Output Modifier Adaptation with Filter-Based Constraints

Output Modifier Adaptation with Filter-Based Constraints

A. Papasavvas, G. Francois

School of Engineering, Institute for Material and Processes, The University of Edinburgh, Edinburgh EH93FB

Abstract

Modifier adaptation (MA) and output modifier adaptation (MAy) are iterative model-based real-time optimization (RTO) algorithms that have the proven ability to drive plants to their optimal operating condition upon convergence despite disturbances and modeling uncertainty, provided the model at hand satisfies model adequacy conditions. But there is no guarantee that constraints are satisfied before convergence. In this article, an improvement of the formulation of MA and MAy is proposed that is proven to bring significant improvements w.r.t. these two limitations – model adequacy and feasibility of iterates. While standard MA or MAy suggests to perform optimization and filtering sequentially, it is proposed to integrate the input filtering stage in the modified model-based optimization problem by means of additional filter-based constraints. The corresponding approach, labeled "KMAy", is (i) proven to preserve constraint qualification despite additional constraints, (ii) proven to preserve the property of MA methods to converge to the true plant optimal inputs, (iii) proven to significantly relax the model adequacy condition - leading it to be independent of the constraints of the optimization problem, (iv) shown to increase the chances of converging from the safe side of the plant constraints and (v) shown to support the choice of input filtering, instead of output or modifier filtering, *if the input filter is appropriately chosen*. A method for the automatic selection of the largest filter gain with the five aforementioned assets, while minimizing the filter-induced conservatism, is also proposed. The performances of KMAy with and without adaptive gain are successfully illustrated by means of the optimization of a benchmark simulated chemical reactor.

Keywords: Real-time optimization, modifier adaptation, model adequacy.

1. Introduction

Industrial processes are operated via the manipulation of input variables. Some of the plant inputs are fixed, while some others can be freely chosen. This choice can be made by the engineers or operators of the plant, based on engineering insight, but also using optimization techniques instead. With model-based optimization techniques, the model is optimized to determine the values of these degrees of freedom, which maximize the performances of the process at hand, while enforcing, from the viewpoint of the model, the satisfaction of the operating and production constraints. When the model is not perfect, real-time optimization (RTO) algorithms are appropriate as they incorporate plant measurements in the optimization framework, to compensate from the inability of the process model to accurately predict the plant optimal inputs.

RTO methods can be classified w.r.t. the type of information they use. With evolutionary techniques for RTO, among which steepest-descent, heuristic search (Nelder-Mead method [1]), or evolutionary optimization [2], *only* past and current plant measurements are used for determining the next set of inputs for the plant. On the other hand, model-based RTO algorithms, e.g. the two-step approach (TS - [3]), suggest to make explicit use of both the measurements and the available model in an integrated manner. With TS, at each iteration (i) plant measurements are used to refine the parameters of the model, (ii) the updated model is optimized to define the next inputs for the plant and (iii) the procedure is repeated until convergence. Unfortunately, TS does not guarantee plant optimality upon convergence in case of structural plant-model mismatch, i.e., when modeling errors are not parametric only. On the other hand, RTO via modifier adaptation (MA) [4, 5, 6] has the mathematically proven ability to reach the plant optimum upon convergence despite structural plant-model mismatch [7, 5]. With standard MA, the model is kept unchanged and measurements are used to construct and add affine corrections to the cost and constraint functions. To ensure that convergence of MA can only be at the true plant optimal inputs despite structural and parametric plant-model mismatch, *model adequacy conditions* must be satisfied, i.e., the reduced Hessian of the Lagrangian must be positive definite at the plant optimum. Similar conditions also exist for TS, which are ways more restrictive and harder to meet in practice [8, 5]. A variant of MA where the cost and constraint functions are indirectly modified through the correction of the modeled outputs is introduced in [5] and analyzed in

[9]. This method, namely "output modifier adaptation" (MAy), also guarantees optimality upon convergence, but leads to "deeper" model corrections enabling faster convergence and reducing the risk of model inadequacy [9]. However, none of these methods is free of weaknesses and research comes up regularly with extensions mainly dealing with the following four main challenges:

(a) Enforcing model adequacy. With MA and MAy, the model adequacy condition [8] reduces to the positive definiteness of the reduced Hessian of the Lagrangian of the modified model-based optimization problem at the plant optimum [5]. Model adequacy can almost never be checked a priori in practice (apart e.g. from convex problems), since the plant optimal inputs are unknown, but it can be easily enforced by using convex approximations of the model [10, 11], as it forces the problem to be convex everywhere. However, it goes with a decrease of the model quality, which can be detrimental to the prediction of plant constraints and to the convergence rate. Another idea is to use second-order modifiers [12]. But this relies on the availability of accurate estimates of the plant Hessians, something that is very unlikely since plant gradients estimation is already challenging (see (c)). Finally, it has been recently shown that model adequacy for MA could be enforced with an appropriate update of model parameters [13].

(b) Avoiding constraints violations before convergence. MA and MAy ensure plant feasibility upon convergence but do not provide any guarantee for the iterates. So far, apart from using additional plant information, which is most often not available, such as Lipschitz constants or quadratic upper bounds [14, 15, 16], there is no way to ensure "absolute" plant feasibility at each iteration in the general case. One approach could be to combine trust region [17] and MA [18, 19, 20], reducing thus the distance between two consecutive iterates, therefore maintaining the next inputs in a region whereby the modified model is the most reliable. Methods enabling the estimation of the plant steady state during the transient operation have also been proposed [21, 22, 23, 24], with emphasis on the convergence rate to the plant optimal operating conditions. This is indeed helpful for plant feasibility since the prediction of the steady state of the plant can be frequently updated and potentially improved during the transient, hopefully leading to better predictions of potential violations of the constraints and better decisions to be taken during the transient. Constraint violations should thus last shorter than with standard MA, where no input update is performed before steady state has been reached.

(c) Dealing with inaccurate measurements. MA and MAy require reliable estimates of plant gradients, which is challenging in the presence of measurement noise. Several approaches have been proposed to mitigate this issue. For instance, it has been suggested to combine MA with quadratic approximation methods [11], or to use plant gradients as the degrees of freedom of an upper optimization layer, solved at a lower frequency than another, inner, MA layer where plant gradients are kept constant [25, 26]. Recently, Gaussian process regression techniques or neural networks have also been investigated [27, 28], to limit the impact of plant-model mismatch when the model is used to predict plant gradients. Alternatively, [29] and [30] suggest to integrate measurements (or estimates) of disturbances in the RTO framework.

(d) Scalability to large-scale problems. One issue with large-scale systems is that the number of plant gradients increases linearly with the number of inputs, to the point that it becomes intractable for MA or MAy. With directional modifier adaptation (DMA-[31]), it is suggested to focus a small number of selected inputs, the so-called "privileged directions" to avoid having to estimate too many plant gradients. More recently, DMA was further extended to include a real-time update of the privileged directions and to provide optimality upon convergence [32]. Large-scale plants can indeed be viewed as a (large) network of interconnected subplants, each of which modeled with a similar (large) network of interconnected submodels. A second issue is thus that each submodel being an approximation of the corresponding subplant, their interconnection typically leads to uncertainty propagation when the whole network is simulated, which can, in turn, lead to poorer predictions of the behavior of the plant and thus to even weaker decisions. Several recent extensions to MA and MAy mitigate this second issue by using more plant measurements than necessary [33, 34, 35, 36].

Most, if not all, MA and MAy variants use a filter to provide asymptotic stability and avoid too large solicitations of the plant [5]. However, it is unclear whether it is better to filter the modifier terms, the outputs or the inputs. Also, the choice of the filter gain remains an open question. The authors have recently proposed to implement filtering in the model-based optimization problem by means of filter-based additional constraints, rather than performing optimization and filtering sequentially (KMAy-[37]). It has been shown that it relaxes the model adequacy criterion [37] and makes the iterates safer [38]. This article motivates, introduces, develops further and extends KMAy by providing: (i) an automatic method to select the filter

gain, (ii) a method to enforce model adequacy that is less detrimental to the accuracy of the model than its replacement by a convex approximation of the cost and constraints [10], and (iii) arguments in favor of input filtering

This article is organized as follows. After a statement of the problem and a review of MAy, KMAy is introduced by means of the extension of two recent conference articles [37, 38], [37] being the conference article that motivated the invitation to contribute to this special issue that is reviewed and extended in Section 3. Then, in Section 4, KMAy is developed further, with the introduction of a new adaptive filter gain selection method (labeled "ad-KMAy"), which ensures that the largest applicable gain is implemented that guarantees model adequacy. A simulated case study illustrating all contributions and algorithms is given in Section 5 and Section 6 concludes the article.

2. Real-Time Optimization via Modifier Adaptation

2.1. Optimization Problem

The optimal operating conditions of a plant $\boldsymbol{u}_p^\star \in \mathbb{R}^{n_u}$ are the input variables $\boldsymbol{u} \in \mathbb{R}^{n_u}$ minimizing the plant operating cost $\Phi_p \in \mathbb{R}$, while satisfying all plant constraints $\boldsymbol{G}_p \in \mathbb{R}^{n_g}$, with the subscript $(.)_p$ used here to indicate quantities that are related to the plant. In mathematical form, $\boldsymbol{u}_p^\star \in \mathbb{R}^{n_u}$ is the solution of the following nonlinear program (NLP):

$$\begin{aligned} \boldsymbol{u}_p^\star := \arg\min_{\boldsymbol{u}} \quad & \Phi_p(\boldsymbol{u}) := \phi(\boldsymbol{u}, \boldsymbol{y}_p) \\ \text{s.t.} \quad & \boldsymbol{G}_p(\boldsymbol{u}) := \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{y}_p) \leq \boldsymbol{0}, \\ & \boldsymbol{y}_p = \boldsymbol{F}_p(\boldsymbol{u}), \end{aligned} \tag{2.1}$$

where $\boldsymbol{y}_p \in \mathbb{R}^{n_y}$ are the measured outputs of the plant, and $\boldsymbol{F}_p : \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$ is the plant input-output mapping. The constraints are further distinguished in the following definition, as it is necessary for further analyses in this article.

Definition 1. *Let $\mathcal{U} \subseteq \mathbb{R}^{n_u}$ denote the set of inputs satisfying the constraints that are* not *subject to uncertainty, and $\mathcal{F}_p \subseteq \mathcal{U}$ denote the subset of $\mathcal{U}$ in which all plant constraints are satisfied. $\mathcal{U}$ generally contains only pure input constraints, typically corresponding to lower ($\boldsymbol{u}^{\mathrm{L}}$) and upper input bounds ($\boldsymbol{u}^{\mathrm{U}}$), i.e., $\mathcal{U} := \{\boldsymbol{u} \in \mathbb{R}^{n_u} \mid \boldsymbol{u}^{\mathrm{L}} \leq \boldsymbol{u} \leq \boldsymbol{u}^{\mathrm{U}}\}$.*

In addition to these definitions, it is assumed that the plant satisfies the five following properties:

Assumption 1 (Plant properties). *Problem* (2.1) *is such that:*

- $\forall \boldsymbol{u} \in \mathcal{F}_p$, *there are no steady-state output multiplicities, i.e., the mapping* $\boldsymbol{F}_p$ *is such that for one input, only one output is possible,*
- Φ_p *and* $G_{i,p}$, $i = 1, ..., n_g$, *are twice continuously differentiable* ($\mathcal{C}^2$) *w.r.t.* $\boldsymbol{u}$ *on* $\mathcal{F}_p$,
- $\mathcal{F}_p$ *is a non-empty compact set,*
- $\forall \boldsymbol{u} \in \mathcal{F}_p$, *the linear independence constraint qualification (LICQ) holds,*
- ϕ *and* g_i, $i = 1, ..., n_g$, *are known functions of manipulated and measured variables (*$\boldsymbol{u}$ *and* $\boldsymbol{y}_p$*)*[1].

In practice the plant input-output mapping $\boldsymbol{F}_p$ is not known, but approximated with a model $\boldsymbol{F}$. Thus, the solution to Problem (2.1) can be approached by solving the following NLP:

$$\begin{aligned} \boldsymbol{u}^\star := \arg\min_{\boldsymbol{u}} \quad & \Phi(\boldsymbol{u}) := \phi(\boldsymbol{u}, \boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{G}(\boldsymbol{u}) := \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{y}) \leq \boldsymbol{0}, \\ & \boldsymbol{y} = \boldsymbol{F}(\boldsymbol{u}). \end{aligned} \tag{2.2}$$

The difference between $\boldsymbol{F}_p$ and $\boldsymbol{F}$, referred to as plant-model mismatch, generally implies that $\boldsymbol{u}^\star$ differs from $\boldsymbol{u}_p^\star$, motivating the need for RTO methods with guaranteed optimality upon convergence. Before moving to RTO, the following assumptions are made for the model:

Assumption 2 (Model properties). *The model is such that:*

- $\forall \boldsymbol{u} \in \mathcal{U}$, *there are no steady-state output multiplicities, i.e., the mapping* $\boldsymbol{F}$ *is such that for one input, only one output is possible,*
- Φ *and* G_i, $i = 1, ..., n_g$, *are* $\mathcal{C}^2$ *w.r.t.* $\boldsymbol{u}$ *in* $\mathcal{U}$.

[1]Note that this assumption does not imply that there is no plant-model mismatch, but only that the knowledge of $\boldsymbol{u}$ and $\boldsymbol{y}_p$ is sufficient to compute accurately the cost and constraints of the plant. Plant-model mismatch is present since the model does not perfectly predict $\boldsymbol{y}_p(\boldsymbol{u})$.

2.2. Output Modifier Adaptation (MAy)

With standard MA, affine-in-input terms are added to the cost and constraints that are updated at each iteration using plant measurements. These terms correct the prediction of the values and gradients of the cost and constraints by the model and, upon convergence, reconcile the conditions of optimality of the model and the plant. With MAy, a similar correction is performed but at the level of the input-output mapping $\boldsymbol{F}$. The way the model predicts the plant outputs is therefore corrected, which leads to an *indirect*, yet simultaneous, affine-in-input correction of the cost and constraints functions of Problem (2.2). Although the correction of the prediction of the conditions of optimality is not direct, it has been shown in [9] that MAy provides (i) the same theoretical guarantees as MA, (ii) while being less sensitive to the values of the model parameters, and that it relaxes the model adequacy condition compared to MA.

Before going further, the following assumption is performed, for later methodological analyses:

Assumption 3. *The values and gradients of the plant outputs are perfectly known at each RTO iteration.*

Remark 1. *Of course, Assumption 3 is hard to meet in an industrial context where measurements are typically corrupted by noise and disturbances, especially w.r.t. plant gradients. Nevertheless, this limitation can be mitigated with gradient estimation methods [21, 25, 11, 23, 28]. Note that this assumption is performed for methodological analysis, just like with other MA variants.*

MAy with input filtering can be summarized as follows [5]:

At the k^{th} iteration, $\boldsymbol{u}_k$ is applied to the plant until steady state is reached, and the modified input-output mapping $F_{i,m,k}$, $\forall i = 1, \ldots, n_y$, is corrected as follows:

$$y_{i,m,k} := F_{i,m,k}(\boldsymbol{u}) := F_i(\boldsymbol{u}) + \varepsilon_k^{y_i} + (\boldsymbol{\lambda}_k^{y_i})^\mathsf{T}(\boldsymbol{u} - \boldsymbol{u}_k), \tag{2.3}$$

where $\varepsilon_k^{y_i} \in \mathbb{R}$ and $\boldsymbol{\lambda}_k^{y_i} \in \mathbb{R}^{n_u}$ are the zeroth and first-order modifiers of the

output i. These modifiers are defined as follows:

$$\boldsymbol{\varepsilon}_k^{y_i} := F_{i,p}(\boldsymbol{u}_k) - F_i(\boldsymbol{u}_k), \tag{2.4}$$

$$\boldsymbol{\lambda}_k^{y_i} := \nabla_{\boldsymbol{u}} F_{i,p}|_{\boldsymbol{u}_k} - \nabla_{\boldsymbol{u}} F_i|_{\boldsymbol{u}_k}\,, \tag{2.5}$$

where $(\cdot)|_{\boldsymbol{u}_k}$ stands for "evaluated at $\boldsymbol{u}_k$". Here, estimates of the plant output gradients $\nabla_{\boldsymbol{u}} \boldsymbol{F}_p|_{\boldsymbol{u}_k}$ are used, while standard MA uses the gradients of the plant cost and constraints. With (2.3), the modified cost and constraint functions at the k^{th} iteration read:

$$\Phi_{\text{MAy},k}(\boldsymbol{u}) := \phi(\boldsymbol{u}, \boldsymbol{y}_{m,k}),$$
$$\boldsymbol{G}_{\text{MAy},k}(\boldsymbol{u}) := \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{y}_{m,k}).$$

The following modified model-based optimization problem is then solved to determine the modified model-based optimal inputs:

$$\begin{aligned} \boldsymbol{u}_{k+1}^{\star} := \arg\min_{\boldsymbol{u}} \quad & \Phi_{\text{MAy},k}(\boldsymbol{u}) \\ \text{s.t.} \quad & \boldsymbol{G}_{\text{MAy},k}(\boldsymbol{u}) \leq \boldsymbol{0}, \\ & \boldsymbol{y}_{m,k} = \boldsymbol{F}_{m,k}(\boldsymbol{u}). \end{aligned} \tag{2.6}$$

The next operating point $\boldsymbol{u}_{k+1}$ is determined by applying a first-order filter:

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k + \boldsymbol{K}(\boldsymbol{u}_{k+1}^{\star} - \boldsymbol{u}_k), \tag{2.7}$$

where $\boldsymbol{K} \in \mathbb{R}^{n_u \times n_u}$ is a gain matrix, typically diagonal, with diagonal elements $K_i \in (0, 1]$, $\forall i \in \{1, \ldots, n_u\}$. In a nutshell:

Output Modifier Adaptation (MAy)

Initialization. Provide $\boldsymbol{u}_0$. Choose $\boldsymbol{K} \in \mathbb{R}^{n_u \times n_u}$ as a diagonal matrix with diagonal elements $K_i \in (0, 1]$, $\forall i \in \{1, \ldots, n_u\}$.
for $k = 0 \to \infty$

1. Apply the inputs $\boldsymbol{u}_k$ to the plant and wait for steady state.

2. Measure the plant outputs $\boldsymbol{y}_p(\boldsymbol{u}_k)$, and estimate the plant output gradients $\nabla_{\boldsymbol{u}} \boldsymbol{y}_p$ at $\boldsymbol{u}_k$. These estimates can require data from perturbed operating points in the neighborhood of $\boldsymbol{u}_k$.

