Process Systems Engineering

Modifier Adaptation as a Feedback Control Scheme

Alejandro G. Marchetti, Tafarel de Avila Ferreira, Sean C. Costello, and Dominique Bonvin

Ind. Eng. Chem. Res., **Just Accepted Manuscript** • DOI: 10.1021/acs.iecr.9b04501 • Publication Date (Web): 11 Jan 2020

Modifier Adaptation as a Feedback Control Scheme

A. G. Marchetti[a], T. de Avila Ferreira[1], S. Costello[2], D. Bonvin[b,*]

[a] *CIFASIS (CONICET - Universidad Nacional de Rosario), S2000EZP Rosario, Argentina.*
[b] *Laboratoire d'Automatique, Ecole Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland.*

Abstract

As a real-time optimization technique, modifier adaptation (MA) has gained much significance in recent years. This is mainly due to the fact that MA can deal explicitly with structural plant-model mismatch and unknown disturbances. MA is an iterative technique that is ideally suited to real-life applications. Its two main features are the way measurements are used to correct the model and the role played by the model in actually computing the next inputs.

This paper analyzes these two features and shows that, although MA computes the next inputs via numerical optimization, it can be viewed as a *feedback control scheme*, with optimization implementing feedback to match the plant KKT conditions. As a result, the role of the model is downplayed to the point that *model accuracy* is not an important issue. The key issues are gradient estimation and *model adequacy*, the latter requiring that the model possesses the correct curvature of the cost function at the plant optimum. The main role of optimization is to identify the proper set of controlled variables (the active constraints and reduced gradients) as these might change with the operating point and disturbances. Thanks to this reduced requirement on model accuracy, MA is ideally suited to drive real-life processes to optimality. This is illustrated through two experimental systems with very different optimization features, namely, a commercial fuel-cell system and an experimental kite setup for harnessing wind energy.

Keywords: Real-time optimization; Plant-model mismatch; Constraint adaptation; Modifier adaptation; Model accuracy; Model adequacy.

[1] Tafarel de Avila Ferreira was with the Laboratoire d'Automatique, EPFL, Switzerland. He is currently with Group of Energy Materials, EPFL, Sion, Switzerland.

[2] Sean Costello was with the Laboratoire d'Automatique, EPFL, Switzerland. He is currently with Leica Geosystems, St-Gallen, Switzerland.

* Corresponding author (dominique.bonvin@epfl.ch)

1. Introduction

Optimization of real-life processes is a difficult task that requires a significant amount of information for successful implementation. This is particularly difficult in the presence of uncertainty, which has three main sources, namely, (i) parametric uncertainty when the values of the model parameters do not correspond to the reality of the plant; (ii) structural plant-model mismatch when the structure of the model is incorrect, for example in the case of unknown phenomena or neglected dynamics; and (iii) unmeasured process disturbances. Of course these three sources are not mutually exclusive.

Real-time optimization (RTO) has emerged over the past forty years to overcome these difficulties. RTO incorporates process measurements in the optimization framework to combat the detrimental effect of uncertainty. The most common strategy is to use measurements and improve the model via parameter adjustment. Then, the updated model is used to compute optimal inputs. Since this correction is local at the current operating point, this "two-step approach" of parameter estimation and process optimization needs to be repeated until convergence is reached [9, 23]. However, convergence to plant optimality is not guaranteed, as the two-step approach is not designed to deal with structural plant-model mismatch. The difficulty of converging to plant optimality motivated the development of a modified two-step approach, referred to as "integrated system optimization and parameter estimation" (ISOPE) [5, 25]. For problems without process-dependent constraints, ISOPE incorporates plant-gradient information in the cost function of the optimization problem such that the Karush-Kuhn-Tucker (KKT) optimality conditions *for the plant* are satisfied upon convergence. Later, the ISOPE algorithm was simplified through elimination of the parameter estimation step [29]. Gao and Engell [19] extended that approach to problems with process-dependent constraints by including first-order correction terms to the constraints. Marchetti et al. [20] proposed to use the same type of affine corrections in both the cost and constraint functions and labeled the approach "modifier adaptation" (MA). MA can be viewed as an extension to the *constraint-adaptation* (CA) approach that consists in shifting the constraints in the model-based optimization problem [7, 16], or, conversely, CA can be seen as a special case of MA, whereby the KKT conditions encompass only active constraints. A comprehensive analysis of the main MA properties, such as KKT matching upon convergence, model-adequacy conditions, and necessary conditions for local asymptotic convergence can be found in [22]. Several variants of

modifier adaptation have been proposed, such as *dual* modifier adaptation [21], *directional* modifier adaptation [11], *nested* modifier adaptation [24], and *second-order* modifier adaptation [15].

The paper analyzes CA and MA and shows that these two schemes based on repeated optimization can be interpreted as feedback control schemes with integral action. Hence, they can reach plant optimality (offset-free RTO) even in the presence of a significant amount of structural plant-model mismatch. Furthermore, the paper investigates the role played by the model to reach plant optimality. It is argued that the key model property is model adequacy and not model accuracy. Based on these investigations, it is possible to provide guidelines for implementation. These guidelines include the choice of the RTO scheme (CA or MA), the selection of a small number of modifiers, ways of estimating the static plant gradients as well as some factors that affect the convergence speed. The performance of CA and MA is illustrated via two experimental studies, namely, a commercial fuel-cell system, the optimality of which is fully determined by active constraints (CA), and an experimental kite setup for harnessing wind energy that has no active constraint (MA).

The paper is organized as follows. Section 2 formulates the RTO problem, presents the corresponding necessary conditions of optimality (NCO), and briefly revisits the CA and MA schemes. In Section 3, CA and MA are analyzed and formulated as feedback control schemes. Section 4 discusses model adequacy and model accuracy requirements for both CA and MA. Section 5 presents some guidelines for MA implementation. Two experimental case studies, namely, a fuel-cell system and a kite for harnessing wind energy, are presented in Section 6, while Section 7 concludes the paper.

2. Preliminaries

2.1. Optimization Problem

The purpose of static real-time optimization (RTO) is to optimize process operation by finding the solution to the following optimization problem:

$$\begin{aligned} \min_{\mathbf{u}} \quad & \Phi_p(\mathbf{u}) := \phi(\mathbf{u}, \mathbf{y}_p(\mathbf{u})) \\ \text{s.t.} \quad & G_{p,i}(\mathbf{u}) := g_i(\mathbf{u}, \mathbf{y}_p(\mathbf{u})) \leq 0, \quad i = 1, \ldots, n_g, \end{aligned} \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^{n_u}$ denotes the decision (or input) variables; $\mathbf{y}_p \in \mathbb{R}^{n_y}$ are the measured output variables; $\phi : \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}$ is the cost function to be minimized; $g_i : \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}$,

$i = 1, \ldots, n_g$, is the set of process-dependent inequality constraints. The notation $(\cdot)_p$ is used throughout for variables associated with the plant.

This formulation assumes that ϕ and g_i are known functions of $\mathbf{u}$ and $\mathbf{y}_p$, that is, they can be directly measured or evaluated from knowledge of $\mathbf{u}$ and measurement of $\mathbf{y}_p$. However, the steady-state input-output mapping of the plant $\mathbf{y}_p(\mathbf{u})$ is typically unknown, and only the approximate nonlinear model $\mathbf{y}(\mathbf{u})$ is available.

The model-based counterpart of Problem (1) is given by the following nonlinear program (NLP):

$$\begin{aligned} \min_{\mathbf{u}} \quad & \Phi(\mathbf{u}) := \phi(\mathbf{u}, \mathbf{y}(\mathbf{u})) \\ \text{s.t.} \quad & G_i(\mathbf{u}) := g_i(\mathbf{u}, \mathbf{y}(\mathbf{u})) \leq 0, \qquad i = 1, \ldots, n_g. \end{aligned} \tag{2}$$

In the presence of plant-model mismatch, the solution to Problem (2) does not generally match the solution to Problem (1).

In real-time optimization, the solution to Problem (1) is approached by iteratively re-evaluating the inputs applied to the plant. Let $\mathbf{u}_k$ denote the inputs applied to the plant at the k^{th} RTO iteration. The next optimal inputs are obtained by solving the following model-based optimization problem:

$$\begin{aligned} \mathbf{u}_{k+1}^{\star} \in \arg\min_{\mathbf{u}} \quad & \Phi_k(\mathbf{u}) \\ \text{s.t.} \quad & G_{i,k}(\mathbf{u}) \leq 0, \qquad i = 1, \ldots, n_g, \end{aligned} \tag{3}$$

where $\Phi_k(\mathbf{u})$ is the cost function and $G_{i,k}(\mathbf{u})$, $i = 1, \ldots, n_g$, the constraint functions at the k^{th} RTO iteration. In order to deal with plant-model mismatch, $\Phi_k(\mathbf{u})$ and $G_{i,k}(\mathbf{u})$, $i = 1, \ldots, n_g$, are typically updated at each RTO iteration on the basis of collected measurements. Examples of updating strategies include the computation of new model parameters, as in two-step approaches [9], and the computation of additive correction terms for the cost and constraint functions, as in modifier-adaptation schemes [20].

2.2. Necessary Conditions of Optimality

Local minima of Problems (1) and (2) are characterized by the NCO [1]. For example, for $\mathbf{u}^{\star}$ to be a (local) minimum of Eq. (2), there must exist a vector $\boldsymbol{\nu}^{\star} = [\nu_1^{\star}, \ldots, \nu_{n_g}^{\star}]^{\mathsf{T}}$ such that the

following Karush-Kuhn-Tucker (KKT) conditions hold [1]:

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}^\star) + \sum_{i=1}^{n_g} \nu_i^\star \frac{\partial G_i}{\partial \mathbf{u}}(\mathbf{u}^\star) = \mathbf{0} \tag{4a}$$

$$G_i(\mathbf{u}^\star) \leq 0, \qquad \forall i \in \{1, \ldots, n_g\} \tag{4b}$$

$$\nu_i^\star \, G_i(\mathbf{u}^\star) = 0, \qquad \forall i \in \{1, \ldots, n_g\} \tag{4c}$$

$$\nu_i^\star \geq 0, \qquad \forall i \in \{1, \ldots, n_g\}. \tag{4d}$$

Since $\nu_i^\star = 0$ for the inactive constraint $G_i(\mathbf{u}^\star) < 0$, one can write Eq. (4) by considering only the n_g^a active constraints:

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}^\star) + \sum_{i=1}^{n_g^a} \nu_i^{a^\star} \frac{\partial G_i^a}{\partial \mathbf{u}}(\mathbf{u}^\star) = \mathbf{0} \tag{5a}$$

$$G_i^a(\mathbf{u}^\star) = 0, \quad \forall i \in \{1, \ldots, n_g^a\} \tag{5b}$$

$$\nu_i^{a^\star} \geq 0, \quad \forall i \in \{1, \ldots, n_g^a\}, \tag{5c}$$

where the superscript $(\cdot)^a$ denotes a quantity associated with the active constraints.

