Available online at www.sciencedirect.com
ScienceDirect
Chemical Engineering and Processing 46 (2007) 1200–1214
Chemical Engineering and Processing Process Intensification
www.elsevier.com/locate/cep

Fast reduced multiple shooting methods for nonlinear model predictive control

Andreas Schäfer, Peter Kühl, Moritz Diehl *, Johannes Schlöder, Hans Georg Bock

Interdisciplinary Center for Scientific Computing (IWR), University of Heidelberg, Germany

Accepted 29 June 2006
Available online 3 March 2007

Abstract

Nonlinear model predictive control (NMPC) has become an appealing control concept for chemical processes because it can directly take into account the multivariable character, nonlinearities and constraints on inputs and states. While linear MPC is frequently applied in the process industries, practical applications of NMPC are rare. One reason is the relatively large computation time still needed to solve the nonlinear constrained optimization problems inherent to NMPC schemes. For faster controller response, the use of an extended partially reduced SQP method within the direct multiple shooting framework is proposed. This method is implemented in the code MSOPT which allows for an accelerated calculation of the directional derivatives and thereby saves computation time. Furthermore, this method is adapted to the real-time iteration scheme, which leads to a tremendous reduction of the time needed to provide new controls. The new NMPC variant is compared with a previously introduced scheme based on the optimization code MUSCOD-II, where the extended reduction is not in use. Numerical results are presented for both NMPC schemes and a decentralized PI controller in closed loop with a simulation model of a highly nonlinear thermally coupled distillation column which separates a ternary mixture.
© 2007 Published by Elsevier B.V.

Keywords: Nonlinear model predictive control; Real-time optimization; Direct multiple shooting; Coupled distillation column; Ternary mixture

1. Introduction

Linear model predictive control has been in use in the (petrochemical) process industry since the early 1980s. Although nonlinear model predictive control (NMPC) has been an active field of research over almost the last two decades, there are still not many industrial applications of NMPC in comparison to linear MPC and other advanced control schemes (see also [2,41]). There seem to be two main reasons for the difficulty of translating theoretical results into practical realizations of NMPC: first, NMPC require a rigorous physical model of the process to be controlled, based on first principles such as mass and energy balances. These are typically formulated as differential–algebraic equations (DAEs) or partial differential equations (PDEs). Exact modeling, however, is time-consuming and expensive. The advent of sophisticated modeling environments and tools is currently bringing relief to this bottleneck, see, e.g., [11,21]. Another possibility is to use data-driven input–output models based on artificial neural networks, impulse models, etc. This approach will not be considered here.

Second, NMPC requires the successive online solution of constrained nonlinear optimization problems. For realistic processes, these optimization problems easily become large while the real-time requirements remain strict: the answer to disturbances or setpoint changes of a process in terms of updated controls should be available as fast as possible. Most optimization methods used in NMPC, however, iterate until a convergence criterion is fulfilled, so that a certain response time cannot be maintained (for an introductory overview on optimization methods for NMPC see [4,5]).

This paper shows two ways to overcome this second obstacle on the way towards real-time NMPC schemes. First, the use of a modified direct multiple shooting optimization method [42] is proposed. Compared to the standard direct multiple shooting method as formulated for NMPC in Ref. [6], it reduces the number of directions for which directional derivatives are evaluated,

* Corresponding author. Tel.: +49 6221 54 88 31.
E-mail address: m.diehl@iwr.uni-heidelberg.de (M. Diehl).

0255-2701/$ – see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.cep.2006.06.024

so that the number is now independent of the state dimension. This results in a reduction of computation time for optimization problems with low degrees of freedom. Second, the real time iteration scheme as suggested in Refs. [16,17] is investigated along the lines of the two direct multiple shooting methods. The real-time iteration scheme can drastically reduce computation times and, thus, feedback delays.

All direct multiple shooting NMPC variants discussed within this paper are used to control a thermally coupled distillation column separating a ternary mixture. This process is modeled by DAEs with 106 differential and 159 algebraic variables and exhibits complicated dynamics. The proposed NMPC schemes manage to control the process and reject disturbances, while the feedback response times are in the range of about one per cent of the sampling time down to milliseconds even.

The paper is organized as follows: in the beginning, the concept of NMPC is reviewed and the open-loop optimization problem appearing in NMPC schemes is described in Section 2. Section 3 reviews the direct multiple shooting method and states the resulting nonlinear programming problem (NLP). In Sections 4–6 the class of partially reduced sequential quadratic programming (PRSQP) methods is introduced with its basic algorithm, its Hessian approximations and its globalisation via line-search, respectively. These PRSQP methods are used to solve the NLPs from Section 3 and forms the basis of the NMPC schemes addressed here. Based on this, two PRSQP implementations are presented. The first one, MUSCOD-II, exploits the consistency conditions of the algebraic variables, the second one, MSOPT, additionally takes advantage of fixed initial conditions, as they appear in the NMPC problems of Section 2, to reduce the number of directional derivatives. The efficient calculation of directional derivatives is treated in Section 8, focusing on tailored internal numerical differentiation (IND) methods and automatic differentiation (AD) of the model functions. The real-time iteration scheme is described in Section 9 and adapted to the two optimization methods implemented in MSOPT and MUSCOD-II. Finally, all NMPC variants are applied to a simulated thermally coupled distillation column in order to assess their control performance and the necessary computation times. Results of the closed-loop systems for two disturbance cases are presented in Section 10 and compared to conventional decentralized proportional integral (PI) control. The paper ends with conclusions in Section 11.

2. Nonlinear model predictive control

Given the current state $\tilde{x}(t_0)$ of a system at time t_0, NMPC computes a feedback control by successively solving open-loop optimal control problems on a finite time horizon. The dynamic optimization and the necessary predictions into the future are based on a mathematical process model (see [1] for a comprehensive introduction to NMPC). The control objectives and performance demands on the closed-loop can be formulated with the help of the cost function of the open-loop control problem (typically a Bolza-type formulation, see below) and/or additional constraints on inputs and states. For example, the cost function can penalize deviations from a given setpoint to achieve setpoint tracking; state constraints are formulated for critical process variables (e.g., temperatures, pressure) which must not exceed a certain value; a penalty on the change of controls u can be used to surpress excessive control moves, and so on.

For this paper it is assumed that the process can be described by semi-explicit differential–algebraic equations (DAE) of index 1. We consider finite horizon open-loop control problems of the form:

$$\min_{x,z,u} \int_{t_0}^{t_0+T} L(x(t), z(t), u(t))\mathrm{d}t + E(x(t_0+T), z(t_0+T)) \tag{1a}$$

subject to

the DAE system, $t \in [t_0, t_0+T]$:

$$B(x(t), z(t), u(t)\dot{x}(t) = f(x(t), z(t), u(t)),$$
$$0 = g(x(t), z(t), u(t)) \tag{1b}$$

the initial value constraint:

$$x(t_0) = \tilde{x}(t_0) \tag{1c}$$

as well as control and path constraints:

$$h(x(t), z(t), u(t)) \geq 0, \quad t \in [t_0, t_0+T] \tag{1d}$$

and possible terminal constraints:

$$r^i(x(t_0+T), z(t_0+T)) \geq 0,$$
$$r^e(x(t_0+T), z(t_0+T)) = 0 \tag{1e}$$

In this formulation, $x(t)$ and $z(t)$ are the differential and algebraic states, respectively, $\tilde{x}(t)$ denotes the current differential states that are either measured or estimated, and $u(t)$ is the set of controls. $L(\cdot)$ represents a Lagrange-term, $E(\cdot)$ is known as the Mayer-term, the sum of which defines the Bolza performance index. The Mayer-term $E(\cdot)$ and terminal constraints $r^i(\cdot)$ play an important role in the theory of stabilizing NMPC schemes (see [36,?] for an overview). In this paper, we consider a quadratic cost function that is known to be suited for tracking problems and disturbance rejection, i.e.:

$$L(x(t), z(t), u(t)) = \bar{y}(t)^T Q \bar{y}(t) + \bar{u}(t)^T R \bar{u}(t) \tag{2}$$

where $y(t)$ denotes those differential and/or algebraic states of the process that are to be controlled and $\bar{y}(t)$ are their deviations from the desired reference values y_{ref}. Similarly, $\bar{u}(t)$ are the deviations from the desired control reference values u_{ref}. Q and R are diagonal weighting matrices. In a more general setting, one can also consider linear combinations of the controlled states y and/or controls u. The vectors $x(t)$, $z(t)$, $y(t)$ and $u(t)$ are of dimensions n^x, n^z, n^y and n^u.

Solving the open-loop control problem (1) with a current initial value $\tilde{x}(t_0)$ yields an optimal control trajectory $u^*(t_0, \tilde{x}(t_0))$ on the horizon $[t_0, t_0+T]$. For a perfect model and no disturbances it would suffice to apply this control trajectory to the process. In the closed-loop NMPC framework, however, only a first part of u^* with the length usually depending on the sampling

time is applied to the process. Then the optimal control problem (1) is solved again with a new initial value coming from the process. The time between the advent of a new $\tilde{x}$ and the response of the NMPC in form of a corresponding u^* creates a feedback delay. It is clear that the controller can respond to disturbances and setpoint changes faster if this delay becomes smaller.

The control horizon and the prediction horizon both have the length T in the setup above. In general this does not have to be the case—one can as well have a prediction horizon which is longer than the control horizon. In this case, the controls u are fixed to the last value of the control horion for any time beyond the control horizon.

NMPC heavily relies on the availability of a (at least approximate) solution to the dynamic optimization problem. In case the optimization problem is infeasible for a given initial value $\tilde{x}(t_0)$, a suitable fall-back strategy is needed to retain practical applicability of the control scheme. This important issue is addressed in Ref. [37] and references therein, where a fall-back strategy is described along with a method to determine the set of feasible initial values a-priori. Throughout this article we assume feasibility of the optimization problem given in (1).

3. Direct multiple shooting methods

The vital part of any NMPC is the solution of an open-loop optimal control problem in a sufficiently short time. For the solution of the optimal control problems three direct methods have been proposed: single shooting, collocation and multiple shooting. The nonlinear programming (NLP) problems resulting from the respective discretization and parameterization of controls and/or states are then solved either with SQP methods, interior point methods or a combination thereof, see e.g., [4,44,48]). Advantages of the direct multiple shooting method for the solution of the optimal control problems have been highlighted in Ref. [33].

Direct multiple shooting methods belong to the group of simultaneous methods for the numerical solution of dynamic optimization problems. They are characterized by the introduction of additional intermediate variables of the state trajectory to the original optimization problem. This approach allows to explicitly incorporate knowledge about the solution into the problem by an adequate choice of initial guesses for these additional variables.

