# An Efficient Algorithm for Optimization in Nonlinear Model Predictive Control of Large-Scale Systems

M. Diehl[1], R. Findeisen[2], S. Schwarzkopf [3], I. Uslu[3],
F. Allgöwer[2], H. G. Bock[1], E. D. Gilles[3,4], and J. P. Schlöder[1]

March 6, 2002

**Abstract**

The purpose of this paper is twofold: first, we present an efficient state-of-the-art algorithm for online optimization in nonlinear model predictive control (NMPC), the so called real-time iteration scheme; secondly, we give an experimental proof-of-concept for NMPC of large-scale systems, by presenting results obtained at a pilot-plant distillation column situated at the *Institut für Systemdynamik und Regelungstechnik* at the University of Stuttgart. A main advantage of the algorithm may be seen in the fact that nonlinear first principle system models – which are often developed anyway for process analysis and design – can in a straightforward way be reused for control purposes, even if they result in large state dimensions. The algorithm is able to treat generic system models of differential-algebraic equation type, as they are formulated in modelling environments, e.g. in gPROMS. In the experimental study, we use a standard model for a binary distillation column, which is stiff and employs more than 200 system states. The algorithm is able to provide reoptimized trajectories every 20 seconds. Experimental results for two scenarios are presented: first, the optimal transition of the distillation column into a new steady state, after a sudden feed flow change of 20%. The optimized transition only takes 15 minutes, which compares well with the 1-2 hours suggested by previous experience at the plant. In a second test, starting from a largely disturbed, overheated system state, the column was safely guided back into the operating point by the NMPC scheme, without violation of constraints.

## Introduction

Online optimization of dynamic process models and nonlinear model predictive control have attracted increasing attention over the past decade, in particular in chemical engineering [GPM89, BR91, SOB95, HAM98, Bie00, BBB+01]. Among the advantages of this approach are the flexibility provided in formulating the objective and the process model, the capability to directly handle equality and inequality constraints, and the possibility to treat unforeseen disturbances fast. It is in particular the availability of detailed nonlinear process models – that are increasingly being used for the *design* of industrial processes [PB93, RBP+99, Sor99] – which promises to make NMPC an appealing alternative to conventional control.

One important precondition for successful NMPC applications, however, is the availability of reliable and efficient numerical optimal control algorithms. These algorithms shall ideally be able to treat large-scale nonlinear first principle models as they are, without further need of modelling or model reduction. One particularly successful algorithm that is designed to achieve this aim, the recently developed *real-time iteration* scheme, will be the focus of this paper. The scheme is based on the direct multiple shooting method [BP84] for *differential algebraic equation (DAE)*

[1]Interdisziplinäres Zentrum für wissenschaftliches Rechnen (IWR), Universität Heidelberg
[2]Institut für Systemtheorie Technischer Prozesse (IST), Universität Stuttgart
[3]Institut für Systemdynamik und Regelungstechnik (ISR), Universität Stuttgart
[4]Max-Planck-Institut für Dynamik komplexer technischer Systeme, Magdeburg

models (Leineweber [Lei99]), as they often arise in practical applications. In its actual implementation, the scheme is realized as part of the optimal control package MUSCOD-II, which offers several advantages in the context of practical online optimization. Among these are the possibility to provide the DAE model equations as generic C or Fortran-Code or in the gPROMS modelling language [Pro00, LSBS02], to make use of efficient state-of-the-art DAE solvers (e.g. DDASAC [CS85], DAESOL [BBS99]), or to employ an existing parallelization in the portable MPI standard, in time critical cases.

The efficiency of the direct multiple shooting method for dynamic optimization, which has been observed in many practical applications, has several reasons. One of the most important is the possibility to incorporate information about the behaviour of the state trajectory into the initial guess for the iterative solution procedure; this can damp the influence of poor initial guesses for the controls (which are usually much less known). In the context of NMPC, where a sequence of neighbouring optimization problems is treated, solution information of the previous problem can be exploited efficiently by an *initial value embedding* strategy, which is explained in detail in Section 3.

For a description of direct multiple shooting and MUSCOD-II we refer to Leineweber [Lei99], Leineweber et al. [LBBS02, LSBS02], or Diehl et al. [DLS01]. For more details on the real-time iteration scheme itself we refer to Diehl [Die02] or Diehl et al. [DBS+02, DBS].

### Overview

The paper pursues two aims: first, the numerical optimization method is explained in a way which allows to capture the main algorithmic ideas. Second, to demonstrate that the method makes NMPC based on first principle models feasible for real world problems, the control of a distillation column is treated experimentally.

1. We start the paper by introducing a moving horizon optimal control problem formulation in Section 1.
2. The direct multiple shooting method is reviewed in Section 2.
3. A crucial ingredient of the real-time iteration scheme, namely the initial value embedding for initialization of subsequent optimization problems, is introduced and visualized in Section 3.
4. In Section 4 the full real-time iteration scheme is outlined step by step.
5. Finally, in Section 5 we show how the algorithm can be practically set up for application at a real system, a binary distillation column.
6. The obtained experimental results are presented in Section 6.
7. The paper ends with a concluding summary.

## 1 Online Optimal Control

Throughout this paper, we consider *differential algebraic equation* (DAE) models of index one in the following form

$$\begin{aligned} B(x(t), z(t), u(t), p)\, \dot{x}(t) &= f(x(t), z(t), u(t), p) \\ 0 &= g(x(t), z(t), u(t), p) \end{aligned}$$

Here, $x$ and $z$ denote the differential and the algebraic state vectors, respectively, $u$ is the vector valued control function, whereas $p$ is a vector of system parameters. Matrix $B(\cdot)$ is assumed to be invertible, so that the DAE is of semi-explicit type. This equation type is typical for practical chemical engineering applications.

For notational simplicity, we will omit the parameters $p$ in the following. Most parameters are constant and can be subsumed in the model equations. Those parameters which are expected to change during the runtime of the process and should therefore be observed in the online context, can be regarded as (constant) differential system states $x_p$ obeying the differential equation $\dot{x}_p = 0$.

## 1.1 Optimal Control Problem

Given an observed system state $\bar{x}_0$ at a time $t_0$, a *Nonlinear Model Predictive Control* (NMPC) control scheme obtains a feedback control $\bar{u}(\bar{x}_0)$ quite indirectly, from the solution of an optimal control problem on a prediction horizon $[t_0, t_0 + T_p]$ with length $T_p$:

$$\min_{u(\cdot),x(\cdot),z(\cdot)} \quad \int_{t_0}^{t_0+T_p} L(x(t), z(t), u(t))\, dt \quad + \quad E(x(t_0 + T_p)) \tag{1a}$$

subject to the initial value constraint

$$x(t_0) \quad = \quad \bar{x}_0 \tag{1b}$$

and the DAE system

$$B(\cdot)\dot{x}(t) \quad = \quad f(x(t), z(t), u(t)), \quad \forall t \in [t_0, t_0 + T_p], \tag{1c}$$

$$0 \quad = \quad g(x(t), z(t), u(t)), \quad \forall t \in [t_0, t_0 + T_p]. \tag{1d}$$

In addition, state and control inequality constraints

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

as well as terminal constraints

$$r(x(t_f)) \left\{ \begin{array}{c} = \\ \geq \end{array} \right\} 0. \tag{1f}$$

have to be satisfied. Solving this problem for a given initial value $\bar{x}_0$, we obtain an open-loop optimal control $u^*(t; \bar{x}_0)$ and corresponding state trajectories $x^*(t; \bar{x}_0), z^*(t; \bar{x}_0)$.

## 1.2 Nonlinear Model Predictive Control

Based on the above solution, the NMPC feedback control law is defined to be

$$\bar{u}(\bar{x}_0) := u^*(t_0; \bar{x}_0). \tag{2}$$

If the real system state $\bar{x}(t), \bar{z}(t)$ would obey the nominal DAE, the closed-loop system would develop according to the *nominal NMPC dynamics*:

$$\begin{aligned} B(\cdot)\, \dot{\bar{x}}(t) \quad &= \quad f(\bar{x}(t), \bar{z}(t), \bar{u}(\bar{x}(t))) \\ 0 \quad &= \quad g(\bar{x}(t), \bar{z}(t), \bar{u}(\bar{x}(t))). \end{aligned}$$

Due to its origin from an optimal control formulation, the NMPC feedback law has several appealing properties: among them are the possibility to base the feedback on economic criteria, to make use of important process knowledge in the form of nonlinear first principle models, and to include constraints (1e) in a straightforward way. Given suitable choices of the objective function and the final state constraint (1f), stability of the nominal NMPC dynamics can be proven even for nonlinear systems [May96, Che97, CA98, May00, DMS00].

One computationally expensive and storage consuming possibility to obtain the feedback law $\bar{u}(\bar{x}_0)$ would be to precalculate it off-line on a sufficiently fine grid. In contrast to this, the present paper is concerned with efficient ways to calculate this feedback control *in real-time*, during the process development.

# 2 Direct Multiple Shooting

The online solution of the optimal control problem (1a)–(1f) is based on the direct multiple shooting method, which is reviewed briefly in this section. This review prepares the presentation of the real-time iteration scheme in Sections 3 and 4.

## 2.1 Parametrization of the Infinite Optimization Problem

The parametrization of the infinite optimization problem consists of two steps. For a suitable partition of the time horizon $[t_0, t_f]$ into $N$ subintervals $[t_i, t_{i+1}]$ with

$$t_0 < t_1 < \ldots < t_N = t_0 + T_p$$

we first discretize the control function $u(\cdot)$. For simplicity, we assume here that it is parametrized as a piecewise constant vector function

$$u(t) = u_i, \text{ for } t \in [t_i, t_{i+1}],$$

but every parametrization with local support could be used without changing the structure of the problem.

