AIChE

Improvement of State Profile Accuracy in Nonlinear Dynamic Optimization with the Quasi-Sequential Approach

Martin Bartl and Pu Li
Institute for Automation and Systems Engineering, Simulation and Optimal Processes Group, Ilmenau University of Technology, 98684 Ilmenau, Germany

Lorenz T. Biegler
Dept. of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213

DOI 10.1002/aic.12437
Published online January 7, 2011 in Wiley Online Library (wileyonlinelibrary.com).

Quasi-sequential methods are efficient and flexible strategies for the solution of dynamic optimization problems. At the heart of these strategies lies the time discretization and approximation of dynamic systems for nonlinear optimization problems. To address this question, we employ a time derivative analysis within the quasi-sequential approach and derive a finite element placement strategy. In addition, methods for direct error prediction are applied to this approach and extended with a proposed time derivative analysis. According to the information for current time derivatives, subintervals are introduced that improve accuracy of state profiles. Since this is only done in the simulation layer, the nonlinear programing solver need not be restarted. An efficient gradient computation is also derived for these subintervals; the resulting enhanced accuracy accelerates convergence performance and increases the robustness of the solution to initialization. A beer fermentation process case study is presented to demonstrate the effectiveness of the proposed approach. © 2011 American Institute of Chemical Engineers *AIChE J,* 57: 2185–2197, 2011
Keywords: accuracy of state profile approximation, time optimal control, optimal switching behavior, moving finite elements, orthogonal collocation

Introduction

Many tasks in the process industry for optimal state transitions (e.g., batch processes, startup and shutdown phases, and product changeovers) lead to dynamic optimization problems. Solving such problems efficiently represents an essential topic of current research. The approaches to dynamic optimization problems can be classified into indirect and direct methods.[1] In the indirect methods (also called variational approach) the solution will be derived based on the necessary optimality conditions, leading to a two-point-boundary-value problem. In the direct methods, the infinite dimensional optimization problem is reformulated with a discretization strategy into a finite dimensional optimization problem, so that it can be solved with a nonlinear programing (NLP) solver. Direct methods can be further classified into three approaches: the sequential, simultaneous, and quasi-sequential approach.[2] Due to their capability to treat large-scale and complex systems, direct methods are becoming more widely considered in different industrial disciplines.

In the sequential approach, where only the controls are discretized (or parameterized), the optimization problem is

Correspondence concerning this article should be addressed to P. Li at pu.li@tu-ilmenau.de.

divided into a simulation and an optimization layer. Given initial conditions for state variables and a set of control parameters, the model equations, usually composed of a differential algebraic equation (DAE) system, will be solved in the simulation layer with a DAE solver. Only the control parameters are handled as variables in the optimization layer, where an NLP solver is used. A well-known disadvantage of the sequential approach is the difficulty in treating path constraints on state variables.

The quasi-sequential approach from Hong et al.,[3] has been applied to many dynamic optimization problems from an industrial point of view,[4–6] illustrating its capability of solving large-scale complex problems. As in the simultaneous approach[2,7–9] both control and state variables are discretized using collocation on finite elements.[10,11] In this way path, constraints on state variables can be held in each element. In contrast to the simultaneous approach, the quasi-sequential approach solves the algebraic equations, resulting from discretization of the DAEs, in a simulation layer. This step reduces the size of the optimization problem by eliminating equality constraints and states. The resulting NLP problem consists only of inequality constraints and controls and, in this way, performance of the line search can be significantly enhanced.[3] However, in comparison to the simultaneous approach more computational effort is required in the quasi-sequential approach to solve model equations at each iterate of NLP. Hence, if the solution of this equation system converges quickly, the quasi-sequential approach can be more efficient.

Both the simultaneous and the quasi-sequential approach use the collocation method to discretize the dynamic optimization problem, resulting in an approximate or inaccurate description of the original system. In most previous investigations constant time intervals (so-called finite elements) for the discretization of the underlying DAE system have been used. With fixed finite elements, the determination of discontinuous points in the control profile (i.e., the switching behavior) cannot be properly identified. A number of investigations to enhance the computational accuracy of the simultaneous approach and the detection of switching behavior have been made.[2] Similar to the work in Tieu et al.,[12] Huntington and Rao[9] analyzed the accuracy by experimentally modifying the number of elements and collocation points. Cuthrell and Biegler[13] formulated an NLP, where element lengths are adjusted by the NLP solver to detect discontinuities in optimal profiles, based on the error equidistribution strategy from De Boor.[14,15] This work was extended by Vasantharajan and Biegler[16] where two methods were proposed to handle the accuracy of state profiles. First, the objective function is modified to achieve an equal distribution of the approximation error. Second and most importantly, additional constraints are added in each finite element to maintain the approximation error within a user-defined tolerance. In the second case, especially when there are insufficient elements to satisfy the error tolerance, an adaptive element addition strategy was proposed in which additional elements (so called ''dummy'' elements) are included at knot locations in an existing partition. This direct error enforcement criterion from Vasantharajan and Biegler[16] was successfully applied to biological processes.[17] However, adding additional dummy elements and nonlinear error constraints increases the dimension of the optimization problem. In this way, the optimization problem is essentially restarted.

An extension of the element placement was proposed by Renfro et al.[18] and by Tanartkit and Biegler[19,20] by considering the information derived from optimal control theory and the accuracy of the approximation. In the latter studies, the equidistribution strategy is replaced by more straightforward direct error enforcement. The simultaneous handling of the discretized optimization problem, the element placement and the nonlinear error constraints leads to a nonlinear problem that can be very difficult to solve (as mentioned in Vasantharajan and Biegler).[16,19] Therefore, a bilevel optimization framework is adopted in Tanartkit and Biegler[19,20]: the outer problem determines the element placement and the inner one performs the dynamic optimization. Here it is important to note that for each iterate of the outer problem the inner problem must be solved. In addition, because the mesh and active sets change in the inner problem, the outer problem is nonsmooth; consequently cutting planes need to be added to ensure descent directions in the outer problem. In Biegler et. al.,[21] an heuristic approach was developed to handle the element placement (so called moving finite elements-MFE) based on an interior point strategy. This approach improves the convergence property of the inner problem and removes the nonsmoothness shown in Tanartkit and Biegler.[19,20] The approach was successfully applied in large-scale optimization problems.[22] However, it has also shortcomings; expanding the optimization problem by additional error constraints is needed and a two-layer optimization problem has to be used.

In this work, we apply some of the above concepts to extend the quasi-sequential approach and control the accuracy of state profiles. In the quasi-sequential approach, the discretized DAEs are solved in a simulation layer. This provides the possibility to handle the accuracy of state profiles only in the simulation layer. This is a fundamental advantage due to the fact that there is no need to handle the element lengths based on adding error constraints. In this way, the state profile accuracy will be ensured in the simulation, and the NLP solver adapts the element lengths only for the optimality of the discretized optimization problem. According to the information of direct error estimation or of a proposed time derivative analysis, subintervals will be introduced to improve the accuracy of the state profiles. Since this is only done in the simulation layer, the size of the NLP problem does not change and the solution strategy need not be restarted. Moreover, an efficient gradient computation scheme can be applied inside the subintervals.

The next section introduces the dynamic optimization problem and provides some background for collocation on finite elements. ''The proposed approach'' section then describes and extends the quasi-sequential strategy, and develops an efficient strategy to introduce subintervals and calculate sensitivity in the inner layer. ''Dynamic optimization case study'' section then presents a case study to demonstrate the performance of the extended quasi-sequential approach with the required accuracy in state profiles. Extensive results from this example indicate that the proposed approach leads to much improved computational performance compared with original quasi-sequential approach. Finally, in the conclusions we summarize the key results of the article and outline areas for future work.

Problem Description

Discretization with collocation

We consider dynamic nonlinear optimization problems with path constraints on state profiles as follows:

$$\min_{\mathbf{u}(t),\mathbf{z}(t),\mathbf{y}(t),t_f} \varphi(\mathbf{z}(t),\mathbf{y}(t),\mathbf{u}(t),t_f), \tag{1}$$

$$\text{s.t. } \frac{d\mathbf{z}(t)}{dt} = \mathbf{F}(\mathbf{z}(t),\mathbf{y}(t),\mathbf{u}(t),t); \quad \mathbf{z}_0 = \mathbf{z}(t_0), \tag{2}$$

$$\mathbf{G}(\mathbf{z}(t),\mathbf{y}(t),\mathbf{u}(t),t) = \mathbf{0}, \tag{3}$$

$$\mathbf{u}(t)_{\min} \le \mathbf{u}(t) \le \mathbf{u}(t)_{\max}; \quad \mathbf{z}(t)_{\min} \le \mathbf{z}(t) \le \mathbf{z}(t)_{\max}; \quad \mathbf{y}(t)_{\min} \le \mathbf{y}(t) \le \mathbf{y}(t)_{\max}, \tag{4}$$

where φ represents a scalar objective function, t_f the final time, $\mathbf{z}(t) \in \Re^{ND}$ differential state profiles described by $\mathbf{F} \in \Re^{ND}$ with $\mathbf{z}_0$ as initial conditions and $\mathbf{y}(t) \in \Re^{NA}$ as algebraic state profiles specified by $\mathbf{G} \in \Re^{NA}$, respectively. Here we assume that the DAE system (2)–(3) is index 1 and that the algebraic variables $\mathbf{y}$(t) can be solved uniquely from (3), once $\mathbf{z}$(t) and $\mathbf{u}$(t) are specified. Eq. 4 represents the bounds on control profiles $\mathbf{u}(t) \in \Re^{NU}$ and path constraints on the state variables. It is noted that the nonlinear inequality constraints can be handled by inserting additional slack variables. As in the simultaneous approach both state and control profiles are discretized on finite elements, i.e., the time period from t_0 to t_f is divided into NL finite elements

