Automatica 155 (2023) 111110

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Auto-tuning of reference models in direct data-driven control☆

Daniele Masti [a], Valentina Breschi [b,*], Simone Formentin [c], Alberto Bemporad [a]

[a] IMT School for Advanced Studies Lucca, Piazza San Francesco 19, Lucca, Italy
[b] Department of Electrical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, Netherlands
[c] Dipartimento di Elettronica, Bioingegneria e Informazione, Politecnico di Milano, Via Ponzio 34/5, Milano, Italy

ARTICLE INFO

Article history:
Received 12 October 2021
Received in revised form 21 January 2023
Accepted 3 May 2023
Available online 12 June 2023

Keywords:
Data-based control
Model-reference control
Autotuning
Direct data-driven control

ABSTRACT

Designing controllers directly from data often requires choosing a reference closed-loop model, whose behavior should be reproduced as tightly as possible by the actual closed-loop system via the selected controller structure (e.g., PID). Within a linear setting, we present a derivative-based approach to jointly select the reference model and controller parameters directly from data. The proposed strategy allows one to maximize closed-loop performance while enforcing user-defined constraints, and it is designed to handle non-minimum phase dynamics. The effectiveness of the proposed approach is shown through three numerical case studies.

1. Introduction

In recent years, Data-Driven Control (DDC) has rapidly become a popular model-free control design method. Compared to classical data-enabled strategies (Schoukens & Ljung, 2019), *direct* DDC does not require a model identification phase prior to control design, thus bypassing a stage that is often time-consuming and mainly targeted to achieve the best simulation/prediction accuracy, rather than closed-loop performance. Several approaches exist to directly design controllers from data, ranging from recent advances in data enabled predictive control (see, *e.g.*, Berberich, Köhler, Müller, and Allgöwer (2021)), to more consolidated strategies, such as Virtual Reference Feedback Tuning (VRFT) (Campi, Lecchini, & Savaresi, 2002) and Iterative Feedback Tuning (IFT) (Hjalmarsson, Gevers, Gunnarsson, & Lequin, 1998) . The *main tuning knob* of these last class of procedures is the reference model that dictates the desired closed-loop response of the system. A key question is therefore how to choose such a reference model, keeping into account not only the desired closed-loop performance, but also that the latter should be achieved by a given controller structure. Recently, techniques have been proposed to automatize the choice of the reference model, while concurrently designing a controller from data, that can be roughly distinguished into one-stage and two-stage strategies. In Selvi, Piga, Battistelli, and Bemporad (2021), Selvi, Piga, and Bemporad (2018), the reference model is tuned by optimizing a performance-oriented cost via Particle Swarm Optimization (PSO) (Rahmat-Samii, Gies, & Robinson, 2003), while the VRFT method is concurrently exploited to design the controller corresponding to each explored reference model. Instead, a combination of Bayesian Optimization (BO) (Brochu, Cora, & De Freitas, 2010) and the VRFT approach is used in Breschi and Formentin (2021) to optimize closed-loop performance by input/output data only. This method has been recently extended to the Linear Parameter Varying (LPV) setting in van Meer, Breschi, Oomen, and Formentin (2021), by replacing the VRFT method with the data-driven strategy introduced in Formentin, Piga, Tóth, and Savaresi (2016). By leaving the zeros of the reference model free to accommodate the presence of unstable zeros in the plant, the two-stage strategies proposed in Campestrini, Eckhard, Gevers, and Bazanella (2011), Lecchini and Gevers (2002) are instead explicitly tailored to handle non-minimum phase systems. In the first stage, unstable zeros are indirectly detected, then the reference model is corrected accordingly to ultimately allow for the design of the controller by either the VRFT approach or IFT. Note that, these approaches rely on the assumption that the poles of the reference model are fixed. An alternative method has been proposed in Cerone, Abuabiah, and Regruto (2020), which relies on the formulation of a fictitious, H_∞ direct control problem to tune the reference model for both minimum-phase and nonminimum-phase plants. For the multi-input multi-output setting, Gonçalves da Silva, Bazanella, and Campestrini (2019) introduce a method for the selection of the reference model, which allows for more freedom

☆ This paper was partially supported by the Italian Ministry of University and Research under the PRIN'17 project "Data-driven learning of constrained control systems", contract no. 2017J89ARP. The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Juan I. Yuz under the direction of Editor Alessandro Chiuso.

* Corresponding author.
E-mail addresses: daniele.masti@imtlucca.it (D. Masti), v.breschi@tue.nl (V. Breschi), simone.formentin@polimi.it (S. Formentin), alberto.bemporad@imtlucca.it (A. Bemporad).

https://doi.org/10.1016/j.automatica.2023.111110
0005-1098/

in the choice of the controller and reference model poles. Within a different setting, namely the Loewner framework, approaches for the automatic selection of reference models have been presented in Kergus, Olivi, Poussot-Vassal, and Demourant (2019a, 2019b). Similar problems could be found also in adaptive control, see, e.g., Lee, Anderson, Mareels, and Kosut (1995).

This paper presents an alternative, deterministic, gradient-based strategy to jointly select a reference model and a linear controller from data that: (*i*) can deal with non-minimum phase plants, without requiring any prior information or choice of the reference model parameters, (*ii*) accounts for both the desired closed-loop performance and the capability of the linear controller to achieve it, and (*iii*) is able to handle constraints that shape the desired closed-loop behavior.

The paper is organized as follows. We formally introduce the setting in Section 2 , while Sections 3–4 present possible design choices to shape the cost and the constraints characterizing the tuning problem. The proposed strategy is presented in Section 5, while its performance on three benchmark simulation examples is discussed in Section 7. The paper is ended by some concluding remarks.

2. Problem statement

Consider a *single-input single-output* (SISO), *linear time invariant* (LTI) discrete-time dynamical system $\mathcal{P}$, whose behavior is governed by a set of *unknown* linear difference equations. Let $u_k \in \mathbb{R}$ be the input , possibly corrupted by an additive zero-mean white noise, fed to the plant at time $k \in \mathbb{N}$ and $y_k \in \mathbb{R}$ be the corresponding noisy output, namely

$$y_k = y_k^o + v_k, \tag{1}$$

where $v_k \in \mathbb{R}$ is a zero-mean additive white noise.

