Modifier Adaptation for Run-to-Run Optimization of Transient Processes

Sean Costello, * **Grégory François,** * **Bala Srinivasan,** ** **Dominique Bonvin** *

* *Laboratoire d'Automatique, Ecole Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland (e-mail: dominique.bonvin@epfl.ch).*
** *Department of Chemical Engineering, Ecole Polytechnique de Montreal, Canada, H3C 3A7 (e-mail: bala.srinivasan@polymtl.ca).*

Abstract: Dynamic optimization can be used to determine optimal input profiles for dynamic processes. Due to plant-model mismatch and disturbances, the optimal inputs determined through model-based optimization will, in general, not be optimal for the plant. Modifier adaptation is a methodology that uses measurements to achieve optimality in the presence of uncertainty. Modifier-adaptation schemes have been developed for the real-time optimization of plants operating at steady state. In this paper, the concept of modifier adaptation is extended to transient plants such as batch processes. Two different schemes are proposed, and their performance is illustrated via the simulation of a semi-batch reaction system.

Keywords: Dynamic optimization, Real-time optimization, Run-to-run optimization, Modifier adaptation, Batch processes.

1. INTRODUCTION

Dynamic optimization provides a framework for increasing the productivity of dynamic processes, without violating safety and quality constraints. The problem is that most optimization techniques are model-based, and reliable models are rarely available at the industrial level. This is particularly true for batch processes, where the effort required for accurate modeling is often incompatible with short product life-time and the necessity for fast time-to-market (Bonvin (1998)). The reactions in batch processes are often more complex than those encountered in continuous processes, thus making the development of detailed models an extremely challenging task (Filippi-Bossy et al. (1989)). In addition to modeling uncertainty, disturbances occur due to variations in the initial conditions and the operating conditions. Luckily, batch processes are usually repeated, and often measured quantities are available at the end of each batch. Run-to-run measurement-based optimization uses these measurements to reduce the uncertainty and recover optimal operation for future batches.

Measurement-based optimization techniques for transient processes can be separated into two categories, namely fixed-rule methods and repeated-optimization methods. The distinction is based upon how the available measurements are used. Fixed-rule methods try to improve the input profiles without solving an optimization problem between batches. NCO tracking (François et al. (2005)), interpolation-based algorithms (Krothapally et al. (1999)), and measurement-based gradient search methods (Ge et al. (2000), Zafiriou and Zhu (1990)) fall into this category. Repeated-optimization methods, on the other hand, modify the optimization problem and recompute the input profiles. Typically the so-called 'two-step' approach is used. The available measurements are used in a first step to re-identify the dynamic model at the end of each batch, and this refined model is used in a subsequent step to optimize for the next batch. An exhaustive review of measurement-based optimization techniques, including run-to-run schemes, is given in Srinivasan et al. (2003). Although recent publications elaborate on some aspects of these techniques (Kadam et al. (2007)), to the best of the authors' knowledge, no fundamentally different approaches have since been published.

The two-step repeated-optimization approach has the advantage of being particularly transparent, and as such it is more easily accepted in industry. It has been well documented in the literature (Clarke-Pringle and Mac Gregor (1998), Cruse et al. (2006)). It suffers, however, from two main drawbacks: (i) The inputs determined through optimization may not provide sufficient excitation to identify the model parameters, and (ii) in the case of structural plant-model mismatch, it is very unlikely that plant optimality can be achieved (Forbes et al. (1994)).

The same problems occur when continuous processes are optimized in real-time, which is static optimization. Extensive work has shown that, rather than trying to identify the plant, it often makes more sense to directly modify the cost and the constraints of the optimization problem (Roberts (1979), Gao and Engell (2004), Marchetti et al. (2010)). In the literature, this is often referred to as *modifier adaptation*. However, when the process is transient in nature, a level of complexity is added to the problem. This paper extends the modifier-adaptation concept to transient (batch) processes. A first attempt was made by Marchetti et al. (2007) using a continuous-time formulation and correction terms on the constraint functions.

In contrast, this work uses a discrete-time formulation. In this context, two approaches are possible. The correction terms may be placed on either the dynamic equations or the cost and constraints. Either way, the cost and the constraints of the optimization problem are modified, whether it is indirectly or directly. It is shown that the two methods share many similarities, yet differ considerably regarding the measurements required for their implementation. The first method requires that the constraints and the cost be measured, while the second method assumes full state measurements (or estimation). The merits of each method are discussed, although the question of which one is more useful is left open.

The paper is organized as follows. Section 2 formulates the problem and presents two schemes for its solving it. The performance of these schemes is illustrated in Section 3 via the simulation of a semi-batch reaction system. Conclusions are drawn in Section 4.

