LONG HORIZON MODEL PREDICTIVE CONTROL FOR NONLINEAR INDUSTRIAL PROCESSES

Andrei A. Tiagounov [*,1] **Jeroen Buijs** [†,2] **Siep Weiland** [*,1]
Bart De Moor [†,2]

[*] *Department of Electrical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*
[†] *Department of Electrical Engineering, ESAT-SISTA, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, 3001 Heverlee (Leuven), Belgium*

Abstract: This paper considers MPC algorithm for nonlinear plants. The MPC problem amounts to a quadratic programming problem. A structured interior-point method (IPM) has been proposed to solve the MPC optimization problem with longer prediction horizons. We compare the proposed method with standard QP solvers. The choice of QP optimizer is investigated for two nonlinear industrial cases, namely evaporation process and high-purity distillation column. The effectiveness of the structured IPM is demonstrated for long horizon MPC controller design of the first process.

Keywords: Model predictive control, quadratic programming, nonlinear processes.

1. INTRODUCTION

Model Predictive Control (MPC), also known as moving or receding horizon control, has originated in industry as a real-time computer control algorithm to solve linear multi-variable problems that have constraints and time delays. MPC has received a great deal of attention and receive an ever growing interest for applications in industrial process control. Various MPC algorithms differ mainly in the type of model used to represent the process and its disturbances, as well as the cost functions to be minimized subject to constraints. In this paper we consider a nonlinear MPC scheme where the predicted future process behaviour is represented as a cumulative effect of a nonlinear prediction component and a component based on linear models defined along the predicted trajectory (de Keyser, 1998; Lee and Ricker, 1994; Santos *et al.*, 1995). The first component constitutes a future output prediction using nonlinear simulation models, given past process inputs and measured disturbance signals. The second component uses linearized models for prediction of future process outputs as required for calculation of optimum future process inputs that bring the process behavior closest to the desired behavior. The online optimization can be typically reduced to either a linear program or a quadratic program (QP). MPC controller solves online a constrained optimization problem and determines optimal control inputs over a fixed future time-horizon, based on the predicted future behaviour of the process, and based on the desired reference trajectory. Certain problems may require longer prediction horizons to cover broader range of dynamics, due to performance specifications or stability reasons (Mayne *et al.*, 2000; Muske and Rawlings, 1993; Qin and Badgwell, 1997; Rawlings and Muske, 1993). Sometimes it is rather difficult to implement long horizon MPC controllers online, since these optimization programs (typically constrained QPs) tend to become too large to be solved in real-time when standard QP solvers are used. To reduce computational complexity we propose

[1] email: {a.tiagounov, s.weiland}@tue.nl, fax: +31 40 243 45 82
[2] email: {jeroen.buijs, bart.demoor}@esat.kuleuven.ac.be, fax: +32 16 32 19 70

to solve QP problem using a structured interior-point method (Rao *et al.*, 1998; Buijs *et al.*, 2002). The cost of this approach is linear in the horizon length, compared with the cubic growth for the standard approach. It will be shown that in some cases the structured method can solve MPC problem much faster for longer prediction horizons than standard QP solvers. The effectiveness of the proposed method will be demonstrated for an industrial evaporation process. At the same time the structured IPM will not prove to be a good choice for a distillation column.

2. PROBLEM FORMULATION

Given system equations of the nonlinear process model

$$\dot{x} = f(x,u),\ u \in \mathbb{R}^{n_\mathrm{u}} \tag{1}$$

$$y = g(x),\ y \in \mathbb{R}^{n_\mathrm{y}}, \tag{2}$$

consider an MPC problem which amounts to find an optimal control sequence $\{u(k+j)\}_{j=0}^{N-1}$ at the current moment k minimizing the performance index:

$$J(u) = \tag{3}$$
$$\sum_{j=1}^{N}\big[(y(j)-y_{\mathrm{ref}}(j))^\top Q(y(j)-y_{\mathrm{ref}}(j)) +$$
$$(u(j-1)-u_{\mathrm{ref}}(j-1))^\top R(u(j-1)-u_{\mathrm{ref}}(j-1))\big]$$

subject to the constraints

$$y_{\min} \le\ y(k+j)\ \le y_{\max} \tag{4}$$

$$u_{\min} \le\ u(k+j)\ \le u_{\max} \tag{5}$$

$$\Delta u_{\min} \le \Delta u(k+j) \le \Delta u_{\max} \tag{6}$$

for $j = 0,\ldots,N-1$, where $\Delta u(k+j) = u(k+j) - u(k+j-1)$. The MPC is trying to make inputs and outputs follow its reference trajectories $u_{\mathrm{ref}}(k+j)$ and $y_{\mathrm{ref}}(k+j)$. The design parameters include prediction horizon N and weighting matrices $Q \in \mathbb{R}^{n_\mathrm{y}}$, $R \in \mathbb{R}^{n_\mathrm{u}}$.