$$\Delta t_n = t_{n+1} - t_n; \quad n = 1,\ldots,NL, \tag{5}$$

with t_n, t_{n+1} as start and end time of the nth interval, also called knot locations. The polynomial approximation of the differential state variables is done by the linear combination of the Lagrange polynomials, that is,

$$z_n^m(t) = \sum_{j=0}^{NC} \ell_j(t) \cdot z_{n,j}^m; \quad m = 1,\ldots,ND; \quad n = 1,\ldots,NL$$

$$\ell_j(t) = \prod_{\substack{i=0 \\ i\neq j}}^{NC} \frac{t - t_{n,i}}{t_{n,j} - t_{n,i}}, \tag{6}$$

where NC denotes the number of collocation points in one element, $z_n^m(t)$ the polynomial approximation of the mth differential state variable in the nth element, $z_{n,j}^m$ the mth differential state variable at the jth collocation point in the nth element and $t_{n,i}$ the ith collocation point in the nth element (note here: $t_{n,0}$ or $z_{n,0}^m$ stands for the starting time and the initial condition of the mth differential state variable, respectively, in the nth element). From Eq. 6 the differential states are represented at the collocation points in an element as follows:

$$z_n^m(t_{n,i}) = \sum_{j=0}^{NC} \ell_j(t_{n,i}) \cdot z_{n,j}^m = z_{n,i}^m; \quad i = 1,\ldots,NC \tag{7}$$

and

$$\frac{dz_n^m(t_{n,i})}{dt} = \sum_{j=0}^{NC} \frac{d\ell_j(t_{n,i})}{dt} \cdot z_{n,j}^m; \quad i = 1,\ldots,NC. \tag{8}$$

To ensure continuity of the differential state profiles between two intervals, Radau collocation is used, i.e., the value on the last collocation point of an element is defined as the starting value (initial condition) of the next element:

$$z_{n-1,NC}^m = z_{n,0}^m. \tag{9}$$

In this work, we define controls, without loss of generality, as piecewise constants in each element. Here we denote the discrete values of controls in the vector $\mathbf{U}_n \in \Re^{NU}$ of the nth interval. With the introduction of a normalized time $\tau \in [0,1]$ within each element Δt_n and the polynomial approximation of Eq. 8 the differential states from Eq. 2 can be represented by the residual equation:

$$R_{n,i}^m(t_{n,i}) = \sum_{j=0}^{NC} \frac{d\ell_j(\tau_i)}{d\tau} \cdot z_{n,j}^m - \Delta t_n \cdot F_m(\mathbf{Z}_{n,i}, \mathbf{Y}_{n,i}, \mathbf{U}_n, t_{n,i}) = 0$$
$$i = 1,\ldots,NC; \quad m = 1,\ldots,ND; \quad n = 1,\ldots,NL. \tag{10}$$

where $\mathbf{Z}_{n,i} \in \Re^{ND}$ and $\mathbf{Y}_{n,i} \in \Re^{NA}$ denote the vectors of differential and algebraic states of the nth element on the ith collocation point.

The quasi-sequential approach

To give a clear description of the discretized optimization problem, all differential states in Eq. 10 are summarized in $\mathbf{Z} = [\mathbf{Z}_{1,1,\ldots},\mathbf{Z}_{NL,NC}]^T \in \Re^{NC*ND*NL}$, all algebraic states in $\mathbf{Y} = [\mathbf{Y}_{1,1,\ldots},\mathbf{Y}_{NL,NC}]^T \in \Re^{NC*NA*NL}$ and all controls in $\mathbf{U} = [\mathbf{U}_{1,\ldots},\mathbf{U}_{NL}]^T \in \Re^{NU*NL}$. Applying the discretization to Eqs. 1–4 the discretized optimization problem is obtained as

$$\min_{\mathbf{U},\mathbf{Z},\mathbf{Y}} \varphi(\mathbf{U},\mathbf{Z},\mathbf{Y}) \tag{11}$$

$$\text{s.t. } R_{n,i}^m(t_{n,i}) = 0; \quad z_{n-1,NC}^m = z_{n,0}^m; \quad z_{1,0}^m = z^m(t_o) \tag{12}$$

$$\mathbf{G}_n(\mathbf{Z}_{n,i}, \mathbf{Y}_{n,i}, \mathbf{U}_n, t_{n,i}) = \mathbf{0} \tag{13}$$

$$i = 1,\ldots,NC; \quad m = 1,\ldots,ND; \quad n = 1,\ldots,NL \tag{14}$$

$$\mathbf{U}_{\min} \le \mathbf{U} \le \mathbf{U}_{\max}; \quad \mathbf{Z}_{\min} \le \mathbf{Z} \le \mathbf{Z}_{\max}; \quad \mathbf{Y}_{\min} \le \mathbf{Y} \le \mathbf{Y}_{\max}. \tag{15}$$

In the simultaneous approach **Z, Y,** and **U** are determined directly by the NLP solver. However, the quasi-sequential approach solves the equation system (12) and (13) with a Newton method and the whole system is decomposed into a control and a state space. The optimization problem is therefore solved in a two-layer structure: the solution of the equation system is done in the simulation layer and the optimization in the optimization layer. In the optimization layer, the following reduced problem will be solved

$$\min_{\mathbf{U}} \varphi(\mathbf{U}, \mathbf{X}(\mathbf{U})), \tag{16}$$

$$\text{s.t. } \mathbf{U}_{\min} \le \mathbf{U} \le \mathbf{U}_{\max}; \quad \mathbf{X}_{\min} \le \mathbf{X}(\mathbf{U}) \le \mathbf{X}_{\max}, \tag{17}$$

where $\mathbf{X}(\mathbf{U}) \in \Re^{NC*(NA+ND)*NL}$ stands for the computed differential and algebraic state variables of all finite elements in the simulation layer for a given value of **U**.

The Proposed Approach

Error estimation

Using a discretization method to reformulate the infinite optimization problem to a finite NLP leads to an approximate description of the optimization problem. This section briefly outlines the most common approaches to estimating the approximation error of state profiles with the collocation method. The error estimation in polynomial approximation at Gaussian points was first made by De Boor and Swartz,[23] where the accuracy of the state profiles and their derivatives were analyzed. In De Boor,[14] the first method of knot placement was derived with equidistributing the approximation error over all elements. De Boor[15] proposed a scheme to compute a grid that guarantees the equidistribution. This approach was then introduced to dynamic optimization[13,16] as well as collocation-based distillation optimization.[24] An overview of common methods in estimating the error of approximated state profiles by using the collocation method is given in Russell and Christiansen,[25] where the error was estimated based on (NC + 1)th derivatives of the polynomial approximated solution or by evaluating the residual function of the approximated differential equations at noncollocation points. Based on Russell and Christiansen[25] and Ascher,[26] the study of Vasantharajan and Biegler[16] indicated that for a "sufficiently smooth" function the absolute error ε in an ODE-System can be estimated by

$$\left|c_{\text{nonc}} \cdot \mathrm{R}^{m}_{n,\text{nonc}}\left(t_{n,\text{nonc}}\right)\right| \leq \varepsilon^{m}_{n,\text{nonc}}; \quad m = 1,\ldots,\text{ND}; \quad n = 1,\ldots,\text{NL}. \tag{18}$$

where $\mathrm{R}^{m}_{n,\text{nonc}}(t_{n,\text{nonc}})$ is the residual of the mth differential variable of the collocation approximation (see Eq. 10) at the noncollocation point $t_{n,\text{nonc}}$ (denoted by nonc) in the nth finite interval. This residual is given by:

$$\mathrm{R}^{m}_{n,\text{nonc}}\left(t_{n,\text{nonc}}\right) = \sum_{j=0}^{\text{NC}} \frac{d\ell_j(\tau_{\text{nonc}})}{d\tau} \cdot z^{m}_{n,j} - \Delta t_n \cdot \mathrm{F}_m\left(\mathbf{Z}_{n,\text{nonc}}, \mathbf{Y}_{n,\text{nonc}}, \mathbf{U}_{n,\text{nonc}}, t_{n,\text{nonc}}\right) \tag{19}$$

where $\mathbf{Z}_{n,\text{nonc}} \in \Re^{\text{ND}}$ and $\mathbf{Y}_{n,\text{nonc}} \in \Re^{\text{NA}}$ describe the approximated values at a noncollocation point using the differential and algebraic state variables in the nth element, τ_{nonc} is the noncollocation point in normalized time $\tau \in [0,1]$ for $t_{n,\text{nonc}}$, and $\mathbf{U}_{n,\text{nonc}}$ denotes controls at the noncollocation point (with piecewise constant control in the finite element $\mathbf{U}_{n,\text{nonc}} = \mathbf{U}_n$), respectively. The approximated differential variables at the noncollocation point can be computed by interpolation (see Eq. 6):

$$z^{m}_{n}(t_{n,\text{nonc}}) = \sum_{j=0}^{\text{NC}} \ell_j(t_{n,\text{nonc}}) \cdot z^{m}_{n,j}; \quad m = 1,\ldots,\text{ND} \tag{20}$$

in vector $\mathbf{Z}_{n,\text{nonc}} \in \Re^{\text{ND}}$. Since the DAE system (2)–(3) is index 1, Eq. 13 can be rewritten at noncollocation points[21] and used for the computation of $\mathbf{Y}_{n,\text{nonc}} \in \Re^{\text{NA}}$, algebraic state variables at the noncollocation points:

$$\mathbf{G}_n\left(\mathbf{Z}_{n,\text{nocn}}, \mathbf{Y}_{n,\text{nonc}}, \mathbf{U}_{n,\text{nonc}}, t_{n,\text{nonc}}\right) = \mathbf{0} \tag{21}$$

