An Online Algorithm for Robust Distributed Model Predictive Control

Walid Al-Gherwi. Hector Budman. Ali Elkamel

Chemical Engineering Department, University of Waterloo, Waterloo, Canada

Abstract: Distributed Model Predictive Control (DMPC) has received significant attention in the literature. However, the robustness of DMPC with respect to model errors has not been explicitly addressed. In this paper, an online algorithm that deals explicitly with model errors for DMPC is proposed. The algorithm requires decomposing the entire system into *N* subsystems and solving *N* convex optimization problems to minimize an upper bound on a robust performance objective by using a time-varying state-feedback controller for each subsystem. Simulations on two typical examples were considered to illustrate the application of the proposed method.

Keywords: Distributed Model Predictive Control; Robust Control.

1. INTRODUCTION

Distributed model predictive control (DMPC) has received significant attention in the literature in recent years. The key potential advantages of DMPC are: i) it can provide better performance than fully decentralized control especially when the interactions ignored in the latter are strong, and ii) it can maintain the flexibility with respect to equipment failure and partial plant shutdowns that may jeopardize the successful operation of centralized MPC. The basic idea of DMPC is to partition the total system of states and controlled and manipulated variables into smaller subsystems and to assign an MPC controller to each subsystem. The design of all the reported DMPC strategies is composed of three parts: (1) Modeling; each controller has access to a local dynamic model of the corresponding subsystem along with an interaction dynamic model that represents the influence of the other subsystems. These models can be obtained by directly decomposing a centralized model of the process (Rawlings and Stewart 2008). (2) Optimization; each MPC solves a local optimization problem. Some reported strategies use modified objective functions that take into account the goals of other controllers to achieve full coordination (Venkat 2006; Zhang and Li 2007) whereas some others use strict local objectives (Li *et al.* 2005), e.g. a Nash-equilibrium objective. (3) Communication; at every control time interval all the controllers exchange the measurements of their local states that are used for subsequent local optimization. These 3 steps are executed at each time interval in an iterative manner until convergence among the controllers is reached. Venkat (2006) showed that increasing the iterations allows the DMPC strategy to reach the optimal centralized solution and the termination at any intermediate iteration maintains system-wide feasibility. Zhang and Li (2007) analyzed the optimality of the iterative DMPC scheme and derived closed-form solution for an unconstrained DMPC and showed that it is identical to centralized MPC solution. The common feature of the reported strategies is that they employ a nominal model of the plant and rely on feedback to account for plant-model mismatch. However, plant–model mismatch may have a significant impact on stability and performance. Thus, the robustness of DMPC to model errors has been identified as a key factor for a successful application of DMPC (Rawlings and Stewart 2008). Kothare *et al.* (1996) proposed a methodology for robust centralized constrained MPC design that maintains robust stability and minimizes a bound on performance in the presence of model errors. The problem is formulated as a convex optimization problem with linear matrix inequalities LMI that is solved efficiently using available algorithms (Boyd *et al.* 1994) and can be used for on-line implementations. This method has been recognized as a good potential candidate for use in process industry to handle the issue of plant-model mismatch (Qin and Badgwell 2003).

The aim of this paper is to present a methodology for Robust DMPC (RDMPC) that explicitly deals with model errors. An LMI-based predictive control formulation (Kothare *et al.* 1996) has been modified to design an on-line iterative algorithm for RDMPC. Issues of robust stability and convergence are analyzed and discussed. Two case studies are used to illustrate the algorithm: a distillation column example (Venkat 2006) when "bad" input-output pairings are chosen and a high-purity column example (Skogestad and Morari, 1988) with high condition number.

2. Definitions and Methodology

2.1 Models

In this work, it is assumed that the process model is given by a linear time-varying (LTV) model of the form:

$$\boldsymbol{x}(k+1)=\boldsymbol{A}(k)\boldsymbol{x}(k)+\boldsymbol{B}(k)\boldsymbol{u}(k) \qquad (1)$$

where the real plant lies within a polytope that is represented by the convex hull:

$$[\boldsymbol{A}(k)\ \boldsymbol{B}(k)] = \sum_{l=1}^{L} \beta_l [\boldsymbol{A}^{(l)}\ \boldsymbol{B}^{(l)}] \quad ; \sum_{l=1}^{L} \beta_l = 1;\ \beta_l \geq 0 \qquad (2)$$