In a second step the DAE are parametrized by *multiple shooting*. We decouple the DAE solution on the $N$ intervals $[t_i, t_{i+1}]$ by introducing the initial values $s_i^x$ and $s_i^z$ (combined: $s_i$) of differential and algebraic states at times $t_i$ as additional optimization variables.

On each subinterval $[t_i, t_{i+1}]$ we compute the trajectories $x_i(t)$ and $z_i(t)$ as the solution of an initial value problem:

$$\begin{aligned} B(\cdot)\dot{x}_i(t) &= f(x_i(t), z_i(t), u_i) && \text{(3a)} \\ 0 &= g(x_i(t), z_i(t), u_i) - \alpha_i(t) g(s_i^x, s_i^z, u_i) && \text{(3b)} \\ x_i(t_i) &= s_i^x && \text{(3c)} \end{aligned}$$

Here the subtrahend in (3b) is deliberately introduced to allow an efficient DAE solution for initial values and controls $s_i^x, s_i^z, u_i$ that may violate temporarily the consistency conditions (1d). Therefore we require for the scalar damping factor $\alpha$ that $\alpha_i(t_i) = 1$. For more details on the relaxation of the DAE the reader is referred, e.g. to [BES88, SBS98, Lei99]. Note that the trajectories $x_i(t)$ and $z_i(t)$ on interval $[t_i, t_{i+1}]$ are functions of the initial values, and controls $s_i^x, s_i^z, u_i, p$ only, so that we will refer to them as $x_i(t; s_i, u_i)$ and $z_i(t; s_i, u_i)$ in the following. The integral part of the cost function is evaluated on each interval independently:

$$L_i(s_i^x, s_i^z, u_i) := \int_{t_i}^{t_{i+1}} L(x_i(t; s_i, u_i), z_i(t; s_i, u_i), u_i)\, dt. \qquad (4)$$

## 2.2 Structured Nonlinear Programming Problem

The parametrization of problem (1a)–(1f) using multiple shooting and a piecewise constant control representation leads to the following structured nonlinear programming (NLP) problem :

$$\min_{u,s} \sum_{i=0}^{N-1} L_i(s_i^x, s_i^z, u_i) \quad + \quad E(s_N^x) \qquad (5a)$$

subject to the initial value constraint

$$s_0^x = \bar{x}_0, \qquad (5b)$$

the continuity conditions

$$s_{i+1}^x = x_i(t_{i+1}; s_i, u_i) \qquad i = 0, 1, \ldots N-1, \qquad (5c)$$

and the consistency conditions

$$0 = g(s_i^x, s_i^z, u_i) \qquad i = 0, 1, \ldots N. \tag{5d}$$

Control and path constraints are imposed pointwise at the multiple shooting nodes

$$h(s_i^x, s_i^z, u_i) \geq 0 \qquad i = 0, 1, \ldots N \tag{5e}$$

as well as at the terminal point

$$r(s_N^x) \left\{ \begin{array}{c} = \\ \geq \end{array} \right\} 0. \tag{5f}$$

The NLP (5a)–(5f) can be summarized as

$$\min_w F(w) \quad \text{subject to} \quad \left\{ \begin{array}{lcl} G(w) & = & 0 \\ H(w) & \geq & 0, \end{array} \right. \tag{6}$$

where $w$ contains all the multiple shooting state variables and controls:

$$w = (s_0^x, s_0^z, u_0, s_1^x, s_1^z, u_1, \ldots, u_{N-1}, s_N^x, s_N^z) \in \mathbb{R}^n.$$

Note that the discretized initial value problem is included in the equality constraints, as

$$G(w) = \left[ \begin{array}{c} s_0^x - \bar{x}_0 \\ s_1^x - x_0(t_1; s_0^x, s_0^z, u_0) \\ g(s_0^x, s_0^z, u_0) \\ \vdots \end{array} \right]. \tag{7}$$

## 2.3 Sequential Quadratic Programming (SQP)

The above NLP problem can be solved by a *sequential quadratic programming (SQP)* method tailored to the multiple shooting structure. Starting from an initial guess $w^0$, a simple SQP method for the solution of (6) iterates

$$w^{k+1} = w^k + \alpha^k \Delta w^k, \quad k = 0, 1, \ldots, \tag{8}$$

where $\alpha^k \in ]0, 1]$ is a relaxation factor to be determined by a line search, and the search direction $\Delta w^k$ is the solution of a quadratic programming (QP) subproblem

$$\min_{\Delta w \in \mathbb{R}^n} \nabla F(w^k)^T \Delta w + \frac{1}{2} \Delta w^T A^k \Delta w \tag{9}$$

subject to

$$\begin{array}{rcl} G(w^k) + \nabla G(w^k)^T \Delta w & = & 0 \\ H(w^k) + \nabla H(w^k)^T \Delta w & \geq & 0. \end{array}$$

Here, $A^k$ denotes an approximation of the Hessian $\nabla_w^2 \mathcal{L}$ of the *Lagrangian function* $\mathcal{L}$,

$$\mathcal{L}(w, \lambda, \mu) = F(w) - \lambda^T G(w) - \mu^T H(w),$$

where $\lambda$ and $\mu$ are the Lagrange multipliers. Several ways exist to obtain the Hessian approximations $A^k$:

- In the exact Hessian SQP method, $A^k := \nabla_w^2 \mathcal{L}(w^k, \lambda^k, \mu^k)$ where multiplier guesses $\lambda^k, \mu^k$ can be obtained from the the solution of the QP (9) at the previous iterate $w^{k-1}$.

- A popular method is to compute the Hessian approximation by an update formula based on the Lagrange gradient differences, which are cheaply available.
- A third approach to obtain the Hessian approximation – the constrained Gauss-Newton method – is recommended in the special case of a least squares type cost function $F(w) = \|C(w)\|_2^2$. The cheaply available matrix

$$A^k := 2\nabla_w C(w^k)\nabla_w C(w^k)^T \tag{10}$$

  provides an excellent approximation of the Hessian, if the residual $\|C(w)\|_2$ of the cost function is small.

## 2.4 Tailoring SQP for Multiple Shooting

Some remarks are in order on how to exploit the multiple shooting structure in the construction of a tailored SQP method. Due to the choice of state and control parametrizations [BP84] the NLP problem and the resulting QP problems have a particular structure: the Lagrangian function $\mathcal{L}$ is *partially separable*, i.e., it can be written in the form

$$\mathcal{L}(w, \lambda, \mu) \;=\; \sum_{i=0}^{N} \mathcal{L}_i(w_i, \lambda, \mu),$$

where $w_i := (s_i^x, s_i^z, u_i)$ are the components of the primal variables $w$ which are effective on interval $[t_i, t_{i+1}]$ only. As a consequence, the Hessian of $\mathcal{L}$ has a *block diagonal structure* with blocks $\nabla_{w_i}^2 \mathcal{L}_i(w_i, \lambda, \mu)$, a fact which can e.g. be exploited by the use of high rank block updates [BP84]. Similarly, the multiple shooting parametrization introduces a characteristic *block sparse structure* of the Jacobian matrices $\nabla G(w)^T$ and $\nabla H(w)^T$.

It is of crucial importance for performance and numerical stability of the direct multiple shooting method that these structures of (5a)–(5f) are fully exploited. We just mention three important points that deserve careful attention:

### 2.4.1 Internal Numerical Differentiation

The solution of the DAE initial value problems (3a)–(3c) and the corresponding derivatives can be computed simultaneously by specially designed integrators which use the principle of *internal numerical differentiation*. In particular, the integrator DAESOL [BFD+97, Bau99], which is based on the backward-differentiation-formulae (BDF), was used in the numerical calculations presented in this paper.

### 2.4.2 Partial Reduction Technique for DAE

Leineweber [Lei99] developed a reduction technique for DAE systems with a large share of algebraic variables, which is also employed for the computations in this paper. He exploits the linearized algebraic consistency conditions (5d) for a reduction in variable space, so that only reduced gradients and Hessian blocks need to be calculated, which correspond to the differential variables $s_i^x$ and controls $u_i$ only [Sch88, BES88]. Note that these reductions are performed only locally, due to the block structure of Hessian and Jacobians.

### 2.4.3 Gauss-Newton Hessian for Tracking Problems

The Gauss-Newton Hessian approximation (10) for least squares cost functions is especially recommended for tracking problems that often occur in NMPC. However, the involved least squares terms in the objective (4) typically arise in integral form:

$$L_i(s_i, u_i) = \int_{t_i}^{t_{i+1}} \|l(x_i(t; s_i, u_i), z(t; s_i, u_i), u_i)\|_2^2 \, dt,$$

and the corresponding Gauss-Newton Hessian blocks $A_i \approx \nabla^2_{w_i} \mathcal{L}_i(w_i, \lambda, \mu)$ have to be computed as integrals

$$A_i(s_i, u_i) = 2 \int_{t_i}^{t_{i+1}} \left( \frac{\partial l(x_i(t; s_i, u_i), z(t; s_i, u_i), u_i)}{\partial (s_i, u_i)} \right)^T \left( \frac{\partial l(x_i(t; s_i, u_i), z(t; s_i, u_i), u_i)}{\partial (s_i, u_i)} \right) dt.$$

An efficient method for computation of these integrals, which delivers the Hessian block approximation $A_i$ at virtually no additional costs during each SQP iteration, has been implemented in the current version of the optimal control package MUSCOD-II [Die02], in conjunction with the implicit DAE solver DAESOL [BBS99].

# 3 Initial Value Embedding

So far we have only described how the direct multiple shooting method can be employed for solution of *one single* optimal control problem (1a)–(1f). In the context of online NMPC, several optimization problems have to be solved consecutively, for different initial values $\bar{x}_0$. We therefore denote the problem (1a)–(1f) resp. its multiple shooting parameterization (5a)–(5f) by $P(\bar{x}_0)$. The question arises of how to find an initial guess $w^0$ for initialization of the SQP iterations towards the solution of each problem $P(\bar{x}_0)$.