The constant c_{nonc} in Eq. 18 depends only on τ_{nonc} and the collocation order NC:

$$c_{\text{nonc}} = \frac{1}{a_{\text{nonc}}} \int_0^{\tau_{\text{nonc}}} \left(\prod_{i=1}^{\text{NC}} (s - \tau_i)\right) \mathrm{ds} \tag{22}$$

with

$$a_{\text{nonc}} = \prod_{i=1}^{\text{NC}} (\tau_{\text{nonc}} - \tau_i). \tag{23}$$

To ensure the accuracy over a whole finite element, it is necessary to evaluate (18) over a fine grid. However this can be computationally expensive. To reduce the computational load only one or a few noncollocation points are evaluated in each element.[21] However, no guidelines on the number and location of selected points have been stated in the previous studies. Here we select the center of neighboring collocation points as the noncollocation point for evaluating the approximation errors (e.g. 3 noncollocation points will be selected for a 3-point-collocation scheme). The error estimation is done by evaluating Eqs. 18–23 for every noncollocation point. In the next subsection, we propose an alternative to this error estimation strategy.

Time derivative analysis

This section presents a criterion for an estimation of the accuracy of polynomial approximated state profiles. Our goal is to derive this criterion based only on available information, that is, without any additional intensive computations. The main reason for this treatment is to reduce the computational load, since the calculations are needed for each element and each iterate of the NLP solver.

Russell and Christiansen[25] estimated the approximation error based on the (NC + 1)th time derivatives. The objective of our derivative analysis is to simplify the procedure based only on known information of the first time derivatives during the simulation. From the NLP problem (11)–(15), one can see that the discretized controls, the differential and algebraic states as well as element lengths will be determined directly or indirectly by the NLP solver (depending on the approach used). Through the analysis of the residual Eq. 10, we have additional information on the time derivatives of the differential states at collocation points. This is obtained without any additional computational load through function evaluations available during the solution of the NLP.

It is well-known that exact state profiles can be obtained with the collocation method for polynomials of a certain order (e.g. 2NC at Radau points). However, with the above mentioned information a polynomial order of state variables for satisfying a predefined accuracy tolerance cannot be determined without additional computations as in the "Error estimation" section (interpolation to noncollocation points, evaluation of the algebraic equation and reevaluation of the residual equation). Instead, we use time derivatives of the differential state variables at collocation points to evaluate the degree of their nonlinearity along the time profiles.

For an easier illustration, we investigate in the following only two neighboring collocation points at t_1 and t_2 and one differential state variable $z(t)$. The basic idea is that the time behavior of a constant time derivative results in a linear time profile. Based on this, we formulate the following criterion:

$$\alpha = ||\dot{z}(t_1)| - |\dot{z}(t_2)||. \tag{24}$$

Here $\alpha > 0$ represents a profile with a nonlinear behavior. A change of the time derivatives indicates a degree of nonlinearity of the state profile. To hold these values in the range of [0, 1] we normalize α by

$$\alpha_{\text{rel}} = \begin{cases} \dfrac{\alpha}{\max\{|\dot{z}(t_1)|, |\dot{z}(t_2)|\}}, & \text{if} \quad \max\{|\dot{z}(t_1)|, |\dot{z}(t_2)|\} \neq 0 \\ \alpha, & \text{else} \end{cases}. \tag{25}$$

In other words, we can expect α_{rel} as a ''relative Manhattan distance'' to a linear profile. One can choose an allowed range tol of this ''distance'', so that the approximation of the state profile is sufficiently accurate:

$$\alpha_{\text{rel}} \leq \text{tol}. \tag{26}$$

To use the curvature of state profiles as an indicator of nonlinearity the relative change of the states on the collocation points to the values on the initial point is considered as a test criterion

$$\alpha_{n,i}^{m} = \begin{cases} \dfrac{\left|\left|\dot{z}_{n,0}^{m}\right| - \left|\dot{z}_{n,i}^{m}\right|\right|}{\max\left\{\left|\dot{z}_{n,0}^{m}\right|, \left|\dot{z}_{n,i}^{m}\right|\right\}}, & \text{if} \quad \max\left\{\left|\dot{z}_{n,0}^{m}\right|, \left|\dot{z}_{n,i}^{m}\right|\right\} \neq 0 \\ \left|\left|\dot{z}_{n,0}^{m}\right| - \left|\dot{z}_{n,i}^{m}\right|\right|, & \text{else} \end{cases} \leq \text{tol}_i^m$$
$$i = 1,\ldots,\text{NC};\ \ m = 1,\ldots,\text{ND};\ \ n = 1,\ldots,\text{NL}. \tag{27}$$

where $\alpha_{n,i}^{m}$ represents the ith criterion of the mth differential state in the nth element, $\dot{z}_{n,k}^{m}$ the first derivative at the kth collocation point of the mth differential state in the nth element and tol_i^m the ith allowed tolerance of the mth differential state in all finite elements, respectively. By specifying the tolerances from Eq. 27, the allowed state curvature can now be set by the user. We have no direct mathematical relation between the α value and the error in the state profiles at the moment. Therefore the time derivative analysis is only empirical. In addition, the simulation of the model equations with an appropriate simulation environment can support the user in selecting these tolerance values.

Extension of the quasi-sequential approach

In this section, we extend the quasi-sequential approach described before by introducing variable finite element lengths and ensuring a user-defined relative error tolerance for the accuracy of state profiles. By changing the length of time elements to satisfy the specified error tolerance increases the degrees of freedom of the NLP. Studies aiming at this have been made within the simultaneous approach.[16–22] In the quasi-sequential approach, the objective function φ is minimized and the constraints on states are handled with respect to $\mathbf{U}$ and $\Delta\mathbf{T} = [\Delta t_1,\ldots,\Delta t_{\text{NL}}]^{\text{T}} \in \Re^{\text{NL}}$ with additional inequality constraints on $\Delta\mathbf{T}$. This results in a modification of Eqs. 16 and 17 to

$$\min_{\mathbf{U},\Delta\mathbf{T}} \varphi(\mathbf{U}, \Delta\mathbf{T}, \mathbf{X}(\mathbf{U}, \Delta\mathbf{T})) \tag{28}$$

$$\text{s.t. } \Psi_{\text{rel}}\{\mathbf{X}(\mathbf{U}, \Delta\mathbf{T}), \varepsilon_{\text{rel}}\} \leq \mathbf{0} \tag{29}$$

$$\mathbf{U}_{\min} \leq \mathbf{U} \leq \mathbf{U}_{\max};\quad \Delta\mathbf{T}_{\min} \leq \Delta\mathbf{T} \leq \Delta\mathbf{T}_{\max};$$
$$\mathbf{X}_{\min} \leq \mathbf{X}(\mathbf{U}, \Delta\mathbf{T}) \leq \mathbf{X}_{\max} \tag{30}$$

with the inequality constraints for the error evaluations of the state profiles Eq. 29 and the results of the simulation layer $\mathbf{X}(\mathbf{U},\Delta\mathbf{T})$ for a given $\mathbf{U}$ and $\Delta\mathbf{T}$. Here the operator Ψ_{rel} represents the relative error evaluations from sections ''Error estimation'' and ''Time derivative analysis.'' Choosing the method of error prediction at noncollocation points, Ψ_{rel} evaluates Eq. 18 at every noncollocation point and estimates the relative error by dividing ε by the values of the computed states at the corresponding noncollocation point (see Eq. 20). With the method of time derivative analysis, the operator Ψ_{rel} computes the state curvatures and compares these with the allowed tolerance values (see Eq. 27).

In the simultaneous approach, the MFE is successfully applied to handle accuracy requirements based on the error prediction method with additional constraints like Eq. 29.[16–22] The major difficulties in solving such NLPs lie in the additional nonlinear constraints of the error observations and the possibility of an empty solution space if not enough elements are present to satisfy Eq. 29.[16] To overcome these difficulties in the simultaneous approach, the NLP was solved with a bi-level optimization framework.[19–21] The main disadvantage of this procedure lies in resolving the dynamic optimization in the inner problem.[21]

In our approach, the above-mentioned disadvantages can be prevented by handling the accuracy requirements, that is, satisfying Eq. 29, in the simulation layer. Thus the NLP in the optimization layer of the quasi-sequential approach can be reformulated as

$$\min_{\mathbf{U},\Delta\mathbf{T}} \varphi(\mathbf{U}, \Delta\mathbf{T}, \mathbf{X}(\mathbf{U}, \Delta\mathbf{T})) \tag{31}$$

$$\text{s.t. } \mathbf{U}_{\min} \leq \mathbf{U} \leq \mathbf{U}_{\max};\ \Delta\mathbf{T}_{\min} \leq \Delta\mathbf{T} \leq \Delta\mathbf{T}_{\max};$$
$$\mathbf{X}_{\min} \leq \mathbf{X}(\mathbf{U}, \Delta\mathbf{T}) \leq \mathbf{X}_{\max}. \tag{32}$$

This extension enables the NLP solver to determine the lengths of time intervals based only on the optimality conditions, without additional nonlinear constraints for handling the accuracy of the approximation of state profiles.

