Acta Universitatis Sapientiae
Electrical and Mechanical Engineering, 2 (2010) 5-15

Comparative Analysis of Model Predictive Control Structures

Katalin GYÖRGY, László DÁVID

Department of Electrical Engineering, Faculty of Technical and Human Sciences,
Sapientia University, Tg. Mureş,
e-mail: kgyorgy@ms.sapientia.ro, ldavid@ms.sapientia.ro

Manuscript received October 28, 2010; revised November 03, 2010.

Abstract: The industrial implementation of advanced multivariable control techniques like Model Predictive Control (MPC) is complex, time consuming and therefore it is expensive. Nowdays it is a popular research area to reduce the complexity of the MPC algorithm while preserving the control performance. This problem could be solvable with implementation of the MPC solution in a distributed way. The main idea of this work is to develop simple software agents that can be easily implemented in low cost embedded systems. Each one of these software agents solves the problem of finding one of the control actions with parallel computational facilities. This paper presents at first some general and theoretical considerations about centralized and distributed model predictive algorithms. The comparison between these algorithms is made using numerical simulation of these methods for a multiple input and multiple output theoretical linear discrete-time system. The comparision is possible to be made from the point of wiew of the normalized absolute reference tracking error. There is described a possible implementation of the distributed MPC algorithm using Matlab Simulink environment. It is important to notice that the algorithm to be solved by each software agent while computing the control action is much simpler than the one to be solved by the centralized algorithm.

Keywords: optimal control, cost function, model predictive control, distributed control, prediction horizon, control horizon, Jacobi over-relaxation method, linear constrained optimal programming problem.

1. Introduction

The model predictive control (MPC), – also called receding horizon control (RHC) – is the most important advanced control technique which has been very successful in practical applications, where the control signal can be obtained by solving a discrete-time optimal control problem over a finite horizon. The most important advantage of the MPC algorithms is the fact that they have the unique ability to take into account constraints imposed on process inputs, process state variables and outputs, which usually determine the quality, the efficiency, and safety of production. Implementation of centralized state space (SS) MPC algorithms is becoming an important issue for different multivariable industrial processes. The main idea of our work is to develop a multi-agent software that can be implemented in low cost embedded systems, with parallel computational facilities. These software agents are valid for a default model and can be multiplied and customized according to the control horizon. Each one solves the problem of finding one of the control actions. This procedure is repeated several times before the control action values are delivered to the final control elements. An agent, as an executive, has to know general information about the system and some others which are specific of his own department. It is important to notice that the algorithm to be solved by each agent while computing its control action is much simpler than the one to be solved by the centralized solution.

Previous works on distributed MPC [2], [3], [4], [6] use a wide variety of approaches, including multi-loop ideas, decentralized computation using standard coordination techniques, robustness to the actions of others, penalty functions, and partial grouping of computations. The key point is that, when decisions are made in a decentralized fashion, the actions of each subsystem must be consistent with those of the other subsystems, so that decisions taken independently do not lead to a violation of the coupling constraints. The decentralization of the control becomes more complex when disturbances act on the subsystems making the prediction of future behavior uncertain.

We will analyze how the overall performance of a distributed system is influenced if one or more agents – except the coordinating agent –, fail or obviously underperforms from some reasons. The objective is to solve SS-MPC problems with locally relevant variables, costs, and constraints, but without solving a centralized SS-MPC problem. The coordinated distributed computations solve an equivalent centralized SS-MPC problem. This means that properties that can be proven for the equivalent centralized MPC problem (e.g., stability, robustness) are valid to the above distributed SS-MPC implementation. The significance of the proposed distributed control scheme is that it reduces the computational requirements in complex large-scale systems and it makes possible the development of fault tolerant control systems.

2. Centralized State Space Model Predictive Control

All the MPC algorithms possess common elements and different options can be chosen for each one of these elements: prediction model, objective function and algorithms for obtaining the control law. In this paper the process model is a discrete input – state – output relationship:

$$\begin{aligned} \underline{x}_{k+1} &= \underline{\Phi} \cdot \underline{x}_k + \underline{\Gamma} \cdot \underline{u}_k \\ \underline{y}_k &= \underline{C} \cdot \underline{x}_k \end{aligned}, \tag{1}$$

where $\underline{x}_k$ is the state vector (*n x 1*), $\underline{u}_k$ is the input vector (*m x 1*), $\underline{y}_k$ is the output vector (*p x 1*), and $\underline{\Phi}$, $\underline{\Gamma}$ and $\underline{C}$ are the matrices of the system. If these matrices (parameters) are unknown, we have to implement a system identification module in the control algorithm.