Each vertex *l* corresponds to a linear model obtained from linearizing a nonlinear model or identification of a linear model in the neighbourhood of a particular operating point. It is assumed that the states are fully measured. The states and the controlled and manipulated variables in model (1) can be decomposed into *N* subsystems as follows:

$$\begin{bmatrix} \boldsymbol{x}_{11}(k+1) \\ \vdots \\ \boldsymbol{x}_{ii}(k+1) \\ \vdots \\ \boldsymbol{x}_{NN}(k+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{11}(k) & \cdots & \cdots & \cdots & \boldsymbol{A}_{1N}(k) \\ \vdots & \ddots & & & \vdots \\ \boldsymbol{A}_{i1}(k) & \cdots & \ddots & \cdots & \boldsymbol{A}_{iN}(k) \\ \vdots & & & \ddots & \vdots \\ \boldsymbol{A}_{N1}(k) & \cdots & \cdots & \cdots & \boldsymbol{A}_{NN}(k) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{11}(k) \\ \vdots \\ \boldsymbol{x}_{ii}(k) \\ \vdots \\ \boldsymbol{x}_{NN}(k) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}_{11}(k) & \cdots & \cdots & \cdots & \boldsymbol{B}_{1N}(k) \\ \vdots & \ddots & & & \vdots \\ \boldsymbol{B}_{i1}(k) & \cdots & \ddots & \cdots & \boldsymbol{B}_{iN}(k) \\ \vdots & & & \ddots & \vdots \\ \boldsymbol{B}_{N1}(k) & \cdots & \cdots & \cdots & \boldsymbol{B}_{NN}(k) \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_{1}(k) \\ \vdots \\ \boldsymbol{u}_{i}(k) \\ \vdots \\ \boldsymbol{u}_{N}(k) \end{bmatrix} \qquad (3)$$

where $i \in \{1,\cdots,N\};\ \boldsymbol{x}_{ii} \in \Re^{n_{ii}};\ \boldsymbol{u}_i \in \Re^{m_i}$. For example, in model (3) the ith controller for the ith subsystem is based on the following model:

$$\boldsymbol{x}_i(k+1) = \boldsymbol{A}_i(k)\boldsymbol{x}_i(k) + \boldsymbol{B}_i(k)\boldsymbol{u}_i(k) + \sum_{\substack{j=1 \\ j\neq i}}^{N} \boldsymbol{B}_j(k)\boldsymbol{u}_j(k) \qquad (4)$$

and similar to the representation given in (2) it is assumed that for the ith subsystem (4):

$$[\boldsymbol{A}_i(k)\ \boldsymbol{B}_i(k) .. \boldsymbol{B}_j(k)..] = \sum_{l=1}^{L} \beta_l [\boldsymbol{A}_i^{(l)}\ \boldsymbol{B}_i^{(l)} .. \boldsymbol{B}_j^{(l)}..] \qquad \forall j \in \{1,...,N\}, j \neq i \qquad (5)$$

where $\boldsymbol{x}_i' = \left[\boldsymbol{x}_{11}', \cdots\cdots, \boldsymbol{x}_{ii}', \cdots\cdots, \boldsymbol{x}_{NN}'\right]'$ is the vector of states of subsystem *i* containing states $\boldsymbol{x}_{ii}$ that can be measured locally augmented with states $\boldsymbol{x}_{jj}$ that affect subsystem *i* measured in the other subsystems and communicated among the subsystems. Therefore the matrix $\boldsymbol{A}_i(k)$ contains all the elements of the matrix $\boldsymbol{A}(k)$. Model (4) also includes the effect of local controller $\boldsymbol{u}_i$ and the other controllers $\boldsymbol{u}_j$ with their corresponding matrices defined as:

$$\begin{aligned} \boldsymbol{B}_i'(k) &= \left[\boldsymbol{B}_{1i}'(k), \cdots\cdots, \boldsymbol{B}_{Ni}'(k)\right]' \\ \boldsymbol{B}_j'(k) &= \left[\boldsymbol{B}_{1j}'(k), \cdots\cdots, \boldsymbol{B}_{Nj}'(k)\right]' \end{aligned} \qquad (6)$$

The model in (4) is general and can be used to represent special limiting cases such as the decentralized case where all the interactions are ignored, e.g. $\boldsymbol{B}_j' = [\boldsymbol{0}]\ \forall j \in \{1,...,N\}, j \neq i$.

2.2 Robust Performance Objective

Kothare *et al.* (1996) proposed a formulation for a centralized problem whereby an upper bound on a robust performance objective is minimized. In the current work a similar formulation is used but the minimization is simultaneously done for every subsystem *i* defined by (4) for which the following min-max problem is solved:

$$\begin{aligned} &\min_{\boldsymbol{u}_i(k+n|k)} \ \max_{[\boldsymbol{A}_i(k+n)\ \boldsymbol{B}_i(k+n)\ \boldsymbol{B}_j(k+n)], n\geq 0} J_i(k) \\ &s.t. \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (7) \\ &\qquad \left|\boldsymbol{u}_i(k+n|k)\right| \leq \boldsymbol{u}_i^{max},\ n \geq 0 \end{aligned}$$

