-1$$

7.3 Infinite Horizon Model Predictive Control with input targets and zone control

The IHMPC with input targets and zone control is a variation of the IHMPC scheme presented in section (7.1), consisting in the addition of an economic stage in order to compute the desired values, u_{des}, of the manipulated variables, u, at each time step k, based on an economic objective.
The computed values of u_{des} on the economic stage are sent to the controller (IHMPC are assumed in this work), and the objective of the controller is to drive the manipulated variables, u, to their desired values, u_{des}, keeping the outputs of the system, y, inside a predefined zone (range of values). Figure 7.1 show the IHMPC scheme with input targets and zone control.
Below, the economic optimization stage and the controller stage of the control structure presented in Figure 7.1 are described.

7.3.1 The economic stage

The economic stage has the aim of maximizing some economic objective (generally the utility associated with the production of some good), subject to the constraints determined by a full steady-state model of the system, and by the feasible set of control actions Ω.
Let us define the profit function P as

$$P(x(t),u(t))=ky(t)-c(x(t),u(t))-l \tag{7.18}$$

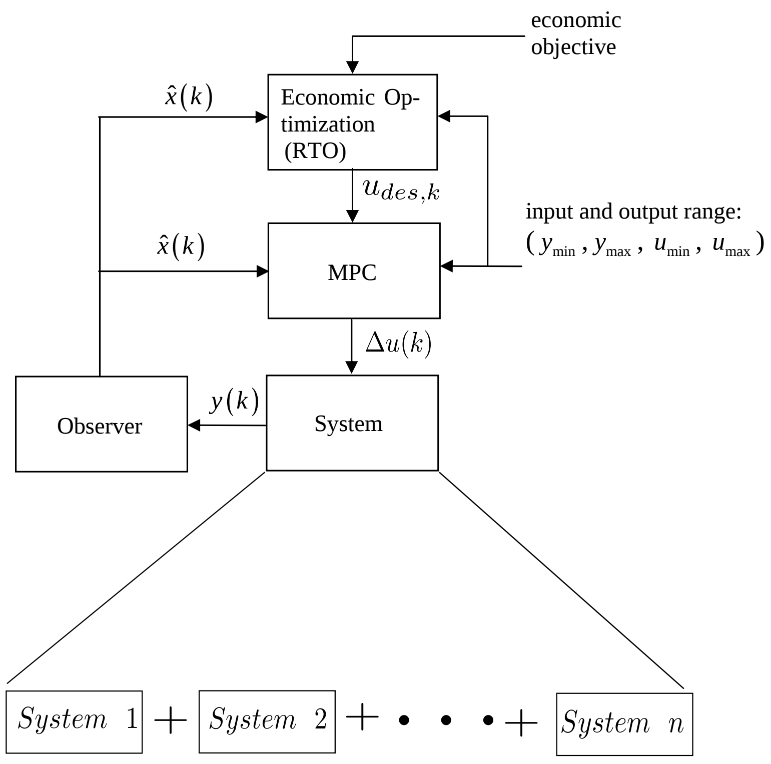

economic objective
$\hat{x}(k)$
Economic Optimization (RTO)
$u_{des,k}$
input and output range: $(y_{\min}, y_{\max}, u_{\min}, u_{\max})$
$\hat{x}(k)$
MPC
$\Delta u(k)$
Observer
$y(k)$
System
$System\ 1 + System\ 2 + \bullet \bullet \bullet + System\ n$

Figure 7.1: Schematic diagram of the infinite horizon model predictive controller with input target([18])

where k is the price per unit of $y(t)$, $c(x(t),u(t))$ denotes the total cost of produce one unit of y, and l denotes the taxes per unit of y. By definition, fixed and variable costs are respectively independent and dependent on product quantity. It should be noted that the fixed costs and taxes are not under the influence of operation. So, in an optimization problem these terms are constants and maximization of equation (7.18) simplifies to the maximization of operational profit

$$O(x(t),u(t)) = ky(t) - V(x(t),u(t)) \tag{7.19}$$

where $V(x(t),u(t))$ denotes the variable costs. Based on the model of the plant (7.1), equation (7.19) becomes