Fortunately, we may assume that we know already, from a previous optimization, the solution of a neighbouring optimization problem $P(\bar{x}'_0)$. We will see that this solution is an astonishingly good initial guess for the current problem $P(\bar{x}_0)$, if the transition from one problem to the next fully exploits the advantages of the direct multiple shooting method. But let us first present an initialization approach that seems straightforward, and that we call the "conventional approach".

## 3.1 Conventional Initialization

Given the optimal solution of a previous optimization problem $P(\bar{x}'_0)$, and given the current initial value $\bar{x}_0$, we may use the *previously* optimal control values $u^*_0(\bar{x}'_0), u^*_1(\bar{x}'_0), \ldots$ for a DAE simulation over the complete interval $[t_0, t_0 + T_p]$, starting at the *current* initial value $\bar{x}_0$. This seems to make use of all the knowledge we have of the system.

In the left column of Figure 1 such an initialization is shown for an example problem, that uses the distillation model as in Section 5, and which has a least squares objective that penalizes deviations of the two shown temperature trajectories (first two graphs) from the setpoints 70°C and 88°C. The initialization for the controls (last two graphs) is taken from the solution of problem $P(\bar{x}'_0)$ (cf. left column of Fig. 2).

It can be seen that the simulated temperature trajectories are far from the setpoint values for the conventional initialization – this is not disturbing in principle, because if we knew already the optimal solution, we would not need an optimization routine at all.

For the solution of the tracking problem, we employ the Gauss-Newton approach that was described in Section 2.4.3, in conjunction with a full step SQP method. After two iterations (middle column), the temperature profile has approached the setpoints 70°C and 88°C, but due to nonlinearities of the system this iterate is still far from the optimal solution (right column), which is only attained after 9 SQP iterations (for a BFGS-update instead of the Gauss-Newton approach, even 32 solution iterations would have been necessary).

**Remark:** For unstable systems, the *open-loop* simulation of the system, starting at the initial value $\bar{x}_0$ and using previously optimal controls, as above, may completely fail. This is because the conventional initialization does not build on one of the major features of the direct multiple shooting method, the possibility to allow discontinuities in the system trajectory.

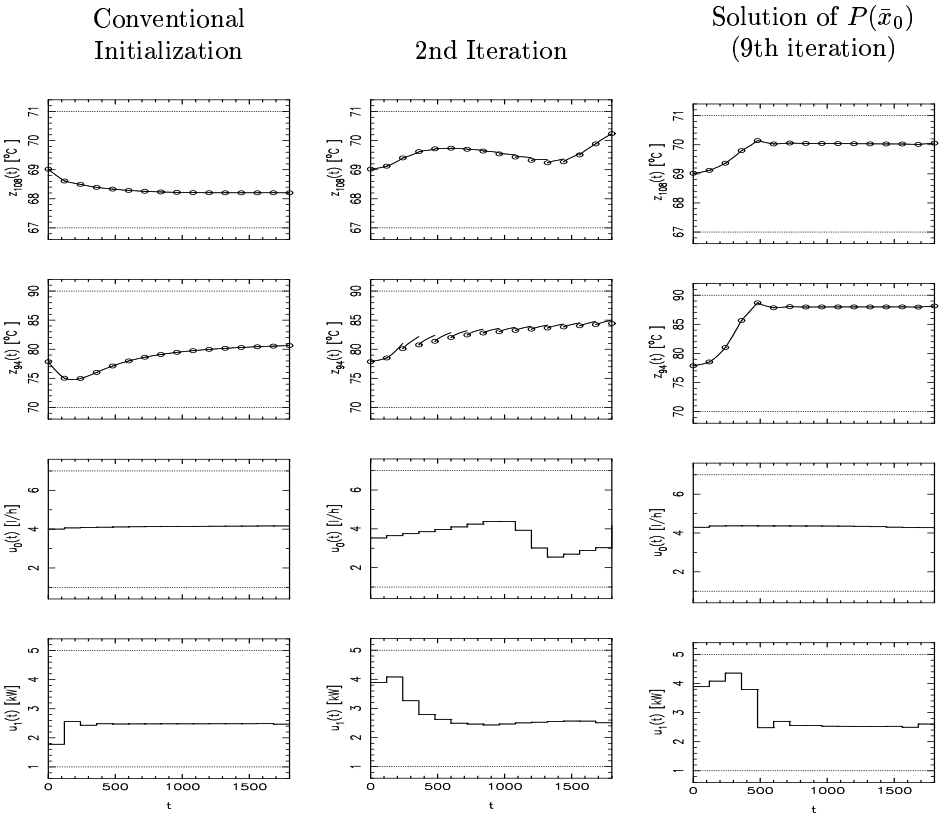


Figure 1: Solution iterations for distillation model, after conventional initialization: two temperatures and two controls, for the initialization (left column), the second iterate (middle column), and the optimal solution (right column). The desired setpoint values are 70°C and 88°C.

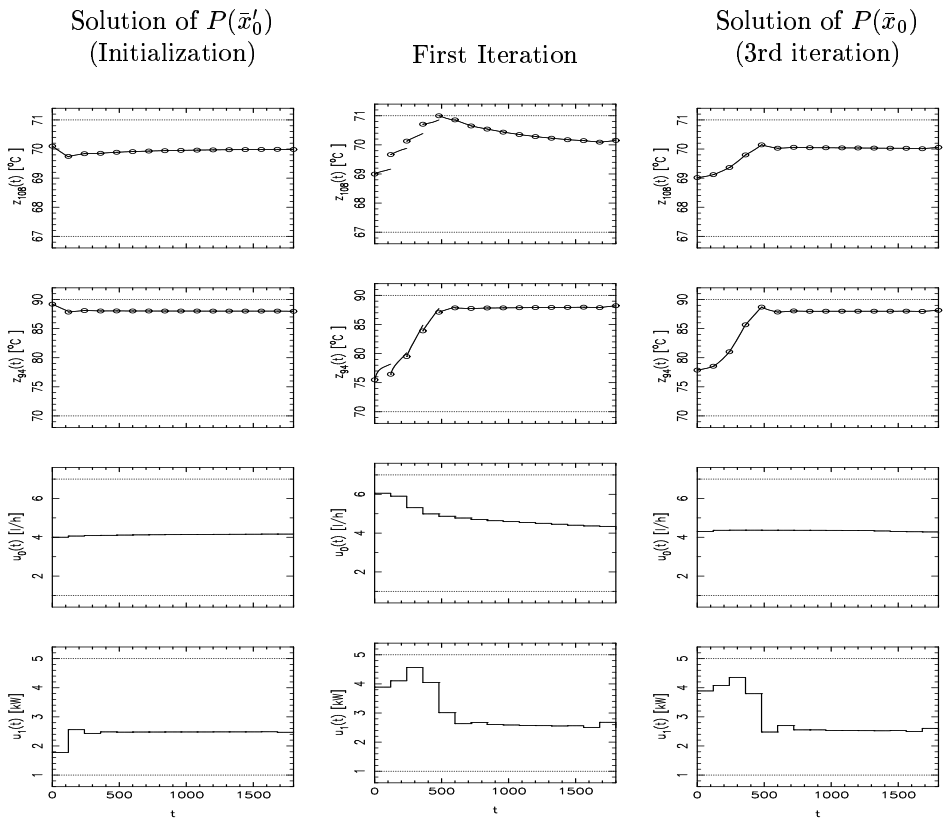


Figure 2: Initial value embedding: after initialization at the solution of $P(\bar{x}_0')$ (left column), the first iteration (middle column) delivers already a good approximation for the exact solution of $P(\bar{x}_0)$ (right column). Cf. Fig. 3. The first two graphs of the first iteration show discontinuous trajectories, as they are typical in the multiple shooting method. This equal distribution of the linearization errors avoids an error accumulation at the end of the horizon and allows the optimizer to stay close to the desired setpoint trajectories during the iterations.

## 3.2 Tangential Predictor by Initial Value Embedding

We will now see that a much better approach to initialize the SQP iterations exists. This approach strongly builds on the advantages of the direct multiple shooting method, and comes quite naturally in its framework.

The main idea is *not* to use the knowledge of $\bar{x}_0$ for the initialization. Instead, we use the solution of the previous problem $P(\bar{x}_0')$ *without any modification*. The corresponding initialization can be seen in the left column of Figure 2.

This simple initialization for the current problem $P(\bar{x}_0)$ results, of course, in a violation of the initial value constraint (5b) in the NLP (5a)–(5f), because $s_0^x = \bar{x}_0' \neq \bar{x}_0$.

However, the constraint is already perfectly satisfied after the first full step SQP iteration, due to its linearity. Moreover, the trajectory of this first iteration (middle column in Figure 2) shows already the main characteristics of the solution (right column), which is attained after 4 iterations.

The formulation of the initial value constraint (5b) in the NLP (5a)–(5f) can be considered a linear embedding of each optimization problem into the manifold of perturbed problems. It is this particular feature of the direct multiple shooting method that allows a very efficient transition from one optimization problem to the next.

A visualization of how the *initial value embedding* works is given in Figure 3. We first plot for each initial value $\bar{x}_0$ the corresponding solution vector $w$ in the NLP variable space (solid line).