In general, the local objective $J_i(k)$ is defined as follows:

$$\begin{aligned} J_i(k) = \sum_{n=0}^{\infty} [&\boldsymbol{x}_i'(k+n|k)\mathbb{Q}_i\boldsymbol{x}_i(k+n|k) \\ &+ \boldsymbol{u}_i'(k+n|k)\boldsymbol{R}_i\boldsymbol{u}_i(k+n|k) \\ &+ \sum_{\substack{i=1 \\ i\neq j}}^{N} \boldsymbol{u}_j^{\bullet\prime}(k+n|k)\boldsymbol{R}_j\boldsymbol{u}_j^{\bullet}(k+n|k)] \end{aligned} \qquad (8)$$

where $\mathbb{Q}_i > 0$, $\boldsymbol{R}_i > 0$, $\boldsymbol{R}_j > 0$. The local objective given in (8) takes into account the goals of the other controllers, third summation in the RHS, in order to achieve the global objective of the entire system. The superscript "•" indicates that the solution was obtained in a previous iteration and remains fixed in the current iteration as will be explained later. It should be pointed out that one can easily modify the problem in (8) to solve particular objectives such as Nash equilibrium or decentralized control. Both strategies are based on minimizing strictly local objectives of the subsystems. The difference is that for Nash the interaction information is shared among the subsystems while for decentralized control the interaction information is neglected. Accordingly, for both Nash and decentralized control $\mathbb{Q}_i$ and $\boldsymbol{R}_i$ in (8) are modified to contain all zeros except for the weights corresponding to the local subsystem and the third summation in the RHS of (8) is excluded. On the other hand the interaction term in (4) is included for Nash but it is ignored for decentralized control.

Since the objective in (8) has an infinite horizon, the problem of finding infinite $\boldsymbol{u}_i$ is computationally intractable. Instead, a state-feedback law is sought for each subsystem *i* as follows:

$$\begin{aligned} \boldsymbol{u}_i(k+n|k) &= \boldsymbol{F}_{ii}\boldsymbol{x}_{ii}(k+n|k) + \sum_{\substack{j=1 \\ j\neq i}}^{N} \boldsymbol{F}_{ij}\boldsymbol{x}_{ij}(k+n|k) \\ &= \boldsymbol{F}_i\boldsymbol{x}_i(k+n|k) \end{aligned} \qquad (9)$$

similarly,

$$\begin{aligned} \boldsymbol{u}_j^{\bullet}(k+n|k) &= \boldsymbol{F}_{jj}^{\bullet}\boldsymbol{x}_{jj}(k+n|k) + \sum_{\substack{i=1 \\ i\neq j}}^{N} \boldsymbol{F}_{ji}^{\bullet}\boldsymbol{x}_{ji}(k+n|k) \\ &= \boldsymbol{F}_j^{\bullet}\boldsymbol{x}_i(k+n|k) \end{aligned} \qquad (10)$$

Using these state-feedback laws in (4) leads to the following closed loop model:

$$\boldsymbol{x}_i(k+1) = \left(\tilde{\boldsymbol{A}}_i(k) + \boldsymbol{B}_i(k)\boldsymbol{F}_i\boldsymbol{u}_i(k)\right)\boldsymbol{x}_i(k) \qquad (11)$$

where $\tilde{A}_i(k) = A_i(k) + \sum_{\substack{j=1\\ j\neq i}}^{N} B_j(k) F_j^{\bullet}$

It is assumed that there exists a quadratic function $V_i(k) = \boldsymbol{x}_i'(k)\boldsymbol{P}_i\boldsymbol{x}_i(k)$, $\boldsymbol{P}_i > 0$, so that, for any plant in (6), this function satisfies the following stability constraint:

$$\begin{aligned} V_i(k+n+1|k) - V_i(k+n|k) \le -[&\boldsymbol{x}_i'(k+n|k)\mathbb{Q}_i\boldsymbol{x}_i(k+n|k) \\ &+ \boldsymbol{u}_i'(k+n|k)\boldsymbol{R}_i\boldsymbol{u}_i(k+n|k) \\ &+ \sum_{\substack{i=1\\ i\neq j}}^{N} \boldsymbol{u}_j^{\bullet\prime}(k+n|k)\boldsymbol{R}_j\boldsymbol{u}_j^{\bullet}(k+n|k)] \\ & n \ge 0 \end{aligned} \qquad (12)$$

