Modifier Adaptation for Constrained Closed-Loop Systems

Sean Costello [*] **Grégory François** [*] **Dominique Bonvin** [*]
Alejandro Marchetti [**]

[*] *Ecole Polytechnique Fédérale de Lausanne, Switzerland (sean.costello@epfl.ch, gregory.francois@epfl.ch, dominique.bonvin@epfl.ch).*
[**] *French-Argentine International Center for Information and Systems Sciences, CIFASIS-CONICET, 27 de Febrero 210 bis, S2000EZP Rosario, Argentina (marchetti@cifasis-conicet.gov.ar).*

Abstract: The steady advance of computational methods makes model-based optimization an increasingly attractive method for process improvement. Unfortunately, the available models are often inaccurate. An iterative optimization method called "modifier adaptation" overcomes this obstacle by incorporating process information into the optimization framework. This paper extends this technique to constrained optimization problems, where the plant consists of a closed-loop system but only a model of the open-loop system is available. The degrees of freedom of the closed-loop system are the setpoints provided to the controller, whereas the model degrees of freedom are the inputs of the open-loop plant. Using this open-loop model and process measurements, the proposed algorithm guarantees both optimality and constraint satisfaction for the closed-loop system upon convergence. A simulated CSTR example with constraints illustrates the method.

Keywords: Real-time optimization, modifier adaptation, plant-model mismatch, optimality

1. INTRODUCTION

The degrees of freedom of industrial processes are generally chosen by operators, on the one hand to meet safety requirements and operating constraints, on the other hand to optimize a performance measure such as product quality or profit. Alternatively, experimental process optimization (Box and Draper, 1969) or model-based process optimization may be used. The first is guided purely by experimental plant data, while the latter is based solely on a (often inaccurate) model. Modifier adaptation (MA) is one of a family of techniques (see e.g. Jang et al. (1987); Tatjewski (2002); Gao and Engell (2005); Skogestad (2000); Srinivasan and Bonvin (2007)) that combines these two radically different approaches by using experimental data to make up for inconsistencies between the model and the plant. For a more comprehensive introduction to this topic, the reader is invited to refer to a previous paper by the same authors (Costello et al., 2013).

MA uses measurements to implement input-affine corrections to the cost and constraint functions in the *model-based* optimization problem, while the process model parameters are kept fixed. MA has been designed to resolve plant-model mismatch, yet the model must still satisfy two conditions:

(1) have the same set of inputs as the plant,
(2) predict a locally convex (concave) cost function at the plant minimum (maximum).

Condition (2) is likely to be satisfied by any reasonable model and its enforcement is discussed by François and Bonvin (2013). Costello et al. (2013) proposed a more general MA formulation that can be applied when Condition (1) does not hold, that is, when the plant and the model have different sets of inputs. However, the "generalized modifier adaptation" algorithm presented in that paper was only applicable to unconstrained problems. In this paper we extend generalized MA to *constrained* problems, retaining the attractive property of optimality upon convergence. Furthermore, we analyze two alternative versions of the algorithm. Although the algorithm is more widely applicable, we focus here on setpoint optimization for a closed-loop system for which only an open-loop model is available.

The paper is organized as follows. After a short review of MA in Section 2, a controlled plant is given as a motivating example in Section 3. Section 4 describes the generalized MA scheme for constrained systems, which is tested in simulation on a continuous stirred-tank reactor in Section 5. Finally, Section 6 concludes the paper.

2. REAL-TIME OPTIMIZATION VIA MODIFIER ADAPTATION

2.1 Problem Formulation

The problem of finding optimal operating conditions for a process is typically expressed mathematically as:

$$\mathbf{u}_p^* := \arg\min_{\mathbf{u}} \phi_p(\mathbf{u}) \quad \text{s.t.} \quad \mathbf{g}_p(\mathbf{u}) \leq \mathbf{0}, \tag{2.1}$$

where $\mathbf{u}$ is the n_u-dimensional vector of inputs, ϕ_p the cost function and $\mathbf{g}_p$ the n_g-dimensional vector of process constraints. Here, the subscript $(\cdot)_p$ indicates a quantity related to the plant.

The functions ϕ_p and $\mathbf{g}_p$ are usually not known accurately, as only the models ϕ and $\mathbf{g}$ are available. Consequently, an approximate solution for the original problem (2.1) is obtained by solving the following model-based problem:

$$\mathbf{u}^* := \arg\min_{\mathbf{u}} \phi(\mathbf{u}) \quad \text{s.t.} \quad \mathbf{g}(\mathbf{u}) \leq \mathbf{0}, \tag{2.2}$$

We will assume in this paper that ϕ and $\mathbf{g}$ are differentiable.

2.2 Standard Modifier-Adaptation Scheme

With MA, process measurements are used to iteratively modify the model-based Problem (2.2) in such a way that, upon convergence, the necessary conditions of optimality (NCO) of the *modified* problem match those of the plant. This is made possible by using modifiers that, at each iteration, are computed as the differences between the measured and predicted values of the constraints, and the measured and predicted cost and constraint gradients. This forces the cost and constraints in the model-based optimization problem to locally match those of the plant. In its simplest form the algorithm proceeds as follows:

Standard MA

(1) Solve the modified model-based optimization problem (P0):

$$\mathbf{u}_k^* := \underset{\mathbf{u}}{\text{argmin}} \quad \phi_{m,k}(\mathbf{u}) \tag{2.3}$$

$$\text{subject to} \quad \mathbf{g}_{m,k}(\mathbf{u}) \leq \mathbf{0}, \tag{2.4}$$

with the modified cost and constraints

$$\phi_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\phi})^T(\mathbf{u} - \mathbf{u}_{k-1}), \tag{2.5}$$

$$\mathbf{g}_{m,k}(\mathbf{u}) := \mathbf{g}(\mathbf{u}) + \boldsymbol{\epsilon}_k + (\boldsymbol{\lambda}_k^{g})^T(\mathbf{u} - \mathbf{u}_{k-1}). \tag{2.6}$$

(2) Apply the plant input $\mathbf{u}_k^*$ to obtain $\phi_p(\mathbf{u}_k^*)$ and $\mathbf{g}_p(\mathbf{u}_k^*)$.