3. Evaluate the modifiers (2.4)-(2.5).

4. Compute $\boldsymbol{u}_{k+1}$ by solving Problem (2.6).

end

Remark 2 (KKT matching). *The KKT conditions of MAy upon convergence match those of the plant [9], since Assumption 3 together with* (2.3) *guarantees that:*

$$\boldsymbol{F}_{m,k}(\boldsymbol{u}_k) = \boldsymbol{F}_p(\boldsymbol{u}_k), \quad \nabla_{\boldsymbol{u}} \boldsymbol{F}_{m,k}|_{\boldsymbol{u}_k} = \nabla_{\boldsymbol{u}} \boldsymbol{F}_p|_{\boldsymbol{u}_k}.$$

Then, it can be easily shown (using Assumption 1) that the affine correction of the model outputs induces an affine correction (as well as higher-order corrections) of the model cost and constraints functions:

$$X_{\mathrm{MAy},k}(\boldsymbol{u}_k) = X_p(\boldsymbol{u}_k), \;\; \nabla_{\boldsymbol{u}} X_{\mathrm{MAy},k}|_{\boldsymbol{u}_k} = \nabla_{\boldsymbol{u}} X_p|_{\boldsymbol{u}_k}, \tag{2.8}$$

for $X := \{\Phi, G_1, \ldots, G_{n_g}\}$, which can be used to easily show that MAy, like MA, guarantees plant optimality upon convergence.

Remark 3. *Filtering is motivated by two main reasons [5]:*

1- Asymptotic stability,

2- Smooth plant manipulations.

The filter can be applied on the inputs as in (2.7)*, or alternatively on the modifiers, i.e. with* (2.4) *and* (2.5) *replaced by:*

$$\begin{aligned} \boldsymbol{\varepsilon}_k^{y_i} &:= \boldsymbol{\varepsilon}_{k-1}^{y_i} + K^{\varepsilon^{y_i}}(F_{i,p}(\boldsymbol{u}_k) - F_i(\boldsymbol{u}_k) - \boldsymbol{\varepsilon}_{k-1}^{y_i}), \\ \boldsymbol{\lambda}_k^{y_i} &:= \boldsymbol{\lambda}_{k-1}^{y_i} + \boldsymbol{K}^{\boldsymbol{\lambda}^{y_i}}(\nabla_{\boldsymbol{u}} F_{i,p}|_{\boldsymbol{u}_k} - \nabla_{\boldsymbol{u}} F_i|_{\boldsymbol{u}_k} - \boldsymbol{\lambda}_{k-1}^{y_i}), \end{aligned} \tag{2.9}$$

where $K^{\varepsilon^{y_i}} \in (0,1]$, and $\boldsymbol{K}^{\boldsymbol{\lambda}^{y_i}} \in \mathbb{R}^{n_u \times n_u}$ are matrices with eigenvalues in $(0,1]$, with $i = 1, \ldots, n_y$.

Next, it is shown that filtering the inputs provides additional practical advantages (if judiciously used) than filtering the modifiers.

3. Output Modifier Adaptation with Additional Filter-Based Constraints (KMAy)

3.1. Integrating filter-based constraints into the model-based optimization problem

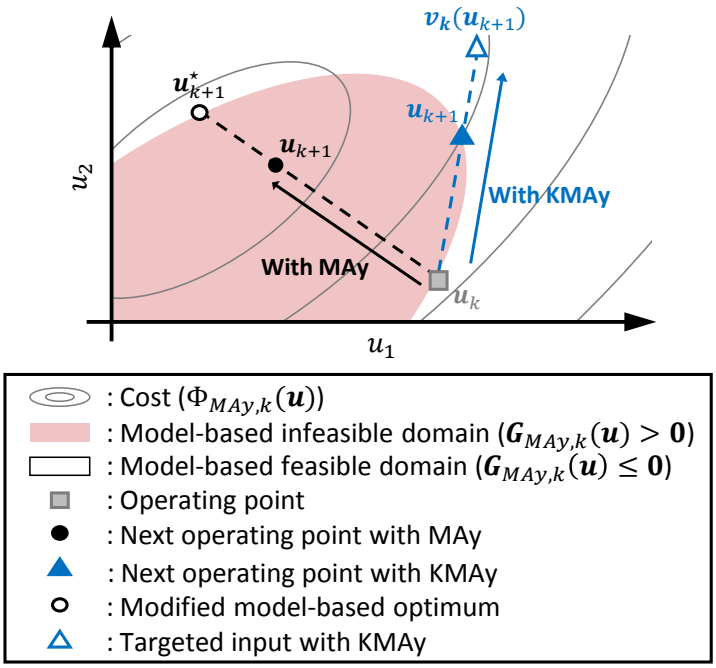


Figure 1: Illustration of the modified model-based feasibility at both the targeted inputs $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ and next applied (to the plant) inputs $\boldsymbol{u}_{k+1}$, with MAy and KMAy.

The section reviews, details and extends the method introduced in [37]. The main idea is to enforce modified model-based feasibility at the applied inputs $\boldsymbol{u}_{k+1}$, i.e. $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}_{k+1}) \leq \boldsymbol{0}$, something that cannot be guaranteed as such, with standard MA. Because, MAy only guarantees $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}^\star_{k+1}) \leq \boldsymbol{0}$, it is suggested to enforce the modified model-based constraints $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})$ to be satisfied at both the next operating point $\boldsymbol{u}_{k+1}$ and at the targeted operating point denoted $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ and defined as:

$$\boldsymbol{v}_k(\boldsymbol{u}) := \boldsymbol{K}^{-1}(\boldsymbol{u} - \boldsymbol{u}_k) + \boldsymbol{u}_k, \tag{3.1}$$

where the function $\boldsymbol{v}_k(\boldsymbol{u})$ is the inverse of the filtering function (2.7), and the targeted inputs are the inputs from which the next operating point is backtracked, when input filtering is implemented. For example, in the case of MAy, $\boldsymbol{u}^\star_{k+1}$ is both the modified mode-based optimum and the targeted point, i.e. $\boldsymbol{u}^\star_{k+1} = \boldsymbol{v}_k(\boldsymbol{u}_{k+1})$. However, by enforcing $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}_{k+1}) \leq \boldsymbol{0}$, the choices of $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ and of the next operating condition $\boldsymbol{u}_{k+1}$ become interdependent, and $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ can be different from $\boldsymbol{u}^\star_{k+1}$. Adding filter-based constraints to Problem (2.6) leads to the following NLP:

$$\begin{aligned} \boldsymbol{u}_{k+1} := \arg\min_{\boldsymbol{u}} \quad & \Phi_{\mathrm{KMAy},k}(\boldsymbol{u}) := \Phi_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u})) \\ \text{s.t. } \boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u}) := & \begin{bmatrix} \boldsymbol{G}^{(1)}_{\mathrm{KMAy},k}(\boldsymbol{u}) := \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u})) \\ \boldsymbol{G}^{(2)}_{\mathrm{KMAy},k}(\boldsymbol{u}) := \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) \end{bmatrix} \leq \boldsymbol{0}. \end{aligned} \tag{3.2}$$

The RTO algorithm using (3.2) instead of (2.6)-(2.7) is referred to as output modified adaptation with filter-based constraint (KMAy) and is summarized as follows [37]:

Output Modifier Adaptation with filter-based constraints (KMAy)

Initialization. Provide $\boldsymbol{u}_0$. Choose $\boldsymbol{K} \in \mathbb{R}^{n_u \times n_u}$ such that it is a diagonal matrix with diagonal elements $K_i \in (0, 1]$, $\forall i \in \{1, \ldots, n_u\}$.
for $k = 0 \to \infty$

1. Apply the inputs $\boldsymbol{u}_k$ to the plant and wait for steady state.
2. Measure the plant outputs $\boldsymbol{y}_p(\boldsymbol{u}_k)$, and estimate the plant output gradients $\nabla_{\boldsymbol{u}} \boldsymbol{y}_p$ at $\boldsymbol{u}_k$. These estimates can require data from perturbed operating points in the neighborhood of $\boldsymbol{u}_k$.
3. Evaluate the modifiers (2.4)-(2.5).
4. Compute $\boldsymbol{u}_{k+1}$ by solving Problem (3.2).

end

At first sight, KMAy and MAy look very similar, but the addition of constraints clearly affects the properties of the model-based optimization problem, with a risk of loosing the property to converge to the plant optimum

of MAy. Indeed, it is shown hereafter that this property is not lost if the additional filter-based constraints satisfy certain conditions [37, 38].

Next, the following properties of KMAy are discussed:
§ 3.2: Plant first-order optimality upon convergence;
§ 3.3: (Modified-)model-based feasibility of the iterates;
§ 3.4: Model adequacy condition of KMAy.

3.2. KKT Matching Upon Convergence

The matrix $\boldsymbol{K}$ being explicitly used in the formulation of the optimization Problem (3.2), its structure and eigenvalues can affect the optimality conditions. Thus, it is clear that rules for selecting $\boldsymbol{K}$ are needed. The current subsection identifies key conditions on $\boldsymbol{K}$ and proves that the most important properties of MAy can be preserved:
• Lemma 1 discusses the filter structure enforcing the geometrical similarity between Problems (2.1) and (3.2).
• Lemma 2 shows that a constraint qualification of Problem (3.2) holds at $\boldsymbol{u}_k$.
• Theorem 1 states the first-order plant optimality upon convergence property of KMAy.

MA ensures the (affine) matching of the behaviors of the plant and modified model cost and constraint functions at $\boldsymbol{u}_k$, as shown in (2.8). However, with KMAy, the matching between the plant and model constraints is less obvious as the model-based optimization problem has two times more constraints than the plant optimization problem. Using the notion of cone of feasible directions (CFD), it is shown hereafter with Lemma 1 that, despite the additional constraints, the geometrical similarity between the plant and model-based problems is preserved, provided an appropriate structure is chosen for $\boldsymbol{K}$.

Definition 2. *Let $\boldsymbol{G}(\boldsymbol{u}) \leq \boldsymbol{0}$ be n_g-dimensional vector of constraints of an optimization problem, with $\boldsymbol{u} \in \mathbb{R}^{n_u}$ its decision variables. At any feasible point $\boldsymbol{u}_k$, such that $\boldsymbol{G}(\boldsymbol{u}_k) \leq \boldsymbol{0}$, the CFD of $\boldsymbol{G}(\boldsymbol{u})$ is defined as:*

$$CFD := \{\boldsymbol{d} \in \mathbb{R}^{n_u} \mid \nabla_{\boldsymbol{u}} G_i|_{\boldsymbol{u}_k}^{\mathsf{T}} \boldsymbol{d} < 0 \text{ if } G_i(\boldsymbol{u}_k) = 0, \forall i\}. \tag{3.3}$$

The CFD is thus the set of descent directions for the active constraints at $\boldsymbol{u}$, as illustrated in Figure 2.

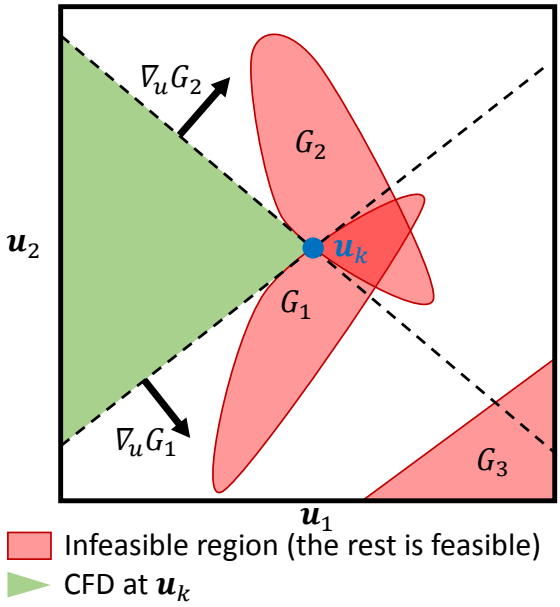


Figure 2: (Definition 2) The CFD (green-shaded area) of an optimization problem at a point $\boldsymbol{u}_k$, where the constraints G_1 and G_2 are active.

Lemma 1. *Consider the KMAy optimization Problem (3.2), the associated KMAy algorithm and the plant optimization Problem* (2.1)*. If the filter has the following structure:*

$$\boldsymbol{K} = K\boldsymbol{I}_{n_u}, \tag{3.4}$$

where $\boldsymbol{I}_{n_u}$ *is the* $n_u \times n_u$ *identity matrix with* K *a scalar* $\in (0, 1]$*, then the CFDs of Problems* (2.1) *and* (3.2) *are identical at* $\boldsymbol{u}_k$*.*

Proof. A simplified version of this proof is available in [37]. The proof follows four steps, whereby the relations between the CFDs of the constraints $\boldsymbol{G}_p(\boldsymbol{u})|_{\boldsymbol{u}_k}$, $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$, $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$, and $\boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ are identified:

- *Step 1:* The CFDs of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and of $\boldsymbol{G}_p(\boldsymbol{u})|_{\boldsymbol{u}_k}$ are the same;
- *Step 2:* The CFDs of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$ are the same;
- *Step 3:* $\boldsymbol{u} \in$ the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k} \Leftrightarrow \boldsymbol{v}_k(\boldsymbol{u}) \in$ the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$.

- *Step 4:* The CFDs of $\boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and of $\boldsymbol{G}_p(\boldsymbol{u})|_{\boldsymbol{u}_k}$ are the same.

Step 1: Since the values and gradients of $\boldsymbol{G}_p(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ are the same, see (2.8), so are their respective CFDs, see (3.3).

Step 2: At $\boldsymbol{u}_k$ Equation (3.1) reads:

$$\boldsymbol{v}_k(\boldsymbol{u}_k) = \boldsymbol{K}^{-1}(\boldsymbol{u}_k - \boldsymbol{u}_k) + \boldsymbol{u}_k = \boldsymbol{u}_k, \tag{3.5}$$

and thus:

$$\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}_k)) = \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}_k). \tag{3.6}$$

Equations (2.8) and (3.6) show that, $\forall \boldsymbol{u}_k$, $\boldsymbol{G}_p$, $\boldsymbol{G}_{\mathrm{MAy},k}$ and $\boldsymbol{G}_{\mathrm{MAy},k}$ share the same values. These constraints are therefore (always) simultaneously (in-) active. Also, by applying chain rule and from (3.5), and (3.6), the gradient w.r.t. $\boldsymbol{u}$ of $X := \{\Phi,\, G_1,\, ...,\, G_{n_g}\}$ reads:

$$\begin{aligned} \nabla_{\boldsymbol{u}} X_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k} &= \nabla_{\boldsymbol{v}_k} X_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{v}_k(\boldsymbol{u}_k)} \\ &\qquad \nabla_{\boldsymbol{u}} \boldsymbol{v}_k(\boldsymbol{u})|_{\boldsymbol{u}_k}\,, \\ &= \nabla_{\boldsymbol{u}} X_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}\, \boldsymbol{K}^{-1}. \end{aligned} \tag{3.7}$$

Injecting (3.4) into (3.7) leads to:

$$\nabla_{\boldsymbol{u}} X_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k} = K^{-1}\, \nabla_{\boldsymbol{u}} X_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}\,. \tag{3.8}$$

It immediately follows that the CFDs of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$ are the same , since $\forall K > 0$ and $\boldsymbol{d} \in \mathbb{R}^{n_u}$ (see Definition 2):

$$\begin{aligned} & \nabla_{\boldsymbol{u}} G_{i,\mathrm{MAy},k}|_{\boldsymbol{u}_k}^{\mathsf{T}}\, \boldsymbol{d} < 0, \\ \Leftrightarrow \quad & K^{-1}\, \nabla_{\boldsymbol{u}} G_{i,\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}^{\mathsf{T}}\, \boldsymbol{d} < 0, \\ \Leftrightarrow \quad & \nabla_{\boldsymbol{u}} G_{i,\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}^{\mathsf{T}}\, \boldsymbol{d} < 0, \end{aligned} \tag{3.9}$$

$\forall i$ corresponding to an active constraint at $\boldsymbol{u}_k$, $G_{i,\mathrm{MAy},k}(\boldsymbol{u}_k) = G_{i,\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}_k)) = 0$. With equation (3.6), this proves Step 2.