Using (11), the robust stability constraint in (12) becomes:

$$\begin{aligned} V_i(k+n+1|k) - V_i(k+n|k) \le -[&\boldsymbol{x}_i'(k+n|k)\tilde{\mathbb{Q}}_i\boldsymbol{x}_i(k+n|k) \\ &+ \boldsymbol{u}_i'(k+n|k)\boldsymbol{R}_i\boldsymbol{u}_i(k+n|k)] \end{aligned}$$

where $\tilde{\mathbb{Q}}_i = \mathbb{Q}_i + \sum_{\substack{i=1\\ i\neq j}}^{N} \boldsymbol{F}_j^{\bullet\prime}(k+n|k)\boldsymbol{R}_j\boldsymbol{F}_j^{\bullet}(k+n|k)$ (13)

which, for all $n \ge 0$, turns out to be:

$$\begin{aligned} \left[\tilde{\boldsymbol{A}}_i(k+n) + \boldsymbol{B}_i(k+n)\boldsymbol{F}_i\right]' \boldsymbol{P}_i \left[\tilde{\boldsymbol{A}}_i(k+n) + \boldsymbol{B}_i(k+n)\boldsymbol{F}_i\right] \\ - \boldsymbol{P}_i + \boldsymbol{F}_i'\boldsymbol{R}_i\boldsymbol{F}_i + \tilde{\mathbb{Q}}_i \le 0 \end{aligned} \qquad (14)$$

By defining an upper bound, i.e.

$$J_i(k) \le \boldsymbol{x}_i'(k)\boldsymbol{P}_i\boldsymbol{x}_i(k) = V_i(k) \le \gamma_i \qquad (15)$$

and substituting the parameterization $\boldsymbol{F}_i = \boldsymbol{Y}_i'\boldsymbol{Q}_i^{-1}$, $\boldsymbol{Q}_i = \gamma_i\boldsymbol{P}_i^{-1}$, followed by performing Schur complements (Boyd et al. 1994) on (14) and (15) it can be easily shown that the minimization of $J_i(k)$ is equivalent to the minimization of its upper bound γ_i as in the following linear minimization problem with LMI constraints (Kothare *et al.* 1996):

$$\min_{\gamma_i, \boldsymbol{Q}_i, \boldsymbol{Y}_i} \gamma_i$$

s.t.

$$\begin{bmatrix} 1 & \boldsymbol{x}_i'(k) \\ \boldsymbol{x}_i(k) & \boldsymbol{Q}_i \end{bmatrix} \ge 0$$

$$\begin{bmatrix} \boldsymbol{Q}_i & \boldsymbol{Q}_i\tilde{\boldsymbol{A}}_i'^{(l)} + \boldsymbol{Y}_i'\boldsymbol{B}_i'^{(l)} & \boldsymbol{Q}_i\tilde{\mathbb{Q}}_i^{1/2} & \boldsymbol{Y}_i'\boldsymbol{R}_i^{1/2} \\ * & \boldsymbol{Q}_i & \boldsymbol{0} & \boldsymbol{0} \\ * & * & \gamma_i\boldsymbol{I} & \boldsymbol{0} \\ * & * & * & \gamma_i\boldsymbol{I} \end{bmatrix} \ge 0$$

$$\forall l \in \{1,\dots,L\} \qquad (16)$$

$$\begin{bmatrix} \left(\boldsymbol{u}_i^{max}\right)^2\boldsymbol{I} & \boldsymbol{Y}_i \\ \boldsymbol{Y}_i' & \boldsymbol{Q}_i \end{bmatrix} \ge 0$$

The key difference between the centralized control algorithm proposed by Kothare *et al.* (1996) and the distributed strategy proposed in this work is that every controller in the set $i \in \{1,\cdots,N\}$ solves a local problem as in (16) and then the solutions are exchanged in an iterative scheme that is further explained in the next subsection. It should be remembered that one of the key reasons to use distributed MPC strategies is to address real time computation issues when dealing with large-scale processes (Li *et al.* 2005). Although the proposed iterative scheme tends to increase the computational time, the problem defined in (16) is numerically advantageous as compared to solving the same problem for the whole system (centralized control). The reason is that the state feedback controller for each subsystem i is obviously of smaller dimensions than a state feedback controller of the centralized MPC strategy. For instance, $\boldsymbol{Y}_i$ for subsystem i is of dimension ($n \times m_i$) instead of ($n \times m$) for centralized system where m is the total number of manipulated variables of the entire process.

2.3 Robust DMPC Algorithm

This section presents the main result of the paper where an on-line algorithm for RDMPC is proposed. It is assumed that there is an ideal communication network available so that the controllers can exchange their information with no delays. The goal of performing communication and exchanging solutions among controllers is to achieve the optimal solution of the entire system in an iterative fashion. The algorithm proceeds according to the Jacobi iteration method used for the solution of systems of algebraic equations. The procedure is summarized in *Algorithm 1* below.

Algorithm 1 (RDMPC)

Step0 (initialization)*: at control interval k=0 set* $\boldsymbol{F}_i$=0.

Step1 (updating) *at control interval (k) all the controllers exchange their local states measurements and initial estimates* $\boldsymbol{F}_i$*'s via communication, set iteration t = 0 and* $\boldsymbol{F}_i = \boldsymbol{F}_i^{(0)}$.