2. RUN-TO-RUN OPTIMIZATION WITH MODIFIER ADAPTATION

2.1 Problem statement

Using a discrete-time formulation, the optimal input profiles for a batch process are obtained by solving the following *model-based* dynamic optimization problem:

$$\begin{aligned} \boldsymbol{U}^* = \underset{\boldsymbol{U}}{\text{argmin}} \quad & \phi\left(\boldsymbol{x}[f]\right) \\ s.t. \quad & \boldsymbol{x}[j+1] = \boldsymbol{F}(\boldsymbol{x}[j], \boldsymbol{u}[j]), \quad \boldsymbol{x}[0] = \boldsymbol{x}_0, \\ & \boldsymbol{g}(\boldsymbol{X}, \boldsymbol{U}) \leq \boldsymbol{0}, \end{aligned} \tag{1}$$

where the superscript $(\cdot)^*$ indicates a quantity calculated via optimization, $\boldsymbol{x}$ is the n_x-dimensional vector of states, $\boldsymbol{u}$ is the n_u-dimensional vector of inputs, ϕ is the terminal cost, $\boldsymbol{F}$ is the discrete dynamic model, $\boldsymbol{g}$ is a vector of constraints, and f is the fixed final sampling instant. Let us denote $\boldsymbol{U} = \left[\boldsymbol{u}[0]^T, \ldots, \boldsymbol{u}[f-1]^T\right]^T$, $\boldsymbol{X} = \left[\boldsymbol{x}[0]^T, \ldots, \boldsymbol{x}[f]^T\right]^T$.

The difficulty is that $\boldsymbol{U}^*$ determined this way is optimal for the model, and not necessarily for the plant. In practice, if $\boldsymbol{U}^*$ is applied to the plant, the resulting performance will probably be sub-optimal and may even be infeasible. This is because the plant behaves differently to the model, i.e. $\boldsymbol{F}_p \neq \boldsymbol{F}$, where the subscript $(\cdot)_p$ denotes a quantity related to the plant, in this case the dynamic equations.

Fortunately, batch processes are typically repeated many times, and measurements are generally available at the end of each batch. In run-to-run optimization, the idea is to iteratively use these measurements to achieve optimality and feasibility *for the plant* in as few runs as possible.

Optimization problem (1) can be modified at the end of each batch, either via the cost and the constraints, or through the discrete dynamic equations, as discussed next.

2.2 Method A - Cost and constraint modification

Let $\boldsymbol{U}_k^*$ and $\boldsymbol{X}_k^*$ be the optimal trajectories for the k^{th} model-based optimization problem. $\boldsymbol{U}_k^*$ is applied to the plant, with the resulting state profile, $\boldsymbol{X}_{p,k}$. The model-based optimization problem for the $(k+1)^{st}$ batch is defined as:

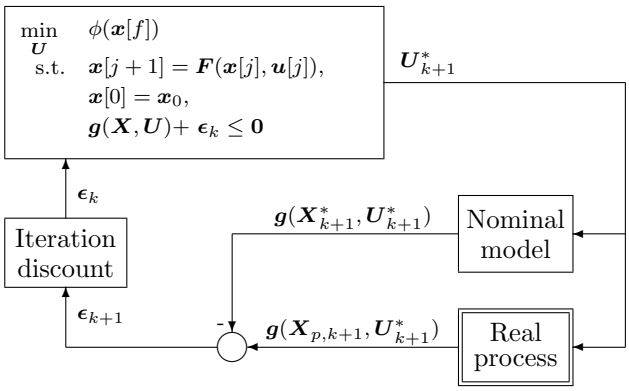

Fig. 1. Method A-0, which modifies the constraints.

$$\begin{aligned} \boldsymbol{U}_{k+1}^* = \underset{\boldsymbol{U}}{\text{argmin}} \quad & \phi(\boldsymbol{x}[f]) + \boldsymbol{\lambda}_k^{\phi}(\boldsymbol{U} - \boldsymbol{U}_k^*) \\ s.t. \quad & \boldsymbol{x}[j+1] = \boldsymbol{F}(\boldsymbol{x}[j], \boldsymbol{u}[j]), \quad \boldsymbol{x}[0] = \boldsymbol{x}_0, \\ & \boldsymbol{g}(\boldsymbol{X}, \boldsymbol{U}) + \boldsymbol{\epsilon}_k + \boldsymbol{\lambda}_k^{\boldsymbol{g}}(\boldsymbol{U} - \boldsymbol{U}_k^*) \leq \boldsymbol{0}, \end{aligned} \tag{2}$$

where the modifier terms are defined as:

$$\boldsymbol{\epsilon}_k = \boldsymbol{g}(\boldsymbol{X}_{p,k}, \boldsymbol{U}_k^*) - \boldsymbol{g}(\boldsymbol{X}_k^*, \boldsymbol{U}_k^*), \tag{3}$$

$$\boldsymbol{\lambda}_k^{\phi} = \frac{d\phi(\boldsymbol{x}_{p,k}[f])}{d\boldsymbol{U}} - \frac{d\phi(\boldsymbol{x}_k^*[f])}{d\boldsymbol{U}}, \tag{4}$$

$$\boldsymbol{\lambda}_k^{\boldsymbol{g}} = \frac{d\boldsymbol{g}(\boldsymbol{X}_{p,k}, \boldsymbol{U}_k^*)}{d\boldsymbol{U}} - \frac{d\boldsymbol{g}(\boldsymbol{X}_k^*, \boldsymbol{U}_k^*)}{d\boldsymbol{U}}. \tag{5}$$