The centralized model predictive algorithm looks for the vector $\Delta\underline{U}_k$ that minimizes a cost function represented by the scalar J, defined as:

$$J\left(\Delta\underline{U}_k\right) = \left(\underline{Y}_k - \underline{Y}_k^{ref}\right)^T \cdot \underline{Q} \cdot \left(\underline{Y} - \underline{Y}_k^{ref}\right) + \Delta\underline{U}_k^T \cdot \underline{R} \cdot \Delta\underline{U}_k, \tag{2}$$

where $\underline{Y}_k^{ref}$ is the vector with the future references, $\underline{Y}_k$ is the vector of the predictions of the controlled variables (output signals), $\Delta\underline{U}_k$ is a vector of future variations of the control signal, $\underline{Q}$ is a diagonal matrix with weights for set-point following enforcement, $\underline{R}$ is a diagonal matrix with weights for control action changes. If the prediction horizon is N and the control horizon is N_c these vectors and matrices are [1]:

$$\underline{Y}_k = \begin{bmatrix} \underline{y}_{k+1/k} \\ \vdots \\ \underline{y}_{k+N/k} \end{bmatrix}, \quad \underline{Y}_k^{ref} = \begin{bmatrix} \underline{y}_{k+1/k}^{ref} \\ \vdots \\ \underline{y}_{k+N/k}^{ref} \end{bmatrix}, \quad \Delta\underline{U}_k = \begin{bmatrix} \underline{\Delta u}_{k/k} \\ \vdots \\ \underline{\Delta u}_{k+N_c-1/k} \end{bmatrix}, \tag{3}$$

$$\underline{Q} = \begin{bmatrix} \underline{Q}_1 & 0 & \dots & 0 \\ 0 & \underline{Q}_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \underline{Q}_N \end{bmatrix}, \qquad \underline{R} = \begin{bmatrix} \underline{R}_0 & 0 & \dots & 0 \\ 0 & \underline{R}_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \underline{R}_{N_c-1} \end{bmatrix}. \tag{4}$$

An incremental state space model can be used if the model input is the control increment $\Delta\underline{u}_k = \underline{u}_k - \underline{u}_{k-1}$. The following representation is obtained for predictions:

$$\underline{Y_k} = \underline{\Phi}^* \cdot \underline{x}_k + \underline{\Gamma}^* \cdot \underline{u}_{k-1} + \underline{G}_y \cdot \Delta\underline{U}_k \,, \tag{5}$$

where

$$\underline{\Phi}^* = \begin{bmatrix} \underline{C}\cdot\underline{\Phi} \\ \vdots \\ \underline{C}\cdot\underline{\Phi}^{N_c} \\ \underline{C}\cdot\underline{\Phi}^{N_c+1} \\ \vdots \\ \underline{C}\cdot\underline{\Phi}^{N} \end{bmatrix} \quad \underline{\Gamma}^* = \begin{bmatrix} \underline{C}\cdot\underline{\Gamma} \\ \vdots \\ \sum_{i=0}^{N_c-1} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} \\ \sum_{i=0}^{N_c} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} \\ \vdots \\ \sum_{i=0}^{N-1} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} \end{bmatrix} \quad \underline{G}y = \begin{bmatrix} & & \\ \underline{C}\cdot\underline{\Gamma} & \cdots & \underline{0} \\ \vdots & \cdots & \vdots \\ \sum_{i=0}^{N_c} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} & \ddots & \underline{C}\cdot(\underline{\Phi}\cdot\underline{\Gamma}+\underline{\Gamma}) \\ \vdots & & \vdots \\ \sum_{i=0}^{N-1} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} & \cdots & \sum_{i=0}^{N-N_c} \underline{C}\cdot\underline{\Phi}^i\cdot\underline{\Gamma} \end{bmatrix}. \tag{6}$$