The extension of the quasi-sequential approach is presented on the right side of Figure 1 and is illustrated in black. After convergence of the Newton method, for solving the model equations in an element, the accuracy conditions will be checked with the error estimation method (see ''Error estimation'' section) or the time derivative analysis (see ''Time derivative analysis'' section). If the conditions are not satisfied, subintervals will be introduced to simulate from one collocation point of the original discretization to the next. In this case, values of state variables over the original finite element at the collocation points will be computed by introducing subintervals (called sub-simulation).

The sub-simulation is illustrated in Figure 2 where the relations between the collocation points of the original (''o'') to those of the subintervals (''x'') are shown (using a 3-point collocation scheme with 1 and 2 necessary subintervals for the first and the remaining original collocation points). As a result, it is possible to ensure the approximation accuracy of state profiles without changing the dimension of the original optimization problem. In this way, there is no need to restart

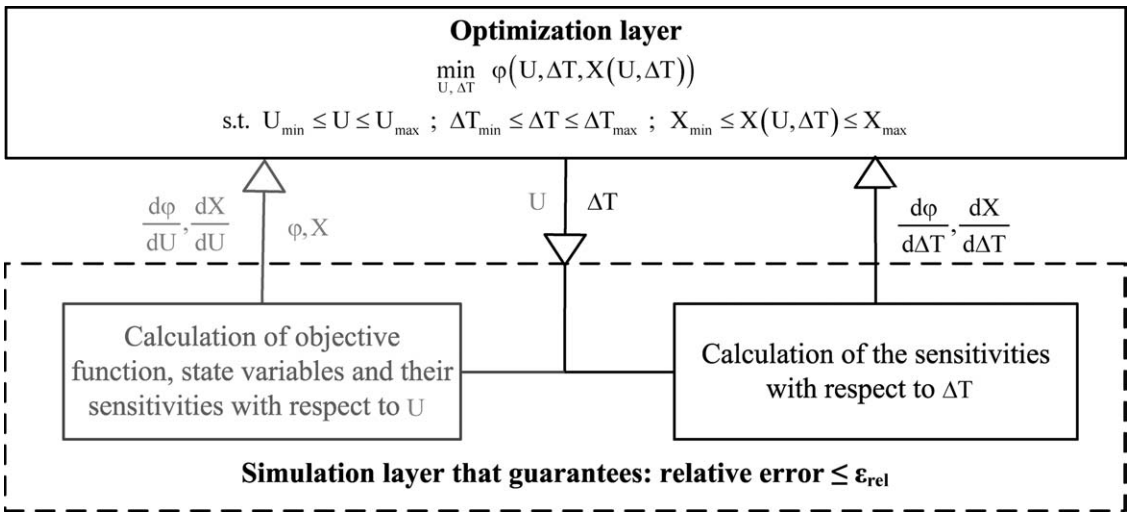

Figure 1. The quasi-sequential approach and the extension.

the NLP solver at each iteration. We call the proposed procedure qMFE. Obviously no subintervals will be introduced if the approximation in a finite element is sufficiently accurate. The sensitivity computations required in Figures 1 and 2 are described in the next section.

Sensitivity computation

In the quasi-sequential approach the DAE system is addressed by solving Eq. 12 and 13 successively from one element to the next. The sensitivities of states with respect to controls and the element lengths can be computed in parallel to the model solution. This is done at every iteration of the NLP solver. The discretized model equations in the elements Eqs. 12 and 13 can be rewritten as

$$\mathbf{C}_n\left(\mathbf{X}_{n,0}, \mathbf{X}_n, \mathbf{U}_n, \Delta t_n\right) = \mathbf{0}; \quad n = 1, \ldots, \mathrm{NL}. \tag{33}$$

where $\mathbf{X}_n$ comprises the differential and algebraic states at all collocation points in the nth interval with the initial conditions $\mathbf{X}_{n,0} = \mathbf{D}\mathbf{X}_{n-1}$, $\mathbf{D}$ represents the mapping matrix of the initial conditions in the finite elements from $\mathbf{X}_{n-1}$ to $\mathbf{X}_{n,0}$, respectively. The first order Taylor expansion of Eq. 33 results in

$$\frac{d\mathbf{C}_n}{d\mathbf{X}_{n,0}} \cdot \Delta\mathbf{X}_{n,0} + \frac{d\mathbf{C}_n}{d\mathbf{X}_n} \cdot \Delta\mathbf{X}_n + \frac{d\mathbf{C}_n}{d\mathbf{U}_n} \cdot \Delta\mathbf{U}_n + \frac{d\mathbf{C}_n}{d(\Delta t_n)} \cdot \Delta(\Delta t_n) = \mathbf{0} \tag{34}$$

or

$$\mathbf{K}_n \cdot \Delta\mathbf{X}_{n,0} + \mathbf{L}_n \cdot \Delta\mathbf{X}_n + \mathbf{M}_n \cdot \Delta\mathbf{U}_n + \mathbf{N}_n \cdot \Delta(\Delta t_n) = \mathbf{0}. \tag{35}$$

From this, we have the sensitivities of states with respect to controls, to element lengths and initial conditions in each element:

$$\frac{d\mathbf{X}_n}{d\mathbf{U}_n} = -\mathbf{L}_n^{-1} \cdot \mathbf{M}_n \tag{36}$$

$$\frac{d\mathbf{X}_n}{d(\Delta t_n)} = -\mathbf{L}_n^{-1} \cdot \mathbf{N}_n \tag{37}$$

$$\frac{d\mathbf{X}_n}{d\mathbf{X}_{n,0}} = -\mathbf{L}_n^{-1} \cdot \mathbf{K}_n \tag{38}$$

The matrix $\mathbf{L}_n$ can be taken from the last Newton step that solves Eq. 33, that is, available information can be used again (e.g., the computed LU-decomposition) to solve Eq. 36–38. The continuity of the state profiles allows to transfer the sensitivities from element to element by using the chain rule.[3] Thus the sensitivities of the states in the nth element with respect to the controls or element lengths of the $i = 1 \ldots n-1$ can be computed by

$$\frac{d\mathbf{X}_n}{d\mathbf{U}_i} = \frac{d\mathbf{X}_n}{d\mathbf{D}\mathbf{X}_{n-1}} \cdot \frac{d\mathbf{D}\mathbf{X}_{n-1}}{d\mathbf{U}_i} = \frac{d\mathbf{X}_n}{d\mathbf{X}_{n,0}} \mathbf{D} \frac{d\mathbf{X}_{n-1}}{d\mathbf{U}_i}, \tag{39}$$

$$\frac{d\mathbf{X}_n}{d(\Delta t_i)} = \frac{d\mathbf{X}_n}{d\mathbf{D}\mathbf{X}_{n-1}} \cdot \frac{d\mathbf{D}\mathbf{X}_{n-1}}{d(\Delta t_i)} = \frac{d\mathbf{X}_n}{d\mathbf{X}_{n,0}} \mathbf{D} \frac{d\mathbf{X}_{n-1}}{d(\Delta t_i)}, \tag{40}$$

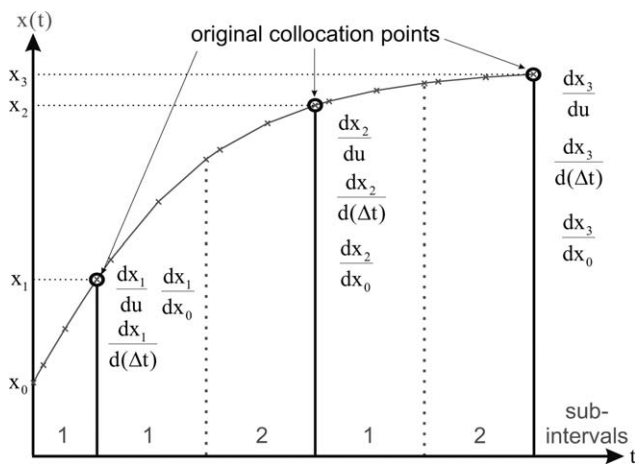

Figure 2. Sub-simulation.

For $i \geq 0$ the sensitivities are given in Eqs. 36 and 37. The evaluation of Eq. 38 is necessary to compute Eqs. 39 and 40. In Figure 1, the sensitivities of $\mathbf{X}$ are decomposed into control variables and element lengths to illustrate the extended approach for element placement-qMFE. Through the above computation we receive all sensitivity information required:

$$\frac{d\mathbf{X}}{d\mathbf{U}} = \begin{bmatrix} \frac{d\mathbf{X}_1}{d\mathbf{U}_1} & & & & \mathbf{0} \\ \vdots & \ddots & & & \\ \frac{d\mathbf{X}_n}{d\mathbf{U}_1} & \cdots & \frac{d\mathbf{X}_n}{d\mathbf{U}_n} & & \\ \vdots & \ddots & \vdots & \ddots & \\ \frac{d\mathbf{X}_{\mathrm{NL}}}{d\mathbf{U}_1} & \cdots & \frac{d\mathbf{X}_{\mathrm{NL}}}{d\mathbf{U}_n} & \cdots & \frac{d\mathbf{X}_{\mathrm{NL}}}{d\mathbf{U}_{\mathrm{NL}}} \end{bmatrix};$$

$$\frac{d\mathbf{X}}{d(\Delta\mathbf{T})} = \begin{bmatrix} \frac{d\mathbf{X}_1}{d(\Delta t_1)} & & & & \mathbf{0} \\ \vdots & \ddots & & & \\ \frac{d\mathbf{X}_n}{d(\Delta t_1)} & \cdots & \frac{d\mathbf{X}_n}{d(\Delta t_n)} & & \\ \vdots & \ddots & \vdots & \ddots & \\ \frac{d\mathbf{X}_{\mathrm{NL}}}{d(\Delta t_1)} & \cdots & \frac{d\mathbf{X}_{\mathrm{NL}}}{d(\Delta t_n)} & \cdots & \frac{d\mathbf{X}_{\mathrm{NL}}}{d(\Delta t_{\mathrm{NL}})} \end{bmatrix}. \quad (41)$$

The above sensitivity computation is performed inside the elements when no subintervals are introduced. As mentioned in the section of ''Extension of the quasi-sequential approach'' and shown in Figure 2, we introduce subintervals in a given element based on locations of the collocation points. Here, the sensitivities of states are still required for the original collocation points with respect to the controls and element lengths (see Figure 1 and Figure 2). Furthermore, the sensitivities at the last point in every subinterval to the initial conditions are needed to transfer the sensitivities of the previous elements using Eq. 39 and 40 up to the last collocation point of the subinterval.