3. NONLINEAR MPC ALGORITHM

For digital MPC controller design, control signal can be assumed to be constant between the sampling intervals. Hence, we can express a discrete version of the model (1) and (2) as

$$x(k) = F_\mathrm{T}(x(k-1),u(k-1))$$
$$y(k) = g(x(k)),$$

where $F_\mathrm{T}(x(k-1),u(k-1))$ denotes the terminal state vector obtained by integrating the ordinary differential equation (ODE) (1) for one sample interval (T) with the initial condition of $x(k-1)$ and constant input $u = u(k-1)$. Thus, the one-step ahead prediction of the states can be written as

$$x(k+1|k) = F_\mathrm{T}(x(k|k),u(k)), \tag{7}$$

where $F_\mathrm{T}(x(k|k),u(k))$ represents the solution obtained by integrating (1) for one sample interval with $x(k|k)$ obtained, for instance, applying state estimation techniques. Note that the state $x(k+1|k)$ is related to the undecided manipulated variable $u(k)$ through nonlinear integration. This makes the optimization required for the input move computation a nonlinear problem. To prevent this we further approximate the equation by linearizing $F_\mathrm{T}(x(k|k),u(k))$ at some nominal input value $u_{\mathrm{nom}}(k)$ (its choice will be explained later):

$$x(k+1|k) \approx$$
$$F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k)) + B_k(u(k)-u_{\mathrm{nom}}(k)), \tag{8}$$

where B_k representing discretized version of one of the following Jacobians:

$$A_k^\mathrm{c} = \left.\frac{\partial f}{\partial x}\right|_{x(k|k),u_{\mathrm{nom}}(k)} \qquad B_k^\mathrm{c} = \left.\frac{\partial f}{\partial u}\right|_{x(k|k),u_{\mathrm{nom}}(k)}.$$

We can generalize this idea to develop multistep prediction. Note from (7) that

$$x(k+2|k) = F_\mathrm{T}(x(k+1|k),u(k+1)). \tag{9}$$

Note that $x(k+2|k)$ is related in a nonlinear fashion not only to $u(k+1)$, but also to $u(k)$ appearing in expression (8) for $x(k+1|k)$. By appropriate linearization, we would like to derive an approximation that is linear with respect to the undecided inputs $u(k)$ and $u(k+1)$. The linear relationship that best approximates the local behavior can be obtained by linearizing the expression $F_\mathrm{T}(x(k+1|k),u(k+1))$ with respect to $x(k+1|k) = F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k))$ and $u(k+1) = u_{\mathrm{nom}}(k+1)$ as follows:

$$F_\mathrm{T}(x(k+1|k),u(k+1)) \approx F_{2\mathrm{T}}(x(k|k),u_{\mathrm{nom}})$$
$$+A_{k+1}(x(k+1|k) - F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k)))$$
$$+B_{k+1}(u(k+1) - u_{\mathrm{nom}}(k+1)), \tag{10}$$

where A_{k+1} and B_{k+1} representing discretized versions of the following Jacobians:

$$A_{k+1}^\mathrm{c} = \left.\frac{\partial f}{\partial x}\right|_{x=F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k)),u_{\mathrm{nom}}(k+1)}$$

$$B_{k+1}^\mathrm{c} = \left.\frac{\partial f}{\partial u}\right|_{x=F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k)),u_{\mathrm{nom}}(k+1)}.$$

Note that u_{nom} is a piecewise constant input taking, for instance, values of $\{u_{\mathrm{nom}}(k), u_{\mathrm{nom}}(k+1)\}$ during the time interval $[k, k+2]$.

Noting from (8) that $x(k+1|k) - F_\mathrm{T}(x(k|k),u_{\mathrm{nom}}(k)) \approx B_k(u(k)-u_{\mathrm{nom}}(k))$ and substituting the affine approximation (10) into (9), we obtain

$$x(k+2|k) \approx F_{2\mathrm{T}}(x(k|k), u_{\mathrm{nom}}) + \begin{bmatrix} A_{k+1}B_k & B_{k+1} \end{bmatrix} \begin{bmatrix} u(k)-u_{\mathrm{nom}}(k) \\ u(k+1)-u_{\mathrm{nom}}(k+1) \end{bmatrix}.$$

Carrying the same derivations for $x(k+j|k)$, $j = 1, \ldots, N$, we obtain

$$x(k+j|k) \approx F_{\mathrm{jT}}(x(k|k), u_{\mathrm{nom}})) + \begin{bmatrix} \prod_{i=1}^{j-1} A_{k+i}B_k & \prod_{i=2}^{j-1} A_{k+i}B_{k+1} & \ldots & B_{k+j-1} \end{bmatrix} \times \begin{bmatrix} u(k)-u_{\mathrm{nom}}(k) \\ u(k+1)-u_{\mathrm{nom}}(K+1) \\ \vdots \\ u(k+j-1)-u_{\mathrm{nom}}(k+j+1) \end{bmatrix}, \quad (11)$$

where A_{k+i} and B_{k+i} representing discretized versions of the following Jacobians:

$$A^{\mathrm{c}}_{k+i} = \left.\frac{\partial f}{\partial x}\right|_{x=F_{\mathrm{iT}}(x(k|k),u_{\mathrm{nom}}),u_{\mathrm{nom}}(k+i)}$$
$$B^{\mathrm{c}}_{k+i} = \left.\frac{\partial f}{\partial u}\right|_{x=F_{\mathrm{iT}}(x(k|k),u_{\mathrm{nom}}),u_{\mathrm{nom}}(k+i)} \quad (12)$$

and $F_{\mathrm{jT}}(x(k|k), u_{\mathrm{nom}})$ represents the terminal states obtained by integrating (1) for j sampling intervals with initial condition $x(k|k)$ and piecewise constant input u_{nom}.