In the following, we present a multiple shooting variant introduced in Ref. [9] for DAE systems of index 1 and extended for DAE of higher index in Ref. [46]. First we introduce a multiple shooting grid with $m+1$ grid points:

$$t_0 = \tau_0 < \tau_1 < \cdots < \tau_m = t_0 + T \tag{3}$$

Then the control space is parameterized with a finite number of control parameters q on the multiple shooting grid (3) to transfer the infinite-dimensional optimal control problem (1) to a finite-dimensional nonlinear programming (NLP) problem. Therefore a piecewise representation of the control function $u(t)$ by basis functions φ_j is chosen with *local* support (e.g., piecewise constant, linear) and *local* control parameters q_j:

$$u(\tau) \equiv \varphi_j(\tau, q_j), \quad \tau \in I_j = [\tau_j, \tau_{j+1}], \quad j = 0, 1, \ldots, m-1 \tag{4}$$

Together with the control parameterization we now introduce a state parameterization on the same multiple shooting grid. Therefore start values s_j^x, s_j^z for the differential and algebraic states, respectively, are used to define relaxed initial value problems on each multiple shooting interval $I_j := [\tau_j, \tau_{j+1}]$:

$$B(\cdot)\frac{\mathrm{d}x(\tau)}{\mathrm{d}\tau} = f(x(\tau), z(\tau), \varphi_j(\tau, q_j)),$$
$$0 = g(x(\tau), z(\tau), \varphi_j(\tau, q_j)) - \alpha_j(\tau) g(s_j^x, s_j^z, \varphi_j(\tau_j, q_j)) \tag{5}$$

with the initial values $x(\tau_j) = s_j^x$ and $z(\tau_j) = s_j^z$. Following the idea of a homotopy strategy, the algebraic conditions are modified to allow for inconsistent algebraic start values (as in the collocation approach). The damping factor α_j is defined as a scalar function, which is non-increasing and non-negative on I_j and fulfills $\alpha_j(\tau_j) = 1$. For realizations of the homotopy parameter $\alpha_j(\tau)$ see, e.g., [9,32]. In order to get a solution of the original problem (1b) we have to fulfill the continuity conditions on each grid point, $x(\tau_{j+1}; s_j^x, s_j^z, q_j) - s_{j+1}^x = 0, \quad j = 0, 1, \ldots, m-1$, and the consistency conditions $g(s_j^x, s_j^z, \varphi_j(\tau_j, q_j)) = 0, \quad j = 0, 1, \ldots, m$.

This particular control and state parameterization is responsible for the favorable structural characteristics (e.g., separability of the Lagrange function) of the NLP. These characteristics can be exploited by, e.g., high-rank updates and condensing routines as described in Refs. [10,39].

The Bolza objective function is transformed to a generalized Mayer objective by augmenting the differential equations with the Lagrange term and an zero initial condition.

Summarizing the discretization and parameterization steps, we get the following structured NLP problem formulation of (1):

$$\min_{s_j^x, s_j^z, q_j} \sum_{j=0}^{m-1} L_j(s_j^x, s_j^z, q_j) + E(s_m^x, s_m^z) \tag{6a}$$

subject to the continuity and consistency conditions:

$$x(\tau_{j+1}; s_j^x, s_j^z, q_j) - s_{j+1}^x = 0, \quad j = 0, \ldots, m-1 \tag{6b}$$

$$g(s_j^x, s_j^z, \varphi_j(\tau_j, q_j)) = 0, \quad j = 0, \ldots, m \tag{6c}$$

the initial conditions:

$$s_0^x = \tilde{x}(t_0) \tag{6d}$$

the discretized control and path constraints:

$$h(s_j^x, s_j^z, \varphi_j(\tau_j, q_j)) \geq 0, \quad j = 0, 1, \ldots, m-1 \tag{6e}$$

and the boundary conditions:

$$r^i(s_m^x, s_m^z) \geq 0, \qquad r^e(s_m^x, s_m^z) = 0 \tag{6f}$$

Note that the constraints are checked only at the multiple shooting points and hence a solution of (6) might be infeasible

for the original problem (1). However, by choosing a suitable parametrization of the controls, one can enforce a potential violation of constraints which is only of second or higher order in $\Delta\tau$, the local mesh width. In almost all cases this is sufficient for practical applications.

4. Partially reduced SQP methods

The NLP problem (6) resulting from discretization and parameterization of the original optimal control problem (1) has a special structure which becomes visible in the necessary conditions of optimality for the NLP—the so-called Karush–Kuhn–Tucker (KKT) conditions. Sequential quadratic programming (SQP) methods can be interpreted as Newton-type methods applied to the necessary KKT conditions for the NLP. Reduced sequential quadratic programming (RSQP) methods [25,29] maintain the advantage of iterating over the whole set of variables, but they reduce the costs for computation of the Newton-increment (to the order of the degrees of freedom) by solving only a reduced QP problem in each step. Partially reduced sequential quadratic programming (PRSQP) methods [32,42,44,45] allow to treat NLP problems with inequality constraints while maintaining the advantages of RSQP methods. In order to present the concepts of partially reduced SQP methods we formalize the NLP problem (6) as follows:

$$\min_{w} F(w) \tag{7a}$$

[NLP] s.t.:

$$G_1(w) = 0 \tag{7b}$$

$$G_2(w) = 0 \tag{7c}$$

$$H(w) \geq 0 \tag{7d}$$

The vector w summarizes all optimization variables. The reason for the distinction of the equality constraints in $G \equiv (G_1, G_2)$ is explained later. In order to motivate PRSQP methods we start with the full-space SQP method with iterates:

$$w_{k+1} = w_k + \beta_k \, \Delta w_k \tag{8}$$

The increment Δw_k is defined as the solution of a quadratic program:

$$\min_{\Delta w_k \in \Omega_k} \nabla F(w_k) \, \Delta w_k + \tfrac{1}{2} \, \Delta w_k^T B_k \, \Delta w_k \tag{9a}$$

[QP] s.t.:

$$G_1(w_k) + \nabla G_1(w_k)^T \, \Delta w_k = 0 \tag{9b}$$

$$G_2(w_k) + \nabla G_2(w_k)^T \, \Delta w_k = 0 \tag{9c}$$

$$H(w_k) + \nabla H(w_k)^T \, \Delta w_k \geq 0 \tag{9d}$$

where B_k is an approximation to the Hessian of the Lagrangian of the NLP problem (7):

$$L(w, \lambda, \mu) := F(w) - \lambda_1^T G_1(w) - \lambda_2^T G_2(w) - \mu^T H(w)$$

where the adjoint variables $\lambda \equiv (\lambda_1, \lambda_2)$ and μ are the Lagrangian multipliers of the constraints G_1, G_2 and H. The step size factor $\beta_k \in (0, 1]$ is used, e.g., in a line-search method described later. Bounds on the variables can be included in Ω_k and used for globalization by trust-region methods.

In order to set up partially reduced methods for (7) we define a kernel-basis of the linearized equality constraints G_1. The variables which are defined by the linearization of the equality constraint G_1 are denoted by w_1 so that the Jacobian of $\nabla_{w_1} G_1$ is regular over the whole variable domain. This partitions the variables into $w \equiv (w_1^T, w_2^T)^T$. In the following we define a coordinate basis which allows exploitation of sparsity structures in $\nabla_{w_1} G_1$. As a consequence, we have a partitioning of the step:

$$\Delta w = (\Delta w_{1,k}^T, \Delta w_{2,k}^T)^T := S_{1,k}^R y_k^R + S_{1,k}^N y_k^N$$

with the range-space basis $S_{1,k}^{RT} = [\nabla_{w_1} G_{1,k}^{-1} : 0]$ and a null-space basis $S_{1,k}^{NT} = [-\nabla_{w_2} G_{1,k} \nabla_{w_1} G_{1,k}^{-1} : I]$, the range-space component $y_k^R := -G_{1,k}$ and the null-space component $y_k^N \equiv \Delta w_{2,k}$ as the solution of the reduced quadratic program [RQP]:

$$\min_{y_k^N \in \Omega_k^N} \nabla F_k^T S_{1,k}^N y_k^N + C_k y_k^N + \frac{1}{2} y_k^{NT} S_{1,k}^{NT} B_k S_{1,k}^N y_k^N \tag{10a}$$

s.t.

$$G_{2,k} + \nabla G_{2,k}^T S_{1,k}^R y_k^R + \nabla G_{2,k}^T S_{1,k}^N y_k^N = 0 \tag{10b}$$

$$H_k + \nabla H_k^T S_{1,k}^R y_k^R + \nabla H_k^T S_{1,k}^N y_k^N \geq 0 \tag{10c}$$

where the second term $C_k y_k^N := y_k^{RT} S_{1,k}^{RT} B_{1,k} S_{1,k}^N y_k^N$ in the quadratic objective function is a cross term involving both the range-space and null-space basis. Up to now, we have simply reformulated the problem of computing an SQP step Δw_k. Formally, the partitioning of the step (8) can be seen as the elimination of Δw_1 in problem (9).

RSQP methods use all equality constraints and active inequality constraints for the elimination of the corresponding variables. Problems occur when the active set changes which makes the computation of the reduced Hessian via updates inefficient. In contrast PRSQP methods use only some of the equality constraints to eliminate the corresponding optimization variables which makes handling of inequalities more efficient. Reduced and partially reduced SQP methods typically set the cross term to zero and directly computes the reduced Hessian $B_{1,k}^N = S_{1,k}^{NT} B_{1,k} S_{1,k}^N$, which is now the only term where second order information in the algorithm is considered. The reduced Jacobians of the RQP are calculated by first computing the null-space basis $S_{1,k}^N$ and then the directional derivatives of the problem functions F, G_2, H with the columns of the null-space basis as directions. This PRSQP method with explicit calculation of the null-space basis is called direct PRSQP method.

5. Computation of reduced Hessians in PRSQP methods

The strength of PRSQP methods arises from the fact that the reduced Hessians of the Lagrangian can be calculated by approximations $B_{1,k}^N$. This avoids expensive applications of $S_{1,k}^N$ or $S_{1,k}^{NT}$. Here, the reduced Hessian is approximated by the BFGS

update formula, see e.g., [24]:

$$B^N_{1,k+1} := B^N_{1,k} + U^{\mathrm{BFGS}}(B^N_{1,k}, \Delta y^N_k, \Delta\gamma^N_k),$$

$$\text{where } U^{\mathrm{BFGS}}(B, \Delta y, \Delta\gamma) := \frac{\Delta\gamma\Delta\gamma^T}{\Delta y^T\Delta\gamma} - \frac{(B\,\Delta y)(B\,\Delta y)^T}{\Delta y^T B\,\Delta y}$$

The key property of this formula is the secant condition $B\,\Delta y = \Delta\gamma$. The intention for this kind of update is to collect second order information from first order information available in each iteration. Therefore Δy is formed by the null-space step, $\Delta y := y^N_k$, and $\Delta\gamma$ by the resulting difference of reduced gradients of the Lagrangian:

$$\Delta\gamma := \bar{S}^{NT}_{1,k}\nabla_w L(\bar{w}_k, \lambda_{k+1}, \mu_{k+1}) - S^{NT}_{1,k}\nabla_w L(w_k, \lambda_{k+1}, \mu_{k+1})$$

where a bar over a symbol means evaluation at an intermediate point. Note that the reduced gradient does not depend on the Lagrange multipliers $\lambda_{1,k+1}$. The intermediate point may be chosen as w_{k+1}, which defines an update strategy in the spirit of Ref. [38] and is cheaply to calculate, or:

$$\bar{w}_k = w_k + S^N_{1,k} y^N_k$$

which defines an asymptotically correct update strategy. A proof for the resulting local two-step super-linear convergence of the algorithm can be found in Ref. [44]. In order to maintain positive definiteness in the constrained case, a Han–Powell modification is used [40].