Eq. (5a) indicates that the cost gradient must be aligned with a linear combination of the gradients of the active constraints. Assuming that the gradients of the active constraints are linearly independent at the model optimum $\mathbf{u}^\star$, one can write:

$$\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}^\star)\, \mathbf{N} = \mathbf{0}, \tag{6}$$

where $\mathbf{N} \in \mathbb{R}^{n_u \times (n_u - n_g^a)}$ is a null-space matrix defined by Eq. (6). Post-multiplying Eq. (5a) by $\mathbf{N}$ gives:

$$\nabla_r \Phi(\mathbf{u}^\star) := \frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}^\star)\, \mathbf{N} = \mathbf{0}, \tag{7}$$

where $\nabla_r \Phi(\mathbf{u}^\star) \in \mathbb{R}^{1 \times (n_u - n_g^a)}$ is the reduced cost gradient of optimization Problem (2).[3] Hence, if the set of active constraints is known, the KKT conditions can be expressed in terms of n_g^a active constraints and $(n_u - n_g^a)$ reduced gradients as follows:

$$\mathbf{G}^a(\mathbf{u}^\star) = \mathbf{0}, \tag{8a}$$

$$\nabla_r \Phi(\mathbf{u}^\star) = \mathbf{0}. \tag{8b}$$

[3] This gradient can also be written $\nabla_r \Phi(\mathbf{u}^\star) = \frac{d\Phi}{d\mathbf{u}}\big|_{\mathbf{G}^a(\mathbf{u}^\star)=\mathbf{0}}(\mathbf{u}^\star)$ since it represents the steady-state cost variation with respect to $\mathbf{u}$ when the active constraints are met.

2.3. Constraint Adaptation

In CA, zero-order correction terms are added to the constraint functions of the optimization problem [7, 16]. At the k^{th} iteration with the inputs $\mathbf{u}_k$, the zero-order modifiers $\boldsymbol{\varepsilon}_k^{\mathbf{G}} \in \mathbb{R}^{n_g}$ are evaluated as follows:

$$\varepsilon_k^{G_i} := G_{p,i}(\mathbf{u}_k) - G_i(\mathbf{u}_k), \quad i = 1, \ldots, n_g, \tag{9}$$

which correspond to bias terms representing the differences between the plant values and the predicted values of the constrained variables at $\mathbf{u}_k$. Note that, if the constraints are *known* functions of the inputs $\mathbf{u}$, the corresponding modifiers are zero since no model correction is necessary. For example, the input upper and lower bounds are constraints that are perfectly known.

At the k^{th} RTO iteration, the next optimal inputs $\mathbf{u}_{k+1}^{\star}$ are computed by solving the following modified optimization problem:

$$\mathbf{u}_{k+1}^{\star} \in \arg\min_{\mathbf{u}} \ \Phi(\mathbf{u}) \tag{10a}$$

$$\text{s.t.} \quad G_i(\mathbf{u}) + \varepsilon_k^{G_i} \leq 0, \qquad i = 1, \ldots, n_g, \tag{10b}$$

and then applied to the plant:

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1}^{\star}. \tag{11}$$

However, the update strategy Eq. (11) may result in excessive correction, which could compromise the convergence of the algorithm. Hence, one usually relies on first-order filters that are applied to either the modifiers or the inputs. In the former case, one updates the modifiers using the following first-order filter equations [20]:

$$\boldsymbol{\varepsilon}_k^{\mathbf{G}} = (\mathbf{I}_{n_g} - \mathbf{K}^{\varepsilon})\boldsymbol{\varepsilon}_{k-1}^{\mathbf{G}} + \mathbf{K}^{\varepsilon}\big(\mathbf{G}_p(\mathbf{u}_k) - \mathbf{G}(\mathbf{u}_k)\big), \tag{12}$$

where the filter matrix $\mathbf{K}^{\varepsilon}$ is typically selected as a diagonal matrix with eigenvalues in the interval $(0, 1]$. In the latter case, one filters the optimal inputs $\mathbf{u}_{k+1}^{\star}$ with $\mathbf{K} \in \text{diag}(k_1, \ldots, k_{n_u})$, $k_i \in (0, 1]$:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{K}(\mathbf{u}_{k+1}^{\star} - \mathbf{u}_k). \tag{13}$$

The appeal of CA lies both in its simplicity and in the fact that it can provide a satisfactory solution to a large number to optimization problems of industrial processes, for which most of the optimization potential lies in keeping a set of constraints active.

2.4. Modifier Adaptation

If the NCO encompass both active constraints and reduced gradients, one can use MA to adjust the inputs and meet the plant NCO. This is done by adding first-order correction terms to the cost and constraint functions of the optimization problem [20]. At the k^{th} iteration with the inputs $\mathbf{u}_k$, the modified cost and constraint functions are constructed as follows:

$$\Phi_{m,k}(\mathbf{u}) := \Phi(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\Phi})^{\mathsf{T}}(\mathbf{u} - \mathbf{u}_k) \tag{14}$$

$$G_{m,i,k}(\mathbf{u}) := G_i(\mathbf{u}) + \varepsilon_k^{G_i} + (\boldsymbol{\lambda}_k^{G_i})^{\mathsf{T}}(\mathbf{u} - \mathbf{u}_k) \leq 0, \quad i = 1, \ldots, n_g, \tag{15}$$

with the modifiers $\varepsilon_k^{G_i} \in \mathbb{R}$, $\boldsymbol{\lambda}_k^{\Phi} \in \mathbb{R}^{n_u}$, and $\boldsymbol{\lambda}_k^{G_i} \in \mathbb{R}^{n_u}$ given by

$$\varepsilon_k^{G_i} := G_{p,i}(\mathbf{u}_k) - G_i(\mathbf{u}_k), \quad i = 1, \ldots, n_g, \tag{16a}$$

$$(\boldsymbol{\lambda}_k^{\Phi})^{\mathsf{T}} := \frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k), \tag{16b}$$

$$(\boldsymbol{\lambda}_k^{G_i})^{\mathsf{T}} := \frac{\partial G_{p,i}}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial G_i}{\partial \mathbf{u}}(\mathbf{u}_k), \quad i = 1, \ldots, n_g. \tag{16c}$$

The zero-order modifiers $\varepsilon_k^{G_i}$ correspond to bias terms representing the differences between the plant and model values at $\mathbf{u}_k$, whereas the first-order modifiers $\boldsymbol{\lambda}_k^{\Phi}$ and $\boldsymbol{\lambda}_k^{G_i}$ represent the differences between the plant and model gradients at $\mathbf{u}_k$. The plant gradients $\frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k)$ and $\frac{\partial G_{p,i}}{\partial \mathbf{u}}(\mathbf{u}_k)$ are assumed to be available at $\mathbf{u}_k$. Again, if the cost and constraints are *known* functions of the inputs $\mathbf{u}$, the corresponding modifiers are zero since no model correction is necessary.

At the k^{th} RTO iteration, the next optimal inputs $\mathbf{u}_{k+1}^{\star}$ are computed by solving the following modified optimization problem:

$$\mathbf{u}_{k+1}^{\star} \in \arg\min_{\mathbf{u}} \ \Phi_{m,k}(\mathbf{u}) := \Phi(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\Phi})^{\mathsf{T}}\mathbf{u} \tag{17a}$$

$$\text{s.t.} \quad G_{m,i,k}(\mathbf{u}) := G_i(\mathbf{u}) + \varepsilon_k^{G_i} + (\boldsymbol{\lambda}_k^{G_i})^{\mathsf{T}}(\mathbf{u} - \mathbf{u}_k) \leq 0, \qquad i = 1, \ldots, n_g, \tag{17b}$$

and then applied to the plant:

$$\mathbf{u}_{k+1} = \mathbf{u}_{k+1}^{\star}. \tag{18}$$

However, since the model is only locally valid, the update strategy Eq. (18) may result in excessive correction, which could compromise the convergence of the algorithm. Hence, one usually relies on first-order filters that are applied to either the modifiers or the inputs. In the former case,

one updates the modifiers using the following first-order filter equations [20]:

$$\boldsymbol{\varepsilon}_k^{\mathbf{G}} = (\mathbf{I}_{n_g} - \mathbf{K}^{\varepsilon})\boldsymbol{\varepsilon}_{k-1}^{\mathbf{G}} + \mathbf{K}^{\varepsilon}\left(\mathbf{G}_p(\mathbf{u}_k) - \mathbf{G}(\mathbf{u}_k)\right), \tag{19a}$$

$$\boldsymbol{\lambda}_k^{\Phi} = (\mathbf{I}_{n_u} - \mathbf{K}^{\Phi})\boldsymbol{\lambda}_{k-1}^{\Phi} + \mathbf{K}^{\Phi}\left(\frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k)\right)^{\mathsf{T}}, \tag{19b}$$

$$\boldsymbol{\lambda}_k^{G_i} = (\mathbf{I}_{n_u} - \mathbf{K}^{G_i})\boldsymbol{\lambda}_{k-1}^{G_i} + \mathbf{K}^{G_i}\left(\frac{\partial G_{p,i}}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial G_i}{\partial \mathbf{u}}(\mathbf{u}_k)\right)^{\mathsf{T}}, \quad i = 1, \ldots, n_g, \tag{19c}$$

where the filter matrices $\mathbf{K}^{\varepsilon}$, $\mathbf{K}^{\Phi}$, and $\mathbf{K}^{G_i}$ are typically selected as diagonal matrices with eigenvalues in the interval $(0, 1]$. In the latter case, one filters as per Eq. (13). The appeal of MA lies in its ability to reach a KKT point of the plant upon convergence, as expressed in the following theorem.

Theorem 1 (KKT matching upon convergence [20])**.** *Consider MA with filters on either the modifiers or the inputs. Let* $\mathbf{u}_\infty = \lim_{k\to\infty} \mathbf{u}_k$ *be a fixed point of the repeated modified optimization Problem* (17)*. Then,* $\mathbf{u}_\infty$ *is a KKT point of the plant Problem* (1)*.*

3. Modifier Adaptation = Feedback Control via Repeated Optimization

Feedback control can achieve offset-free behavior in the absence of a process model. This is well understood and accepted by industrial practitioners. However, when it comes to process optimization, there is the preconception that a fairly accurate model is required. This preconception is justified, as there are many examples in industry where the optimization performance depends strongly on the accuracy of the model at hand [26]. Also, for the classical two-step approach used in RTO, the accuracy of the model and the ability to estimate online the uncertain model parameters are key factors in achieving optimal process operations [6, 30]. Although the two-step approach uses online measurements to update the model, the measurements and the adaptation strategy are not designed to cope with structural plant-model mismatch [8].