Step2 (iterations)

```
while $t \le t_{max}$
Solve all N LMI problems (16) in parallel to obtain the minimizers $\boldsymbol{Y}_i^{(t+1)}, \boldsymbol{Q}_i^{(t+1)}$ to estimate the feedback solutions $\boldsymbol{F}_i^{(t+1)} = \boldsymbol{Y}_i'^{(t+1)}\boldsymbol{Q}_i^{-1(t+1)}$. If problem is infeasible set $\boldsymbol{F}_i^{(t)} = \boldsymbol{F}_i^{(t-1)}$. Check the convergence for a specified error tolerance $\varepsilon_i$ for all the controllers
    if $\left\|\boldsymbol{F}_i^{(t+1)} - \boldsymbol{F}_i^{(t)}\right\| \le \varepsilon_i \quad \forall i \in \{1,\dots,N\}$
        break
    end if
Exchange the solutions ($\boldsymbol{F}_i$'s) and set $t = t + 1$
end while
```

Step3 (implementation) *apply the control actions* $\boldsymbol{u}_i = \boldsymbol{F}_i\boldsymbol{x}_i$ *to the corresponding subsystems, increase the control interval* $k = k + 1$*, return to step1 and repeat the procedure.*

Algorithm1 is implemented in MATLAB® and problem (16) is solved via MATLAB® LMI solver. Convergence of the iterations in Step 2 and stability properties are discussed in the following subsection.

2.4 Convergence and Robust Stability Analysis of RDMPC Algorithm

Regarding convergence, it can be shown that at each time interval, each one of the *N* convex problems defined in *Algorithm1* will converge to the same solution which is the solution of the centralized problem, i.e. $\gamma_1 = \gamma_2 = \cdots = \gamma_i = \cdots = \gamma_N = \gamma$ where γ is the performance upper bound of centralized MPC. For brevity, a two subsystem situation, i.e. *N*=2 is considered without loss of generality. It is also assumed that the solutions are feasible.

Define:

$$\textit{for subsystem 1} \quad \gamma_1^{(t)} = \min_{\boldsymbol{F}_1^{(t)}} \gamma_1(\boldsymbol{F}_1^{(t)}, \boldsymbol{F}_2^{(t-1)})$$

$$\textit{for subsystem 2} \quad \gamma_2^{(t)} = \min_{\boldsymbol{F}_2^{(t)}} \gamma_2(\boldsymbol{F}_1^{(t-1)}, \boldsymbol{F}_2^{(t)})$$

$$\textit{Then,} \ \gamma_1^{(t)} \le \gamma_2^{(t-1)} \qquad \text{(a)}$$

and the reason being that both sides of this inequality are using the same value of $\boldsymbol{F}_2 = \boldsymbol{F}_2^{(t-1)}$ but the LHS minimizes γ with respect to $\boldsymbol{F}_1$ whereas the RHS of the inequality uses a not necessarily optimal value of $\boldsymbol{F}_1 = \boldsymbol{F}_1^{(t-1)}$. Following the same argument:

$$\gamma_2^{(t)} \le \gamma_1^{(t-1)} \qquad \text{(b)}$$

Thus, the γ_i's decrease until (a) or (b) become equalities. Since the minimizations are convex and lead to global optimal solutions, this occurs only when $\boldsymbol{F}_1^{(t)} = \boldsymbol{F}_1^{(t-1)}$ and $\boldsymbol{F}_2^{(t)} = \boldsymbol{F}_2^{(t-1)}$ and consequently $\gamma_{sub1}^{(t)} = \gamma_{sub2}^{(t)} = \gamma$, i.e. the minimization with respect to both $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ give the same solution which must be, following convexity of problem (16), equal to the global optimum of the centralized control problem that has an identical formulation to (16). The robust stability of *Algorithm1* follows from the fact that for each subsystem, a robust stability related constraint is enforced by one of the linear matrix inequalities in problem (16). Thus each one of the *N* controllers satisfies robust stability. Although theoretical convergence of the Jacobi iteration can be proven, it was found that numerical noise exists due to inaccuracies of the LMI solvers in obtaining the solution of problem (16). Consequently, to speed up convergence in the presence of this numerical noise when *Algorithm1* is implemented, the *successive Relaxation* (SR) method is employed (Hageman and Young 1981). The SR method is applied to the solution obtained from (16) for each subsystem to estimate a weighted average between the current and previous iterate solutions. The method is given by the following recurrence formula:

$$\boldsymbol{F}_i^{(t+1)} = \alpha\bar{\boldsymbol{F}}_i^{(t+1)} + (1-\alpha)\boldsymbol{F}_i^{(t)} \qquad (17)$$

where α is a parameter to be specified by the user in order to accelerate convergence. $\bar{\boldsymbol{F}}_i^{(t+1)}$ denotes the solution obtained at the current iteration from (16) whereas $\boldsymbol{F}_i^{(t+1)}$ is the estimate to be used in the next iteration. Typically, α can be chosen from values between 0 and 2 and when it is set to 1 the normal iterative scheme is retrieved. Since there is no systematic way to select a value for α in advance, simulations with different values of α have to be performed as shown in the first example.