Step 3: The filter (3.4) is such that, $\forall \boldsymbol{u} \in \mathbb{R}^{n_u}$, the points $\boldsymbol{u}_k$, $\boldsymbol{u}$ and $\boldsymbol{v}_k(\boldsymbol{u})$ are aligned in this order in the input space. Therefore the directions $\boldsymbol{d}_1 = (\boldsymbol{u} - \boldsymbol{u}_k)/||\boldsymbol{u} - \boldsymbol{u}_k||_2$ and $\boldsymbol{d}_2 = (\boldsymbol{v}_k(\boldsymbol{u}) - \boldsymbol{u}_k)/||\boldsymbol{v}_k(\boldsymbol{u}) - \boldsymbol{u}_k||_2$ are the same. As a result, since the CFDs of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$ are the same (see Step 2), $\forall$ $\boldsymbol{u} \in$ the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$, $\boldsymbol{v}_k(\boldsymbol{u})$ is in the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$.

Step 4: By definition, a point $\boldsymbol{u}$ is in the CFD of $\boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ if and only if: (i) $\boldsymbol{u}$ is in the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u})|_{\boldsymbol{u}_k}$, and (ii) $\boldsymbol{v}_k(\boldsymbol{u})$ is in the CFD of $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$. Indeed, both constraints are part of Problem (3.2). As proven at Step 3, the conditions (i) and (ii) are equivalent when (3.4) holds. Then, it is sufficient to focus condition (i), which, according to Step 1, is equivalent to $\boldsymbol{u} \in$ the CFD of $\boldsymbol{G}_p(\boldsymbol{u})|_{\boldsymbol{u}_k}$. This concludes the proof. □

Lemma 1 and its proof are illustrated hereafter with a simple numerical example.

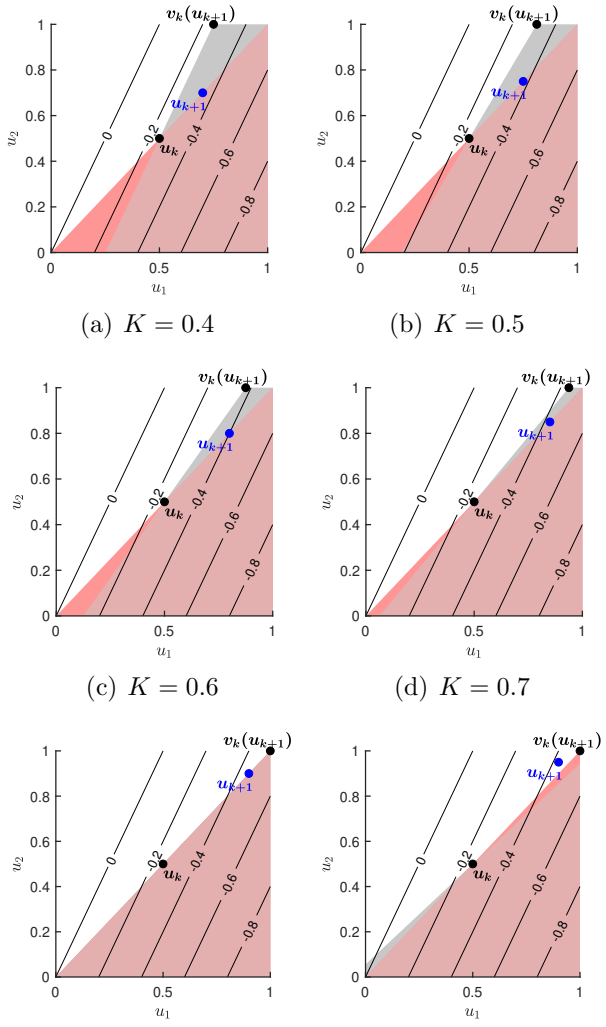
Example 1. *(Effect of the filter structure KMAy) Consider the following problem:*

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & \phi(\boldsymbol{u}, \boldsymbol{y}_p(\boldsymbol{u})) := y_{1,p} \\ s.t. \quad & g(\boldsymbol{u}, \boldsymbol{y}_p(\boldsymbol{u})) := y_{2,p} \leq \boldsymbol{0}, \\ & \boldsymbol{u} \in [0,1] \times [0,1], \end{aligned} \tag{3.10}$$

where $\boldsymbol{u} \in \mathbb{R}^2$, $\boldsymbol{y}_p \in \mathbb{R}^2$, *and*

$$y_{1,p} := \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}^{\mathsf{T}} \boldsymbol{u}, \qquad y_{2,p} := \begin{bmatrix} 1 \\ -1 \end{bmatrix}^{\mathsf{T}} \boldsymbol{u}. \tag{3.11}$$

The optimal solution of Problem (3.10) *is* $\boldsymbol{u}_p^\star = [1,1]^{\mathsf{T}}$. *In this example, no plant-model mismatch is considered and KMAy is initialized at* $\boldsymbol{u}_0 = [0.5, 0.5]^{\mathsf{T}}$. *The filter is defined as* $\boldsymbol{K} = diag(0.8, K)$, *where* $K \in (0,1]$, *and the simulation results for* $K = \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ *are depicted in Figure 3. The red-shaded area is the exterior of the CFD of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ *and does not depend on* K. *Indeed, as shown in Step 1 of Lemma 1 proof, the CFDs of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ *and of the plant are always the same, irrespective of* $\boldsymbol{K}$. *On the other hand, the grey-shaded area, i.e. the exterior of the CFD of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$, *depends on* K. *The only case for which the CFDs of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ *and* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$ *are the same (and the shaded areas are superimposed), is when the filter satisfies* (3.4). *This illustrates Step 2 of the proof of Lemma 1. Also, when the filter satisfies* (3.4), *the points* $\boldsymbol{u}_k$, $\boldsymbol{u}_{k+1}$, *and* $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ *are aligned in this specific order (Step 3). Therefore, if* $\boldsymbol{u}_{k+1}$ *lies in the CFD of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$, *then, because (i) the CFDs of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ *and* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$ *are the same and (ii) given that* $\boldsymbol{u}_k$, $\boldsymbol{u}_{k+1}$, *and* $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ *are aligned in this order,* $\boldsymbol{v}_k(\boldsymbol{u}_{k+1})$ *also lies in the CFD of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$, *which illustrates Step 3 of Lemma 1.*



(a) $K = 0.4$ (b) $K = 0.5$

(c) $K = 0.6$ (d) $K = 0.7$

(e) $K = 0.8$ (f) $K = 0.9$

Figure 3: (**Example 1**) The red- and grey-shaded areas are the exterior of the CFDs of $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ and $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$, respectively. The red-shaded areas are also the infeasible regions, and anywhere else is feasible. The cost function is represented by the contour curves. The filter structure affect the points $[\boldsymbol{u}_k, \boldsymbol{u}_{k+1}, \boldsymbol{v}_k(\boldsymbol{u}_{k+1})]$ alignment and the gray shaded area.

The number of active constraints at $\boldsymbol{u}_k$ being doubled (see (3.6)), the constraint qualification of Problem (3.2) can be affected at $\boldsymbol{u}_k$. In particular, the standard linear constraint qualification, which is easily verified when all constraints are known, assumed or shown to be linearly independent at the solution point, can be lost since the constraints and the duplicated constraints share the same values (simultaneously active) while their respective gradients are proportional (and thus no longer independent). The following Lemma proves that another constraint qualification holds for Problem (3.2) at $\boldsymbol{u}_k$.

Lemma 2. *If the LICQ holds at any $\boldsymbol{u} \in \mathcal{F}_p$ for Problem (2.1), then the Mangasarian-Fromovitz constraint qualification (MFCQ) holds for the Problem (3.2) at $\boldsymbol{u}_k$, $\forall k$, provided $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$ and $K \in (0, 1]$.*

Proof. [37] To prove that MFCQ holds, we need to show that there exists a direction $\boldsymbol{d} \in \mathbb{R}^{n_u}$ for each active constraint $G_{i,\mathrm{KMAy},k}$ at $\boldsymbol{u}_k$, such that $\nabla_{\boldsymbol{u}} G_{i,\mathrm{KMAy},k}|_{\boldsymbol{u}_k}^{\mathsf{T}} \boldsymbol{d} < 0$. Since LICQ holds for Problem (2.1) $\forall \boldsymbol{u} \in \mathcal{F}_p$, its CFD is never empty. Lemma 1 holds since $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$ and $K \in (0, 1]$. Thus, $\forall k$, the CFD for Problems (2.1) and (3.2) at $\boldsymbol{u}_k$ are identical. Therefore, the requested $\boldsymbol{d}$ direction exists and MFCQ holds for Problem (3.2) at $\boldsymbol{u}_k$. □

In summary, the standard LICQ assumption for the plant is sufficient to guarantee MFCQ for Problem (3.2) at $\boldsymbol{u}_k$ $\forall k$, and, thus, upon convergence. The following theorem shows that KMAy can only converge to a KKT point of the plant.

Theorem 1 (1st-order NCO matching upon convergence). *If the input sequence generated by KMAy converges to a limit value $\boldsymbol{u}_\infty = \lim_{k\to\infty} \boldsymbol{u}_k$ with $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$ and $K \in (0, 1]$, then $\boldsymbol{u}_\infty$ is a KKT-point of Problem (2.1).*

Proof. [37]

According to Lemma 2, MFCQ holds for Problem (3.2) at $\boldsymbol{u}_k, \forall k$ and thus also at $\boldsymbol{u}_\infty$. Said differently, upon convergence, the MFCQ holds for the model-based optimization Problem (3.2) modified at $\boldsymbol{u}_\infty$. Therefore, the KKT-condition of optimality of the modified model-based optimization Problem (3.2) at $\boldsymbol{u}_\infty$ corresponds to the existence of $\boldsymbol{\mu} := [\boldsymbol{\mu}^{(1)\mathsf{T}}, \boldsymbol{\mu}^{(2)\mathsf{T}}]^{\mathsf{T}} \in$

$\mathbb{R}^{2n_g}$ [39] such that:

$$\boldsymbol{G}_{\mathrm{KMAy},\infty}(\boldsymbol{u}_\infty) \leq \boldsymbol{0}, \tag{3.12}$$
$$\boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{G}_{\mathrm{KMAy},\infty}(\boldsymbol{u}_\infty) = 0, \tag{3.13}$$
$$\boldsymbol{\mu} \geq \boldsymbol{0}, \tag{3.14}$$
$$\nabla_{\boldsymbol{u}} \Phi_{\mathrm{KMAy},\infty}|_{\boldsymbol{u}_\infty} + \boldsymbol{\mu}^{\mathsf{T}} \nabla_{\boldsymbol{u}} \boldsymbol{G}_{\mathrm{KMAy},\infty}|_{\boldsymbol{u}_\infty} = \boldsymbol{0}, \tag{3.15}$$

where $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ are the KKT-multipliers associated to $\boldsymbol{G}^{(1)}_{\mathrm{KMAy,k}}$ and $\boldsymbol{G}^{(2)}_{\mathrm{KMAy,k}}$, respectively. From the definition of $\boldsymbol{G}_{\mathrm{KMAy},k}$ of equation(3.2) and (3.8), (3.15) can be rewritten as:

$$\nabla_{\boldsymbol{u}} \Phi_{\mathrm{MAy},\infty}|_{\boldsymbol{u}_\infty} + \left(\boldsymbol{\mu}^{(1)} + K\boldsymbol{\mu}^{(2)}\right)^{\mathsf{T}} \nabla_{\boldsymbol{u}} \boldsymbol{G}_{\mathrm{MAy},\infty}|_{\boldsymbol{u}_\infty} = \boldsymbol{0}. \tag{3.16}$$

From Equations (2.8) and (3.12), it follows that:

$$\boldsymbol{G}_p(\boldsymbol{u}_\infty) \leq \boldsymbol{0}. \tag{i}$$

With (3.2), (3.6) and (3.12), we have:

$$\begin{aligned} {\boldsymbol{\mu}^{(1)}}^{\mathsf{T}} \boldsymbol{G}_{\mathrm{MAy},\infty}(\boldsymbol{u}_\infty) &= 0, \\ {\boldsymbol{\mu}^{(2)}}^{\mathsf{T}} \boldsymbol{G}_{\mathrm{MAy},\infty}(\boldsymbol{v}_\infty(\boldsymbol{u}_\infty)) = {\boldsymbol{\mu}^{(2)}}^{\mathsf{T}} &\boldsymbol{G}_{\mathrm{MAy},\infty}(\boldsymbol{u}_\infty) = 0. \end{aligned}$$

Multiplying the second equation by K, summing the two equations and rearranging yields:

$$\left({\boldsymbol{\mu}^{(1)}}^{\mathsf{T}} + K{\boldsymbol{\mu}^{(2)}}^{\mathsf{T}}\right) \boldsymbol{G}_{\mathrm{MAy},\infty}(\boldsymbol{u}_\infty) = 0.$$

Defining $\boldsymbol{\mu}_p := \boldsymbol{\mu}^{(1)} + K\boldsymbol{\mu}^{(2)}$ and observing that $\boldsymbol{G}_{\mathrm{MAy},\infty}(\boldsymbol{u}_\infty) = \boldsymbol{G}_p(\boldsymbol{u}_\infty)$ leads to:

$$\boldsymbol{\mu}_p^{\mathsf{T}} \boldsymbol{G}_p(\boldsymbol{u}_\infty) = 0. \tag{ii}$$

From (3.14), $\boldsymbol{\mu}^{(1)} \geq \boldsymbol{0}$, $\boldsymbol{\mu}^{(2)} \geq \boldsymbol{0}$, and $K > 0$:

$$\boldsymbol{\mu}_p = \boldsymbol{\mu}^{(1)} + K\boldsymbol{\mu}^{(2)} \geq 0. \tag{iii}$$

Finally combining (2.8), (3.16) and (iii) leads to:

$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}_\infty} + \boldsymbol{\mu}_p^{\mathsf{T}} \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}_\infty} = \boldsymbol{0}. \tag{iv}$$

By writing down (i), (ii), (iii) and (iv) in matrix form, the KKT conditions of the plant-optimization Problem (2.1) are identified:

$$\boldsymbol{G}_p(\boldsymbol{u}_\infty) \leq \boldsymbol{0}, \tag{i}$$
$$\boldsymbol{\mu}_p^\mathsf{T} \boldsymbol{G}_p(\boldsymbol{u}_\infty) = 0, \tag{ii}$$
$$\boldsymbol{\mu}_p \geq \boldsymbol{0}, \tag{iii}$$
$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}_\infty} + \boldsymbol{\mu}_p^\mathsf{T} \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}_\infty} = \boldsymbol{0}. \tag{iv}$$

At this point, it has been shown that if $(\boldsymbol{u}_\infty, \boldsymbol{\mu})$ is a KKT-point of Problem (3.2), then $(\boldsymbol{u}_\infty, \boldsymbol{\mu}_p)$ is a KKT-point of Problem (2.1) with $\boldsymbol{\mu}_p := \boldsymbol{\mu}^{(1)} + K\boldsymbol{\mu}^{(2)}$, which concludes the proof. □

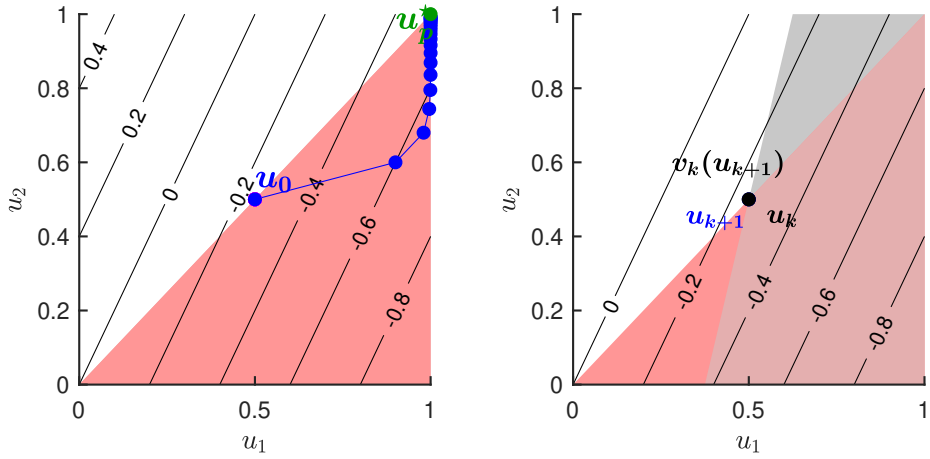
Theorem 1 shows that the addition of filter-based constraints to the model-based optimization problem (3.2) is not detrimental to the KKT-matching – upon convergence – property of MAy, provided the filter structure satisfies (3.4). Forcing (3.4) might be seen to be a limitation of KMAy, but, as illustrated in the next example, it is indeed very much welcome, and could also be even for MAy. Indeed, selecting a filter with a different structure can lead to constraint violations *even when the model is perfect and the problem convex.*

Example 2. *Consider again the problem of Example 1, but this time with the filter* $\boldsymbol{K} = diag(0.8, 0.2)$*, which purposely does not satisfy* (3.4)*. Simulation results with MAy and KMAy are shown in Figure 4. Figure 4b shows the effects of the input filter on the CFD of* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{u})|_{\boldsymbol{u}_k}$ *and* $\boldsymbol{G}_{\mathrm{MAy,k}}(\boldsymbol{v}_k(\boldsymbol{u}))|_{\boldsymbol{u}_k}$*. Here, KMAy remains stuck at its initial point* $\boldsymbol{u}_0 = [0.5, 0.5]$ *while MAy converges to the plant optimum, but with iterates lying in the infeasible part of the input space, as illustrated with Figure 4a. The fact that KMAy does not converge to the plant optimum does not mean that it has failed. While MAy* must *violate the plant constraints (despite the availability of a perfect model and the convexity of this trivial LP) to converge, KMAy, by construction, prevents potential violations of plant constraints (here due to the wrong choice of the filter) by sacrificing optimality upon convergence.*

Further analysis of the safety improvements provided by KMAy are provided in the next subsection.