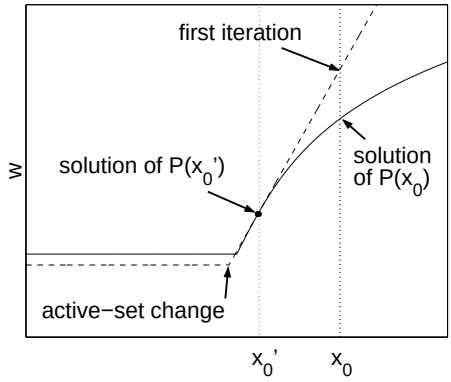


Figure 3: Exact solution manifold (solid line) and tangential predictor after initial value embedding (dashed line), when initialized with the solution of $P(\bar{x}_0')$. The first iteration delivers already a good predictor for the exact solution of $P(\bar{x}_0)$, cf. Fig. 2. This is even true in the presence of active set changes.

This is the exact solution manifold, which is in practice never computed. Note that it usually also has some nondifferentiable points, that occur when the set of active inequality constraints changes. The dashed line shows the tangential predictor that is delivered from the first SQP iteration, when the variables are initialized with the solution of $P(\bar{x}_0')$. This predictor is in particular valid for the current value $\bar{x}_0$ and thus delivers a first iterate $w^1$ that is close to the exact solution $w^*(\bar{x}_0)$ of $P(\bar{x}_0)$. This astonishing property, namely that

$$\|w^1 - w^*(\bar{x}_0)\| = O\left(\| \ x_0' - \bar{x}_0 \ \|^2\right),$$

which can be proven under mild conditions, is true *even at points where the active set changes* [Die02].

The sketched property that the predictor is tangential to the solution manifold is strictly true only for the exact Hessian SQP method; for the Gauss-Newton approach the predictor is still approximately tangential.

**Remark 1:** Note that the initialization by the initial value embedding does require *no* computational effort at all, as the previous solution exists already – this is in contrast to the conventional initialization, where an additional simulation of the DAE system had to be performed. Furthermore, most computations that are necessary for the solution of the first QP (9) can already be executed *before* the current value of $\bar{x}_0$ is actually known [Die02]. This fact will become important in the context of the real-time iteration scheme, because it allows to deliver a first feedback response with negligible delay.

**Remark 2:** Note that for *linear* systems with least squares objective the first iterate would already be the solution, as in this case the multiple shooting NLP (5a)–(5f) is itself nothing else than a QP, and therefore the first QP (9) is already identical to the full NLP (6) and delivers the same solution.

# 4 Real-Time Iterations

Let us now go cosider the full real-time scenario, where we want to solve a sequence of optimization problems $P(\bar{x}_0)$ where $\bar{x}_0$ is the current system state and thus changes continuously with time. We will therefore write $\bar{x}_0(t)$ and $P(\bar{x}_0(t))$ in the following.

## 4.1 The Online Dilemma

Given that the computational time for one SQP iteration is more or less constant, we have to confront the following dilemma: If we want to obtain a quite exact solution for a problem $P(\bar{x}_0(t))$, we have to perform several SQP iterations until a prespecified convergence criterion is satisfied. Assume that we have to perform $n$ iterations, and that each iteration takes a time $\delta$. This means that we obtain the feedback control $\bar{u}(\bar{x}_0(t))$ only at a later time $t + n\delta$, i.e., with a considerable delay. At this time, the system will already be in some system state $\bar{x}_0(t + n\delta) \neq \bar{x}_0(t)$, and $\bar{u}(\bar{x}_0(t))$ is *not* the exact NMPC feedback, $\bar{u}(\bar{x}_0(t + n\delta))$. In the best case the system state has not changed much in the meantime and it is a good approximation of the exact NMPC feedback. The problem of which controls have to be applied in the meantime is still unsolved: a possible choice would be to use previously optimized controls in an open-loop manner. Note that with this approach we can realize an NMPC sampling rate with intervals of length $n\delta$ (under the assumption that *each* problem needs $n$ iterations), and that each feedback comes with a delay of one sampling interval.

We may initialize, of course, subsequent problems with the help of the initial value embedding. Note, however, that due to long sampling times the system state may change considerably from one problem to the next, and that the SQP procedure may therefore still take a relatively high number $n$ of iterates.

## 4.2 Real-Time Iteration Idea

We will now present our answer to the online dilemma. The approach is based on three ideas:

- Due to the above observations, we will never be able to compute the *exact* NMPC feedback control $\bar{u}(\bar{x}_0(t))$ without delay. Therefore, it may be better to compute only an *approximation* $\tilde{u}(\bar{x}_0(t))$ of $\bar{u}(\bar{x}_0(t))$, if this approximation can be computed much faster. Fortunately, such an approximation is delivered by the initial value embedding, as the first iteration was shown to be a tangential predictor for the exact solution. By using just the first iteration, we would already reduce the feedback delay considerably, to the time $\delta$.

- Fortunately, as mentioned in Remark 1 in Section 3.2, the computations for the first iteration can be largely done *before* the initial value $\bar{x}_0(t)$ is known. Therefore, we can even reduce the delay time further, if we perform all these computations before time $t$, and at time $t$ we can quickly compute the feedback response $\tilde{u}(\bar{x}_0(t))$ to the current state, with a delay that is even considerably smaller than $\delta$.

- Given the fact that the exact solution of the optimization problems is no longer required for the feedback computation, we have now to reconsider if it is really necessary to iterate to convergence for each optimization problem. Instead, we can reduce the sampling time considerably, if we perform only one iteration per sampling interval. As a positive side-effect, this shorter sampling time ($\delta$ instead of $n\delta$) most probably leads to smaller differences in subsequent initial states $\bar{x}_0(t)$ and $\bar{x}_0(t + \delta)$ for consecutive optimization problems.

Based on these three ideas, we arrive at the real-time iteration scheme. Now, the focus is shifted from the sequence of optimization problems towards the sequence of iterates: we may regard the SQP procedure iterating uninterrupted, with the only particularity that the system state $\bar{x}_0$ is modified *during* the iterations, and the generation of the feedback controls can be regarded as a by-product of the SQP iterations. Due to the initial value embedding property, the iterates remain close to the exact solution manifold for each new problem. In Figure 4 we have sketched four consecutive real-time iterates, where the dashed lines show the respective tangential predictors.

## 4.3 Real-Time Iteration Algorithm

Let us assume that we start with an initial guess $w^0$ for all multiple shooting variables. We set the iteration index $k$ to zero and perform the following steps:

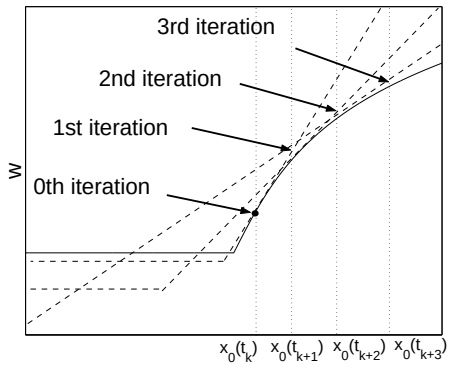


Figure 4: The real-time iterations stay close to the exact solution manifold (solid line), cf. Fig. 3. Each iterate is at the crossing point of the tangential predictor due to the previous iterate (dashed lines) and the current initial value (dotted lines).

1. Preparation: Based on the initial guess $w^k = ({s_0^x}^k, {s_0^z}^k, {u_0}^k, \dots)$, compute nearly all components of the QP (9):

$$\min_{\Delta w \in \mathbb{R}^n} \nabla F(w^k)^T \Delta w + \frac{1}{2} \Delta w^T A(w^k) \Delta w \quad \text{s.t.} \quad \begin{cases} G(w^k) + \nabla G(w^k)^T \Delta w &= 0 \\ H(w^k) + \nabla H(w^k)^T \Delta w &\geq 0. \end{cases}$$

   It is only in the first component of $G(w^k)$, where the initial value $\bar{x}_0$ actually enters, cf. (7); all other components, in particular all derivatives and the Gauss-Newton Hessian $A(w^k)$, can already be computed, which takes already the lion's share of the computations for each SQP iteration. Furthermore, the above QP will be presolved to a large extent, as much as possible without knowledge of $\bar{x}_0$, to prepare the next step (for details we refer to [DBS+02, Die02]).

2. Feedback Response: when the preparation phase has been finished, say at time $t_k$, we observe the real system, and finally obtain the missing initial value $\bar{x}_0(t_k)$. We quickly finish the solution of the QP subproblem to obtain the step vector $\Delta w^k = (\Delta {s_0^x}^k, \Delta {s_0^z}^k, \Delta {u_0}^k, \dots)$ and give the tangential approximation of the NMPC feedback control,

$$\tilde{u}(\bar{x}_0(t_k)) := {u_0}^k + \Delta {u_0}^k,$$

   immediately to the real system.

3. Transition: Set the next initial guess as

$$w^{k+1} := w^k + \Delta w^k,$$

   set $k = k + 1$, and go to 1.

The time for each cycle corresponds to the time of one SQP iteration, $\delta$, i.e., a sampling rate with intervals of length $\delta$ can be realized, and at every sampling instant a new feedback is delivered.

But note that nearly all computations of a cycle are concentrated in the preparation phase 1; therefore, the feedback delay in step 2 is minimized to a value $\delta' \ll \delta$. An overview on the employment of computation time in the real-time iteration scheme is given in Figure 5, together with a sketch of state and control trajectories. Note that the control delay is in practice two orders of magnitude smaller than a sampling time, cf. Fig. 10.