Assume a dataset $\mathcal{D}_N = \{u_k, y_k\}_{k=1}^N$ of input/output samples has been collected. We want to design a controller for $\mathcal{P}$ to attain a target stable closed-loop behavior described by the following LTI *reference model*

$$\mathcal{M}_{\theta_M}: \ \tilde{y}_k = \sum_{i=1}^{n_{a_M}} a_i^M \tilde{y}_{k-i} + \sum_{j=1}^{n_{b_M}} b_j^M r_{k-j}, \tag{2a}$$

where $\tilde{y}_k$ is the desired response to a user-defined reference r_k, for $k \in \mathbb{N}$. The reference model is thus characterized by the parameter vector

$$\theta_M = \begin{bmatrix} a_1^M & a_2^M & \cdots & a_{n_{a_M}}^M & b_1^M & \cdots & b_{n_{b_M}}^M \end{bmatrix}' \in \mathbb{R}^{n_M}, \tag{2b}$$

of fixed dimension $n_M = n_{a_M} + n_{b_M}$. To attain such a target behavior $\mathcal{M}_{\theta_M}$, we exploit the fixed-order, dynamic, error-feedback controller $\mathcal{C}_{\theta_C}$:

$$\mathcal{C}_{\theta_C}: \ u_k = \sum_{i=1}^{n_{a_C}} a_i^C u_{k-i} + \sum_{j=0}^{n_{b_C}} b_j^C e_{k-j}, \tag{3a}$$

where $e_k = r_k - y_k$ is the tracking error, with $e_k \in \mathbb{R}$, and

$$\theta_C = \begin{bmatrix} a_1^C & \cdots & a_{n_{a_C}}^C & b_0^C & b_1^C & \cdots & b_{n_{b_C}} \end{bmatrix}' \in \mathbb{R}^{n_C}, \tag{3b}$$

is the vector of parameters characterizing $\mathcal{C}_{\theta_C}$, n_{a_C}, n_{b_C} are *fixed* a priori and $n_C = n_{a_C} + n_{b_C} + 1$.

Many techniques exist to synthesize $\mathcal{C}_{\theta_C}$ given the reference model $\mathcal{M}_{\theta_M}$. Choosing $\mathcal{M}_{\theta_M}$, however, is usually far from trivial, especially when little to no prior knowledge on $\mathcal{P}$ is available. More importantly, whether $\mathcal{M}_{\theta_M}$ can be attained when closing the loop on $\mathcal{P}$ with $\mathcal{C}_{\theta_C}$ is typically unpredictable. To address this issue, we propose to tune $\mathcal{M}_{\theta_M}$ and $\mathcal{C}_{\theta_C}$ *simultaneously*, treating both θ_C and θ_M as degrees of freedom. The goal is to find a trade-off between two competing objectives: (*i*) $\mathcal{M}_{\theta_M}$ must reflect a satisfactory closed-loop tracking response to an extensive range of set-points, and (*ii*) $\mathcal{C}_{\theta_C}$ must be able to make the closed-loop system behave like θ_M. We also seek to satisfy a set of constraints on the desired response to user-defined references.

We formalize the above objectives and constraints in the following optimization problem

$$\begin{aligned} \min_{\theta_M, \theta_C} \ & \ell_{fit}(\theta_C, \theta_M) + \gamma \ell_{perf}(\theta_M) \\ \text{s.t.} \ & g(\theta_M) \leq \mathbf{0}_\eta, \end{aligned} \tag{4}$$

where $\ell_{perf} : \mathbb{R}^{n_M} \to \mathbb{R}$ and $\ell_{fit} : \mathbb{R}^{n_C} \to \mathbb{R}$ penalize, respectively, the performance of the reference model and the capability of the associated data-driven controller $\mathcal{C}_{\theta_C}$ to realize the target behavior, $\gamma > 0$ is a tunable relative importance weight, and $g : \mathbb{R}^{n_M} \to \mathbb{R}^\eta$, $\eta \geq 0$, is exploited to enforce user-defined closed-loop response specifications.

3. Characterization of the reference model

To formulate $\ell_{perf}(\theta_M)$, we quantify the closed-loop tracking performance over a simulation window of $\nu \in \mathbb{N}$ steps, with $\nu \geq 1$. Let

$$\bar{R}_\nu = \begin{bmatrix} \bar{r}_0 & \bar{r}_1 & \cdots & \bar{r}_\nu \end{bmatrix}' \in \mathbb{R}^\nu, \tag{5a}$$

$$\tilde{Y}_\nu(\theta_M) = \begin{bmatrix} \tilde{y}_0(\theta_M) & \tilde{y}_1(\theta_M) & \cdots & \tilde{y}_\nu(\theta_M) \end{bmatrix}' \in \mathbb{R}^\nu. \tag{5b}$$

stack some prefixed reference values[1], and the corresponding desired outputs according to (2a) respectively. We set

$$\ell_{perf}(\theta_M) = \frac{1}{2\nu} \|\tilde{Y}_\nu(\theta_M) - \bar{R}_\nu\|_2^2. \tag{6}$$