(3) Evaluate (or estimate) the plant gradients $\frac{\partial \phi_p}{\partial \mathbf{u}}(\mathbf{u}_k^*)$ and $\frac{\partial \mathbf{g}_p}{\partial \mathbf{u}}(\mathbf{u}_k^*)$. These gradient terms must be estimated using measurements collected at successive operating points close to $\mathbf{u}_k^*$, for example using finite differences, or with more elaborate methods (Marchetti et al., 2010; Bunin et al., 2013).

(4) Calculate the modifiers for the next iteration:

$$(\boldsymbol{\lambda}_{k+1}^{\phi})^T := \frac{\partial \phi_p}{\partial \mathbf{u}}(\mathbf{u}_k^*) - \frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}_k^*), \tag{2.7}$$

$$(\boldsymbol{\lambda}_{k+1}^{g})^T := \frac{\partial \mathbf{g}_p}{\partial \mathbf{u}}(\mathbf{u}_k^*) - \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*), \tag{2.8}$$

$$\boldsymbol{\epsilon}_{k+1} := \mathbf{g}_p(\mathbf{u}_k^*) - \mathbf{g}(\mathbf{u}_k^*), \tag{2.9}$$

where the n_g-dimensional vector $\boldsymbol{\epsilon}$ encompasses the zeroth-order modifiers, and the n_u-dimensional vector $\boldsymbol{\lambda}^{\phi}$ and the $(n_u \times n_g)$ matrix $\boldsymbol{\lambda}^{g}$ are the first-order modifiers.

(5) $k := k + 1$, return to Step (1).

The main advantage of this approach is that, if the MA scheme converges, then it will do so to the (local) plant optimum, provided the process model is adequate (Marchetti et al., 2009).

3. MOTIVATING EXAMPLE: CLOSED-LOOP SYSTEM

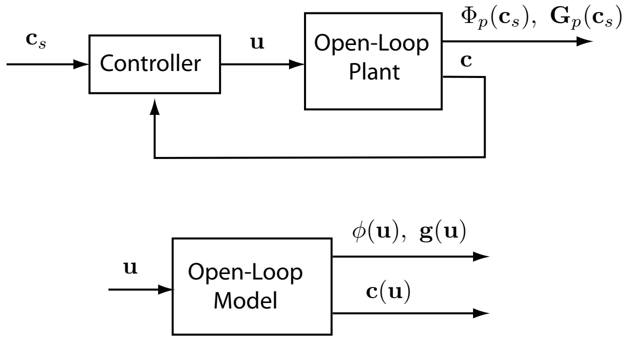

Fig. 1. Controlled plant to be optimized and, for comparison, the plant model that is available.

As discussed in the previous section, standard MA is based on the model having the same inputs as the plant. Depending on the available model, we argue that many systems will not satisfy this criterion (see Costello et al. (2013) for an industrial example of an 80 MW urban waste-incineration plant that does not). In particular closed-loop systems, where only the open-loop process has been modeled, will not satisfy this criterion. For example, consider the controlled plant shown in Figure 1. A plant model will allow the computation of the optimal inputs $\mathbf{u}^*$. However, since the plant is operated in closed loop, there is no direct way of manipulating $\mathbf{u}$ to enforce optimality. Although the model can be used to predict the optimal values of the controlled variables $\mathbf{c}(\mathbf{u}^*)$, the resulting plant inputs will typically differ from $\mathbf{u}^*$ due to imperfect control and plant-model mismatch. It follows that the predicted optimal performance will not be achieved.

In standard modifier adaptation, the open-loop plant inputs are perturbed to estimate the gradient of the plant cost and constraints. This eventually leads to obtaining the optimal open-loop plant input $\mathbf{u}_p^*$. Yet, for closed-loop systems, we are interested in determining the optimal setpoints $\mathbf{c}_s^*$ since optimality of the *closed-loop* system is sought. Furthermore, the plant gradients can be measured with respect to these setpoints (and not $\mathbf{u}$). Fortunately, the fact that the model can predict the controlled variables, and thus also the setpoints required to achieve a certain performance, provides the link to the closed-loop plant. Two approaches exist for applying modifier adaptation in this case:

(1) Invert the model such that its degrees of freedom become the setpoints, as shown in Figure 2. This may be achieved by modeling the steady state behavior of the controller with a law of the form:

$$\mathbf{u} = \mathbf{F_c}\left(\mathbf{c}(\mathbf{u}), \mathbf{c_s}\right). \tag{3.1}$$

For a given $\mathbf{c}_s$, these n_u equations can be solved for $\mathbf{u}$, allowing $\phi(\mathbf{u})$ and $\mathbf{G}(\mathbf{u})$ to be calculated. For this approach the steady-state behavior of the controller

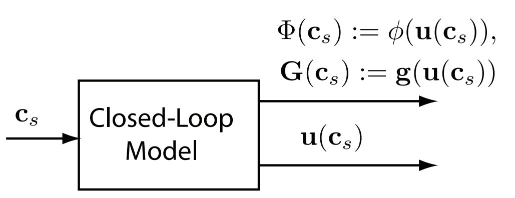
$\Phi(\mathbf{c}_s) := \phi(\mathbf{u}(\mathbf{c}_s))$,
$\mathbf{G}(\mathbf{c}_s) := \mathbf{g}(\mathbf{u}(\mathbf{c}_s))$
$\mathbf{c}_s$
Closed-Loop Model
$\mathbf{u}(\mathbf{c}_s)$

Fig. 2. The closed-loop model that can be obtained by solving model equations for the controller.

must be known, which is not always the case. Alternatively, an ideal controller can be assumed, which ensures:

$$\mathbf{c}(\mathbf{u}) - \mathbf{c}_s = 0. \tag{3.2}$$

These n_c equations can be solved for $\mathbf{u}$ if $n_u = n_c$. Even if one of the above approaches can be applied (which is not always the case), it is likely to result in a closed-loop model that is slower to evaluate, as it will involve solving a system of n_u equations. Slower computation times can be problematic for online optimization.