3.3. Modified Model-Based Feasibility

As discussed before, KMAy provides safety improvements to the classical modifier adaptation algorithms. Of course, KMAy does not provide "abso-



(a) MAy iterates. (b) KMAy iterates.

Figure 4: (**Example 2**) Violations of plant constraints (MAy) vs. loss of optimality only (KMAy), for an inappropriate choice of the structure of the filter, even when the model-based optimization problem is ideal (no plant-model mismatch) and convex.The red- and gray shaded areas have the same meanings as in Figure 3.

lute" plant feasibility guarantees as in [14, 16], due to the fact that it does not use additional, most often unavailable, plant information. KMAy instead ensures modified model-based feasibility at the operating points that will be applied to the plant. Yet, modified model-based feasibility is not plant feasibility, but whenever the model is capable of fairly predicting the constraints, KMAy is sufficient to reduce significantly the risk of plant constraint violations as it avoids *implementing a filtered input whereby the model feasibility has not been checked* [38].

Hereafter, sufficient conditions for feasibility of a RTO iterate obtained with MAy or with KMAy are compared. It is shown that when the *input* filter decreases, KMAy becomes more reliable yet conservative, while MAy does not get any safer.

Definition 3. *Let define the following objects:*
• *A* model *is an object* $\mathfrak{m}$ *in the universe of models* $\mathcal{M}$*:* $\mathfrak{m} \in \mathcal{M}$.
• *At each iteration* k, $\mathcal{C}^k_{\text{KMAy}}$ *is the set of inputs that satisfy the constraints of Problem* (3.2) *and* $\mathcal{C}^k_{\text{MAy}}$ *denotes the set of inputs generated by the application of the filter of* (2.7) *to any input satisfying the constraint* (2.6)*, which can*

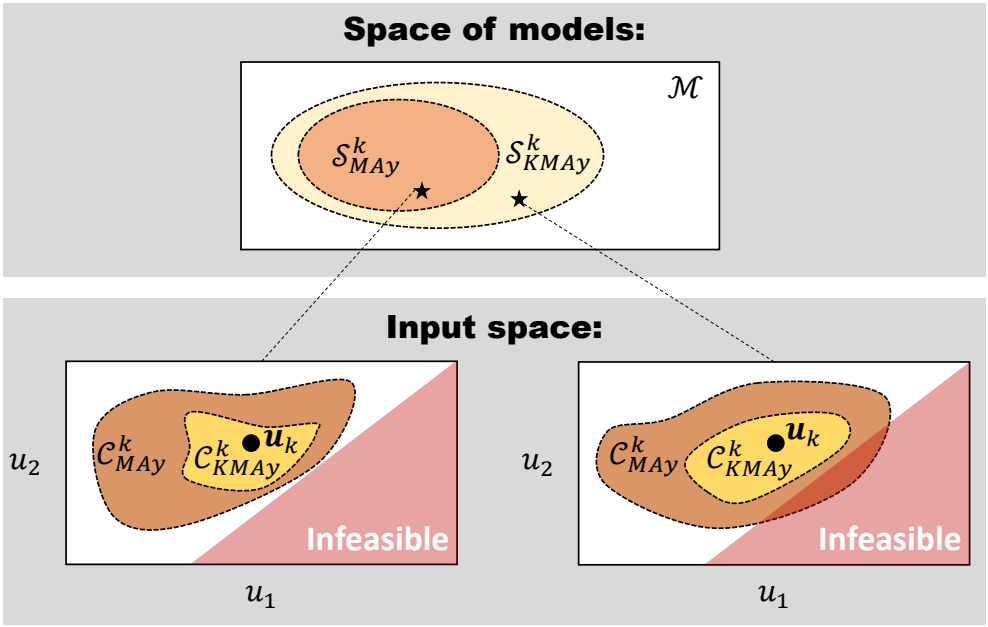


Figure 5: Illustration of Definition 3 and Proposition 1.

be written in mathematical form as:

$$\mathcal{C}^k_{\mathrm{KMAy}} := \{\boldsymbol{u} \in \mathbb{R}^{n_u} | \boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u}) \leq \boldsymbol{0}\},$$
$$\mathcal{C}^k_{\mathrm{MAy}} := \{\boldsymbol{u} \in \mathbb{R}^{n_u} | \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{K}^{-1}(\boldsymbol{u} - \boldsymbol{u}_k) + \boldsymbol{u}_k) \leq \boldsymbol{0}\},$$

respectively. In other words, $\mathcal{C}^k_{\mathrm{Y}}$ *are the sets of candidate inputs* $\boldsymbol{u}$ *to which belong the next operating conditions* $\boldsymbol{u}_{k+1}$ *that the RTO algorithm* $\mathrm{Y} \in \{\mathrm{MAy}, \mathrm{KMAy}\}$ *will suggest to apply to the plant.*
At each iteration k *and for each method* $\mathrm{Y} \in \{\mathrm{MAy}, \mathrm{KMAy}\}$, $\boldsymbol{\mathcal{S}}^k_{\mathrm{Y}} \subseteq \boldsymbol{\mathcal{M}}$ *are the sets of models* $\boldsymbol{m}$ *such that the candidate inputs* $\boldsymbol{u}_{k+1}$ *are feasible for the plant, i.e.,*

$$\boldsymbol{\mathcal{S}}^k_{\mathrm{Y}} := \left\{ m \in \boldsymbol{\mathcal{M}} \mid \mathcal{C}^k_{\mathrm{Y}} \subseteq \mathcal{F}_p \right\}.$$

From the definition of $\boldsymbol{\mathcal{S}}^k_{\mathrm{Y}}$, it is clear that $\boldsymbol{m} \in \boldsymbol{\mathcal{S}}^k_{\mathrm{Y}}$ is a *sufficient condition for the plant feasibility at the next iterate.*

Assuming that plant modeling corresponds to the selection of a model $\boldsymbol{m} \in \boldsymbol{\mathcal{M}}$, the following proposition shows that the chance that the selected model satisfies the sufficient condition for the plant feasibility at the next

iterate is always higher when KMAy is used instead of MAy.

Proposition 1. *At each iteration k, the set of models satisfying the sufficient condition for plant feasibility at $k+1$ with MAy is a subset of the set of models satisfying it with KMAy, i.e.,*

$$\boldsymbol{\mathcal{S}}^k_{\mathrm{MAy}} \subseteq \boldsymbol{\mathcal{S}}^k_{\mathrm{KMAy}}.$$

Proof. See [38]. □

Indeed, Proposition 1 implies that any model satisfying the aforementioned sufficient condition for feasibility with MAy also satisfies it for KMAy, while the opposite does not hold. Thus, from the point of view of the feasibility of the iterates, KMAy should be preferred to MAy, since the chances that $\boldsymbol{u}_{k+1}$ is in $\mathcal{F}_p$ are bigger.

Remark 4. *One way to interpret the proof of Proposition 1 is that adding more constraints to the optimization problem can only shrink the subset of the input space where the candidates $\boldsymbol{u}_{k+1}$ lie. This is illustrated with Figure 5 where $\boldsymbol{\mathcal{S}}^k_{\mathrm{KMAy}}$ is always a subset of $\boldsymbol{\mathcal{S}}^k_{\mathrm{MAy}}$. In fact, adding constraints to an optimization technique is a way to increase safety, but of course, this can lead to the loss of key properties, such as plant optimality. It might be interesting to investigate the addition of constraints that are never active at $\boldsymbol{u}_k$, as they will not affect the geometrical similarity between the plant-based and model-based optimization problems at $\boldsymbol{u}_k$. For example, step-size limitation between two consecutive RTO iterations, as with trust-region (TR) methods [18, 19, 20], could be suitable. However, this is beyond the scope of this article.*

Now, it is shown that irrespective of accuracy of the available model, reducing the filter gain increases the safety of KMAy. This property relies on the following definition and assumption:

Definition 4. *Define $\mathcal{D}_k$ as the set of inputs where the model provides appropriate predictions of the plant infeasibility, i.e.:*

$$\begin{aligned}\mathcal{D}_k := \{\boldsymbol{u} \in \mathcal{U} \mid\ & \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) > \boldsymbol{0} \text{ if } \boldsymbol{G}_p(\boldsymbol{u}) > \boldsymbol{0}, \\ & \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) \in \mathbb{R}^{n_g} \text{ if } \boldsymbol{G}_p(\boldsymbol{u}) \leq \boldsymbol{0}\}.\end{aligned}$$

Clearly, $\forall \boldsymbol{u} \in \mathcal{D}_k$, the fact that the plant constraints are infeasible ($\boldsymbol{G}_p(\boldsymbol{u}) > \boldsymbol{0}$) is correctly predicted and $\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) > \boldsymbol{0}$. On the other hand, in $\mathcal{D}_k$, anything can be predicted ($\boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) \in \mathbb{R}^{n_g}$) by the model where plant constraints are satisfied ($\boldsymbol{G}_p(\boldsymbol{u}) \leq \boldsymbol{0}$).

Assumption 4. *At any iteration k:*

$$\boldsymbol{u}_k \in \mathcal{D}_k^o \tag{3.17}$$

where $\mathcal{D}_k^o$ is the (strict) interior of $\mathcal{D}_k$.

Remark 5. *If the filter is applied to the inputs (not to the modifiers), then Assumption 4 is likely to hold since at $\boldsymbol{u}_k$ the modified-model and the plant have local identical first-order properties thanks to the affine corrections of the model, see* (2.8)*. This is not the case when the filter is performed at the level of the modifiers since the equalities* (2.8) *only hold upon convergence as these corrections are only partially implemented* (2.9)*. This implies that even close to $\boldsymbol{u}_k$, the predictions of plant feasibility with the modified model can be unreliable, as illustrated in Figure 6, whereby a case where a plant constraint is activated at $\boldsymbol{u}_k$ is depicted. On the left-hand side of Figure 6, input filtering is implemented and it is seen that the modified model-based constraint matches the plant constraint around $\boldsymbol{u}_k$. This is not the case with modifiers filtering (right-hand side), whereby the whole neighborhood of $\boldsymbol{u}_k$ is predicted to be feasible. The fact that the modified model-based constraint is not activated at $\boldsymbol{u}_k$ makes the next step more likely to violate the plant constraints*[2]*.*

Remark 6. *The conceptual analysis of Figure 6 implicitly carries the assumption that accurate estimates of the plant values and gradients are available. Since in practice these estimates are polluted by noises and other uncertainties, filtering the modifiers might stay a useful tool to "average" plant measurements and reduce the uncertainties impacts on the value of modifiers. The most efficient methods to do so generally use several, if not all, past data [21, 25, 11, 23, 28], see Remark 1. In theory, nothing prevents*

[2]In fact, filtering the modifiers implies that the cost and constraint functions of the modified model does not match locally those of the plant. Therefore, plant constraints can be over- or underestimated. Figure 6 illustrate the case where constraints are underestimated

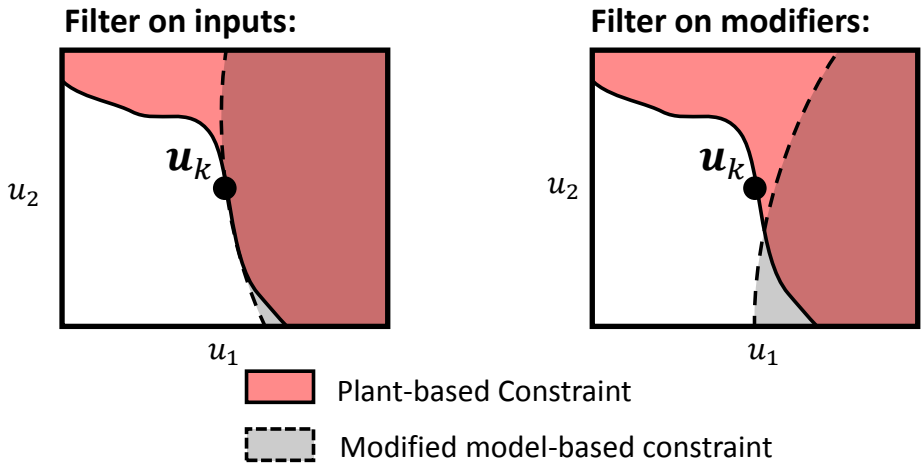


Figure 6: Filtering the inputs versus filtering the modifiers. The exterior of $\mathcal{D}_k$ is the part of the input space where the gray domain does not overlap the red domain.

KMAy to be augmented by the addition of a filter at the level of the input-output mapping modifiers. This way, the combination of the two filters would add management of noise to the benefits of incoporating the input filter in the optimization problem, but the safety argument illustrated by Figure 6 would be weakened. In any case, the measured outputs that are used for computing the modifiers or estimating plant gradients can be filtered, using classical signal processing techniques.

With the next theorem, it is shown when the filter gain in KMAy decreases, $\mathcal{C}^k_{\text{KMAy}}$ shrinks towards $\mathcal{D}_k$ and, that if K is small enough, $\mathcal{C}^k_{\text{KMAy}}$ is entirely within $\mathcal{D}_k$. For the latter, the feasibility of the next iterate is guaranteed given that in $\mathcal{D}_k$, plant infeasibility is correctly predicted. *Notice that no assumption about the quality of the model is necessary.*

Theorem 2. *If (i) Assumption 4 holds, and (ii) $\mathcal{U}$ is bounded, then there exist a diagonal filter $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$ such that $\mathcal{C}^k_{\text{KMAy}} \subseteq \mathcal{F}_p$ and, in turn, such that $\boldsymbol{u}_{k+1} \in \mathcal{F}_p$.*

Proof. See [38]. □

Remark 7. *A similar result cannot be obtained for MAy, as can be easily understood by inspecting Figure 1. Here, reducing K with MAy would move*

$\boldsymbol{u}_{k+1}$ *closer to* $\boldsymbol{u}_k$*, but still on the segment* $[\boldsymbol{u}^\star_{k+1}, \boldsymbol{u}_k]$*, while the whole segment lies in the red-shaded area and is therefore predicted to be infeasible by the modified model. Then, this would ultimately just slow down the convergence to* $\boldsymbol{u}^\star_{k+1}$ *and increase the number of infeasible RTO iterates for the modified model, and most likely also for the plant.*

3.4. Model Adequacy Condition

A model is adequate for a given iterative RTO method if and only if it is capable of predicting that the (unknown) plant optimum $\boldsymbol{u}^\star_p$ is a local minimum of the model-based optimization problem. This will be the case for MAy if the reduced Hessian of the model based optimization problem modified and evaluated at $\boldsymbol{u}^\star_p$ is positive definite [5]. In this subsection it is shown that with KMAy this condition is significantly relaxed, as shown in the next theorem, which requires the following Lemma.