Note that only the cost and the constraints are modified. It can be shown that this approach is equivalent to standard modifier adaptation for the static case. The derivative operator, $\frac{d(\cdot)}{d\boldsymbol{U}}$, represents the derivatives with respect to $\boldsymbol{U}$, taking into account that $\boldsymbol{X}$, and only $\boldsymbol{X}$, is determined by $\boldsymbol{U}$.

The 0^{th}-order term, $\boldsymbol{\epsilon}_k$, matches the *values* of the constraints for the plant and the model at $\boldsymbol{U}_k^*$. The 1^{st}-order terms, $\boldsymbol{\lambda}_k^{\phi}$ and $\boldsymbol{\lambda}_k^{\boldsymbol{g}}$, match the *gradients* of the cost and the constraints for the plant and the model, also at $\boldsymbol{U}_k^*$. As we will see in Section 2.4, this can be used to ensure optimality upon convergence.

The modifier $\boldsymbol{\epsilon}_k$ is determined by measuring the values of the constraints for the plant. If only the 0^{th}-order term is used, i.e. $\boldsymbol{\lambda}_k^{\phi} = 0$ and $\boldsymbol{\lambda}_k^{\boldsymbol{g}} = 0$ for all k, we will refer to the method as Method A-0. The method is then equivalent to that proposed by Marchetti et al. (2007). If the 1^{st}-order correction terms are used, the method is referred to as Method A-1.

2.3 Method B - Modification of the difference equations

An alternative to correcting the cost and the constraints is to add modifier terms to the discrete dynamic equations, $\boldsymbol{F}$. At the end of the k^{th} batch, the model-based optimization problem for the $(k+1)^{st}$ batch then reads:

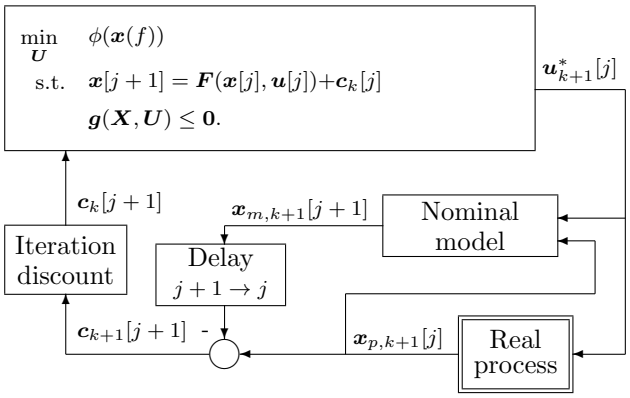

Fig. 2. Method B-0, which modifies the dynamic equations.

$$\begin{aligned} \boldsymbol{U}^*_{k+1} = \underset{\boldsymbol{U}}{\text{argmin}} \quad & \phi(\boldsymbol{x}[f]) \qquad (6)\\ s.t. \quad & \boldsymbol{x}[j+1] = \boldsymbol{F}(\boldsymbol{x}[j], \boldsymbol{u}[j]) + \boldsymbol{c}_k[j] \\ & \qquad +\boldsymbol{C}_k[j] \begin{bmatrix} \boldsymbol{x}[j] - \boldsymbol{x}_{p,k}[j] \\ \boldsymbol{u}[j] - \boldsymbol{u}^*_k[j] \end{bmatrix}, \qquad \boldsymbol{x}[0] = \boldsymbol{x}_0, \\ & \boldsymbol{g}(\boldsymbol{X}, \boldsymbol{U}) \leq \boldsymbol{0}, \end{aligned}$$

where the modifier terms are defined as:

$$\boldsymbol{c}_k[j] = \boldsymbol{x}_{p,k}[j+1] - \boldsymbol{x}_{m,k}[j+1], \qquad (7)$$

$$\boldsymbol{C}_k[j] = \left[\boldsymbol{A}_{p,k}[j] - \boldsymbol{A}_k[j],\ \boldsymbol{B}_{p,k}[j] - \boldsymbol{B}_k[j]\right], \qquad (8)$$

with $\boldsymbol{x}_{m,k}[j+1] = \boldsymbol{F}(\boldsymbol{x}_{p,k}[j], \boldsymbol{u}^*_k[j])$, the one-step-ahead state prediction that the model would have made. $\boldsymbol{A}_{p,k}[j]$ and $\boldsymbol{B}_{p,k}[j]$ are the partial derivatives of the dynamic equations for the plant:

$$\boldsymbol{A}_{p,k}[j] = \frac{\partial \boldsymbol{F}_p\left(\boldsymbol{x}_{p,k}[j], \boldsymbol{u}^*_k[j]\right)}{\partial \boldsymbol{x}[j]}, \quad j = 0, \ldots, f-1. \qquad (9)$$

$$\boldsymbol{B}_{p,k}[j] = \frac{\partial \boldsymbol{F}_p\left(\boldsymbol{x}_{p,k}[j], \boldsymbol{u}^*_k[j]\right)}{\partial \boldsymbol{u}[j]}, \quad j = 0, \ldots, f-1. \qquad (10)$$