(2) The second approach is to use the 'modifier adaptation when the plant and model have different inputs' scenario, discussed in Costello et al. (2013), and termed 'generalized modifier adaptation'. This allows modifier adaptation to be applied without any model-inversion or remodeling effort. Furthermore, it can be proved that generalized modifier adaptation achieves the optimal setpoints for the plant upon convergence.

We will explore the latter option in this paper. In fact, we will argue that it is far simpler and has no disadvantages compared to the first option. While Costello et al. (2013) only developed the generalized modifier-adaptation theory for the unconstrained case, we will now extend it to constrained optimization problems.

4. GENERALIZED MODIFIER ADAPTATION

We show next how the standard MA scheme can be altered to optimize a controlled plant on the basis of the plant model. Clearly, the controlled plant and the model have different sets of inputs, the setpoints $\mathbf{c}_s$ and the manipulated inputs $\mathbf{u}$, respectively. The aim is to avoid having to model the closed-loop system, with a controller that may not be fully known. As will be shown, this is completely unnecessary! Generalized modifier-adaptation can be applied in the following context (see Figure 1):

(1) The plant cost function $\Phi_p(\mathbf{c}_s) := \phi_p(\mathbf{u})$ and constraint functions $\mathbf{G}_p(\mathbf{c}_s) := \mathbf{g}_p(\mathbf{u})$ are expressed in terms of the n_c setpoints $\mathbf{c}_s$.
(2) The model cost function $\phi(\mathbf{u})$ and constraint functions $\mathbf{g}(\mathbf{u})$ have n_u inputs $\mathbf{u}$, with $n_u \geq n_c$.
(3) A model $\mathbf{c}(\mathbf{u})$ expressing the mapping from $\mathbf{u}$ to $\mathbf{c}$ is available.

We will introduce two algorithms, each one with a different way of computing the gradient modifiers from the measured/estimated gradients of the controlled plant ,$\frac{\partial \Phi_p}{\partial \mathbf{c}_s}$, and the gradients computed from the open-loop plant model, $\frac{\partial \phi}{\partial \mathbf{u}}$. Since these gradients are computed with respect to different variables, they cannot be compared directly. The first algorithm computes the modifiers in the space of the setpoints $\mathbf{c}_s$. For this, the model gradients are computed by inverting the relationship $\frac{\partial \mathbf{c}}{\partial \mathbf{u}}$. The second algorithm computes the modifiers in the space of the inputs $\mathbf{u}$, by expressing the experimental gradients with respect to $\mathbf{c}_s$ in terms of the inputs $\mathbf{u}$. We present each algorithm and prove optimality and constraint satisfaction upon convergence. Then, the similarity between the two methods is explored, and the effect of filtering on convergence is briefly discussed.

4.1 Method A

Generalized MA: Method A

(1) Solve the modified model-based optimization problem (P1):

$$\mathbf{u}_k^* := \underset{\mathbf{u}}{\text{argmin}} \quad \tilde{\phi}_{m,k}(\mathbf{u}), \tag{4.1}$$

$$\text{subject to} \quad \tilde{\mathbf{g}}_{m,k}(\mathbf{u}) \leq \mathbf{0}, \tag{4.2}$$

with

$$\tilde{\phi}_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\tilde{\boldsymbol{\lambda}}_k^{\Phi})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}), \tag{4.3}$$

$$\tilde{\mathbf{g}}_{m,k}(\mathbf{u}) := \mathbf{g}(\mathbf{u}) + \tilde{\boldsymbol{\epsilon}}_k + (\tilde{\boldsymbol{\lambda}}_k^{G})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}). \tag{4.4}$$

(2) Apply the setpoints $\mathbf{c}_{s,k} := \mathbf{c}(\mathbf{u}_k^*)$ to the plant to obtain $\Phi_p(\mathbf{c}_{s,k})$ and $\mathbf{G}_p(\mathbf{c}_{s,k})$.
(3) Evaluate (or estimate) the plant gradients: $\frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k})$ and $\frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k})$.
(4) Calculate the modifiers for the next iteration:

$$(\tilde{\boldsymbol{\lambda}}_{k+1}^{\Phi})^T := \frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k}) - \frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}_k^*)\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+, \tag{4.5}$$

$$(\tilde{\boldsymbol{\lambda}}_{k+1}^{G})^T := \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k}) - \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+, \tag{4.6}$$

$$\tilde{\boldsymbol{\epsilon}}_{k+1} := \mathbf{G}_p(\mathbf{c}_{s,k}) - \mathbf{g}(\mathbf{u}_k^*), \tag{4.7}$$

with $(\cdot)^+$ indicating the Moore-Penrose pseudo-inverse.
(5) $k := k+1$, return to Step (1).

We claim next that all fixed points of this iterative procedure satisfy the plant NCO.

Theorem 4.1. [Optimality for Method A]
If Method A converges, it will do so to a point satisfying the plant first-order necessary conditions of optimality.

Proof: Upon convergence after K iterations, $\mathbf{u}_{K+1} = \mathbf{u}_K$, and $\mathbf{c}_{s,K+1} = \mathbf{c}_{s,K}$, and the modifier terms will also have converged (function arguments are mostly dropped in the following derivation, they are all evaluated at this stationary point). First, we will derive relationships between $\tilde{\phi}_{m,k}$ and $\tilde{\mathbf{g}}_{m,k}$ and the plant cost and constraints Φ_p and $\mathbf{G}_p$. Upon convergence, one has:

$$(\tilde{\boldsymbol{\lambda}}_K^{\Phi})^T = \frac{\partial \Phi_p}{\partial \mathbf{c}_s} - \frac{\partial \phi}{\partial \mathbf{u}}\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+. \tag{4.8}$$

The gradient of the cost function $\tilde{\phi}_{m,K}$ in Problem (P1) is

$$\frac{\partial \tilde{\phi}_{m,K}}{\partial \mathbf{u}} = \frac{\partial \phi}{\partial \mathbf{u}} + (\tilde{\boldsymbol{\lambda}}_K^{\Phi})^T\frac{\partial \mathbf{c}}{\partial \mathbf{u}}. \tag{4.9}$$