Lemma 3. *Consider the two following algebra results:*

(a) Let $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *be two arbitrary* $\mathbb{R}^{n\times n}$ *matrices, and* K *be a scalar in* $\mathbb{R}^{+*}$*. There can be some values of* K *such that* $\boldsymbol{A} + K\boldsymbol{B} > 0$*.*

(b) Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ *be a positive definite matrix,* $\boldsymbol{B} \in \mathbb{R}^{n\times n}$*, and* $K \in \mathbb{R}^{+*}$*. Then,* $\exists K_{max} > 0$*, such that* $\forall K \leq K_{max}$*,* $\boldsymbol{A} + K\boldsymbol{B} > 0$*, and* $\forall K > K_{max}$*,* $\boldsymbol{A} + K\boldsymbol{B} \not> 0$*.*

Proof. Denote first:

$$a_v := \boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}, \qquad b_v := \boldsymbol{v}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{v}, \qquad \boldsymbol{v}^{\mathrm{T}}(\boldsymbol{A} + K\boldsymbol{B})\boldsymbol{v} = a_v + Kb_v.$$

Consider (a): Define the set $\mathcal{K}_v$, such that $\forall K \in \mathcal{K}_v$, $a_v + Kb_v > 0$ as:

$$\mathcal{K}_v := \begin{cases} (0, -a_v/b_v) & \text{if } a_v > 0 \text{ and } b_v < 0, \\ (-a_v/b_v, \infty) & \text{if } a_v < 0 \text{ and } b_v > 0, \\ (0, \infty) & \text{if } a_v \geq 0 \text{ and } b_v \geq 0, \\ \emptyset & \text{otherwise.} \end{cases} \tag{3.18}$$

The intersection of all $\mathcal{K}_v$ sets $\forall \boldsymbol{v} \in \mathbb{R}^n$ is denoted:

$$\mathcal{K}_{tot} := \bigcap_{\boldsymbol{v}\in\mathbb{R}^n} \mathcal{K}_v. \tag{3.19}$$

From the definitions of $\mathcal{K}_{tot}$, any K value in $\mathcal{K}_{tot}$ is such that $a_v + Kb_v = \boldsymbol{v}^{\mathrm{T}}(\boldsymbol{A} + K\boldsymbol{B})\boldsymbol{v} > 0$, $\forall \boldsymbol{v} \in \mathbb{R}^n$. Said differently, $\mathcal{K}_{tot}$ is the set of values of K such that $\boldsymbol{A} + K\boldsymbol{B} > 0$. Two cases can be distinguished:

- if $\mathcal{K}_{tot} = \emptyset$, then $\nexists K \geq 0$ such that $\boldsymbol{A} + K\boldsymbol{B} > 0$; this is typically the case when $\boldsymbol{A} \leq 0$ and $\boldsymbol{B} \leq 0$;
- otherwise, $\mathcal{K}_{tot} = [\underline{K}_{tot}, \overline{K}_{tot}]$.

The latter case is when there are values of $K \geq 0$ such that $\boldsymbol{A} + K\boldsymbol{B} > 0$. This case happens, and one example is when $\boldsymbol{A}$ and $\boldsymbol{B}$ are both positive definite, whereby any $K \geq 0$ is such that $\boldsymbol{A} + K\boldsymbol{B} > 0$. The intersection of all K_v is not empty and is such that $\underline{K}_{tot} = 0$ and $\overline{K}_{tot} = \infty$. This is also true when $\boldsymbol{A}$ is positive definite and $\boldsymbol{B}$ is positive semi-definite or vice-versa. Finally, when $\boldsymbol{A}$ and $\boldsymbol{B}$ are indefinite, there clearly could be some values of $K \geq 0$ such that $\boldsymbol{A} + K\boldsymbol{B} > 0$. Part (b) focuses the special case of (a) where $\boldsymbol{A} > 0$ and nothing is said about $\boldsymbol{B}$.

Consider (b): (b) is a special case of (a) where $a_v > 0$. Therefore, (3.18) reduces to:

$$\mathcal{K}_v := (0, \overline{K}_v] := \begin{cases} (0, -a_v/b_v) & \text{if } b_v < 0, \\ (0, \infty) & \text{otherwise,} \end{cases}$$

The intersection of all these sets $\forall \boldsymbol{v} \in \mathbb{R}^n$ reads:

$$\mathcal{K}_{tot} := \bigcap_{\boldsymbol{v} \in \mathbb{R}^n} \mathcal{K}_v = \big(0, \min_{\boldsymbol{v} \in \mathbb{R}^n} \{\overline{K}_v\}\big]. \tag{3.20}$$

Because $a_v > 0$ and $b_v \neq -\infty$ ($b_v \in \mathbb{R}$), $\overline{K}_v > 0$, and $\mathcal{K}_{tot}$ is *never empty* and corresponds to the set $(0, K_{max}]$ where $K_{max} := \min_{\boldsymbol{v} \in \mathbb{R}^n} \{\overline{K}_v\}$. Then, by definition $\forall K \in (0, K_{max}]$, $\boldsymbol{A} + K\boldsymbol{B} > 0$. This concludes the proof of (b). □

Theorem 3. *(Model adequacy condition for KMAy) Consider the plant optimization Problem* (2.1) *and its solution* $\boldsymbol{u}_p^\star$. *Consider the KMAy optimization Problem (3.2) and the associated KMAy algorithm. Consider the following notation:*

- *The subscript* $\star$ *denotes that modification is performed at* $\boldsymbol{u}_p^\star$, *e.g. Problem* (3.2)$|_\star$ *denotes that Problem* (3.2) *modified at* $\boldsymbol{u}_p^\star$,

- $\boldsymbol{N}_{KMAy,\star}$ *is the null space of the strongly active constraints*[3] *of Problem* (3.2)$|_\star$ *at* $\boldsymbol{u}_p^\star$.
- $\boldsymbol{N}_{MAy,\star}$ *is the null-space of the strongly active constraints of Problem* (2.6)$|_\star$ *at* $\boldsymbol{u}_p^\star$.

If (i) the filter is such that $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$*, with* $K \in (0,\, 1]$*, and (ii) the model cost function satisfies :*

$$\boldsymbol{N}_{MAy,\star}^{\mathsf{T}} \nabla^2_{\boldsymbol{u}\boldsymbol{u}} \Phi_{\mathrm{MAy},\star}(\boldsymbol{u}))|_{\boldsymbol{u}_p^\star} \boldsymbol{N}_{MAy,\star} > 0, \tag{3.21}$$

then $\exists K_{opt} \leq 1$*, such that* $\forall K \leq K_{opt}$ *the model is adequate [8] for KMAy.*

Proof. For the model to be adequate for KMAy, the 1st- and 2nd-order conditions of optimality of Problem (3.2)$|_\star$ must hold at $\boldsymbol{u}_p^\star$.

1st-order conditions of optimality: Given that $\boldsymbol{u}_p^\star$ is the plant optimum, $\exists \boldsymbol{\mu}_p \in \mathbb{R}^{n_g}$ such that $(\boldsymbol{u}_p^\star, \boldsymbol{\mu}_p)$ is a KKT point of Problem (2.1), i.e. such that:

$$\boldsymbol{G}_p(\boldsymbol{u}_p^\star) \leq \boldsymbol{0}, \tag{3.22}$$
$$\boldsymbol{\mu}_p^{\mathsf{T}} \boldsymbol{G}_p(\boldsymbol{u}_p^\star) = 0, \tag{3.23}$$
$$\boldsymbol{\mu}_p \geq \boldsymbol{0}, \tag{3.24}$$
$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}_p^\star} + \boldsymbol{\mu}_p^{\mathsf{T}}\, \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}_p^\star} = \boldsymbol{0}. \tag{3.25}$$

Now the properties of Problem (3.2)$|_\star$ at $\boldsymbol{u}_p^\star$ are deduced from (3.22)-(3.25). According to (2.8), (3.6) and (3.22):

$$\begin{aligned} \boldsymbol{G}_p(\boldsymbol{u}_p^\star) &= \boldsymbol{G}_{\mathrm{MAy},\star}(\boldsymbol{u}_p^\star) = \boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) \leq \boldsymbol{0}; \\ \boldsymbol{G}_p(\boldsymbol{u}_p^\star) &= \boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) = \boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) \leq \boldsymbol{0}. \end{aligned} \tag{3.26}$$

Thus:

$$\boldsymbol{G}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) := \begin{pmatrix} \boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) \\ \boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) \end{pmatrix} \leq \boldsymbol{0}. \tag{3.27}$$

Similarly to the proof of Theorem 1, the multipliers of Problem (3.2) are distinguished $\boldsymbol{\mu} = [\boldsymbol{\mu}^{(1)\mathsf{T}}, \boldsymbol{\mu}^{(2)\mathsf{T}}]^{\mathsf{T}}$, depending on whether they correspond to

[3]Strongly active constraints are such that their values are zero while their associated Lagrange multipliers are not.

$\boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}$ or $\boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}$, and we propose the following choice of $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ to prove the existence of a suitable $\boldsymbol{\mu}$:

$$\boldsymbol{\mu}^{(1)} = \mathbf{0}, \qquad\qquad \boldsymbol{\mu}^{(2)} = \boldsymbol{\mu}_p/K. \tag{3.28}$$

From (3.24) and $K > 0$, it is clear that the choice (3.28) is such that:

$$\boldsymbol{\mu} = [\boldsymbol{\mu}^{(1)\mathsf{T}}, \boldsymbol{\mu}^{(2)\mathsf{T}}]^{\mathsf{T}} \geq \mathbf{0} \tag{3.29}$$

From (3.28), (3.23) and (3.26), it follows:

$$\begin{aligned}
\boldsymbol{\mu}^{\mathsf{T}}\boldsymbol{G}_{\mathrm{KMAy},\star}(\boldsymbol{u}^\star_p) &= (\boldsymbol{\mu}^{(1)\mathsf{T}}, \boldsymbol{\mu}^{(2)\mathsf{T}}) \begin{pmatrix} \boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}(\boldsymbol{u}^\star_p) \\ \boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}(\boldsymbol{u}^\star_p) \end{pmatrix}, \\
&= \frac{\boldsymbol{\mu}_p}{K}^{\mathsf{T}} \boldsymbol{G}_p(\boldsymbol{u}^\star_p), \\
&= 0.
\end{aligned} \tag{3.30}$$

Finally, (3.25) is rewritten to exhibit of KKT elements for KMAy, according to (3.28):

$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}^\star_p} + \boldsymbol{\mu}_p^{\mathsf{T}}\, \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} = \mathbf{0},$$

$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}^\star_p} + \frac{1}{K}\boldsymbol{\mu}_p^{\mathsf{T}} K\, \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} = \mathbf{0},$$

$$\nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}^\star_p} + \begin{pmatrix} \mathbf{0} \\ \frac{1}{K}\boldsymbol{\mu}_p \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} \\ K\, \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} \end{pmatrix} = \mathbf{0},$$

$$\frac{1}{K}\, \nabla_{\boldsymbol{u}} \Phi_p|_{\boldsymbol{u}^\star_p} + \begin{pmatrix} \mathbf{0} \\ \frac{1}{K}\boldsymbol{\mu}_p \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \frac{1}{K}\, \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} \\ \nabla_{\boldsymbol{u}} \boldsymbol{G}_p|_{\boldsymbol{u}^\star_p} \end{pmatrix} = \mathbf{0},$$

$$\nabla_{\boldsymbol{u}}\Phi_{\mathrm{KMAy},\star}|_{\boldsymbol{u}_p^\star} + \begin{pmatrix} \mathbf{0} \\ \frac{1}{K}\boldsymbol{\mu}_p \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} \nabla_{\boldsymbol{u}}\boldsymbol{G}^{(1)}_{\mathrm{KMAy},\star}\Big|_{\boldsymbol{u}_p^\star} \\ \nabla_{\boldsymbol{u}}\boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}\Big|_{\boldsymbol{u}_p^\star} \end{pmatrix} = \mathbf{0},$$

$$\nabla_{\boldsymbol{u}}\Phi_{\mathrm{KMAy},\star}|_{\boldsymbol{u}_p^\star} + \boldsymbol{\mu}^{\mathsf{T}}\, \nabla_{\boldsymbol{u}}\boldsymbol{G}_{\mathrm{KMAy},\star}|_{\boldsymbol{u}_p^\star} = \mathbf{0}. \tag{3.31}$$

The three first equations above result from simple manipulations, while (3.2) and (3.8) are used for the three last equations.

With (3.28), a candidate value for $\boldsymbol{\mu}$ has therefore been exhibited, such that (3.27), (3.29), (3.30) and (3.31) hold simultaneously, i.e.:

$$\begin{aligned}
&\boldsymbol{G}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) \leq \mathbf{0}, \\
&\boldsymbol{\mu}^{\mathsf{T}}\boldsymbol{G}_{\mathrm{KMAy},\star}(\boldsymbol{u}_p^\star) = 0, \\
&\boldsymbol{\mu} \geq \mathbf{0}, \\
&\nabla_{\boldsymbol{u}}\Phi_{\mathrm{KMAy},\star}|_{\boldsymbol{u}_p^\star} + \boldsymbol{\mu}^{\mathsf{T}}\nabla_{\boldsymbol{u}}\boldsymbol{G}_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star} = \mathbf{0},
\end{aligned}$$

which shows that the 1st-order NCO of Problem $(3.2)|_\star$ hold at $\boldsymbol{u}_p^\star$.

2nd-order condition of optimality: Given that Problem $(3.2)|_\star$ is MFCQ at $\boldsymbol{u}_p^\star$ (Lemma 2), the 2nd-order NCO at $\boldsymbol{u}_p^\star$ is [39] the existence of KKT-multipliers $\boldsymbol{\mu} \in \mathbb{R}^{2n_g}$ such that:

$$\boldsymbol{N}^{\mathsf{T}}_{\mathrm{KMAy},\star}\nabla^2_{\boldsymbol{uu}}\mathcal{L}_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star}\boldsymbol{N}_{\mathrm{KMAy},\star} > \mathbf{0}, \tag{3.32}$$

where:

$$\mathcal{L}_{\mathrm{KMAy},\star}(\boldsymbol{u},\boldsymbol{\mu}) := \Phi_{\mathrm{KMAy},\star} + \sum_{i=1}^{2n_g} (\mu_i G_{i,\mathrm{KMAy},\star}) \tag{3.33}$$

is the Lagrangian of Problem $(3.2)|_\star$.

With the same candidate multipliers (3.28) (with which it has been proven that 1st-order NCO hold), (3.33) can be rewritten at $\boldsymbol{u}_p^\star$ as:

$$\mathcal{L}_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star} = \Phi_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star} + \sum_{i=1}^{n_g} \left(\frac{\mu_{i,p}}{K} G^{(2)}_{i,\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star}\right).$$

Applying the chain rule twice gives:

$$\nabla^2_{\boldsymbol{uu}}\Phi_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star} = \frac{1}{K^2}\,\nabla^2_{\boldsymbol{uu}}\Phi_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star},$$

Noticing that:

$$\begin{aligned}&\boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star} := \boldsymbol{G}_{\mathrm{MAy},\star},\\ \Rightarrow\ &\nabla^2_{\boldsymbol{uu}}\boldsymbol{G}^{(2)}_{\mathrm{KMAy},\star}\big|_{\boldsymbol{u}_p^\star} = \nabla^2_{\boldsymbol{uu}}\boldsymbol{G}_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star},\end{aligned}$$

enables to rewrite (3.32) as follows:

$$\frac{1}{K^2}\boldsymbol{H}_r\Phi_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star} + \sum_{i=1}^{n_g}\left(\frac{\mu_{i,p}}{K}\boldsymbol{H}_r G_{i,\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star}\right) > \boldsymbol{0},$$

where $\boldsymbol{H}_r(\cdot) := \boldsymbol{N}^{\mathsf{T}}_{\mathrm{KMAy},\star}\nabla^2_{\boldsymbol{uu}}(\cdot)\boldsymbol{N}_{\mathrm{KMAy},\star}$. Since the additional constraints does not affect the local geometrical properties of the problem (Lemma 1), the null-space of the active constraints are the same for Problem $(2.6)|_\star$ and $(3.2)|_\star$ ($\boldsymbol{N}_{\mathrm{KMAy},\star} = \boldsymbol{N}_{\mathrm{MAy},\star}$) and:

$$\boldsymbol{H}_r\Phi_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star} = \boldsymbol{N}^{\mathsf{T}}_{\mathrm{MAy},\star}\nabla^2_{\boldsymbol{uu}}\Phi_{\mathrm{MAy},\star}(\boldsymbol{u}))|_{\boldsymbol{u}_p^\star}\boldsymbol{N}_{\mathrm{MAy},\star}.$$

Multiplying both sides by $K^2 \neq 0$, yields:

$$\boldsymbol{H}_r\Phi_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star} + K\sum_{i=1}^{n_g}\left(\mu_{i,p}\boldsymbol{H}_r G_{i,\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star}\right) > \boldsymbol{0}.$$

This equation is of the form $\boldsymbol{A} + K\boldsymbol{B}$, where $\boldsymbol{A} = \boldsymbol{H}_r\Phi_{\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star}$, and $\boldsymbol{B} = \sum_{i=1}^{n_g}\left(\mu_{i,p}\boldsymbol{H}_r G_{i,\mathrm{MAy},\star}\big|_{\boldsymbol{u}_p^\star}\right)$, i.e. the weighted sum of the reduced Hessians of the modified model constraints. Given (3.21), i.e. $\boldsymbol{A} > \boldsymbol{0}$, Lemma 3 (b) can be called to state that $\exists K_{max}$ such that $\forall K \in (0, K_{\mathrm{opt}}]$ where $K_{opt} = \max\{1, K_{max}\}$, $\boldsymbol{A} + K\boldsymbol{B} > \boldsymbol{0}$. This shows that the 2nd-order NCO can be enforced with the appropriate choice of K, i.e. $K \in (0, K_{opt}]$.