$$O(x(t),u(t)) = kg(x(t),u(t)) - V(x(t),u(t)) \tag{7.20}$$

Therefore, the optimization problem of the economical stage can be formulated as

$$\begin{aligned} &\max_{u_{des}} \int_0^{t_f} O(x(t),u_{des})dt \\ \text{subject to: } &f(x(t),u_{des}) = 0 \\ &u_{des} \in \Omega \end{aligned} \tag{7.21}$$

where t_f denotes the time at which the function $O(t)$ is expected to be maximal. Additional to the behavior constraints ($f(x(t),u(t)) = 0$), and operational constraints ($u \in \Omega$), scheduling constraints also can be included (at time t_{f1} the requirements are $y(t_{f1})$ and so on). If the required product quantity is not allowed to vary, then problem (7.21) can be reduced to

$$\begin{aligned}&\min_{u_{des}} \int_0^{t_f} V(x(t),u_{des})dt\\ \text{subject to: } & f(x(t),u_{des})=0\\ & u_{des}\in\Omega\end{aligned} \tag{7.22}$$

Here, the economic optimization is complete. Below, the control stage is presented.

7.3.2 Control stage

Infinite Horizon MPC with input target [19]

The IHMPC with targets and control zones, is an optimal control problem similar to the IHMPC problem presented in section 7.1, but in the case in which the targets and zone control are added, the cost function J defined in (7.5) must be modified in order to minimize also the offset between the current inputs and their desired values. So, the cost function becomes

$$\begin{aligned}J_{k,\infty} = &\sum_{j=1}^{\infty} (e(k+j|k)-\delta_k)^T Q\, (e(k+j|k)-\delta_k) + \sum_{j=1}^{\infty} \left(e_{u(k+j|k)}-\delta_{k,u}\right)^T Q_u \left(e_{u(k+j|k)}-\delta_{k,u}\right)\\ &+\sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S\delta_k + \delta_{k,u}^T S_u \delta_{k,u}\end{aligned} \tag{7.23}$$

where $e_{u(k+j|k)} = u(k+j)-u_{des}$, u_{des} is the vector of desired values for the system inputs, $\delta_{k,u}$ is a vector of slack variables related to the inputs, which have the economic target, Q_u and S_u are positive weighting matrices of appropriate dimensions.
To deal with the new control objective function of the IHMPC, it is necessary to redefine the state space model as follows:

$$\begin{aligned}\begin{bmatrix} x_1(k+1)\\ x_2(k+1)\\ u(k)\end{bmatrix} &= \begin{bmatrix} P & 0 & 0\\ 0 & I_{ny} & 0\\ 0 & 0 & I_{nu}\end{bmatrix}\begin{bmatrix} x_1(k)\\ x_2(k)\\ u(k-1)\end{bmatrix} + \begin{bmatrix} B_1\\ B_2\\ I_{nu}\end{bmatrix}\Delta u(k)\\ \begin{bmatrix} y(k)\\ y_u(k)\end{bmatrix} &= \begin{bmatrix} C_1 & C_2 & 0\\ 0 & 0 & I_{nu}\end{bmatrix}\begin{bmatrix} x_1(k)\\ x_2(k)\\ u(k-1)\end{bmatrix}\end{aligned} \tag{7.24}$$

In order to force the extended objective function defined in (7.24) to be bounded, the constraint represented in (7.14) must be satisfied, and a new constraint related to state x_u have to be imposed also. This new constraint is similar to the constraint represented in equation (7.14) and has the following form:

$$u(k-1)-u_{des}+\tilde{B}_u\Delta u_k-\delta_{k,u}=0 \tag{7.25}$$

where,

$$\tilde{B}_u = \begin{bmatrix} I_{nu}, .., I_{nu}\end{bmatrix} \in \Re^{n_u\times N_c\cdot n_u}$$

