Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

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, which results in significant plant-model mismatch. An iterative optimization method called "modifier adaptation" overcomes this obstacle by incorporating plant information into the optimization framework. A particular situation is when the plant operates under feedback control and only a model of the open-loop system is available. This paper extends modifier adaptation for constrained optimization problems to that particular situation. The inputs of the controlled plant are the setpoints, whereas the model inputs are the manipulated variables. Using this open-loop model and process measurements, the proposed algorithm guarantees both optimality and constraint satisfaction for the controlled plant upon convergence. A simulated CSTR example with constraints illustrates the method.

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

1. INTRODUCTION

Industrial processes have a certain number of degrees of freedom, the values of which are chosen by operators to meet safety requirements and operating constraints and to optimize a performance measure such as product quality or profit. For this, both experimental process optimization (Box and Draper, 1969) and model-based numerical optimization can be used. The former is guided purely by experimental plant data, while the latter is based solely on an (often inaccurate) model. In the framework of real-time optimization (RTO), several techniques have been developed to appropriately combine these two radically different approaches by using experimental data to make up for inconsistencies between the model and the plant. A comprehensive overview identifies three main classes of techniques (Chachuat et al., 2009), namely, repeated identification and optimization (Jang et al., 1987; Marlin and Hrymak, 1996), modifier adapatation (Tatjewski, 2002; Gao and Engell, 2005; Marchetti et al., 2009), and optimizing control (Skogestad, 2000; Srinivasan and Bonvin, 2007).

Modifier adaptation (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 cost function at the plant minimum (François and Bonvin, 2013).

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 is only applicable to unconstrained problems. In this paper, we extend generalized MA to *constrained* problems while retaining the attractive property of optimality upon convergence. The focus is on setpoint optimization for a closed-loop system for which only an open-loop model is available, and two alternative versions of the algorithm are presented.

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. RTO VIA MODIFIER ADAPTATION

2.1 Problem Formulation

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

$$\begin{aligned} \mathbf{u}_p^* := \arg\min_{\mathbf{u}} \; & \phi_p(\mathbf{u}) \\ \text{subject to} \quad & \mathbf{g}_p(\mathbf{u}) \leq \mathbf{0}, \end{aligned} \tag{2.1}$$

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 to the original problem (2.1) is obtained by solving the following model-based problem:

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

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

2.2 Standard Modifier Adaptation

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) Initialize the modifier terms: the n_g-dimensional vector of zeroth-order modifiers $\boldsymbol{\epsilon}_1 = \mathbf{0}$, the n_u-dimensional vector of first-order cost modifiers $\boldsymbol{\lambda}_1^{\phi} = \mathbf{0}$, and the $(n_u \times n_g)$ matrix of first-order gradient modifiers $\boldsymbol{\lambda}_1^{g} = \mathbf{0}$. Also, choose arbitrarily $\mathbf{u}_0^* = \mathbf{0}$, and initialize the iteration counter $k = 1$.

(2) 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

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

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

The subscript $(\cdot)_m$ indicates a quantity that has been modified.

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

(4) Evaluate/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).

(5) Calculate the modifiers for the next iteration:

$$\boldsymbol{\epsilon}_{k+1} := \mathbf{g}_p(\mathbf{u}_k^*) - \mathbf{g}(\mathbf{u}_k^*), \tag{2.7}$$

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

$$(\boldsymbol{\lambda}_{k+1}^{g})^T := \frac{\partial \mathbf{g}_p}{\partial \mathbf{u}}(\mathbf{u}_k^*) - \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*). \tag{2.9}$$

(6) Set $k := k + 1$ and return to Step (2).

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; François and Bonvin, 2013).

3. MOTIVATING EXAMPLE: CONTROLLED PLANT

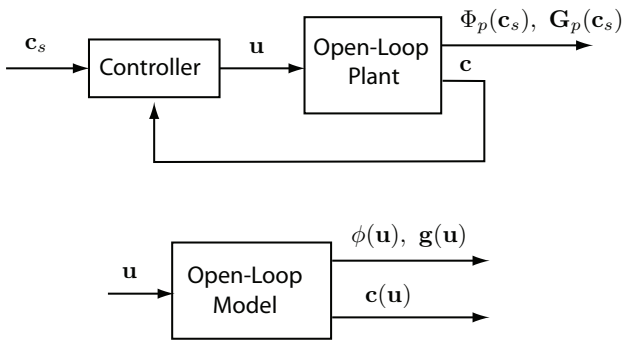

Fig. 1. Controlled plant to be optimized and, for comparison, the plant model that is available.