Note that the expression (11) requires integration of the ODE (1) for j sample time steps into the future and computation of the sensitivity matrices for each of the j sample times (i.e., A_{k+i} and B_{k+i} for $i = 0, \ldots, j-1$). To reduce the computational complexity, the sensitivity matrices A_{k+i} and B_{k+i} are often kept constant at the initial values of A_k and B_k throughout the prediction horizon. To avoid the notational complexity, we will adapt this simplification. Hence, (11) simplifies to

$$x(k+j|k) \approx F_{\mathrm{jT}}(x(k|k), u_{\mathrm{nom}}) + \begin{bmatrix} A_k^{j-1}B_k & A_k^{j-2}B_k & \ldots & B_k \end{bmatrix} \begin{bmatrix} u(k)-u_{\mathrm{nom}}(k) \\ u(k+1)-u_{\mathrm{nom}}(k+1) \\ \vdots \\ u(k+j-1)-u_{\mathrm{nom}}(k+j+1) \end{bmatrix}. \quad (13)$$

Similar techniques as described above can be found in, for example (Lee and Ricker, 1994; Ricker and Lee, 1995).

In order to develop a prediction for the output that is linear with respect to the undecided input moves, we linearize equation (2) with respect to $x(k|k)$:

$$y(k) \approx g(x(k|k)) + C_k(x(k) - x(k|k)),$$

where C_k is a Jacobian matrix defined as

$$C_k = \left.\frac{\partial g}{\partial x}\right|_{x(k|k)}.$$

Since the last two terms in the right-hand side have an expectation of zero, then $y(k|k) = g(x(k|k))$. Carrying out with the same idea,

$$y(k+j|k) = g(x(k+j|k)) \approx g(x(k|k)) + C_k(x(k+j|k) - x(k|k)). \quad (14)$$

Combining (14) with the optimal multistep prediction equation (13) we developed for $x(k+j|k)$ and substituting $u(k+j) - u_{\mathrm{nom}}(k+j)$ with so-called optimizing control input $\delta u(k+j)$, we obtain:

$$\begin{bmatrix} y(k+1|k) \\ y(k+2|k) \\ \vdots \\ y(k+N|k) \end{bmatrix} = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix} (g(x(k|k)) - C_k x(k|k)) + \begin{bmatrix} C_k F_{\mathrm{T}}(x(k|k),u_{\mathrm{nom}}) \\ C_k F_{2\mathrm{T}}(x(k|k),u_{\mathrm{nom}}) \\ \vdots \\ C_k F_{\mathrm{NT}}(x(k|k),u_{\mathrm{nom}}) \end{bmatrix} + \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ g_2 & g_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ g_N & g_{N-1} & \cdots & g_1 \end{bmatrix} \begin{bmatrix} \delta u(k) \\ \delta u(k+1) \\ \vdots \\ \delta u(k+N-1) \end{bmatrix}, \quad (15)$$

where $g_j = C_k A_k^{j-1} B_k$, $j = 1, \ldots, N$ represent Markov parameters. Note that $F_{\mathrm{NT}}(x(k|k), u_{\mathrm{nom}})$ can be computed recursively since

$$F_{\mathrm{NT}}(x(k|k), \{u_{\mathrm{nom}}(k+i)\}_{i=0}^{N-1}) = F_{\mathrm{T}}(F_{(\mathrm{N-1})\mathrm{T}}(x(k|k), \{u_{\mathrm{nom}}(k+i)\}_{i=0}^{N-2}), u_{\mathrm{nom}}(k+N-1)).$$

Note also that the first term of the right-hand side drops out if the output vector consists of linear combinations of the state (i.e., $y(k) = C_k x(k)$). In order to keep the notation simple, we will denote the above prediction equation as

$$Y = Y_{\mathrm{nom}} + G_{\mathrm{u}} \delta U.$$

Note that Y_{nom} can be computed from the state estimate $x(k|k)$ by performing integration of the nonlinear ODE. Matrix G_{u} must be recomputed at each time step based on the updated Jacobian matrices.

Remark 1 The local linearization in (11) makes sense only when the computed inputs $\{u(k+i)\}_{i=0}^{j-1}$ do not deviate much from u_{nom}. For nonlinear models it is the objective to find a control policy $u_{\mathrm{nom}}(k+j|k)$ which is as close as possible to the optimal strategy $u_{\mathrm{opt}}(k+j|k)$. A simple but effective choice is to start with $u_{\mathrm{nom}}(k+j|k) = u(k+j|k-1)$, i.e. the optimal control policy derived at the previous sample. Naturally, the input trajectories computed in the subsequent optimization is likely to be a better approximation of the actual future input sequence (de Keyser, 1998; Santos *et al.*, 1995).