## 4.4 Theoretical Contraction Properties

The contraction properties of the scheme have so far been investigated theoretically for the related class of shrinking horizon problems, and for the nominal system. It could be shown that

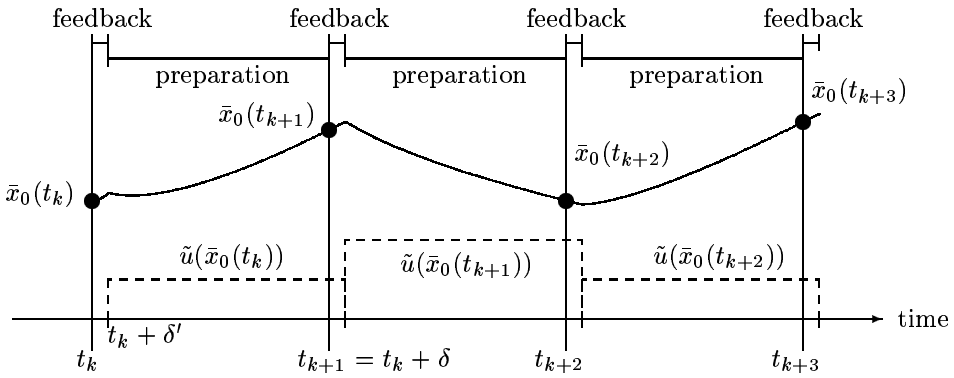


Figure 5: Use of computation time in the real-time iteration scheme; real system state and control trajectory, for sampling time $\delta$ and feedback delay $\delta' \ll \delta$. For computation times in a practical application, see Fig. 10.

the scheme delivers approximations $w^k$ that come closer and closer to the solutions of the corresponding optimization problems, and that the loss of optimality – compared to the exact optimal feedback control – can be bounded [Die02, DBS]. For the moving horizon case, as treated here, the theoretical convergence properties are currently under investigation, and a nominal stability result will be published soon.

# 5 Application to a Distillation Column

As an application example for the proposed real-time iteration scheme we consider the control of a high purity binary distillation column. We have performed a variety of closed-loop experiments at a pilot plant distillation column that is located at the *Institut für Systemdynamik und Regelungstechnik (ISR)* of the University of Stuttgart.

In first numerical tests, the feasibility of the real-time iteration scheme could be shown, with computation times in the range of seconds for a 164th order model [DBS+02], and the practical applicability was confirmed in a first series of closed-loop experiments [DUF+01]; however, the observed closed-loop performance suffered from oscillations that were due to time delays in the real plant, that have *not* been captured by the 164th order distillation model, probably due to the the fact that hydrodynamic effects were ignored.

In this paper, we therefore use a more detailed distillation model that includes hydrodynamics, resulting in a considerably stiffer and larger system model, with 82 differential and 122 algebraic system states. Parts of the presentation follow the lines of [Die02].

## 5.1 The Distillation Column

The distillation column is used for the separation of a binary mixture of Methanol and n-Propanol. It has a diameter of 0.10 m and a height of 7 m and consists of 40 bubble cap trays. The overhead vapor is totally condensed in a water cooled condenser which is open to atmosphere. The reboiler is heated electrically. A flowsheet of the distillation system is shown in Fig. 6. The preheated feed stream enters the column at the feed tray as saturated liquid. It can be switched automatically between two feed tanks in order to introduce well defined disturbances in the feed concentration. In the considered configuration, the process inputs that are available for control purposes are the heat input to the reboiler, $Q$, and the reflux flow rate $L_{\mathrm{vol}}$. Control aim is to maintain high purity specifications for the distillate and bottom product streams $D_{\mathrm{vol}}$ and $B_{\mathrm{vol}}$.

The column is controlled by a distributed control system (DCS), that is used for the lower

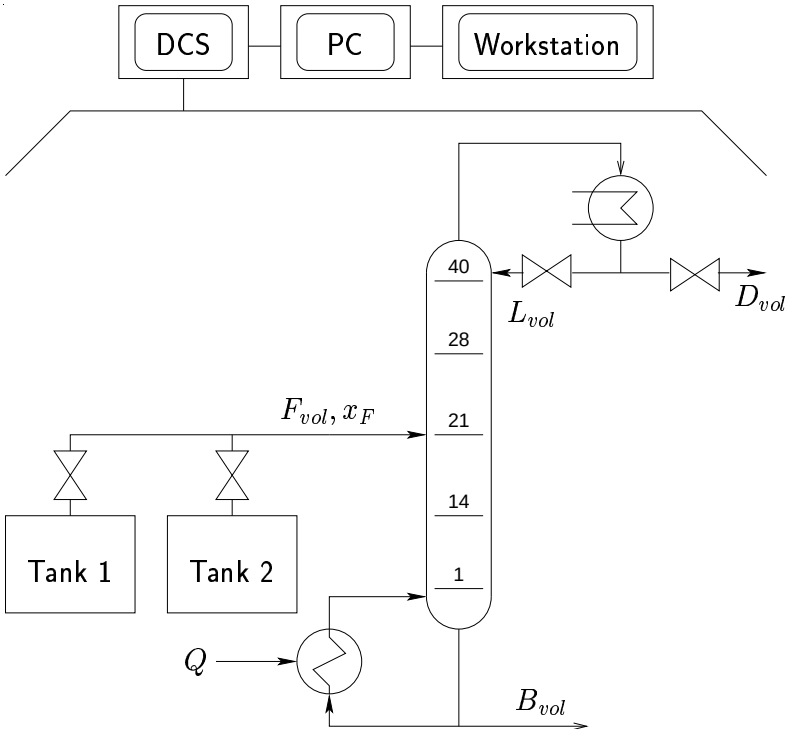


Figure 6: Flowsheet of the distillation column

level control and data acquisition. Basic control loops for the levels, the flow rates, and the heat input are realized on the DCS system. To implement more advanced control schemes the DCS is connected to a PC from and to which direct access from UNIX workstations is possible.

## 5.2 Distillation Model

Depending on the model simplifications different kinds of models can be obtained for the dynamics of the distillation column. For the predictions in the NMPC controller we use a simple equilibrium stage model with hydrodynamics, which is considered to capture the main features of the column dynamics. The employed model is based on the following assumptions:

- total condenser
- negligible vapor holdup
- variable liquid holdup
- liquid outflow determined by Francis weir formula
- constant pressure losses
- perfect mixing
- the mixture is at equilibrium temperature
- Murphree efficiency is applied for each tray

We will in the following only give an overview of the 82 differential and the 122 algebraic state variables, but do not present the differential and algebraic equations. Details can be found in [Die02].

We refer to the $N = 40$ trays by $\ell = 1, 2, \ldots, N$, counting from the bottom to the top, with $\ell = N_F = 20$ being the feed tray. For notational convenience, let us refer with $\ell = 0$ to the reboiler and to the condenser by $\ell = N + 1$.

The (molar) Methanol concentrations in reboiler, on the 40 trays, and in the condenser $X_\ell$ for $\ell = 0, 1, \ldots, N + 1$ are the first 42 components of the differential state vector $x$, and the molar tray holdups $n_\ell$ for $\ell = 1, \ldots, N$, which may strongly vary during operation, are the second 40 components.

The (molar) liquid and vapor fluxes $L_\ell$ and $V_\ell$ ($\ell = 1, 2, \ldots, N$) out of the 40 trays as well as the 42 temperatures $T_\ell$ ($\ell = 0, 1, 2, \ldots, N + 1$) of reboiler, trays and condenser form the 122 components of the algebraic state vector $z = (L_1, \ldots, L_N, V_1, \ldots, V_N, T_0, \ldots, T_{N+1})^T$. Note that many algebraic variables that can easily be eliminated in the given model formulation – as e.g. volume holdups, n-Propanol concentrations, enthalpies, equilibrium pressures, etc. – need not be taken as components of the algebraic state vector $z$ in this formulation.

The two components of the control vector $u = (L_{\text{vol}}, Q)^T$ are the volumetric reflux flow $L_{\text{vol}}$, and the heat input $Q$, that determine implicitly the molar fluxes out of condenser and out of the reboiler. The resulting DAE system has index one and can be brought into the form (1c)–(1d).

### 5.2.1 Determination of System Parameters

The employed values for a variety of constant system parameters – e.g. tray holdups, Murphree coefficients, pressure losses – have been estimated by fitting them to a series of steady state and and open-loop dynamic experiments. To obtain measurements of the dynamic behaviour of the column, step changes in the feed rate $F_{\text{vol}}$ and composition $X_{\text{F}}$, the reflux rate $L_{\text{vol}}$, and heat input $Q$ were performed. Measurements of *all* temperatures $T_0, \ldots, T_{N+1}$ were taken for least squares fitting of the simulated to the observed behaviour. The fitting was done with help of the off-line version of MUSCOD-II.

### 5.2.2 Treatment of Disturbances in Feedstream

Though the the feed flow $F_{\text{vol}}$ and feed concentration $X_{\text{F}}$ are inputs to the system, we augment the differential state vector by these two components in the model that is finally used for the optimization. We simply add the two trivial differential equations $\dot{F}_{\text{vol}} = 0$ and $\dot{X}_{\text{F}} = 0$. This formulation, that nominally leads to 84 differential states, takes account of the fact that we will treat these two parameters as constant disturbances. In the prediction, it is assumed that they will not change. However, the estimator may observe if they change, and the formulation conveniently allows to react to changes by the initial value embedding.

## 5.3 Optimal Control Problem Formulation

Control aim is to maintain the specifications on the product purities $X_0$ and $X_{N+1}$ in reboiler and condenser despite disturbances. As usual in distillation control, the concentrations $X_0$ and $X_{N+1}$ are not controlled directly – instead, an inferential control scheme which controls the deviation of the concentrations on tray 14 and 28 from a given setpoint is used. These two concentrations are much more sensitive to changes in the inputs of the system than the product concentrations; if they are kept constant, the product purities are safely maintained for a large range of process conditions. As concentrations are difficult to measure, we consider instead the tray *temperatures*, which correspond directly to the concentrations via the Antoine equation. In the following we will use the projection $\tilde{T}z := (T_{28}, T_{14})^T$ to extract the controlled temperatures from the vector $z$, and define $\tilde{T}_{\text{ref}} := (70\,^\circ\text{C},\ 88\,^\circ\text{C})^T$ for the desired setpoints. Let us denote the setpoint of the controls by $u_S$.