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 controlled plants will not satisfy this criterion, in particular when the controller is not perfectly known. [1] Furthermore, closed-loop systems, where only the open-loop plant has been modeled, will not satisfy this criterion.

As an 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 MA, the open-loop plant inputs are perturbed to estimate the gradients of the plant cost and constraints. This eventually leads to obtaining the optimal open-loop plant inputs $\mathbf{u}_p^*$. 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 inferred 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) "Model the closed-loop plant" such that the setpoints become the degrees of freedom, 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.

[1] See Costello et al. (2013) for an industrial example of an 80 MW urban waste-incineration plant that does not. Often in industrial settings, little information is available about the industrial control systems that have been installed by a third-party.

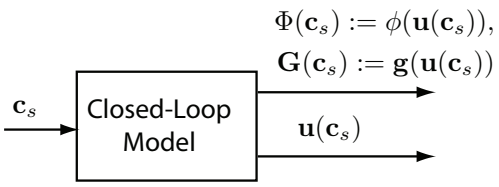

Fig. 2. Closed-loop model obtained by computing an ideal feedforward controller and associating it with the open-loop model.

Alternatively, an ideal controller can be assumed, which ensures:

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

These n_c equations can be solved for $\mathbf{u}$ if $n_u = n_c$, where n_c is the number of controlled variables. 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 difficult to evaluate, as it will involve solving a system of n_u equations. Large computation times can be problematic for online optimization.

(2) The second approach uses "generalized modifier adaptation", which considers the case where the plant and the model have different inputs (Costello et al., 2013). This allows MA to be applied, without any knowledge of the control system. Furthermore, it can be proved that generalized modifier adaptation reaches the optimal plant setpoints 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 former option. While Costello et al. (2013) 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 standard MA can be altered to optimize a controlled plant on the basis of the open-loop plant model. Clearly, the controlled plant and the open-loop 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 as follows (see Figures 1 and 2):

(1) The plant cost function $\Phi_p(\mathbf{c}_s) := \phi_p\big(\mathbf{u}(\mathbf{c}_s)\big)$ and constraint functions $\mathbf{G}_p(\mathbf{c}_s) := \mathbf{g}_p\big(\mathbf{u}(\mathbf{c}_s)\big)$ 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 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 (Method A) computes the modifiers in the space of the setpoints $\mathbf{c}_s$. For this, the model gradients $\frac{\partial \phi}{\partial \mathbf{c}}$ are computed by inverting the relationship $\frac{\partial \mathbf{c}}{\partial \mathbf{u}}$. The second algorithm (Method B) computes the modifiers in the space of the inputs $\mathbf{u}$ by expressing the experimental gradients $\frac{\partial \Phi_p}{\partial \mathbf{c}}$ in terms of the inputs $\mathbf{u}$. We present each algorithm and prove optimality and constraint satisfaction upon convergence. The two methods are then compared, and the effect of filtering on convergence is briefly discussed.

4.1 Method A

The procedure is as follows:

(1) Initialize the modifier terms: the n_g-dimensional vector of zeroth-order modifiers $\boldsymbol{\epsilon}_1 = \mathbf{0}$, the n_c-dimensional vector of first-order cost modifiers $\boldsymbol{\lambda}_1^{\Phi} = \mathbf{0}$, and the $(n_c \times n_g)$ matrix of first-order gradient modifiers $\boldsymbol{\lambda}_1^{G} = \mathbf{0}$. Also, choose arbitrarily $\mathbf{c}_{s,0}^* = \mathbf{0}$, and initialize the iteration counter $k = 1$.
(2) Solve the modified model-based optimization problem (P1):

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

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

with

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

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

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

$$\boldsymbol{\epsilon}_{k+1} := \mathbf{G}_p(\mathbf{c}_{s,k}) - \mathbf{g}(\mathbf{u}_k^*), \tag{4.5}$$

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

$$(\boldsymbol{\lambda}_{k+1}^{G})^T := \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k}) - \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_k^*)\right)^+, \tag{4.7}$$

with $(\cdot)^+$ indicating the Moore-Penrose pseudo-inverse.
(6) Set $k := k + 1$ and return to Step (2).