Remark 2 Note that a simple relationship exists between the control actions Δu and δu:

$$\begin{aligned} \Delta u(k) &= u(k) - u(k-1) \\ &= u_{\mathrm{nom}}(k) + \delta u(k) - u(k-1) \\ \Delta u(k+1) &= u(k+1) - u(k) \\ &= u_{\mathrm{nom}}(k+1) + \delta u(k+1) \\ &\quad - u_{\mathrm{nom}}(k) - \delta u(k) \end{aligned}$$

etc.

Then

$$\begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N-1) \end{bmatrix} = E \begin{bmatrix} \delta u(k) \\ \delta u(k+1) \\ \vdots \\ \delta u(k+N-1) \end{bmatrix} + F,$$

with

$$E = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ -I & I & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & -I & I \end{bmatrix}, F = EU_{\text{nom}} - \begin{bmatrix} u(k-1) \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

where $U_{\text{nom}} = [\, u_{\text{nom}}(k) \; u_{\text{nom}}(k+1) \; \ldots \; u_{\text{nom}}(k+N-1) \,]^\top$. The latter relationship will be used to formalize constraints on input increments.

The objective function (3) is a quadratic form in δU:

$$\begin{aligned} &J(\delta U) = \\ &(Y_{\text{nom}} + G_{\text{u}}\delta U - Y_{\text{ref}})^\top \bar{Q}(Y_{\text{nom}} + G_{\text{u}}\delta U - Y_{\text{ref}}) \\ &+(U_{\text{nom}} + \delta U - U_{\text{ref}})^\top \bar{R}(U_{\text{nom}} + U - U_{\text{ref}}), \end{aligned}$$

where $\bar{Q} = \text{diag}(Q, \ldots, Q) \in \mathbb{R}^{Nn_{\text{y}} \times Nn_{\text{y}}}$ and $\bar{R} = \text{diag}(R, \ldots, R) \in \mathbb{R}^{Nn_{\text{y}} \times Nn_{\text{u}}}$. The objective can easily be transformed into the standard quadratic cost index

$$\frac{1}{2}U^\top H U + f^\top U + c, \tag{16}$$

where

$$H = 2(G_{\text{u}}^\top \bar{Q} G_{\text{u}} + \bar{R}) \tag{17}$$

$$f = G_{\text{u}}^\top \bar{Q}(Y_{\text{nom}} - Y_{\text{ref}}) + \bar{R}(U_{\text{nom}} - U_{\text{ref}}) \tag{18}$$

$$\begin{aligned} c = &(Y_{\text{u}} - Y_{\text{ref}})^\top \bar{Q}(Y_{\text{u}} - Y_{\text{ref}}) \\ &+ (U_{\text{nom}} - U_{\text{ref}})^\top \bar{R}(U_{\text{nom}} - U_{\text{ref}}). \end{aligned} \tag{19}$$

Then the optimization problem amounts to a quadratic programing problem with the objective function (16) with (17)–(19) subject to the constraints (4)–(6), which can be transformed onto the following form:

$$\begin{bmatrix} E \\ -E \\ I \\ -I \\ G \\ -G \end{bmatrix} \delta U \leq \begin{bmatrix} b_1 - F \\ -b_2 + F \\ d_1 - U_{\text{nom}} \\ -d_2 + U_{\text{nom}} \\ y_1 - Y_{\text{nom}} \\ -y_2 + Y_{\text{nom}} \end{bmatrix}, \tag{20}$$

where b_1, b_2, d_1 and d_2 are of dimension Nn_{u} and consist of N copies of $\Delta u_{\max}$, $\Delta u_{\min}$, $u_{\max}$ and $u_{\min}$ respectively. In the same way, vectors y_1 and y_2 are of dimension Nn_{y} and consist of N copies of $y_{\max}$, $y_{\min}$.

4. QUADRATIC OPTIMIZATION: ASM AND IPM

The constrained quadratic optimization problem can be solved in various ways. First of all, one can choose to eliminate the states and solve (3). If the states are eliminated, one can either use an active set method or an interior-point method (Nash and Sofer, 1996; Nocedal and Wright, 1999). In this section we shortly discuss the properties of the various algorithms. In particular, active set methods (ASM) try iteratively to find the set of constraints that are active at the optimum. To obtain that goal they solve an equality constrained problem at each time step to determine a new search direction. The set of equality constraints consists of the guess at that iteration of which constraints will be active in the optimum. This means that in each iteration a dense set of equations in Nn_{u} variables has to be solved. Moreover, the total number of iterations will increase with the number of active constraints since typically in each iteration only one or some constraints will become active. This leads to the fact that ASMs (such as MATLAB's "quadprog" routine), though still widely used as the standard methods for solving the QPs in the MPC algorithms, may lead to very expensive combinatorial procedures if applied on large MPC problems with many (active) constraints. ASMs are nevertheless easy to understand which leads to the advantage of being able to provide initial guesses of the active set or the optimal point to the algorithm (hot starting) based on the MPC solution of the previous time step.