Clearly $\ell_{perf}(\theta_M)$ is heavily influenced by the chosen reference sequence $\bar{R}_\nu$, which should be sufficiently varied to provide an extensive overview of the reference-model performance. Possible choices in this direction are pseudo-random binary Refs. Selvi et al. (2018) and sine-sweep set points. Based on the set points of interest, the constraint in (4) is shaped by the desired closed-loop behavior. As an example, the desired response to unit steps can be constrained by setting $\eta = \nu$ and imposing $g(\theta_M)$ in (4) to satisfy the following:

$$g_i(\theta_M) = \begin{cases} \tilde{y}_i(\theta_M) - (1 + \bar{S}), & \text{if } i \in \{0, 1, \ldots, \kappa_s\}, \\ |\tilde{y}_i(\theta_M) - 1| - \delta, & \text{if } i \in \{\kappa_s + 1, \ldots, \nu\}, \end{cases} \tag{7}$$

where $\bar{S} \in [0, 1]$ is the upper bound on admissible overshoots and δ the maximum acceptable tracking error after κ_s settling samples, with $\kappa_s = t_s/T_s$, where t_s [s] is the chosen settling time and T_s [s] is the sampling time of the system. Defining $g(\theta_M)$ as in (7) bounds the steady-state tracking error, while imposing the settling time of the desired closed-loop step response. At the same time, it (implicitly) enforces a constraint on the DC gain of the reference model and discourages closed-loop instability. In addition, we may want to explicitly match a desired closed-loop DC gain G_∞, e.g., $G_\infty = 1$, which is easily accomplished by adding in (4) the linear equality constraint

$$G_\infty \left(1 - \sum_{i=1}^{n_{a_M}} a_i^M \right) = \sum_{j=1}^{n_{b_M}} b_j^M. \tag{8}$$

[1] We stress that $\bar{r}$ might not be the reference r that the user aims at tracking, since it is exploited for learning purposes only.

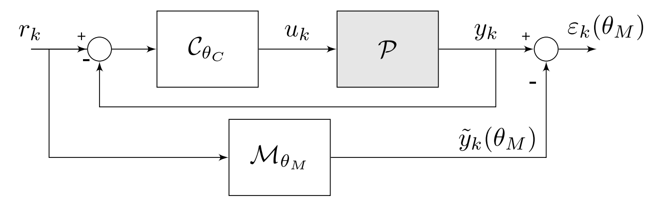


Fig. 1. Model-matching scheme, with the mismatch between the desired and attained closed-loop behavior $\varepsilon_k(\theta_M) = y_k - \tilde{y}_k(\theta_M)$.

4. Assessing the model-matching quality

It is well-known that poor performance (up to closed-loop instability) are likely to result from the choice of excessively demanding reference models (Nijmeijer & Savaresi, 1998). At the same time, tuning the reference model should not require closed-loop experiments, not to hamper the safety of $\mathcal{P}$. We therefore need to define $\ell_{fit}(\theta_C, \theta_M)$ in (4) to penalize the mismatch between the desired and attained closed-loop behavior, without actually closing the loop during the tuning phase.

To this end, we rely on the matching scheme in Fig. 1, according to which the larger the error $\varepsilon_k(\theta_M) = y_k - \tilde{y}_k(\theta_M)$ is , the less the desired behavior is attained in closed-loop. In order to use the available data $\mathcal{D}_N$ to judge model matching, we select the *fictitious reference* $\mathcal{R}_N^\star(\theta_M) = \{r_k^\star(\theta_M)\}_{k=1}^N$ as a function of θ_M by solving the smooth problem

$$\begin{aligned} \mathcal{R}_N^\star(\theta_M) = \ & \underset{\mathcal{R}_N}{\operatorname{argmin}} \ \ \ell_M(\mathcal{R}_N(\theta_M)) \\ & \text{s.t. } \tilde{y}_k(\theta_M) = X_k\theta_M, \ \ k = n_M+1, \ldots, N, \end{aligned} \tag{9}$$

where $X_k = \begin{bmatrix} y_{k-1} & \cdots & y_{k-n_{a_M}} & r_{k-1} & \cdots & r_{k-n_{b_M}} \end{bmatrix}$, the loss is

$$\ell_M(\mathcal{R}_N(\theta_M)) = \frac{1}{2}\sum_{k=n_M+1}^{N}\left[\|y_k - \tilde{y}_k(\theta_M)\|_2^2 + \frac{\rho_{\mathcal{R}}}{2}\|\Delta r_k\|_2^2\right], \tag{10}$$

where $\Delta r_k = r_k - r_{k-1}$, and $\rho_{\mathcal{R}} > 0$ is a tunable regularization parameter, penalizing changes in the virtual reference. Since the matching performance depends on the designed controller, the fictitious (or virtual) reference sequence $\mathcal{R}_N^\star(\theta_M)$ is then used to design (3a). Moreover, as problem (9) depends on the measured outputs in $\mathcal{D}_N$, the controller has to be designed so that the inputs fed to $\mathcal{P}$ fit the sequence $\{u_k\}_{k=1}^N$ in $\mathcal{D}_N$. To attain a tight match between the achieved and desired behaviors, we thus choose the controller by solving

$$\begin{aligned} \theta_c^\star(\mathcal{R}_N^\star(\theta_M)) = \ & \operatorname{argmin}_{\theta_C} \ \ \ell_{fit}(\theta_c, \theta_M) \\ & \text{s.t. } \hat{u}_k(\theta_C) = \mathcal{X}_k(\mathcal{R}_N^\star(\theta_M))\theta_C, \ \ k \in \mathcal{I}_N^c, \end{aligned} \tag{11a}$$

where $\mathcal{I}_N^c = \{n_c+1, \ldots, N\}$, the cost function is

$$\ell_{fit}(\theta_c, \theta_M) = \frac{1}{2}\sum_{k=n_C+1}^{N}\left[\|u_k - \hat{u}_k(\theta_C)\|_2^2 + \frac{\rho}{2}\|\theta_C\|_2^2\right], \tag{11b}$$

where $\rho > 0$ is a tunable regularization parameter, and

$$\mathcal{X}_k(\mathcal{R}_N^\star(\theta_M)) = \begin{bmatrix} u_{k-1} & \cdots & u_{k-n_{a_C}} & e_k^\star(\theta_M) & \cdots & e_{k-n_{b_C}}^\star(\theta_M) \end{bmatrix},$$

is the (fictitious) controller regressor constructed based on the outcome of (9) , with $e_k^\star(\theta_M) = r_k^\star(\theta_M) - y_k$ being the fictitious tracking error at k. Note that, $\mathcal{R}_N^\star(\theta_M)$ corresponds to the set-point obtained by directly inverting the reference model as in Formentin et al. (2016), whenever $\rho_{\mathcal{R}} = 0$ and this inverse exists and it is well defined.