An option is a limited memory strategy to avoid blowup of the condition number.

As we have seen, the iterates of SQP and PRSQP methods are the same if we additionally compute the cross-term. In the following we demonstrate that this term is cheap to calculate in case of a least squares objective function with a Gauß-Newton Hessian approximation. Let us define the objective function in the NLP (7) as:

$$F(w) \equiv \frac{1}{2}||F^{ls}(w)||^2_2$$

A Gauß-Newton Hessian approximation is defined by $B_k \equiv \nabla F^{ls}(w_k)\nabla F^{ls}(w_k)^T$. This type of Hessian approximation ensures convergence to statistically stable optima and is accurate in the unconstrained case if the residuals of F^{ls} are small, see [8]. Note that we can easily compute the cross term in RQP if we explicitly calculate the reduced Jacobians $\nabla F^{ls}(w_k)^T(S^N_{1,k}, S^R_{1,k}y^R_k)$ via directional derivatives, since the cross term C_k is the following product:

$$C_k := y^R_k S^{RT}_{1,k}\nabla F^{ls}(w_k)\cdot\nabla F^{ls}(w_k)^T S^N_{1,k}$$

Using this cross term gives us a one-step linear convergence rate of the resulting Gauß-Newton PRSQP method (same as for Gauß-Newton SQP method) for constrained least squares problems.

6. Globalization of PRSQP methods

The globalization of the PRSQP variants can be achieved by line-search or trust-region methods where, e.g., an (exact) l_1-penalty function ϕ is used to measure (sufficient) decrease:

$$\phi_{\rho,\sigma,\tau}(w) = F(w) + \sum_{i\in E_1}\rho_i|G_{1,i}(w)| + \sum_{i\in E_2}\sigma_i|G_{2,i}(w)| + \sum_{j\in I}\tau_j H_j(w)^-$$

with $H- := \max(-H, 0)$ and E_1, E_2, I being the set of constraints indices in G_1, G_2, H, respectively. In case of direct PRSQP methods, where the Lagrangian multipliers are not at all calculated, the compatibility between step and sufficient decrease of the penalty function can be achieved by a partially multiplier free globalization method. Therefore we calculate lower bounds for the penalty parameters by:

$$\rho_i||G_1||_1 \geq |\lambda_1^T G_1|, \qquad \sigma_i \geq |\lambda_{2,i}|, \qquad \tau_j \geq \mu_j$$

Note that $\lambda_1^T G_1$ can be computed cheaply by the following formula (motivated by the stationarity condition of the Lagrangian):

$$\lambda_1^T G_1 = -\nabla F^T S^R_1 y^R + \lambda_2^T\nabla G_2^T S^R_1 y^R + \mu^T\nabla H^T S^R_1 y^R$$

For ensuring convergence to a KKT point (w^*, λ^*, μ^*) the conditions for the penalty parameters have to be fulfilled also for the optimal multipliers (λ^*, μ^*). One-way to achieve this is to have monotone increasing penalty parameters in a line-search step of the SQP method.

7. PRSQP implementations

For a comparison of the direct multiple shooting methods described in this article remember the parameterized and discretized NLP problem (6). The free variables of the NLP are denoted by $w := (s^x_0, s^z_0, q_0, \ldots, s^x_{m-1}, s^z_{m-1}, q_{m-1}, s^x_m, s^z_m)$. The equality constraints $G = (G_1^T, G_2^T)^T$ include the multiple shooting conditions (consistency and continuity constraints) as well as initial conditions and equality multi-point boundary constraints. The two PRSQP variants differ in the way G is divided into G_1 and G_2. The inequality constraints are summarized in H and include the control and path constraints and the inequality multipoint boundary constraints. The boundary constraints, control and path constraints are abbreviated as BCP conditions. In Fig. 1, different PRSQP levels ranging from full-space SQP method to fully reduced SQP methods are shown. In this article, we present two direct multiple shooting implementations with different PRSQP levels: MUSCOD-II [31,32] and MSOPT [42]. They are applied to solve the process engineering NMPC application described later.

The first PRSQP variant, MUSCOD-II, is characterized by defining G_1 as the consistency conditions, G_2 as the initial, continuity, and discretized BCP equality conditions and H as the discretized BCP inequality conditions. This has several consequences: as in the full-space variant, the Hessian of the reduced KKT system is block diagonal. This block structure can be exploited by high rank updates for a Quasi-Newton update Hessian and leads to a faster convergence behavior of the NLP solver compared to updating the complete Hessian instead of the sin-

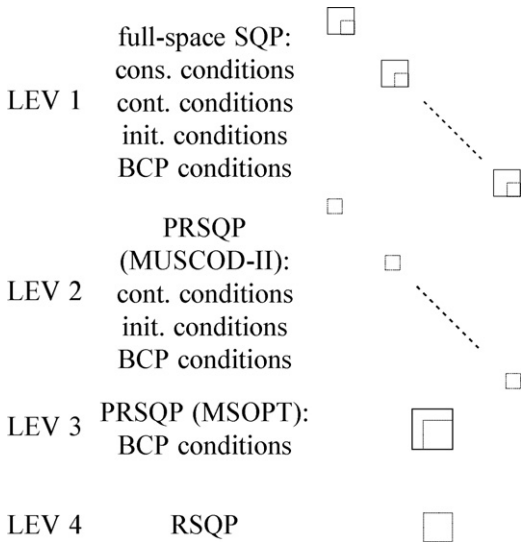


Fig. 1. Family of PRSQP methods for optimal control problems.

gle blocks. Also, the integrations on different multiple shooting intervals are completely decoupled and can therefore be performed in parallel. This algorithm was developed by Leineweber [32].

The second PRSQP variant, MSOPT, is characterized by additionally using the continuity and initial conditions on the differential states in G_1. The continuity conditions in G_1 lead to the fact that the Hessian is not block diagonal, since they define a linear coupling of the Jacobians of neighboring multiple shooting nodes. In case of, e.g., fixed initial values for the differential states as in NMPC the number of directional derivatives in the reduced QP (10) is now independent from the state dimension. In combination with tailored derivative generation, explained in Section 8, this leads to a speed up of computation time compared to MUSCOD-II, if large-scale problems are treated with low degrees of freedom. In the context of parameter estimation this method was developed and implemented by Schlöder [43]. The optimal control variant was later developed by Schäfer [42].

Both algorithms intertwine the projection and gradient evaluation steps in the reduction process. It can be shown that it suffices to compute a reduced number of *directional* derivatives in the null-space basis $S^N_{1,k}$ and the reduced Jacobians $\nabla(F, G_2, H)^T S^N_{1,k}$ for one (reduced) SQP step:

$$\underbrace{m(n_x + n_z + n_u)}_{\text{full-space}} \rightarrow \underbrace{m(n_x + n_u + 1)}_{\text{MUSCOD-II}} \rightarrow \underbrace{\tfrac{1}{2}m((m+1)n_u + 1)}_{\text{MSOPT}}$$

As already mentioned in Section 5, for least-squares problems MUSCOD-II and MSOPT calculate the same SQP iterates if Gauß-Newton Hessian and the corresponding cross term are used.

8. Efficient computation of directional derivatives

In order to solve the reduced quadratic programs efficiently, special care must be taken of the structure of the NLP problem for the calculation of derivatives. For the multiple shooting parameterization we have to compute the directional derivatives of differential states, also called directional sensitivities, see [32,42]. This is done efficiently by the methods realized in the BDF integrator DAESOL [3]. The derivatives are the solution of the directional variational differential–algebraic equations (VDAE).

In case of MUSCOD-II the linearized BDF equations of the directional VDAE are *directly* solved at the solution of the nominal trajectory. Therefore the iteration matrix of the BDF equations has to be computed in every BDF step. The costs in every BDF step are the costs for evaluating the model derivatives w.r.t. the full set of state variables and control directions and additionally the solution of a linear system. For a high number of directions this approach is advantageous and shows good performance.

In case of MSOPT the nonlinear BDF equations of the directional VDAE are solved *simultaneously* together with the BDF equations of the nominal trajectory. Here the same iteration matrix is used. The costs in every BDF step are the costs for evaluating the *directional* model derivatives times the number of Newton iterations (up to three) for the solution of the BDF equations. This is efficient, if the number of directions are *independent* of the state dimension as is the case in MSOPT.

Both approaches conform to the principle of internal numerical differentiation (IND), see [7], where the main idea is to differentiate the integrator scheme that is used to calculate the nominal trajectory. As a result, the directional derivatives are computed efficiently and with high accuracy. For integration and sensitivity generation (directional) model derivatives need to be evaluated. It is necessary to do this very fast with the required accuracy. Here this is achieved by applying the algorithmic differentiation tool Adol-C [26]. Since most model derivatives are sparse, a seed matrix compression [14] is used to further speed up computation time. Note that the sparsity structure of the model derivatives has to be computed only once and can then be reused.

9. Real-time iteration scheme for NMPC

The conventional approach in NMPC is to solve the NMPC problem (1) to desired accuracy at every time t_0 with initial value $x_0 \equiv \tilde{x}(t_0)$ and the optimal control solution of (1) is instantaneously given to the real process at time t_0. Unfortunately this is impossible with finitely fast computers. In the following we define an algorithm scheme for nonlinear model predictive control that delivers feedback very quickly. It is based on the direct multiple shooting algorithm for solution of (1) explained in Section 7. Here, the initial value x_0 is embedded as a homotopy parameter with the trivial equality constraint $S^x_0 - \tilde{x}(t_0) = 0$. Doing so, a feedback control is available after only one PRSQP iteration. This approach is called the real-time iteration and is described in its latest version in Ref. [18]. The real-time iteration scheme allows to divide the necessary computations during each real-time iteration into two parts: a long *preparation phase* where the entire QP problem is set up and decomposed except for entries involving $\tilde{x}(t_0)$ and a short *feedback phase* where the remaining entries are calculated the moment $\tilde{x}(t_0)$ becomes available to deliver the feedback control. Note that the calcu-

lations in the preparation phase can be done *before* the system state $\tilde{x}(t_0)$ is known. This minimizes the feedback delay typical to online optimization-based control methods.

