Dynamic Optimization under Uncertainty via NCO Tracking: A Solution Model Approach

B. Srinivasan and D. Bonvin
Laboratoire d'Automatique
École Polytechnique Fédérale de Lausanne
CH-1015 Lausanne, Switzerland

May 12, 2004

Abstract

The use of measurements to compensate the effect of uncertainty has recently gained attention in the context of optimization of dynamic systems. In this field, termed measurement-based optimization, two main categories can be distinguished depending on whether a model of the process or a model of the solution is used for processing the measurements. The former has been studied extensively in the literature, while tracking of the Necessary Conditions of Optimality (NCO tracking) recently proposed by the authors falls in the latter category. In this paper, the NCO-tracking scheme is viewed from the perspective of *a model of the solution*. The fixed part of the solution model is the one not affected by uncertainty, while the free part includes scalars and time functions that typically change with uncertainty. The variables of the free part are assigned to constraints and sensitivities, both during the run and at terminal time, resulting in a "color coding" of all input elements. Optimization is then performed by adjusting the free variables using measurements. NCO tracking using solution models is demonstrated on a simple illustrative reactor example.

Keywords: Dynamic optimization, Measurement-based optimization, On-line optimization, Run-to-run optimization, NCO tracking,

1 Introduction

Process optimization has received attention recently because, in the face of growing competition, it represents a natural choice for reducing production costs, improving product quality, meeting safety requirements and environmental regulations. The standard *nominal optimization* approach consists of determining numerically the optimal solution for a given process model. In practical situations, however, an accurate process model can rarely be found with affordable effort [3, 4]. The resulting modeling errors, together with process variations and disturbances, may lead to either infeasible operation in the presence of constraints or non-optimality [6, 22, 28]. Thus, in the presence of uncertainty, open-loop implementation of off-line calculated optimal inputs is clearly insufficient.

Two main classes of optimization methods are available for handling uncertainty. The essential difference relates to whether or not measurements are used in the calculation of the optimal strategy. In the absence of measurements, a *robust optimization* approach is typically used, whereby conservatism (and thus loss in performance) is introduced in order to guarantee feasibility for the whole range of expected variations [31, 25]. When measurements are available, a *measurement-based optimization* approach can help adapt to process changes and disturbances, thereby leading to less conservatism [29]. It is interesting to note that the above classification is similar to that found in control problems with the robust and adaptive approaches.

In the context of measurement-based optimization, one can distinguish between explicit and implicit schemes as illustrated in Figure 1 and discussed next:

- Explicit schemes involve two steps: (i) model update (estimation) – which consists of updating the current states (or initial conditions for the subsequent optimization) and, optionally, the parameters of the process model, and (ii) numerical optimization based on the updated process model. Since the two steps are repeated with the advent of new measurements, the procedure is also referred to as repeated optimization. These ideas have been widely discussed in the literature and used in the context of both static (Real Time Optimization - RTO [20, 36]) and dynamic (Model Predictive Control - MPC [9, 2]) optimizations.
- In implicit scheme, the measurements are used to update the inputs directly, i.e. without the intermediary of a process model. Here, optimality is achieved by meeting the necessary conditions of optimality (NCO), and are referred to as NCO-tracking schemes [29]. The structure of the NCO-tracking scheme is derived from the numerical optimization of a nominal model. Note that these schemes are not process model free; they simply do not use the process model in the feedback loop.

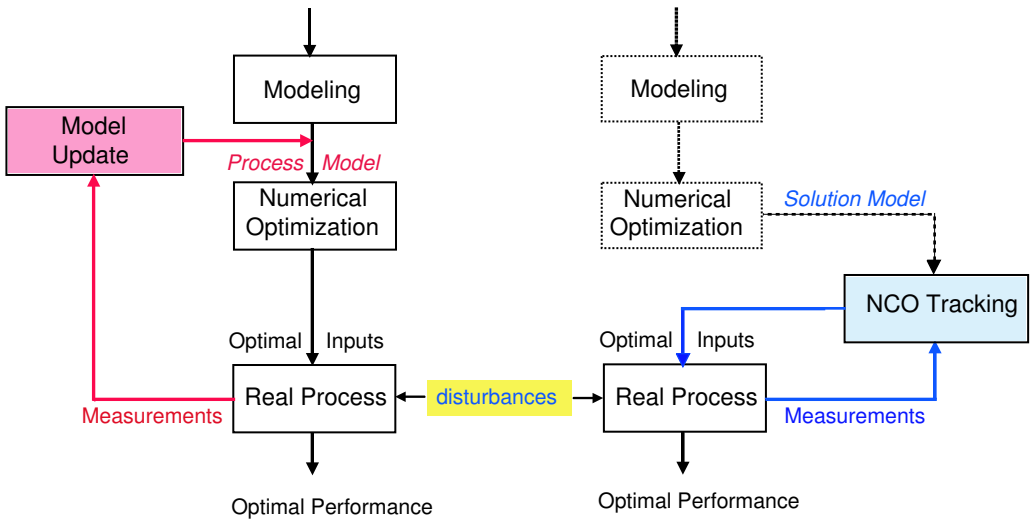

Figure 1: Measurement-based optimization: Explicit scheme where the measurements are used to update a process model before repeating the numerical optimization (left); implicit scheme where the measurements are used to update selected elements of a solution model (right)

To apply NCO tracking to a *dynamic optimization* problem, it is important to note that the solution of constrained terminal-time dynamic optimization problems are typically discontinuous

and consists of various arcs or intervals [17, 5]. Hence, the NCO include both path and terminal objectives, since there are conditions that have to be met during the operation while others need to be satisfied at final time. Also, optimality implies keeping certain constraints active and certain sensitivities at zero. Thus, enforcing the NCO in dynamic optimization problems corresponds to using control to meet four sets of conditions (a constraint and a sensitivity part both during the operation and at final time) using appropriate measurements. The relation between the NCO parts and various control schemes found in the literature are presented in Table 1.

	Path objectives	Terminal objectives
Constraints	Constraint control [19, 34]	Run-to-run constraint control [33, 30]
Sensitivities	Neighboring extremal control [24, 15]	Extremum-seeking control [18, 35] Self-optimizing control [27]

Table 1: Relation between the four NCO parts and various control schemes

Though path constraints can be met easily using *constraint control*, it is important to note that the constraints could be active only over a certain interval and, furthermore, the set of active constraints changes from one interval to the next. Path sensitivities are difficult to evaluate on-line without a reliable process model since they require information regarding the future. Thus, for pushing the path sensitivities to zero, one typically resorts to approximate methods such as *neighboring extremal control*. In many cases, meeting the terminal objectives is very important from a cost point of view. The active terminal constraints can be enforced using *run-to-run constraint control* [30] or on-line mid-course correction [33]. Finally, the terminal sensitivities can be pushed to zero using static optimization techniques such as *extremum-seeking control*, *self-optimizing control*, or evolutionary optimization. The ingenuity in extremum-seeking methods (mostly single input) lies in the way the gradient is computed or approximated. In self-optimizing controllers (typically multi inputs), outputs are chosen whose optimal values vary least with uncertainty. As far as adaptive extremum-seeking methods [35] are concerned, it is more appropriate to classify them under explicit schemes since they use a process model as an intermediary.