With this extended state space model defined en equation (7.24), the control cost defined en equation (7.23) can be written as follows:

$$\begin{aligned} J_{k,\infty} = & \sum_{j=1}^{N_c} (e(k+j|k)-\delta_k)^T Q (e(k+j|k)-\delta_k) + \sum_{j=1}^{N_c} \left(e_{u(k+j|k)} - \delta_{k,u}\right)^T Q \left(e_{u(k+j|k)} - \delta_{k_u}\right) \\ & + x_1(k+N_c|k)^T \overline{Q} x_1(k+N_c|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k + \delta_{k,u}^T S_u \delta_{k,u} \end{aligned} \tag{7.26}$$

Finally, the control optimization problem of the extended infinite horizon MPC (IHMPC) can be formulated as:

$$\begin{aligned} \min_{\Delta u_k, \delta_k, \delta_{k,u}} J_{k,\infty} = & \sum_{j=1}^{N_c} (e(k+j|k)-\delta_k)^T Q (e(k+j|k)-\delta_k) + \sum_{j=1}^{N_c} \left(e_{u(k+j|k)} - \delta_{k,u}\right)^T Q \left(e_{u(k+j|k)} - \delta_{k_u}\right) \\ & + x_1(k+N_c|k)^T \overline{Q} x_1(k+N_c|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k + \delta_{k,u}^T S_u \delta_{k,u} \end{aligned}$$

subject to

$$\begin{gathered} \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} P & 0 & 0 \\ 0 & I_{ny} & 0 \\ 0 & 0 & I_{nu} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \\ I_{nu} \end{bmatrix} \Delta u(k) \\ \begin{bmatrix} y(k) \\ y_u(k) \end{bmatrix} = \begin{bmatrix} C_1 & C_2 & 0 \\ 0 & 0 & I_{nu} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ u(k-1) \end{bmatrix} \\ \overline{C}_2 x_2(k) - r + \overline{C}_2 \tilde{B}_2 \Delta u_k - \delta_k = 0 \\ u(k-1) - u_{des} + \tilde{B}_u \Delta u_k - \delta_{k,u} = 0 \\ -\Delta u^{max} \leq \Delta u(k+j|k) \leq \Delta u^{max} \\ \Delta u(k+j|k) = 0 \ \ ; \ \ j \geq m \\ u^{min} \leq u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \leq u^{max}; \ \ j = 0,1,\ldots,m-1 \end{gathered} \tag{7.27}$$

Using model equation 7.24 to represent the output prediction as a function of the future control actions and the current state, the control objective represented in 7.26 can be written as follows:

$$J_{k,\infty} = \frac{1}{2} \begin{bmatrix} \Delta u_k^T & \delta_k^T & \delta_{k,u}^T \end{bmatrix} H \begin{bmatrix} \Delta u_k \\ \delta_k \\ \delta_{k,u} \end{bmatrix} + c_f^T \begin{bmatrix} \Delta u_k \\ \delta_k \\ \delta_{k,u} \end{bmatrix} + c \tag{7.28}$$

where

$$H = \begin{bmatrix} (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u})^T \overline{Q}_1 (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u}) + \overline{R} + F_{1u}^T \overline{Q}_2 F_{1u} & -(\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u})^T \overline{Q}_1 \bar{I} & -B_u^T \overline{Q}_{u1} \bar{I}_u \\ -\bar{I}\overline{Q}_1 (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u}) & S + \bar{I}^T \overline{Q}_1 + \bar{I} + Q & 0 \\ -\bar{I}_u^T \overline{Q}_{u1} B_u & 0 & \bar{I}_u^T \overline{Q}_{u1} \bar{I}_u \end{bmatrix}$$

$$c_f = \begin{bmatrix} (\overline{C}_1 F_{1u} + \overline{C}_2 F_{2u})^T \overline{Q}_1 (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r}) + F_{1u}^T \overline{Q}_2 (\overline{C}_1 F_x x_1(k)) + B_u^T \overline{Q}_{u1} (\bar{I}_u u(k) - \bar{r}_u) \\ -\bar{I}\overline{Q}_1 (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r}) - Q(\overline{C}_2 \bar{I} x_2(k) - \bar{r}) \\ -\bar{I}_u^T \overline{Q}_{u1} (\bar{I}_u u(k) - \bar{r}_u) \end{bmatrix}$$