The cost function can be written as:

$$J\left(\Delta\underline{U}_k\right) = \frac{1}{2}\Delta\underline{U}_k{}^T \cdot \underline{H} \cdot \Delta\underline{U}_k + \underline{f}^T \cdot \Delta\underline{U}_k + const \,, \tag{7}$$

where

$$\begin{aligned} \underline{H} &= 2\cdot\left(\underline{G}_y{}^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}\right), \\ \underline{f} &= -2\cdot\underline{G}_y{}^T \cdot \underline{Q} \cdot \underline{E}_k \,, \\ \underline{E}_k &= \underline{Y}_k^{ref} - \underline{\Phi}^* \cdot \underline{x}_k - \underline{\Gamma}^* \cdot \underline{u}_{k-1} \,. \end{aligned} \tag{8}$$

For problems without constraints the centralized model predictive control determines the vector $\Delta\underline{U}_k$ that makes

$$\frac{\partial J(\Delta\underline{U}_k)}{\partial(\Delta\underline{U}_k)} = 0 \Rightarrow \Delta\underline{U}_k^{opt} = \frac{1}{2}\cdot\left(\underline{H}+\underline{H}^T\right)^{-1}\cdot\underline{f} = \left(\underline{G}_y^T\cdot\underline{Q}\cdot\underline{G}_y + \underline{R}\right)^{-1}\cdot\underline{G}_y^T\cdot\underline{Q}\cdot\underline{E}_k \tag{9}$$

It is to be mentioned that only the first control action is taken at each instant, and the procedure is repeated for the next control decision in a receding horizon fashion.

3. Distributed State Space Model Predictive Control

The implementation of distributed model predictive control needs to search for $\Delta\underline{u}_{k/k}, \Delta\underline{u}_{k+1/k}, \ldots, \Delta\underline{u}_{k+N_c-1/k}$ [5], [7], that makes the

$$\frac{\partial J(\Delta\underline{U}_k)}{\partial(\Delta\underline{u}_{g/k})} = 0 \ . \tag{10}$$

for $k \le g \le k + N_c - 1$.

If the cost function is written as:

$$J(\Delta \underline{U}_k) = \Delta \underline{U}_k{}^T \cdot \left(\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}\right) \cdot \Delta \underline{U}_k - 2 \cdot \underline{E}_k^T \cdot \underline{Q} \cdot \underline{G}_y \cdot \Delta \underline{U}_k + \underline{E}_k^T \cdot \underline{Q} \cdot \underline{E}_k \tag{11}$$

then the first order optimality condition can be determined in the following way:

$$\begin{aligned} \frac{\partial J(\Delta \underline{U}_k)}{\partial (\Delta \underline{u}_{g/k})} &= 2 \cdot \left[\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}\right]_{g-k+1,g-k+1} \cdot \Delta \underline{u}_{g,k} - \\ &- 2 \cdot \sum_{i=1}^{N} \left(\left[\underline{Q} \cdot \underline{G}_y\right]_{i,g-k+1}^T \cdot \left[\underline{E}_k\right]_i \right) + \\ &+ \sum_{\substack{i=0 \\ i \neq g-k}}^{N_c-1} \left(\left([\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}]_{i+1,g-k+1}^T + [\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}]_{g,i+1} \right) \Delta \underline{u}_{k+i,k} \right) \end{aligned} \tag{12}$$

The variation of input signal is

$$\begin{aligned} \Delta \underline{u}_{g/k} = &\left(2 \cdot \left[G_y^T \cdot Q \cdot G_y + R\right]_{g-k+1,g-k+1}\right)^{-1} \cdot \left(2 \cdot \sum_{i=1}^{N} \left(\left[\underline{Q} \cdot \underline{G}_y\right]_{i,g-k+1}^T \cdot \left[\underline{E}_k\right]_i \right) - \right. \\ &\left. - \sum_{\substack{i=0 \\ i \neq g-k}}^{N_c-1} \left(\left([\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}]_{i+1,g-k+1}^T + [\underline{G}_y^T \cdot \underline{Q} \cdot \underline{G}_y + \underline{R}]_{g,i+1} \right) \Delta \underline{u}_{k+i,k} \right) \right) \end{aligned} \tag{13}$$

The first value of every $\Delta \underline{u}_{g/k}$ is only an approximation since it depends on the other $\Delta \underline{u}_{i+k/k}$ values ($i \neq g-k$). It should be noticed that the computation burden to obtain $\Delta \underline{u}_{g/k}$ is much smaller than the one needed to compute the whole vector $\Delta \underline{U}_k$. As already discussed, in this distributed approach, the vector $\Delta \underline{U}_k$ is determined by software agents using a combination of repeated computation of $\Delta \underline{u}_{g/k}$ and exchange of information.