Individually, the four sub-problems of NCO tracking have been investigated quite extensively for both static and dynamic optimization scenarios. The novel contribution of this paper will be to consider all four sub-problems *simultaneously* for solving a constrained terminal-time dynamic optimization problem. The fact that the four sub-problems in Table 1 have very different adaptation strategies is the main difficulty, which calls for a dissectionist approach. Instead of being considered as a whole, the input profiles are dissected so that the various parts can be directly related to one of the four sub-problems in Table 1. This leads to the concept of solution model, whose main purpose is the assignment of the free variables to the various parts of the NCO.

NCO tracking using a solution model is the subject of this paper. In static optimization, one chooses a set of inputs to keep certain constraints active and the remaining inputs are adapted for optimizing the cost by pushing the sensitivities to zero (using e.g. extremum-seeking or self-optimizing techniques). The model of the solution is an extension of this choice for the dynamic case, where the input profiles are dissected into various parts, with certain of them attributed to the path constraints, others to the terminal constraints, and the rest to force the path and terminal sensitivities to zero. Also, the model of the solution may enforce a certain shape of the input profiles by parameterizing them in such a manner that the number of variables to be adapted is reduced. Clearly, as in the static case, the choice of the solution model is not unique, and this non-uniqueness can be exploited to ease the adaptation and improve the performance.

The paper is organized as follows. Section 2 introduces preliminary material related to optimization. The concept of solution model and its use in NCO tracking are presented in Section 3. Section 4 discusses the main differences between the use of a process model and that of a solution model. A simple reactor example is used throughout the paper to illustrate the concepts and the simulation results of this illustrative example are presented in Section 5. Finally, conclusions are drawn in Section 6.

2 Preliminaries

2.1 Formulation of the Dynamic Optimization Problem

Constrained dynamic optimization problems with finite operational time are considered in this paper, where the input profiles that optimize an objective function need to be determined. In addition to constraints corresponding to the dynamic system equations, there might be path constraints (involving inputs and states) as well as terminal constraints. Input constraints are dictated by actuator limitations, while state-dependent constraints typically result from safety and operability considerations. Terminal constraints normally arise from quality or performance considerations. These dynamic optimization problems can be formulated mathematically as follows [17, 5].

$$\min_{u(t),\, t_f} \; J = \phi(x(t_f), \theta, t_f) \tag{1}$$

$$s.t. \qquad \dot{x} = F(x, u, \theta), \quad x(0) = x_0 \tag{2}$$

$$S(x, u, \theta) \leq 0, \quad T(x(t_f), \theta) \leq 0 \tag{3}$$

where J is the scalar performance index to be minimized, x the n-dimensional vector of states with the known initial conditions x_0, u the m-dimensional vector of inputs, S the ζ-dimensional vector of path constraints, T the τ-dimensional vector of terminal constraints, F a smooth vector function, ϕ a smooth scalar function representing the terminal cost, θ the vector of uncertain parameters that include parametric uncertainty as well as exogeneous disturbances, and t_f the final time that is finite but can be either fixed or free (the more general case of a free final time is considered in (1)). The solution of Problem (1)-(3) is typically discontinuous and consists of several intervals.

Illustrative example:

A simple semi-batch reactor example will be considered to illustrate the concepts throughout the theoretical part.

- *Reaction system: $A + B \rightarrow C$, $2B \rightarrow D$, isothermal, exothermic reactions.*
- *Objective: Maximize the amount of C at a given final time.*
- *Manipulated input: Feed rate of B.*
- *Path constraints: Bounds on the input; bound on the cooling jacket temperature.*
- *Terminal constraints: Selectivity - bound on the amount of D at final time.*

Model equations:

$$\dot{c_A} = -k_1 c_A c_B - \frac{u}{V} c_A \qquad c_A(0) = c_{A_0} \tag{4}$$

$$\dot{c_B} = -k_1 c_A c_B - 2k_2 c_B^2 + \frac{u}{V}(c_{B_{in}} - c_B) \qquad c_B(0) = c_{B_0} \tag{5}$$

$$\dot{V} = u \qquad V(0) = V_0 \tag{6}$$

$$T_j = T_r - \frac{V}{UA}\left((-\Delta H_1)k_1 c_A c_B + (-\Delta H_2)k_2 c_B^2\right) \quad (7)$$

$$n_C = V_0 c_{A_0} - V c_A \quad (8)$$

$$n_D = \frac{1}{2}\left[V(c_A - c_B) + V_0(c_{B_0} - c_{A_0}) + c_{B_{in}}(V - V_0)\right] \quad (9)$$

Variables and parameters: *c_X: concentrations of species X, n_X: number of moles (amount) of species X, V: volume, k_i: kinetic coefficient of reaction i, u: feed rate of B, $c_{B_{in}}$: inlet concentration of B, ΔH_i: enthalpy of reaction i, T_r: reactor temperature, T_j: cooling jacket temperature, U: heat transfer coefficient, A: reactor heat exchange area.*

k_1	0.11	l/mol min	k_2	0.13	l/mol min
ΔH_1	-8×10^4	J/mol	ΔH_2	-10^5	J/mol
UA	1.25×10^4	J/min °C	$c_{B_{in}}$	5	mol/l
T_r	30	°C	$T_{j,min}$	10	°C
u_{max}	1	l/min	$n_{D_{f,max}}$	100	mol
c_{A_0}	0.5	mol/l	c_{B_0}	0	mol/l
V_0	1000	l	t_f	180	min

Table 2: Model parameters, operating bounds and initial conditions

Optimization problem:

$$\begin{aligned} \max_{u(t)} \quad & J = n_C(t_f) \quad (10) \\ s.t. \quad & \textit{dynamic system } (4)-(9) \\ & 0 \le u(t) \le u_{max}, \qquad T_j(t) \ge T_{j,min}, \qquad n_D(t_f) \le n_{D_{f,max}} \end{aligned}$$

Optimal solution: *The optimal input profile is depicted in Figure 2 and shows three intervals: (i) The input is initially at its upper bound u_{max}, (ii) the input u_{path} keeps the path constraint $T_j = T_{j,min}$ active, and (iii) the input u_{sens} seeks a compromise between producing the desired C and the undesired D. The switching times t_1 and t_2 are linked to path and terminal constraints.*