On the other hand, MA seems to have a few features that make it very appealing for practical applications (see [22] for an overview of MA case studies):

- Model correction is targeted to reaching plant optimality. KKT matching is enforced locally via affine corrections to the cost and constraint functions. This way, in MA, the model is *locally perfect for optimization*!

- The approach is rather insensitive to model accuracy. This may sound surprising, but it is related to the role played by the model and the modifiers in MA. While most RTO schemes update the model and compute a KKT point *for the model*, MA modifies the nominal model so as to be able to enforce a KKT point *for the plant*. This requires (i) determining the correct set of active constraints at the plant optimum, and (ii) matching the reduced plant gradient. Since there are as many KKT conditions to satisfy as there are degrees of freedom, the MA algorithm can be viewed as an *integral feedback control scheme* that tracks the plant KKT conditions with zero offset. Hence, although MA uses numerical optimization at each iteration and has been classified as a RTO scheme that uses optimization-specific measurements and re-optimization [28], it can also be interpreted as a feedback control scheme.

In this section, we show that CA can be viewed as a feedback control scheme that tracks the active plant constraints. Similarly, MA can be viewed as a feedback control scheme that tracks the plant KKT conditions.

3.1. Constraint Adaptation as a Feedback Control Scheme

For simplicity of illustration, we consider the CA Problem (10) without filtering, that is, with the modifiers computed as $\boldsymbol{\varepsilon}_k^{\mathbf{G}} = \mathbf{G}_p(\mathbf{u}_k) - \mathbf{G}(\mathbf{u}_k)$ and the optimal inputs $\mathbf{u}_{k+1}^\star$ applied to the plant as per Eq. (11). Upon elimination of the inactive constraints, the NCO of Problem (10) read:

$$\mathbf{G}^a(\mathbf{u}) + \boldsymbol{\varepsilon}_k^{\mathbf{G}^a} = \mathbf{0}, \tag{20a}$$

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}) + (\boldsymbol{\nu}^a)^\mathsf{T} \frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}) = \mathbf{0}. \tag{20b}$$

These conditions are satisfied at the optimum $\mathbf{u}_{k+1}^\star$, and also at the next input $\mathbf{u}_{k+1}$ since no filtering is assumed. As a first-order approximation to the NCO (20), evaluated at $\mathbf{u}_{k+1}$ and $\boldsymbol{\nu}_{k+1}^a$, one can write:

$$\mathbf{G}^a(\mathbf{u}_k) + \frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \boldsymbol{\varepsilon}_k^{\mathbf{G}^a} = \mathbf{0}, \tag{21a}$$

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k) + (\mathbf{u}_{k+1} - \mathbf{u}_k)^\mathsf{T} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k) + (\boldsymbol{\nu}_{k+1}^a)^\mathsf{T} \left(\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k \right) = \mathbf{0}, \tag{21b}$$

where the i^{th} row of matrix $\mathbf{M}_k \in \mathbb{R}^{n_g^a \times n_u}$ is given by:

$$\mathbf{m}_{i,k}^\mathsf{T}(\mathbf{u}_{k+1}) = (\mathbf{u}_{k+1} - \mathbf{u}_k)^\mathsf{T} \frac{\partial^2 G_i^a}{\partial \mathbf{u}^2}(\mathbf{u}_k). \tag{22}$$

Assuming that the active constraints are linearly independent, one can compute the null space of the matrix $\left(\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k\right)$ as $\mathbf{N}_k \in \mathbb{R}^{n_u \times (n_u - n_g^a)}$ such that:

$$\left(\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k\right) \mathbf{N}_k = \mathbf{0}. \tag{23}$$

Using $\boldsymbol{\varepsilon}_k^{\mathbf{G}^a} = \mathbf{G}_p^a(\mathbf{u}_k) - \mathbf{G}^a(\mathbf{u}_k)$ in Eq. (21a), and post-multiplying Eq. (21b) by $\mathbf{N}_k$ and transposing gives:

$$\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \mathbf{G}_p^a(\mathbf{u}_k) = \mathbf{0}, \tag{24a}$$

$$\mathbf{N}_k^\mathsf{T} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \mathbf{N}_k^\mathsf{T} \left(\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k)\right)^\mathsf{T} = \mathbf{0}, \tag{24b}$$

which allows eliminating the Lagrange multipliers $\boldsymbol{\nu}_{k+1}^a$ from the NCO. The next inputs $\mathbf{u}_{k+1}$ can be computed from Eq. (24) as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \begin{pmatrix} \frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k) \\ \mathbf{N}_k^\mathsf{T} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k) \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{G}_p^a(\mathbf{u}_k) \\ \nabla_r \Phi(\mathbf{u}_k) \end{pmatrix}, \tag{25}$$

where $\nabla_r \Phi(\mathbf{u}_k)^\mathsf{T} = \frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k)\mathbf{N}_k$ is the reduced cost gradient predicted by the model. Eq. (25) corresponds to a run-to-run feedback control law that (i) enforces the active plant constraints, and (ii) drives the reduced gradient predicted by the model to zero.

3.2. Modifier Adaptation as a Feedback Control Scheme

For simplicity of illustration, we consider the MA Problem (17) without filtering, that is, with the modifiers computed as per Eq. (16) and the optimal inputs $\mathbf{u}_{k+1}^\star$ applied to the plant as per Eq. (18). Upon elimination of the inactive constraints, the NCO of Problem (17) read:

$$\mathbf{G}^a(\mathbf{u}) + \boldsymbol{\varepsilon}_k^{\mathbf{G}^a} + (\boldsymbol{\lambda}_k^{\mathbf{G}^a})^\mathsf{T}(\mathbf{u} - \mathbf{u}_k) = \mathbf{0}, \tag{26a}$$

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}) + (\boldsymbol{\lambda}_k^\Phi)^\mathsf{T} + (\boldsymbol{\nu}^a)^\mathsf{T} \left(\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\mathbf{G}^a})^\mathsf{T}\right) = \mathbf{0}. \tag{26b}$$

As a first-order approximation to the NCO (26), evaluated at $\mathbf{u}_{k+1}$ and $\boldsymbol{\nu}_{k+1}^a$, one can write:

$$\mathbf{G}^a(\mathbf{u}_k) + \frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \boldsymbol{\varepsilon}_k^{\mathbf{G}^a} + (\boldsymbol{\lambda}_k^{\mathbf{G}^a})^\mathsf{T}(\mathbf{u}_{k+1} - \mathbf{u}_k) = \mathbf{0}, \tag{27a}$$

$$\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k) + (\mathbf{u}_{k+1} - \mathbf{u}_k)^\mathsf{T} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k) + (\boldsymbol{\lambda}_k^\Phi)^\mathsf{T} + (\boldsymbol{\nu}_{k+1}^a)^\mathsf{T} \left(\frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k + (\boldsymbol{\lambda}_k^{\mathbf{G}^a})^\mathsf{T}\right) = \mathbf{0}. \tag{27b}$$

Using $\varepsilon_k^{\mathbf{G}^a} = \mathbf{G}_p^a(\mathbf{u}_k) - \mathbf{G}^a(\mathbf{u}_k)$, $(\boldsymbol{\lambda}_k^{\Phi})^{\mathsf{T}} = \frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_k)$ and $(\boldsymbol{\lambda}_k^{\mathbf{G}^a})^{\mathsf{T}} = \frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k) - \frac{\partial \mathbf{G}^a}{\partial \mathbf{u}}(\mathbf{u}_k)$ in Eq. (27) gives:

$$\mathbf{G}_p^a(\mathbf{u}_k) + \frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) = \mathbf{0}, \tag{28a}$$

$$\frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k) + (\mathbf{u}_{k+1} - \mathbf{u}_k)^{\mathsf{T}} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k) + (\boldsymbol{\nu}_{k+1}^a)^{\mathsf{T}} \left(\frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k \right) = \mathbf{0}. \tag{28b}$$

Assuming that the active constraints are linearly independent, one can compute the null space of the matrix $\left(\frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k \right)$ as $\mathbf{N}_k \in \mathbb{R}^{n_u \times (n_u - n_g^a)}$ such that:

$$\left(\frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k) + \mathbf{M}_k \right) \mathbf{N}_k = \mathbf{0}. \tag{29}$$

Post-multiplying Eq. (28b) by $\mathbf{N}_k$ and transposing gives:

$$\frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \mathbf{G}_p^a(\mathbf{u}_k) = \mathbf{0}, \tag{30a}$$

$$\mathbf{N}_k^{\mathsf{T}} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k)(\mathbf{u}_{k+1} - \mathbf{u}_k) + \mathbf{N}_k^{\mathsf{T}} \left(\frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k) \right)^{\mathsf{T}} = \mathbf{0}, \tag{30b}$$

which allows eliminating the Lagrange multipliers $\boldsymbol{\nu}_{k+1}^a$ from the NCO. The next inputs $\mathbf{u}_{k+1}$ can be computed from Eq. (30) as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \begin{pmatrix} \frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k) \\ \mathbf{N}_k^{\mathsf{T}} \frac{\partial^2 \Phi}{\partial \mathbf{u}^2}(\mathbf{u}_k) \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{G}_p^a(\mathbf{u}_k) \\ \nabla_r \Phi_p(\mathbf{u}_k) \end{pmatrix}, \tag{31}$$

where $\nabla_r \Phi_p(\mathbf{u}_k)^{\mathsf{T}} = \frac{\partial \Phi_p}{\partial \mathbf{u}}(\mathbf{u}_k)\mathbf{N}_k$ is the reduced cost gradient of the plant. Eq. (31) corresponds to a run-to-run feedback control law that (i) enforces the active plant constraints, and (ii) drives the reduced gradient of the plant to zero.

This analysis has shown that MA can be viewed as a run-to-run integral feedback control scheme that tracks the plant NCO with zero offset. Note that the feedback control law (31) has been derived directly from the first-order approximation to the NCO of Problem (17).