3. Case Studies

3.1 Example 1

A distillation column control problem studied by Venkat (2006) is considered with the difference that uncertainties in the steady-state gains of the model are added to illustrate the robustness of the proposed algorithm. Accordingly, the real model lies within a polytope defined within the two vertices:

$$G_1 = \begin{bmatrix} \dfrac{32.63}{(99.6s+1)(0.35s+1)} & \dfrac{-33.89}{(98.02s+1)(0.42s+1)} \\ \dfrac{34.84}{(110.5s+1)(0.03s+1)} & \dfrac{-18.85}{(75.43s+1)(0.3s+1)} \end{bmatrix}$$
$$G_2 = \begin{bmatrix} \dfrac{326.3}{(99.6s+1)(0.35s+1)} & \dfrac{-338.9}{(98.02s+1)(0.42s+1)} \\ \dfrac{348.4}{(110.5s+1)(0.03s+1)} & \dfrac{-188.5}{(75.43s+1)(0.3s+1)} \end{bmatrix} \qquad (21)$$

A state-space model, not shown for brevity, is obtained from a canonical realization of equation (21). To demonstrate the effectiveness of the proposed method the bad pairings, according to the *Relative Gain Array* RGA, are selected, i.e. the RGA element λ_{11} is -1.0874 and accordingly the "bad" pairings are u_1-y_1 (*subsystem1*) and u_2-y_2 (*subsystem2*). The physical constraints on manipulated variables are given by:

$$|u_1(k+n)| \le 1.5; \ |u_2(k+n)| \le 2; \ n \ge 0 \qquad (22)$$

For the purpose of comparison between different cases, a cost function is defined as follows:

$$J_{cost} = (1/2Ns)\sum_{j=0}^{Ns}\sum_{i=1}^{N}\left(\boldsymbol{x}_i'(j)\mathbb{Q}_i\boldsymbol{x}_i(j) + \boldsymbol{u}_i'(j)\boldsymbol{R}_i\boldsymbol{u}_i(j)\right) \qquad (23)$$

where *Ns* is the simulation time. The following parameters are used for the two controllers: $\mathbb{Q}_{y1} = \mathbb{Q}_{y2} = 50$ so that $\mathbb{Q}_i = \boldsymbol{C}_i'\mathbb{Q}_{yi}\boldsymbol{C}_i + 10^{-6}\boldsymbol{I}$ where $\boldsymbol{C}_i$ is the measurement matrix such that $\boldsymbol{y}_i = \boldsymbol{C}_i\boldsymbol{x}_i$; $\boldsymbol{R}_1 = \boldsymbol{R}_2 = 1$; $\alpha = 0.95$. The value of α is selected, as mentioned above, based on simulations by trial and error to speed convergence of the Jacobi iteration. The number of iterations that was required to satisfy the convergence criteria of *Algorithm1* for different values of α

is given in Table1. $\alpha=0.95$ resulted in the fastest convergence.