Using (4.8) gives

$$\frac{\partial \tilde{\phi}_{m,K}}{\partial \mathbf{u}} = \frac{\partial \phi}{\partial \mathbf{u}} + \left(\frac{\partial \Phi_p}{\partial \mathbf{c}_s} - \frac{\partial \phi}{\partial \mathbf{u}}\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+\right)\frac{\partial \mathbf{c}}{\partial \mathbf{u}}. \tag{4.10}$$

Multiplying both sides of this equation by $\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}$ and using the identity $\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}} = \left(\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^2$ yields:

$$\frac{\partial \tilde{\phi}_{m,K}}{\partial \mathbf{u}} \left(\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}} \right) = \frac{\partial \Phi_p}{\partial \mathbf{c}_s} \frac{\partial \mathbf{c}}{\partial \mathbf{u}} \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}} \tag{4.11}$$

$$= \frac{\partial \Phi_p}{\partial \mathbf{c}_s} \frac{\partial \mathbf{c}}{\partial \mathbf{u}}. \tag{4.12}$$

The same argument can be used to show that

$$\frac{\partial \tilde{\mathbf{g}}_{m,K}}{\partial \mathbf{u}} \left(\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}} \right) = \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s} \frac{\partial \mathbf{c}}{\partial \mathbf{u}}. \tag{4.13}$$

From the definition of $\tilde{\boldsymbol{\epsilon}}$ given in (4.7)), we can write

$$\tilde{\mathbf{g}}_{m,K}(\mathbf{u}_K^*) = \mathbf{g}(\mathbf{u}_K^*) + \mathbf{G}_p(\mathbf{c}_{s,K}) - \mathbf{g}(\mathbf{u}_K^*) \tag{4.14}$$

$$= \mathbf{G}_p(\mathbf{c}_{s,K}). \tag{4.15}$$

Now, by definition, $\mathbf{u}_K^*$ is a KKT point for Problem (P1). Thus, $\exists\ \boldsymbol{\mu} \geq \mathbf{0}$ such that

$$\frac{\partial \tilde{\phi}_{m,K}}{\partial \mathbf{u}} + \boldsymbol{\mu}^T \frac{\partial \tilde{\mathbf{g}}_{m,K}}{\partial \mathbf{u}} = \mathbf{0}\,. \tag{4.16}$$

Based on equations (4.12) and (4.13) and assuming *rank* $\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right) = n_c$, we can conclude that

$$\frac{\partial \Phi_p}{\partial \mathbf{c}_s} + \boldsymbol{\mu}^T \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s} = \mathbf{0}\,. \tag{4.17}$$

The KKT conditions state that $\boldsymbol{\mu}^T\, \tilde{\mathbf{g}}_{m,K} = 0$. As we have shown that $\tilde{\mathbf{g}}_{m,K} = \mathbf{G}_p$, it follows that

$$\boldsymbol{\mu}^T\, \mathbf{G}_p = 0\,. \tag{4.18}$$

Hence, $\mathbf{c}_{s,K}$ is also a KKT point for the plant. Hence, *if* the scheme converges, it converges to a point satisfying the *plant* NCO. ■

4.2 Method B

This is a an alternative, equally intuitive, way of adapting the standard MA to our problem.

Generalized MA: Method B

(1) Solve the modified model-based optimization Problem (P2):

$$\mathbf{u}_k^* := \underset{\mathbf{u}}{\operatorname{argmin}} \quad \phi_{m,k}(\mathbf{u}), \tag{4.19}$$

$$\text{subject to} \quad \mathbf{g}_{m,k}(\mathbf{u}) \leq \mathbf{0}, \tag{4.20}$$

with

$$\phi_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\phi})^T (\mathbf{u} - \mathbf{u}_{k-1}), \tag{4.21}$$

$$\mathbf{g}_{m,k}(\mathbf{u}) := \mathbf{g}(\mathbf{u}) + \boldsymbol{\epsilon}_k + (\boldsymbol{\lambda}_k^{g})^T (\mathbf{u} - \mathbf{u}_{k-1}). \tag{4.22}$$

(2) Apply the setpoints $\mathbf{c}_{s,k} := \mathbf{c}(\mathbf{u}_k^*)$ to the plant to obtain $\Phi_p(\mathbf{c}_{s,k})$ and $\mathbf{G}_p(\mathbf{c}_{s,k})$.

(3) Evaluate (or estimate) the plant gradients: $\frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k})$ and $\frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k})$.

(4) Calculate the modifiers for the next iteration:

$$(\boldsymbol{\lambda}_{k+1}^{\phi})^T := \frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k}) \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*) - \frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}_k^*) \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*), \tag{4.23}$$

$$(\boldsymbol{\lambda}_{k+1}^{g})^T := \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k}) \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*) - \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*) \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*), \tag{4.24}$$

$$\boldsymbol{\epsilon}_{k+1} := \quad \mathbf{G}_p(\mathbf{c}_{s,k}) - \mathbf{g}(\mathbf{u}_k^*). \tag{4.25}$$

(5) $k := k+1$, return to Step (1).

The idea is to correct the gradients of the model cost and constraints only in those directions that locally influence $\frac{\partial \mathbf{c}}{\partial \mathbf{u}}$. To this end, the post multiplication by $\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)$ removes any components of $\frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}_k^*)$ and $\frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*)$ in the null space of $\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)$. The advantage with respect to Method A is that the modified cost and constraint functions in Step (1) do not contain the nonlinear term $\mathbf{c}(\mathbf{u})$, which could make the optimization problem in Step (1) harder to solve. Just as for Method A, it can be shown that all fixed points of this iterative procedure satisfy the plant NCO.

Theorem 4.2. [Optimality for Method B]
If Method B converges, it will do so to a point satisfying the plant first-order necessary conditions of optimality.