In summary, (3.21) is a sufficient condition of model adequacy for KMAy since:

- the 1st-order conditions of optimality of KMAy are satisfied at $\boldsymbol{u}_p^\star$ with no further assumption,

- the 2nd-order NCO at $\boldsymbol{u}_p^\star$ can be enforced with the appropriate choice of K, i.e. $K \in (0, K_{opt}]$, provided (3.21) holds.

which concludes the prove. □

The model adequacy condition is obviously less restrictive with KMAy than with MAy. In particular, the positive definiteness of the reduced Hessian of the *cost function* is sufficient, while the positive definiteness of the reduced Hessian of the model *Lagrangian* would be required for MAy. Because the model adequacy condition takes the form of $\boldsymbol{A} + K\boldsymbol{B} > \boldsymbol{0}$, K can be seen as tuning parameter to enforce model adequacy, something that can be easily done if $\boldsymbol{A} > \boldsymbol{0}$. This is not the case with MAy since the filter gain does not influence the second-order conditions of optimality.

The focus was intentionally set here to $\boldsymbol{A} > \boldsymbol{0}$, as, in practice, the model cost function is often convex (non-convexity of the problem arising from the constraints), and if not, it can easily be approximated by a convex function [10]. This would *not* be sufficient for enforcing model adequacy for MA or MAy and constraints would have also to be convexified [10] to enforce positive definiteness of the Lagrangian, leading to poorer model-based predictions of the constraints.

Remark 8. *The model adequacy condition for KMAy can indeed be more generically formalized as the existence of a nonempty set of filter gains $\mathcal{K}_{tot}$, see* (3.19)*. Lemma 3 (a) shows that values of K such that $\boldsymbol{A} + K\boldsymbol{B} > \boldsymbol{0}$ might also exist fwhen neither $\boldsymbol{A}$ nor $\boldsymbol{B}$ are positive definite. In particular, if $\mathcal{K}_{tot} \cap (0, 1]$ is not empty, then the model can be made adequate for KMAy, irrespective of $\boldsymbol{A}$ being positive definite or not, by an appropriate choice of the filter gain, i.e. $K \in \mathcal{K}_{tot} \cap (0, 1]$. Still, the filter has to be appropriately chosen (e.g. $K < K_{opt}$ when $\boldsymbol{A} > \boldsymbol{0}$) and, without additional knowledge, it is recommended to use (potentially very) small filter gains to enforce this model adequacy condition. However, this would not be fully satisfactory since using small filters is detrimental to the convergence rate. In the next section, a novel real-time adaptation of the filter gain is introduced, to simultaneously maximize convergence speed and enforce model adequacy.*

Remark 9. *One might think that increasing the number of constraints is detrimental to the properties of RTO methods in general, and to KMAy in particular. However, the properties upon convergence of KMAy are only affected by the cost function and the active constraints (and nothing else) as*

seen from (3.21). *In fact, the larger the number of independent active constraints at the plant optimum, the smaller the null space* $\boldsymbol{N}_{\mathrm{MAy}}$*, and the higher the chances that* (3.21) *is satisfied. In other words, constraints that are active at* $\boldsymbol{u}_p^\star$ *are helpful for KMAy to converge to* $\boldsymbol{u}_p^\star$*.*

3.5. A Variant of KMAy

So far, focus was on KMAy as previously introduced in [37, 38]. However, minimizing the cost at the targeted point and enforcing feasibility at the applied point *only*, would have the same properties as KMAy upon convergence. In other words, let us define "KMAyB" as the simplified version of KMAy with the following model-based optimization problem:

$$\begin{aligned} \boldsymbol{u}_{k+1} := \arg\min_{\boldsymbol{u}} \quad & \Phi_{\mathrm{KMAy},k}(\boldsymbol{u}) := \Phi_{\mathrm{MAy},k}(\boldsymbol{v}_k(\boldsymbol{u})) \\ \text{s.t. } & \boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u}) := \boldsymbol{G}_{\mathrm{MAy},k}(\boldsymbol{u}) \leq \boldsymbol{0}, \end{aligned} \tag{3.34}$$

instead of (3.2). It is easy to see that KMAy-B has the same properties than KMAy when the filter satisfies Lemma 1 condition. Indeed, the Theorems 1, 2, and 3 are valid when $\boldsymbol{\mu}^{(1)} = 0$ and $\boldsymbol{\mu}^{(2)} = \boldsymbol{\mu}_p/K$. In other words, when $\boldsymbol{G}^{(1)}_{\mathrm{KMAy},k}(\boldsymbol{u})$ of (3.2) plays virtually no role. There are two differences between KMAy et KMAy-B, that are:

- The constraints $\boldsymbol{G}^{(1)}_{KMAy,k}(\boldsymbol{u})$ of (3.2) do not need to be evaluated when KMAy-B is used. This is not really a computational advantage over KMAy since it is anyway necessary to evaluate the steady-state model at two points $\{\boldsymbol{u}, \boldsymbol{v}_k(\boldsymbol{u})\}$.
- The trajectory from $\boldsymbol{u}_0$ to $\boldsymbol{u}_\infty$ are not the same, which is illustrated hereafter.

Example 3. *Consider again the problem of Example 2, initialized this time at* $\boldsymbol{u}_0 = [0.1, 0.9]$ *with a filter* $\boldsymbol{K} = 0.5\boldsymbol{I}$*. Figure 7 depicts the iterates with KMAy and KMAy-B. One can observe that the first step of KMAy-B activates the constraint and does no belong to the segment* $[\boldsymbol{u}_0, \boldsymbol{u}_p^\star]$*. This is because the point targeted by KMAy-B at the first RTO iteration is model-based infeasible. Nevertheless, both methods converge to the plant optimum.*

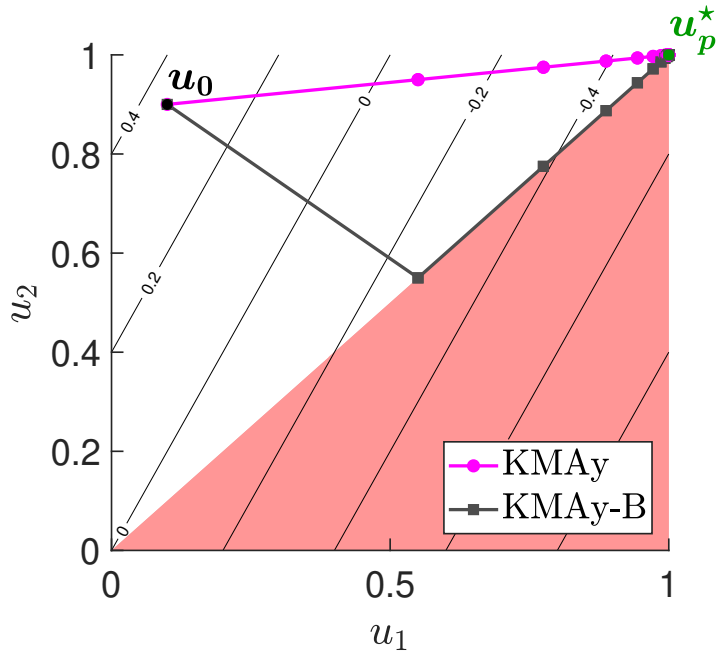


Figure 7: **Example 3**: KMAy and KMAy-B differ w.r.t. the trajectories they generate.

Generally speaking, it is the authors' opinion that KMAy should be preferred to KMAy-B for the three following reasons:

(i) KMAy is expected to be safer. As illustrated with Example 3 and Figure 7, KMAy-B is likely target a point which is predicted to be infeasible by the model, while the constraint enforces the implementation of an input whereby model feasibility is predicted. In many occasions, KMAy-B will have the tendency to "jump" to the constraint. On the other hand, the presence of the constraint on the targeted inputs with KMAy will typically lead to what is illustrated on Figure 7, i.e. KMAy targets model-feasible inputs, and looks for an input to implement on the segment that joins the previous operating conditions to the targeted inputs, which is likely to lie in the interior of the model feasible domain, and therefore further from the modelled constraints. Whenever the model underestimates the constraints KMAy has obviously more chances to lead to a feasible plant iterate. This safer behavior of KMAy is preferable from a practical implementation viewpoint and is more likely to be trusted by engineers and operators than an approach leading to jumps to constraints, as it might as well be easier to implement and supervise.

(ii) Figure 1 shows that while the trajectory with KMAy lies strictly in-

side the feasible domain of the model, KMAy-B jumps to and then slides along the constraint, increasing at each iteration the risks of plant infeasibility. But Figure 1 depicts a case whereby the constraint does not "move" between iterations. In practice, due to the presence of the modifier terms, the prediction of the constraints by the model is corrected at each iteration, and the constraint can "move". If, e.g., the update of the modifiers lead to a shift of the constraint on Figure 7 towards the bottom right, KMAy-B will, for the next iteration, jump again to the constraint, while KMAy will (again) target a model feasible point and choose the next inputs inside the feasible domain of the model, again further from the adapted constraint than KMAy-B. In other words, this tendency of KMAy-B to jump to the constraints can persist over successive iterations.

(iii) Finally, it can be argued that with KMAy-B, the filter does not play its most fundamental role. Initially, filters have been introduced in the MA framework to provide asymptotic stability and *smooth the plant manipulations*, i.e. exponential convergence to the plant optimum *(and to the corresponding active constraints)*[4]. As discussed before and illustrated with Example 3, this latter role is not really fulfilled as KMAy-B can have the tendency to jump on the constraint, which is in contradiction with the main motivation behind filtering[5].

4. Automatic tuning of the filter gain in KMA schemes

4.1. Automatic Selection of the Filter Gain

So far, the choice of the filter gain $\boldsymbol{K}$ is left to the implementation stage and is mainly guided by engineering insight. Once this choice is made, to the best of the authors' knowledge, the only way to update and improve $\boldsymbol{K}$ can be found in [14, 15] that requires the knowledge of the plant Lipschitz constants, with the additional asset to ensure plant feasibility at all iterates.

[4]This has never been stated this way, since constraints satisfaction have always been enforced at the targeted point. But, approaching the constraints with caution has always been a motivation for using filters.

[5]On the other hand, KMAy converges exponentially to the targeted point *and to the constraint*. This would always be the case, unless converging to the plant optimal inputs implies getting around a concave constraint as for Figure 1, where KMAy and KMAy-B would both jump to the constraint. Therefore, there is still room for improvements of KMAy, e.g. adding backoffs to the constraints and exponentially reduce them.

But Lipschitz constants are unfortunately rarely available, even when they can be shown to exist. In the following subsection, a filter gain selector that does not require additional information about the plant is presented. This approach enforces the satisfaction of the model adequacy condition (3.21).

The following filter gain selector is proposed:

$$K_k := \max \left\{ \arg\min_{K \in \mathcal{K}} \left(||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2 \right) \right\}, \tag{4.1}$$

K_k is thus taken at each k as the largest filter gain, which minimizes the distance $||\boldsymbol{u}_k - \boldsymbol{v}_k(\boldsymbol{u}_{k+1})||_2$ over a set $\mathcal{K}$ to be defined by the user. $\mathcal{K} := [K^{\mathrm{L}}, K^{\mathrm{U}}]$ must be chosen as a subset of $(0, 1]$, where $K^{\mathrm{L}} > 0$ and $K^{\mathrm{U}} \leq 1$ are the lower and upper bounds on K_k, respectively. The next theorem proves that selection of K_k ensures that (3.21) is still sufficient for the model to be adequate.

Theorem 4. *Consider the KMAy optimization Problem* (3.2) *and the associated KMAy algorithm, with a filter* $\boldsymbol{K}_k = K_k \boldsymbol{I}_{n_u}$ *satisfying the structure suggested in Lemma 1 and* K_k *updated using* (4.1)*. If* (3.21) *is satisfied, i.e.,*

$$\boldsymbol{N}_{MAy,\star}^{\mathsf{T}} \nabla_{\boldsymbol{u}\boldsymbol{u}}^2 \Phi_{\mathrm{MAy},\star}(\boldsymbol{u}))|_{\boldsymbol{u}_p^\star} \boldsymbol{N}_{MAy,\star} > 0,$$

then $\exists K_{opt} > 0$ *such that* $\forall K_k \in (0, K_{opt}]$ *the model is adequate (Theorem 3) and the four following statements are true:*

(1) If $K^{\mathrm{L}} \leq K_{opt}$ *and* $\boldsymbol{u}_k = \boldsymbol{u}_p^\star$*, then the gain selector* (4.1) *selects* $K_k = K_{opt}$*.*

(2) The gain selector (4.1) *does not affect the KKT-matching property upon convergence of KMAy (Theorem 1);*

(3) K^{L} *guarantees no infinitely small steps;*

(4) K^{U} *let the user define the maximal filter allowed to both (potentially) guarantee asymptotic stability and smooth the manipulations of the plant.*

Proof.

Model Adequacy and Statement (1):

The distance $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2$ is an indirect indication of the satisfaction of the condition of optimality at $\boldsymbol{u}_k$. Indeed, when this distance is zero,

$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k = \boldsymbol{u}_\infty$, which means that ad-KMAy has converged with $\boldsymbol{u}_\infty$ satisfying the 1st- and 2nd-order optimality conditions of Problem $(3.2)|_\infty$. From Theorem 1, it is known that, if KMAy converges, it is to a KKT-point of the plant. Also, if (3.21) holds, then Theorem 3 states that $\boldsymbol{u}_p^\star$ is a fixed point of KMAy $\forall K \in (0, K_{opt}]$. Combining these two theorems and introducing the bound K^{L}, it follows that:

- If $K \in [K^{\mathrm{L}}, K_{opt}]$ and $\boldsymbol{u}_k = \boldsymbol{u}_p^\star$, then $\boldsymbol{u}_{k+1} = \boldsymbol{u}_p^\star = \boldsymbol{u}_\infty$, and $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2 = 0$.
- If $K > K_{opt}$ and $\boldsymbol{u}_k = \boldsymbol{u}_p^\star$, then $\boldsymbol{u}_{k+1} \neq \boldsymbol{u}_p^\star$, $\boldsymbol{u}_p^\star \neq \boldsymbol{u}_\infty$, and $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2 > 0$.

Therefore, if $[K^{\mathrm{L}}, K_{opt}] \neq \emptyset$, i.e. if $K^{\mathrm{L}} \leq K_{opt}$, then the largest filter gain such that the distance $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2$ is minimized is K_{opt}. So, the filter gain selector (4.1) will select K_{opt} and model adequacy is guaranteed.

Statement (2): Assume that KMAy with the adaptive gain of (4.1) converges. Because Theorem 1 is independent of the filter gain value, if convergence occurs, it is at the plant optimum.

Statement (3): For some $\boldsymbol{u} \in \mathcal{U} \backslash \boldsymbol{u}_p^\star$ it is possible that the distance $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2$ decreases with K_k. Such cases are expected when a concave constraint is activated, see e.g. Figure 8. For such cases, (4.1) selects K^{L} instead of any other value closer to 0. This is the main motivation behind the introduction of K^{L}.