The equation (13) can be written in the following general form:

$$\Delta \underline{u}_{k+j/k}^n = \sum_{\substack{i=0 \\ i \neq j}}^{N_c-1} \left(\underline{A}_{j+1,i+1} \cdot \Delta \underline{u}_{k+i/k}^{n-1} + \underline{B}_{j+1} \right), \tag{14}$$

where $0 \le j \le N_c - 1$, the matrices $\underline{A}_{i,j}$ have the dimension $m \times m$ and vector $\underline{B}_i$ has the dimension $m \times 1$, where m is the number of inputs. Matrix $\underline{A}_{i,i}$ is zero. A centralized expression for $\Delta \underline{U}_k$ using equation (14) can be written as:

$$\begin{bmatrix} \underline{\Delta u}_{k/k} \\ \underline{\Delta u}_{k+1/k} \\ \vdots \\ \underline{\Delta u}_{k+N_c-1/k} \end{bmatrix}^n = \begin{bmatrix} \underline{0} & \underline{A}_{1,2} & \cdots & \underline{A}_{1,Nc} \\ \underline{A}_{2,1} & \underline{0} & \cdots & \underline{A}_{2,Nc} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{A}_{Nc,1} & \underline{A}_{Nc,2} & \cdots & \underline{0} \end{bmatrix} \cdot \begin{bmatrix} \underline{\Delta u}_{k/k} \\ \underline{\Delta u}_{k+1/k} \\ \vdots \\ \underline{\Delta u}_{k+N_c-1/k} \end{bmatrix}^{n-1} + \begin{bmatrix} \underline{B}_1 \\ \underline{B}_2 \\ \vdots \\ \underline{B}_{Nc} \end{bmatrix} \tag{15}$$

which, in a compact form becomes

$$\underline{\Delta U}_k^n = \underline{A} \cdot \underline{\Delta U}_k^{n-1} + \underline{B} \,. \tag{16}$$

The convergence of the $\underline{\Delta U}_k$ vectors to their true values has to be assured for a reliable application. For unconstrained applications the results obtained in the field of distributed computation can be used [5]. The Jacobi over-relaxation approach is adopted here by recomputing $\underline{\Delta U}_k$ as a linear combination of the value computed using equation (16) and the value obtained in the previous iteration,

$$\Delta \underline{U}_{k,\,filtered}^n = (I - diag(\underline{\alpha})) \cdot \Delta \underline{U}_k^n + diag(\underline{\alpha}) \cdot \Delta \underline{U}_k^{n-1} \tag{17}$$

where $\underline{\alpha}$ is a vector of the filter parameters. Applying the filter according to equation (16), it results:

$$\begin{aligned} \underline{\Delta U}_k^n &= \left(\left(\underline{I} - diag(\underline{\alpha})\right) \cdot \underline{A} + diag(\underline{\alpha})\right) \cdot \underline{\Delta U}_k^{n-1} + \left(\underline{I} - diag(\underline{\alpha})\right) \cdot \underline{B} = \\ &= \underline{A}(\underline{\alpha}) \cdot \underline{\Delta U}_k^{n-1} + \left(\underline{I} - diag(\underline{\alpha})\right) \cdot \underline{B} \end{aligned} \tag{18}$$

A sufficient condition for convergence of the iterative process is to have $\left\|\underline{A}(\underline{\alpha})\right\| < 1$ for $\alpha \in (0,1)$. The search for a filter vector $\underline{\alpha}$ which minimizes $\left\|\underline{A}(\underline{\alpha})\right\|$ can be reduced to a linear constrained optimal programming problem.