Interior-point methods (IPM) try to solve the non-linear set of Karush-Kuhn-Tucker equations iteratively by making linear approximations to this set. An advantage over ASMs is that interior-point methods (such as Mosek) can efficiently make use of sparsity, e.g. in the constraints, and will therefore often be faster for solving MPC related problems, since bound or rate of change constraints on the control inputs give rise to sparse constraint matrices. Also the number of iterations to reach a point close to the optimum is typically independent of the number of constraints and will be smaller as for ASMs. A main disadvantage is that they are less suited to use a priori information and difficulties may arise finding a good starting point.

One problem with these standard methods is that they generally require a lot of memory usage to store large dense matrices based on the Hessians of QPs. For long prediction horizons this can lead to memory problems. Another problem is that solving QPs with these standard methods typically requires a computational time that increases with the *third power* of the number of variables.

5. STRUCTURED IPM

An other way to solve the constrained quadratic optimization problem is to retain the states as optimization variables. The advantage of the standard methods is that the number of variables is smaller, the advantage of the second way is that for MPC related problems the structure in the given QP, originating from the system dynamics, can be exploited to develop methods that have a computational time increasing *linearly* with the horizon length. Sometimes it may be difficult to solve the QP optimization problem with long prediction horizon in the given time using standard optimization techniques, so we propose an other method that will give a faster solution for some specific cases. This algorithm is trying to exploit the relation between the optimization variables as given by the model equations,

in order to reduce the number of operations in the optimization. We again consider the nonlinear MPC optimization problem with the quadratic cost function at time k:

$$\min_{\{u(j)\}_{j=0}^{N-1}} \sum_{j=1}^{N} \big[(y(j) - y_{\mathrm{ref}}(j))^\top Q(y(j) - y_{\mathrm{ref}}(j)) \quad (21)$$
$$(u(j-1) - u_{\mathrm{ref}}(j-1))^\top R(u(j-1) - u_{\mathrm{ref}}(j-1))\big]$$

subject to

$$C_{\mathrm{u(k+j)}}u(k+j) + C_{\mathrm{x(k+j)}}x(k+j) \le C_{\mathrm{c}}(k+j), \quad (22)$$

$$x(k+j+1) = A_{k+j}x(k+j) + B_{k+j}u(k+j),$$
$$y(k+j) = C_{k+j}x(k+j) + D_{k+j}u(k+j)$$

for $j = 0, \ldots, N-1$. Since the QP comes from the dynamic NMPC optimization problem, it is clear that the state space equations give a relation between the optimization variables. Moreover, considering all the constraints and the cost function, it is clear that there are typically only constraints and cost terms connecting variables at time step j with variables at time step $j-1$ or $j+1$. Once the state space equations are used to eliminate the states, the number of variables is highly reduced, but then all variables are interconnected since the state space equations were used to write every state $x(k)$ as a function of all inputs up to k. If this elimination is not carried out, the structure given by the dynamics of the plant (i.e. states and inputs at time step k only interact with states and inputs of the nearest time steps) will reflect in the KKT equations, as can be seen in (Rao *et al.*, 1998). The result is a linear set of equations which is much larger than the one obtained in (16), but having a banded structure. The question now is whether these KKT equations can be solved more efficiently as compared to the smaller, but non-structured set. To solve these new sets of linear equations efficiently at each iteration, the interior-point optimizer exploits the banded structure by following a discrete Riccati recursive scheme. We can expect this recursion to be numerically stable although it was motivated originally by structural rather than numerical considerations. The details of the optimization algorithm can be found in (Rao *et al.*, 1998; Buijs *et al.*, 2002). Note that constraints (20) can easily be rewritten in the form of (22).

6. DESIGN CASE STUDIES

Example 1 The first nonlinear process is based on the forced-circulation evaporator described in (Newell and Lee, 1989), and shown in Figure 1. A feed stream enters the process at concentration X_1 and temperature T_1, with flow rate F_1. It is mixed with a recirculating liquor, which is pumped through the elevator at flow rate F_3. The evaporator itself is a heat exchanger, which is heated by steam flowing at a rate F_{100}, with entry temperature T_{100} and pressure P_{100}. The mixture of feed and recirculating liquor boils inside the heat exchanger, and the resulting mixture of vapor and liquid enters a separator, in which the liquid level is L_2. The operating pressure inside the evaporator is P_2. Most of liquid from the separator becomes the recirculating liquor. A small portion of it is drawn off as product, with concentration X_2, at a flow rate F_2 and temperature T_2. The vapor from the separator flows to a condenser at flow rate F_4 and temperature T_3, where it is condensed by being cooled with water flowing at a rate F_{200}, with entry temperature T_{200} and exit temperature T_{201}. State variables L_2, X_2 and