The open-loop objective is formulated as the integral of a least squares term

$$L(z, u, u_S) := \|\tilde{T}z - \tilde{T}_{\text{ref}}\|_2^2 \quad + \quad \|R(u - u_S)\|_2^2\,, \tag{11}$$

where the second term is introduced for regularization, with a small diagonal weighting matrix

$$R = \mathrm{diag}(0.05\,^\circ\mathrm{C\,h\,l^{-1}}, 0.05\,^\circ\mathrm{C\,kW^{-1}}).$$

### 5.3.1 Optimal Control Problem

Slightly deviating from the problem formulation in Section 1, we divide the prediction horizon into a control horizon $[t_0, t_0 + T_c]$ and a prediction interval $[t_0 + T_c, t_0 + T_p]$ on which the controls are fixed to the setpoint values $u_S$. The objective contribution of this prediction interval provides an upper bound of the neglected future costs that are due after the end of the control horizon, if its length is sufficiently large. A length of $T_p - T_c = 36000$ seconds has been considered to be sufficient in all performed experiments.

The resulting optimal control problem is formulated as follows:

$$\min_{u(\cdot),x(\cdot),u_S} \quad \int_{t_0}^{t_0+T_p} \left\{ \left\| \tilde{T} z(t) - \tilde{T}_{\mathrm{ref}} \right\|_2^2 \quad + \quad \|R(u(t) - u_S)\|_2^2 \right\} \, dt \tag{12}$$

subject to the model DAE

$$\begin{aligned} B(\cdot)\,\dot{x}(t) &= f(x(t), z(t), u(t)) \\ 0 &= g(x(t), z(t), u(t)) \end{aligned} \qquad \text{for } t \in [t_0, t_0 + T_p].$$

Initial values for the differential states are prescribed:

$$x(t_0) \quad = \quad \bar{x}_0.$$

State and control inequality constraints are formulated by

$$h(x(t), z(t), u(t)) \geq 0 \quad t \in [t_0, t_0 + T_p],$$

where

$$h(x, z, u) := \begin{bmatrix} D(x, z, u) - D_{\min} \\ B(x, z, u) - B_{\min} \\ u - u_{\min} \\ u_{\max} - u \end{bmatrix}$$

formulates lower bounds for the fluxes $D(x, z, u)$ and $B(x, z, u)$ out of condenser and reboiler (determined according to the model assumptions) which should always maintain small positive values, and lower und upper bounds for the controls.

The steady state control $u_S$ is determined implicitly by the requirements that $u$ is constant on the long prediction interval

$$u(t) = u_S \quad \text{for } t \in [t_0 + T_c, t_0 + T_p],$$

and by the final state constraint

$$\tilde{T} z(t_0 + T_p) - \tilde{T}_{\mathrm{ref}} = 0.$$

For an alternative formulation of the steady state condition, see [Die02].

## 5.4 Numerical Realization

The length $T_c$ of the control horizon and the control discretization have to be chosen such that the computation time for one real-time iteration does not exceed the relevant time scale of the system or of the disturbances. Based on numerical experiments on the available computer (AMD Athlon processor with 1009 MHz) and on the requirement that one real-time iteration should not exceed 20 seconds, we found that $T_c$=600 seconds with 5 control intervals each of 120 seconds length is a good choice. Note that the control interval length of 120 seconds is 6 times longer than the desired sampling time of 20 seconds – but this does not pose any difficulty.

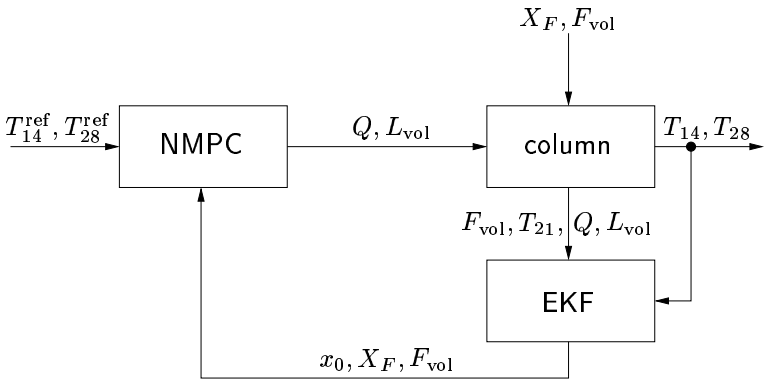


Figure 7: Closed-loop NMPC setup

### 5.5 On-Line State Estimation

To obtain an estimate of the 82 differential system states and of the model parameter $X_F$ by measurements of the three temperatures $T_{14}$, $T_{21}$, $T_{28}$ and the feedflow $F_{\text{vol}}$ only, we have implemented a variant of an Extended Kalman Filter (EKF).

In contrast to an ordinary EKF our estimator can incorporate additional knowledge about the possible range of states and parameters in form of bounds. This is especially useful as the tray concentrations need to be constrained to be in the interval $[0, 1]$ from physical reasoning. The algorithm is described in [Die02]. A comparison of estimated and measured temperature profiles can be found in Fig. 9 – note that only the temperatures $T_{14}$, $T_{21}$ and $T_{28}$ are available to the state estimator.

### 5.6 Coupling with the Process Control System

As mentioned above, the distillation column is controlled by a lower level distributed control system (DCS), which is connected to a PC (cf. Fig. 6). Access to this PC from UNIX workstations is possible via ftp, so that all higher level algorithms, in particular the state estimator and the real-time iteration scheme, could be implemented on a powerful LINUX workstation with an AMD Athlon processor (1009 MHz). With the given equipment it was only possible to obtain measurements and to write the computed control inputs to the DCS every 10 seconds, i.e., a sampling time of 10 seconds was used for the state estimator.

The three processes – data acquisition, state estimation and real-time optimization – were running independently and communicating only via input and output files, in such a way that a breakdown of one component did not cause an immediate breakdown of the others. Missing new inputs were automatically replaced by old values. This construction made the whole system sufficiently stable against variations in computation and data transfer times.

Figure 7 shows the overall controller/plant/estimator setup.

## 6 Experimental Results

We have tested the NMPC scheme on various scenarios [Die02]. Here, we will present only two scenarios, a step change in the feed flow rate $F_{\text{vol}}$, and a large disturbance scenario where the column was driven with too low reflux flow for over ten minutes.

### 6.1 Feed Flow Change

The closed loop response of the scheme to a step change in $F_{\text{vol}}$ is shown in Fig. 8 on the left column, where it is also compared with the performance of an existing PI controller (right column).

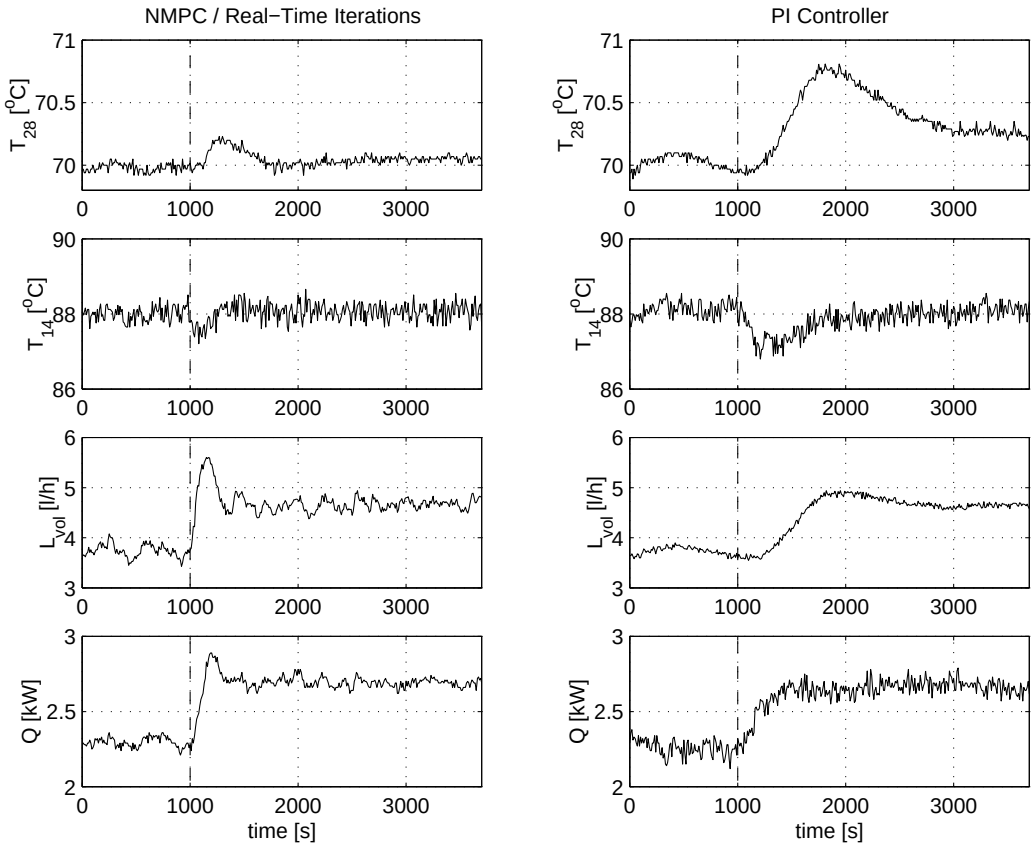


Figure 8: Feed flow change: Comparison of real-time iteration NMPC with a conventional PI controller, for a feed flow step change by 20 %

The PI control scheme, which is usually employed to control the column, consists of two decoupled single-input/single-output PI loops, one of which uses the heat input $Q$ to control the temperature $T_{14}$, whereas the other uses the reflux $L_{\text{vol}}$ to control the temperature $T_{28}$.