$$c = (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r})^T \overline{Q}_1 (\overline{C}_1 F_x x_1(k) + \overline{C}_2 \bar{I} x_2(k) - \bar{r}) + (\overline{C}_1 F_x x_1(k))^T \overline{Q}_2 (\overline{C}_1 F_x x_1(k)) + (\bar{I}_u u(k) - \bar{r}_u)^T \overline{Q}_{u1} (\bar{I}_u u(k) - \bar{r}_u)$$

$$\bar{I}_u = \begin{bmatrix} I_{nu} \\ \vdots \\ I_{nu} \end{bmatrix}, B_u = \begin{bmatrix} I_{nu} & \cdots & I_{nu} \\ \vdots & \ddots & \vdots \\ I_{nu} & \cdots & I_{nu} \end{bmatrix}, \overline{Q}_{u1} = \operatorname{diag}[Q_u, \cdots, Q_u],$$

Finally the control optimization problem can be formulated as:

$$\min_{\Delta u_k, \delta_k, \delta_{k,u}} J_{k,\infty} = \tfrac{1}{2} \begin{bmatrix} \Delta u_k^T & \delta_k^T & \delta_{k,u}^T \end{bmatrix} H \begin{bmatrix} \Delta u_k \\ \delta_k \\ \delta_{k,u} \end{bmatrix} + c_f^T \begin{bmatrix} \Delta u_k \\ \delta_k \\ \delta_{k,u} \end{bmatrix} + c \tag{7.29}$$

subject to

$$\begin{gathered} \overline{C}_2 x_2(k) - r + \overline{C}_2 \tilde{B}_2 \Delta u_k - \delta_k = 0 \\ u(k-1) - u_{des} + \tilde{B}_u \Delta u_k - \delta_{k,u} = 0 \\ -\Delta u^{max} \leq \Delta u(k+j|k) \leq \Delta u^{max} \\ \Delta u(k+j|k) = 0 \ ; \ j \geq m \\ u^{min} \leq u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \leq u^{max}; \ \ j = 0,1,...,m-1 \end{gathered} \tag{7.30}$$

Infinite Horizon MPC with input target and zone control

The results presented here are based on [18]
The zone control strategy is implemented in applications where the exact values of the controlled outputs are not important, as long as they remain inside a range with specific limits.
The control structure considered in this work is represented in Figure 7.2. In this structure, at time step k, the real time economic optimization (RTO) stage, which is based on a rigorous stationary model, computes the optimal target,for the manipulated input variables. Here, it is assumed that the

control stage corresponding to the MPC is dedicated to guide the manipulated inputs to the desired targets defined by the supervisory economic stage, while keeping the outputs within specified zones. In Figure 7.2 it is assumed that the PID or multivariable regulatory level is included in the system level and that the regulatory level is capable of enforcing the set points determined by the MPC level. The MPC optimization problem that implements the zone control strategy and enforces the economic target is as follows:

$$\begin{aligned}\min_{\Delta u_k, y_{sp,k}, \delta_k, \delta_{k,u}} J_{k,\infty} &= \sum_{j=1}^{N_c} \left(y(k+j) - y_{sp,k} - \delta_k\right)^T Q \left(y(k+j) - y_{sp,k} - \delta_k\right) \\ &+ \sum_{j=1}^{N_c} \left(u(k+j) - u_{des,k} - \delta_{k,u}\right)^T Q \left(u(k+j) - u_{des,k} - \delta_{k_u}\right) \\ &+ x_1(k+N_c|k)^T \overline{Q} x_1(k+N_c|k) + \sum_{j=1}^{N_c-1} \Delta u(k+j|k)^T R \Delta u(k+j|k) + \delta_k^T S \delta_k + \delta_{k,u}^T S_u \delta_{k,u}\end{aligned}$$

subject to

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} P & 0 & 0 \\ 0 & I_{ny} & 0 \\ 0 & 0 & I_{nu} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \\ I_{nu} \end{bmatrix} \Delta u(k)$$