Table 1. Effect of α on convergence with $\varepsilon_1=\varepsilon_2=10^{-3}$

α	# iterations
1.05	55
1.00	38
0.95	28
0.90	32
0.8	38

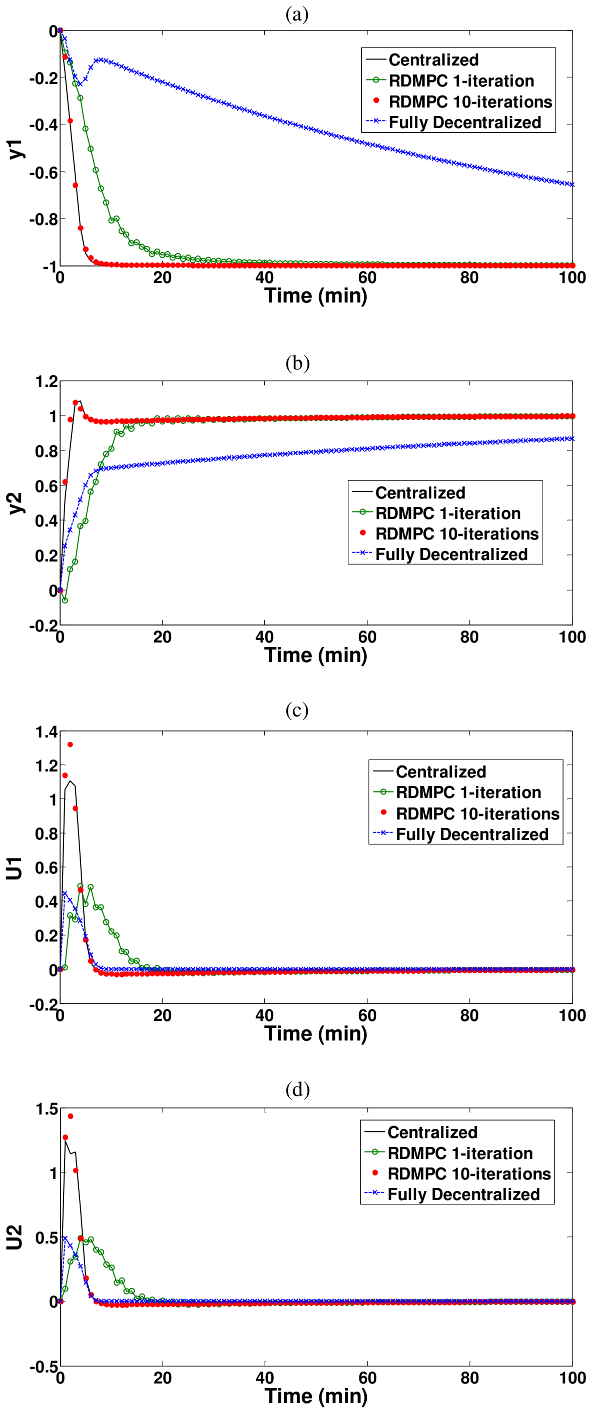

Fig. 1. Dynamic response in controlled and manipulated variables for set-point changes in y1 and y2.

Three cases are considered for the application of *Algorithm1*; fully decentralized, RDMPC with one iteration, and RDMPC with 10 iterations. It should be remembered that as indicated in section 2, the cost γ decreases monotonically with the number of iterations. Thus, even after one iteration, a performance improvement is expected. The motivation for using a small number of iterations, as mentioned earlier, is to use distributed MPC strategies to address real time computation issues when dealing with large scale processes. The decentralized strategy used in this study is obtained, as explained in Section 2.2, with *Algorithm1* by ignoring interactions in equation (4). Then, the performance of *Algorithm1* with these 3 different schemes was compared to the centralized strategy in Figure1. The simulations correspond to simultaneous changes in set-points of both controlled variables y1 and y2 by -1 and 1; respectively.

In comparison with the centralized scheme, the performance of RDMPC approaches that of the centralized scheme as the number of iterations is increased. The fully decentralized case resulted as expected in the worst performance. A comparison of the cost in (23) for different schemes is given in Table2. This table illustrates that *Algorithm1* can be used, depending on the chosen number of iterations, to obtain a performance that varies between two extremes corresponding to the fully decentralized and the centralized strategies; respectively. It is also clear, from figures 1(c) and 1(d), that the constraints given in (22) are satisfied.

Table 2. Cost for different strategies (example1)

Strategy	Cost (23)
Centralized	0.92
RDMPC (10 iteration)	0.93
RDMPC (1 iteration)	2.43
Fully decentralized	35.9

3.2 Example 2

This example considers the high-purity column originally studied by Skogestad and Morari (1988). The nominal transfer function of this system is given by:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{1}{75s+1}\begin{bmatrix} 0.878 & 0.864 \\ 1.082 & 1.096 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad (24)$$

A state-space model is obtained based on a canonical realization of equation (24) and not shown for brevity. Due to the high condition number, this process has been used in the past to illustrate closed-loop sensitivity to model errors. The model uncertainty is given by errors in steady-state gains. The gains in the first column of the transfer matrix in (24) are expected to change by up to +80% whereas the gains in the second column are expected to change by up to -80%. The constraints on manipulated variables are represented by $|\mathbf{u}(k+n)| \le 1$, $n\geq 0$. The system given above was decomposed into two subsystems; *viz.*, y_1-u_1 (*subsystem1*) and y_2-u_2 (*subsystem2*). The controllers parameters used in simulation are; $\mathbb{Q}_1 = \mathbb{Q}_2 = 1$, $R_1 = R_2 = 1$, $\alpha=1$, $\varepsilon_1=\varepsilon_2=10^{-2}$.