Proof: Based on the definition of $(\boldsymbol{\lambda}^{\phi})^T$, it follows upon convergence that

$$\frac{\partial \phi_{m,K}}{\partial \mathbf{u}} = \frac{\partial \phi}{\partial \mathbf{u}} + \left(\frac{\partial \Phi_p}{\partial \mathbf{c}_s} - \frac{\partial \phi}{\partial \mathbf{u}} \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right)^+ \right) \frac{\partial \mathbf{c}}{\partial \mathbf{u}}, \tag{4.26}$$

which is exactly the same as for Method A in equation (4.10). From here onwards, the proof is the same as for Method A. ■

4.3 Similarity between Methods A and B

With methods A and B, the optimization problems to be solved numerically at each iteration are certainly different. The optimization problem in Method A may be harder to solve, as the cost function contains the nonlinear term $\mathbf{c}(\mathbf{u})$. The similarity of the two methods are stated in the following proposition.

Proposition 4.1. [Similarity between Methods A and B]
Consider the Methods A and B of Eqns (4.1-4.7) and Eqns (4.19-4.25), respectively. The first-order modifier terms of Method B are first-order approximations of those in Method A

Proof: A Taylor-series expansion of the modifier term for the cost function in Problem (P1), with $\mathbf{c}_{s,k-1} = \mathbf{c}(\mathbf{u}_{k-1}^*)$, gives :

$$(\tilde{\boldsymbol{\lambda}}_k^{\Phi})^T (\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}) = (\tilde{\boldsymbol{\lambda}}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) \right) (\mathbf{u} - \mathbf{u}_{k-1}^*) + O\left((\mathbf{u} - \mathbf{u}_{k-1}^*)^2\right). \tag{4.27}$$

But from with the definition of $\tilde{\boldsymbol{\lambda}}^{\Phi}$ in (4.5):

$$(\tilde{\boldsymbol{\lambda}}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) \right) = \frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k-1}) \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) - \frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*) \right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*). \tag{4.28}$$

Comparing the right-hand side with (4.23) gives:

$$(\tilde{\boldsymbol{\lambda}}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) \right) = (\boldsymbol{\lambda}_k^{\phi})^T. \qquad (4.29)$$

Hence, the modifiers in Method B are actually first-order approximations of those in Method A. ■

4.4 Filtering

One important aspect regards the filtering of the modifier terms. The algorithms given above might not always converge. One way to improve the convergence characteristics is to use a first-order, low-pass exponential filter, as suggested by Marchetti et al. (2009) to obtain the filtered modifiers $\bar{\boldsymbol{\lambda}}_k^{\Phi}, \bar{\boldsymbol{\lambda}}_k^{G}$ and $\bar{\boldsymbol{\epsilon}}_k$ (here we describe the procedure for Method A, but it is identical for Method B). An additional step must be added after Step (4):

(4a) Filter the modifiers:

$$\bar{\boldsymbol{\lambda}}_{k+1}^{\Phi} = (\mathbf{I}_{n_c} - \mathbf{K}^{\Phi})\bar{\boldsymbol{\lambda}}_k^{\Phi} + \mathbf{K}^{\Phi}\,\tilde{\boldsymbol{\lambda}}_{k+1}^{\Phi} \qquad (4.30)$$

$$\bar{\boldsymbol{\lambda}}_{k+1}^{G} = (\mathbf{I}_{n_c} - \mathbf{K}^{G})\bar{\boldsymbol{\lambda}}_k^{G} + \mathbf{K}^{G}\,\tilde{\boldsymbol{\lambda}}_{k+1}^{G} \qquad (4.31)$$

$$\bar{\boldsymbol{\epsilon}}_{k+1} = (\mathbf{I}_{n_g} - \mathbf{K}^{\epsilon})\bar{\boldsymbol{\epsilon}}_k + \mathbf{K}^{\epsilon}\,\tilde{\boldsymbol{\epsilon}}_{k+1} \qquad (4.32)$$

The filtered modifiers are then used to compute the modified cost and constraint functions in Step (1):

$$\tilde{\phi}_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\bar{\boldsymbol{\lambda}}_k^{\Phi})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}), \qquad (4.33)$$

$$\tilde{\mathbf{g}}_{m,k}(\mathbf{u}) := \mathbf{g}(\mathbf{u}) + \bar{\boldsymbol{\epsilon}}_k + (\bar{\boldsymbol{\lambda}}_k^{G})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}). \qquad (4.34)$$

For the exponential filters to be stable and have non-oscillatory responses, the matrices, $\mathbf{K}^{\Phi}, \mathbf{K}^{G}$ and $\mathbf{K}^{\epsilon}$ should be chosen with positive eigenvalues in the interval $[0, 1]$. The choice of the filter matrices is discussed in detail in Marchetti et al. (2009). As can be expected, with more filtering, the method is more likely to converge, but it will do so more slowly. Currently, the only practical way to choose these filter matrices is through tuning. A simple heuristic that can automate this tuning procedure for unconstrained problems is given in Appendix A.

5. SIMULATED EXAMPLE

The method is illustrated on the Williams-Otto reactor (Williams and Otto, 1960). We will use the model from Roberts (1979), which has become a standard test problem for real-time optimization techniques (Marchetti et al., 2010). The plant (here simulated reality) is an ideal continuous stirred-tank reactor with the following reactions:

$$A + B \overset{k_1}{\to} C, \qquad k_1 = k_{10}e^{-E_1/(RT)}, \qquad (5.1)$$

$$C + B \overset{k_2}{\to} P + E, \quad k_2 = k_{20}e^{-E_2/(RT)}, \qquad (5.2)$$

$$C + P \overset{k_3}{\to} G, \qquad k_3 = k_{30}e^{-E_3/(RT)}. \qquad (5.3)$$

We choose the (open-loop) plant inputs to be $\mathbf{u} = [F_A, F_B, T]^T$, that is, the feed rates of A and B, and the reactor temperature. However, the degrees of freedom of the controlled plant are the controller setpoints $\mathbf{c}_s = [X_{A,s}, F_{B,s}]^T$ for the mass fraction of A in the reactor and the inlet flow rate of B. The desired products are P and E and the reactor mass holdup is 2105 kg.