A further advantage of the real-time iteration scheme with initial value embedding is that it gives a second order prediction for the exact solution, thereby taking system nonlinearities into account. Note, that the real-time iteration scheme on shrinking horizons is contracting under the *same* sufficient conditions as the corresponding off-line scheme with fully converged solutions, see [18]. This proof can be generalized to a shift strategy on moving horizons and leads to a convergent closed-loop behavior in this case. Even more appealing from an NMPC point of view, nominal stability results of the NMPC with real-time iterations have been derived in Ref. [19]. It is one of rather few stability results where *both* system and optimizer dynamics are considered as one single, intertwined system.

In what follows, the aforementioned real-time iteration scheme is applied to the two direct multiple shooting algorithms MUSCOD-II and MSOPT. This leads to two real-time iteration NMPC schemes that differ in their underlying PRSQP methods and hence in the necessary calculations in the feedback and preparation phase. Note that in case of integral least squares terms in the objective of (1) it we apply a Gauß-Newton approximation of the Lagrange-Hessian. The major difference between the methods is that the initial value constraints $s_0^x - \tilde{x}(t_0) = 0$ is not part of G_1 in MUSCOD-II while it is in MSOPT.

The real-time iteration of MUSCOD-II consists of the following five steps:

(1) Calculate coordinate-basis S^N and inhomogeneity $S^R y^R$ of the consistency constraints G_1.
(2) Integrate and evaluate reduced constraint residuals of continuity, boundary and path constraints $G_2 + \nabla G_2^T S^R y^R$, $H + \nabla H^T S^R y^R$. Calculate reduced Jacobians of objective $\nabla F^T S^N$ and constraints $\nabla(G_2, H)^T S^N$ and reduced Hessian approximation.
(3) Use linearized reduced continuity conditions of G_2 to eliminate differential state variables except first node (condensing).
(4) At the moment $\tilde{x}(t_0)$ becomes available, solve the reduced QP in initial state Δs_0^x and controls $\Delta q_0, \ldots, \Delta q_{m-1}$. The value $q_0 + \Delta q_0$ can immediately be given as a control to the real system.
(5) Expand the fully reduced QP solution to yield the full QP solution $\Delta w = S^R y^R + S^N y^N$. Based on this QP solution, pass over to the next SQP iterate and go back to step 1.

Only step 4 has to be done in the feedback phase, steps 5, 1, 2 and 3 are done in the preparation phase. Note that due to the derivative calculation the preparation phase may take orders of magnitude longer than the feedback phase. In order to reduce the time of the whole iteration the number of directional derivatives (see Section 7) can be reduced as is done in the extended PRSQP approach of MSOPT.

The real-time iteration of MSOPT is built on these five steps:

(1) Calculate coordinate-basis S^N of consistency, continuity and initial conditions G_1.
(2) Compute residuals of G_2, H, reduced Jacobians $\nabla(F, G_2, H)^T S^N$ and reduced Hessian approximation.
(3) At the moment $\tilde{x}(t_0)$ is known, calculate inhomogeneity $S^R y^R$ and the projected function values of the restrictions: $G_2 + \nabla G_2^T S^R y^R$, $H + \nabla H^T S^R y^R$.
(4) Solve the reduced QP in $y^N := (\Delta q_0, \ldots, \Delta q_{m-1})$ and give the feedback control $q_0 + \Delta q_0$ to the real system.
(5) Expand the fully reduced QP solution to yield the full QP solution $\Delta w := S^R y^R + S^N y^N$. Based on this QP solution, pass over to the next SQP iterate and go back to step 1.

In this case, steps 3 and 4 have to be executed in the feedback phase and the remaining steps 5, 1 and 2 in the preparation phase. Since the differential and algebraic variables are eliminated in this PRSQP approach we have to calculate only directional derivatives w.r.t. combined control and state directions which reduces the preparation phase for the burden of a slightly longer feedback phase where the inhomogeneity has to be computed, too.

Both NMPC real-time iteration schemes are used to control the coupled distillation columns presented in the following section. The numerical results of the simulation experiments are presented in Section 10.4 and show how the low degrees of freedom of NMPC problem (1) can be exploited by MSOPT.

10. Application example: a thermally coupled distillation column

The NMPC schemes based on the software realizations MUSCOD-II and MSOPT are applied to the nonlinear DAE model of a thermally coupled distillation column with a side withdrawal to a rectifying column (cf. Fig. 2). This column configuration is used to separate a ternary mixture of methanol (light boiler, A), ethanol (intermediate boiler, B) and 1-propanol (heavy boiler, C). The energy saving potential associated with thermally coupled distillation columns has attracted interest of both the process industries and the process engineering research community ([23,47] and references therein). Accordingly, current studies focus on the design and control (cf. [15] for the closed operation of a coupled batch distillation column) and on the difficult static and dynamical behavior [47]. The latter has been identified as a bottleneck for practical applications [20] and is due to the strong coupling between controls and outputs, the presence of nonminimum phase behavior and strong nonlinearities. With this in mind, a working NMPC scheme does not only demonstrate the feasibility of the underlying numerical methods, but also contributes to the practical applicability of thermally coupled distillation columns in general.

10.1. The distillation column model

The model of a coupled distillation column with side withdrawal and an additional rectifying column that we use for simulation purposes has been derived in Ref. [30], where fur-

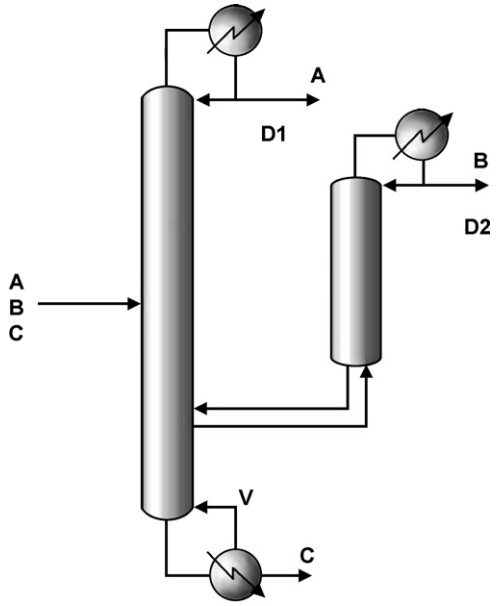


Fig. 2. Coupled distillation column with side withdrawal for the separation of a ternary mixture of A, B, and C (the schematic drawing is taken from Ref. [47]). The controls are the boil-up V and the distillate flows D_1 and D_2.

ther details about the control of coupled columns can be found. More recently, the model has also been used in Ref. [28] for the demonstration of a hierarchical controller for start-up procedures.

The main column consists of 42 stages (including boiler and condenser stage). The side withdrawal is located at stage 11, the feed enters the column at stage 21. The rectifying column consists of 10 stages and an additional condenser stage. The model is derived under some typical assumptions:

- chemical and thermal equilibrium on each stage;
- constant liquid holdup on all stages;
- negligible vapor holdup;
- perfect mixing with ideal gas phase;
- constant pressure throughout the columns;
- total condenser behavior;
- saturated feed and reflux liquid flows.

An overall mass balance and component mass balances for A and B yield differential equations for the composition on each tray. The phase equilibrium of the ternary mixture on each tray is described by constant volatilities relative to 1-propanol in form of algebraic equations. The concentration of component C in both liquid and vapor phase is computed with the summation equation, leading to another set of algebraic equations. The overall nonlinear algebraic-differential model consists of 106 differential and 159 algebraic variables. The model equations are sketched in the following. Each tray takes the general form:

$$H\dot{x}^I_{i,n} = F^I_{\text{in}}x^I_{i,n+1} - F^I_{\text{out}}x^I_{i,n} + V^I_{\text{in}}y^I_{i,n-1} - V^I_{\text{out}}y^I_{i,n} \tag{11}$$

where I denotes the column ($I=1$ is the main column, $I=2$ is the side column), $i=\{\text{A, B}\}$ denotes the component, and n is the tray number. F_{in} and F_{out} are the liquid flows into and out of tray n, while V represents the corresponding vapor streams. H is the molar holdup of a tray. Molar composition in liquid phase is denoted by x, while the vapor phase composition writes y. The flows F and V vary depending on the position of the trays because of the different inlets and outlets. The controls influence the column dynamics by manipulating these internal liquid and vapor flows. For example, the reboiler of the main column is described by the following equation:

$$H\dot{x}^1_{i,0} = F^1_{\text{in}}x^1_{i,1} - F^1_{\text{out}}x^1_{i,0} - V^1_{\text{out}}y^1_{i,0} \tag{12}$$

with the flows $F^1_{\text{in}} = V - D_1 - D_2 + F$, $F^1_{\text{out}} = -D_1 - D_2 + F$, $V^1_{\text{in}} = 0$, and $V^1_{\text{out}} = V$ (cf. Fig. 2). F is the feed to the main column. The third component C is determined by:

$$x^I_{\text{C},n} = 1 - x^I_{\text{A},n} - x^I_{\text{B},n} \tag{13}$$

and the same holds for $y^I_{\text{C},n}$ in the vapor phase. The phase equilibrium takes the form:

$$y^I_{i,n} = \frac{\alpha_i x^I_{i,n}}{1 + \alpha_{\text{A}}x^I_{\text{A},n} + \alpha_{\text{B}}x^I_{\text{B},n}}, \quad i = \text{A, B} \tag{14}$$

The coupled column is controlled in a D–V configuration with the two-distillate flow rates D_1 and D_2 and the boil-up rate V being the controls (also called manipulating variables, MV). The controlled variables (CV) are the molar concentrations of the desired products in the liquid phase: methanol in the main column distillate (x_{A}), ethanol in the rectifying column distillate (x_{B}) and 1-propanol in the bottoms of the main column (x_{C}). All model parameters are summarized in Table 1.

The steady state of the column is obtained by simulating the model with the specified parameters with a simulation time of 4 days without changes. The steady state values of the CV serve as setpoints (SP) for the simulation experiments.