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: We consider the iterative scheme upon convergence, i.e. $\lim_{k\to\infty} \mathbf{u}_k = \mathbf{u}_\infty$. We will first derive relationships between $\tilde{\phi}_{m,k}$ and $\tilde{\mathbf{g}}_{m,k}$ and the plant cost and constraints Φ_p and $\mathbf{G}_p$.[2] Upon convergence, one has:

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

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

[2] The function arguments will be dropped in the following derivation as all variables are evaluated at the stationary point corresponding to $\mathbf{u}_\infty$.

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

which, using (4.8), gives:

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

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:

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

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

The same argument can be used to show that

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

From the definition of $\boldsymbol{\epsilon}$ given in (4.5), one can write:

$$\begin{aligned} \tilde{\mathbf{g}}_{m,\infty}(\mathbf{u}_{\infty}^*) &= \mathbf{g}(\mathbf{u}_{\infty}^*) + \mathbf{G}_p(\mathbf{c}_{s,\infty}) - \mathbf{g}(\mathbf{u}_{\infty}^*) \\ &= \mathbf{G}_p(\mathbf{c}_{s,\infty}). \end{aligned} \tag{4.14}$$

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

$$\frac{\partial \tilde{\phi}_{m,\infty}}{\partial \mathbf{u}} + \boldsymbol{\mu}^T \frac{\partial \tilde{\mathbf{g}}_{m,\infty}}{\partial \mathbf{u}} = \mathbf{0}\,. \tag{4.15}$$

It follows from equations (4.12) and (4.13) and assuming $rank\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}\right) = n_c$ that

$$\frac{\partial \Phi_p}{\partial \mathbf{c}_s} + \boldsymbol{\mu}^T \frac{\partial \mathbf{G}_p}{\partial \mathbf{c}_s} = \mathbf{0}\,. \tag{4.16}$$

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

$$\boldsymbol{\mu}^T\, \mathbf{G}_p = 0\,. \tag{4.17}$$

Hence, $\mathbf{c}_{s,\infty}$ 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 an alternative, equally intuitive, way of adapting the standard MA to our problem. The procedure is as follows:

(1) Initialize the modifier terms: the n_g-dimensional vector of zeroth-order modifiers $\boldsymbol{\epsilon}_1 = \mathbf{0}$, the n_u-dimensional vector of first-order cost modifiers $\boldsymbol{\lambda}_1^{\phi} = \mathbf{0}$, and the $(n_u \times n_g)$ matrix of first-order gradient modifiers $\boldsymbol{\lambda}_1^{g} = \mathbf{0}$. Also, choose arbitrarily $\mathbf{u}_0^* = \mathbf{0}$, and initialize the iteration counter $k = 1$.

(2) Solve the modified model-based optimization problem (P2):

$$\mathbf{u}_k^* := \underset{\mathbf{u}}{\text{argmin}} \quad \phi_{m,k}(\mathbf{u}), \tag{4.18}$$

$$\text{subject to} \quad \mathbf{g}_{m,k}(\mathbf{u}) \leq \mathbf{0}, \tag{4.19}$$

with

$$\phi_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\phi})^T(\mathbf{u} - \mathbf{u}_{k-1}^*), \tag{4.20}$$

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

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

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

(5) Calculate the modifiers for the next iteration:

$$\boldsymbol{\epsilon}_{k+1} := \quad \mathbf{G}_p(\mathbf{c}_{s,k}) - \mathbf{g}(\mathbf{u}_k^*), \tag{4.22}$$

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

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

(6) Set $k := k + 1$ and return to Step (2).