Remark 1. *Despite the strong similarity between the feedback laws (25) and (31), there is a subtle but major difference: the feedback law associated with CA controls the active plant constraints and the reduced cost gradient predicted by the model, whereas the feedback law associated with MA controls the active plant constraints and the reduced cost gradient of the plant.*

Remark 2. *CA and MA correspond to integral feedback laws, the gains of which are given by the inverted matrices in Eqs. (25) and (31). The CA gains are purely model-based, whereas they are both model- and measurement-based in MA.*

Remark 3. *Since* $\mathbf{M}_k$ *depends on* $\mathbf{u}_{k+1}$ *as per Eq.* (22)*, the computation of the null space matrix* $\mathbf{N}_k$ *requires the knowledge of* $\mathbf{u}_{k+1}$*. Hence, for a practical implementation of the control scheme* (31)*, one can avoid using* $\mathbf{M}_k$ *and compute* $\mathbf{N}_k$ *as the null space of the Jacobian matrix* $\frac{\partial \mathbf{G}_p^a}{\partial \mathbf{u}}(\mathbf{u}_k)$*. However, the purpose here is not to replace MA by a feedback control scheme, but rather to show that the solution* $\mathbf{u}_{k+1}$ *obtained by solving the modified optimization Problem* (17) *can be interpreted as the inputs computed by a feedback controller that tracks the plant NCO. Solving optimization Problem* (17) *has the advantage that the active constraints at the plant optimum need not to be known a priori.*

4. Model Adequacy vs. Model Accuracy

The reformulation of CA and MA as feedback control schemes helps understand that these schemes can achieve offset-free behavior in the absence of an accurate model. In this section, we address two important optimization-related model properties, namely, model adequacy and model accuracy.

4.1. Model Adequacy

The question of whether a model is adequate for use in an RTO scheme was addressed by Forbes and Marlin [17], who proposed the following point-wise model-adequacy definition:

Definition 1 (Model adequacy)**.** *A process model is said to be adequate if it is capable of producing a fixed point of the RTO algorithm at the plant optimum* $\mathbf{u}_p^\star$*.*

In other words, the plant optimum $\mathbf{u}_p^\star$ must be a local minimum of the RTO problem when the underlying model is updated at $\mathbf{u}_p^\star$.

4.1.1. Model adequacy for CA

In the presence of plant-model mismatch, constraint adaptation does not, in general, have the ability to converge to plant optimality, except if there are, at the plant optimum, as many independent active constraints as input variables. For this case, the model-adequacy conditions proposed by Forbes and Marlin [16] are given in the following proposition:

Proposition 1 (Model-adequacy conditions for CA [16])**.** *Let* $\mathbf{u}_p^\star$ *be a regular point for the constraints and the unique plant optimum, and let the number of active constraints at* $\mathbf{u}_p^\star$ *be equal to*

the number of inputs. A candidate model is said to be adequate for use in the CA scheme if the following conditions are satisfied:

$$\sum_{i=1}^{n_g^a} \nu_i^a \frac{\partial G_i^a}{\partial \mathbf{u}}(\mathbf{u}_p^\star) = -\frac{\partial \Phi}{\partial \mathbf{u}}(\mathbf{u}_p^\star) \tag{32a}$$

$$\nu_i^a \geq 0, \quad \forall i \in \{1, \ldots, n_g^a\}, \quad \textit{with} \;\; n_g^a = n_u. \tag{32b}$$

In other words, the negative of the cost gradient should belong to the convex cone generated by the gradients of the active constraints. These conditions ensure that the gradient errors in the cost and constraint functions predicted by the model do not change the set of active constraints. As will be illustrated in Section 4.2.1, this condition can often be satisfied despite the presence of significant plant-model mismatch.

4.1.2. Model adequacy for MA

The plant optimum satisfies the first- and second-order KKT conditions of the plant optimization Problem (1). The adequacy criterion requires that $\mathbf{u}_p^\star$ also satisfies the first- and second-order KKT conditions for the modified optimization Problem (17), with the modifiers (16) evaluated at $\mathbf{u}_p^\star$. As MA matches the plant first-order KKT elements by construction, the model-adequacy conditions involve meeting the second-order conditions. That is, the reduced Hessian of the Lagrangian must be positive semi-definite at $\mathbf{u}_p^\star$. The following proposition characterizes model adequacy based on the second-order conditions.

Proposition 2 (Model-adequacy conditions for MA [20])**.** *Let $\mathbf{u}_p^\star$ be a regular point for the constraints and the unique plant optimum. Let $\nabla_r^2 \mathcal{L}(\mathbf{u}_p^\star)$ denote the reduced Hessian of the Lagrangian of Problem (17) at $\mathbf{u}_p^\star$. Then, the following statements hold:*

i If $\nabla_r^2 \mathcal{L}(\mathbf{u}_p^\star)$ is positive definite, then the process model is adequate for use in the MA scheme.

ii If $\nabla_r^2 \mathcal{L}(\mathbf{u}_p^\star)$ is not positive semi-definite, then the process model is inadequate for use in the MA scheme.

iii If $\nabla_r^2 \mathcal{L}(\mathbf{u}_p^\star)$ is positive semi-definite and singular, then the second-order conditions are not conclusive with respect to model adequacy.

4.2. Model Accuracy

This section investigates the interplay between model adequacy and model accuracy. How accurate does an adequate model need to be? Or said differently, how inaccurate can a model be and still be adequate?

4.2.1. Model accuracy for CA

In the case of optimization problems with solution at the intersection of active constraints, it is important that the model correctly predicts the constrained quantities. Instead of solving a parameter estimation problem online, which does not necessarily improve the model in the presence of structural plant-model mismatch, CA offers a simple and effective strategy for matching the plant constraints.

Consider the two-dimensional optimization problem illustrated in Figure 1a. The plant optimum is found at the intersection of the constraints $G_{p,1}$ and $G_{p,2}$. The situation using three different linear models is also illustrated in Figure 1. Models A and B, which are shown in Figures 1b and 1c, are both adequate as the negative of the cost gradient belongs to the convex cone generated by the gradients of the active constraints (shaded cone). Note that both models present significant plant-model mismatch, including mismatch between the plant and the model gradients; yet, it is possible to converge to plant optimality by simply shifting the constraints G_1 and G_2 to the plant optimum, as indicated by the green arrows. The situation for the inadequate Model C is shown in Figure 1d. In this case, all the input values belonging to the lightly shaded cone above the cost function exhibit a better cost than the plant optimum. Hence, the model is inadequate, and it is not possible to reach plant optimality by simply shifting the constraints, as indicated by the red arrows. This example shows that CA can tolerate a large amount of modeling error and still predict the correct active set, with the ability to converge to plant optimality.

Constraint adaptation can also provide a satisfactory solution to optimization problems for which the model-adequacy conditions cannot be satisfied because the optimum is not fully determined by active constraints. Indeed, as discussed in Chachuat et al. [7], it often happens that most of the optimization potential lies in keeping a set of constraints active, while there is little to gain by zeroing the reduced cost gradient.

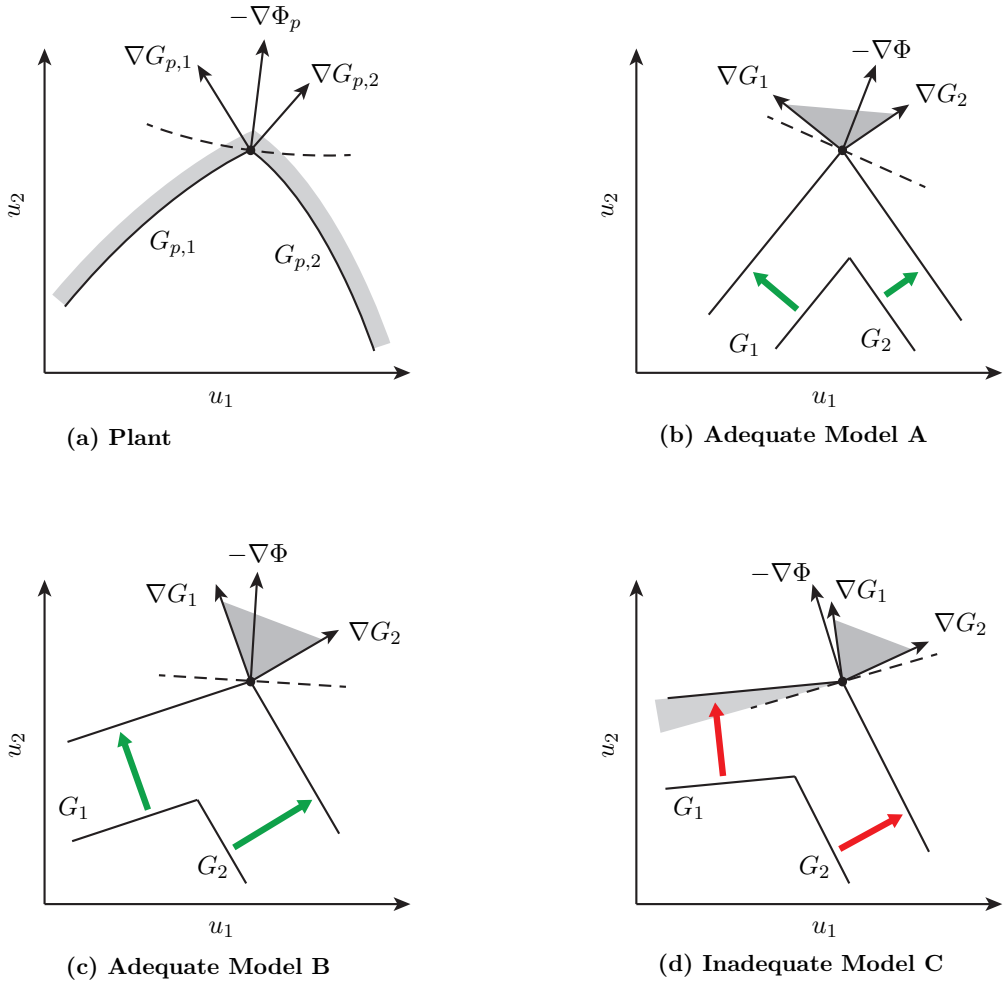


Figure 1: Illustration of model adequacy and model accuracy for CA in a two-dimensional problem. Nonlinear plant, and three different linear models. The dashed curve represents the cost function. The black dot, at which the gradients are shown, represents the plant optimum $\mathbf{u}_p^\star$. The model adequacy condition is satisfied when $-\nabla\Phi$ belongs to the shaded cone generated by ∇G_1 and ∇G_2.

4.2.2. Model accuracy for MA

MA was developed to overcome the effect of *all* types of plant-model mismatch on the KKT conditions. It does it by adding affine corrections to the cost and constraint functions as seen in Problem (17). The modifier terms are so constructed that the modified model and the plant locally share the same first-order KKT conditions [1]. The role of the model in MA can be better understood by considering two extreme cases:

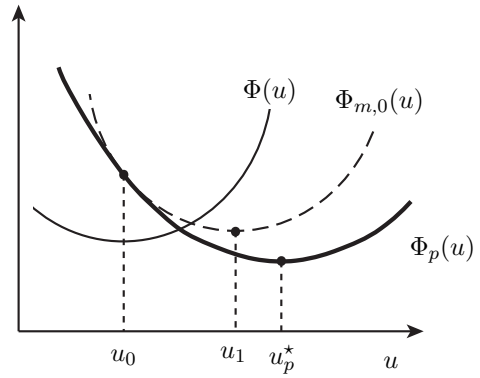
(i) In the absence of a model, or with a very elementary surrogate model, MA may still converge to plant optimality, however at the cost of a large number of iterations.