To illustrate, we derive the sensitivities over $\mathrm{NL_{AC}}$ subintervals, starting from the first collocation point in the nth finite element. For these subintervals, Eqs. 12 and 13 can be reformulated to

$$\mathbf{C}_j\left(\mathbf{S}_{j,0}, \mathbf{S}_j, \mathbf{U}_j^S, \Delta t_j^S\right) = \mathbf{0};\ \mathbf{S}_{0,0} = \mathbf{X}_{n,0};\ \mathbf{S}_{j,0} = \mathbf{D}\mathbf{S}_{j-1};\quad j = 1, \ldots, \mathrm{NL_{AC}}, \quad (42)$$

where $\mathbf{S}$ represents the discretized states in the sub-simulation (at collocation points ''x'' in Figure 2) as was done with $\mathbf{X}$ in Eq. 33. Applying the above sensitivity analysis to Eq. 42 we have:

$$\frac{d\mathbf{S}}{d\mathbf{U}^{\mathrm{S}}} = \begin{bmatrix} \frac{d\mathbf{S}_1}{d\mathbf{U}_1^{\mathrm{S}}} & & & & \mathbf{0} \\ \vdots & \ddots & & & \\ \frac{d\mathbf{S}_n}{d\mathbf{U}_1^{\mathrm{S}}} & \cdots & \frac{d\mathbf{S}_n}{d\mathbf{U}_n^{\mathrm{S}}} & & \\ \vdots & \ddots & \vdots & \ddots & \\ \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d\mathbf{U}_1^{\mathrm{S}}} & \cdots & \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d\mathbf{U}_n^{\mathrm{S}}} & \cdots & \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d\mathbf{U}_{\mathrm{NL_{AC}}}^{\mathrm{S}}} \end{bmatrix};$$

$$\frac{d\mathbf{S}}{d(\Delta\mathbf{T}^{\mathrm{S}})} = \begin{bmatrix} \frac{d\mathbf{S}_1}{d(\Delta t_1^{\mathrm{S}})} & & & & \mathbf{0} \\ \vdots & \ddots & & & \\ \frac{d\mathbf{S}_n}{d(\Delta t_1^{\mathrm{S}})} & \cdots & \frac{d\mathbf{S}_n}{d(\Delta t_n^{\mathrm{S}})} & & \\ \vdots & \ddots & \vdots & \ddots & \\ \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d(\Delta t_1^{\mathrm{S}})} & \cdots & \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d(\Delta t_n^{\mathrm{S}})} & \cdots & \frac{d\mathbf{S}_{\mathrm{NL_{AC}}}}{d(\Delta t_{\mathrm{NL_{AC}}}^{\mathrm{S}})} \end{bmatrix} \quad (43)$$

Based on the above discussion, the sensitivities of the first collocation point in the original finite element now correspond to the last collocation point, for every state in the sub-simulation to be computed. Moreover, the last rows in Eq. 43 are of interest only if they depend on the original controls and element lengths. Inside these subintervals the control variables and element lengths must depend on the controls and element lengths of the original finite element.

With piecewise constant controls in the original finite element and a given number of subintervals the dependencies can be described as

$$\mathbf{U}_j^{\mathrm{S}} = \mathbf{U}_n;\quad j = 1, \ldots, \mathrm{NL_{AC}} \quad (44)$$

$$\Delta t_j^{\mathrm{S}} = \frac{\Delta t_n \cdot \tau_1}{\mathrm{NL_{AC}}};\quad j = 1, \ldots, \mathrm{NL_{AC}}, \quad (45)$$

where τ_1 is the first collocation point in the normalized range $\tau \in [0,1]$ and $\mathrm{NL_{AC}}$ represents the subintervals needed to the first collocation point (c.f. Figure 2, there $\mathrm{NL_{AC}}$ equals to one exemplary). By replacing $\mathbf{U}_j^{\mathrm{S}}$ and Δt_j^{S} in Eq. 42 with Eqs. 44 and 45, the sensitivities of the states inside the subintervals can be computed with respect to the original controls and element lengths and then transferred by the chain rule to the last collocation points of the sub-simulation (which are the first collocation points of the original finite element). This is similar to Eqs. 36 and 37 during a normal simulation.

The sensitivities at the first collocation point (of the original finite element) with respect to the initial condition can be easily computed with the chain rule from the first subinterval to the last collocation point of the sub-simulation (similar to Eq. 38 in a normal simulation). As mentioned before, the computational efforts for the sensitivities can be reduced to the calculation of the sensitivities of the last collocation point of the subintervals, since the other sensitivities are not necessary for the sensitivities of the original collocation points. The sensitivities of the states at the remaining original collocation points can be easily derived with the same procedure.

The resulting procedure calculates the reduced gradients with respect to **U** and $\Delta\mathbf{T}$ that are used in the NLP (31)–(32). It should be noted that the introduction of subintervals at a particular NLP iteration does lead to structural changes in the collocation equations and the sensitivity calculation. Therefore, even though the NLP has the same number of variables, adding more elements in the simulation layer will produce different function values, reduced gradients and KKT multipliers, than if none were added. However, this should not disrupt the convergence of the NLP. Instead, it effectively moves the current iterate to a different level of approximation, where the algorithm can continue until convergence. On the other hand, some care should be taken with the application of SQP algorithms, as "memory dependent" factors (e.g., BFGS updates and estimates of the line search parameters) could be affected. These difficulties were not observed in this study.

Dynamic Optimization Case Study

The optimization problem

This section considers an optimal control problem of a batch beer fermentation process taken from Gee and Ramirez.[27] In this case we consider a weighted objective function that seeks maximum ethanol production while simultaneously minimizing operating time. The fermentation medium contains three sugar sources, glucose g(t), maltose m(t), and maltotriose n(t). The nonlinear DAE description is given by the rate equations, temperature temp(t) change in the medium and by the biomass xb(t), and ethanol e(t) relations as follows:

$$
\begin{aligned}
\frac{d\mathrm{g}(t)}{dt} &= -\mu_1(\mathrm{g}(t))\mathrm{xb}(t)\\
\frac{dm(t)}{dt} &= -\mu_2(m(t),\mathrm{g}(t))\mathrm{xb}(t)\\
\frac{dn(t)}{dt} &= -\mu_3(n(t),m(t),\mathrm{g}(t))\mathrm{xb}(t)\\
\frac{d\mathrm{temp}(t)}{dt} &= \frac{1}{\rho \mathrm{C_P}}\left[\Delta \mathrm{H}_{\mathrm{F_G}}\frac{d\mathrm{g}(t)}{dt}+\Delta \mathrm{H}_{\mathrm{F_M}}\frac{dm(t)}{dt}+\Delta \mathrm{H}_{\mathrm{F_N}}\frac{dn(t)}{dt}\right.\\
&\qquad \left.-u(t)(\mathrm{temp}(t)-\mathrm{TEMP_C})\right]\\
&\mathrm{xb}(0)+\mathrm{R}_{\mathrm{X_g}}(\mathrm{g}(t_0)-\mathrm{g}(t))+\mathrm{R}_{\mathrm{X}_m}(m(t_0)-m(t))\\
&\qquad +\mathrm{R}_{\mathrm{X}_n}(n(t_0)-n(t))-\mathrm{xb}(t)=0\\
&\mathrm{e}(0)+\mathrm{R}_{\mathrm{E_g}}(\mathrm{g}(t_0)-\mathrm{g}(t))+\mathrm{R}_{E_m}(m(t_0)-m(t))\\
&\qquad +\mathrm{R}_{En}(n(t_0)-n(t))-\mathrm{e}(t)=0
\end{aligned}
\tag{46}
$$

A detailed description of the model is given in Appendix. There are four differential state variables g(t), m(t), n(t), and temp(t) and two algebraic states xb(t) and e(t) with the initial condition g(0) = 70 mol m^{-3}; m(0) = 220 mol m^{-3}; n(0) = 40 mol m^{-3}; temp(0) = 8°C; xb(0) = 175 mol m^{-3}, and e(0) = 0 mol m^{-3}. The process is controlled with the cooling rate u(t) with a lower and upper boundary

$$u_{\min} \le u(t) \le u_{\max} \tag{47}$$

where $u_{\min}$ = 0 kJ h^{-1} m^{-3} °C^{-1} and $u_{\max}$ = 30 kJ h^{-1} m^{-3} °C^{-1}. The fermentation mechanism is to be ensured by a constrained temperature profile during the operation

$$\mathrm{temp}_{\min} \le \mathrm{temp}(t) \le \mathrm{temp}_{\max} \tag{48}$$

with $\mathrm{temp}_{\min}$ = 0°C and $\mathrm{temp}_{\max}$ = 12°C. In Gee and Ramirez,[27] the objective function of the optimization problem is defined by trading off maximum final ethanol concentration and minimum final time t_f (batch time) with a weighting factor w, that is,

$$\min_{u(t),t_\mathrm{f}} \varphi = \min_{u(t),t_\mathrm{f}} \{-e(t_\mathrm{f}) + w \cdot t_\mathrm{f}\}. \tag{49}$$

The constrained optimal control problem consists of Eqs. 46–49 (here w = 3 mol h^{-1} m^{-3} is used). We apply in the next section our qMFE strategy to this dynamic optimization problem.