A (rather poor) controller adjusts F_A, F_B and T in the following manner:

- $F_B = F_{B,s} + 2$, that is, with an offset in F_B.
- $F_A = \frac{F_B}{2.4}$, that is, F_A is proportional to F_B.
- T is manipulated so as to meet the setpoint $X_{A,s}$, however there is a large steady-state, proportional offset:

$$X_A = 1.5X_{A,s}\,. \qquad (5.4)$$

The block diagram of the controlled CSTR is shown in Figure 3.

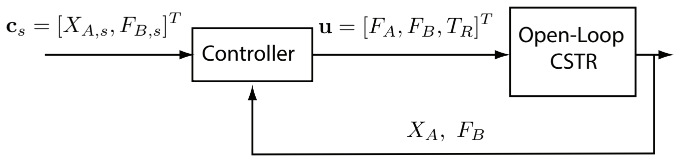
$\mathbf{c}_s = [X_{A,s}, F_{B,s}]^T$
Controller
$\mathbf{u} = [F_A, F_B, T_R]^T$
Open-Loop CSTR
$X_A,\ F_B$

Fig. 3. The controlled CSTR reactor.

The plant model is a two-reaction approximation of the simulated reality:

$$A + 2B \overset{k_1^*}{\to} P + E, \quad k_1^* = k_{10}^*e^{-E_1^*/(RT)}, \qquad (5.5)$$

$$A + B + P \overset{k_2^*}{\to} G, \quad k_2^* = k_{20}^*e^{-E_2^*/(RT)}, \qquad (5.6)$$

with the parameters k_{10}^*, k_{20}^*, E_1^* and E_2^*. The parameters k_{10}^* and k_{20}^* are fixed, whereas E_1^* and E_2^* are adjusted to fit the plant data. The material balance equations for both the plant and its model are given in Costello et al. (2013). From the implementation point of view, the plant controller behavior is considered to be completely unknown. In particular, no knowledge is available regarding the manner in which F_A is regulated.

The profit function to be maximized is

$$\text{Profit} = 1143.38X_P(F_A + F_B) + 25.92X_E(F_A + F_B) \\ -76.23F_A - 114.34F_B, \qquad (5.7)$$

where X_P and X_E are the mass fractions of the products P and E. There are two operational constraints:

$$X_A \leq 0.09, \qquad (5.8)$$

$$X_G \leq 0.6\,. \qquad (5.9)$$

The cost and constraint functions $\Phi_p(\mathbf{c}_s)$, $\mathbf{G}_p(\mathbf{c}_s)$, $\phi(\mathbf{u})$ and $\mathbf{g}(\mathbf{u})$ are constructed by combining the above profit and constraint functions with the plant and model equations, respectively. Table 1 gives the numerical values of the fixed plant and model parameters. The input-output representations of the controlled reactor and the reactor model are shown in Figure 4. The model can be used to *approximately* compute (a) the values of the controlled variables $\mathbf{c}$, and thus the setpoints for the controlled reactor that lead to particular inputs $\mathbf{u}$, and (b) the resulting cost and constraint values.

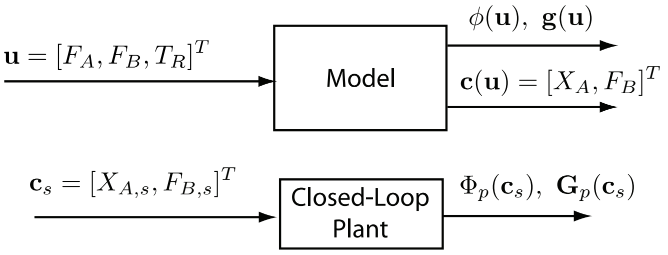
$\phi(\mathbf{u}),\ \mathbf{g}(\mathbf{u})$
$\mathbf{u} = [F_A, F_B, T_R]^T$
Model
$\mathbf{c}(\mathbf{u}) = [X_A, F_B]^T$
$\mathbf{c}_s = [X_{A,s}, F_{B,s}]^T$
Closed-Loop Plant
$\Phi_p(\mathbf{c}_s),\ \mathbf{G}_p(\mathbf{c}_s)$

Fig. 4. Controlled reactor and reactor model for the CSTR.

Table 1. Values of the plant parameters and the two fixed model parameters (the other model parameters are adjusted as shown in Table 2 to generate the investigation cases A-C).

parameter	unit	value
k_{10}	s^{-1}	1.660×10^6
k_{20}	s^{-1}	7.212×10^8
k_{30}	s^{-1}	2.675×10^{12}
E_1	$\mathrm{kJ\,mol}^{-1}$	5.5427×10^4
E_2	$\mathrm{kJ\,mol}^{-1}$	6.9280×10^4
E_3	$\mathrm{kJ\,mol}^{-1}$	9.2377×10^4
k^*_{10}	s^{-1}	6.7157×10^4
k^*_{20}	s^{-1}	1.0341×10^5

Figures 5-8 show the performance of Methods A and B for the three different sets of the adjusted model parameters given in Table 2. The filter matrices are:

$$\mathbf{K}^\Phi = \mathbf{K}^G = \mathbf{K}^\epsilon = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}. \qquad (5.10)$$

Both algorithms converge rapidly to the optimal solution for the plant, where the constraint on X_A is active ($X_{A,s} = 0.06$, which results in $X_A = 0.09$). Figure 7 shows that this constraint is not satisfied prior to convergence. Indeed, while generalized modifier adaptation guarantees constraint satisfaction upon convergence, it may violate constraints in the process of converging. The main observation to be made is that both algorithms behave very similarly. This is to be expected, as we proved in Section 4.3 that Method B is a linearized version of Method A. Hence, either algorithm can be used, bearing in mind that Method B may be computationally less expensive.