(ii) With a perfect model, the modifiers are equal to zero, and one can reach plant optimality without the need for iteration.

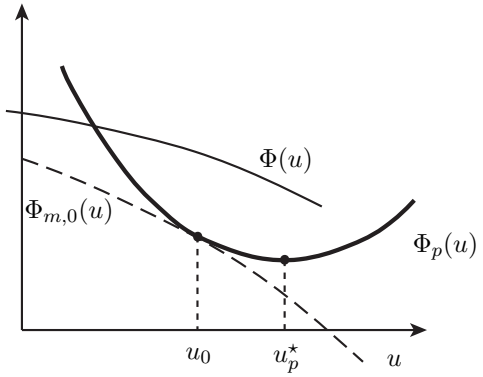
In most practical applications the available model will be somewhere between these two extreme cases. Often, the model captures the main trade-offs in the system but is not able to correctly predict the plant KKT. This is where the modifiers come in handy. With the corrections provided by the modifiers, the nominal (often inaccurate) model is made "locally perfect for optimization" at $\mathbf{u}_k$. Note that, if the model were globally perfect, optimization would compute the plant optimum right away. However, since the model is only locally perfect, one needs to take small steps (only move there where the model holds) and repeat the operation to reach plant optimality.

In MA, predicting the correct set of active constraints at the optimum is not a model-adequacy condition as this is guaranteed upon convergence. Also, note that KKT matching implies that the Lagrange multipliers of the modified model and the plant match [20]. The model-adequacy condition only states that the model should posses the right curvature at the plant optimum.

The effect of using a model with the correct or incorrect curvature can be visualized in Figure 2 for the case of a one-dimensional optimization problem. Figure 2a considers the case of the adequate Model A. At the initial point u_0, the modified model $\Phi_{m,0}$ matches the plant value and gradient. Optimization of the modified model yields the new operating point u_1, which is closer to the plant optimum $u_p^\star$ than u_0. If this procedure is repeated, plant optimality can be reached iteratively. On the other hand, the situation with Model B that does not possess the right curvature is illustrated in Figure 2b. In this case, optimization of the model modified at u_0 will result in some infinite value. If a sufficiently small trust region is defined in the neighborhood of u_0, then it is still possible



(a) Adequate Model A



(b) Inadequate Model B

Figure 2: Model adequacy and model accuracy for MA in a one-dimensional problem.

to improve the plant cost, but the model is of little use for defining this trust region.

4.3. Model Requirement

Through local corrections and iterations, one can drive an uncertain plant to optimality using a simplified model and specific measurements. In other words, even a "poor" nominal model can be made appropriate for real-time optimization. This notion of appropriateness is associated with model adequacy rather than model accuracy.

From a different perspective, one can say that CA and MA are relatively independent on model accuracy because of the presence of feedback. Feedback is implemented in two ways: (i) adapt the model and make it locally optimal for optimization, and (ii) compute the inputs via re-optimization. Note, however, that CA and MA do not work with *any* model. The model must satisfy model-adequacy conditions, which basically say that the model must predict the correct set

of active constraints and possess the right cost curvature at the *unknown* plant optimum $\mathbf{u}_p^\star$ [22]. The important point here is that the emphasis is not on (quantitative) model accuracy but rather on (qualitative) model adequacy!

5. Guidelines for Implementation

This section discusses some of the difficulties associated with the implementation of CA and MA. We will discuss successively the choice of the RTO strategy, the selection of modifiers, gradient estimation, and the optimization speed.

5.1. Constraint Adaptation vs. Modifier Adaptation

MA is designed to cope with structural plant-model mismatch. However, the mere fact that the available model presents structural uncertainty is not enough to justify the implementation of MA. One has to investigate the characteristics of the optimal solution. As mentioned in Section 4.2, CA often suffices when the solution is mostly determined by active constraints. To justify the use of MA, there must be some gain from correcting the gradients.

5.2. Selection of Modifiers

Modifiers are used to adjust the model cost and constraint functions due to the presence of uncertainty. Hence, if a constraint function does not carry any uncertainty, such as an input bound, there is no need to adjust it with modifiers. With n_u inputs, there are in principle n_g modifiers $\boldsymbol{\varepsilon}^{\mathbf{G}}$, n_u modifiers $\boldsymbol{\lambda}^{\Phi}$ and $n_g n_u$ modifiers $\boldsymbol{\lambda}^{\mathbf{G}}$. The $(n_g + 1)n_u$ first-order modifiers are the most challenging to estimate.

In setting up a MA scheme, and in order to ease the task of estimating plant gradients, one may decide to reduce the number of first-order modifiers. This can be done in two different ways:

- With n_u inputs, optimization will enforce n_u optimality conditions, namely, n_g^a active constraints and $(n_u - n_g^a)$ reduced gradients. Hence, in principle, it is only necessary to use modifiers to adjust these optimality conditions. The difficulty is that it is not known a priori which n_g^a out of n_g constraints are going to be active. Hence, modifiers are typically added to more constraints than necessary. However, there is no need to add modifiers to a constraint that is clearly not going to be active at the plant optimum!

- The number of first-order modifiers is proportional to the number of inputs n_u. It turns out that, in the n_u-dimensional input space, certains direction are more important than others. Since gradient estimation can be experimentally expensive, Costello et al. [11] proposed to reduce the burden of gradient estimation by estimating plant derivatives only along a handful of input directions called *privileged directions*. These privileged directions are determined by computing the sensitivities of the Lagrangian gradient with respect to parametric perturbations associated with the uncertainty. Singhal et al. [27] extended this concept and proposed a global sensitivity analysis that ranks the input directions according to their global effect.

5.3. Gradient Estimation

The estimation of plant gradients represents the main difficulty in implementing MA. Gradient estimation relies on either *steady-state perturbation methods* that use only steady-state data or *dynamic perturbation methods* that use transient data. These two classes of methods have been discussed by [22, 28] and the reader is invited to look at these publications and the references therein for more details.

5.4. Optimization Speed

The time-consuming task in MA (not there in CA) is really the estimation of experimental steady-state gradients. Hence, the speed for a RTO scheme to converge depends heavily on the method used for gradient estimation as discussed in the previous subsection.

5.4.1. Adaptation using steady-state measurements

When based on steady-state measurements, optimization is significantly slower than the system dynamics. For continuous processes, this means that the system dynamics need to be stable and fast compared to the time scale of the optimization, which implies that the plant can effectively be considered as a static system. For batch processes, this means that the adaptation is on a run-to-run basis with no adaptation during a given batch.

5.4.2. Adaptation using transient measurements

Since waiting for the plant to reach steady state may be inappropriate for slow processes, it has been proposed to speed up the procedure by using transient measurements instead [18]. However, the measurements need to represent the plant *steady state* that would be reached with the current

inputs. To estimate this steady state, one can process transient measurements through a dynamic model that corrects the measurements for the fact that the plant has not yet reached steady state [14]. This way, there is no need to wait for the plant to settle to steady state between successive RTO iterations, and plant optimality can be enforced in a single transient operation to steady state. More details can be found in [12].

6. Case Studies

Two experimental case studies are presented next, namely, a commercial fuel-cell system with a nominal power of 1.5 kW for electricity production and a kite for harnessing wind energy. These two systems have different optimization features: optimality of the fuel-cell system is determined by active constraints, whereas the kite setup has no active constraint. Together, these two applications illustrate the power of CA and MA in the presence of significant plant-model mismatch. Since these two studies have been published already, we will only present the basic features to illustrate the topic of this paper. For more details about the case studies, the reader is invited to look at the original references.

6.1. Fuel-cell System

This application is described in [12]. It deals with a commercial fuel-cell system that includes several units to produce electricity from methane at very high efficiency. This example is particularly interesting because (i) the optimal operating point is entirely determined by active constraints and thus RTO can by implemented via CA, and (ii) the set of active constraints changes with the operating point (in this case, the electrical power demand).

6.1.1. A commercial device

BlueGEN is a fully integrated SOFC fuel-cell module that converts fuel into electrical energy. Natural gas or biomethane can be used as the source of energy. At the electrical power of 1.5 kW and electrical efficiency of about 60%, the fuel consumption is about 2.5 kW.

The BlueGEN system includes a pre-reformer, a stack, an afterburner and several heat exchangers. The stack consists of 70 planar anode-supported cells and is operated at temperatures between 650°C and 850°C. The flowsheet is given in Figure 3. The electrical current and the flows of air, fuel and cooling air are the system inputs and represent the decision variables (or degrees of

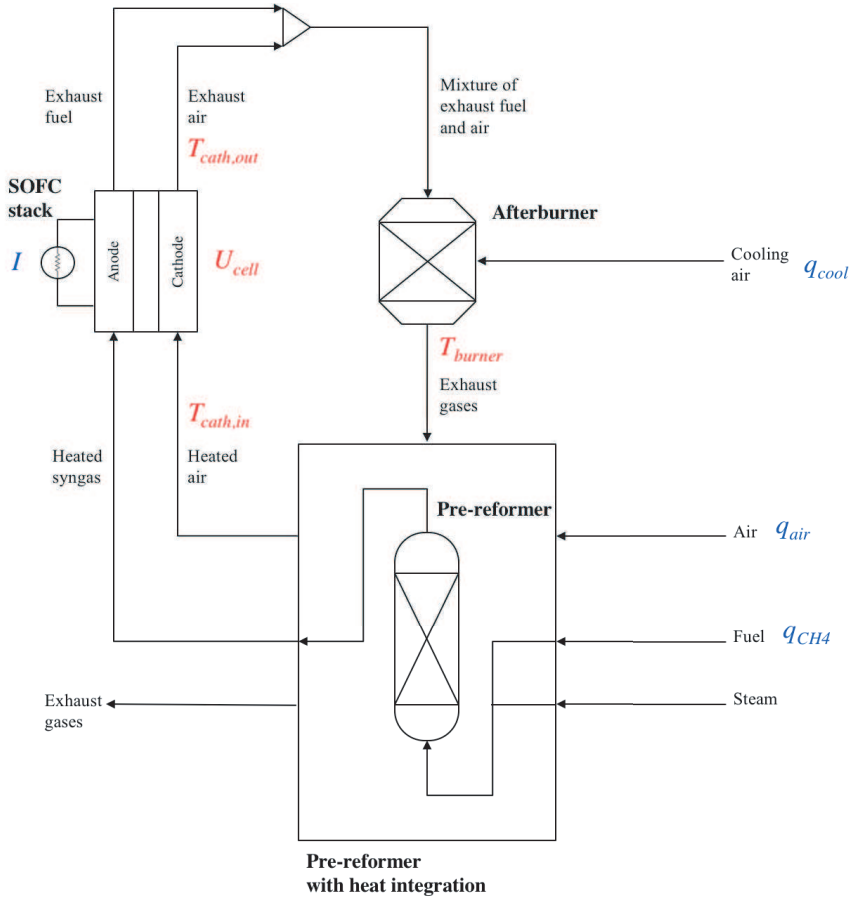


Figure 3: BlueGEN flowsheet. The blue variables represent manipulated variables, while the red variables denote output measurements.

freedom) for optimization. The measured outputs are the cathode inlet and outlet temperatures of the stack, the afterburner temperature, and the electrical potential of the stack. The electrical power, system efficiency, fuel utilization and air-excess ratio are computed based on the input and output variables.