Remark 1 (*Constraining the Structure of the Controller*). The problem in (4) can be easily extended to include additional constraints on the controller structure. As in standard VRFT, all the poles of $\mathcal{C}_{\theta_C}$ can be constrained by setting

$$[\theta_C]_i = [\bar{\theta}_C]_i, \quad i = 1, \ldots, n_{a_C},$$

where $\bar{\theta}_C \in \mathbb{R}^{n_{a_C}}$ is a vector stacking the fixed coefficients of the controller. In addition, an integrator can be incorporated into the controller by modifying its dynamics as:

$$\underbrace{u_k - u_{k-1}}_{\Delta u_k} = \sum_{i=1}^{n_{a_C}-1} a_i^C\,\Delta u_{k-i} + \sum_{j=0}^{n_{b_C}} b_j^C\, e_{k-j}, \tag{12}$$

with $n_C = n_{a_C} + n_{b_C}$. The loss $\ell_{fit}(\theta_C, \theta_M)$ should thus be recast as:

$$\ell_{fit}(\theta_c, \theta_M) = \frac{1}{2}\sum_{k=n_C+1}^{N}\left[\|\Delta u_k - \widehat{\Delta u}_k(\theta_C)\|_2^2 + \frac{\rho}{2}\|\theta_C\|_2^2\right], \tag{13}$$

with

$$\widehat{\Delta u}_k(\theta_C) = \mathcal{X}_k(\mathcal{R}_N^\star(\theta_M))\theta_C, \tag{14}$$

and $\mathcal{X}_k(\mathcal{R}_N^\star(\theta_M))$ being the following vector:

$$\begin{bmatrix} \Delta u_{k-1} & \cdots & \Delta u_{k-n_{a_C}-1} & e_k^\star(\theta_M) & \cdots & e_{k-n_{b_C}}^\star(\theta_M) \end{bmatrix}.$$

Alternatively, by considering (3), one can impose the following equality in the back-shift operator q^{-1}:

$$(1 - q^{-1})\left(1 - \sum_{i=1}^{n_{a_C}-1} a_i^C q^{-i}\right) = 1 - \sum_{i=1}^{n_{a_C}} [\theta_C]_i q^{-i},$$

by enforcing

$$1 - \sum_{i=1}^{n_{a_C}} [\theta_C]_i = 0. \tag{15}$$

Algorithm 1 Sensitivity-based tuning procedure

Input: dataset $\mathcal{D}_N$; initial candidate reference model $\theta_M^{(0)}$, fictitious set point $(\mathcal{R}_N^v)^{(0)}$ and controller $\theta_C^{(0)}$.

1. **for** $i = 0, 1, \ldots$ **do**
   - 1.1 **compute** $\mathcal{R}_N^\star(\theta_M^{(i)})$ as in (9);
   - 1.2 **compute** $\partial_{\theta_M}\mathcal{R}_N^\star(\theta_M^{(i)})$ as in Section 5.1;
   - 1.3 **find** $\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)}))$ by solving (11);
   - 1.4 **compute** $\ell_{fit}(\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)}), \theta_M^{(i)})$;
   - 1.5 **compute** $\partial_{\theta_M}\ell_{fit}(\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)})))$ as in Section 5.1;
   - 1.6 compute $\partial_{\theta_M}\ell_{perf}(\theta_M^{(i)})$ and $\partial_{\theta_M} g(\theta_M^{(i)})$;
   - 1.7 **find** a new candidate $\theta_M^{(i+1)}$ through any gradient-based approach;
2. **until termination**;

Output: reference model $\theta_M^\star$, associated fictitious set point $\mathcal{R}_N^\star$ and controller $\theta_C^\star$.

5. Sensitivity-based reference model tuning

Due to the constraints in (9), (11), the solution of problem (4) under the losses $\ell_{perf}(\theta_M)$ in (6) and $\ell_{fit}(\theta_c, \mathcal{R}_N^\star(\theta_M))$ in (11) entails tackling a multilevel optimization problem. Specifically, one needs to consider three hierarchically nested optimization layers: (i) the inner one, devoted to the inversion of the reference model and, thus, the computation of the virtual reference sequence $\mathcal{R}_N^\star$ according to (9), (*ii*) the central layer, dedicated to the controller tuning phase via the solution of (11), and lastly (*iii*) the third layer, which is devoted to reference model tuning. In this work, we propose to handle this three layered problem via derivative based-approaches, by heavily relying on the *sensitivity theorem* (Luenberger & Ye, 2016). Hence, the denomination of the approach as "sensitivity-based".

Starting from an initial candidate $\theta_M^{(0)}$, the proposed sensitivity-based strategy is summarized in Algorithm 1. At the ith iteration, we find the fictitious reference by solving (9) for the current candidate $\theta_M = \theta_M^{(i)}$ (Step 1.1). We then compute the associated sensitivity

$$\partial_{\theta_M}\mathcal{R}_N^\star(\theta_M^{(i)}) = \left.\frac{\partial \mathcal{R}_N^\star(\theta_M)}{\partial \theta_M}\right|_{\theta_M=\theta_M^{(i)}}$$

and design the controller $\theta_c^\star(\mathcal{R}_N^\star(\theta_M^{(i)}))$ associated to $\theta_M^{(i)}$ as in (11) (Steps 1.2–1.3). Based on the designed controller, we compute $\ell_{fit}(\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)})), \theta_M^{(i)})$ (see Step 1.4) and, then, at Step 1.5, we rely on the sensitivity theorem (Luenberger & Ye, 2016, Chapter 11.7) and on the previously computed sensitivity to retrieve

$$\partial_{\theta_M}\ell_{fit}(\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)}))) = \left.\frac{\partial \ell_{fit}(\theta_c, \theta_M)}{\partial \theta_M}\right|_{\substack{\theta_C=\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)}))\\ \theta_M=\theta_M^{(i)}}}. \quad (16)$$

We then compute $\ell_{perf}(\theta_M^{(i)})$ and its derivative

$$\partial_{\theta_M}\ell_{perf}(\theta_M^{(i)}) = \left.\frac{\partial \ell_{perf}(\theta_M)}{\partial \theta_M}\right|_{\theta_M=\theta_M^{(i)}},$$

along with $g(\theta_M^{(i)})$ and

$$\partial_{\theta_M}g(\theta_M^{(i)}) = \left.\frac{\partial g(\theta_M)}{\partial \theta_M}\right|_{\theta_M=\theta_M^{(i)}},$$

at Step 1.6. Once the cost function, the constraints and their derivatives are constructed as previously outlined, the next candidate model $\theta_M^{(i+1)}$ can be obtained via any gradient-based optimization strategy (see Step 1.7).