4. Numerical simulation

This section presents the application of the centralized and distributed model predictive algorithm to a multiple input and a multiple output theoretical system which is characterized by following state space model:

$$\begin{aligned} \underline{x}_{k+1} &= \begin{bmatrix} 0.7 & 0 & 0.1 & 0 \\ 0 & -0.5 & 0.2 & 0 \\ 0 & 0.01 & 0.1 & 0 \\ 0.01 & 0 & 0 & -0.5 \end{bmatrix} \cdot \underline{x}_k + \begin{bmatrix} 4 & 0 \\ 3 & 9 \\ -10 & 1 \\ 0 & 2 \end{bmatrix} \cdot \underline{u}_k \\ \underline{y}_k &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \underline{x}_k \end{aligned} \tag{19}$$

where

$\underline{x}_k = \begin{bmatrix} x_{1,k} & x_{2,k} & x_{3,k} & x_{4,k} \end{bmatrix}^T$, $\underline{u}_k = \begin{bmatrix} u_{1,k} & u_{2,k} \end{bmatrix}^T$ and $\underline{y}_k = \begin{bmatrix} y_{1,k} & y_{2,k} \end{bmatrix}^T$.

For both algorithms the Simulink models have been built and the following parameters were used for both simulations:

$$N = 4; \quad N_c = 3; \quad R = 0.1 \cdot I_2 \quad Q = 10 \cdot I_2 \tag{20}$$

The Simulink diagram of the centralized predictive control is shown in *Fig. 1*, where the "Centralized_MPC_control" subsystem contains one complex S-function module for centralized control algorithm. The Simulink model of the distributed predictive algorithm is shown in *Fig. 2*.

The "Distributed_MPC_control" subsystem is presented separately in *Fig. 3*, where the three interdependent modules for calculating $\Delta\underline{u}_{1/k}, \Delta\underline{u}_{2/k}, \Delta\underline{u}_{3/k}$ can be observed. The structure of these modules is one and the same, just the input signals and parameters are different.

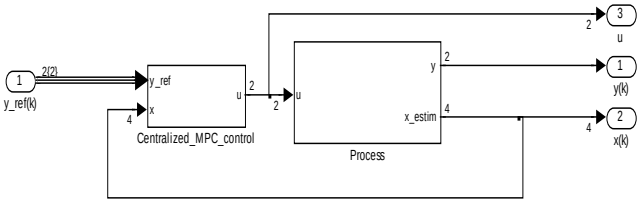


Figure 1: Simulink diagram for numerical simulation of the centralized predictive control.

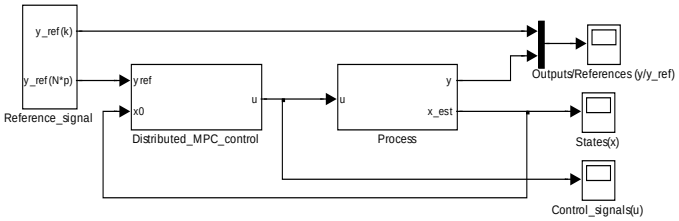


Figure 2: Simulink diagram for numerical simulation of the distributed predictive control.

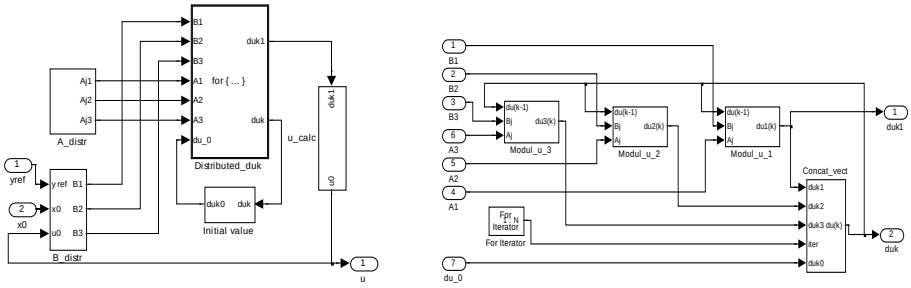


Figure 3: Subsystem diagram for distributed predictive algorithms (Nc=3).