The Figure shows the controlled outputs $T_{28}$ and $T_{14}$ and the input responses $L_{\text{vol}}$ and $Q$. Starting from a steady state, we increase the feedflow at time $t = 1000$ seconds by 20 %. The NMPC controller completes the transition into the new steady state 1000 seconds after the feed flow change, with a maximum deviation in $T_{28}$ of 0.3°C. This compares well with the PI performance, which has a maximum deviation of 0.8°C, and which did not even complete the transition to the new steady state 3500 seconds after the step change.

## 6.2 Large Disturbance Scenario

To have larger disturbance effects, we consider the following scenario: starting with a steady state for an increased feed flow rate (by 20 %), we reduce at time $t = 700$ seconds simultaneously the feedflow (back to its nominal value) and the reflux, from $L_{\text{vol}} = 5.3\ \frac{\text{l}}{\text{h}}$ down to $L_{\text{vol}} = 2\ \frac{\text{l}}{\text{h}}$, while maintaining the heating power constant at its (high) value $Q = 2.9$ kW. These inputs, that are maintained constant for 800 seconds, heat the column up and move the temperature profile far away from the nominal operating conditions, as can be seen in the right hand side of Fig. 9, where the distorted temperature profile at time $t = 1500$ is shown.

Only at this time the NMPC feedback is switched on. The closed-loop response can be seen on the left hand side in Fig. 9. While $Q$ jumps immediately down to its minimum value of 1.5kW,

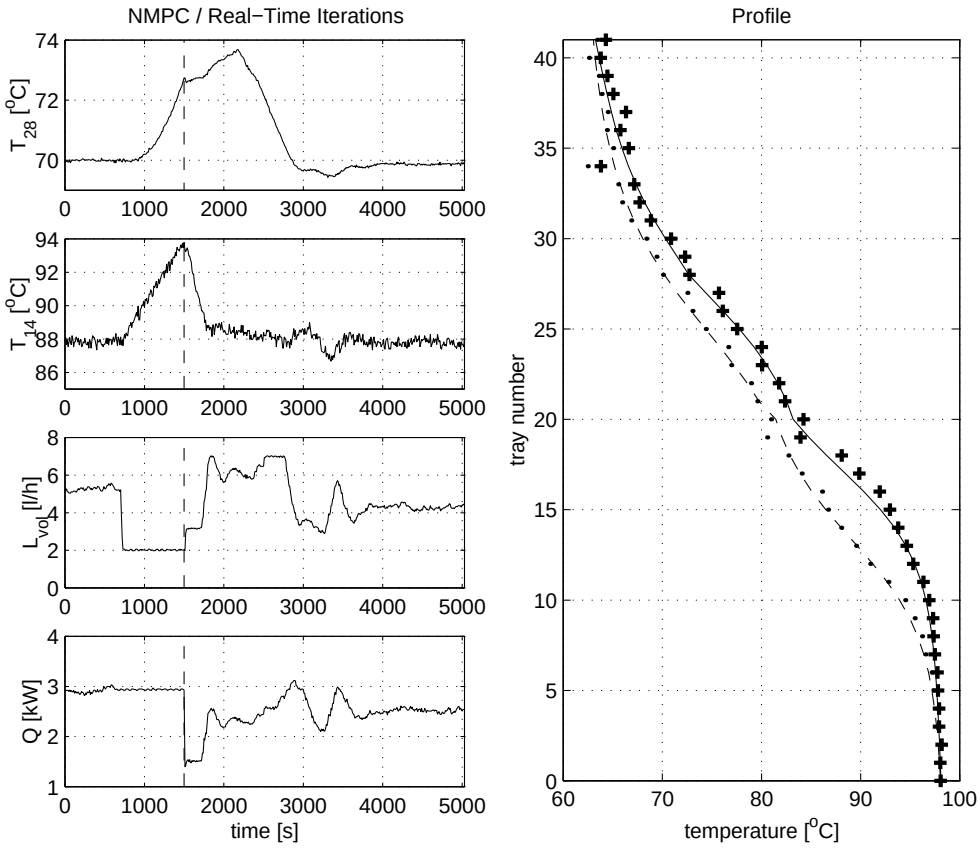


Figure 9: Large disturbance scenario. Left: Closed-loop response. Feedback starts only at time $t = 1500$ seconds. The temperature profile at this time is shown on the right hand side (+), together with the estimated profile (solid) and compared to the desired steady state profile (dots/dashed).

$L_{\text{vol}}$ is *not* increased to its maximum value, as would from first sight be the best thing to cool the column. However, this would have resulted in valve saturation; it was the path constraint $D \geq D_{\min}$ that caused this interesting feature of the closed-loop behaviour.

The scenario could not be compared with the PI controller, as the PI controller started strong oscillations due to valve saturation.

**Observed Computation Times:** Let us have a look on the computation times under experimental conditions. We have measured both, the preparation time and the feedback response time for each real-time iteration. Both time measurements were done externally (from a MATLAB environment), i.e., they are not CPU times in the strict sense, but the overall times that the computations required under the given CPU load conditions. The observed times can be seen in Fig. 10.

Note that due to the fact that the communication sampling rate was technically restricted to be not shorter than 10 seconds, the immediate response may in our realization have taken up to 10 seconds until it arrives at the distillation column, depending on the phase difference of the (self-synchronizing) optimizer and the data transfer system.

**Optimal Solution:** It is interesting to compare the experimental closed-loop trajectory with a theoretical off-line result, namely with the optimal open-loop trajectory, that is shown on the right column of Fig. 11. It can be seen that the experimental closed-loop trajectories (shown again in

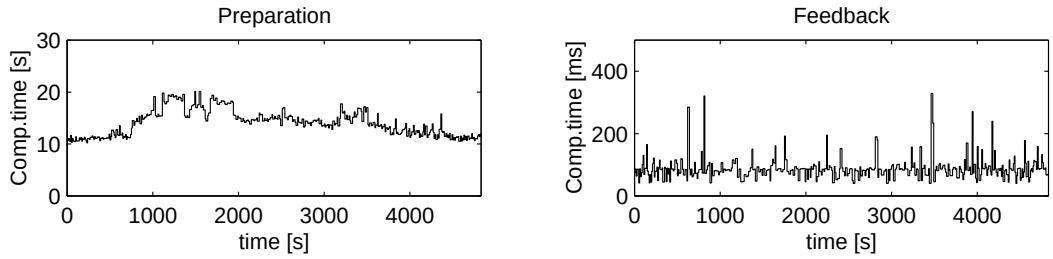


Figure 10: Computation times for preparation phase and for feedback phase, during the large disturbance experiment (cf. Fig. 9). Note that the graphs have different scales.

the left column) show considerable similarity with the theoretically optimal solution.

The computation of the optimal trajectory with the off-line multiple shooting method required 23 major SQP iterations with a CPU time of 3356 seconds (AMD Athlon processor with 1009 MHz), where the control horizon was chosen to consist of 45 multiple shooting intervals, each of 30 seconds length. Note that the computation time for this problem is in the same order as the whole process duration.

### 6.3 Discussion

We have seen that the proposed real-time iteration NMPC control scheme is not only feasible for a practical large scale application, but that it results even in a good closed-loop performance. The application of NMPC did not require much tuning, and a standard distillation model could be used for control; the most time consuming step in the controller setup is, however, parameter estimation and model validation.

## 7 Conclusions

We have presented a new numerical approach to optimization in NMPC, the *real-time iteration* scheme. The scheme is based on the direct multiple shooting method and employs an *initial value embedding* for optimal transition from one optimization problem to the next. The algorithm is implemented within the dynamic optimization package MUSCOD-II and offers the following practical advantages in the context of online process optimization:

- The DAE model equations can either be provided in the gPROMS modelling language [Pro00, LSBS02] or as generic C or Fortran-Code.
- Efficient state-of-the-art DAE solvers (e.g. DDASAC [CS85], DAESOL [Bau99]) are employed to calculate the system trajectories and derivatives quickly and accurately.
- The direct multiple shooting method, as a simultaneous strategy, allows to efficiently treat highly nonlinear and unstable systems [DBS].
- The approach allows a robust treatment of control and path constraints as well as boundary conditions.
- The method is well suited for parallel computation and allows considerable speedups, since the integrations are decoupled on different multiple shooting intervals [LSBS02]. An existing parallel implementation is based on the portable MPI standard.