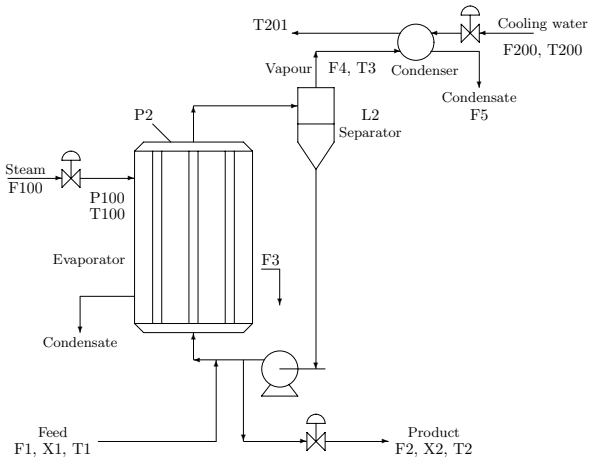


Fig. 1. *Evaporation process.*

P_2 are the controlled outputs for this process. The manipulated variables are chosen to be F_2, P_{100} and F_{200}. There are five disturbance signals, namely F_3, F_1, X_1, T_1 and T_{200}, which are left fixed at their equilibrium values. Input and output constraints are imposed in the optimization. Figure 2 shows the controlled outputs after using MPC described in the previous sections. The level L_2 is kept at the value of 1m. The setpoint for X_2 is ramped down linearly from 25% to 15% over a period of 20 minutes, and the operating pressure P_2 is simultaneously ramped up from 50.5 kPa to 70 kPa. Different QP solvers gave good performance results, where the reference trajectories were followed quite closely. Figures 3 and 4 show the maximal time required to solve one MPC problem using ASM and IPM with different number of variables (Nn_{u}). We see that although Mosek is much faster, both QPs exhibit the behaviour related to the third power of the number of variables. Figure 5 presents computational time required to solve MPC problem ($N = 50$) at each cycle using the structured IPM. It takes the MPC controller approximately 5 seconds to solve each problem. We noticed that for horizons $N > 25$ the structured IPM was faster than MATLAB's ASM, and so it can be implemented online with longer horizons, if required, considering the sampling time $T = 1$ min for this process. The tests also showed later that the time increased linearly with the horizon length, and the algorithm became faster than Mosek for $N > 160$.

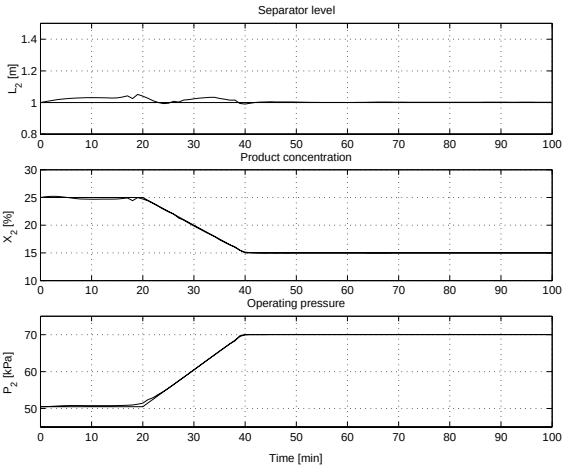


Fig. 2. *MPC simulation of the evaporation process.*

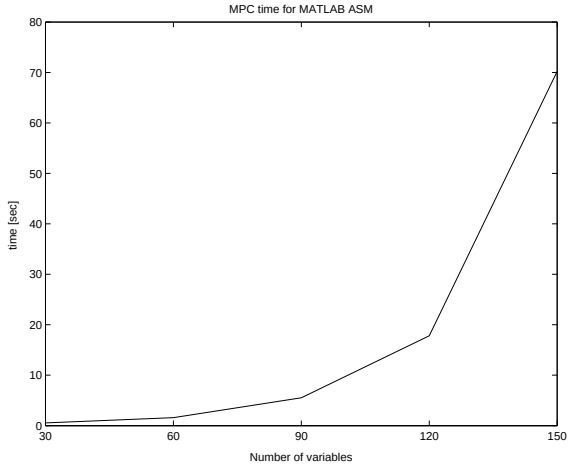


Fig. 3. *Evaporator simulation time: MATLAB's ASM.*

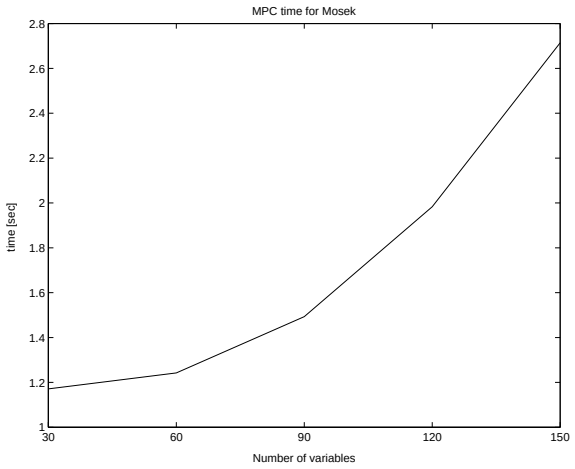


Fig. 4. *Evaporator simulation time: Mosek.*

Thus the evaporation process is an example which can prove the efficiency of the structured IPM for longer prediction horizons in certain cases.