We point out that the necessary conditions to apply the sensitivity theorem in computing (16) are rather mild, as the former requires the satisfaction of second-order optimality conditions in the neighborhood of $\theta_C^\star(\mathcal{R}_N^\star(\theta_M^{(i)}))$ only. Moreover, as assembling the cost and computing its derivative require the solution of two successive and tightly related optimization problems (one to find the fictitious reference and the other one to find the associated controller), we remark that the use of a solver supporting warm-starting is advised.

Remark 2 (*On the Quality Of* θ_C)**.** For a fixed reference model, the design problem in (11) knowingly results in a biased controller (see Lecchini, Campi, and Savaresi (2001)). To overcome this problem, we could resort to an instrumental variable scheme, as conventionally done in model-reference direct control design (see *e.g.,* Lecchini et al. (2001)). This can be done by setting $\rho = 0$ in (11), introducing the instrument and filtering the inputs and virtual tracking errors as discussed in Formentin, Campi, Carè, and Savaresi (2019), since the problem solved in (11) coincides with the one tackled with the VRFT approach. At the same time, following this procedure to replace the cost in (11) increases the complexity of the sensitivity-based procedure, as the instruments depend on the chosen reference model and, thus, they have to be updated at each iteration of Algorithm 1. Not to increase the complexity of the sensitivity-based procedure, while designing an unbiased controller, θ_C can be refined once θ_M has been fixed, by employing an instrumental variable scheme.

5.1. Derivatives computation

Suppose that the fictitious reference $\mathcal{R}_N^\star(\theta_M^{(i)})$ is retrieved by solving (9) via an algorithm based on fixed-point iterations, such as gradient descent, until convergence. Let $z(\mathcal{R}_N(\theta_M))$ be a given function of $\mathcal{R}_N(\theta_M)$, that characterizes the chosen fixed-point iteration. Then, the following holds:

$$\frac{\partial \mathcal{R}_N^\star(\theta_M)}{\partial \theta_M} = \left(I - \left.\frac{\partial z(\mathcal{R}_N(\theta_M))}{\partial \mathcal{R}_N(\theta_M)}\right|_{\mathcal{R}_N=\mathcal{R}_N^\star}\right)^{-1}\frac{\partial z(\mathcal{R}_N^\star(\theta_M))}{\partial \theta_M}, \quad (17)$$

as a consequence of the results presented in Jeon, Lee, and Choi (2020), with I being an identity matrix of appropriate dimensions.

Example 1. Assume problem (9) is solved by gradient descent, hence $z(\mathcal{R}_N(\theta_M))$ is given by:

$$z(\mathcal{R}_N(\theta_M)) = \mathcal{R}_N(\theta_M) - \alpha\nabla_{\mathcal{R}_N(\theta_M)}\ell_M(\mathcal{R}_N(\theta_M)),$$

where α is the step size of the gradient descent updates, and $\nabla_{\mathcal{R}_N(\theta_M)}\ell_M(\mathcal{R}_N(\theta_M))$ is the gradient of $\ell_M(\mathcal{R}_N(\theta_M))$ with respect to $\mathcal{R}_N(\theta_M)$. According to (17), it follows that

$$\frac{\partial \mathcal{R}_N^\star(\theta_M)}{\partial \theta_M} = -\left(\mathcal{H}_{\theta_M}\right)^{-1}\nabla_{\mathcal{R}_N,\theta_M}\ell_M(\mathcal{R}_N^\star(\theta_M)),$$

where $\mathcal{H}_{\theta_M} = \nabla^2_{\mathcal{R}_N(\theta_M)}\ell_M(\mathcal{R}_N^\star(\theta_M))$ is the Hessian of the cost in (9) when considering the fictitious reference $\mathcal{R}_N^\star(\theta_M)$, computed at the fixed point, and

$$\nabla_{\mathcal{R}_N,\theta_M}\ell_M(\mathcal{R}_N^\star(\theta_M)) = \left.\frac{\partial^2 \ell_M(\mathcal{R}_N(\theta_M))}{\partial \mathcal{R}_N(\theta_M)\partial \theta_M}\right|_{\mathcal{R}_N=\mathcal{R}_N^\star}.$$

To compute the sensitivity in (16) we can instead rely on the following decomposition:

$$\frac{\partial \ell_{fit}(\theta_C, \theta_M)}{\partial \theta_M} = \frac{\partial \ell_{fit}(\theta_C, \theta_M)}{\partial \mathcal{R}_N^\star(\theta_M)}\frac{\partial \mathcal{R}_N^\star(\theta_M)}{\partial \theta_M}, \quad (18)$$

which stems from the dependence of the design loss in (11) on $\mathcal{R}_N^\star(\theta_M)$. While the second partial derivative on the right-hand-side of (18) can be readily obtained from (17), we have still to retrieve $\frac{\partial \ell_{fit}(\theta_C, \theta_M)}{\partial \mathcal{R}_N^\star(\theta_M)}$. As previously mentioned, the latter can be computed by relying on the sensitivity theorem (Luenberger & Ye, 2016, Chapter 11.7).

5.2. Dealing with non-minimum phase systems

When the plant $\mathcal{P}$ is non-minimum phase, additional care must be taken in tuning the reference model. Indeed, the controller in (3a) should not cancel out the unstable (and unknown) zeros of $\mathcal{P}$, as they must be shared by the plant and the reference model. If additional insights on $\mathcal{P}$ or non-minimum phase behaviors can be detected from the available data $\mathcal{D}_N$, constraints can be readily incorporated within the general tuning problem (4), so as to enforce the presence of unstable zeros in the reference model. Otherwise, we can introduce additional constraints in (4) to avoid critical cancellations between the controller and the unknown plant. In turn, this operation is expected to implicitly enforce matching between the plant and reference model zeros.