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 (2) do not contain the nonlinear term $\mathbf{c}(\mathbf{u})$, which could make the optimization problem 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

$$\frac{\partial \phi_{m,\infty}}{\partial \mathbf{u}} = \frac{\partial \phi}{\partial \mathbf{u}} + \left(\frac{\partial \Phi_p}{\partial \mathbf{c}_s} - \frac{\partial \phi}{\partial \mathbf{u}} \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}} \right)^+ \right) \frac{\partial \mathbf{c}}{\partial \mathbf{u}}, \tag{4.25}$$

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 Comparison between Methods A and B

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

Proposition 4.1. [First-order equivalence between Methods A and B]
Consider the Methods A and B of Eqns (4.1-4.5) and Eqns (4.18-4.22), respectively. The first-order modifier terms in 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:

$$\begin{aligned} (\boldsymbol{\lambda}_k^{\Phi})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}) =& (\boldsymbol{\lambda}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}_{k-1}^*) \right) (\mathbf{u} - \mathbf{u}_{k-1}^*) \\ & + O\left((\mathbf{u} - \mathbf{u}_{k-1}^*)^2 \right). \end{aligned} \tag{4.26}$$

From the definition of $\tilde{\boldsymbol{\lambda}}^{\Phi}$ in (4.6), one can write:

$$(\boldsymbol{\lambda}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1})\right) = \frac{\partial \Phi_p}{\partial \mathbf{c}_s}(\mathbf{c}_{s,k-1})\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1}) - \frac{\partial \phi}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1})\left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1})\right)^+ \frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1}). \quad (4.27)$$

Comparing the right-hand sides of (4.27) and (4.23) gives:

$$(\boldsymbol{\lambda}_k^{\Phi})^T \left(\frac{\partial \mathbf{c}}{\partial \mathbf{u}}(\mathbf{u}^*_{k-1})\right) = (\boldsymbol{\lambda}_k^{\phi})^T. \quad (4.28)$$

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 modifiers. The algorithms given above might not always converge. One way to improve the convergence characteristics is to use first-order, low-pass exponential filtering, 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 (5):

(5a) Filter the modifiers:

$$\bar{\boldsymbol{\lambda}}_{k+1}^{\Phi} = (\mathbf{I}_{n_c} - \mathbf{K}^{\Phi})\bar{\boldsymbol{\lambda}}_k^{\Phi} + \mathbf{K}^{\Phi}\,\boldsymbol{\lambda}_{k+1}^{\Phi}, \quad (4.29)$$

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

$$\bar{\boldsymbol{\epsilon}}_{k+1} = (\mathbf{I}_{n_g} - \mathbf{K}^{\epsilon})\bar{\boldsymbol{\epsilon}}_k + \mathbf{K}^{\epsilon}\,\boldsymbol{\epsilon}_{k+1}. \quad (4.31)$$

The filtered modifiers are then used to compute the modified cost and constraint functions in Step (2) of the next iteration:

$$\tilde{\phi}_{m,k}(\mathbf{u}) := \phi(\mathbf{u}) + (\bar{\boldsymbol{\lambda}}_k^{\Phi})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}), \quad (4.32)$$

$$\tilde{\mathbf{g}}_{m,k}(\mathbf{u}) := \mathbf{g}(\mathbf{u}) + \bar{\boldsymbol{\epsilon}}_k + (\bar{\boldsymbol{\lambda}}_k^{G})^T(\mathbf{c}(\mathbf{u}) - \mathbf{c}_{s,k-1}). \quad (4.33)$$

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 real, 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. These filter matrices are typically chosen through tuning.

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:

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

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

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

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 feed 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 offset:

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

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

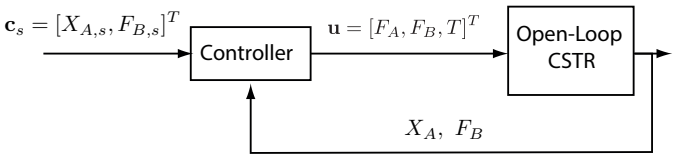

Fig. 3. The controlled CSTR.

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

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

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

with the parameters k_{10}^*, k_{20}^*, E_1^* and E_2^*. Three different nominal models will be considered depending on the values taken by the parameters E_1^* and E_2^*, the parameters k_{10}^* and k_{20}^* being fixed. 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 controller is considered to be unknown. In particular, no knowledge is available regarding the manner in which F_A is manipulated.