$\boldsymbol{A}_k[j]$ and $\boldsymbol{B}_k[j]$ are the partial derivatives of the model $\boldsymbol{F}$ evaluated at $\boldsymbol{x}_{p,k}[j]$ and $\boldsymbol{u}^*_k[j]$.

It is particularly interesting to examine the effect of the 0^{th}-order modifier, $\boldsymbol{c}_k[j]$. We call this a 0^{th}-order correction because the correction involves only the *values* of the states. For any $\boldsymbol{x}_{p,k}[j]$ and $\boldsymbol{u}^*_k[j]$ from the k^{th} batch, $\boldsymbol{c}_k[j]$ is the difference between what the model would have predicted for the states at the sampling instant t_{j+1} and the states which actually occurred, $\boldsymbol{x}_{p,k}[j+1]$. Hence, $\boldsymbol{c}_k[j]$ can be seen as a one-step-ahead prediction correction. This ensures that the modified dynamic equations for the $(k+1)^{st}$ batch have no prediction error for the input trajectory $\boldsymbol{U}^*_k$, that is:

$$\boldsymbol{F}(\boldsymbol{x}_{p,k}, \boldsymbol{u}_k) + \boldsymbol{c}_k[j] = \boldsymbol{F}_p(\boldsymbol{x}_{p,k}, \boldsymbol{u}_k) \qquad \forall j. \qquad (11)$$

If only the 0^{th}-order term $\boldsymbol{c}_k[j]$ is used, i.e. $\boldsymbol{C}_k[j] = 0$, the method will be referred to as Method B-0. If the 1^{st}-order term, $\boldsymbol{C}_k[j]$, is also used, the method will be referred to as Method B-1.

2.4 Properties upon convergence

The attraction of Methods A-0, A-1, B-0 and B-1 is that, upon convergence, it is possible to prove that either feasibility (Methods A-0 and B-0), or feasibility and optimality (Methods A-1 and B-1) will be achieved for the plant.

Theorem 1. If either Method A-0 or Method B-0 converges, then it will converge to an input trajectory that is feasible for the plant.

Theorem 2. If either Method A-1 or Method B-1 converges, then it will converge to an input trajectory that satisfies the 1^{st}-order necessary conditions of optimality for the plant.

Outlines of the proofs are given in Appendix A.

It should be noted that, for these statements to hold regarding Method B, we must assume that ϕ and $\boldsymbol{g}$ are known and that the only uncertainty in the model-based optimization problem comes from the model itself. This is a reasonable assumption. Yet, even if this is not the case, it is often possible, through a suitable choice of state variables, to transfer the uncertainty to $\boldsymbol{F}$.

2.5 Comparison of Methods A and B

Methods A and B have much in common. If they converge, they converge to feasible trajectories for the plant, and they both require gradient estimation to also guarantee optimality upon convergence. Which method is more useful? We now give two reasons why Method B has the potential to be superior, yet, for the moment, Method A is easier to implement.

Integration with a control layer. The optimized input profiles are rarely applied open-loop to a batch process, although for simplicity it was considered to be the case in this paper. A within-batch control layer is used to ensure that safety and quality constraints are respected. Hence, the optimization problem will typically supply *reference trajectories* to low-level controllers. In this respect, Method B will supply accurate reference trajectories, as the updated model has no prediction error upon convergence. On the other hand, even if Method A yields the optimal plant inputs, it will not provide the correct state trajectories to be used as references, as the model is not updated.

Gradient estimation. Gradients are likely to be experimentally costly to evaluate for both methods as estimating them requires many measurements. However, both the type of measurements and the number of batches required to obtain the measurements differ for each method.

For Method A-1, it will generally be necessary to perform multiple batches, each with slightly different inputs, to determine the 1^{st}-order terms $\boldsymbol{\lambda}^{\phi}_k$ and $\boldsymbol{\lambda}^{\boldsymbol{g}}_k$. The derivatives can then be calculated via finite differences. The problem is that this requires $fn_u + 1$ batches. Hence, the number of batches required to estimate one set of gradients rises linearly with f.