Statement (4): Equation (4.1) is designed to ensure that the largest acceptable value of K_k enabling (3.21) to remain a sufficient for model adequacy is taken, while preventing the selection of too small values for the filter gain. However, it does not enforce asymptotic stability nor does it ensure smooth manipulations of the plant. To do so, the upper-bound K^{U} is introduced. Indeed, if K^{U} is small enough it is expected that the asymptotic stability condition for MAy of [5] is enforced. But it is not possible to check this a priori, since it requires unavailable knowledge about the plant. Therefore, the choice of K^{U} is left to the practitioners, and ultimately K^{U} would have to be reduced to enforce asymptotic stability. Yet, equation (4.1) would still ensure that the largest possible value of the gain is taken. □

Remark 10. *Statement (1) of Theorem 4 can be extended to the cases where* (3.21) *does not hold. In such cases, from Lemma 3 and Remark 8 there can be some values of* $K \in \mathcal{K}_{tot}$ *such that* $\boldsymbol{u}_p^\star$ *satisfies the 2nd-order optimality*

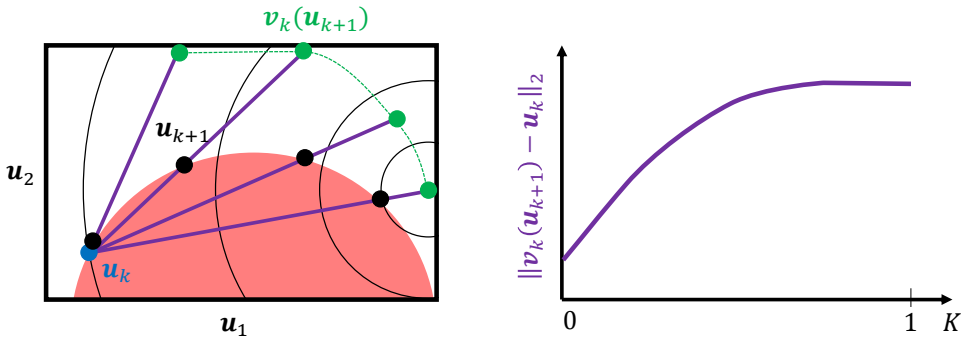


Figure 8: Conceptual example of a case where selecting the filter gain without lower-bound K^{L} would lead to infinitely small steps $[\boldsymbol{u}_k, \boldsymbol{u}_{k+1}]$. On the LHS the optimization problem is depicted with a shaded area for the constraints and contour curves for the cost function. On the RHS, the distance $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2$ is plotted w.r.t. the value of the filter gain K.

conditions of Problem $(3.2)|_\star$*, i.e. whenever* $\boldsymbol{A} + K\boldsymbol{B}$ *can be forced to be positive definite, with no assumption about the positive definiteness of* $\boldsymbol{A}$*. If such values exist, then*

- *If* $K \in [K^{\mathrm{L}}, K^{\mathrm{U}}] \cap \mathcal{K}_{tot}$ *and* $\boldsymbol{u}_k = \boldsymbol{u}_p^\star$*, then* $\boldsymbol{u}_{k+1} = \boldsymbol{u}_p^\star = \boldsymbol{u}_\infty$*, and* $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2 = 0$.
- *If* $K \notin [K^{\mathrm{L}}, K^{\mathrm{U}}] \cap \mathcal{K}_{tot}$ *and* $\boldsymbol{u}_k = \boldsymbol{u}_p^\star$*, then* $\boldsymbol{u}_{k+1} \neq \boldsymbol{u}_p^\star$*,* $\boldsymbol{u}_p^\star \neq \boldsymbol{u}_\infty$*, and* $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2 > 0$.

If $[K^{\mathrm{L}}, K^{\mathrm{U}}] \cap \mathcal{K}_{tot} \neq \emptyset$*, then* (4.1) *will select again the largest filter gain that minimizes the distance* $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k||_2$*, which is obviously* $\max\{[K^{\mathrm{L}}, K^{\mathrm{U}}] \cap \mathcal{K}_{tot}\}$*, that also provides model adequacy.*

Theorem 4 provides a general method for the selection of the filter relying only on available knowledge. Thus, the following model-based optimization problem with adaptive filter is suggested as an alternative to Problem (3.2) is proposed:

$$
\begin{aligned}
\boldsymbol{u}_{k+1} := \ & \arg\min_{\boldsymbol{u}} \quad \Phi_{\mathrm{KMAy},k}(\boldsymbol{u}) \\
& \text{s.t. } \boldsymbol{G}_{\mathrm{KMAy},k}(\boldsymbol{u}) \leq \boldsymbol{0}, \\
& \quad K_k := \max\left\{ \arg\min_{K \in \mathcal{K}} \left(||\boldsymbol{v}_k(\boldsymbol{u}) - \boldsymbol{u}_k||_2 \right) \right\}, \\
& \quad \boldsymbol{v}_k(\boldsymbol{u}) = K_k^{-1}(\boldsymbol{u} - \boldsymbol{u}_k) + \boldsymbol{u}_k.
\end{aligned}
\tag{4.2}
$$

The associated RTO algorithm is:

Output Modifier Adaptation with adaptive filter-based constraint (ad-KMAy)

Initialization. Provide $\boldsymbol{u}_0$. Choose $K^{\mathrm{L}} \in (0, 1]$, $K^{\mathrm{U}} \in (0, 1]$ with $K^{\mathrm{L}} \leq K^{\mathrm{U}}$.
for $k = 0 \rightarrow \infty$

1. Follow the steps (1) to (3) of KMAy.
2. Compute $\boldsymbol{u}_{k+1}$ by solving Problem (4.2).

end

It can be noticed that Problem (4.2) is a bi-level optimization problem that can be computationally challenging to solve. Nevertheless, it is possible to solve this optimization problem in parallel, for several fixed values of the filter gain, e.g., $K_k = \{0.3, 0.4, \ldots, 0.8\}$. Then, the $\boldsymbol{u}_{k+1}$ returned by the optimization problem for which $||\boldsymbol{v}_k(\boldsymbol{u}_{k+1}) - \boldsymbol{u}_k)||_2$ is minimal will be selected. Of course, doing so would return only an approximation of the true solution of Problem (4.2), but this would not be detrimental to the properties listed in Theorem 4, although the selected value K_k will not exactly be the largest admissible K.

As discussed before, if (3.21) does not hold, then a relatively easy way to enforce model adequacy for (ad-)KMAy is to replace the model of the cost function by an approximated model that is convex at $\boldsymbol{u}_k$ for each RTO iteration. The next section explains how this can be automatically performed.

4.2. Enforcing Model Adequacy

To enforce model adequacy, what is suggested in [10] can be followed, and a convex approximation of the model can be substituted to the model at hand. But because the model adequacy condition can be reduced to (3.21), *only the cost function needs to be convexified*. Therefore, less of the model information is lost, especially w.r.t. the constraints. In the case of ad-KMAy, the modifiers perform affine corrections on the model outputs $\boldsymbol{y}$ that indirectly induce affine *and high-order* corrections of the model cost and constraints functions. These high-order corrections affecting the model Hessians, it is preferable to convexify the cost model at each RTO iteration. Strictly speaking, with

(ad-)KMAy, the model approximation has to be positive definite only at $\boldsymbol{u}_p^\star$. This is also the case for [10], where the cost and constraints of the model are convexified everywhere just to enforce local convexity at the unknown $\boldsymbol{u}_p^\star$. We suggest here to implicitly enforce local convexity at the current point, i.e at $\boldsymbol{u}_k$, $\forall k$. This forces convexity to also hold upon convergence, and thus at the unknown $\boldsymbol{u}_p^\star$. This is done by adding a quadratic penalty term δ_k to the modified cost function, similarly to [16]:

$$\Phi^c_{\mathrm{KMAy},k}(\boldsymbol{u}) := \Phi_{\mathrm{KMAy},k}(\boldsymbol{u}) + \frac{\delta_k}{2}(\boldsymbol{u}-\boldsymbol{u}_k)^\mathsf{T}(\boldsymbol{u}-\boldsymbol{u}_k), \tag{4.3}$$

where

$$\delta_k := \max\left\{0, \sigma(-\nabla^2_{\boldsymbol{uu}}\Phi_{\mathrm{KMAy},k}(\boldsymbol{u})\big|_{\boldsymbol{u}_k}) + \epsilon\right\}, \tag{4.4}$$

with $\sigma(.)$ being a function returning the largest eigenvalue of the matrix (.), $\epsilon \in \mathbb{R}^{+*}$ a strictly positive number, and $\Phi^c_{\mathrm{KMAy},k}$ a locally convex approximation of $\Phi_{\mathrm{KMAy},k}$ at $\boldsymbol{u}_k$. Then, the cost function of Problem (4.2) is replaced by (4.3). If the cost function is already locally convex around $\boldsymbol{u}_k$, the penalty term δ_k is set to 0 and no convexification is performed, while otherwise, the penalty term makes it convex at $\boldsymbol{u}_k$.

Remark 11. *This paper is mostly focused on MAy and extensions, but similar improvements - inherited from the filter-based additional constraints - can also be obtained with standard MA, DMA [31], or with any other MA method, since all the Theorems and Lemmas introduced in this paper can indeed be easily adapted to most if not all MA configurations.*

5. Illustrative Example

5.1. The Williams-Otto Reactor

The standard benchmark case study for RTO considered here is the continuous stirred-tank reactor of [40], where the three following reactions take place (for the plant):

$$\mathrm{A} + \mathrm{B} \xrightarrow{k_{p,1}} \mathrm{C}, \qquad k_{p,1} = A_{p,1}e^{-B_{p,1}/(T_R+273.15)} \tag{5.1}$$

$$\mathrm{C} + \mathrm{B} \xrightarrow{k_{p,2}} \mathrm{P} + \mathrm{E}, \qquad k_{p,2} = A_{p,2}e^{-B_{p,2}/(T_R+273.15)} \tag{5.2}$$

$$\mathrm{P} + \mathrm{C} \xrightarrow{k_{p,3}} \mathrm{G}. \qquad k_{p,3} = A_{p,3}e^{-B_{p,3}/(T_R+273.15)}, \tag{5.3}$$

The reactants A and B are fed separately, with mass flowrates of F_A and F_B, respectively. P and E are the desired products, C is an intermediate product and G is an undesired by-product. The reactor is operated isothermally at a controlled temperature T_R. Steady-state mass balances can be found in [41]. The same optimization problem as in [16] is considered, wherein the input variables are $\boldsymbol{u} = [F_A, F_B, T_R]^\mathsf{T}$ and the outputs are $\boldsymbol{y}_p = [X_E, X_P, X_G]^\mathsf{T}$ with X_i denoting the concentration of species i.

There is significant structural plant-model mismatch since the model *only considers two reactions* [8, 16]. The objective is to maximize profit at steady-state, while satisfying an upper bound on X_G and input bounds:

$$\begin{aligned}
\max_{\boldsymbol{u}} \quad & \phi(\boldsymbol{u}, \boldsymbol{y}_p) = (1143.38X_P + 25.92X_E)(F_A + F_B) - \\
& \qquad \alpha F_A - \beta F_B && (5.4)\\
\text{s.t.} \quad & g(\boldsymbol{y}_p) = X_G - 0.08 \leq 0, && (5.5)\\
& F_A \in [3,\ 4.5]\ \ (\text{kg/s}), \qquad F_B \in [6,\ 11]\ \ (\text{kg/s}),\\
& T_R \in [80,\ 105]\ \ (^\circ\text{C}),
\end{aligned}$$

where α and β are model parameters that simulate market fluctuations. The scenario considers three successive different operation modes, which are detailed in Table 1.

Table 1: Scenario

Mode	α	β	RTO Iterations
1	76.23	114.34	0 to 6
2	68.61	102.91	7 to 12
3	77.75	114.34	13 to 24

Also, to illustrate the effects of increasing the number of constraints, 3 cases are analyzed.

Case 1: The classical benchmark case study presented above. In that case, the model is inadequate for MA and MAy when the values α and β for Modes 1 or 3 [16, 37]. However, the cost function always satisfies condition (3.21) for KMAy. Therefore, the model can be made adequate by an appropriate choice of K.

Case 2: An additional constraint on X_A is added, i.e. (5.5) is replaced

by

$$\boldsymbol{g}(\boldsymbol{y}_p) = \begin{pmatrix} X_G - 0.08 \\ X_A - 0.18 \end{pmatrix} \leq \mathbf{0},$$

which are the constraint used in [27], and that are *active* at the plant optimum of each Mode.

Case 3: An additional constraint on the ratio X_C/X_G is added, i.e. (5.5) is replaced by

$$\boldsymbol{g}(\boldsymbol{y}_p) = \begin{pmatrix} X_G - 0.08 \\ X_C/X_G - 0.5 \end{pmatrix} \leq \mathbf{0},$$

which is made up to illustrate effects of *inactive* constraints at the plant optimum on the analyzed RTO methods.

KMAy and ad-KMAy are implemented, both initialized at the same point $\boldsymbol{u}_0 = [3.5, 9.5, 92]^\mathsf{T}$, which is not any of the plant optimal inputs of Modes 1, 2 and 3. Both algorithms must first reach the optimal inputs of Mode 1, and then track the changes of the plant optimal inputs when the scenario switches from Mode 1 to 2 and later from Mode 2 to 3. Of note is that no information about the current mode and/or corresponding plant optimal inputs is available at any time during the simulations. Modes and plant optimal inputs (in green) are only displayed for illustration purposes. KMAy uses input filter matrices of the form $\boldsymbol{K} = K\boldsymbol{I}_{n_u}$ with two different values for $K = \{0.6,\ 0.7\}$.

KMAy and ad-KMAy are compared on Figures 9 to 11 for the cases 1 to 3, respectively.

Case 1: Figure 9 shows that, in any case, gain adaptation leads to, at least, as faster convergence as the corresponding fixed-gain approach. Also, it can be observed that during Mode 1, the adaptive filter does not converge to the expected optimal filters gains, which can be identified as belonging to the interval $[0.6, 0.7]$, since KMAy converges for $K = 0.6$, and does not for $K = 0.7$. Instead, it seems that the filter is set at its lower bound. A longer simulation of Mode 1 has been performed and the results are depicted on Figure 12. This simulation clearly shows that the adaptive gain oscillates around the expected best filter value. These oscillations are due to the asymptotic convergence to $\boldsymbol{u}_p^\star$ and limited to its close neighborhood. They are most likely due to numerical errors. Indeed, the values of K_k must be

discussed in light of the size of the corresponding input moves $[\boldsymbol{u}_k, \boldsymbol{u}_{k+1}]$, as illustrated with Figure 12(c).

What this simulation shows is that the exact K_{opt} is not likely to be met in practice or in simulation, and that it is more a theoretical "ideal" filter just like $\boldsymbol{u}_p^\star$ is the "ideal" operating point. Still, the filter gain selector we proposed does what it has been designed for and drives the plant to its optimum within an error of $\approx 10^{-5}$, see Figure 12(d). Note that the same observation can be made for Mode 3 results.

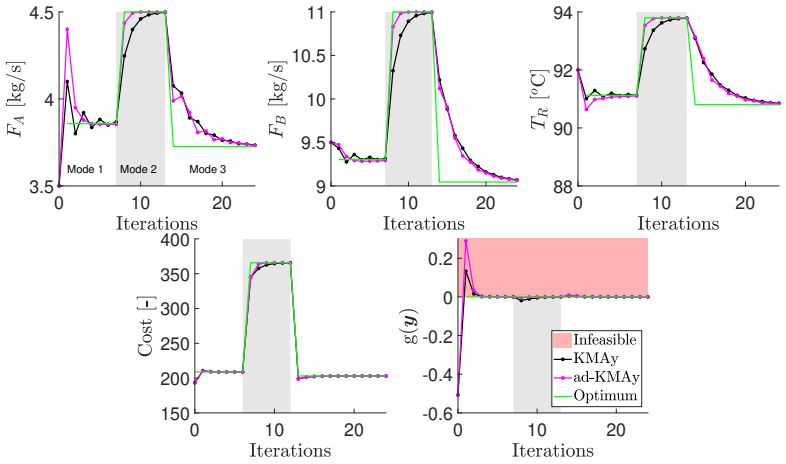
Case 2: Figure 10 shows that the additional constraint is always active at the plant optimum and enables convergence for both KMAy and ad-KMAy, even for high filter gain such as 0.9. This illustrates Remark 9, where it is highlighted that increasing the number of active constraints at the plant optimum helps relaxing the model adequacy condition.

Case 3: Figure 11 shows that the additional constraint is not active at the plant optimum and does not affect the convergence properties of (ad-) KMAy. Indeed, the inputs, cost, and constraint on X_G follows the same path as for Case 1.