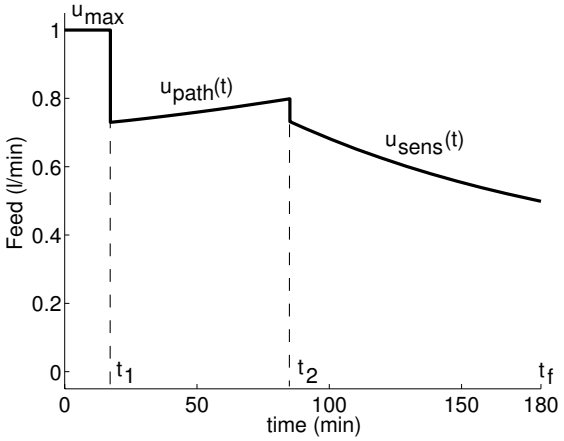

Figure 2: Optimal input consisting of three arcs

2.2 Necessary Conditions of Optimality

Using Pontryagin's Minimum Principle, the problem of minimizing the *scalar* cost functional J in (1)–(3) can be reformulated as that of minimizing the Hamiltonian *function* $H(t)$ as follows [23, 5]:

$$\min_{u(t),\, t_f} H(t) = \lambda^T F(x,u,\theta) + \mu^T S(x,u,\theta) \tag{11}$$

$$s.t. \quad \dot{x} = F(x,u,\theta), \quad x(0) = x_0 \tag{12}$$

$$\dot{\lambda}^T = -\frac{\partial H}{\partial x}, \quad \lambda^T(t_f) = \left.\frac{\partial \phi}{\partial x}\right|_{t_f} + \nu^T \left.\left(\frac{\partial T}{\partial x}\right)\right|_{t_f} \tag{13}$$

$$\mu^T S = 0, \qquad \nu^T T = 0 \tag{14}$$

where $\lambda(t) \neq 0$ is the n-dimensional vector of adjoint variables (Lagrange multipliers for the system equations), $\mu(t) \geq 0$ the ζ-dimensional vector of Lagrange multipliers for the path constraints, and $\nu \geq 0$ the τ-dimensional vector of Lagrange multipliers for the terminal constraints. The Lagrange multipliers μ and ν are nonzero when the corresponding constraints are active and zero otherwise so that $\mu^T S = 0$ and $\nu^T T = 0$ always. Note that $\mu \geq 0$, $S \leq 0$ and $\mu^T S = \sum_{j=1}^{\zeta} \mu_j S_j = 0$ imply that every term $\mu_j S_j$ of the summation has to be identically equal to zero. The same also holds for $\nu_i T_i$, $i = 1,...\tau$. The sensitivity part of the NCO are $H_u = \frac{\partial H}{\partial u} = 0$, which implies:

$$\frac{\partial H(t)}{\partial u} = \lambda^T \frac{\partial F}{\partial u} + \mu^T \frac{\partial S}{\partial u} = 0 \tag{15}$$

For a free-terminal-time problem, an additional condition, referred to as the transversality condition, needs to be satisfied:

$$H(t_f) + \left.\frac{\partial \phi}{\partial t}\right|_{t_f} = 0 \tag{16}$$

The NCO (14)-(16) can be rewritten in the partitioned form of Table 3 by separating:

- The conditions linked to the active constraints from those related to sensitivities (first and second rows in Table 3)
- The conditions linked to path objectives from those related to terminal objectives (first and second columns in Table 3)

	Path objectives	Terminal objectives
Constraints	$\mu^T S = 0$	$\nu^T T = 0$
Sensitivities	$H_u = 0$	$H(t_f) + \left.\frac{\partial \phi}{\partial t}\right\|_{t_f} = 0$

Table 3: Separation of the NCO into four distinct parts

NCO for the illustrative example (Table 4):

	Path objectives	*Terminal objectives*
Constraints	*Interval 1:* $u = u_{max}$ *Interval 2:* $T_j = T_{j,min}$	$n_D(t_f) = n_{D_{f,max}}$
Sensitivities	*Interval 3:* $H_u = 0$	-

Table 4: *NCO for the illustrative example*

3 NCO Tracking using a Solution Model

Optimality requires meeting the necessary conditions of optimality. Consequently, the idea of tracking the NCO, which allows treating the optimization problem as a control problem, is quite attractive. Furthermore, tracking the NCO using measurements ensures optimality for the *real process*, i.e. also in the presence of uncertainty (model mismatch, process variations and disturbances).

3.1 Concept of Solution Model

The real challenge in NCO tracking for determining optimal profiles lies in the fact that four different objectives are involved in achieving optimality, i.e. meeting constraint and sensitivity conditions both on-line and at final time as shown in Table 3. Hence, it becomes important to appropriately parameterize the inputs using time functions and scalars and assign them to the different tasks. This assignment, which corresponds to choosing the *solution model*, is a way of looking at the NCO through the inputs. The generation of a solution model includes two steps: (i) input dissection based on the effect of uncertainty and determination of the fixed and free variables of the inputs, and (ii) linking the input free variables to the NCO.

Input Dissection Based on the Effect of Uncertainty

As shown in Figure 1, the generation of a solution model typically starts with the numerical optimization of a (possibly incorrect) nominal process model. The resulting optimal solution is analyzed for several uncertainty realizations. For some of the intervals, the inputs are (or are assumed to be) independent of the prevailing uncertainty, e.g., intervals where the inputs are at their bounds, i.e. they can be applied in an open-loop fashion. These input elements can thus be considered fixed in the solution model. In other intervals, the inputs are affected by uncertainty and need to be adjusted for optimality. All the elements affected by uncertainty constitute the free variables of the optimization problem. Since these consist of time functions or arcs, $\eta(t)$, and scalars or parameters, π, the inputs can be written generically as $u = \mathcal{U}(x, \eta, \pi, t)$. Different parameterizations of the inputs are possible as will be discussed later.

Input dissection for the illustrative example:

Considering the optimal input profile in Figure 2, the input can be expressed as:

$$u = \begin{cases} u_{max} & \text{for} \quad t \leq \pi_1 \\ \eta_1(t) = u_{path}(t) & \text{for} \quad \pi_1 < t \leq \pi_2 \\ \eta_2(t) = u_{sens}(t) & \text{for} \quad \pi_2 < t \leq t_f \end{cases} \tag{17}$$

with the following fixed (known) and free (adjustable) parts:

- *Fixed part: Structure of the solution involving three arcs; u_{max} for the first arc.*
- *Free part: The arcs $\eta_1(t)$and $\eta_2(t)$ and the parameters $\pi_1 = t_1$ and $\pi_2 = t_2$.*

Assigning the Input Free Variables to the NCO parts

The next step is to provide an unambiguous link between the free variables of the inputs and the NCO. The active path constraints fix certain time functions and the active terminal constraints

certain scalar parameters. The remaining degrees of freedom are used to meet the path and terminal sensitivities. This procedure gives the following "color scheme" to the different parts of the inputs:

- *Blue variables:* These are parts that do not change with uncertainty such as the structure and open-loop arcs.
- *Violet variables:* Parts that change with uncertainty but can be adjusted by making the path constraints active.
- *Pink variables:* Parts that change with uncertainty but can be adjusted by making the terminal constraints active.
- *Red variables:* Parts that change with uncertainty and need sensitivity measurements for adaptation.

An important assumption for this "coloring scheme" is that the set of active constraints is correctly determined and does not vary with uncertainty. Fortunately, this restrictive assumption can often be relaxed by considering a super-structure for the constraints as will be discussed later.