For Method B-1, obtaining $\boldsymbol{C}_k[j]$ is also a difficult task. It can be shown to be equivalent to identifying a linear time-varying model around the trajectories $\boldsymbol{U}^*_k$ and $\boldsymbol{X}_{p,k}$. This requires sufficient excitation and so, as for the gradient terms for Method A-1, it will generally be necessary to perform multiple batches. The computation of the gradients of the dynamic equations requires $n_x + n_u + 1$

measurements at each sampling instant. To obtain these measurements, it would be necessary to perform $n_x + n_u + 1$ batches, each with different input and state profiles. Determining input profiles that result in measurements that are suitable for gradient calculation is a complicated problem that will not be adressed in this paper.

The key difference is that, for Method A, the number of batches required to estimate one set of gradients rises linearly with the number of sampling instants f, while the number of batches required by Method B to estimate one set of gradients is independent of f. This appears to give Method B an advantage over Method A. Unfortunately, for Method B, there currently exists no systematic way of determining what inputs should be applied to the plant to determine the necessary measurements.

3. SIMULATED EXAMPLE

3.1 Plant

Methods A-0, A-1, B-0 and B-1 are tested on the simulated example of an isothermal semi-batch reactor (Ruppen et al. (1998)). Two reactions occur: $A + B \rightarrow C$ and $2B \rightarrow D$. The objective is to maximize the production of C at a fixed final time. The manipulated variable is the feed rate of B. There are limits on the concentrations of B and D at final time.

If we consider the state vector $[c_A \ c_B \ V]^T$, where c_A and c_B are the concentrations of A and B and V is the volume of material in the reactor, the plant is governed by the continuous dynamic equations:

$$\begin{bmatrix} \dot{c_A} \\ \dot{c_B} \\ \dot{V} \end{bmatrix} = \begin{bmatrix} -k_1 c_A c_B \\ -k_1 c_A c_B - 2k_2 {c_B}^{\kappa} \\ 0 \end{bmatrix} + \frac{1}{V} \begin{bmatrix} -c_A \\ c_{B_{in}} - c_B \\ V \end{bmatrix} u. \quad (12)$$

The concentrations of C and D can be expressed as:

$$c_C = \frac{1}{V}(c_{A0} V_0 - c_A V), \quad (13)$$

$$c_D = \frac{1}{2V}\big((c_A + c_{Bin} - c_B)V - (c_{A0} + c_{Bin} - c_{B0})\big). \quad (14)$$

The numerical values used in this simulation are given in Table 1. We consider a piecewise-constant input, with the sampling period $h = 15.625$ min between each switching instant.

3.2 Model

The discrete-time nominal model is:

$$\begin{bmatrix} x_1[j+1] \\ x_2[j+1] \\ x_3[j+1] \end{bmatrix} = \begin{bmatrix} x_1[j] \\ x_2[j] \\ x_3[j] \end{bmatrix} + h \begin{bmatrix} -k_{1,o}\, x_1[j] x_2[j] \\ -k_{1,o}\, x_1[j] x_2[j] - 2k_{2,o}\, x_2[j]^{\kappa_o} \\ 0 \end{bmatrix} + \frac{h}{x_3[j]} \begin{bmatrix} -x_1[j] \\ c_{B_{in}} - x_2[j] \\ x_3[j] \end{bmatrix} u[j], \qquad \boldsymbol{x}[0] = \boldsymbol{x}_0, \quad (15)$$

where $k_{1,o}$, $k_{2,o}$ and κ_o are the nominal values of the uncertain parameters, and the concentrations c_C and c_D are given by equations (13) and (14). The model-based optimization problem (without modifiers) is formulated as:

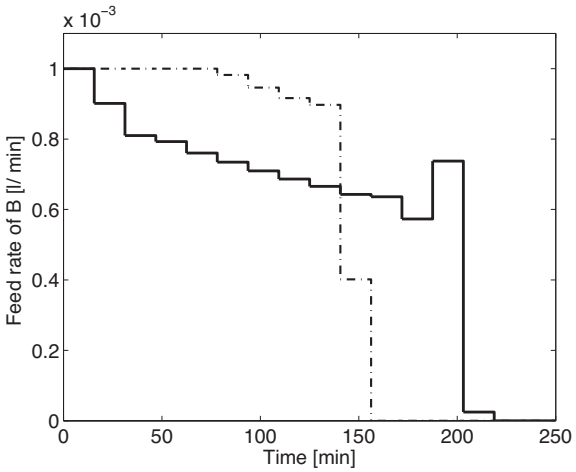

Fig. 3. Optimal inputs: plant in solid, nominal model in dashed.

$$\begin{aligned} \boldsymbol{U}^* = \underset{\boldsymbol{U}}{\text{argmax}} \quad & x_3[f] c_C[f] \quad (16) \\ s.t. \quad & \boldsymbol{x}[j+1] = \boldsymbol{F}(\boldsymbol{x}[j], u[j]), \quad \boldsymbol{x}[0] = \boldsymbol{x}_0, \\ & \text{equations } (13) - (14) \\ & u_{min} \le u[j] \le u_{max} \\ & c_B[f] \le c_{Bf,max} \\ & c_D[f] \le c_{Df,max}. \end{aligned}$$

Notice that all the uncertainty in the model-based optimization problem is contained in $\boldsymbol{F}$.