6.1.2. Simplified model

As discussed above, MA in general (and CA in this example) does not require an accurate model since optimality is computed by enforcing the plant KKT conditions. The model is only there to guarantee that the correct set of constraints is active. To reflect this aspect, we say here that we use a *simplified model* (in practice, we use the best available model, which, however, does not need to be very accurate). In particular, we will use a lumped model as this will be sufficient to

capture the fundamental aspects of the SOFC system. For additional details, the reader is referred to [13].

The dynamic behavior of a SOFC stack is characterized by a fast response of the cell voltage to a step change in current density, followed by an equilibration process that is considerably slower. BlueGEN has several components, with each component being characterized by its own dynamics. Moreover, the system has a thermal recycle, which increases its dynamic complexity. Hence, the system is expected to have (i) fast dynamics associated with the electrochemical reactions, (ii) slow dynamics associated with the thermal inertia of the individual components, and (iii) still slower dynamics associated with the thermal recycle. The four available inputs are the current and the flowrates of fuel, air and cooling air, that is, $\mathbf{u} = [I,\ q_{CH_4},\ q_{air},\ q_{cool}]^{\mathrm{T}}$.

6.1.3. Optimization problem

Offline numerical optimization has confirmed that optimal operation corresponds to four (1 equality and 3 inequality) constraints being active. Hence, RTO can be implemented via CA, a simpler version of MA that only considers zeroth-order modifiers.

BlueGEN has several operational constraints. The constraints that most limit efficiency are an upper bound on fuel utilization and a lower bound on cell voltage. The static optimization problem, with all the constraints used in this study and the corresponding zeroth-order modifiers, can be written mathematically as follows:

$$
\begin{aligned}
\mathbf{u_{k+1}^{\star}} \in \arg\max_{\mathbf{u}} \quad & \eta(\mathbf{u}) = \frac{P_{el}}{q_{CH_4}\ LHV_{CH_4}} \\
\text{s.t.} \quad & P_{el}(\mathbf{u}) + \epsilon_k^{P_{el}} = P_{el}^S\ [W] \\
& U_{cell}(\mathbf{u}) + \epsilon_k^{U_{cell}} \geq 0.76\ [V] \\
& 650 \leq T_{cath,in}(\mathbf{u}) + \epsilon_k^{T_{cath,in}} \leq 750\ [^{\circ}C] \\
& 650 \leq T_{cath,out}(\mathbf{u}) + \epsilon_k^{T_{cath,out}} \leq 790\ [^{\circ}C] \\
& \nu(\mathbf{u}) \leq 0.8 \\
& \lambda_{air}(\mathbf{u}) \geq 3 \\
& 0 \leq I \leq 50\ [A] \\
& 1 \leq q_{CH_4} \leq 7\ [L.min^{-1}] \\
& 85 \leq q_{air} \leq 200\ [L.min^{-1}] \\
& 0 \leq q_{cool} \leq 40\ [L.min^{-1}],
\end{aligned}
\tag{33}
$$

where η is the efficiency of the SOFC system, P_{el} represents the power demand and P_{el}^S its setpoint, U_{cell} denotes the cell voltage, ν is the fuel utilization, λ_{air} represents the air excess ratio, LHV_{CH_4} is the low heat value of the fuel, I is the electrical current, q_{CH_4}, q_{air} and q_{cool} denote the flowrates of fuel, air and cooling air, $T_{cath,in}$ and $T_{cath,out}$ represent the cathode inlet and outlet temperatures, respectively.

Remark 4. *Optimization Problem* (33) *contains four zeroth-order modifiers. Note that the four inputs do not have modifiers since there are perfectly known. Furthermore, the fuel utilization* ν *and the air excess ratio* λ_{air} *do not have modifiers either because they are only functions of the inputs and thus do not carry any uncertainty.*

Remark 5. *Implementation of CA is straightforward: At the* k^{th} *iteration (be it at steady state or during transient, see Section 5.4), the four modifiers are computed from measurements according to Eq.* (9)*, NLP* (33) *is solved for* $\mathbf{u}_{k+1}^{\star}$*, and input filtering is applied according to Eq.* (13)*.*

6.1.4. Experimental results

This section describes the experimental results obtained upon applying fast CA to BlueGEN. The electrical power demand acts as an unknown disturbance. The dynamic model is the simplified

model described in Section 6.1.2. A diagonal filter matrix with $K_f = 0.4$ is used. The power setpoint P_{el}^S is changed every 2 h as follows:

$$P_{el}^S(t) = \begin{cases} 1.0 \text{ [kW]}, & \text{t} \leq 2 \text{ h} \\ 1.25 \text{ [kW]}, & 2 \text{ h} < \text{t} \leq 4 \text{ h} \\ 1.5 \text{ [kW]}, & \text{t} > 4 \text{ h}. \end{cases} \tag{34}$$

BlueGEN starts at steady state with an electrical power of 1 kW and about 55% DC efficiency. CA is repeated with the RTO period of 5 min until convergence. Figures 4 and 5 depict the performance of fast CA.

DC electrical efficiency increases to about 64.5% in the first 30 min. However, since the system is not at thermal equilibrium, efficiency decreases slightly with temperature to eventually reach 64% after 2 h. Hence, electrical efficiency is increased by about 16% with respect to its initial value for the same electrical power.

Next, the power setpoint is changed to 1.25 kW after 2 h. It takes the system about 15 min to reach 96% of the new power setpoint and 45 min to completely reach it. The cell potential and the temperatures keep increasing, with the electrical efficiency reaching 62% at this point.

Finally, after 4 h, the electrical power setpoint is changed to 1.5 kW. The system takes about 11 min to complete 95% of the change and 40 min to fully complete it. Electrical efficiency reaches about 61% after 2 h.

For all tested electrical power demands, and upon convergence, optimal operation is determined by active constraints (Figures 4 and 5). At steady state, for the power setpoint of 1 kW, the following four constraints are active: electrical power, cathode inlet temperature, air and cooling air flowrates. At the power of 1.25 kW, another set of four constraints are active, namely, electrical power, fuel utilization, air-excess ratio, and cooling air flowrate. Finally, at the power of 1.5 kW, the active constraints are electrical power, fuel utilization, cathode outlet temperature, and cooling air flowrate.

6.1.5. Why it works so well in practice

The commercial fuel-cell system is a complex dynamic system composed of several units. It involves chemical reactions, heat exchange and thermal recycle. Detailed modeling of the system would represent a very time-consuming task. However, operational optimization turns out

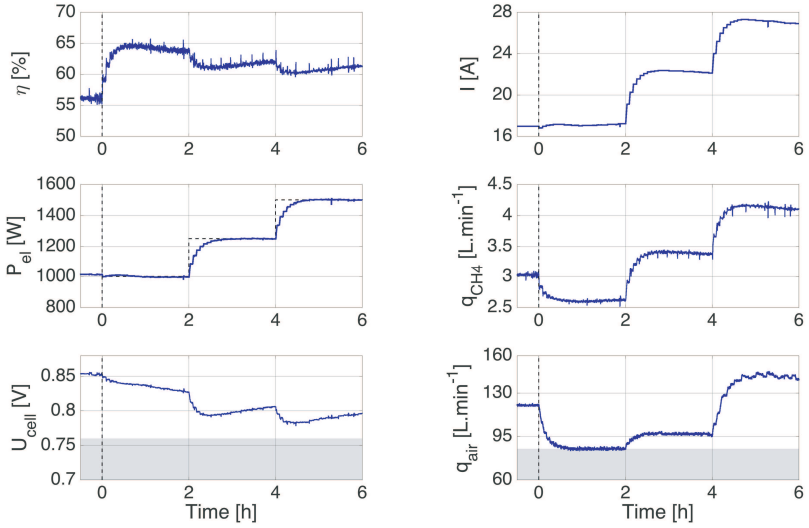


Figure 4: *Fast CA* applied to BlueGEN during power variation of 1-1.5 kW: Efficiency, current, electrical power, methane flowrate, cell voltage, and air flowrate vs. time.

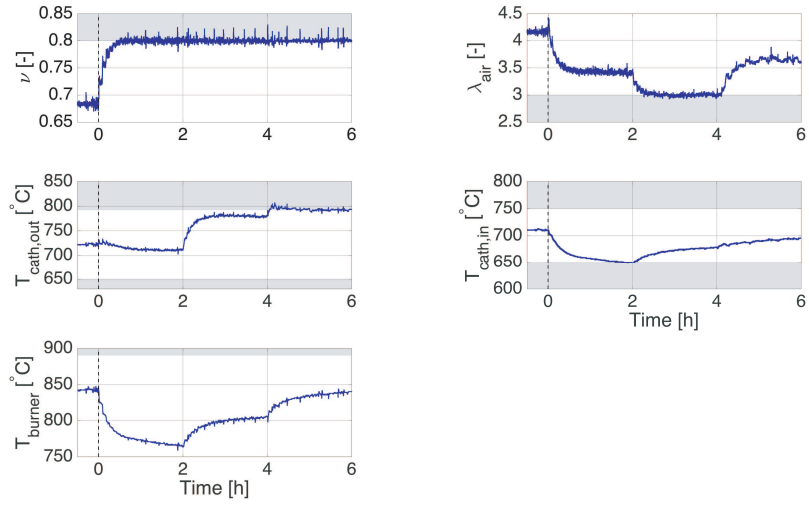


Figure 5: *Fast CA* applied to BlueGEN during power variation of 1-1.5 kW: Fuel utilization, air-excess ratio, cathode outlet and inlet temperatures, and afterburner temperature vs. time.

to be rather simple since optimal operation is determined by the activation of four operational constraints.