Another key issue in the implementation of this scheme is the evaluation of the plant gradient, which is done via finite differences. At the kth iteration, three different values of $\mathbf{c}_s$ are applied to the plant, $\mathbf{c}_{s,k}$, $\mathbf{c}_{s,k} + [\Delta X_{A,s}, 0]^T$ and $\mathbf{c}_{s,k} + [0, \Delta F_{B,s}]^T$, where $\Delta X_{A,s}$ and $\Delta F_{B,s}$ are small perturbations. The gradient is then computed as:

$$\left(\frac{\partial \Phi_p}{\partial \mathbf{c}_s}\right)^T (\mathbf{c}_{s,k}) = \begin{bmatrix} \frac{\Phi_p(\mathbf{c}_{s,k}+[\Delta X_A,0]^T - \Phi_p(\mathbf{c}_{s,k}))}{\Delta X_A} \\ \frac{\Phi_p(\mathbf{c}_{s,k}+[0,\Delta F_B]^T - \Phi_p(\mathbf{c}_{s,k}))}{\Delta F_B} \end{bmatrix}. \qquad (5.11)$$

As gradient estimation is not the focus of this paper, our simulations assume no measurement noise. In practice, the gradient calculation method needs to be robust to measurement noise. While this is outside the scope of this paper, the interested reader is referred to Marchetti et al. (2010), and Marchetti (2013).

Table 2. Values of the adjusted model parameters for the three different cases

Case	E^*_1 ($\mathrm{kJ\,mol}^{-1}$)	E^*_2 ($\mathrm{kJ\,mol}^{-1}$)
A	7900	12500
B	8100	12500
C	8100	12300

6. CONCLUSIONS

Like many other process optimization techniques, modifier adaptation relies on a model of the process. It is typically assumed that the model and the plant have the same inputs. The industrial example of an 80 MW incineration plant has illustrated that this may not be the case. The current paper shows that this assumption will often not hold for closed-loop systems either. Adapting the model such that its inputs and the closed-loop plant inputs are the same can be infeasible if the model is complex (and leads to increased computation times if it is possible). Generalized MA avoids remodeling the system, and this at no extra computational cost. It follows that a broader class of process optimization problems can be tackled, including problems where the plant has an unmodeled control structure.

This work has extended generalized modifier adaptation to constrained optimization problems, proving optimality and constraint satisfaction upon convergence. A simulated CSTR example has shown that the proposed approach can satisfy operational constraints and ensure optimality upon convergence. This is achieved despite significant control error and both structural and parametric plant-model mismatch. Two methods with similar properties have been presented. If computation time is not an issue,

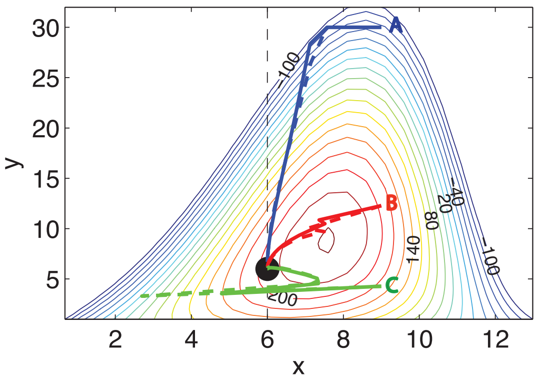

Fig. 5. Evolution of the setpoints during the first 20 iterations of the generalized MA scheme for Cases A-C. The letters A/B/C are the nominal optimal solutions, which correspond to the initial points. Solid = Method A, Dashed = Method B. The contour lines are for the plant cost. The dotted black line indicates the plant constraint on X_A. The black dot indicates the location of the plant optimum.

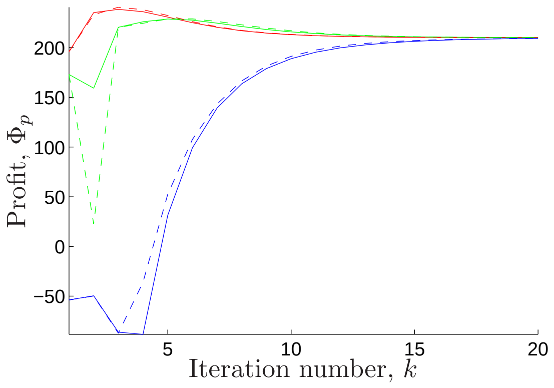

Fig. 6. The profit as a function of the iteration number k. Blue/Red/Green = Cases A/B/C. Solid = Method A, Dashed = Method B. Note that, at each iteration, the plant must be evaluated at 3 slightly different operating points in order to estimate the gradient according to (5.11).

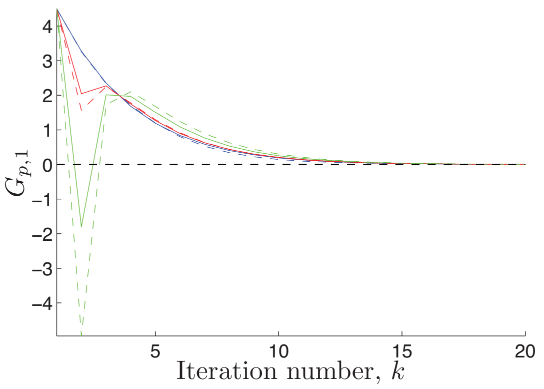

Fig. 7. The constraint on X_A as a function of the iteration number k. Blue/Red/Green = Cases A/B/C. Solid = Method A, Dashed = Method B. The dotted line indicates $G_{p,1} = 0$.

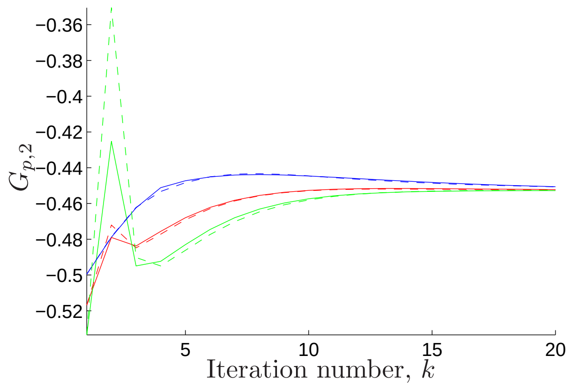

Fig. 8. The constraint on X_G as a function of the iteration number k. Blue/Red/Green = Cases A/B/C. Solid = Method A, Dashed = Method B.

we recommend Method A as its unconstrained version was shown to be closely related to standard MA with a very logical choice of cost function (Costello et al., 2013). The structure of the optimization problem to be solved online in Method B is favorable in terms of computation time. As shown in this paper, Method B is a first-order approximation to Method A, with the same properties upon convergence. For the simulated example shown in this paper, there is negligible difference between the two methods.