Table 1. Parameters, operating bounds and initial conditions.

k_1	0.053	l/mol min	$k_{1,o}$	0.03	l/mol min
k_2	0.128	l/mol min	$k_{2,o}$	0.15	l/mol min
κ	2		κ_o	2.5	
c_{Bin}	5	mol/l	u_{min}	0	l/min
u_{max}	0.001	l/min	$c_{Bf,max}$	0.025	mol/l
$c_{Df,max}$	0.15	mol/l	c_{A0}	0.72	mol/l
c_{B0}	0.05	mol/l	V_0	1	l
h	15.625	min	f	16	

3.3 Simulation of A-0 and B-0

Figure 3 displays the optimal input for the plant and for the nominal model. The convergence towards feasibility for Methods A-0 and B-0 is shown in Figure 4. Both methods also converge to near optimality. Figures 5 and 6 indicate that the convergence path is infeasible for both methods. Indeed, there is no guarantee that either of the methods will converge following a feasible path. Constraint violation could be avoided by including an inner controller to monitor constraints. Another possibility would be to add a 'back-off' (negative bias) to the constraint.

3.4 Simulation of A-1 and B-1

Figure 4 also shows that both Methods A-1 and B-1 converge to optimality, although by different paths. The path by which Method A-1 converges results in violation of the constraint on B, while the path by which Method B-1 converges results in violation of the constraint on D. This again highlights that, for a practical implementation,

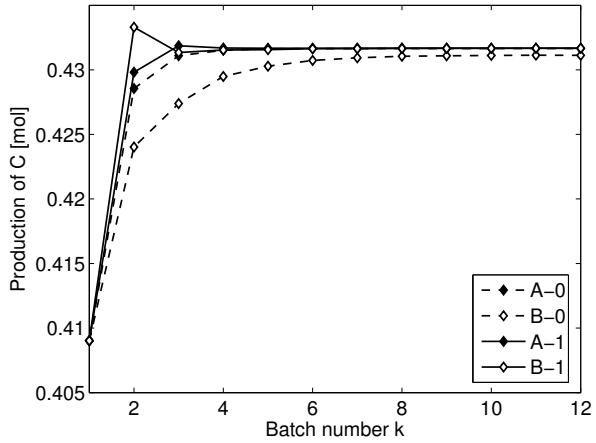

Fig. 4. Cost function: production of C vs. the batch number. The dot-dashed line indicates the optimal production for the plant.

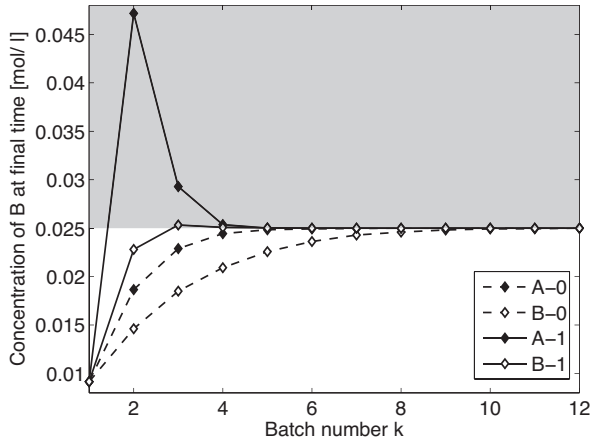

Fig. 5. First constraint: Concentration of B at final time vs. the batch number. The shaded region indicates constraint violation.

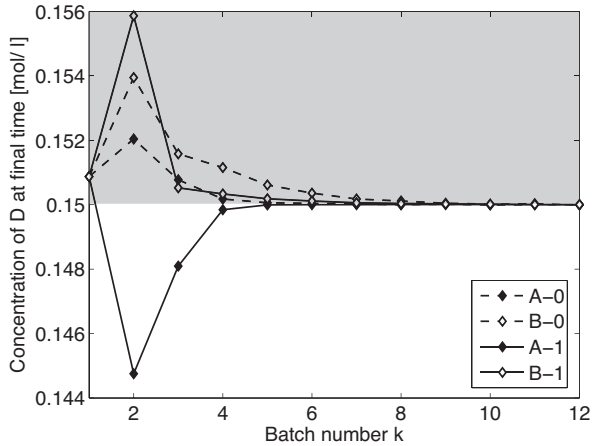

Fig. 6. Second constraint: concentration of D at final time vs. the batch number. The shaded region indicates constraint violation.

the methods would need to be combined with within-batch control.

It is assumed that a method is available for calculating suitable inputs, such that the gradients required in Method B-1 could be determined. Given this assumption, Method A-1 requires 17 batches to computete each set of gradients, while Method B-1 requires only 5 batches.