Results and discussions

The problem solution is realized with a discretization of the DAE description from Eq. 46 through the collocation method presented in section "Problem description.". Here we use NC = 3 collocation points, NL = 40 finite elements and a piecewise constant control in every element, that is

$$\mathbf{U} = [u_1, u_2, \ldots, u_{40}]^T; \quad \Delta\mathbf{T} = [\Delta t_1, \Delta t_2, \ldots, \Delta t_{40}]^\mathrm{T}. \tag{50}$$

From Gee and Ramirez,[27] the optimal control strategy has a hard switch in the time profile where the control goes to the upper bound and a second time point where the control leaves the upper bound. To detect these time points with the qMFE strategy (see "Extension of the quasi-sequential approach" section) the following reduced NLP problem is solved in the optimization layer

$$\min_{\mathbf{U},\Delta\mathbf{T}} \left\{ -\mathrm{e}(t_\mathrm{f}) + w \sum_{i=1}^{\mathrm{NL}} \Delta t_i \right\} \tag{51}$$

$$
\begin{aligned}
\text{s.t. } u_{\min} \le \mathbf{U} \le u_{\max}; \quad & \Delta\mathbf{T}_{\min} \le \Delta\mathbf{T} \le \Delta\mathbf{T}_{\max};\\
& \mathrm{temp}_{\min} \le \mathbf{TEMP} \le \mathrm{temp}_{\max}
\end{aligned}
\tag{52}
$$

where e(t_f) is the ethanol concentration at the last collocation point and **TEMP** the vector of the discrete temperature description. The values of e(t_f), **TEMP** and the corresponding gradients with respect to the control variables and element lengths are calculated in the simulation layer with the proposed procedure from section "Sensitivity computation" (see also Figure 1). The approach is implemented in the MATLAB® environment and uses the automatic differentiation software INTLAB[28] to calculate the sensitivities on an Intel Core 2 duo with 2.4 GHz and Microsoft Windows XP® The initial starting values (denoted by "^{0}") of the control variable are assumed as constant over the whole time period and the initial element lengths are chosen as:

$$\mathbf{U}^0 = [30, \ldots, 30]^\mathrm{T}; \ \Delta\mathbf{T}^0 = [400/\mathrm{NL}, \ldots, 400/\mathrm{NL}]^\mathrm{T}. \tag{53}$$

This corresponds to an initial final time of t_f^0 = 400h. The constraints on **U**, **TEMP,** and $\Delta\mathbf{T}$ are given by Eqs. 47 and 48, respectively. For the boundary of time intervals we choose

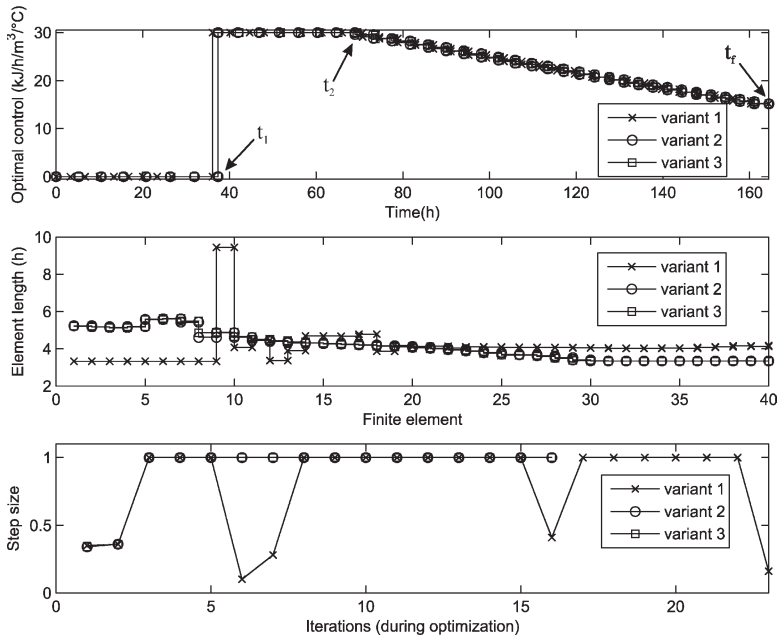

Figure 3. Comparison of the solution strategies.

$\Delta \mathbf{T}_{\min} = \Delta \mathbf{T}^0/3$ and $\Delta \mathbf{T}_{\max} = 5\Delta \mathbf{T}^0/3$. The solution of the NLP (51) and (52) is obtained by the SQP algorithm using SNOPT[29] and the result is shown in Figure 3. The problem is solved first without controlling the accuracy of the state profiles (the original quasi-sequential, called Variant 1) and then with the accuracy method from the sections "Error estimation" (Variant 2) and "Time derivative analysis" (Variant 3).

The first part of Figure 3 shows the optimal cooling strategy, the second part the optimal lengths of elements, and the third part characterizes the step length behaviors during the solution process of the three variants. The interpretation of the optimization results from a chemical or biological point of view it not in the aim of this work. Therefore optimal state profiles are not shown or discussed. The optimal cooling strategies obtained by Variants 2 and 3 are nearly

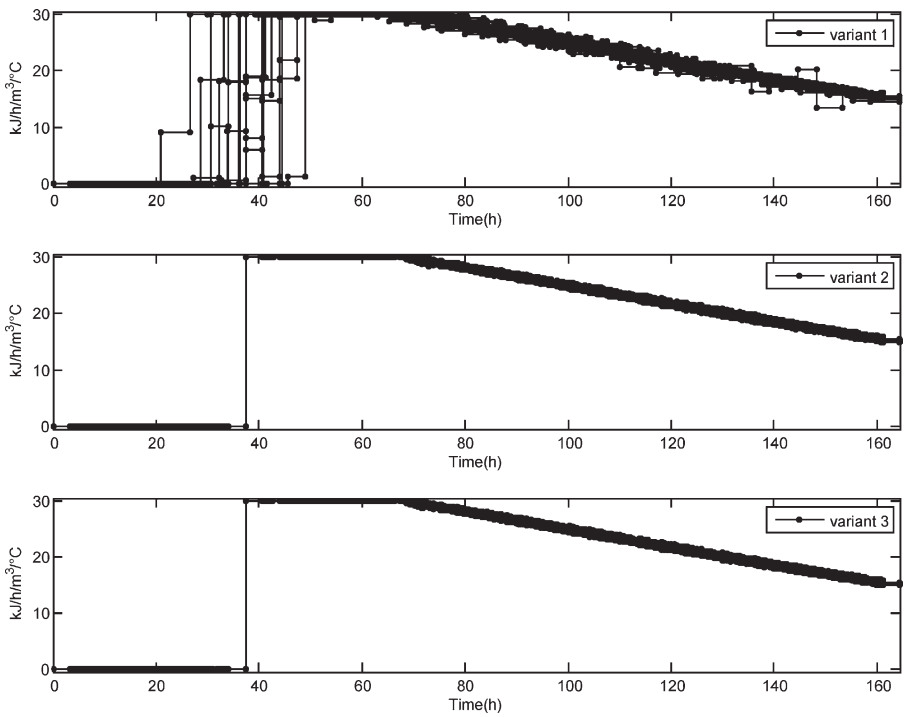

Figure 4. Comparison of optimal cooling profiles from randomly starting points.

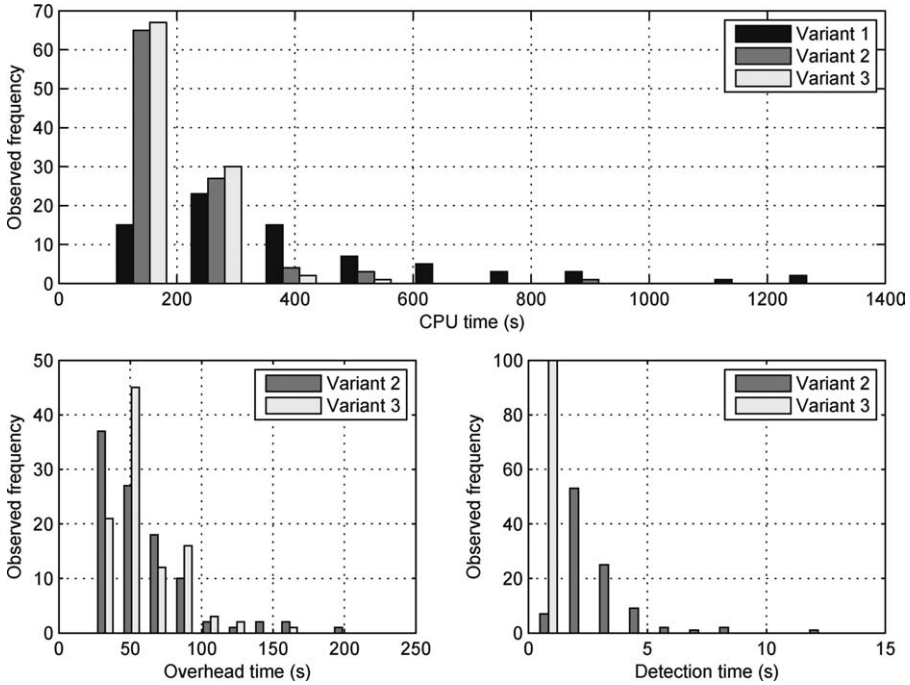

Figure 5. Time performance to random initializations.

identical and reproduce the analytical solution from Gee and Ramirez[27] very well. More importantly, they are more accurate than the results of Variant 1. The optimal time points are $t^*_{1,1} \approx 36.11$h, $t^*_{1,2} \approx t^*_{1,3} \approx 37.37$h, $t^*_{2,1} \approx 70.74$h, $t^*_{2,2} \approx 69.40$h, $t^*_{2,3} \approx 69.41$h and $t^*_{f,1} \approx t^*_{f,2} \approx t^*_{f,3} \approx 164.4$h, where the second index denotes the corresponding variant.