$$\begin{bmatrix} y(k) \\ y_u(k) \end{bmatrix} = \begin{bmatrix} C_1 & C_2 & 0 \\ 0 & 0 & I_{nu} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ u(k-1) \end{bmatrix}$$

$$\overline{C}_2 x_2(k) - y_{sp,k} + \overline{C}_2 \tilde{B}_2 \Delta u_k - \delta_k = 0$$

$$u(k-1) - u_{des,k} + \tilde{B}_u \Delta u_k - \delta_{k,u} = 0$$

$$-\Delta u^{max} \leq \Delta u(k+j|k) \leq \Delta u^{max}$$

$$y_{min} \leq y_{sp,k} \leq y_{max}$$

$$\Delta u(k+j|k) = 0 \;\; ; \;\; j \geq m$$

$$u^{min} \leq u_{k-1} + \sum_{i=0}^{j} \Delta u_{k+i} \leq u^{max}; \;\; j = 0,1,\ldots,m-1$$

7.4 Conclusion

This work was proposed a strategy to implement a MPC controller for large scale systems in which the system outputs are controlled in specified zones and the manipulated inputs have targets associated to the economic objectives of the controlled system.

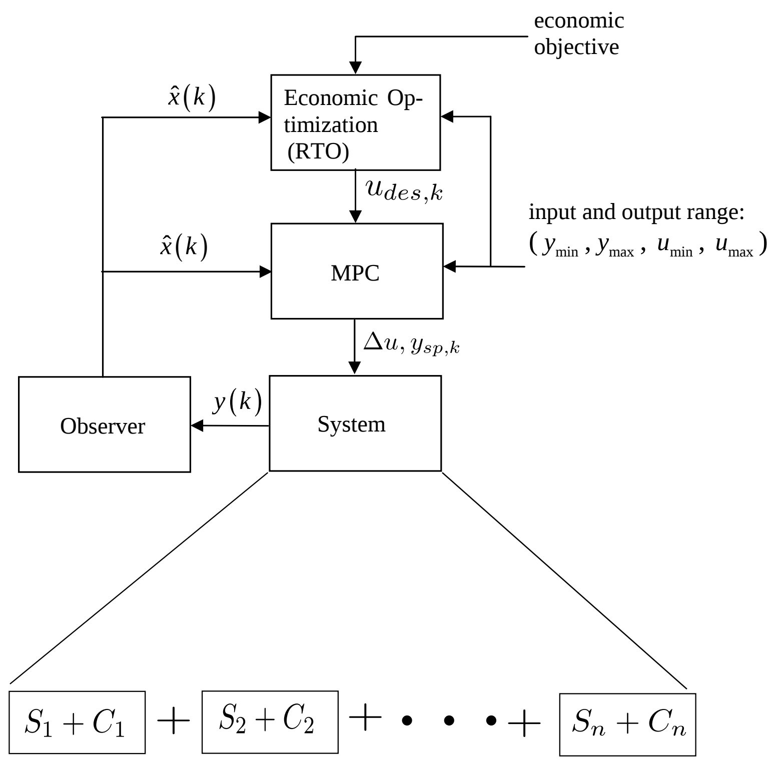


Figure 7.2: Schematic diagram of the infinite horizon model predictive controller with input target and zone control([18])

Bibliography

[1] W. Al-Gherwi, H. Budman, and A. Elkamel. An online algorithm for robust distributed model predictive control. In *ADCHEM 2009 (International Symposium on Advanced Control of Chemical Processes)*, Istanbul, Turkey, July 2009.

[2] W. Al-Gherwi, H. Budman, and A. Elkamel. Selection of control structure for distributed model predictive control in the presence of model errors. *Journal of Process Control*, 20(3):270–284, 2010.

[3] A. Alessio and A. Bemporad. Decentralized model predictive control of constrained linear systems. In *European Control Conference*, pages 2813–2818, 2007.

[4] A. Alessio and A. Bemporad. Stability conditions for decentralized model predictive control under packet drop communication. In *American Control Conference*, pages 3577–3582, 2008.

[5] I. Alvarado. *Model Predictive Control for Tracking Constrained Linear Systems*. PhD thesis, Univ. de Sevilla, 2007.