The update of the violet and pink variables are based on path and terminal constraint measurements, respectively. On the other hand, the update equations for the red variables are based on sensitivity evaluation involving measurements as well. With $\bar{S}$ and $\bar{T}$ denoting the active path and terminal constraints, the adaptation laws are given by:

$$\eta_{blue}(t) = \eta_{blue}^{fixed}(t), \qquad \pi_{blue} = \pi_{blue}^{fixed} \tag{18}$$

$$\eta_{violet}(t) = \mathcal{K}_\eta(\bar{S}), \qquad \pi_{violet} = \mathcal{K}_\pi(\bar{S}) \tag{19}$$

$$\eta_{pink}(t) = \mathcal{R}_\eta(\bar{T}), \qquad \pi_{pink} = \mathcal{R}_\pi(\bar{T}) \tag{20}$$

$$\eta_{red}(t) = \mathcal{G}_\eta\left(\frac{\partial H}{\partial u}\frac{\partial \mathcal{U}}{\partial \eta_{red}}\right) \qquad \pi_{red}(t) = \mathcal{G}_\pi\left(\frac{\partial(\phi + \nu^T T)}{\partial \pi_{red}}\right) \tag{21}$$

where the superscript $(\cdot)^{fixed}$ is used to represent the fixed part of the inputs and $\mathcal{K}$, $\mathcal{R}$ and $\mathcal{G}$ are appropriate operators/controllers.

Assignment of the free variables for the illustrative example:

- *The open-loop arc u_{max} corresponds to a blue variable.*
- *The path constraint $T_j(t) = T_{j,min}$ fixes π_1 and η_1. Thus, π_1 and η_1 are violet variables.*
- *The terminal constraint $n_D(t_f) = n_{D_{f,max}}$ is used to determine π_2. Thus, π_2 is a pink variable.*
- *$\eta_2(t)$ is left undetermined by the constraints and corresponds to a red variable for the problem.*

*Formally, the solution model then reads (**Solution model A**):*

$$u = \begin{cases} u_{max} & \text{for} \quad t < \pi_1 \\ \eta_1(t) = \mathcal{K}_\eta(T_{j,min} - T_j(t)) & \text{for} \quad \pi_1 < t \leq \pi_2 \\ \eta_2(t) = \mathcal{G}_\eta\left(\frac{\partial H}{\partial u}\frac{\partial \mathcal{U}}{\partial \eta_2}\right) & \text{for} \quad \pi_2 < t \leq t_f \end{cases} \tag{22}$$

$$\pi_1 = t \quad \text{with} \quad T_j(t) = T_{j,min} \quad \text{and} \quad T_j(t_-) > T_{j,min}$$

$$\pi_2 = \mathcal{R}_\pi(n_{D_{f,max}} - n_D(t_f))$$

where t_- indicates a time just prior to t.

There is, of course, a strong analogy between the solution model (22) and the NCO given in Table 4. What is new in the solution model is the link between the NCO and the inputs, especially the information that the switching instant π_1 is determined implicitly upon reaching the path constraint, and that π_2 is linked to the terminal constraint. Note that, since the variations in the third arc $\eta_2(t)$ will also affect the terminal constraints, the choice between the pink and red variables in this example is not unique. It is also possible to formulate an alternative assignment where $\eta_2(t)$ is pink and π_2 red. This will be discussed later.

3.2 Implementation of the Solution Model

So far, nothing has been said regarding whether the update laws will be implemented on-line or on a run-to-run basis. Run-to-run adaptation has two main drawbacks: (i) it does not compensate within-run variations since only disturbances that are correlated over several runs can be rejected, and (ii) it requires multiple runs to be optimal. So, it is preferable, if possible, to do most of the adaptation on-line. However, the required information on the path sensitivities and terminal objectives is typically not available during the run. This necessitates information regarding the future, a task that requires a *reliable* process model, which was assumed to be unavailable in this study. This implies that full adaptation cannot, in general, be accomplished within a single run. Consequently, an important implementation aspect is to determine how the various operators in (22) (and with them also the color-coded variables) are adapted, i.e. on-line or on a run-to-run basis. As opposed to schemes available in the literature where everything is done either on-line [9, 26, 1] or on a run-to-run basis [10, 8, 7], the methodology proposed here allows a judicious mix of the two.

The violet variables are typically adapted on-line using measurements of path constraints. Regarding the pink variables, since the measurements of terminal constraints are only available at the end of the run, it is natural to adapt them on a run-to-run basis. However, if a prediction of the terminal constraints can be made using on-line measurements (e.g. using an empirical model such as Partial Least Squares [11, 12]), the pink variables can also be adapted in an on-line manner. On the other hand, for pushing the path sensitivities to zero on-line, one typically resorts to approximate methods such as neighboring extremal control. As an alternative, one could consider run-to-run implementation, where the sensitivities are obtained using the adjoint equations integrated backwards in time. Terminal sensitivities are typically computed experimentally using a finite-perturbation approach over several runs and forced to zero in a run-to-run manner.

Implementation scheme for Solution model A:

For the solution model (22), the constraint-seeking arc $\eta_1(t)$ and the sensitivity-seeking arc $\eta_2(t)$ can be implemented on-line via tracking of $T_{j,min}$ using a PI controller and a neighboring extremal controller, respectively, while the adaptation of π_2 is performed on a run-to-run basis via a discrete integral controller (Figure 3).

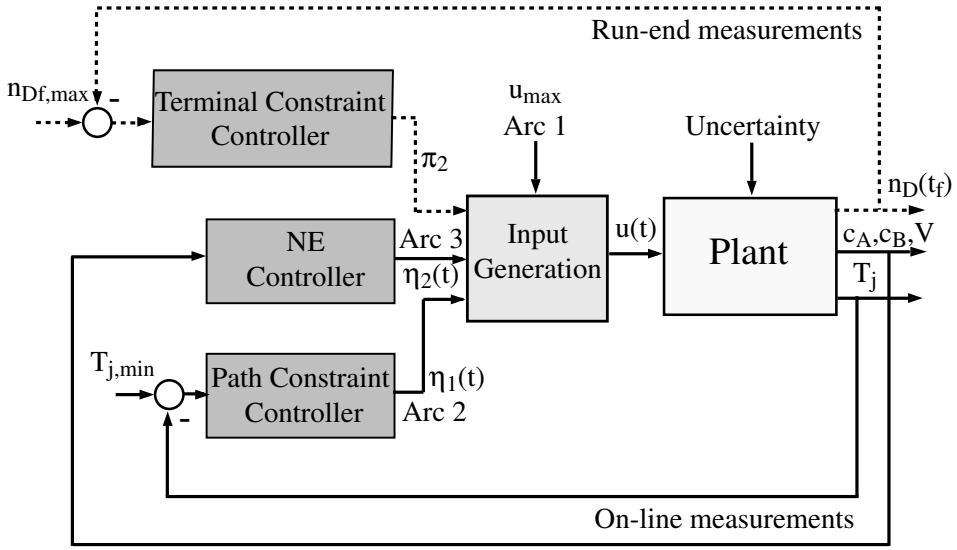

Figure 3: Implementation Scheme for Solution Model A

Using the run index k as a superscript, the adaptation laws read:

$$\begin{aligned}
\eta_1(t) = \eta_1^{nom}(t) &+ K_{Tj}(T_{j,min} - T_j(t)) + \frac{K_{Tj}}{T_{Tj}} \int_{t_1}^{t} (T_{j,min} - T_j(t))\, d\tau \qquad (23)\\
\eta_2(t) = \eta_2^{nom}(t) &+ K_A(t)(c_A^{nom}(t) - c_A(t))\\
&+ K_B(t)(c_B^{nom}(t) - c_B(t)) + K_V(t)(V^{nom}(t) - V(t))\\
\pi_2^{k+1} &= \pi_2^k + K_\pi(n_{D_{f,max}} - n_D^k(t_f))
\end{aligned}$$

where the superscript $(\cdot)^{nom}$ is used to represent the nominal optimal values, K_{Tj} and T_{Tj} are the gain and integral time constant of the PI-controller that keeps the path constraint active, $K_A(t)$, $K_B(t)$ and $K_C(t)$ are the time-varying gains of a neighboring-extremal controller designed off-line, and K_π is the gain matrix of the run-to-run controller.

Alternatively, if an empirical model can be built to predict $n_D(t_f)$ on-line, the controller $\mathcal{R}_\pi$ in (22) can be implemented on-line. On the other hand, $\eta_1(t)$ and $\eta_2(t)$ could also be implemented on a run-to-run basis using Iterative Learning Control [21]. The update equations would then be:

$$\begin{aligned}
\eta_1^{k+1}[t_1,t_2] = \eta_1^k[t_1,t_2] &+ K_{\eta_1}(T_{j,min} - T_j^k[t_1,t_2]) \qquad (24)\\
\eta_2^{k+1}[t_2,t_3] = \eta_2^k[t_2,t_3] &+ K_{\eta_2}\left(\frac{\partial H}{\partial u}\frac{\partial \mathcal{U}}{\partial \eta_2}\right)
\end{aligned}$$

where $\eta_i^k[t_a,t_b]$ represents the signal η_i in the run k for the time period between t_a and t_b, and K_{η_1} and K_{η_2} are the gains of the run-to-run controllers. $\frac{\partial H}{\partial u}$ is computed as in (15) using adjoints. The disadvantage with run-to-run adaptation is that it takes several runs to complete it. However, pushing the path sensitivities to zero using run-to-run adaptation is not approximate, as this is the case with the neighboring-extremal approach for on-line adaptation.

The design of path constraint controllers is a standard control design problem. Though simple PI-type controllers are sufficient in most practical situations, it might be necessary to have more sophisticated controllers in special cases. Regarding the terminal constraint controllers, the system dynamics in run time t are simply ignored and, in the run index k, the plant is seen by the controllers as a static nonlinear map with an implicit one-run delay. This special structure has been exploited to show that a simple integral controller is sufficient if the static map corresponds to a sector nonlinearity [14].

3.3 Approximation of Solution Model

Appropriate choice of the solution model is key in making the NCO-tracking problem tractable and efficient. Since there is some flexibility in specifying the solution model, simplifications and approximations can be brought in. For example, operator experience may suggest the qualitative shape that is needed to obtain the optimal solution for any practical purpose, i.e. without considering all the little arcs.

At the input dissection level, the most natural choice for $\eta(t)$ corresponds to all the input arcs that are not determined by input bounds, and for π, to all the switching times between intervals. However, this choice is by far not the only one, and parameterization of the input profiles can lead to major simplifications. In addition, there are many ways of assigning the free variables to the NCO, thereby resulting in different solution models.

Parameterization of Input Profiles

In many cases, it is easier to deal with scalars than time functions. A simple strategy is to directly parameterize the input arcs (time functions) using a finite number of parameters (e.g. piecewise-constant, piecewise-linear or exponential profiles). Such a parameterization is especially useful for sensitivity-seeking arcs, since path sensitivities are typically more difficult to evaluate than terminal sensitivities. This converts red time functions into red scalars.

Solution model with approximation of the sensitivity-seeking arc:

*The approximation $\eta_2(t) = \pi_3$ is used, and is considered as a red scalar (as opposed to η_2 which was a red time-function). The solution model then reads (**Solution model B**):*

$$
\begin{aligned}
u &= \begin{cases} u_{max} & \text{for} \quad t < \pi_1 \\ \eta_1(t) = \mathcal{K}_\eta(T_{j,min} - T_j(t)) & \text{for} \quad \pi_1 < t \le \pi_2 \\ \pi_3 = \mathcal{G}_\pi\left(\frac{\partial}{\partial \pi_3}\left(\phi + \nu(n_D(t_f) - n_{D_{f,max}})\right)\right) & \text{for} \quad \pi_2 < t \le t_f \end{cases} \\
\pi_1 &= t \quad \text{with} \quad T_j(t) = T_{j,min} \quad \text{and} \quad T_j(t_-) > T_{j,min} \\
\pi_2 &= \mathcal{R}_\pi\left(n_{D_{f,max}} - n_D(t_f)\right)
\end{aligned}
\tag{25}
$$

Here, $\frac{\partial(\phi+\nu^T T)}{\partial \pi_3}$ can be interpreted as the constrained derivative $\frac{\partial \phi}{\partial \pi_3}\Big|_{\bar{T}=0}$. Hence, the gradient can be computed using finite difference once the terminal constraints are pushed back to zero for every variation of π_3.

The model of the solution enforces some pre-specified pairing between free variables and constraints. In addition, it is also possible to use sensitivity techniques for decoupling [13]. This does not change the final result, but could improve the convergence properties. In this example, it is possible to choose any combination of π_2 and π_3 to address the terminal constraint and the other combination to deal with the terminal sensitivity. A good choice for sensitivity adaptation is $\tilde{\pi} = \alpha_2\pi_2 + \alpha_3\pi_3$, with $\frac{\partial \bar{T}}{\partial \tilde{\pi}} = 0$. The advantage of this choice is that it does not require the Lagrange multipliers ν or, in other words, does not necessitate explicit pushing of the terminal constraints to zero [13].

Parameterization of Selected State Profiles

For determining the optimal inputs, it is not necessary to parameterize the input profiles directly. In some cases, it is considerably easier to parameterize selected state variables and perform the optimization with the resulting degrees of freedom. The input profiles are then determined by approximate inversion (via feedback) of the dynamic relationship between the inputs and the selected state variables. There are many reasons for considering state variables for parameterization and subsequent trajectory following, of which two are given next:

- *Ease of parameterization:* It might be easier and more intuitive to specify the qualitative profile of a state variable. For example, in batch or semi-batch reactors, it is easier to specify the qualitative profile for the reactor temperature T_r than for the inlet jacket temperature $T_{j,in}$, the actual input variable. In such a case, T_r is parameterized and the actual input $T_{j,in}(t)$ obtained via the tracking of $T_r(t)$ using feedback control.
- *Terminal constraints:* If a state variable is constrained at final time, it can be parameterized and the optimization performed by adjusting a free variable such that its value at final time meets the constraint [32]. Then, adjusting the inputs to follow this state trajectory guarantees that the terminal constraint is met within each run. This is desirable in many cases since meeting the active terminal constraints can lead to significant improvement in cost.