Example 2 The second case study is concerned with MPC controller design for a high purity distillation column. Figure 6 presents the distillation process, its 82-state nonlinear model can be found in (Skogestad

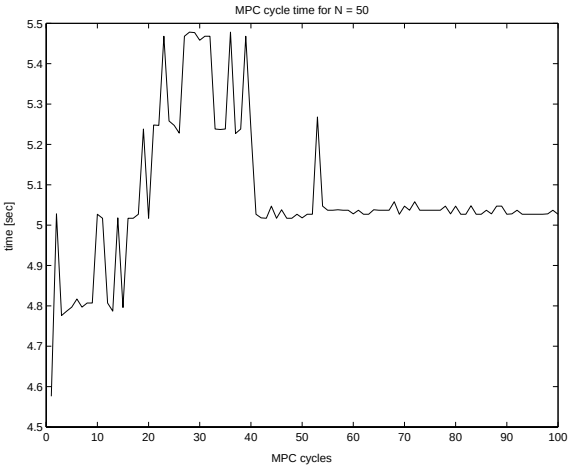


Fig. 5. *MPC cycle time using structured IPM.*

and Postlethwaite, 1996). The column contains 41 trays

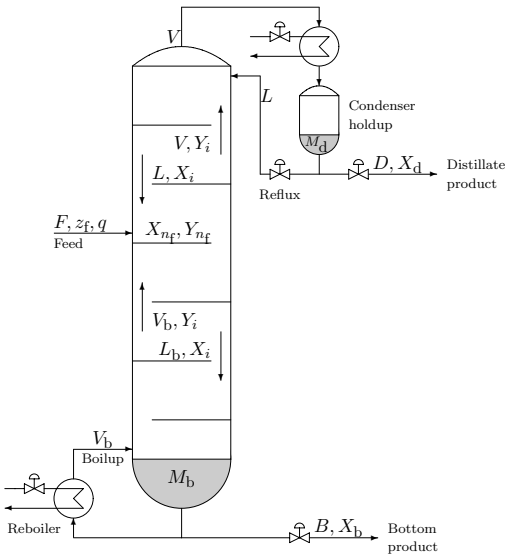


Fig. 6. *Binary high-purity distillation column.*

that are located along its length. The raw material enters the column at a flow rate of F and with composition z_f. The top product, the distillate, is condensed and removed at a flow rate of D and with composition X_d. The bottom product is removed as a liquid at a flow rate of B and with composition X_b. The operation of the column requires that some of the bottom product is reboiled at a rate of V_b to ensure the continuity of the vapor flow. In the same way, some of the distillate is refluxed to the top tray at a rate of L to ensure the continuity of the liquid flow. The vapor boilup V and the reflux flow L are the manipulated variables. Feed flow rate F with composition z_f act as disturbances. The MPC objective is to bring the controlled outputs X_b and X_d to the setpoints of 1% and 99%, respectively (see Figure 7). Figures 8 and 9 show the maximal time required to solve one MPC problem using ASM and IPM with different number of variables (Nn_u), where we see similar behavior as in the previous example. Figure 10 presents computational time required to solve MPC problem at each cycle using the structured