As noted before, thermally coupled distillation columns exhibit a whole range of complex dynamics that complicate any control task. Fig. 3 shows the response of the considered controlled variables to a drop of 20% in the boil-up flow. It can be seen that the time constants (rise time, settling time) between

Table 1
List of process parameters and initial values

Parameter	Value
Holdup of trays in column 1	1.0 mol
Holdup of condenser column 1	4.17 mol
Holdup of reboiler column 1	114 mol
Holdup of trays in column 2	0.2 mol
Holdup of trays in column 2	3.08 mol
Rel. volatility methanol α_{A}	3.7
Rel. volatility ethanol α_{B}	2.05
Side outlet flow rate	0.0265 mol/s
Feed flow rate F	0.0555 mol/s
Feed composition methanol $x_{\text{A},F}$	0.3
Feed composition ethanol $x_{\text{B},F}$	0.1
Initial concentration methanol $x_{\text{A},0}$	0.3
Initial concentration ethanol $x_{\text{B},0}$	0.1
Nominal value D_1	0.016840 mol/s
Nominal value D_2	0.004732 mol/s
Nominal value V	0.12700 mol/s

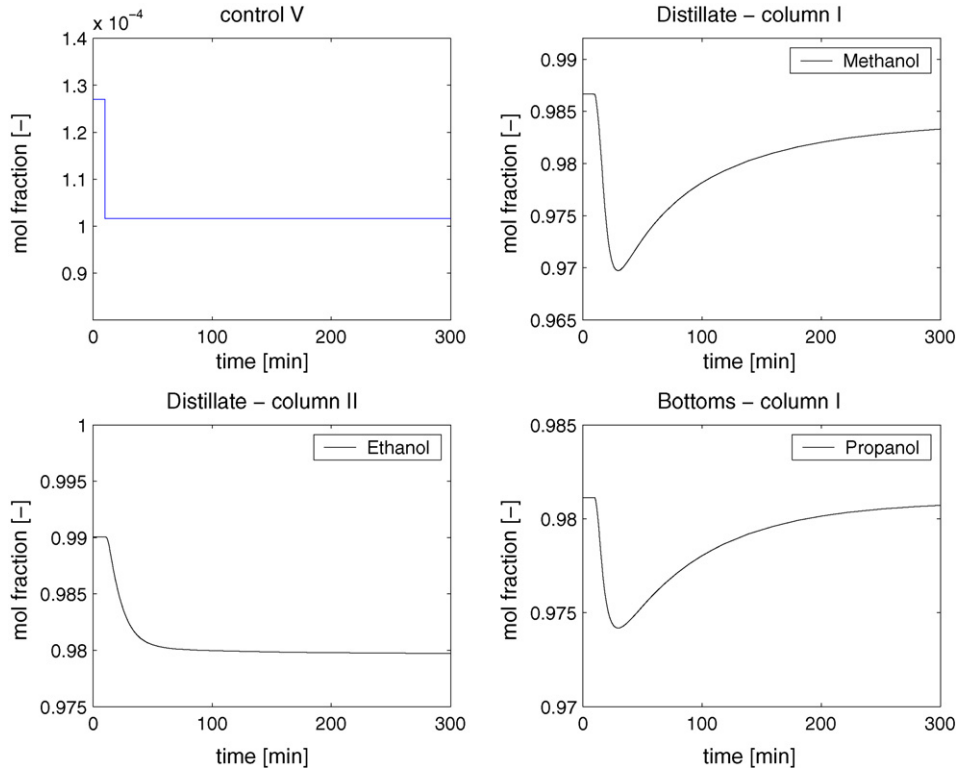


Fig. 3. Dynamic response of x_A, x_B, and x_C to a 20%-drop in the control V (boil-up). The other controls remain at their nominal values.

components A and C on the one hand and B on the other hand differ significantly. For the same step change in V and an additional small step of -0.08×10^{-5} mol/s in the distillate flow D_1, the concentration of methanol in the head of the main column even shows nonminimum phase behavior (see Fig. 4). In this situation, conventional PI control is known to show limited control performance (indeed, the limitations can even occur without the nonminimum phase behavior by closing single control loops, see [13]). An indication of strong nonlinearity is the gain change of the input–output pairing D_2 to the distillate in the side column (x_B). While for small changes in D_2 the steady state gain is negative, the gain becomes positive for larger steps as can be verified through simulations. All these difficulties turn the design of linear controllers for this coupled column into a challenging task.

10.2. The control task

In all simulation experiments presented here, the distillation column is assumed to be driven away from steady state by a (measurable) disturbance that occurs at a given time and lasts. The control task then is to steer back the controlled variables (CV) to their reference values which reflect the demanded purity specifications. Note, that due to the disturbance, the entire system will have a new steady state. The described scenario complies with practical demands, especially for highly integrated plant networks where the output of one plant serves, at least partially, as an input to the next plant. In these cases, a change in the production rate of one plant, e.g., due to changed market demands, will result in a lasting disturbance for the subsequent plants, while they have to keep their individual targets and specifications.

In the following, we consider two cases, where the disturbances have been chosen to meet a realistic scenario—Case I: the feed flow rate to the column is reduced considerably by $\Delta F = 0.0055$ to $F = 0.05$ mol/s, which is a reduction of about 10% of the nominal case. Case II: the feed flow composition has been changed from $x_{A,F} = 0.3$ to 0.2 and from $x_{B,F} = 0.1$ to 0.15 at the same time.

10.3. NMPC setup

For both NMPC schemes we have chosen the same setup of parameters that arise with the use of model predictive control. Most of the parameters are dictated by the time constants of the process to be controlled. They are summarized in Table 2. The control and prediction horizons have been chosen long enough to ensure that the process in closed-loop will reach steady

Table 2
List of controller parameters

Parameter	Value
Sampling time	100 s
Control discretization	600 s
Prediction horizon	3600 s
Control horizon	3600 s
Weights on CVs: Q_{ii}	100.0
Weights on MVs: R_{ii}	0.0
Multiple shooting intervals m	6
Accuracy of optimizer	10^{-3}

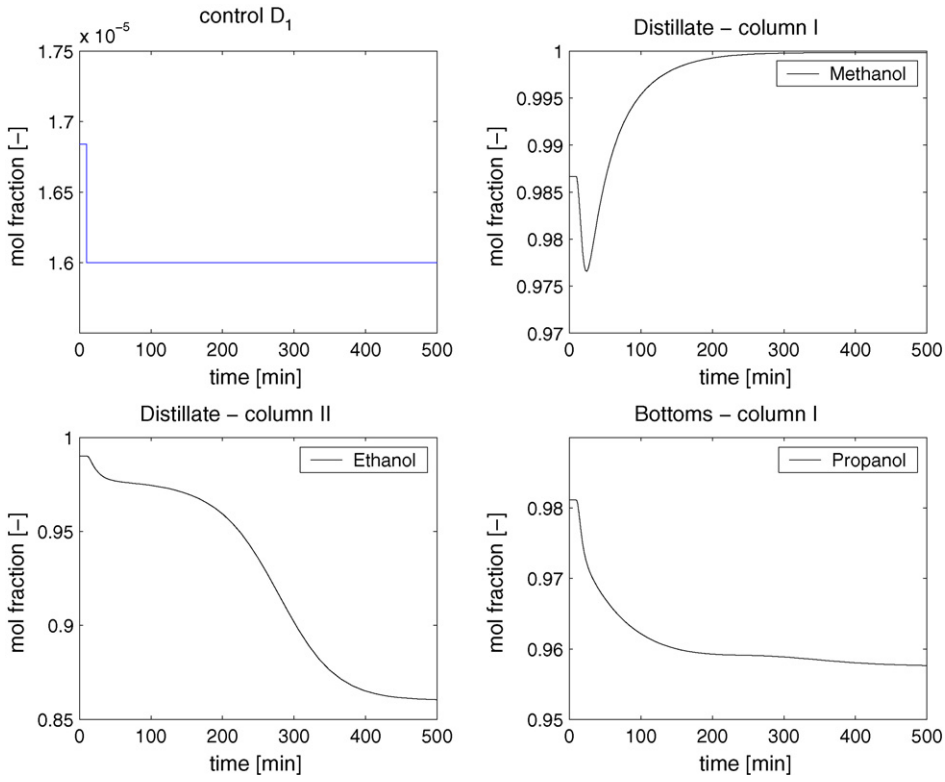


Fig. 4. Dynamic response of x_A, x_B, and x_C to a 20%-drop in the boil-up V and an additional small step of -0.08×10^{-5} mol/s in the distillate flow D_1. Note that the output variable x_A shows non-minimum phase behavior.

state after a single step disturbance within this time. The process itself is open-loop stable. With these two considerations in mind it proved unnecessary to use especially designed stabilizing NMPC schemes. In particular, no terminal constraints have been imposed and no terminal costs accounting for the finite horizon have been added (for a summary on stabilizing NMPC schemes see [35]). In our example we consider state feedback, i.e., we assume that all needed states are measured. In practical applications this is usually not the case. Then, an observer has to be used together with the controller, leading to an output feedback formulation (cf. [27] for an overview). Alternatively, the model has to be reformulated in a way that measurements are available for all modeled states. For example, one could model the distillation columns using temperatures instead of concentrations on each tray. Because of the strong correlation between concentrations and temperatures the control task of keeping concentrations at a given setpoint can then be reformulated as keeping corresponding temperatures constant. Numerical results for both NMPC schemes along with their computing times are presented in the next subsection.

10.4. Numerical results

The NMPC schemes are simulated within a Mat-lab control environment that calls the DAE solver, DAESOL [3], to simulate the process for one sampling time and delivers the current states and parameters to the controller. Then the control environment is waiting for the results of the dynamic optimization performed by MUSCOD-II or MSOPT, depending on the desired NMPC variant. It reads the available results from a communication file and gives the new controls to the DAE solver for the simulation of the next sampling period. This procedure is repeated for each sampling interval until the end of the simulation time is reached. The computation times for the NMPC schemes are measured in CPU time needed for calculations. We consider all numerical steps to find the controls but we do not count time spent on communication tasks which do not contribute to the solution of the subsequent optimization and feedback problems.

For comparison, the two different numerical NMPC schemes are demonstrated in two further modes. In the first mode, we allow the underlying optimization method to converge to a predefined accuracy (weighted norm of the SQP step Δw). In the second mode, we apply the real-time iteration described in Section 9. All simulations have been run on an Intel Pentium 4 machine with 2.8 GHz, 1024 kB L2 cache, 1 GB main memory, under Linux operating system Suse 9.3.

10.4.1. Case I: feed rate disturbance

At time $t = 10$ min the feed rate entering the main distillation column is changed from $F = 0.0555$ to $F = 0.05$ mol/s. Fig. 5 shows the closed loop behavior of the controlled variables (CVs) for both MUSCOD-II and MSOPT NMPC schemes in "converged" and in "real-time iteration" mode. The real-time iteration mode clearly differs from the converged mode in the transient phase. As described in Section 9, the real-time iteration delivers a second order prediction of the exact solution. Hence it does not fully recover the exact optimal solution but gets sufficiently close to it to control the process and quickly reject the

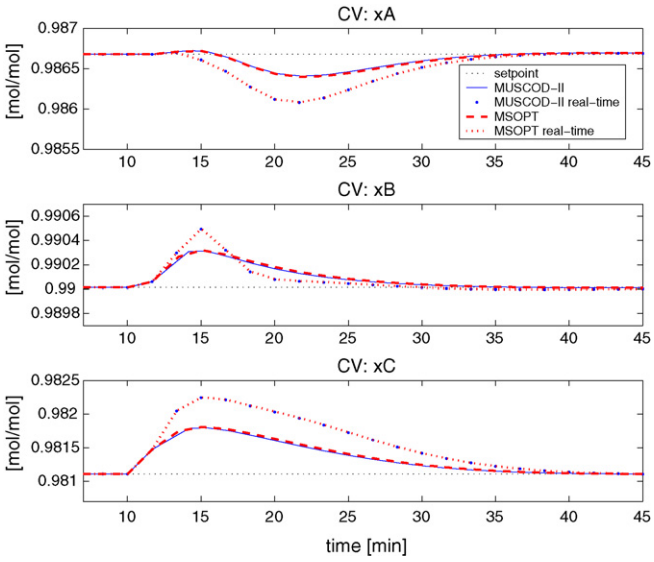


Fig. 5. Case I: NMPC closed loop responses of x_A, x_B, and x_C to a step disturbance in the feed concentration. The NMPC schemes differ numerically (MUSCOD-II, MSOPT) and in the number of SQP iterations (convergence to specified accuracy vs. real-time iteration—denoted 'real-time' in the legend).