[6] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[7] F. Borrelli, T.ás 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*, pages 3955–3960, Seville, Spain, December 2005.

[8] E. F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry. Second Edition*. Springer-Verlag, London, England, 2004.

[9] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, February 2002.

[10] E. Camponogara and S.N. Talukdar. Distributed model predictive control: Synchronous and asynchronous computation. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 37(5):732–745, September 2007.

[11] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operation Research*, (8):101–111, 1960.

[12] C. de Boor. *A Practical Guide to Splines*. Springer, New York, 1978.

[13] 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*, 11(3):6–15, September 2009.

[14] 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.

[15] X. Du, Y. Xi, and S. Li. Distributed model predictive control for large-scale systems. In *American Control Conference*, volume 4, pages 3142–3143, 2001.

[16] W. B. Dunbar and R. M. Murray. Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):549–558, April 2006.

[17] A. Ferramosca, D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. MPC for tracking with optimal closed-loop performance. *Automatica*, 45:1975–1978, 2009.

[18] A. H. González and D. Odloak. A stable MPC with zone control. *Journal of Process Control*, 19(1):110 – 122, 2009.

[19] A.H. González, D. Odloak, J.L. Marchetti, and O.A.Z. Sotomayor. Infinite horizon MPC of a heat-exchanger network. *Chemical Engineering Research and Design*, 84(11):1041 – 1050, 2006.

[20] S.-P. Han and G. Lou. A parallel algorithm for a class of convex programs. *SIAM Journal on Control and Optimization*, 26(2):345–355, March 1988.

[21] D. Jia and B. Krogh. Min-max feedback model predictive control for distributed control with communication. In *American Control Conference*, volume 6, pages 4507–4512, 2002.

[22] D. Jia and B. H. Krogh. Distributed model predictive control. In *American Control Conference*, volume 4, pages 2767–2772, 2001.

[23] K. H. Johansson. The quadruple-tank process. *IEEE Trans. Cont. Sys. Techn.*, 8:456–465, 2000.

[24] T. Keviczky, F. Borrelli, and G. J. Balas. Decentralized receding horizon control for large scale dynamically decoupled systems. *Automatica*, 42(12):2105–2115, December 2006.

[25] L. S. Lasdon. Duality and decomposition in mathematical programming. In *IEEE Transactions on Systems Science and Cybernetics*, volume 4, pages 86–100, July 1968.

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

[27] S. Li, Y. Zhang, and Q. Zhu. Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem. *Information Sciences*, 170(2-4):329–349, February 2005.

[28] D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho. MPC for tracking of piece-wise constant references for constrained linear systems. *Automatica*, 44:2382–2387, 2008.

[29] G. Lionis and K. J. Kyriakopoulos. Approximate control of formations of muliagent systems. In *Proceedings of the 44th Conference on Decision and Control, and the European Control Conference 2005*, pages 4958–4963, seville, Spain, December 2005.

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

[31] J. M. Maestre, D. Munoz de la Pena, and E. F. Camacho. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, 2010. doi:10.1002/oca.940.

[32] M. S. Mahmoud. Multilevel systems control and applications: A survey. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-7(3):125–143, March 1977.

[33] P. Massioni and M. Verhaegen. Distributed control of vehicle formations: a decomposition approach. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 2906–2912, Cancun, Mexico, December 2008.

[34] M. Mercangoz and F. J. Doyle III. Distributed model predictive control of an experimental four-tank system. *Journal of Process Control*, 17(3):297–308, March 2007.

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

[36] M. D. Mesarovic, D. Macko, and Y. Takahara. Two coordination principles and their application in large scale systems control. *Automatica*, 6:261–270, 1970.

[37] M. Morari and J.H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682, 1999.

[38] P-D. Moroşan, R. Bourdais, D. Dumur, and J. Buisson. Building temperature regulation using a distributed model predictive control. *Energy and Buildings*, 42(9):1445–1452, 2010.

[39] P-D. Moroşan, R. Bourdais, D. Dumur, and J. Buisson. A dynamic horizon distributed predictive control approach for temperature regulation in multi-zone buildings. In *Mediterranean Conference on Control and Automation*, pages 622–627. IEEE, jun. 2010.