The optimal element lengths of Variant 1 (see middle graphic of Figure 3) are between 3.3333 h (i.e., at lower bound of the elements) and 9.4417 h. The optimal element lengths of Variants 2 and 3 are nearly identical and they range between 3.3345 and 5.6213 h. The large elements of Variant 1 impair the approximation of the collocation method since Variant 1 has no accuracy control, while the smaller range of the element lengths from Variants 2 and 3 improve the approximation accuracy.

Interestingly, the Variants 2 and 3 also have better convergence performance than Variant 1. This can be observed from the third graphic of Figure 3, where Variant 1 needs more iterations (23 vs. 16 by Variants 2 and 3) and requires lower step sizes at four iterations compared to Variants 2 and 3. Here the performance of Variants 2 and 3 is nearly identical. However, Variants 2 and 3 take more computational time to ensure the required accuracy in each iteration by detection and handling of approximation errors than Variant 1 where the standard quasi-sequential approach (without error control) is used.

The total CPU time in solving the problem is $t_{CPU,1} = 195.74$s, $t_{CPU,2} = 125.26$s and $t_{CPU,3} = 123.41$s, respectively. The accuracy controlling methods allow the optimization to be accelerated by about 40% due to fewer iterations of NLP and larger step sizes in the line search. In Variants 2 and 3, the total CPU time to compute the error criteria during the optimization (in the following called detection time) is $t_{detect,2} = 2.101$s and $t_{detect,3} = 0.377$s (i.e., with variant 3 82% of the detection time can be saved), respectively. The CPU time per iteration of Variants 2 and 3 is $t_{iteration,2} = 7.829$s and $t_{iteration,3} = 7.713$s, respectively. Nevertheless, MATLAB® and INTLAB are used to implement this test environment, which require relatively high CPU time.

In addition, we now analyze the robustness of convergence to the solution for the three variants with different starting control profiles. We consider 100 random initializations of the control profile in every finite element between 0 and 30 kJ h^{-1} m^{-3} $°C^{-1}$ and the element lengths between 5 and 10 h. For all three variants the same randomly generated initial profiles are used. The results of the optimal cooling strategies are shown in Figure 4. It can be seen that in Variant 1 the optimal switching strategy cannot be identified (and the optimization algorithm with Variant 1 fails for 26 cases). In Variants 2 and 3 the switching behavior is clearly identified in every randomly initialized case. As a result, the extension by using the qMFE strategy proposed in this article makes the quasi-sequential approach more robust to the initialization in solving the problem. The optimal element lengths of Variant 1 again have the trend to spread over a wide range.

Figure 5 compares the performance in solving the optimization problem for the randomly initialized starting points. The top graphic of Figure 5 shows the observed frequency of the total CPU time needed to solve the problem. It can be seen that Variant 1 needs from 150 to 1250 CPU seconds, and shows worse time performance in comparison to Variants 2 and 3. The distributions of Variants 2 and 3 are more skewed to the left side, and the behavior is again much better than that of Variant 1.

The bottom graphics of Figure 5 compare Variants 2 and 3 based on the frequency of the required total CPU time (left) and the detection time needed (right). The overhead time is the time needed for the simulations and gradient

computations in the subintervals. The detection time of Variant 3 is expectedly shorter than that of Variant 2, which is caused by the better performance of the error detection algorithm of Variant 3. From Figure 5, it can be concluded that it is important to consider accuracy control in state profiles by applying a moving finite element strategy in the optimization problem within the quasi-sequential approach.

Conclusions

We extend the quasi-sequential approach by an accuracy controlling strategy for state profiles called qMFE. The accuracy required is ensured in the simulation layer of the quasi-sequential approach with an error estimation method and a time derivative analysis. To solve the problem with an SQP method, we derive the modified sensitivity computations during the accuracy handling in an efficient way. Based on a case study we demonstrate the necessity of accurate control of the state profiles by adding subintervals. Indeed, the application of these strategies leads to better convergence, reduces the computation time and results in greater robustness to the initial trajectory of the dynamic optimization problem.

The two accuracy handling methods presented produce nearly the same results. But from the aspect of time performance the time derivative analysis outperforms the error estimation method because of its much faster error detection. The time derivative approach is empirical but it works well in practice and motivates the need for additional investigations.

Our approach guarantees the accuracy of state profiles in the simulation layer without increasing the number of finite elements in the optimization problem. Thus in the optimization layer only the task of solving the optimization problem is carried out and there is no need to restart the NLP solver. In this way we can guarantee the accuracy of the state profiles and solve the optimization problem more efficiently.

In future work we will apply qMFE to a set of large-scale dynamic optimization problems and compare them with the MFE strategy in the simultaneous approach. In addition, it is necessary to extend our approach to handle mathematical programs with equilibrium constraints (MPEC), where a moving finite elements strategy is required.[2] Furthermore a method will be considered to automatically determine the number of necessary finite elements for sufficiently accurate description of the real optimal solution.

Acknowledgments

We would like to thank Horst Puta and Stefan Röll for many helpful discussions.

Notation

a_{nonc}, c_{nonc} = constants at noncollocation point
$\mathbf{C}_n$ = function vector after discretization in the nth element
$\mathbf{D}$ = mapping matrix of initial conditions
$\mathbf{F}$ = function vector of differential states
F_m = mth function of $\mathbf{F}$
$\mathbf{G}$ = function vector of algebraic states
$\mathbf{G}_n$ = function vector of algebraic states in the nth element
$\mathbf{K}_n$ = Jacobi matrix of $\mathbf{C}_n$ with respect to $\mathbf{X}_{n,0}$ in the nth element
$\mathbf{L}_n$ = Jacobi matrix of $\mathbf{C}_n$ with respect to $\mathbf{X}_n$ in the nth element
$\mathbf{M}_n$ = Jacobi matrix of $\mathbf{C}_n$ with respect to $\mathbf{U}_n$ in the nth element
$\mathbf{N}_n$ = Jacobi matrix of $\mathbf{C}_n$ with respect to Δt_n in the nth element
NA = number of algebraic states
NC = number of collocation points
ND = number of differential states
NE = number of finite elements
$\mathrm{NL}_{\mathrm{AC}}$ = counterpart of NL during sub-simulation
NU = number of control variables
$\mathrm{R}^m_{n,i}$, $\mathrm{R}^m_{n,\text{nonc}}$ = residual (mth differential state; nth element; ith or noncollocation point)
$\mathbf{S}$ = counterpart of $\mathbf{X}$ during sub-simulation
$\mathbf{S}_j$ = counterpart of $\mathbf{X}_n$ during sub-simulation
$\mathbf{S}_{j,0}$ = counterpart of $\mathbf{X}_{n,0}$ during sub-simulation
t = time
t_f, t_n = final time or final time of the nth element
$t_{n,i}$, $t_{n,\text{nonc}}$ = ith or noncollocation point in the nth element
tol = tolerance of relative nonlinearity in the state profile
tol^m_i = ith tolerance of relative nonlinearity in the mth differential state
$\mathbf{u}(t)$ = vector of control profiles
$\mathbf{U}$ = vector of all discrete controls
$\mathbf{U}_n$ = vector of discrete controls in the nth element
$\mathbf{U}^{\mathrm{S}}_j$ = counterpart of $\mathbf{U}_n$ during sub-simulation
$\mathbf{U}_{\min}$, $\mathbf{U}_{\max}$ = vectors of path constraints on control profiles
$\mathbf{U}_{n,\text{nonc}}$ = vector of discrete controls (nth element; noncollocation point)
$\mathbf{X}$ = vector of all discrete differential and algebraic states
$\mathbf{X}_n$ = vector of all discrete differential and algebraic states in the nth element
$\mathbf{X}_{n,0}$ = vector of mapped initial conditions in the nth element
$\mathbf{X}_{\min}$, $\mathbf{X}_{\max}$ = vectors of path constraints on state profiles
$\mathbf{y}(t)$ = vector of algebraic state profiles
$\mathbf{Y}$ = vector of all discrete algebraic states
$\mathbf{Y}_{n,i}$,$\mathbf{Y}_{n,\text{nonc}}$ = vectors of algebraic states (nth element; ith or noncollocation point)
$\mathbf{z}_0$ = vector of initial conditions
$\mathbf{z}(t)$ = vector of differential state profiles
$z^m_{n,j}$ = mth differential state at the jth collocation point in the nth element
$z^m_n(t)$ = polynomial approximation of the mth differential state in the nth element
$\mathbf{Z}$ = vector of all discrete differential states
$\mathbf{Z}_{n,i}$, $\mathbf{Z}_{n,\text{nonc}}$ = vector of differential states (nth element; ith or noncollocation point)
$\mathbf{Z}_{\min}$, $\mathbf{Z}_{\max}$ = vectors of path constraints on differential state profiles
α = degree of nonlinearity in the state profile
α_{rel} = relative degree of nonlinearity in the state profile
$\alpha^m_{n,i}$ = ith criterion of the mth differential state in the nth element
Δt_n = length of the nth finite element
Δt^{S}_j = counterpart of Δt_n during sub-simulation
$\Delta\mathbf{T}$ = vector of all finite elements
$\Delta\mathbf{T}_{\min}$, $\Delta\mathbf{T}_{\max}$ = boundaries for all finite elements
$\varepsilon^m_{n,\text{nonc}}$ = approximation error of the mth differential state at noncollocation point in the nth element
φ = objective function
τ = normalized time
τ_{nonc} = normalized noncollocation point
Ψ_{rel} = operator for the evaluation of relative error estimation or time derivative analysis

Literature Cited

1. van Schijndel J, Pistikopoulos EN. Towards the Integration of Process Design, Process Control and Process Operability-Current Status and Future Trends. Foundations of Computer Aided Process Design '99. *AIChE Symp Ser*. 2000;323:99–112.