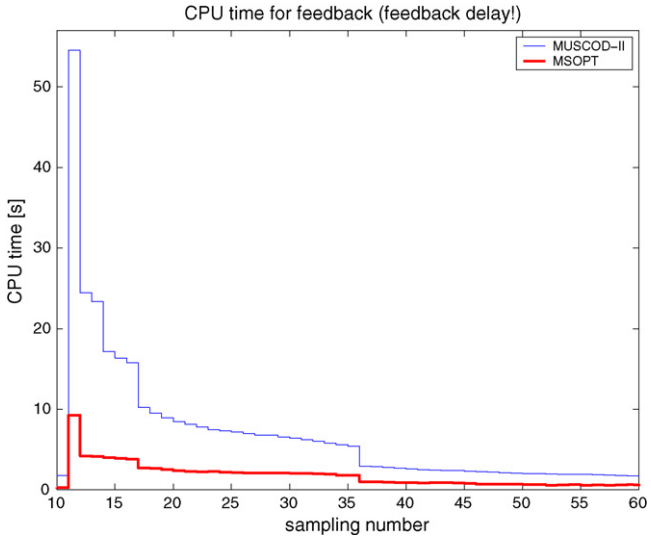


Fig. 7. CPU times for the feedback and the preparation phase at each sampling time for the numerical NMPC schemes (MUSCOD-II, MSOPT) with *convergence to desired accuracy*. The maximum feedback delay is 54.55 s (MUSCOD-II) and 9.26 s (MSOPT).

disturbance. A corresponding behavior is observed in Fig. 6, where the three manipulated variables (MV) are plotted for the four different NMPC versions. Another difference between the "converged" and the "real-time iteration" mode becomes obvious: while in the first mode the exact solution is available soon after the disturbance has been detected, the second has to slide into the exact solution along with the evolving process. This leads to an increase in the controller action.

The control performance of both numerical NMPC schemes, MUSCOD-II and MSOPT, is very good. We have a fast disturbance rejection with only little deviation from the desired setpoints in the transient phase with advantages for the converged case.

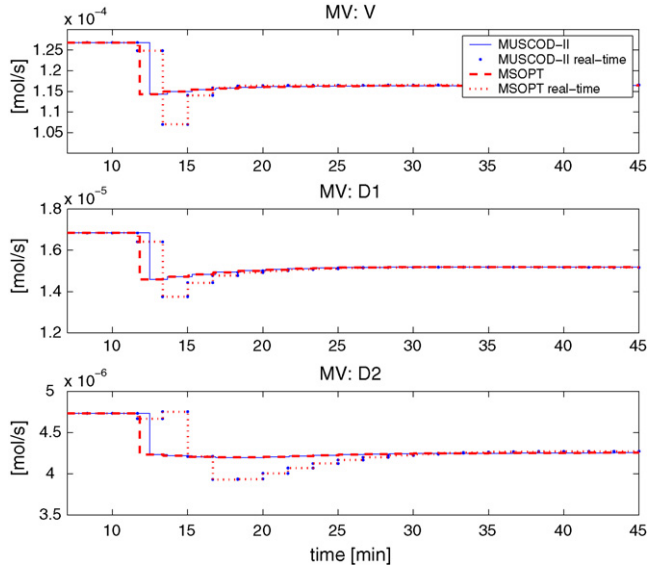


Fig. 6. Case I: NMPC manipulating variables for the disturbed feed concentration, calculated by MUSCOD-II and MSOPT to desired accuracy and with real-time iterations (real-time). Real-time iterations have almost no feedback delay. Note that a delay of 1.67 min is due to the sampling delay (sampling time: 100 s) and not caused by computation times.

The differences of the NMPC schemes lie in their computation times. Fig. 7 shows the CPU time of the converged mode for both the MUSCOD-II and the MSOPT scheme for calculating the controls. The maximum CPU time occurs right after the disturbance has been detected. It amounts to 54.55 s for the MUSCOD-II scheme, which is still below the sampling time of 100 s. The MSOPT scheme is able to reduce this feedback time to 9.26 s. Note that the actual feedback delay is even slightly shorter because some parts of the calculation can be done before the new states are available. The simulation results show that even the feedback delay of about half a sampling time is tolerable. However, using the faster MSOPT variant one could reduce the overall sampling time and obtain improved disturbance rejection properties. When discussing delays, it is important to note that there are two different sources. One source is the computation time which is necessary to obtain the optimal control based on the latest information about the process. Another source is the sampling time itself. The disturbance occurs at time $t = 10$ min but it is only detected one sampling time later, i.e., at about $t = 11.67$ min. This measurement delay cannot be reduced by the optimization schemes, no matter how fast they are.

To almost completely avoid feedback delays due to computation times, the real-time iteration idea has been developed. See Fig. 8 for a graphical representation of both feedback and preparation CPU time for the "real-time iteration" mode. Here, the maximum feedback time is reduced to only 0.81 s (MSOPT) and 0.0026 s (MUSCOD-II). The reason for the larger feedback time for MSOPT lies in the additional integration effort for calculating the inhomogeneity needed to set up the reduced QP and compute the feedback control. The feedback delay in the range of milliseconds comes very close to no feedback delay at

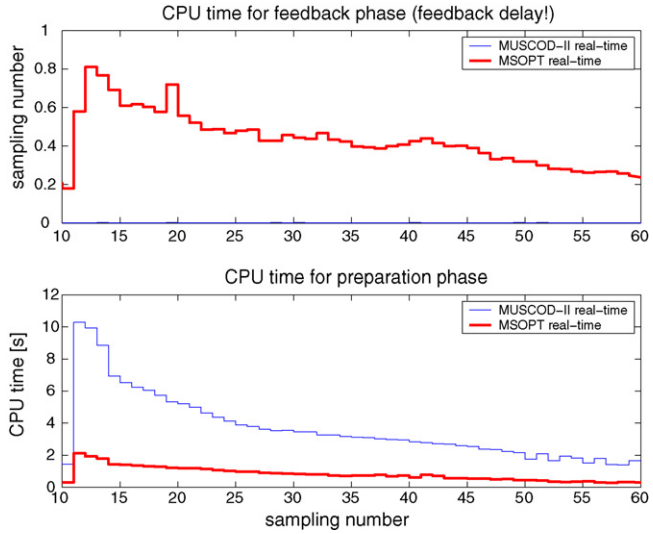


Fig. 8. CPU times for the feedback and the preparation phase at each sampling time for the numerical NMPC schemes (MUSCOD-II, MSOPT) with *real-time iterations*. Note that the feedback time for MUSCOD-II is in the range milliseconds!

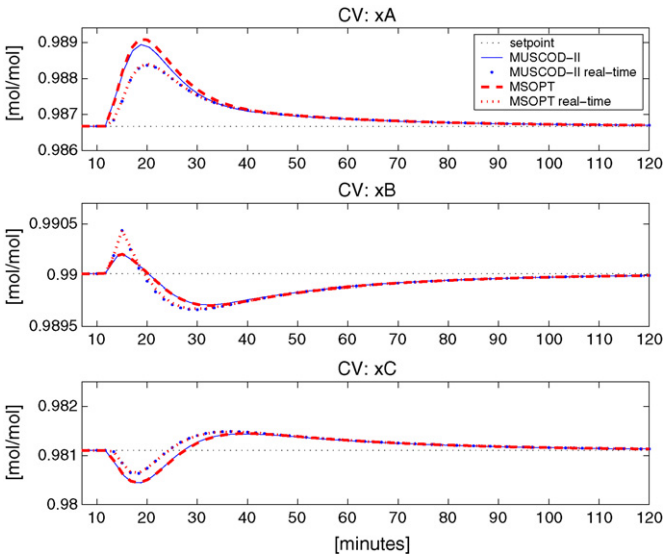


Fig. 9. Case II: step disturbance in feed composition. Shown are NMPC closed loop responses of x_A, x_B, and x_C in distillates and bottoms. The NMPC schemes differ numerically (MUSCOD-II, MSOPT) and in the number of SQP iterations (convergence to specified accuracy vs. real-time iteration—denoted 'real-time' in the legend).

all. However, one has to pay a price for these extremely short times achieved with the MUSCOD-II variant: the preparation time involving derivative calculation and linear algebra has a maximum of about 10 s, which is ca. five times longer than the comparable preparation time of MSOPT for the real-time iteration scheme. This limits possible reductions of the overall sampling time. Here, the MSOPT scheme is of advantage again because of the reduced solution space and lower number of directions of the directional derivatives. Note that during steady state the number of integration steps are much lower than at times where the system is perturbed. This explains the almost identical feedback computation times of MUSCOD-II and MSOPT before the perturbation and when the steady state is reached again.

10.4.2. Case II: feed composition disturbance

In this disturbance case, at time $t = 10$ min the composition of the feed entering the process is changed from $x_{A,F} = 0.3$ to 0.2 and from $x_{B,F} = 0.1$ to 0.15. The closed loop behavior of CVs and MVs is depicted in Figs. 9 and 10, again for both MUSCOD-II and MSOPT in "converged" and in "real-time iteration" mode. The qualitative behavior does not differ much from the one described in case I. Only, this time the control performance of the real-time iteration schemes is slightly better than for the converged case. This is due to the significantly larger feedback delays for the "converged" mode. In case II it pays off to provide feedback almost instantaneously instead of waiting for a solution that has converged to desired accuracy. The CPU times in Figs. 11 and 12 also correspond to the findings in case I. This time, since the perturbation of the process is stronger, the optimization procedure needs longer to converge. The maximum feedback time of the MUSCOD-II scheme in convergence mode is 69.21 s in comparison to 12.79 s of MSOPT. When applying the real-time iteration scheme, the feedback delay is reduced to a maximum of 0.0027 s (MUSCOD-II) and 0.89 s (MSOPT), which is very close to the values in case I and emphasizes the real-time character of these NMPC schemes. From a theoretical point of view, one has to see that despite the reductions presented above, at least a very small feedback delay is still present. This raises questions about stability of the closed-loop and theoretical performance limitations and has been studied in Refs. [12,22]. A summary of the computation times can be found in Table 3.