[40] I. Necoara, D. Doan, and J. A. K. Suykens. Application of the proximal center decomposition method to distributed model predictive control. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 2900–2905, Cancun, Mexico, December 2008.

[41] A. Núñez, B. Picasso, D. De Vito, R. Scattolini, P. Colaneri, J. Espinosa, F. Valencia, B. De Schutter, and K. Stankova. Report on newly developed coordination mechanisms for hierarchical and distributed MPC. Deliverable D3.3.2, European FP7 STREP project HD-MPC, 2010.

[42] D. Odloak. Extended robust model predictive control. *AIChE Journal*, 50(8):1824–1836, 2004.

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

[44] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[45] C. V. Rao and J. B. Rawlings. Linear programming and model predictive control. *Journal of Process Control*, 10(2-3):283–289, 2000.

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

[47] J. B. Rawlings and B. T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839–845, October 2008.

[48] A.G. Richards and J.P. How. Robust distributed model predictive control. *International Journal of Control*, 80(9):1517–1531, September 2007.

[49] M. A. Rodrigues and D. Odloak. MPC for stable linear systems with model uncertainty. *Automatica*, 39(4):569 – 583, 2003.

[50] N. R. Sandell, P. Varaiya, M. Athans, and M. G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, 23(2):108–128, April 1978.

[51] C. Savorgnan and M. Diehl. Report on literature survey and analysis of (optimization) methods for robust distributed MPC. Deliverable D3.2.1, European FP7 STREP project HD-MPC, 2009.

[52] R. Scattolini. Architectures for distributed and hierarchical model predictive control - A review. *Journal of Process Control*, 19(5):723–731, May 2009.

[53] H. F. Scherer, E. Camponogara, and A. Plucenio. Distributed optimization for predictive control of a distillation column with output and control-input constraints. In *ADCHEM 2009 (International Symposium on Advanced Control of Chemical Processes)*, Istanbul, Turkey, July 2009.

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

[55] H. Scheu, J. C. Calderon, D. Doan, J. F. Garcia, R. Negenborn, A. Tarau, F. Valencia Arroyave, B. De Schutter, J.J. Espinosa, and W. Marquardt. Report on assessment of existing coordination mechanisms for simple case studies, and on possible options for improving and extending these coordination mechanisms. Deliverable D3.3.1, European FP7 STREP project HD-MPC, 2009.

[56] H. Scheu and W. Marquardt. Report on literature survey on hierarchical and distributed nonlinear MPC, including analysis and comparison, and description of the resulting methodological framework. Deliverable D3.1.1, European FP7 STREP project HD-MPC, 2009.

[57] H. Scheu, B. Picasso, D. Desiderio, R. Scattolini, and D. Limon. Report on newly developed methods for hierarchical and distributed robust nonlinear dynamic MPC. Deliverable D3.2.2, European FP7 STREP project HD-MPC, 2010.

[58] M. Schlegel, K. Stockmann, T. Binder, and W. Marquardt. Dynamic optimization using adaptive control vector parameterization. *Computers & Chemical Engineering*, 29(8):1731–1751, 2005.

[59] 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*, 125(4):189–195, July 1999.

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

[61] N. J. Smith and A. P. Sage. An introduction to hierarchical systems theory. *Comput. & Elect. Engng*, 1:55–71, 1973.

[62] A. N. Venkat, J. B. Rawlings, and S. J. Wright. Stability and optimality of distributed model predictive control. In *Conference on Decision and Control and European Control Conference*, pages 6680–6685, dec. 2005.

[63] A.N. Venkat, I.A. Hiskens, J.B. Rawlings, and S.J. Wright. Distributed MPC strategies with application to power system automatic generation control. *IEEE Transactions on Control Systems Technology*, 16(6):1192–1206, November 2008.

[64] 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*, pages 381–386, Cancun, Mexico, December 2008.

[65] Y. Zhang and S. Li. Networked model predictive control based on neighbourhood optimization for serially connected large-scale systems. *Journal of Process Control*, 17(1):37–50, 2007.