It is interesting to note that the performance of Methods A-0 and B-0 is similar to that of Methods A-1 and B-1. This is because, for this particular problem, achieving optimality is mainly dependent on meeting the active constraints. Methods A-1 and B-1 are expected to outperform Methods A-0 and B-0 for problems where optimality is mainly dependent on driving the cost sensitivity to zero.

It is worth noting that the convergence of the proposed algorithms can be significantly altered by filtering the modifier terms. For example, a low-pass filter generally results in slower, yet less oscillatory convergence.

4. CONCLUSIONS

In the presence of modeling uncertainty, resolving the model-based optimization problem will, in general, not lead to optimality for the plant. Modifier adaptation is a methodology for using plant measurements to achieve optimality despite modeling uncertainty. Until now, this has only been applied to processes operating at steady state. In this paper, we have successfully extended the modifier-adaptation concept to transient (batch) processes.

Two measurement-based optimization methodologies have been proposed. Method A modifies the cost and the constraints in the model-based optimization problem, while Method B modifies the dynamic equation. Each methodology has two variants, depending on whether gradients are used, giving a total of four methods.

Both methodologies guarantee feasibility upon convergence, and also optimality if gradient terms are used. An attractive aspect of the methods is that this holds even in the case of *structural* mismatch between the nominal model and the plant. This is because, unlike the standard two-step approach, the corrections are aimed at correcting the necessary conditions of optimality, rather than the (structurally incorrect) model of the process.

The methods were compared in simulation, which showed that there is little difference in terms of speed of convergence. The main difference between the two methodologies lies in the measurements required to obtain the necessary modifiers.

This paper raises an important question: Is it better to modify the cost and constraints (Method A) or the dynamic equations (Method B)? A preliminary discussion on this subject in Section 2.5 shows that Method B has the potential to be more useful, yet has greater implementation difficulties.

REFERENCES

Bonvin, D. (1998). Optimal Operation of Batch Reactors - A Personal View. *Journal of Process Control*, 8(5-6), 355–368.

Clarke-Pringle, T.L. and Mac Gregor, J.F. (1998). Optimization of molecular weight distribution using batch-to-batch adjustments. *Ind. Eng. Chem. Res*, 37, 3660–3669.

Cruse, A., Marquardt, W., Oldenburg, J., and Schlegel, M. (2006). Batch process modeling and optimization. In E. Korovessi and A. Linninger (eds.), *Batch Processes*, 305–387. CRC Press, Boca Raton, FL.

Filippi-Bossy, C., Bordet, J., Villermaux, J., Marchal-brassely, S., and Georgakis, C. (1989). Batch reactor optimization by use of tendency models. *Computers & Chemical Engineering*, 13(1-2), 35–47.

Forbes, F., Marlin, T.E., and MacGregor, J.F. (1994). Model adequacy requirements for optimizing plant operations. *Computers and Chemical Engineering*, 18(6), 497–510.

François, G., Srinivasan, B., and Bonvin, D. (2005). Use of measurements for enforcing the necessary conditions of optimality in the presence of constraints and uncertainty. *Journal of Process Control*, 15(6), 701–712.

Gao, W. and Engell, S. (2004). A modified iterative set-point optimization strategy with application to batch chromatography. In *IEEE International Conference on Control Applications, 2004*, 705–710. Taipei.

Ge, M., Wang, Q., Chiu, M., Lee, T., Hang, C., and Teo, K. (2000). An Effective Technique for Batch Process Optimization with Application to Crystallization. *Chemical Engineering Research and Design*, 78(1), 99–106.

Kadam, J., Schlegel, M., Srinivasan, B., Bonvin, D., and Marquardt, W. (2007). Dynamic optimization in the presence of uncertainty: From off-line nominal solution to measurement-based implementation. *Journal of Process Control*, 17(5), 389–398.

Krothapally, M., Bennett, B., Finney, W., and Palanki, S. (1999). Experimental implementation of an on-line optimization scheme to batch PMMA synthesis. *ISA Transactions*, 38(2), 185–198.

Marchetti, A., Chachuat, B., and Bonvin, D. (2007). Batch process optimization via run-to-run constraints adaptation. In *ECC*. Kos, Greece.

Marchetti, A., Chachuat, B., and Bonvin, D. (2010). A dual modifier-adaptation approach for real-time optimization. *Journal of Process Control*, 20(9), 1027 – 1037.

Roberts, P.D. (1979). An algorithm for steady-state system optimization and parameter estimation. *Int. J. Systems Sci.*, 10, 719–734.

Rossenwasser, E. and Yusupov, R. (2000). *Sensitivity of Automatic Control Systems*. CRC Press, Boca Raton (FL).

Ruppen, D., Bonvin, D., and Rippin, D. (1998). Implementation of adaptive optimal operation for a semi-batch reaction system. *Computers & Chemical Engineering*, 22(1-2), 185–199.