2. Biegler LT. An overview of simultaneous strategies for dynamic optimization. *Chem Eng Process*. 2007;46:1043–1053.
3. Hong W, Wang S, Li P, Wozny G, Biegler LT. A quasi-sequential approach to large-scale dynamic optimization problems. *AIChE J*. 2006;52:255–268.
4. Li P, Garcia HA, Wozny G, Reuter E. Optimization of a semibatch distillation process with model validation on the industrial site. *Ind Eng Chem Res*. 1998;37:1341–1350.
5. Li P, Wendt M, Arellano-Garcia H, Wozny G. Optimal operation of distillation processes under uncertain inflows accumulated in a feed tank. *AIChE J*. 2002;48:1198–1211.
6. Wendt M, Konigseder R, Li P, Wozny G. Theoretical and experimental studies on startup strategies for a heat-integrated distillation column system. *Chem Eng Res Des*. 2003;81:153–161.
7. Cervantes A, Biegler LT. Large-scale DAE optimization using a simultaneous NLP formulation. *AIChE J*. 1998;44:1038–1050.
8. Kameswaran S, Biegler LT. Simultaneous dynamic optimization strategies: recent advances and challenges. *Comput Chem Eng*. 2006;30:1560–1575.
9. Huntington GT, Rao AV. Comparison of global and local collocation methods for optimal control. *J Guid Control Dyn*. 2008;31:432–436.
10. Villadsen J, Stewart WE. Solution of boundary-value problems by orthogonal collocation. *Chem Eng Sci*. 1967;22:1483–1502.
11. Finlayson BA. *Nonlinear analysis in chemical engineering*. New York: McGraw-Hill, 1980.
12. Tieu D, Cluett WR, Penlidis A. A comparison of collocation methods for solving dynamic optimization problems. *Comput Chem Eng*. 1995;19:375–381.
13. Cuthrell JE, Biegler LT. On the optimization of differential-algebraic process systems. *AIChE J*. 1987;33:1257–1270.
14. De Boor C. Good approximation by splines with variable knots–II. *Lect Notes Math*. 1974;363:12–20.
15. De Boor C. *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.
16. Vasantharajan S, Biegler LT. Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria. *Comput Chem Eng*. 1990;14:1083–1100.
17. Riascos CAM, Pinto JM. Optimal control of bioreactors: a simultaneous approach for complex systems. *Chem Eng J*. 2004;99:23–34.
18. Renfro JG, Morshedi AM, Asbjornsen OA. Simultaneous optimization and solution of systems described by differential algebraic equations. *Comput Chem Eng*. 1987;11:503–517.
19. Tanartkit P, Biegler LT. A nested, simultaneous approach for dynamic optimization problems-I. *Comput Chem Eng*. 1996;21:735–741.
20. Tanartkit P, Biegler LT. A nested, simultaneous approach for dynamic optimization problems-II: the outer problem. *Comput Chem Eng*. 1997;21:1365–1388.
21. Biegler LT, Cervantes AM, Wachter A. Advances in simultaneous strategies for dynamic process optimization. *Chem Eng Sci*. 2002;57:575–593.
22. Lang YD, Biegler LT. A software environment for simultaneous dynamic optimization. *Comput Chem Eng*. 2007;31:931–942.
23. De Boor C, Swartz B. Collocation at Gaussian Points. *SIAM J Numer Anal*. 1973;10:582–606.
24. Seferlis P, Hrymak AN. Adaptive collocation on finite elements models for optimization of multistage distillation units. *Chem Eng Sci*. 1994;49:1369–1382.
25. Russell RD, Christiansen J. Adaptive Mesh Selection Strategies for Solving Boundary Value Problems. *SIAM J Numer Anal*. 1978;15:59–80.
26. Ascher U. Collocation for two-point boundary value problems revisited. *SIAM J Numer Anal*. 1986;23:596–609.
27. Gee DA, Ramirez WF. Optimal temperature control for batch beer fermentation. *Biotechnol Bioeng*. 1988;31:224–234.
28. Rump SM. *INTLAB-INTerval LABoratory*. In: Csendes T, editor. *Developments in Reliable Computing*. Dordrecht: Kluwer Academic Publishers, 1999:224–234.
29. Gill PE, Murray W, Saunders MA. SNOPT: an SQP algorithm for large-scale constrained optimization. *SIAM J Optim*. 2002;12:979–1006.

Appendix: Model of the Beer Fermentation Process

In the following we present more details on the example in the ''Dynamic Optimization Case Study'' section. The symbols of Eq. 46 have the following meaning:

$g(t)$	glucose concentration in mol m^{-3}
μ_i	specific rate of sugar uptake in h^{-1}
$m(t)$	maltose concentration in mol m^{-3}
Cp	mixture heat capacity in kJ kg^{-1} $°C^{-1}$
$n(t)$	maltotriose concentration in mol m^{-3}
ρ	mixture density in kg m^{-3}
temp(t)	temperature in °C
ΔH_{Fi}	heat of reaction in kJ mol^{-1}
xb(t)	biomass concentration in mol m^{-3}
$TEMP_C$	coolant temperature (here = 0°C)
e(t)	ethanol concentration in mol m^{-3}
R_{Xi}; R_{Ei}	specific stoichiometric yields

And the sugar uptake rates are defined by the following Michaelis terms

$$\mu_1(\mathrm{g}(t)) = \frac{\mathrm{V_g g}(t)}{\mathrm{K_g} + \mathrm{g}(t)}$$

$$\mu_2(m(t), \mathrm{g}(t)) = \frac{\mathrm{V}_m m(t)}{\mathrm{K}_m + m(t)} \frac{\mathrm{K'_g}}{\mathrm{K'_g} + \mathrm{g}(t)}$$

$$\mu_3(n(t), m(t), \mathrm{g}(t)) = \frac{\mathrm{V}_n n(t)}{\mathrm{K}_n + n(t)} \frac{\mathrm{K'_g}}{\mathrm{K'_g} + \mathrm{g}(t)} \frac{\mathrm{K'}_m}{\mathrm{K'}_m + m(t)}. \quad \text{(A1)}$$

The maximum rate V, Michealis constants K and inhibition constants K' depend on the temperature. This is modeled by the following Arrhenius temperature dependencies

$$\left.\begin{aligned} \mathrm{V}_i(t) &= \mathrm{V}_{i0} \exp[-E_{\mathrm{V}_i}/(\mathrm{R}(\mathrm{temp}(t) + 273.15))] \\ \mathrm{K}_i(t) &= k_{i0} \exp[-E_{\mathrm{K}_i}/(\mathrm{R}(\mathrm{temp}(t) + 273.15))] \\ \mathrm{K'}_i(t) &= k'_{i0} \exp[-E_{\mathrm{K'}_i}/(\mathrm{R}(\mathrm{temp}(t) + 273.15))] \end{aligned}\right\} \quad i = \mathrm{g}, m, n \quad \text{(A2)}$$

where

V_{i0}	Arrhenius frequency factor for maximum velocity in h^{-1}
k_{i0}; k_{i0}	Arrhenius frequency factors for Michaelis/inhibitation constants in mol m^{-3}
E_{Vi}	Arrhenius activation energy for maximum velocity kJ mol^{-1}
E_{Ki}; $E_{K'i}$	Arrhenius activation energy for Michaelis/inhibitation constants kJ mol^{-1}
R	gas constant in J mol^{-1} K^{-1}.

The model parameters are listed in Table A1. The reader is recommended to Gee and Ramirez[27] for a detailed derivation of the process model.

Table A1. Model Parameters of the Fermentation Process

Dimensionless		ΔH [kJ mol^{-1}]		E [kJ mol^{-1}]		Different
$R_{X_G} = 0.134$	$R_{E_G} = 1.92$	$\Delta H_{F_G} = -91.2$	$E_{V_G} = 94.6$	$E_{K_G} = -284.5$	$E_{K'_G} = 42.7$	$\rho = 1040$ kg m^{-3}
$R_{X_M} = 0.268$	$R_{E_M} = 3.84$	$\Delta H_{F_M} = -226.3$	$E_{V_M} = 47.3$	$E_{K_M} = -43.2$	$E_{K'_M} = 110$	$C_P = 4016$ J kg^{-1} K^{-1}
$R_{X_N} = 0.402$	$R_{E_N} = 5.76$	$\Delta H_{F_N} = -361.3$	$E_{V_N} = 31.8$	$E_{E_N} = -83.3$		R = 8.314 J mol^{-1} k^{-1}

Manuscript received Jun. 25, 2010, and revision received Sept. 8, 2010.