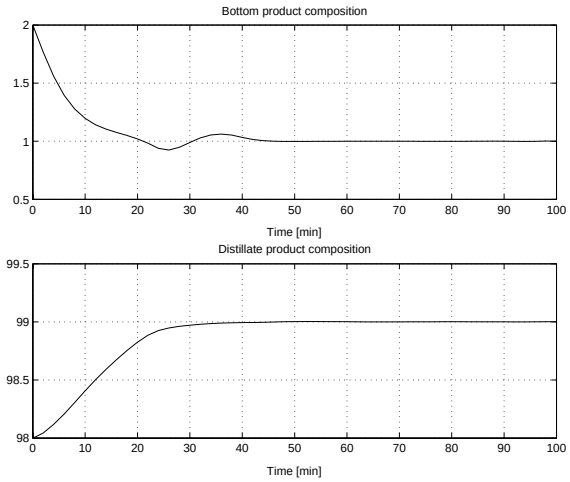


Fig. 7. *MPC simulation of the distillation column.*

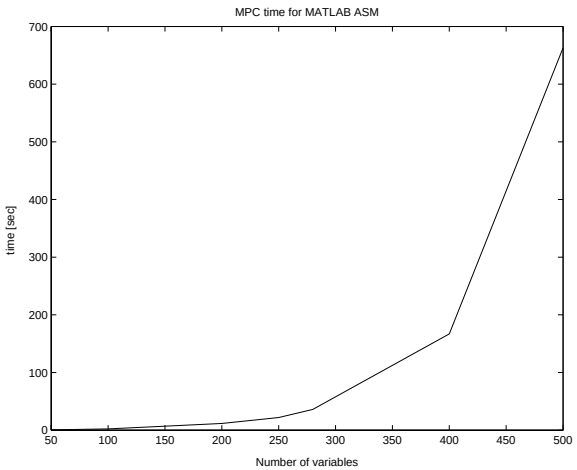


Fig. 8. *Column simulation time: MATLAB's ASM.*

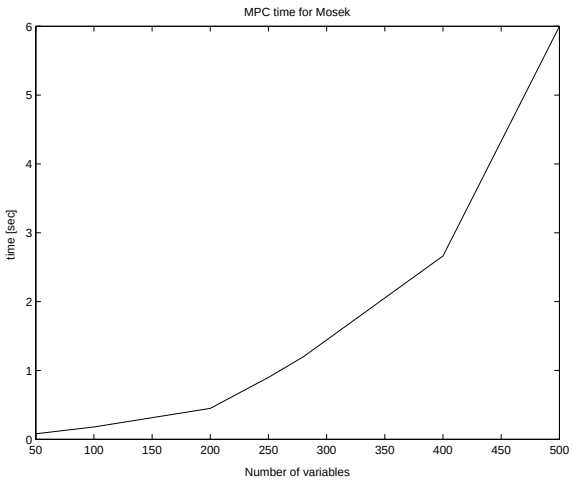


Fig. 9. *Column simulation time: Mosek.*

IPM for the distillation column. We noticed that the structured IPM became faster than MATLAB's ASM only for $N > 140$ and it failed to compete with Mosek even for much longer horizons. This could definitely be attributed to the fact that the algorithm retains states as optimization variables. In this process we have 2 inputs and 82 states which give this big increases in the number of optimization variables. So, the structured IPM is definitely not a good choice for the MPC optimizer of the distillation process.

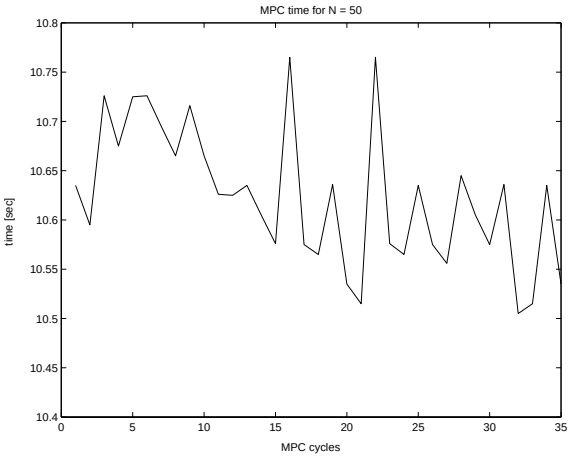


Fig. 10. *MPC cycle time using structured IPM.*

7. CONCLUSIONS

In this paper a structured IPM was proposed to solve the nonlinear MPC problem. The algorithm explicitly takes the structure of the given problem into account such that the MPC controllers will be able to solve a dynamic optimization problem in shorter time. The first example proved that for MPC problems with long horizons, naive implementations of standard QP solvers could be inefficient. So, in this paper we showed that for certain systems, the structured IPM could give a real-time optimal MPC solution within the available time for much longer horizons.

8. REFERENCES

Buijs, J., J. Ludlage, W. van Brempt and B. De Moor (2002). Quadratic programming in model predictive control for large scale systems. In: *Proc. IFAC World Congress*. Barselona, Spain.

de Keyser, R. (1998). A gentle introduction to model based predictive control. In: *EC-PADI2 Int. Conference on Control Engineering and Signal Processing*. Peru.

Lee, J.H. and N.L. Ricker (1994). Extended kalman filter based nonlinear model predictive control. *Ind. Eng. Chem. Res.* **33**, 1530–1541.

Mayne, D.Q., J.B. Rawlings, C.V. Rao and P.O.M. Scokaert (2000). Constrained model predictive control: Stability and optimality. *Automatica* **36**, 789–814.

Muske, K.R. and J.B. Rawlings (1993). Model predictive control with linear models. *AIChE Journal* **39**(2), 262–287.

Nash, S.G. and A. Sofer (1996). *Linear and nonlinear programming*. The McGraw-Hill Companies, Inc.

Newell, R.B. and P.L. Lee (1989). *Applied process control. A case study*. Prentice Hall.

Nocedal, J. and S.J. Wright (1999). *Numerical optimization*. Springer Series in Operations Research.

Qin, S.J. and T.A. Badgwell (1997). An overview of industrial model predictive control technology. In: *AIChE Symposium Series: Fifth Int. Conf. on Chemical Process Control*. Vol. 316. pp. 232–256.

Rao, C.V., S.J. Wright and J.B Rawlings (1998). Application of interior-point methods to model predictive control. *Journal of Optimization Theory and Applications* **99**, 723–757.

Rawlings, J.B. and K.R. Muske (1993). Stability of constrained receding horizon control. *IEEE Transactions on Automatic Control* **38**(10), 1512–1516.

Ricker, N.L. and J.H. Lee (1995). Nonlinear model predictive control of the tennessee eastman challenge process. *Computers and Chemical Engineering* **19**(9), 961–981.

Santos, L.O., N.M.C. de Oliveira and L.T. Biegler (1995). Reliable and efficient optimization strategies for nonlinear model predictive control. In: *Proc. of the IFAC Symposium DYCORD+95*. Helsingor, Denmark. pp. 33–38.

Skogestad, S. and I. Postlethwaite (1996). *Multivariable feedback control*. Wiley.