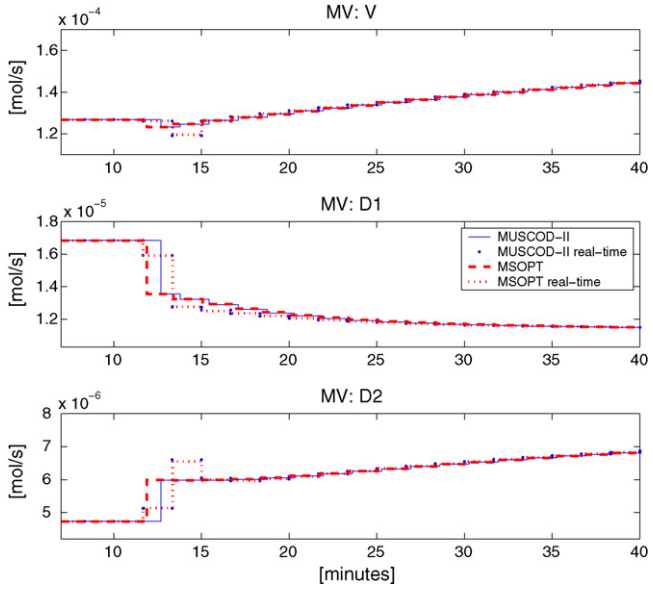


Fig. 10. Case II: corresponding NMPC manipulating variables for the disturbed feed composition case. Note the different feedback delays due to different (feedback) computation times. Also note that a delay of 100 s is always present due to the sampling delay independent from any computation times.

Table 3
Summary of CPU times for the two disturbance cases (step in feed flow rate, step in feed composition)

CPU times (s) NMPC mode phase	Feed rate		Feed composition	
	Max	Average	Max	Average
MUSCOD-II feedback converged	54.55	6.10	69.21	11.91
MSOPT feedback converged	9.26	1.60	12.79	3.01
MUSCOD-II feedback	0.0026	0.0021	0.0027	0.0018
Real-time iteration preparation	10.29	3.36	13.90	5.58
MSOPT feedback	0.81	0.41	0.89	0.58
Real-time iteration preparation	2.12	0.76	2.33	1.27

The CPU time for each NMPC scheme (MUSCOD-II, MSOPT) and each mode (converged, real-time iteration) is shown, divided into feedback and preparation phase for the real-time mode.

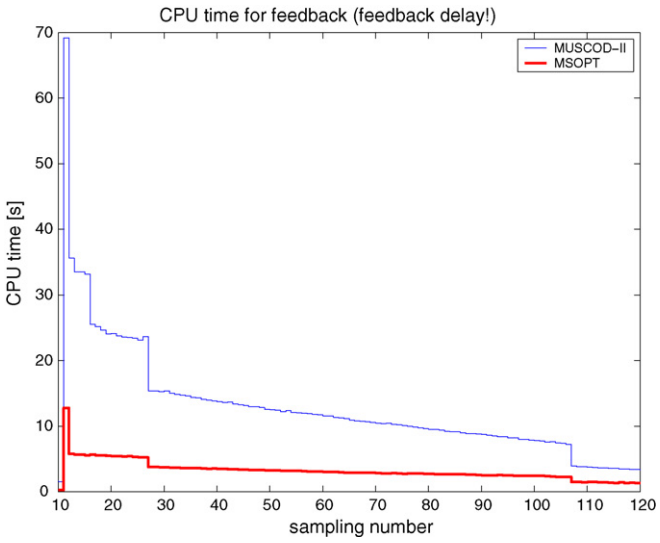


Fig. 11. Case II: CPU times for the feedback and the preparation phase at each sampling time for the numerical NMPC schemes (MUSCOD-II, MSOPT) with *convergence to desired accuracy*. The maximum feedback delay here is 69.21 s (MUSCOD-II) and 12.79 s (MSOPT).

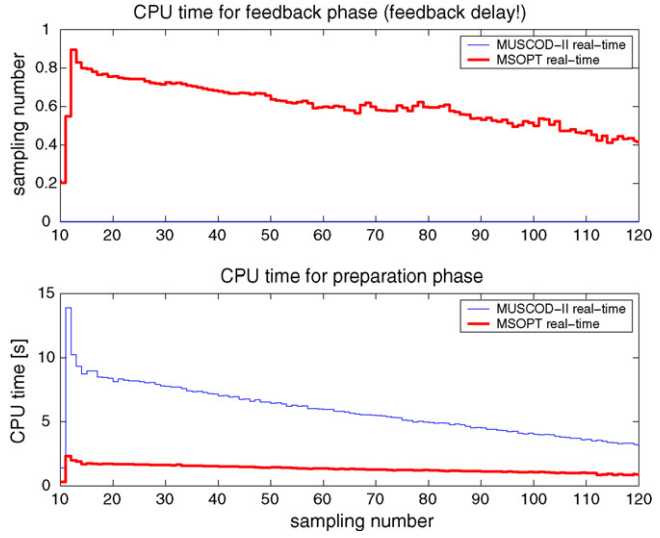


Fig. 12. Case II: CPU times for the feedback and the preparation phase at each sampling time for the numerical NMPC schemes (MUSCOD-II, MSOPT) with *real-time iterations*. The maximum preparation time (MUSCOD-II) is 13.9 s. Note that the feedback time for MUSCOD-II is in the range milliseconds!

10.4.3. Decentralized PI control

To compare the NMPC controllers with conventional control strategies, a decentralized PI (proportional-integral) controller has been designed for the considered thermally coupled distillation column. To design the controller, step experiments have been carried out with the fully nonlinear model simulating a real plant. The step responses from the three MV to the three CV have then been used to formulate an approximate input–output model of the process based on first and second order dynamics with inputs U and outputs Y. The obtained model of the form $Y(s) = G(s)U(s)$ in the frequency domain with $G(s)$ being the transfer function matrix has been the starting point for the decentralized PI control design (in decentralized control, a multi-input multi-output (MIMO) control problem is solved by an appropriate number of closed single-input single-output loops). The chosen design method is the Direct Nyquist approach as described in Ref. [34], which is based on designing the single loops and then checking stability of the combined loops in the frequency domain. Simulations show that the obtained PI control is indeed stable and allows to reject the distur-

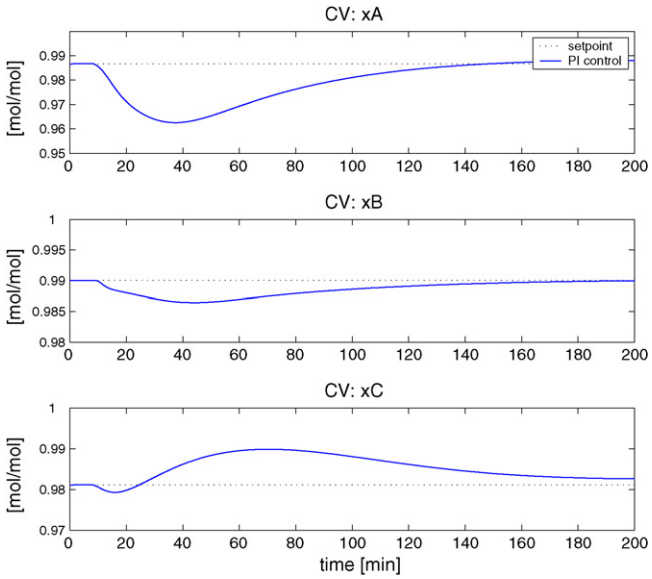


Fig. 13. Case I: decentralized PI control for a step disturbance in the feed flow rate.

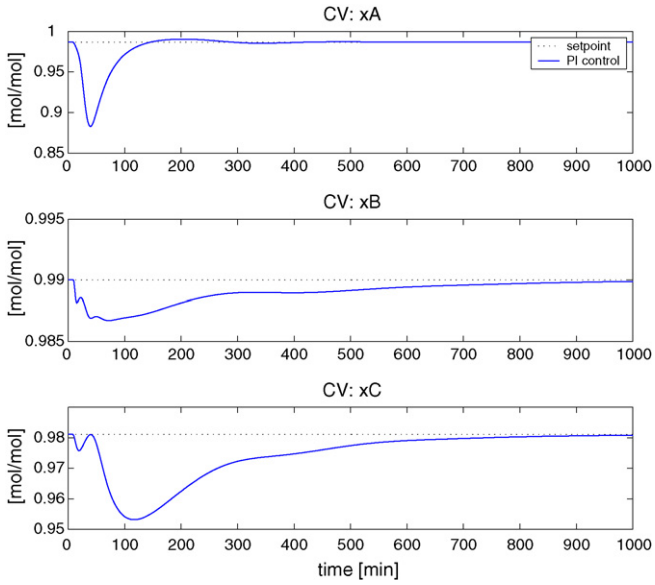


Fig. 14. Case II: decentralized PI control for a step disturbance in the feed concentration.

bances. The control performance, however, is clearly inferior with respect to both settling time and deviation from the setpoint as can be seen in Figs. 13 and 14. This is not very surprising because any control action is only based on setpoint deviations and no model information is actively used. Furthermore, the PI controller does not use any knowledge about the occurring disturbances, although measurements thereof are available. The latter can be circumvented by adding feed forward action. For a better visualization of the different control performances, Fig. 15 shows the process behavior for case I without control and with both the NMPC and the decentralized PI controller in closed-loop.

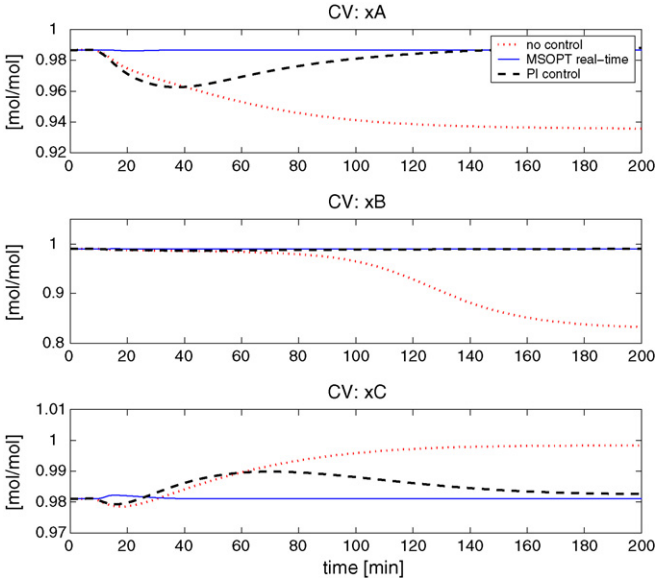


Fig. 15. Comparison of NMPC (MSOPT in real-time iteration mode), decentralized PI-control and no control for case I (feed disturbance).