Consider the following polynomial in the back-shift operator q^{-1}:

$$D(q^{-1}, \theta_C) = 1 - \sum_{i=1}^{n_{a_C}}[\theta_C]_i q^{-i}, \quad (19a)$$

that characterizes the denominator of the controller to be designed according to (3a). Let us now decompose it into a product of monomials and binomials in q^{-1} as follows:

$$D(q^{-1},\theta_C)=\prod_{m=1}^{\mu}(1+\tau_m q^{-1})\prod_{h=1}^{\beta}(1+\varphi_h q^{-1}+\phi_h q^{-2}), \tag{19b}$$

where $\tau_m \in \mathbb{R}$, for all $m = 1,\dots,\mu$ and $\varphi_h, \phi_h \in \mathbb{R}$, for $h = 1,\dots,\beta$, with $\mu + 2\beta = n_{a_C}$. Thanks to the simplicity of the reformulation in (19b), we can apply the Jury criterion (Ogata, 1994) to enforce the stability of the controller through the linear inequality constraints

$$|\tau_m| < 1,\ m = 1,\dots,\mu, \tag{20a}$$

$$1+\varphi_h+\phi_h > 0,\ h = 1,\dots,\beta, \tag{20b}$$

$$1-\varphi_h+\phi_h > 0,\ h = 1,\dots,\beta, \tag{20c}$$

$$|\phi_h| < 1,\ h = 1,\dots,\beta. \tag{20d}$$

While the decomposition in (19b) simplifies the enforcement of controller stability, it introduces a set of nonlinearities with respect to the controller parameters. A solution to preserve the linearity in the controller structure, while enforcing stability , is to add the nonlinear equality constraint

$$\prod_{m=1}^{\mu}(1+\tau_m q^{-1})\prod_{h=1}^{\beta}(1+\varphi_h q^{-1}+\phi_h q^{-2})=1-\sum_{i=1}^{n_{a_C}}[\theta_C]_i q^{-i}, \tag{21}$$

that imposes the correspondence between the two decompositions of $D(q^{-1},\theta_C)$, respectively given by (19a) and (19b). Overall, to enforce the controller stability we thus have to impose the $2\mu+4\beta$ linear inequalities in (19) and the nonlinear equality in (21). Note that the nonlinear constraint in (21) can be handled efficiently by selecting a feasible initial guess of the auxiliary parameters in (19b).

Remark 3. When some poles of the controller are fixed beforehand, the constraints in (20) can be imposed on the subset of free poles, by decomposing only the polynomial depending on them. Note that, with the parameterization with integral action in (12), one can decompose the entire denominator of the controller. Indeed, the pole on the unit circle is "hidden" by resorting to input variations, and it will only be made explicit once the controller will have to be deployed.

6. Practical hints for numerical complexity reduction

In this section, we provide a set of practical hints to reduce the computational complexity of the tuning procedure and to enhance its effectiveness.

6.1. Window-based computation of the fictitious reference

Throughout the tuning procedure, several instances of the optimization problem in (9) must be solved. This operation might be computationally intense, especially when the fictitious reference must be retrieved over a large horizon N at once. To reduce computations, we can work sequentially on short windows of length $w \in \mathbb{N}$, with $w \ll N$, by defining the window-based predictor

$$\tilde{Y}_{k,w}(\theta_M)=F_w(\theta_M)R_{k,w}(\theta_M)+G_w(\theta_M)x_k^M(\theta_M), \tag{22}$$

where

$$R_{k,w}(\theta_M)=\begin{bmatrix}r_k(\theta_M) & r_{k+1}(\theta_M) & \cdots & r_{k+w}(\theta_M)\end{bmatrix}',$$
$$\tilde{Y}_{k,w}(\theta_M)=\begin{bmatrix}\tilde{y}_k(\theta_M) & \tilde{y}_{k+1}(\theta_M) & \cdots & \tilde{y}_{k+w}(\theta_M)\end{bmatrix}',$$

where $F_w\colon \mathbb{R}^{n_M}\to \mathbb{R}^{w\times w}$ and $G_w\colon \mathbb{R}^{n_M}\to \mathbb{R}^{w\times n_{a_M}}$ can be straightforwardly obtained from the observable canonical realization of (2a), and $k>0$ is a counter of past windows. The initial state $x_k^M\colon \mathbb{R}^{n_M}\to\mathbb{R}^{n_{a_M}}$ is instead equal to the last state obtained over the previous window. By exploiting this predictor, the fictitious reference can be retrieved by solving

$$\min_{\{r_\kappa^v\}_{\kappa=k}^{k+w}} \frac{1}{2}\|Y_{k,w}-\tilde{Y}_{k,w}(\theta_M)\|_2^2+\sum_{\kappa=k}^{k+w}\frac{\rho_{\mathcal{R}}}{2}\|r_\kappa-r_{\kappa-1}\|_2^2$$
$$\text{s.t. } \tilde{Y}_{k,w}(\theta_M)=F_w(\theta_M)R_{k,w}(\theta_M)+G_w(\theta_M)x_k^M(\theta_M),$$

in a receding horizon fashion, with

$$Y_{k,w}=\begin{bmatrix}y_k & y_{k+1} & \cdots & y_{k+w}\end{bmatrix}'.$$

To avoid spurious results due to the limited width w, it is advisable to overlap consecutive windows. The amount of overlapping is dictated by an additional parameter $\omega \ll w$.