The profit function to be maximized is

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

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

$$X_A \le 0.09, \quad (5.8)$$

$$X_G \le 0.6\,. \quad (5.9)$$

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 plant parameters and the fixed model parameters. The input-output representations of the open-loop model and the controlled CSTR 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 would lead to particular inputs $\mathbf{u}$, and (b) the resulting cost and constraint values.

Figures 5-8 show the performance of Methods A and B with three different nominal models as given in Table 2. Optimization is initialized at the solution of the nominal model and proceeds, mostly in the infeasible region, toward the plant optimum. The three trajectories labeled I, II and III correspond to the use of the three models in Table 2. Diagonal filter matrices, with eigenvalues of

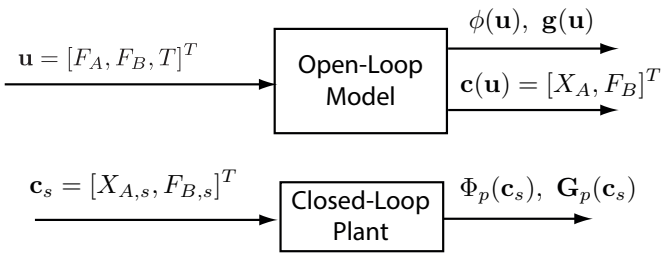

Fig. 4. Open-loop model and controlled CSTR.

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

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	kJ mol^{-1}	5.5427×10^4
E_2	kJ mol^{-1}	6.9280×10^4
E_3	kJ mol^{-1}	9.2377×10^4
k_{10}^*	s^{-1}	6.7157×10^4
k_{20}^*	s^{-1}	1.0341×10^5

0.2 were used. Both algorithms converge rapidly to the optimal solution for the plant, where the constraint on X_A is active ($X_A = 0.09$, which results in $X_{A,s} = 0.06$). 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 easier.

Another key issue in the implementation of this scheme is the evaluation of the plant gradients. The finite-difference method is used in this work; at the k^{th} 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:

$$\left(\frac{\partial \Phi_p}{\partial \mathbf{c}_s}\right)^T (\mathbf{c}_{s,k}) = \begin{bmatrix} \frac{\Phi_p(\mathbf{c}_{s,k}+[\Delta X_A,0]^T)-\Phi_p(\mathbf{c}_{s,k})}{\Delta X_A} \\ \frac{\Phi_p(\mathbf{c}_{s,k}+[0,\Delta F_B]^T)-\Phi_p(\mathbf{c}_{s,k})}{\Delta F_B} \end{bmatrix}. \quad (5.10)$$

As gradient estimation is not the focus of this paper, our simulations assume no measurement noise. In the presence of measurement noise, more advanced schemes should be used (Marchetti et al., 2010; Bunin et al., 2013). Furthermore, since these advanced methods rely on successive operating points, there is no need to perturb the system around the next iterate.

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

Case	E_1^* (kJ mol^{-1})	E_2^* (kJ mol^{-1})
I	7900	12500
II	8100	12500
III	8100	12300

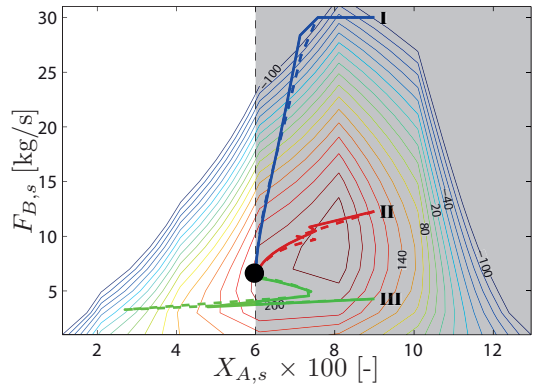

Fig. 5. Evolution of the setpoints during the first 20 iterations of the generalized MA scheme for Cases I-III. Solid = Method A, Dashed = Method B. The contour lines represent the plant profit. The shaded region is infeasible for the plant due to the constraint on X_A. Black dot = plant optimum.