11. Conclusions

We have introduced a new optimization algorithm, MSOPT, to be used for NMPC, which is capable of reducing the costs of calculating an SQP iteration. This is achieved by reducing the number of directions in the directional derivatives which have to be calculated to set up the reduced QP. Since the null-space basis of the partially reduced SQP method is explicitly calculated, handling of inequality constraints is efficiently possible via active set strategies. This is also valid for bounds on the states. In combination with the real-time iteration scheme, one can further reduce the computation time for a feedback control to the expense of one integration of the inhomogeneity and the solution of a reduced QP. Compared to a standard NMPC scheme based on MUSCOD-II, the preparation time for the next iteration is significantly reduced because of the lower number of directions of the directional derivatives that have to be computed in every SQP step. In fact, the number is now independent of the state dimension. The performance of the newly developed NMPC algorithm is shown for the control of a coupled distillation column for separating a ternary mixture of methanol, ethanol and 1-propanol with difficult static and dynamical behavior. It has been demonstrated that for two practical disturbance cases the real-time iteration algorithms are capable of delivering a feedback control nearly instantaneously. The preparation time for the next iteration was significantly reduced by the MSOPT variant, since the number of directions for the directional derivatives is significantly reduced (from 660 in MUSCOD-II to 132 in MSOPT). This was achieved at the expense of slightly larger feedback times for the MSOPT variant. The short preparation times leave the freedom to reduce the overall sampling time to react to process changes even more rapidly.

References

[1] F. Allgöwer, T.A. Badgwell, J.S. Qin, J.B. Rawlings, S.J. Wright, Nonlinear predictive control and moving horizon estimation—an introductory overview, in: P.M. Frank (Ed.), Advances in Control, Springer, 1999, pp. 391–449.

[2] R.D. Bartusiak, NMPC: a platform for optimal control of feed-or product-flexible manufacturing, in: Proceedings of the Assessment and Future Directions of NMPC, Freudenstadt-Lauterbad, Germany, 2005, pp. 3–14.

[3] I. Bauer, Numerische Verfahren zur Lösung von Anfangswertaufgaben und zur Generierung von ersten und zweiten Ableitungen mit Anwendungen bei Optimierungsaufgaben in Chemie und Verfahrenstechnik, Ph.D. Thesis, University of Heidelberg, 1999.

[4] L.T. Biegler, Efficient solution of dynamic optimization and NMPC problems, in: F. Allgöwer, A. Zheng (Eds.), Nonlinear Predictive Control, Birkhäuser, Basel, 2000, pp. 219–244.

[5] T. Binder, L. Blank, H.G. Bock, R. Burlisch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, O.v. Stryk, Introduction to model based optimization of chemical processes on moving horizons, in: M. Grötschel, S.O. Krumke, J. Rambau (Eds.), Online Optimization of Large Scale Systems, Springer, Berlin, Heidelberg, 2001, pp. 295–339.

[6] H.G. Bock, A direct multiple shooting method for real-time optimization of nonlinear DAE processes, in: F. Allgöwer, A. Zheng (Eds.), Nonlinear Predictive Control, Birkhäuser, Basel, 2000, pp. 245–267.

[7] H.G. Bock, Numerical treatment of inverse problems in chemical reaction kinetics, in: K.H. Ebert, P. Deuflhard, W. Jäger (Eds.), Modelling of Chemical Reaction Systems (Springer Series in Chemical Physics 18), Springer, Heidelberg, 1981, pp. 102–125.

[8] H.G. Bock, Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen, volume 183 of Bonner Mathematische Schriften, University of Bonn, Bonn, 1987.
[9] H.G. Bock, E. Eich, J.P. Schlöder, Numerical solution of constrained least squares boundary value problems in differential–algebraic equations, in: K. Strehmel (Ed.), Numerical Treatment of Differential Equations, Teubner, Leipzig, 1988.
[10] H.G. Bock, K.-J. Plitt, A multiple shooting algorithm for direct solution of optimal control problems, in: Proceedings of the 9th IFAC World Congress, Budapest, Pergamon Press, 1984.
[11] B.L. Braunschweig, C.C. Pantelides, H.I. Britt, S. Sama, Process modeling: the promise of open software architectures, Chem. Eng. Prog. 96 (9) (2000) 65–76.
[12] W. Chen, D.J. Ballance, J. O'Reilly, Model predictive control of nonlinear systems: computational delay and stability, IEE Proc. Contr. Theory Appl. 147 (4) (2000) 387–394.
[13] H. Cui, E.W. Jacobsen, Performance limitations in decentralized control, J. Proc. Contr. 12 (2002) 485–494.
[14] A.R. Curtis, M.J.D. Powell, J.K. Reid, On the estimation of sparse Jacobian matrices, J. Inst. Math. Appl. 13 (1974) 117–119.
[15] D. Demicoli, J. Stichlmair, Separation of ternary mixtures in a batch distillation column with side withdrawal, Comp. Chem. Eng. 28 (2004) 643–650.
[16] M. Diehl, Real-Time Optimization for Large Scale Nonlinear Processes, volume 920 of Fortschr.-Ber. VDI Reihe 8, Meß, Steuerungs-und Regelungstechnik, VDI Verlag, Dusseldorf, 2002.
[17] M. Diehl, H.G. Bock, J.P. Schlöder, R. Findeisen, Z. Nagy, F. Allgöwer, Real-time optimization and nonlinear model predictive control of processes governed by differential–algebraic equations, J. Process Contr. 12 (2002) 577–585.
[18] M. Diehl, H.G. Bock, J.P. Schlöder, A real-time iteration scheme for nonlinear optimization in optimal feedback control, SIAM J. Contr. Optim. 43 (5) (2005) 1714–1736.
[19] M. Diehl, R. Findeisen, F. Allgöwer, H.G. Bock, J.P. Schloder, Nominal stability of real-time iteration scheme for nonlinear model predictive control, IEE Proc. Contr. Theory Appl. 152 (3) (2005) 296–308.
[20] G. Dunnebier, C. Pantelides, Optimal design of thermally coupled distillation columns, Ind. Eng. Chem. Res. 38 (1999) 162.
[21] H. Elmqvist, S.E. Mattsson, M. Otter, Modelica: the new object-oriented modeling language, in: Presented at the 12th European Simulation Multiconference, Manchester, UK, 1998.
[22] R. Findeisen, F. Allgöwer, Computational delay in nonlinear model predictive control, in: Proc. Int. Symp. Adv. Control of Chemical Processes (ADCHEM'03), 2003.
[23] A.J. Finn, Rapid assessment of thermally coupled side columns, Gas. Sep. Purif. 10 (3) (1996) 169–175.
[24] R. Fletcher, Practical Methods of Optimization, John Wiley & Sons, 1987.
[25] D. Gabay, Reduced Quasi-Newton methods with feasibility improvement for nonlinear constrained optimization, Math. Program. Study 16 (1982) 18–44.
[26] A. Griewank, D. Juedes, H. Mitev, J. Utke, O. Vogel, A. Walther, ADOL-C: a package for the automatic differentiation of algorithms written in C/C++, ACM TOMS 22 (2) (1996) 131–167.
[27] L. Imsland, R. Findeisen, E. Bullinger, F. Allgöwer, B.A. Foss, A note on stability, robustness and performance of output feedback nonlinear model predictive control, J. Process Contr. 13 (2003) 633–644.
[28] A. Itigin, J. Raisch, T. Moor, A. Kienle, A two-level hybrid control strategy for the start-up of a coupled distillation plant, in: European Control Conference 2003 (ECC'2003), Cambridge, UK, September 1–4, 2003.
[29] F.-S. Kupfer, An infinite-dimensional convergence theory for reduced SQP methods in Hilbert space, SIAM J. Optim. 6 (1) (1996) 126–163.
[30] L. Lang, Prozeßführung gekoppelter Mehrstoffkolonnen am Beispiel einer Destillationsanlage mit Seitenabzug, Ph.D. Thesis, Universitat Stuttgart, 1991.
[31] D.B. Leineweber, The Theory of MUSCOD in a Nutshell, IWR-Preprint 96-19, University of Heidelberg, 1996.
[32] D.B. Leineweber, Efficient Reduced SQP Methods for the Optimization of Chemical Processes Described by Large Sparse DAE Models, volume 613 of Fortschr.-Ber. VDI Reihe 3, Verfahrenstechnik, VDI Verlag, Düsseldorf, 1999.
[33] D.B. Leineweber, I. Bauer, H.G. Bock, J.P. Schlöder, An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimisation. Part 1. Theoretical aspects, Comp. Chem. Eng. 27 (2003) 157–166.
[34] J. Lunze, Regelungstechnik 2, Springer-Verlag, Berlin, 1997.
[35] L. Magni, R. Scattolini, Stabilizing model predictive control of nonlinear continuous time systems, Ann. Rev. Contr. 28 (2004) 1–11.
[36] D.Q. Mayne, J.B. Rawlings, C.V. Rao, P.O.M. Scokaert, Constrained model predictive control: stability and optimality, Automatica 36 (2000) 789–814.
[37] P. Mhaskar, N. El-Farra, P.D. Christofides, Robust hybrid predictive control of nonlinear systems, Automatica 41 (2005) 209–217.
[38] J. Nocedal, M.L. Overton, Projected hessian updating algorithms for nonlinearly constrained optimization, SIAM J. Numer. Anal. 22 (5) (1985) 821–850.
[39] K.J. Plitt, Eiin superlinear konvergentes Mehrzielverfahren zur direkten Berechnung beschränkter optimaler Steuerungen, Diploma Thesis, University of Bonn, 1981.
[40] M.J.D. Powell, A fast algorithm for nonlinearly constrained optimization calculations, in: G.A. Watson (Ed.), Numerical Analysis, number 630 in Lect. Not. in Math., 1978, pp. 144–157.
[41] S.J. Qin, T.A. Badgewell, A survey of industrial model predictive control technology, Contr. Eng. Pract. 11 (2003) 733–764.
[42] A. Schäfer, Effiziente reduzierte Newton-ähnliche Verfahren zur Behandlung hochdimensionaler strukturierter Optimierungsprobleme mit Anwendung bei biologischen und chemischen Prozessen, Ph.D. Thesis, University of Heidelberg, 2004.
[43] J.P. Schlöder, Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung, volume 187 of Bonner Mathematische Schriften, University of Bonn, Bonn, 1988.
[44] V.H. Schulz, Reduced SQP Methods for Large-Scale Optimal Control Problems in DAE with Application to Path Planning Problems for Satellite Mounted Robots, Ph.D. Thesis, Ruprecht-Karls-Universität Heidelberg, 1996.
[45] V.H. Schulz, Solving discretized optimization problems by partially reduced SQP methods, Comput. Vis. Sci. 1 (2) (1998) 83–96.
[46] V.H. Schulz, H.G. Bock, M.C. Steinbach, Exploiting invariants in the numerical solution of multipoint boundary value problems for DAEs, SIAM J. Sci. Comp. 19 (1998) 440–467.
[47] J.G. Segovia-Hernandez, S. Hernandez, V. Rico-Ramirez, A. Jimenez, A comparison of the feedback control behavior between thermally coupled and conventional distillation schemes, Comp. Chem. Eng. 28 (2004) 811–819.
[48] M.C. Steinbach, Fast recursive SQP methods for large-scale optimal control problems, Ph.D. Thesis, Ruprecht-Karls-Universität Heidelberg, 1995.