6.2. Enhanced tuning by truncated propagation through time

When used to assess the performance of the actual closed-loop behavior, the cost and constraints in (11) might lead to a controller that, albeit matching the desired behavior over short horizons, does not provide satisfactory performance in the long run. A possible strategy to keep the complexity of the tuning scheme low, while inheriting some desirable features of simulation-based strategies, is to propagate input predictions for a limited number ξ of steps, with $\xi \ll N$. In the literature, such an idea usually goes by the name of *truncated back-propagation through time* (BPTT) (Werbos et al., 1990). To adopt such a solution, one needs to change the constraint in the control design problem (11) and have it replaced by

$$\hat{u}_k=\begin{cases}\mathcal{X}_k(\theta_M)\theta_C, & \text{if } mod(k,\xi)=0,\\ \hat{\mathcal{X}}_k(\theta_M)\theta_C, & \text{otherwise},\end{cases} \tag{23}$$

$$\mathcal{X}_k(\theta_M)=\begin{bmatrix}u_{k-1} & \cdots & u_{k-n_{a_C}} & e_k^\star(\theta_M) & \cdots & e_{k-n_{b_C}}^\star(\theta_M)\end{bmatrix},$$
$$\hat{\mathcal{X}}_k(\theta_M)=\begin{bmatrix}\hat{u}_{k-1} & \cdots & \hat{u}_{k-n_{a_C}} & e_k^\star(\theta_M) & \cdots & e_{k-n_{b_C}}^\star(\theta_M)\end{bmatrix},$$

where $mod(k,\xi)=k-\xi\lfloor\frac{k}{\xi}\rfloor$ and the dependence of $\hat{u}_k$ on θ_C is not explicitly shown, to simplify the notation. Note that, the number of steps ξ becomes an additional tuning parameter to be selected, which should be chosen to compromise between the complexity of the design problem and the potential degradation in the final closed-loop performance.

7. Numerical examples

We assess the performance of the proposed model reference tuning strategy in three numerical examples involving a minimum phase system, a non-minimum phase system, and the two-cart system considered in Carè, Torricelli, Campi, and Savaresi (2019). The sensitivity-based approach discussed in Section 5 has been implemented by using Casadi (Andersson, Gillis, Horn, Rawlings, & Diehl, 2019), IPOPT (Biegler & Zavala, 2009), and the MATLAB Optimization Toolbox (The MathWorks, Inc., 2020). To avoid the selection of unstable reference models, the cost ℓ_{perf} has been augmented with the indicator function $\mathcal{I}_M(\theta_M)$ of the set of stable reference models[2].

[2] This function returns ∞ when unstable reference models are considered.

Table 1
High-order benchmarks: hyper-parameters of the procedure.

n_{a_M}, n_{b_M}	n_{a_C}, n_{b_C}	γ	$\rho_{\mathcal{R}}$	ρ	w	ω	ξ
4	3	0.1	10^{-4}	10^{-4}	200	30	50

7.1. Minimum-phase plant

Consider the system characterized by the transfer function

$$P_1(q) = \frac{q^3 - 0.01534q^2 - 0.1239q + 0.005819}{q^4 + 1.452q^3 + 0.515q^2 - 0.04788q - 0.02777},$$

which has all stable poles and zeros. We want to tune a 4th order reference model, such that its step response has an overshoot $\bar{S}_{\%}$ of at most 7% and the tracking error is upper-bounded by $\delta = 0.01$ after at most 12 samples. Concurrently, we want to design a 3rd order controller featuring an integrator, that induces such a closed-loop behavior. As such, (11) is reformulated according to Remark 1 and, in particular, using the parameterization in (12). The parameters used to compute the reference model are reported in Table 1. A set of 1000 data points is collected by exciting the plant with a random white signal, drawn from a uniform distribution $\mathcal{U}(0, 1)$, and superimposing the output with a Gaussian white noise with zero mean and standard deviation $\sigma = 0.3$,[3] . In tuning the reference model, we further impose a unitary DC gain for the desired closed-loop system, *i.e.*, $G_\infty = 1$ (see (8)). As shown in Fig. 2 by initializing θ_M with a zero-mean Gaussian distributed random vector with standard deviation 0.05 satisfying (8), the overall loss

$$J^{(i)} = \ell_{fit}\left(\theta_C^{(i)}, \theta_M^{(i)}\right) + \gamma\, \ell_{perf}\left(\theta_M^{(i)}\right), \tag{24}$$

assumes an almost stationary value after about $i = 40$ iterations. The stopping criteria leading to the termination[4] of Algorithm 1 are instead met after 140 iterations. It is worth pointing out that the final outcome of Algorithm 1 depends on the selected initial condition, whose choice is particularly critical when no constraints other than (8) are imposed on θ_M. A possible solution to overcome this problem would be to test several initial conditions and pick the one that results in the best performance (e.g., the least cost), which we aim at investigating in the future.

The resulting reference model and the controller are

$$M_1(q) = \frac{0.662q^3 - 0.3314q^2 - 0.2641q + 0.5245}{q^4 - 0.8564q^3 - 0.1826q^2 + 0.902q - 0.2721},$$
$$C_1(q) = \frac{0.6051q^3 + 0.8248q^2 + 0.274q}{q^3 - 1.074q^2 + 0.01669q + 0.05775},$$

where $C_1(q)$ is obtained by using an instrumental variable scheme,[5] after the iteration of Algorithm 1 according to Remark 2. As reported in Fig. 3, the step response of the chosen reference model satisfies all the specifications. Meanwhile, the attained closed-loop output resembles the desired behavior, thus also satisfying the constraints imposed on the desired closed-loop behavior.

7.2. Nonminimum-phase plant

Consider the plant described by the transfer function

$$P_2(q) = \frac{q^3 - 0.801q^2 - 2.398q}{q^4 + 0.1184q^3 - 0.1259q^2 - 0.01075q + 0.0007783},$$

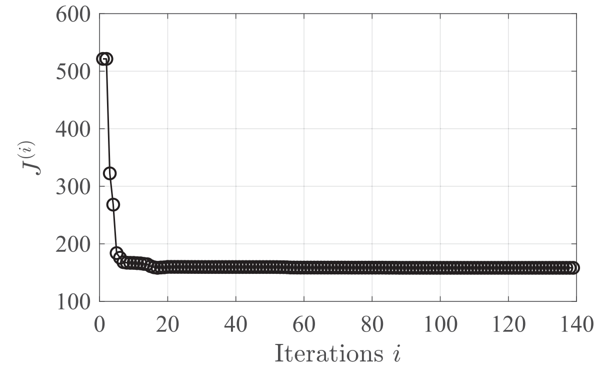


Fig. 2. Fourth-order minimum-phase plant: cost function $J^{(i)}$ in (24) *vs* iterations of Algorithm 1 during learning.