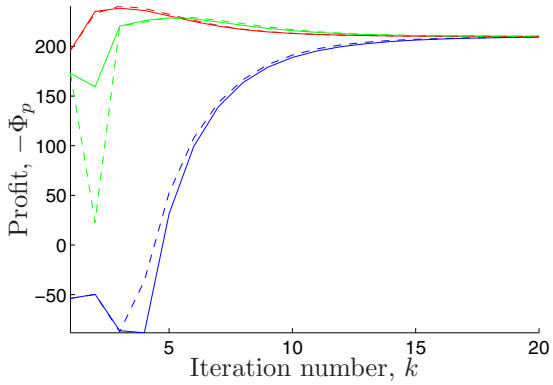

Fig. 6. The profit as a function of the iteration number k. Blue/red/green = Cases I/II/III. Solid = Method A, Dashed = Method B. Note that, at each iteration, the plant is evaluated at 3 slightly different operating points in order to estimate the gradient according to (5.10).

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. However, this assumption does not hold in general for controlled plants as was illustrated by the industrial example of an incineration plant. Adapting the closed-loop model such that its inputs and the inputs of the controlled plant are the same may not be feasible when the plant model is complex or the control system is not fully known. 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

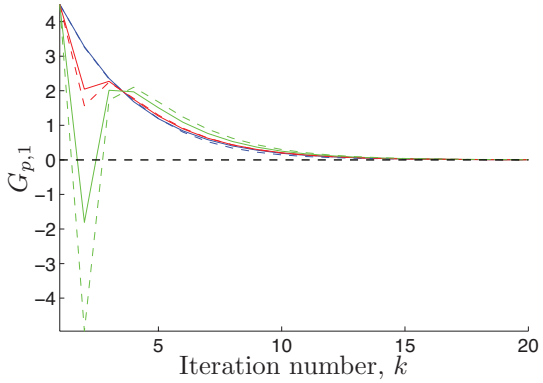

Fig. 7. The constraint on X_A as a function of the iteration number k. Blue/red/green = Cases I/II/III. Solid = Method A, Dashed = Method B. The dotted line indicates $G_{p,1} = 0$.

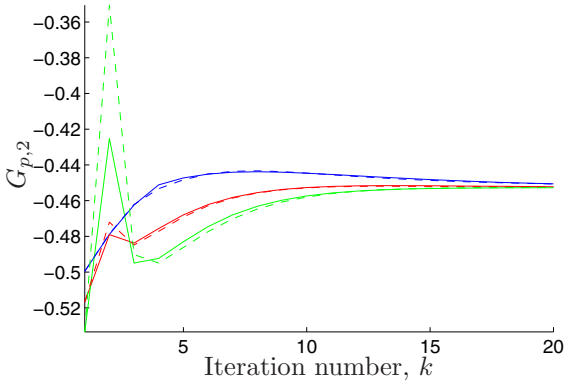

Fig. 8. The constraint on X_G as a function of the iteration number k. Blue/red/green = Cases I/II/III. Solid = Method A, Dashed = Method B.

can satisfy operational constraints and ensure optimality upon convergence. This can be 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, 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). On the other hand, 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.

Even if a feasible operating point is guaranteed to be reached upon convergence, the iterates may follow an infeasible path, as seen in the simulated example. One can combine modifier adaptation with on-line control to enforce constraint satisfaction at all times (Marchetti et al., 2011). More work in that direction is needed.

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., François, 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.

Chachuat, B., Srinivasan, B., and Bonvin, D. (2009). Adaptation strategies for real-time optimization. *Comp. Chem. Eng.*, 33(10), 1557–1567.

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). Measurement-based real-time optimization of chemical processes. In S. Pushpavanam (ed.), *Control and Optimisation of Process Systems*, volume 43, 1–50. Academic Press, Oxford.

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. (2009). Modifier-adaptation methodology for real-time optimization. *Ind. Eng. Chem. Res.*, 48(13), 6022–6033.

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., Gopalakrishnan, A., Chachuat, B., Bonvin, D., Tsikonis, L., Nakajo, A., Wuillemin, Z., and van Herle, J. (2011). Robust real-time optimization of a solid oxide fuel cell stack. *J. Fuel Cell Sci. Technol.*, 8(5), (11 pages).

Marlin, T. and Hrymak, A.N. (1996). Real-time operations optimization of continuous processes. In *Chemical Process Control - CPC-V*. Tahoe City, NV.

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.