The algorithm was applied for the NMPC of a binary distillation column, using a stiff DAE model with 82 differential and 122 algebraic state variables for the optimization. Experimental results

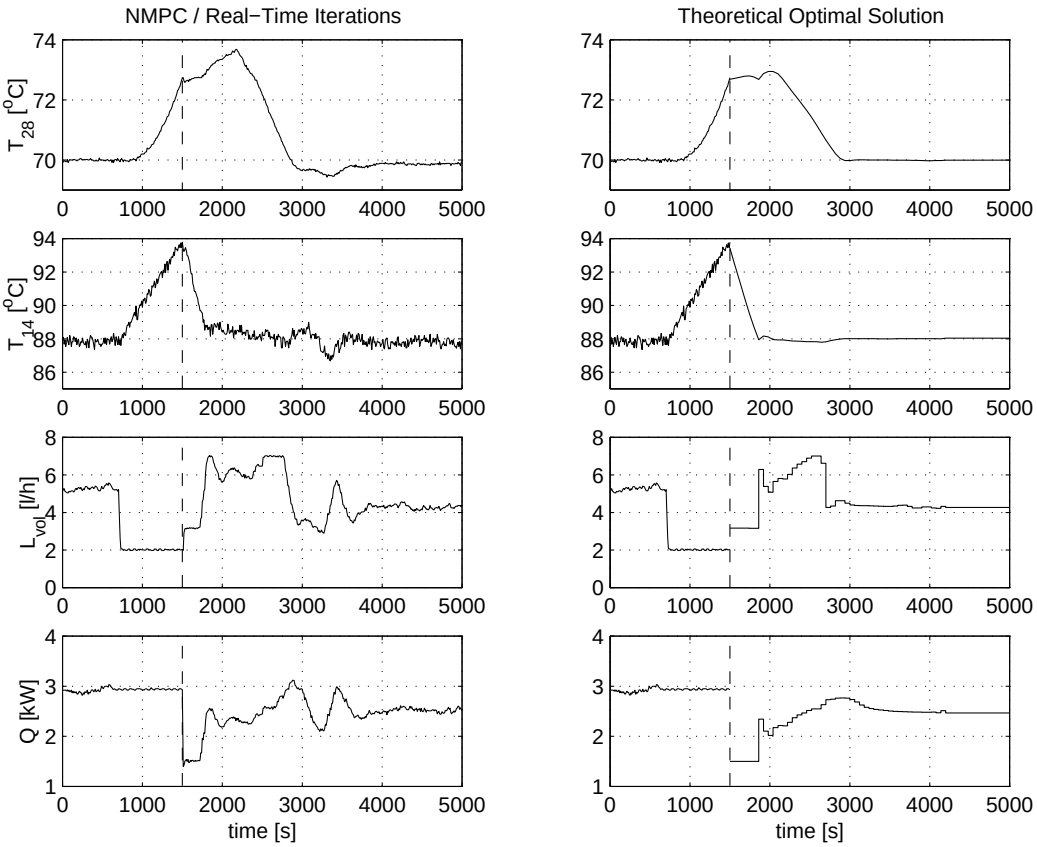


Figure 11: Large disturbance scenario. Left: Closed-loop response. Right: Theoretically optimal solution.

have been obtained at a pilot-plant distillation column situated at the *Institut für Systemdynamik und Regelungstechnik* at the University of Stuttgart. Two scenarios are presented: first, the optimal transition of the distillation column into a new steady state after a sudden feed flow change of 20%. The optimized transition only takes 15 minutes, which compares well with the 1-2 hours suggested by previous experience at the plant. In a second test, the column was on purpose driven with too low reflux flow for over ten minutes, leading to a largely disturbed system state, before the NMPC optimizer was allowed to control the plant again: in the course of half an hour, the column was safely guided back into steady state, without violation of constraints.

The experimental study demonstrates that NMPC is a feasible control alternative; it does not need much tuning and can directly employ nonlinear first principle models. Due to the computational efficiency of the real-time iteration scheme, even large scale models can be treated and need not be simplified for use in online control – they can be used in exactly the same form in which they are delivered from the modeller.

## Acknowledgements

Financial support by the *Deutsche Forschungsgemeinschaft (DFG)* within the *DFG-Schwerpunktprogramm* "Online Optimization of Large Scale Systems" is gratefully acknowledged.

## References

[Bau99] I. Bauer. *Numerische Verfahren zur Lösung von Anfangswertaufgaben und zur Generierung von ersten und zweiten Ableitungen mit Anwendungen bei Optimierungsaufgaben in Chemie und Verfahrenstechnik.* PhD thesis, University of Heidelberg, 1999. Download at: http://www.ub.uni-heidelberg.de/archiv/1513.

[BBB+01] T. Binder, L. Blank, H. G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. P. Schlöder, and O. v. Stryk. Introduction to model based optimization of chemical processes on moving horizons. In M. Grötschel, S.O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pages 295–340. Springer, 2001. download at: http://www.zib.de/dfg-echtzeit/Publikationen/Preprints/Preprint-01-15.html.

[BBS99] I. Bauer, H. G. Bock, and J. P. Schlöder. DAESOL – a BDF-code for the numerical solution of differential algebraic equations. Internal report, IWR, SFB 359, University of Heidelberg, 1999.

[BES88] H. G. Bock, E. Eich, and J. P. Schlöder. Numerical solution of constrained least squares boundary value problems in differential-algebraic equations. In K. Strehmel, editor, *Numerical Treatment of Differential Equations.* Teubner, Leipzig, 1988.

[BFD+97] I. Bauer, F. Finocchi, W. J. Duschl, H.-P. Gail, and J. P. Schlöder. Simulation of chemical reactions and dust destruction in protoplanetary accretion discs. *Astron. Astrophys.*, 317:273–289, 1997.

[Bie00] L. T. Biegler. Efficient solution of dynamic optimization and NMPC problems. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, volume 26 of *Progress in Systems Theory*, pages 219–244, Basel, 2000. Birkhäuser.

[BP84] H. G. Bock and K. J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proc. 9th IFAC World Congress Budapest.* Pergamon Press, 1984.

[BR91] L. T. Biegler and J. B. Rawlings. Optimization approaches to nonlinear model predictive control. In Y. Arkun and W. H. Ray, editors, *Chemical Process Control – CPC IV*, pages 543–571, Austin, Texas, 1991. The CACHE Corp.

[CA98] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1218, 1998.

[Che97] H. Chen. *Stability and Robustness Considerations in Nonlinear Model Predictive Control.* Fortschr.-Ber. VDI Reihe 8 Nr. 674. VDI Verlag, Düsseldorf, 1997.

[CS85] M. Caracotsios and W. E. Stewart. Sensitivity analysis of initial value problems with mixed odes and algebraic equations. *Comput. Chem. Engng.*, 9:359–365, 1985.

[DBS] M. Diehl, H. G. Bock, and J. P. Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. (submitted).

[DBS+02] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by large-scale DAE. *J. Proc. Contr.*, 2002. (in print).

[Die02] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes*, volume 920 of *Fortschr.-Ber. VDI Reihe 8, Meß, Steuerungs- und Regelungstechnik.* VDI Verlag, Düsseldorf, 2002. Download also at: http://www.ub.uni-heidelberg.de/archiv/1659/.

[DLS01] Moritz Diehl, Daniel B. Leineweber, and Andreas A. S. Schäfer. MUSCOD-II Users' Manual. IWR-Preprint 2001-25, University of Heidelberg, 2001.

[DMS00] G. De Nicolao, L. Magni, and R. Scattolini. Stability and robustness of nonlinear receding horizon control. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, volume 26 of *Progress in Systems Theory*, pages 3–23, Basel, 2000. Birkhäuser.

[DUF+01] M. Diehl, I. Uslu, R. Findeisen, S. Schwarzkopf, F. Allgöwer, H. G. Bock, T. Bürner, E. D. Gilles, A. Kienle, J. P. Schlöder, and E. Stein. Real-time optimization for large scale processes: Nonlinear model predictive control of a high purity distillation column. In M. Grötschel, S.O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pages 363–384. Springer, 2001. download at: http://www.zib.de/dfg-echtzeit/Publikationen/Preprints/Preprint-01-16.html.

[GPM89] C. E. García, D. M. Prett, and M. Morari. Model predictive control: Theory and practice – a survey. *Automatica*, 25:335ff, 1989.

[HAM98] A. Helbig, O. Abel, and W. Marquardt. Model predictive control for on-line optimization of semi-batch reactors. In *Proc. Amer. Contr. Conf.*, pages 1695–1699, Philadelphia, 1998.

[LBBS02] D. B. Leineweber, I. Bauer, H. G. Bock, and J. P. Schlöder. An efficient multiple shooting based reduced sqp strategy for large-scale dynamic process optimization. part i: Theoretical aspects. *Comp. & Chem. Eng.*, 2002. (submitted).

[Lei99] 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.

[LSBS02] D. B. Leineweber, A. Schäfer, H. G. Bock, and J. P. Schlöder. An efficient multiple shooting based reduced sqp strategy for large-scale dynamic process optimization. part ii: Software aspects and applications. *Comp. & Chem. Eng.*, 2002. (submitted).

[May96] D.Q. Mayne. Nonlinear model predictive control: An assessment. In J.C. Kantor, C.E. Garcia, and B. Carnahan, editors, *Fifth International Conference on Chemical Process Control – CPC V*, pages 217–231. American Institute of Chemical Engineers, 1996.

[May00] D.Q. Mayne. Nonlinear model predictive control: Challenges and opportunities. In F. Allgöwer and A. Zheng, editors, *Nonlinear Predictive Control*, volume 26 of *Progress in Systems Theory*, pages 23–44, Basel, 2000. Birkhäuser.

[PB93] C.C. Pantelides and P.I. Barton. Equation oriented dynamic simulation: Current status and future perspectives. *Comp. Chem. Eng.*, 17S:S263–S285, 1993.

[Pro00] Process Systems Enterprise Ltd., London. *gPROMS Advanced User's Manual*, 2000.

[RBP+99] R. Ross, V. Bansal, J.D. Perkins, E.N. Pistikopoulos, J.M.G. van Schijndel, and G.L.M. Koot. Optimal design and control of an industrial distillation system. *Comput. Chem. Eng.*, 23(SS):S875–S878, 1999.

[SBS98] V. H. Schulz, H. G. Bock, and M. C. Steinbach. Exploiting invariants in the numerical solution of multipoint boundary value problems for daes. *SIAM J. Sci. Comp.*, 19:440–467, 1998.

[Sch88] J. P. Schlöder. *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung*, volume 187 of *Bonner Mathematische Schriften*. University of Bonn, Bonn, 1988.

[SOB95] L. O. Santos, N. M.C de Oliveira, and L. T. Biegler. Reliable and efficient optimization strategies for nonlinear model predictive control. In *Proc. Fourth IFAC Symposium DYCORD+'95*, pages 33–38, Oxford, 1995. Elsevier Science.

[Sor99] E. Sorensen. A cylic operating policy for batch distillation - theory and practice. *Comput. Chem. Eng.*, 23 (4-5):533–542, 1999.