Figure 2 depicts the performance of *Algorithm1* compared with centralized MPC for a unit set-point change in y_1 and it illustrates that RDMPC algorithm results in an identical response as the centralized MPC. For this example, the RDMPC algorithm converges very quickly in about three iterations after which the error tolerances specified above ($\varepsilon_1=\varepsilon_2=10^{-2}$) are met. Figure 3 shows the convergent behaviour of the RDMPC algorithm obtained in the first sampling interval. The upper bounds γ_1 and γ_2 for subsystems 1 and 2 respectively, obtained by solving (16) in parallel and by applying *Algorithm1*, converge to the same value after about 3 iterations and this value is identical to that obtained for centralized MPC. The cost, defined by equation (23), for both strategies, is equal to *7.48*. To show the ability of the method to deal with different objective functions an RDMPC with a Nash equilibrium objective and a robust decentralized MPC were designed by proper choice of the weights $\mathbb{Q}_i$ and $\boldsymbol{R}_i$ as explained in section 2.2. The results with the Nash-equilibrium based controller, shown also in Figure 2, are similar to the centralized case and the cost was *7.55*, slightly larger than the centralized MPC cost. The decentralized MPC, not shown in the Figure, resulted as expected in a slightly higher cost than Nash of *7.73*.

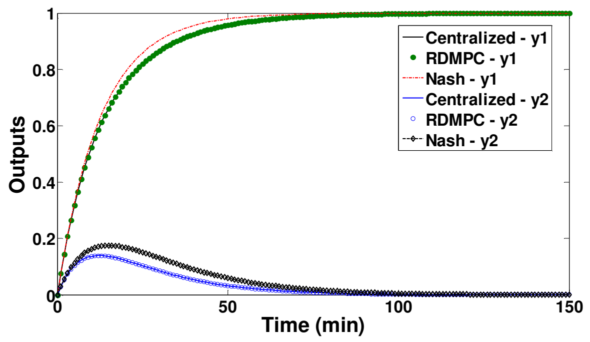

Fig. 2. Dynamic response to unit set-point change in y1.

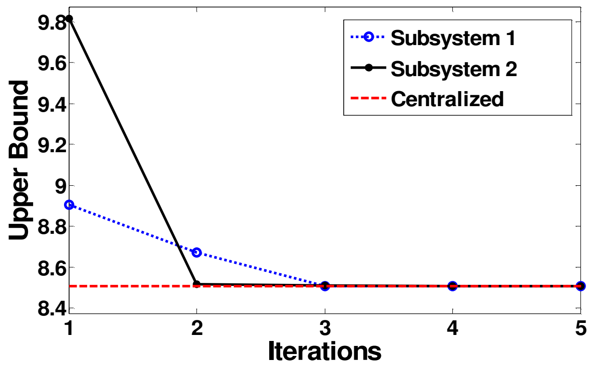

Fig. 3. Convergence characteristics of *Algorithm1* at the first sampling time.

4. CONCLUSIONS

The main goal of this work was to propose an on-line algorithm for DMPC strategy that explicitly considers model errors. The main idea of the proposed method is to decompose the model of the whole system into N subsystems and then obtain a local state feedback controller by minimizing an upper bound on a robust performance objective for each subsystem. The subsystem performance takes into account the objectives of the other subsystems in order to achieve the goal of the entire system. The method was also suitable for pursuing other objectives such as Nash equilibrium or decentralized control in the presence of model errors. The problem was converted into N convex problems with linear matrix inequalities and solved iteratively by using the Jacobi iteration method with successive relaxation (SR). Although convergence of the iterative solution was proven, the SR feature was helpful for filtering numerical noise in the LMI solutions resulting in faster convergence. When convergence was reached, the algorithm led to the same solution of the centralized MPC problem. The examples showed that RDMPC can achieve, after a sufficient number of iterations, equivalent performance to centralized control. Moreover, the examples illustrated that improvements in RDMPC performance as compared to decentralized control can be achieved with a relatively small number of iterations.

ACKNOWLEDGMENT

The first author would like to thank the General Secretariat of Higher Education of Libya for financial support. Also, the authors acknowledge additional support of Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

Boyd, S., El.Ghaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*, SIAM, USA.

Hageman, L. and Young, D. (1981) *Applied iterative methods.* Academic Press, NY.

Kothare, M. V., Balakrishnan, V., and Morari, M. (1996) Robust constrained model predictive control using linear matrix inequalities. *Automatica*, **32** (10), 1361-1379.

Li, S., Zhang, Y., and Zhu, Q. (2005). Nash-optimization enhanced distributed model predictive control applied to the shell benchmark problem. *Information science,* **170,** 329-349.

Qin, S. J. and Badgwell, T. A. (2003). A survey of industrial model predictive control technology. *Control engineering practice,* **11,** 733-764.

Rawlings, James B. and Stewart, Brett T. (2008). Coordinating multiple optimization-based controllers: new opportunities and challenges. *Journal of Process Control,* **18,** 839-845.

Skogestad, S. and Morari, M. (1988). Robust control of ill-conditioned plants: high-purity distillation. *IEEE transactions on automatic control,* **12,** 1092-1104.

Venkat, A. N. (2006). *Distributed model predictive control: theory and applications*, PhD thesis, USA.

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

.