The choice of alpha (α) provides all eigenvalues of matrix $A(\underline{\alpha})$ of equation (18) smaller than 1, which is sufficient to assure that the iterative method converges. These values were determined before the numerical simulation, and the one optimal constrained problem was solved in Matlab environment. It seems that the parameter tuning for the distributed algorithm does not need to be exactly the same as the one used for the centralized version. For the same amount of information exchange among agents, a faster reference filter improves the response.

The results obtained by numerical simulation for the centralized control algorithm using a variable reference signal are shown in *Fig. 4.* and results of numerical simulation of the distributed algorithm after 500 iterations are shown in *Fig. 5.*

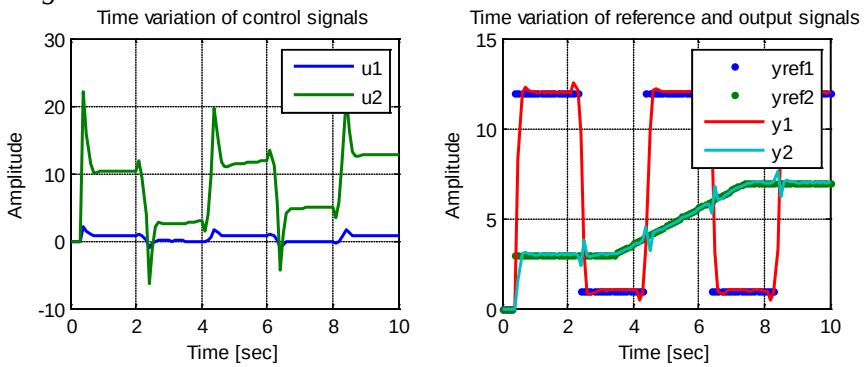
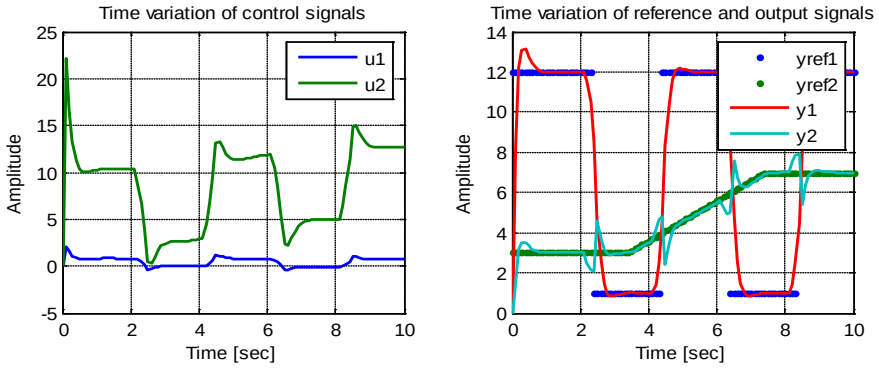


Figure 4: Time variation of the control signals (a) and outputs signals (b) in case of the centralized algorithm.



(a) (b)

Figure 5: Time variation of the control signals (a) and outputs signals (b) in case of distributed algorithm after 500 iterations.

It is noticeable that control signal obtained with the distributed algorithm is smoother than the control signal obtained with the centralized algorithm.

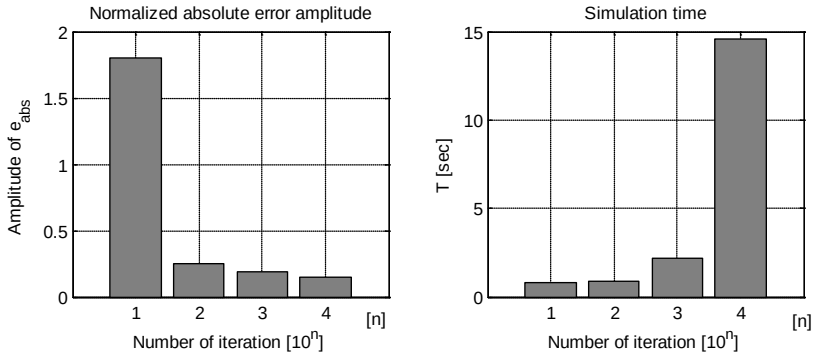
In order to have an idea on the number of information interchange iterations between agents needed for a certain performance, an analysis has been made based on the error betveen the outputs and reference signals. For both outputs (i=1,2) it was computed the error at every sample time k =1,…, N_t:

$$e_{i,k} = \frac{(y_{i,k} - y_{i,k}^{ref})}{y_{i,k}^{ref}} \tag{21}$$

A normalized absolute reference tracking error was computed using following relationship:

$$e_{abs} = \frac{1}{2}\sum_{i=1}^{2}\left(\frac{1}{N_t}\sum_{k=1}^{N_t}\left|e_{i,k}\right|\right) \tag{22}$$