Srinivasan, B., Bonvin, D., Visser, E., and Palanki, S. (2003). Dynamic optimization of batch processes II. Role of measurements in handling uncertainty. *Computers & Chemical Engineering*, 27(1), 27–44.

Zafiriou, E. and Zhu, J. (1990). Optimal control of semi-batch processes in the presence of modeling error. In *American Control Conference*, 1644–1649. San Diego, CA.

Appendix A. OUTLINE OF CONVERGENCE PROOFS

A.1 Method A

We assume that there exists a unique solution to the difference equations $\boldsymbol{F}$, which can be written as an explicit function of the control variables:

$$\boldsymbol{x}[j] = \boldsymbol{S}_j(\boldsymbol{U}). \tag{A.1}$$

The optimization problem (1) can be written as:

$$\begin{aligned} \boldsymbol{U}^* = \underset{\boldsymbol{U}}{\text{argmin}} \quad & \Phi\left(\boldsymbol{U}\right) \\ s.t. \quad & \boldsymbol{G}(\boldsymbol{U}) \leq 0, \end{aligned} \tag{A.2}$$

where $\Phi(\boldsymbol{U}) = \phi(\boldsymbol{S}_f(\boldsymbol{U}))$ and $\boldsymbol{G}(\boldsymbol{U}) = \boldsymbol{g}(\boldsymbol{S}(\boldsymbol{U}), \boldsymbol{U})$. From this formulation we can see that Method A is equivalent to modifier adaptation for the static case. The proofs regarding optimality and feasibility upon convergence are given in Marchetti et al. (2010).

A.2 Method B

If convergence occurs, with $\lim_{k\to\infty} \boldsymbol{U}_k^* = \boldsymbol{U}_\infty^*$, then, due to the 0^{th}-order correction terms, there is no prediction error, and:

$$\boldsymbol{G}_\infty(\boldsymbol{U}_\infty^*) = \boldsymbol{G}_p(\boldsymbol{U}_\infty^*), \tag{A.3}$$

where $\boldsymbol{G}_p$ are the plant constraints. Hence, feasibility is obtained upon convergence.

The effect of the 1^{st}-order correction term is a little less obvious. We will first show that $\boldsymbol{\nabla}\Phi(\boldsymbol{U})$ can be expressed as a function of the partial derivatives of $\boldsymbol{F}$ and ϕ. We have:

$$\boldsymbol{\nabla}_{\boldsymbol{u}[r]}\Phi(\boldsymbol{U}) = \frac{\partial\phi(\boldsymbol{x}[f])}{\partial\boldsymbol{x}[f]}\frac{\partial\boldsymbol{S}_f(\boldsymbol{U})}{\partial\boldsymbol{u}[r]}, \tag{A.4}$$

Differentiating $\boldsymbol{S}_f$ using the chain rule gives:

$$\frac{\partial\boldsymbol{S}_f(\boldsymbol{U})}{\partial\boldsymbol{u}[r]} = \left(\prod_{i=f-1}^{r+1}\frac{\partial\boldsymbol{F}(\boldsymbol{x}[i],\boldsymbol{u}[i],i)}{\partial\boldsymbol{x}[i]}\right)\frac{\partial\boldsymbol{F}(\boldsymbol{x}[r],\boldsymbol{u}[r],r)}{\partial\boldsymbol{u}[r]}.$$

Combining this with equation (A.4) shows that $\boldsymbol{\nabla}\Phi(\boldsymbol{U})$ depends on the partial derivatives of $\boldsymbol{F}$ and ϕ. In the same manner, we can express $\boldsymbol{\nabla}\boldsymbol{G}(\boldsymbol{U}_p)$ as a function of the partial derivatives of $\boldsymbol{g}$ and $\boldsymbol{F}$. Such expressions are used in numerical optimization to compute functional derivatives via forward sensitivity analysis (Rossenwasser and Yusupov (2000)). We have assumed that the only uncertainty in the model-based optimization problem comes from $\boldsymbol{F}$. The above development shows that by correcting the gradient of $\boldsymbol{F}$ we also correct the gradients of Φ and $\boldsymbol{G}$. Hence, we have:

$$\boldsymbol{\nabla}\Phi_\infty(\boldsymbol{U}_\infty^*) = \boldsymbol{\nabla}\Phi_p(\boldsymbol{U}_\infty^*), \tag{A.5}$$

$$\boldsymbol{\nabla}\boldsymbol{G}_\infty(\boldsymbol{U}_\infty^*) = \boldsymbol{\nabla}\boldsymbol{G}_p(\boldsymbol{U}_\infty^*), \tag{A.6}$$

where Φ_p is the cost for the plant. From this, it is easily shown that, if $\boldsymbol{U}_\infty^*$ is a KKT point for the model, it is also a KKT point for the plant.