If the four active constraints were known a priori, it would be trivial to implement optimal operation via constraint tracking [3]. However, the active constraints are not known a priori, as the set of active constraints changes with the operating point (power demand). Nevertheless, knowledge of the fact that optimal operation is determined by active constraints let us work with CA instead of MA, that is, there is no need to introduce (and estimate) first-order modifiers.

What is the role of the model in this application? The model is not used to predict concentrations and temperatures accurately as these quantities are not needed for optimization. Optimization simply requires that the *modified model* predicts the active constraints well, in fact all the constraints well since the set of active constraints is not known. However, this is trivially achieved via the use of the zeroth-order (bias) correction terms. Hence, the only real requirement on the modified model is to rank correctly in terms of cost the various scenarios corresponding to different sets of four active constraints. This in fact corresponds to selecting the CVs of a multivariable feedback control scheme.

Note that the optimization results do not rely on the ability of the modified model to predict the plant behavior accurately. The KKT conditions only dependent on certain quantities to be at their constrained values, with the model predictions of these quantities being adjusted via measurements. Hence, optimization is based on measurements and feedback control (implemented here via CA), with the accuracy of the nominal model being secondary. And this is why it works so well in practice!

6.2. Kite for Energy Harnessing

This application deals with an experimental kite setup built by students to demonstrate both control and optimization [10]. This example is particularly interesting for two reasons: (i) The system is very uncertain, with significant plant-model mismatch and large random disturbances due to unknown and changing wind conditions, and (ii) since there are too many input parameters to allow gradient estimation in this noisy environment, a small number of privileged input directions is selected via sensitivity analysis, and so-called *directional MA* is implemented.

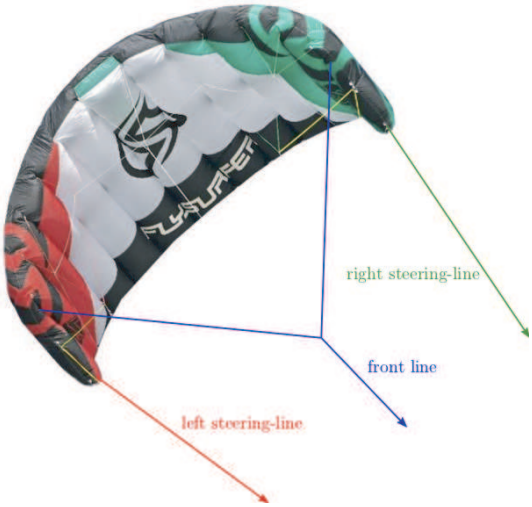


Figure 6: The 2.5-m^2 Flysurfer Viron power kite used in this study.

6.2.1. Kite power

Kite power is an emerging wind power technology with both great potential and many technical challenges. A dynamically flying power kite is a fast, unstable system influenced by unpredictable wind disturbances, with usually only noisy and incomplete measurements being available. The path the kite flies determines how much power is produced. Hence, in addition to keeping the kite from crashing, the autopilot must ensure the kite follows a path that is efficient for power production. A horizontal figure-of-eight pattern is generally considered to be the most efficient type of path for extracting energy from the wind. Although the controller has only a few tuning parameters, the effect of these parameters on the kite trajectory is difficult to determine a priori.

6.2.2. Experimental setup

The experimental setup used in this work is a small (2.5 m^2) kite on a short (35 m) fixed-length line, which is sensed and actuated from the ground by a mechanized station. The kite is shown in Figure 6. The front line takes about 90% of the force generated by the kite, the two lightly tensioned rear lines allow the kite to be maneuvered. There are two degrees of freedom to operate the kite: (i) Adjusting the length difference between the rear lines allows steering the kite left or right, and (ii) adjusting the length of both rear lines relative to the front line allows controlling the kite velocity by changing its angle-of-attack to the on-rushing air.

The angle of the rod, from which the line angle can be inferred, is measured by rotary encoders.

The rear (steering) lines are wound in opposite directions around a reel, which is turned by a responsive and powerful servomotor. Rotating the reel shortens one line, while lengthening the other, achieving a steering effect. A high-precision ultrasonic anemometer mounted on a 3-m pole measures the wind speed and direction just above ground. The control algorithm runs in real time on a laptop with a sampling period of 30 ms. Hence, accurate measurements of the kite position, the front-line tension and the wind speed and direction are available at the ground station, where the steering input is manipulated.

6.2.3. Simplified model

The cart model for the kite reads:

$$\dot{\vartheta} = \omega_{\mathrm{k}} \cos\gamma, \tag{35}$$

$$\dot{\varphi} = \frac{\omega_{\mathrm{k}}}{\sin\vartheta} \sin\gamma, \tag{36}$$

$$\dot{\gamma} = -\left(\omega_{\mathrm{k}} r g_{\mathrm{s}} \delta + \dot{\varphi} \cos\vartheta\right), \tag{37}$$

$$\omega_{\mathrm{k}} = \frac{w}{r} E \cos\vartheta, \tag{38}$$

where ϑ and φ are the spherical coordinates of the kite position, γ the kite orientation, r the tether length, w the wind speed, ω_{k} the normalized kite speed, E the lift-to-drag ratio, g_{s} the turning constant, and δ the steering deflection as the single manipulated input.

This compact model is simple enough for the values of the parameters to be easily identified experimentally. Although this model can be used for controller design, resulting in good control performance during crosswind flight, it is too simple for tether-force optimization.

6.2.4. Optimization problem

The objective is to fly the kite as efficiently as possible. The line tension, which is closely related to the kite speed, is used as a measure of efficiency. Hence, the aim is to find a path such that the kite travels as rapidly as possible. The only operational constraint is that the kite must fly above a certain minimal elevation.

The choice of the decision variables $\mathbf{u}$ is critical since too many degrees of freedom will increase the load for gradient estimation. Based on offline sensitivity analysis [10], the inputs are chosen as the parameters u_1 and u_2 that determine the height and the curvature of the reference path.

The kite system is characterized by a considerable amount of nearly random perturbations caused by wind variations. The simplest solution to dealing with noise is to reduce its effect via

averaging. The line tension is averaged over $N_{\text{avg}} = 7$ path cycles. This reduces the noise to a manageable (if still quite high) level. The price to pay is that the RTO algorithm only iterates every 7 path cycles, and thus proceeds more slowly. The static optimization problem can be written mathematically as follows:

$$\begin{aligned} \mathbf{u}_{k+1} \in \arg\max_{\mathbf{u}} \quad & \bar{T}(\mathbf{u}) + (\boldsymbol{\lambda}_k^{\bar{T}})^{\mathrm{T}}(\mathbf{u} - \mathbf{u}_k) \\ \text{s.t.} \quad & z_{min} \leq z(\mathbf{u}) \end{aligned} \tag{39}$$

$$(\boldsymbol{\lambda}_k^{\bar{T}})^{\mathrm{T}} := \nabla_u \bar{T}_{p,k} - \nabla_u \bar{T}_k, \tag{40}$$

where $\mathbf{u} = [u_1 \; u_2]^T$, with u_1 and u_2 the height and the curvature of the reference path, $\bar{T}$ is the average line tension and z the kite elevation.

Remark 6. *Optimization Problem (39) contains only two first-order modifiers. Since the constraint on minimal elevation is never active, it is not included in the MA scheme.*

6.2.5. Experimental results

RTO performance was tested over several days of experiments. Depending on the conditions, the resulting optimal reference path was quite different. For example, in light winds, the path tended to be much wider than in stronger winds.

Figure 7 shows the line tension and the kite elevation during 30 min of autonomous flight. During the first 7 min (13 iterations) and the last 5 min (10 iterations), the kite follows a constant reference path that is rather high and narrow, resulting in a low average line tension of about 80 kg. The RTO algorithm markedly improved the average line tension, increasing it to about 135 kg, during the intermediate 22 min that are depicted with a grey-shaded area on Figure 7.

6.2.6. Why it works so well in practice

The real-time optimization of kites for harnessing wind energy is a very challenging problem due to the large amount of uncertainty associated with wind variations. Model-based approaches would need, in addition to an accurate model of the kite behavior, a good description of the wind conditions, the latter requiring appropriate on-board sensors such as accelerometers and a reliable GPS.

In contrast, this study was designed to rely on measurements of the plant cost and constraints involved in the KKT conditions rather than on the kite model and wind measurements. It turns out

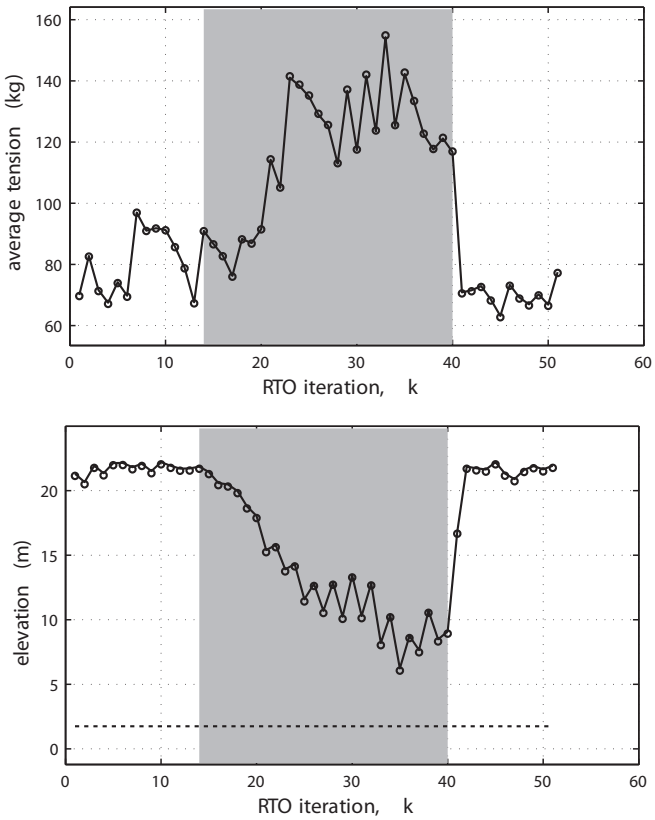


Figure 7: Performance of the MA algorithm with $N_{\text{avg}} = 7$. Each circle is the average value for the tension/elevation during N_{avg} path cycles. The dotted line indicates the minimal height constraint. The RTO algorithm was activated during the shaded iterations. The total experiment lasted 29 min, and the RTO algorithm was active for 17 min.

that the only operational constraint (dealing with minimal elevation) is never active because of wind shear, that is, the wind speed drops close to the ground. Hence, this constraint can be discarded for the purpose of optimization. With no active constraint, optimal operation is determined by forcing the cost gradient to zero. Hence, the challenge is to reliably estimate this cost gradient with respect to the inputs, in this case the parameters used to describe the path to follow. Since it is not possible to estimate reliably many plant derivatives in this noisy environment, a sensitivity analysis based on the nominal model was used offline to determine the important (privileged) input directions. It turned out that only two parameters were found important, one associated with the size of the figure of eight, and the other one with its elevation. Hence, derivatives in only two directions need to be estimated experimentally.