Solution model with trajectory following:

In the illustrative example, $\eta_2(t)$ can be used to meet the terminal constraint $n_D(t_f) = n_{D_{f,max}}$ on-line by tracking a desired n_D-trajectory that ends up at $n_D(t_f) = n_{D_{f,max}}$. Here, the trajectory is chosen linear as follows:

$$n_{D_{traj}}(t) = n_D(\pi_2) + \left(\frac{t-\pi_2}{t_f-\pi_2}\right)\left(n_{D_{f,max}} - n_D(\pi_2)\right) \quad \text{for } \pi_2 < t \leq t_f \tag{26}$$

*The resulting solution model then reads (**Solution model C**):*

$$\begin{aligned} u &= \begin{cases} u_{max} & \text{for} \quad t < \pi_1 \\ \eta_1(t) = \mathcal{K}_\eta(T_{j,min} - T_j(t)) & \text{for} \quad \pi_1 < t \leq \pi_2 \\ \eta_2(t) = \mathcal{T}[n_{D_{traj}}(t) - n_D(t)] & \text{for} \quad \pi_2 < t \leq t_f \end{cases} \\ \pi_1 &= t \quad \text{with} \quad T_j(t) = T_{j,min} \quad \text{and} \quad T_j(t_-) > T_{j,min} \\ \pi_2 &= \mathcal{G}_\pi\left(\frac{\partial}{\partial \pi_2}\left(\phi + \nu(n_D(t_f) - n_{D_{f,max}})\right)\right) \end{aligned} \tag{27}$$

where $\mathcal{T}$ represents an appropriate operator. Note that as opposed to solution model A, π_2 has become a red variable and η_2 pink.

4 Comparison of Process- and Solution-model Approaches

Model building: The process and solution models are built in almost the same manner. The procedure starts with some assumptions regarding the process or the signals; it then proposes a model structure which contains both fixed and free parts. The fixed part is known and does not vary with uncertainty, while the uncertain part is expressed in terms of free variables (scalars in the process model case, and time functions and scalars for the solution model). The validity of the assumptions and the appropriateness of the proposed structure will affect the accuracy of the resulting model. However, even if the chosen structure is not able to represent the reality exactly, it might be sufficient for any practical purpose since the unmodeled part constitutes the modeling error that will be kept small through adjustment of the free parameters.

The key difference between the process and solution models is in the use of measurements. In the process model, the measurements are external in the sense that the process model is valid even without measurements. In contrast, the free variables of the inputs (time functions and parameters) are intimately linked to the measurements to form the solution model. Hence, a solution model not only has free variables that change with uncertainty, but it also incorporates the link from the measurements to the free variables.

Model validation: As with process models, the solution models will also need to be validated. The fit criterion is the cost function to be minimized. The more parameters, the lower the cost function, and the cost function as a function of the number of parameters gives the traditional L-curve [16]. The number of parameters in the model can be chosen at the knee of the L-curve. Similarly to a process model being validated using new experimental data, a solution model can be validated in simulation using different realizations of the uncertainty.

Model complexity: The main reason for using a solution model rather than a process model is that, in many cases, an approximate solution model is considerably simpler in terms of the number of parameters to adapt. For example, the optimal solution of systems described by hundreds of differential-algebraic equations can often be approximated and parameterized using only a few parameters, the adjustment of which is relatively straightforward using measurements.

Model invariance: It is important to note that the solution model varies not only with the process to be optimized but also with the formulation of the optimization problem. For a given process, if a different optimization problem is considered, i.e. with a different cost function, a different set of constraints or operating conditions, then the solution model can be entirely different. A typical example is the grade change problem, where going from grade A to grade B and from grade B to grade A require completely different solution models.

Inversion and disturbance rejection: A process is typically driven by inputs and generates outputs and a corresponding performance. The control and optimization problems aim at determining the inputs to apply to produce the desired outputs or maximize performance. Hence, these two problems are indeed inverse problems.

But system inversion leads to robustness problems and poor disturbance rejection. Hence, a robust controller tends to only approximately invert the system in order to provide robustness and acceptable disturbance rejection. The same philosophy also applies to the optimization framework, i.e. only approximate system inversion is sought in the presence of constraints, uncertainty and distur-

bances. This is precisely what the solution model approach does, whereby the path and terminal controllers are tuned by considering not only the tracking performance, but also the disturbance characteristics. Contrarily, the process model approach tends to perform inversion.

Change in constraint structure: One of the main advantages of the solution model is its dissectionist's approach to the problem and the considerable insight it generates. However, since it does not have the global picture, it cannot easily cope with changing active constraints. It is worth mentioning here that, when a system is subject to several constraints, only the most restrictive one will typically be active over a particular interval. Hence, it is sufficient for implementation purposes to specify only the super-set of constraints with the indication that the most restrictive one needs to be active.

On the other hand, since the standard repeated-optimization approach based on a process model does not require assumptions regarding the general features of the solution and the solution structure is determined at every stage by the re-optimization, it can handle changes in the active set of constraints. Thus, a combination of the two could be envisaged to get the better of both worlds.

5 Simulation Results for the Illustrative Example

Simulation results for the illustrative example considered throughout this paper are presented in this section. It is assumed that uncertainty is present in the form of time-varying kinetic coefficients k_1 and k_2:

$$k_1(t) = k_1 \frac{\alpha}{\alpha + t}, \qquad k_2(t) = k_2 \frac{\beta}{\alpha - t} \tag{28}$$

with $\alpha = 1800$ min and $\beta = 900$ min. This variation might, for example, correspond to the variation of catalyst efficiency with time. This information on the variation is, of course, not revealed to the measurement-based optimization algorithms. If this information had been available, the ideal value for the cost function would have been $J^* = 392.3$ mol of desired product C.

The approaches based on the process and solution models are compared. In the process-model approach, two cases are distinguished depending on whether or not the process model is re-identified. In the solution-model approach, the three solution models presented in the earlier sections (solution models A, B, and C) are considered.

5.1 Repeated Optimization - Approach Based on Process Model

In the first case, where the parameters are not identified on-line, the current state information is used as initial conditions for the re-optimization every 10, 30 and 60 min using the nominal process model. In the second case, the model parameters k_1 and k_2 are updated using standard least squares, assuming that the entire state information is available every 5 min. Then, re-optimization is performed with the refined model every 10, 30 and 60 min, using the current state information as initial conditions. The repeated optimization results are given in Table 5, and the solution corresponding to the re-optimization period 10 min is shown in Figure 4.