There was estimated also the simulation time using the Matlab functions *clock* and *etime* for different numbers of iteration in case of the distributed model predictive control algorithms. *Fig. 6.a* presents the variation of the average errors and *Fig. 6.b* presents the estimated simulation time versus the number of iterations in case of the distributed algorithms. The simulation time is more important to be calculated for both algorithms (centralized version and distributed version) with different prediction horizon (N) values. This comparision is presented at *Fig. 7* in case of a fixed control horizon (Nc=3).



(a) (b)

Figure 6: Variation of the reference tracking error (a) and of the simulation time (b) versus the number of iterations in case of the distributed algorithm.

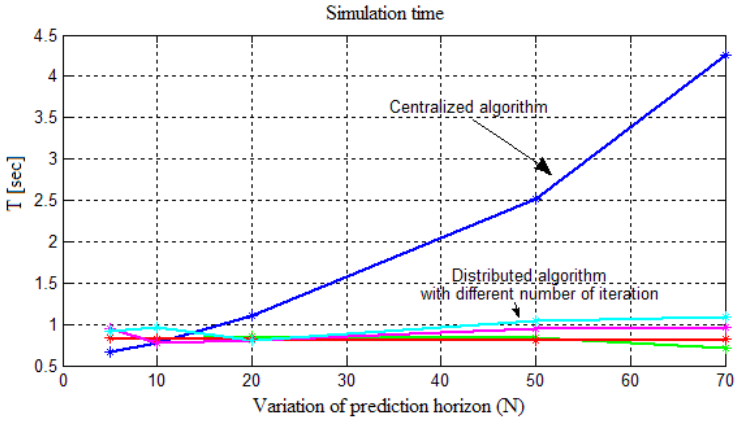


Figure 7: Comparison of the simulation times represented in function of the prediction horizon (N), obtained in case of the centralized respectively in case of the distributed algorithm.

5. Conclusion

The performance of the distributed control applied for the example discussed in the paper is comparable to that obtained with the centralized model predictive control. The computation power needed to solve the distributed problem is smaller than that is needed for the centralized case. This fact may allow the utilization of the model predictive control executed in distributed hardware with low computational power. The size of the centralized problem grows considerably with the number of inputs/outputs while the size of the

distributed problem remains the same for the same control horizon. One point to mention is that unlike in the case of the presented example, most of the multivariable problems do not have a complete interaction. In case of the distributed algorithms the problem is to choose the convenient sample time and the correct filter parameters' vector. The choice of the filter should be done off-line and the condition presented is enough to ensure the convergence of the algorithm. Future developments are needed to provide the best filter option (assuring the fastest convergence with robustness) and to introduce some constraints in the model predictive applications. The main benefit expected in case of the distributed MPC control is the improvement of the system's maitainability and the 'apparent' simplicity to the user.

References

[1] Camacho, E. F., "Model Predictive Control", Springer Verlag, 2004.

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

[3] Venkat, A. N., Rawlings, J. B. and Wright, S. J., "Implementable distributed model predictive control with guaranteed performance properties", *American Control Conference Minneapolis, Minnesota, USA*, June 14-16, 2006, pp. 613-618.

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

[5] Plucenio, A., Pagano, D. J., Camponogara, E., Sherer, H. F. and Lima, M., "A simple distributed MPC algorithm", Rio de Janerio, Brasil.

[6] Maestre, J. M., Munoz de la Pena, D. and Camacho, E. F., "Distributed MPC: a supply chain case study", *IEEE Conference on Decision and Control, Shanghai, China*, December 16-18, 2009, pp. 7099 – 7104.

[7] Venkat, A. N., Hiskens, I. A., Rawlings, J. B. and Wright, S. J. "Distributed MPC Strategies With Application to Power System Automatic Generation Control", *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192-1206, November, 2008.