The role of the model is very minimal in this application. There is no need for the model to predict the kite trajectory accurately as the trajectory is not used to compute the optimal inputs. Optimization simply requires that the *modified model* be able to predict the cost gradient with respect to the two shape parameters, which is achieved trivially with the use of first-order modifiers. Moreover, model adequacy requires that the model cost function exhibits qualitatively the same curvature as the *unknown* cost function of the kite, which corresponds to pushing the system in the right (and not the opposite) direction!

In this application, optimization amounts to estimating the gradient in two directions and pushing it to zero using feedback control. The feedback control part is implemented via MA, for which the nominal model need not be accurate. And this is why it works so well in practice!

7. Conclusions

This paper has discussed the use of MA as a practical tool for real-time optimization. In appearance, MA uses (i) the knowledge (measurement or estimation) of the plant KKT elements to enforce optimal performance, and (ii) a plant model and numerical optimization to compute the optimal inputs. Hence, MA has been perceived as a model-based optimization technique and, as such, was thought to be highly dependent on model accuracy for good performance.

This paper has clearly shown that, in fact, MA is a feedback-based RTO scheme, the model only helping implement multivariable feedback. As a consequence, MA performance depends more on the quality of the measured/estimated KKT elements than on model accuracy. Compared to

traditional model-based RTO schemes, the model requirement has shifted from being quantitatively correct (model accuracy) to being qualitatively relevant (model adequacy). This does not say that MA comes for free: the requirement on model accuracy is replaced by a requirement on measurement/estimation accuracy, as it is difficult to estimate steady-state gradients for a dynamic plant in a noisy environment.

The lesser dependence on model accuracy comes as a boost to practical implementation. Practitioners have often been puzzled by how well MA performs with such a "simple and by no means accurate" model! Indeed, there is a lot to be said regarding the choice of the model in a given application. However, modeling is not a religion, and there is no such thing as the "true model" [2]. Whether a given model is useful or not depends on the application at hand. Simulation, control, and optimization typically require models with specific (that is different) attributes [4]. Unfortunately, the obvious concept of "modeling for a goal" does not seem to have percolated fully into the research community. As a result, the modeler often spends most of his/her time trying to make the model fit the available data (mostly process outputs) instead of modeling what matters most in a given application. In the context of RTO, "modeling for optimization" implies paying close attention to the optimality conditions, which is exactly what MA does when it uses modifiers to locally match the plant KKT!

Acknowledgment

This work is the results of many years of research and rich discussions with many people, in particular former doctoral and post-doctoral collaborators. The authors would like to thank them all for their contribution to this topic.

References

[1] Bazaraa, M. S., Sherali, H. D., Shetty, C. M., 2006. Nonlinear Programming: Theory and Algorithms, 3rd Edition. John Wiley and Sons, New Jersey.

[2] Bonvin, D., Georgakis, C., Pantelides, C. C., Barolo, M., Grover, M. A., Rodrigues, D., Schneider, R., Dochain, D., 2016. Linking models and experiments. Ind. Eng. Chem. Res. 55 (25), 6891–6903.

[3] Bonvin, D., Srinivasan, B., 2003. Optimal operation of batch processes via the tracking of active constraints. ISA Trans. 42 (1), 123–134.

[4] Bonvin, D., Srinivasan, B., 2013. On the role of the necessary conditions of optimality in structuring dynamic real-time optimization schemes. Comp. Chem. Eng. 51 (0), 172–180.

[5] Brdyś, M., Tatjewski, P., 2005. Iterative Algorithms for Multilayer Optimizing Control. Imperial College Press, London UK.

[6] Câmara, M. M., Quelhas, A. D., Pinto, J. C., 2016. Performance evaluation of real industrial RTO systems. Processes 4 (4), 54.

[7] Chachuat, B., Marchetti, A. G., Bonvin, D., 2008. Process optimization via constraints adaptation. J. Process Contr. 18 (3), 244–257.

[8] Chachuat, B., Srinivasan, B., Bonvin, D., 2009. Adaptation strategies for real-time optimization. Comp. Chem. Eng. 33, 1557–1567.

[9] Chen, C. Y., Joseph, B., 1987. On-line optimization using a two-phase approach: An application study. Ind. Eng. Chem. Res. 26, 1924–1930.

[10] Costello, S., 2015. Real-time Optimization via Directional Modifier Adaptation, with Application to Kite Control. Doctoral thesis No. 6571, EPFL Lausanne, Switzerland.

[11] Costello, S., François, G., Bonvin, D., 2016. A directional modifier-adaptation algorithm for real-time optimization. J. Process Contr. 39, 64–76.

[12] de Avila Ferreira, T., 2019. Real-time Optimization of Solid Oxide Fuel Cell Systems – Methodology and Implementation. Doctoral thesis No. 9276, EPFL Lausanne, Switzerland.

[13] de Avila Ferreira, T., Wuillemin, Z., Faulwasser, T., Salzmann, C., Van herle, J., Bonvin, D., 2019. Enforcing optimal operation in solid-oxide fuel-cell systems. Energy 181, 281–293.

[14] de Avila Ferreira, T., Wuillemin, Z., Marchetti, A. G., Salzmann, C., Van herle, J., Bonvin, D., 2019. Real-time optimization of an experimental solid-oxide fuel-cell system system. J. Power Sources 429, 168–179.

[15] Faulwasser, T., Bonvin, D., 2014. On the use of second-order modifiers for real-time optimization. In: 19th IFAC World Congress. Cape Town.

[16] Forbes, J. F., Marlin, T. E., 1994. Model accuracy for economic optimizing controllers: The bias update case. Ind. Eng. Chem. Res. 33, 1919–1929.

[17] Forbes, J. F., Marlin, T. E., 1996. Design cost: A systematic approach to technology selection for model-based real-time optimization systems. Comp. Chem. Eng. 20, 717–734.

[18] François, G., Bonvin, D., 2014. Use of transient measurements for the optimization of steady-state performance via modifier adaptation. Ind. Eng. Chem. Res. 53 (13), 5148–5159.

[19] Gao, W., Engell, S., 2005. Iterative set-point optimization of batch chromatography. Comp. Chem. Eng. 29, 1401–1409.

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

[21] Marchetti, A. G., Chachuat, B., Bonvin, D., 2010. A dual modifier-adaptation approach for real-time optimization. J. Process Contr. 20, 1027–1037.

[22] Marchetti, A. G., François, G., Faulwasser, T., Bonvin, D., 2016. Modifier adaptation for real-time optimization – Methods and applications. Processes 4 (4), 55.

[23] Marlin, T. E., Hrymak, A. N., 1997. Real-time operations optimization of continuous processes. In: AIChE Symposium Series - CPC-V. Vol. 93. pp. 156–164.

[24] Navia, D., Gutiérrez, G., de Prada, C., 2013. Nested modifier-adaptation for RTO in the Otto-Williams reactor. In: IFAC Symp. Dycops. Mumbai, India, pp. 123–128.

[25] Roberts, P. D., 1979. An algorithm for steady-state system optimization and parameter estimation. J. System Science 10, 719–734.

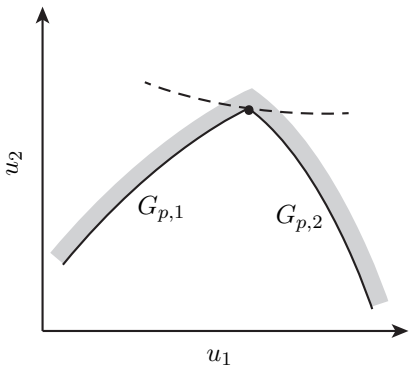
[26] Shin, S. B., Han, S. P., Lee, W. J., Im, Y. H., Chae, J. H., Lee, D. I., Lee, W. H., Urban, Z., 2007. Optimize terephthaldehyde reactor operations. Hydrocarbon Processing, April 2007, 83–90.

[27] Singhal, M., Marchetti, A. G., Faulwasser, T., Bonvin, D., 2018. Active directional modifier adaptation for real-time optimization. Comput. Chem. Engng. 115 (6), 246–261.

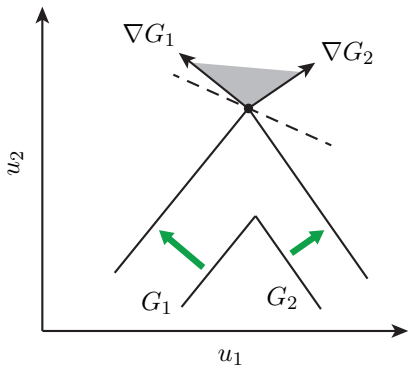
[28] Srinivasan, B., Bonvin, D., 2019. A feature-based analysis of static real-time optimization schemes. Ind. Eng. Chem. Res. 58, 14227–14238.

[29] Tatjewski, P., 2002. Iterative optimizing set-point control - The basic principle redesigned. In: 15th IFAC World Congress. Barcelona, Spain.

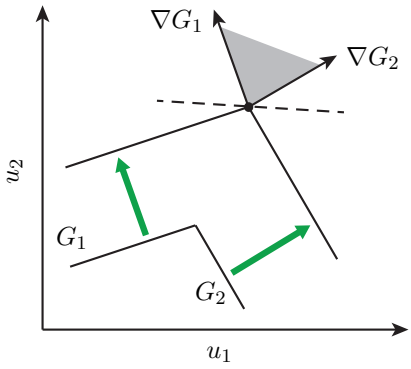
[30] Yip, W. S., Marlin, T. E., 2004. The effect of model fidelity on real-time optimization performance. Comp. Chem. Eng. 28, 267–280.



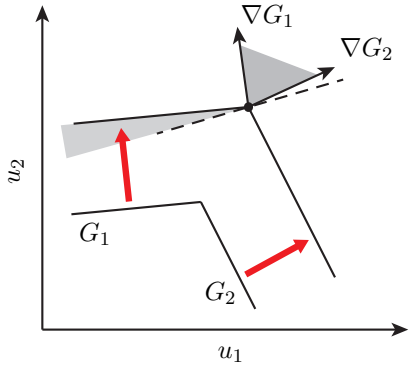
(a) Plant



(b) Adequate Model



(c) Adequate Model



(d) Inadequate Model