Two remarks are in order. First, it can be seen from Figure 4 that the input with the process-

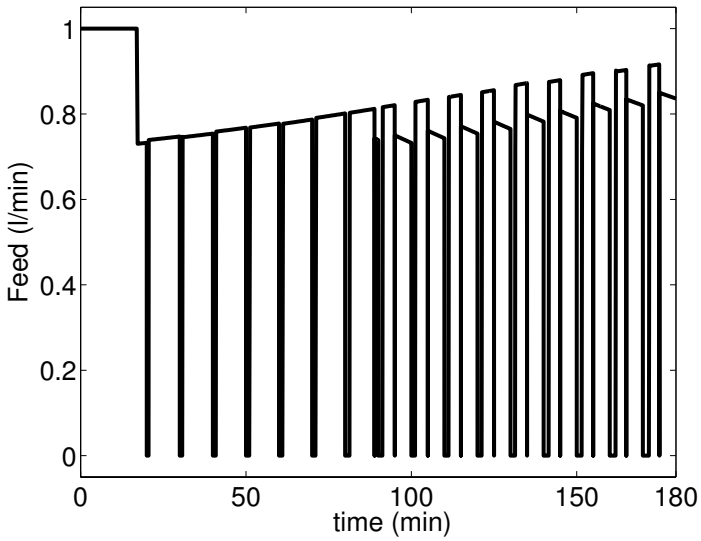

Figure 4: Input profile for re-optimization every 10 min

	Re-optimization period (min)	Minimum jacket temperature ($°C$) $T_{j,min} = 10\ °C$	Final amount of D (mol) $n_{D_{f,max}} = 100$ mol	Final amount of C (mol) Cost J
Open-loop	∞	10.1	77.5	374.6
Update of	60	10.1	84.9	380.4
only initial	30	10.1	85.8	380.4
conditions	10	10.6	79.7	374.6
Update of	60	10.1	88.8	383.7
initial conditions	30	10.0	77.1	372.8
and parameters	10	10.1	87.4	382.1

Table 5: Cost and constrained quantities for various repeated optimization periods (the cost values need to be compared to $J^* = 392.3$ mol)

model approach exhibits chattering-type behavior. Secondly, since the actual values of the kinetic parameters are smaller than their nominal ones, more B is present in the reactor. The prediction with the nominal model (which has larger values for k_1 and k_2) sees it as a potential threat (both in terms of heat production and selectivity) and stops the feed for some time every time re-optimization is done. However, in reality, more B could be added since k_1 and k_2 are smaller. This conflict gives rise to the fact that increasing the re-optimization frequency does not necessarily lead to a better cost function. The situation does not improve with parameter identification either, since the optimal input is not sufficiently exciting, especially in the first part of the reaction.

5.2 NCO Tracking Based on Various Solution Models

The input profiles with NCO tracking are presented in Figure 5, where it can be seen that the profiles are fairly smooth. Though the final solutions are quite different, the optimal costs are nearly the same since the switching time and the value of the input in the final interval compensate each other.

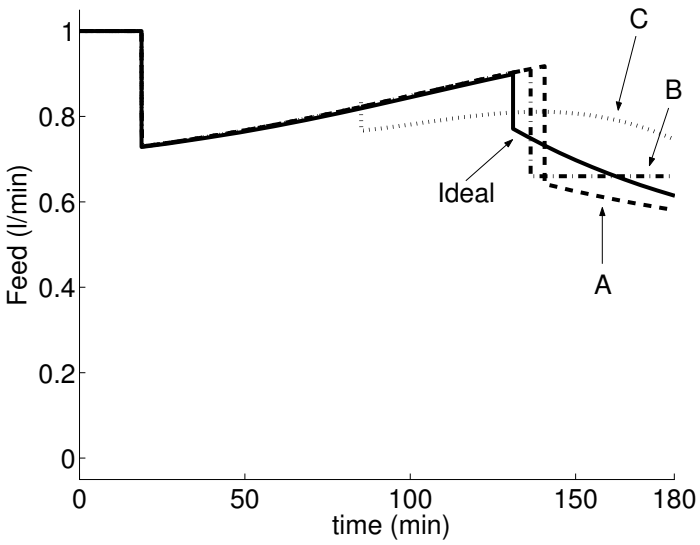

Figure 5: Input profiles: Ideal (solid) and after the 5th run with NCO tracking using various solution models (A - dashed, B - dash-dotted, C - dotted).

The results are compared in Table 6 where, in all cases, convergence is achieved in maximum 10 runs. The constraint on the jacket temperature is active in all cases. The first runs of Models A and B indicate how much can be gained by meeting the path constraint alone. It can be seen that, in this case, enforcing the path sensitivity using neighboring extremals (Model A) is not really better than a brute-force approximation by a constant (Model B), which, of course, depends on the initial value of the constant approximation. Over the runs, the terminal constraint gets active and improves the cost. The behavior of Models A and B are quite similar, though it is important to note that Model B is much easier to implement in terms of off-line calculations and measurement requirements.

In Model C, the path and terminal constraints are met right from the first run and the cost is nearly optimal. The initial switching time was chosen as its nominal value and is supposed to be adapted using terminal sensitivities. However, in this example, the terminal sensitivities are so low that no adaptation took place. The same occurs with the value of the constant approximation in Model B. So, even if the adaptations based on terminal sensitivities were switched off in Models B and C, the results would be exactly the same.

6 Conclusions

This paper presents the concept of solution model and its use for tracking the necessary conditions of optimality in the context of measurement-based optimization in the presence of uncertainty. The procedure for obtaining the solution model consists of dissecting the input profiles and relating their elements to the different parts of the NCO. It was argued that the use of a solution model could simplify the implementation of optimal operation as opposed to re-optimization using a process model. The pros and cons of either approaches have been discussed, and a judicious combination of the two could be used depending upon the application at hand.

Solution model	Run index	Minimum jacket temperature ($°C$) $T_{j,min} = 10\ °C$	Final amount of D (mol) $n_{D_{f,max}} = 100$ mol	Final amount of C (mol) Cost
A	1	10.0	79.6	376.8
	5	10.0	96.2	389.8
	10	10.0	99.6	392.1
B	1	10.0	83.6	383.3
	5	10.0	98.1	391.6
	10	10.0	99.9	392.2
C	1	10.0	100	391.8
	5	10.0	100	391.8
	10	10.0	100	391.8

Table 6: Cost and constrained quantities with NCO tracking for various numbers of runs and using 3 different solution models (the cost values need to be compared to $J^* = 392.3$ mol)

With the solution-model approach, the framework of measurement-based optimization can be strongly linked to the domains of control and identification. Thus, the tools therein could be used to provide the mathematical framework necessary for analysis and design. Also, it was shown that the solution model for a given problem is not unique, and the particular choice has a strong influence on the adaptation strategy. This part needs to be formalized and understood more precisely, and systematic procedures for obtaining better solution models are clearly needed.

Acknowledgments

The authors acknowledge the help of Mr. L. Bodizs, Laboratoire d'Automatique, École Polytechnique Fédérale de Lausanne, Switzerland, in generating the simulation results of this paper.