One of the limitations of this entire approach is that, being gradient-based, the control law and the system equations are required to be continuous. Hence, switching behavior in the controller or the plant would invalidate our analysis. Although this is an affliction affecting most RTO techniques, future work will hopefully address this issue.

REFERENCES

Box, G.E. and Draper, N.R. (1969). *Evolutionary Operation: A Statistical Method for Process Improvement*, volume 25. Wiley New York.

Bunin, G.A., Francois, G., and Bonvin, D. (2013). From discrete measurements to bounded gradient estimates: A look at some regularizing structures. *Ind. Eng. Chem. Res.*, 52(35), 12500–12513.

Costello, S., François, G., and Bonvin, D. (2013). Real-time optimization when the plant and the model have different inputs. In *Proc. IFAC Symp. DYCOPS*. Mumbai.

François, G. and Bonvin, D. (2013). Use of convex model approximations for real-time optimization via modifier adaptation. *Ind. Eng. Chem. Res.*, 52(33), 11614–11625.

Gao, W. and Engell, S. (2005). Iterative set-point optimization of batch chromatography. *Comp. Chem. Eng.*, 29(6), 1401–1409.

Jang, S.S., Joseph, B., and Mukai, H. (1987). On-line optimization of constrained multivariable chemical processes. *AIChE J.*, 33(1), 26–35.

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

Marchetti, A.G. (2013). A new dual modifier-adaptation approach for iterative process optimization with inaccurate models. *Comp. Chem. Eng.*, 59, 89–100.

Marchetti, A., Chachuat, B., and Bonvin, D. (2009). Modifier-adaptation methodology for real-time optimization. *Ind. Eng. Chem. Res.*, 48(13), 6022–6033.

Roberts, P. (1979). An algorithm for steady-state system optimization and parameter estimation. *Int. J. Systems Sci.*, 10(7), 719–734.

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

Srinivasan, B. and Bonvin, D. (2007). Real-time optimization of batch processes by tracking the necessary conditions of optimality. *Ind. Eng. Chem. Res.*, 46(2), 492–504.

Tatjewski, P. (2002). Iterative optimizing set-point control - The basic principle redesigned. In *Proc IFAC World Congress*. Barcelona.

Williams, T.J. and Otto, R.E. (1960). A generalized chemical processing model for the investigation of computer control. *Trans. of the American Inst. of Elec. Engineers, Part I: Communication and Electronics*, 79(5), 458–473.

Appendix A. ENFORCING CONVERGENCE FOR UNCONSTRAINED PROBLEMS

The following simple heuristic can be used to ensure that all steps taken by the algorithm decrease the plant cost. Insert the following supplementary step after Step (2):

(2a) If $\Phi_p(\mathbf{c}_{s,k}) > \Phi_p(\mathbf{c}_{s,k-1})$, reduce the bandwidth of the exponential filters:

$$\mathbf{K}^{\Phi} := \eta\,\mathbf{K}^{\Phi}, \tag{A.1}$$

with $\eta \in [0, 1]$. Decrement the iteration index (go back a step)

$$k := k - 1, \tag{A.2}$$

and return to Step (1).

Step (2a) guarantees that each step of the algorithm (corresponding to an increment of k) improves the plant cost function. It can be shown that a filter always exists such that a finite step is taken that results in cost improvement. While monotonic cost decrease is not strictly sufficient to prove convergence, we observed in numerous (unconstrained) simulations that this appears to enforce convergence of Methods A and B.

Figures A.1 and A.2 show Method A with this additional step applied to the unconstrained example from Section 5. The values of the uncertain parameters are $E_1^* = 7900\ \mathrm{kJ\,mol^{-1}}$ and $E_2^* = 12300\ \mathrm{kJ\,mol^{-1}}$. The initial filter matrix $\mathbf{K}^{\Phi}$ is the identity matrix, which is equivalent to having no filter. The value of $\eta = 0.5$ was used. It can be seen that without Step (2a) the algorithm oscillates, while including Step (2a) forces convergence in very few iterations. The filter matrix needs to be decreased twice. The experimental cost of this primitive line-search algorithm is not high; it merely requires one plant iteration to determine that the step taken decreases the profit. Unfortunately, adapting this rule to the constrained case is not straightforward.

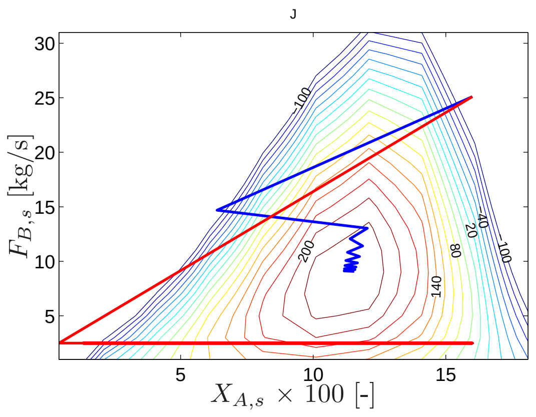

Fig. A.1. Evolution of the plant setpoints during the first 15 iterations of Method A for the unconstrained CSTR. Blue: with Step (2a), Red: without Step (2a). The contour lines are for the plant cost.

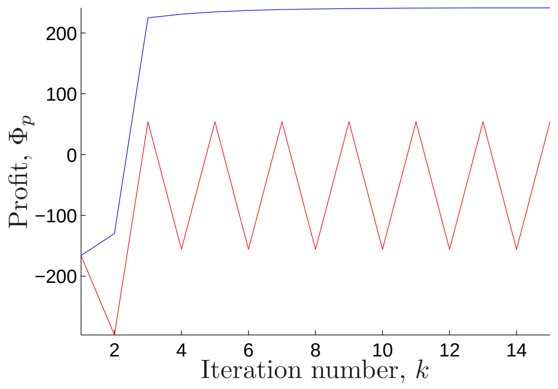

Fig. A.2. The profit as a function of the iteration number k for the unconstrained CSTR. Blue: with Step (2a), Red: without Step (2a).