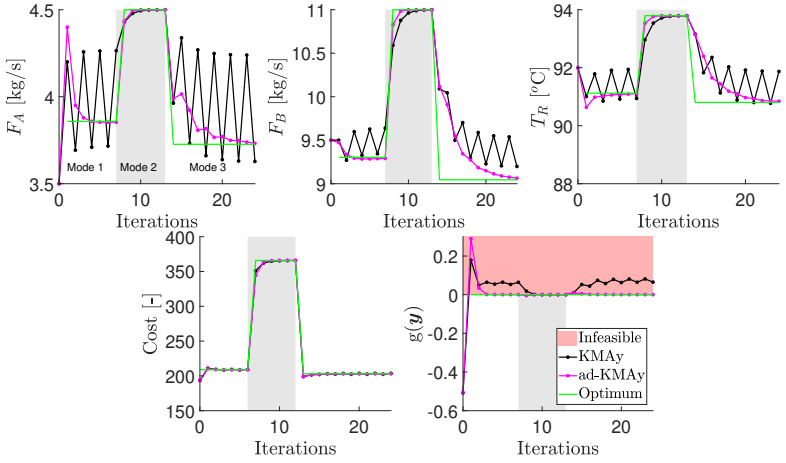
6. Conclusions

Significant improvements of MAy have been introduced, analyzed and successfully applied in simulation. Not only do the two new algorithms (ad-)KMAy of this article use filtering as a tool to (i) provide asymptotic stability and (ii) smooth manipulations of the plant, but also to (iii) relax the model adequacy criterion and (iv) increase the safety of MA methods. This is made possible by the introduction of filter-based additional constraints, which is proven for both KMAy and ad-KMAy, to preserve the ability of MA approaches to converge only at the true plant optimal inputs, even in the presence of structural plant-model mismatch, a likely situation whereby the standard two-step approaches fail.

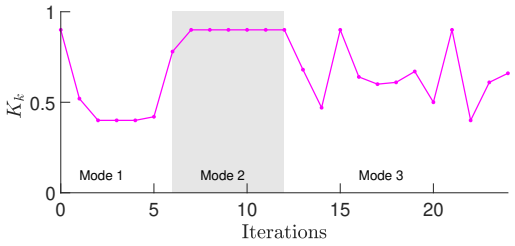
New arguments in favor of filtering the inputs instead of the modifiers are provided, as well as a method to perform an automatic selection and adaptation of the filter gain. KMAy and its variant only require additional filter-based constraints. While this could be expected to endanger constraint qualification, it has been shown that the method proposed to duplicate the constraints and explicitly incorporate the input filter stage into the optimization problem, ensures that MFCQ holds if LICQ holds for the plant problem, a standard assumption that would anyway be made (as it ensures that the



(a) $\boldsymbol{K} = 0.6\boldsymbol{I}_{n_u}$

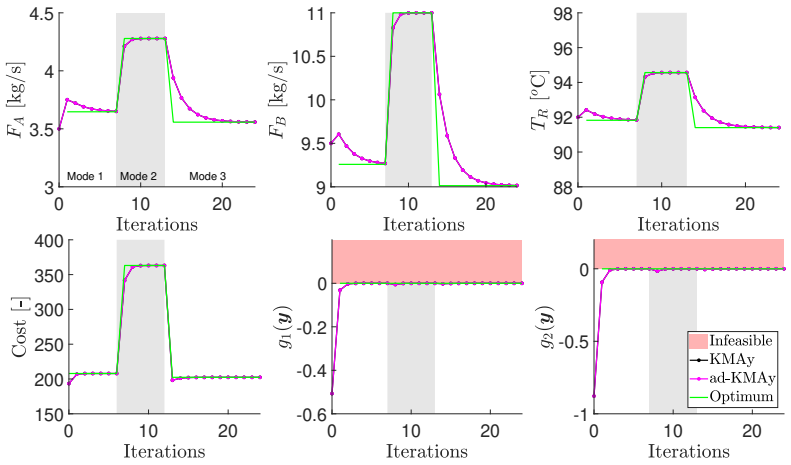


(b) $\boldsymbol{K} = 0.7\boldsymbol{I}_{n_u}$

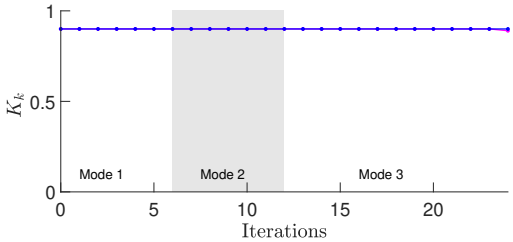


(c) Value of the adaptive filter gain K_k.

Figure 9: **Case 1:** Simulation results comparing ad-KMAy to KMAy with several filter gains.

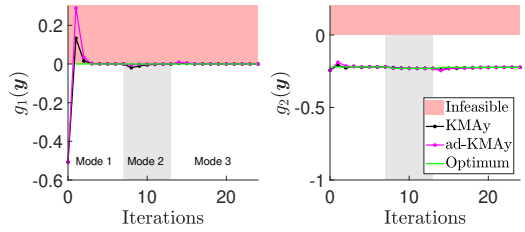


(a) $\boldsymbol{K} = 0.9\boldsymbol{I}_{n_u}$

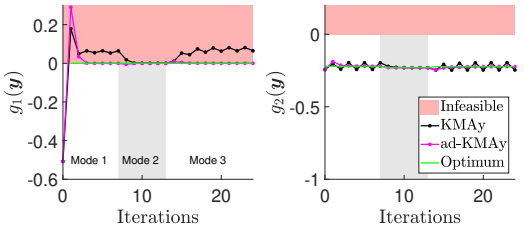


(b) Value of the adaptive filter gain K_k.

Figure 10: **Case 2:** Simulation results comparing ad-KMAy to KMAy.



(a) $\boldsymbol{K} = 0.6\boldsymbol{I}_{n_u}$



(b) $\boldsymbol{K} = 0.7\boldsymbol{I}_{n_u}$

Figure 11: **Case 3:** Simulation results comparing ad-KMAy to KMAy with several filter gains. For the inputs, cost, and adaptive filter values, the simulation results being the as for case 1, they are not displayed.

KKT conditions are necessary conditions of optimality). Overall, the proposed algorithms provide net improvements to the existing methods and are easy to implement and to tune.

Future research will focus, as mentioned in Section 3.3 Remark 4, the addition of constraints as a way to improve safety of the corresponding RTO-MA scheme, with the aim of limiting any potential increase of conservatism. Since the filter is integrated now into the KMAy problem formulation by means of additional constraints, an idea would be to investigate its implementation at the level of the cost function. With the emergence of methods like Gaussian-Process MA [27], some weight is given to the exploration of the input space and corresponding data collection, which becomes part of the objective function, which of course still mainly aims at improving the operation. With the framework of [27], the "exploration" point $\boldsymbol{u}_{k+1}$ is selected with the only objective to be feasible for the model, while the targeted point $\boldsymbol{u}^\star_{k+1}$ is the expected optimum. Adding a value to the exploration objective, such that the point $\boldsymbol{u}_{k+1}$ is also selected to improve the confidence on the next predictions of $\boldsymbol{u}^\star_{k+1}$ is a research direction we envisage.

References

[1] A. R. Conn, K. Scheinberg, L. N. Vicente, Introduction to Derivative-Free Optimization, Cambridge University Press, 2009.

[2] G. E. P. Box, N. R. Draper, Evolutionary Operation. A Statistical Method for Process Improvement, John Wiley, New York, 1969.

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

[4] W. Gao, S. Engell, Iterative set-point optimization of batch chromatography, Computers & Chemical Engineering 29 (2005) 1401–1409.

[5] A. Marchetti, B. Chachuat, D. Bonvin, Modifier-adaptation methodology for real-time optimization, Industrial & engineering chemistry research 48 (2009) 6022–6033.

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

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

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

[9] A. Papasavvas, T. de Avila Ferreira, A. Marchetti, D. Bonvin, Analysis of output modifier adaptation for real-time optimization, Computers & Chemical Engineering 121 (2019) 285–293.

[10] G. François, D. Bonvin, Use of convex model approximations for real-time optimization via modifier adaptation, Industrial & Engineering Chemistry Research 52 (2013) 11614–11625.

[11] W. Gao, S. Wenzel, S. Engell, A reliable modifier-adaptation strategy for real-time optimization, Computers & chemical engineering 91 (2016) 318–328.

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

[13] A. Ahmad, W. Gao, S. Engell, A study of model adaptation in iterative real-time optimization of processes with uncertainties, Computers & Chemical Engineering 122 (2019) 218–227.

[14] G. A. Bunin, G. François, D. Bonvin, Sufficient conditions for feasibility and optimality of real-time optimization schemes - I. Theoretical foundations, ArXiv:1308.2620 (2013).

[15] G. A. Bunin, G. François, D. Bonvin, Sufficient conditions for feasibility and optimality of real-time optimization schemes - II. Implementation issues, ArXiv:1308.2625 (2013).

[16] A. G. Marchetti, T. Faulwasser, D. Bonvin, A feasible-side globally convergent modifier-adaptation scheme, Journal of Process Control 54 (2017) 38–46.

[17] A. R. Conn, N. I. Gould, P. L. Toint, Trust region methods, volume 1, Siam, 2000.

[18] G. A. Bunin, On the equivalence between the modifier-adaptation and trust-region frameworks, Computers & Chemical Engineering 71 (2014) 154–157.

[19] M. Jonin, M. Singhal, S. Diwale, C. N. Jones, D. Bonvin, Active directional modifier adaptation with trust region-application to energy-harvesting kites, in: 2018 European Control Conference (ECC), IEEE, 2018, pp. 2312–2317.

[20] E. del Rio Chanona, J. A. Graciano, E. Bradford, B. Chachuat, Modifier-adaptation schemes employing gaussian processes and trust regions for real-time optimization, IFAC-PapersOnLine 52 (2019) 52–57.

[21] G. François, D. Bonvin, Use of transient measurements for the optimization of steady-state performance via modifier adaptation, Industrial & Engineering Chemistry Research 53 (2013) 5148–5159.

[22] T. de Avila Ferreira, G. François, A. G. Marchetti, D. Bonvin, Use of transient measurements for static real-time optimization, IFAC-PapersOnLine 50 (2017) 5737–5742.

[23] T. Rodríguez-Blanco, D. Sarabia, J. L. Pitarch, C. De Prada, Modifier adaptation methodology based on transient and static measurements for rto to cope with structural uncertainty, Computers & chemical engineering 106 (2017) 480–500.

[24] D. Krishnamoorthy, B. Foss, S. Skogestad, Steady-state real-time optimization using transient measurements, Computers & Chemical Engineering 115 (2018) 34–45.

[25] D. Navia, L. Briceño, G. Gutiérrez, C. De Prada, Modifier-adaptation methodology for real-time optimization reformulated as a nested optimization problem, Industrial & Engineering Chemistry Research 54 (2015) 12054–12071.

[26] T. Rodríguez-Blanco, D. Sarabia, C. de Prada, Efficient nested modifier-adaptation methodology for dealing with process constraints in real-time optimization, Computers & Chemical Engineering 122 (2019) 372–382.

[27] T. de Avila Ferreira, H. A. Shukla, T. Faulwasser, C. N. Jones, D. Bonvin, Real-time optimization of uncertain process systems via modifier adaptation and gaussian processes, in: 2018 European Control Conference (ECC), IEEE, 2018, pp. 465–470.

[28] J. Matias, J. Jäschke, Using a neural network for estimating plant gradients in real-time optimization with modifier adaptation, IFAC-PapersOnLine 52 (2019) 808–813.

[29] D. H. Jeong, J. M. Lee, Enhancement of modifier adaptation scheme via feedforward decision maker using historical disturbance data and deep machine learning, Computers & Chemical Engineering 108 (2018) 31–46.

[30] D. Navia, A. Puen, P. Quintanilla, L. Briceño, L. Bergh, On dealing with measured disturbances in the modifier adaptation method for real-time optimization, Computers & Chemical Engineering (2019).

[31] S. Costello, G. François, D. Bonvin, A directional modifier-adaptation algorithm for real-time optimization, Journal of Process Control 39 (2016) 64–76.

[32] M. Singhal, A. G. Marchetti, T. Faulwasser, D. Bonvin, Active directional modifier adaptation for real-time optimization, Computers & Chemical Engineering 115 (2018) 246–261.

[33] R. Schneider, P. Milosavljevic, D. Bonvin, Distributed modifier-adaptation schemes for the real-time optimisation of uncertain interconnected systems, International Journal of Control 92 (2019) 1123–1136.

[34] P. Milosavljevic, R. Schneider, T. Faulwasser, D. Bonvin, Distributed modifier adaptation using a coordinator and measured interconnection variables, IFAC-PapersOnLine 50 (2017) 5743–5748.

[35] J. O. A. Matias, J. Jäschke, Online model maintenance via output modifier adaptation, Industrial & Engineering Chemistry Research (2019).

[36] A. Papasavvas, G. Francois, Internal modifier adaptation for the optimization of large-scale plants with inaccurate models., Industrial & Engineering Chemistry Research (2019).

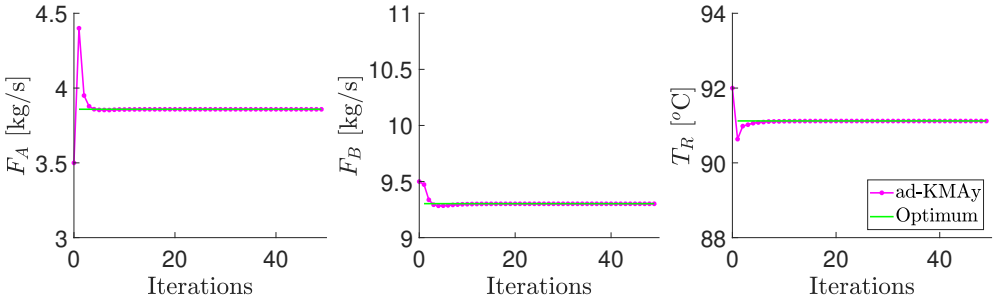
[37] A. Papasavvas, G. Francois, Filter-based additional constraints to relax the model adequacy conditions in modifier adaptation, IFAC-PapersOnLine 52 (2019) 58–63.

[38] A. Papasavvas, G. Francois, Filter-based constraints to easier plant feasibility in modifier adaptation schemes, Computer Aided Chemical Engineering 46 (2019) 1267–1272.

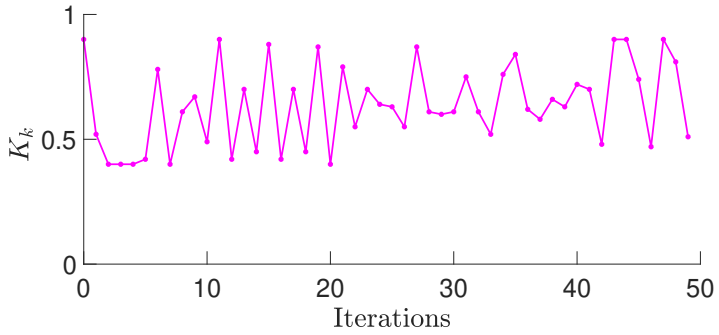
[39] E. Chong, Z. Zak, An Introduction to Optimization, 2 ed., John Wiley and Sons, Inc, 2001.

[40] T. J. Williams, R. E. Otto, A generalized chemical processing model for the investigation of computer control, AIEE Trans. 79 (1960) 458.

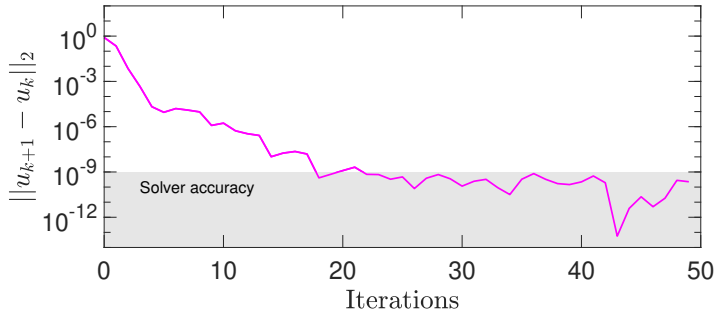
[41] Y. Zhang, J. F. Forbes, Extended design cost: a performance criterion for real-time optimization systems, Computers & Chemical Engineering 24 (2000) 1829–1841.



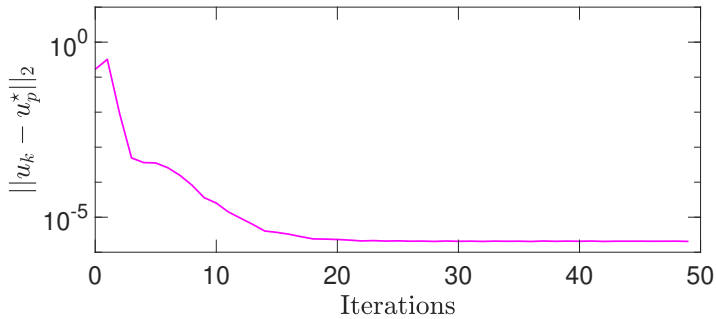
(a) ad-KMAy vs ad-KMAy-B: inputs vs iteration number.



(b) Adaptive filter gain K_k.



(c) Step size (y axis is in log scale).



(d) Distance to optimum (y axis is in log scale).

Figure 12: Case 1: Analysis of ad-KMAy if Mode 1 lasted longer.