References

[1] O. Abel, A. Helbig, W. Marquardt, H. Zwick, and T. Daszkowski. Productivity optimization of an industrial semi-batch polymerization reactor under safety constraints. *J. Process Contr.*, 10(4):351–362, 2000.

[2] O. Abel and W. Marquardt. A model predictive control scheme for safe and optimal operation of exothermic semi-batch reactors. In *IFAC DYCOPS-5*, pages 761–766, Corfu, Greece, 1998.

[3] D. Bonvin. Optimal operation of batch reactors - A personal view. *J. Process Contr.*, 8(5–6):355–368, 1998.

[4] D. Bonvin, B. Srinivasan, and D. Ruppen. Dynamic optimization in the batch chemical industry. In *Proceedings of the Chemical Process Control - VI Conference, American Institute of Chemical Engineers, Symposium Series, 326*, volume 98, pages 255–273, 2002.

[5] A. E. Bryson and Y. C. Ho. *Applied Optimal Control*. Hemisphere, Washington DC, 1975.

[6] W. B. Z. Chu and A. Constantinides. Modeling, optimization, and computer control of cephalosporin C fermentation process. *Biotechnol. Bioengng.*, 32:277–288, 1988.

[7] T. L. Clarke-Pringle and J. F. MacGregor. Optimization of molecular weight distribution using batch-to-batch adjustments. *Ind. Eng. Chem. Res.*, 37:3660–3669, 1998.

[8] D. Dong, T. J. McAvoy, and E. Zafiriou. Batch-to-batch optimization using neural networks. *Ind. Eng. Chem. Res.*, 35:2269–2276, 1996.

[9] J. W. Eaton and J. B. Rawlings. Feedback control of nonlinear processes using on-line optimization techniques. *Comp. Chem. Eng.*, 14:469–479, 1990.

[10] C. Filippi-Bossy, J. Bordet, J. Villermaux, S. Marchal-Brassely, and C. Georgakis. Batch reactor optimization by use of tendency models. *Comp. Chem. Eng.*, 13:35–47, 1989.

[11] J. Flores-Cerrillo and J. F. MacGregor. Within-batch and batch-to-batch inferential-adaptive control of semibatch reactors: A partial least squares approach. *Ind. Eng. Chem. Res.*, 42:3334–3335, 2003.

[12] J. Flores-Cerrillo and J. F. MacGregor. Control of batch product quality by trajectory manipulation using latent variable models. *J. Process Contr.*, 14:539–553, 2004.

[13] G. François, B. Srinivasan, and D. Bonvin. Run-to-run optimization of batch emulsion polymerization. In *15th IFAC World Congress*, page 1258, Barcelona, Spain, 2002.

[14] G. François, B. Srinivasan, and D. Bonvin. Convergence analysis of run-to-run control for a class of nonlinear systems. In *American Control Conference*, page Accepted, Denver, Colorado, 2003.

[15] S. Gros, B. Srinivasan, and D. Bonvin. Robust predictive control based on neighboring extremals. In *IFAC DYCOPS-7*, page Accepted, Boston, MA, 2004.

[16] P. C. Hansen. *Rank-deficient and Discrete Ill-posed Problems*. Monographs on Mathematical Modeling and Computation. Philadelphia, 1998.

[17] D. E. Kirk. *Optimal Control Theory : An Introduction*. Prentice-Hall, London, 1970.

[18] M. Kristic and H-H. Wang. Stability of extremum seeking feedback for general nonlinear dynamic systems. *Automatica*, 36:595–601, 2000.

[19] A. Maarleveld and J. E. Rijnsdorp. Constraint control of distillation columns. *Automatica*, 6:51–58, 1970.

[20] T. Marlin and A. N. Hrymak. Real-time operations optimization of continuous processes. In *Chemical Process Control-V Conference*, Tahoe City, Nevada, 1996.

[21] K. L. Moore. *Iterative Learning Control for Deterministic Systems*. Springer-Verlag, Advances in Industrial Control, London, 1993.

[22] S. R. Ponnuswamy, S. L. Shah, and C. A. Kiparissides. Computer optimal control of batch polymerization reactors. *Ind. Eng. Chem. Res.*, 26:2229–2236, 1987.

[23] L. S. Pontryagin, V. G. Boltyanskil, R. V. Gamkrelidge, and E. F. Mishchenko. *The Mathematical Theory of Optimal Processes*. Interscience, New York, 1962.

[24] E. Ronco, B. Srinivasan, J. Y. Favez, and D. Bonvin. Predictive control with added feedback for fast nonlinear systems. In *European Control Conference*, pages 3167–3172, Porto, Portugal, 2001.

[25] D. Ruppen, C. Benthack, and D. Bonvin. Optimization of batch reactor operation under parametric uncertainty - Computational aspects. *J. Process Contr.*, 5(4):235–240, 1995.

[26] D. Ruppen, D. Bonvin, and D. W. T. Rippin. Implementation of adaptive optimal operation for a semi-batch reaction system. *Comp. Chem. Eng.*, 22:185–189, 1998.

[27] S. Skogestad. Plantwide control: The search for the self-optimizing control structure. *J. Process Contr.*, 10:487–507, 2000.

[28] M. Soroush and S. Valluri. An approach to optimization and control of batch processes. In *American Control Conference*, pages 490–494, Baltimore, MD, 1994.

[29] B. Srinivasan, D. Bonvin, E. Visser, and S. Palanki. Dynamic optimization of batch processes: II. Role of measurements in handling uncertainty. *Comp. Chem. Eng.*, 44:27–44, 2003.

[30] B. Srinivasan, C. J Primus, D. Bonvin, and N. L. Ricker. Run-to-run optimization via generalized constraint control. *Control Eng. Practice*, 9:911–919, 2001.

[31] P. Terwiesch, M. Agarwal, and D. W. T. Rippin. Batch unit optimization with imperfect modeling - A survey. *J. Process Contr.*, 4:238–258, 1994.

[32] C. Welz, B. Srinivasan, and D. Bonvin. Combined on-line and run-to-run optimization of batch processes with terminal constraints. In *IFAC ADCHEM'03*, pages 55–62, Hong-Kong, China, 2004.

[33] Y. Yabuki and J. F. MacGregor. Product quality control in semi-batch reactors using mid-course correction policies. In *IFAC ADCHEM'97*, pages 189–194, Banff, Canada, 1997.

[34] S. H. Yang, P. W. H. Chung, X. Z. Wang, and C. H. McGreavy. Multi-objective constraint control for fcc reactor-regenerator system. *Comp. Chem. Eng.*, 22:S831–834, 1998.

[35] T. Zang, M. Guay, and D. Dochain. Adaptive extremum seeking control of continuous stirred tank bioreactors. *AIChE J.*, 40(2):10–20, 2001.

[36] Y. Zhang, D. Monder, and J. F. Forbes. Real-time optimization under parametric uncertainty: A probability constrained approach. *J. Process Contr.*, 12(3):373–389, 2002.