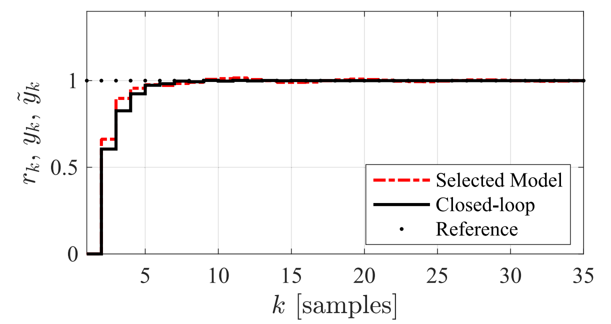


Fig. 3. Fourth-order minimum-phase plant: noiseless closed-loop step response (black) *vs* desired output (dashed red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

that has two zeros outside the unit circle, namely $q_1 = -1.199$ and $q_2 = 2$. We aim at designing a 3rd-order controller featuring an integrator and a 4th order reference model, the latter characterized by a step response with maximum overshoot $\bar{S}_{\%} = 7\%$ and maximum tracking error of $\delta = 0.01$ for $\kappa_s \leq 12$ samples, while using the parameters reported in Table 1. The dataset used for learning comprises 1000 samples, collected by feeding the plant with a random input in $\mathcal{U}(0, 1)$, with the outputs corrupted by noise with zero mean standard deviation $\sigma = 0.125$,[6]. As before, the linear constraint $G_\infty = 1$ is imposed to enforce perfect tracking of step-like set points by the reference model. Accordingly, an integrator is enforced into the controller, leading us to modify the controller parameterization loss in (11) as in Remark 1 (see (12) and (13)). As shown in Fig. 4 Algorithm 1 terminates after about 104 iterations, (*i.e.*, after about 50 seconds) while the the overall loss (24) becomes stationary after about 50 iterations.

The behavior of the automatically selected reference model and the attained output are compared in Fig. 5. Clearly, the chosen reference model allows us to design a stabilizing controller, preventing zero-pole cancellation between the plant and the controller. Indeed, the selected reference model turns out to be:

$$M_2(q) = \frac{-0.08465q^3 - 0.08806q^2 + 0.3471q + 0.396}{q^4 + 0.3343q^3 - 0.8347q^2 - 0.2146q + 0.2854},$$

which features a zero in $q_1' \approx -1.19$ and to other ones in $q_2' \approx 2.06$ and $q_3' \approx -1.90$. Therefore, the proposed strategy allows us

[3] This corresponds to a Signal-to-Noise Ratio on the output of approximately 20.6 dB.

[4] The termination criteria are the conventional ones of fmincon in The MathWorks, Inc. (2020), with both function and step tolerances fixed at 10^{-6}.

[5] The instrument corresponds to the regressor $\mathcal{X}_k$ constructed with a new dataset, by exploiting the chosen reference model.

[6] This corresponds to a Signal-to-Noise Ratio of 26.4 dB.

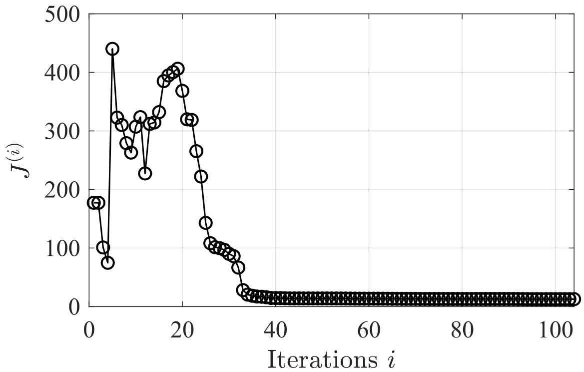


Fig. 4. Fourth-order nonminimum-phase plant: cost function $J^{(i)}$ in (24) *vs* iterations of Algorithm 1 during learning.

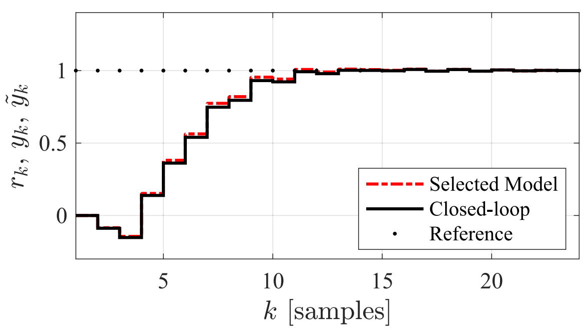


Fig. 5. Fourth-order nonminimum-phase plant: actual noiseless closed-loop step response (black) *vs* desired output (dashed red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

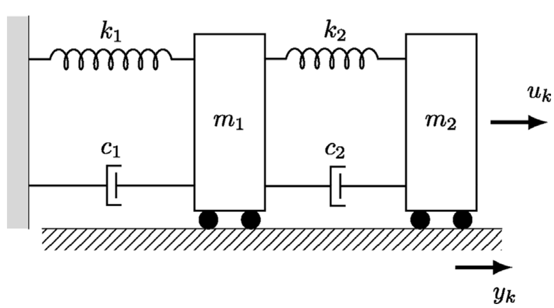


Fig. 6. Schematic representation of the two-cart system (Carè et al., 2019).

Table 2
Parameters of the two-cart system (Carè et al., 2019).

m_1 [kg]	m_2 [kg]	c_1 [N/m^2]	c_2 [N/m^2]	k_1 [N/m]	k_2 [N/m]
1	0.5	0.2	0.5	1	0.5

to enforce the unstable zeros of P_2 into the reference model, with the controller refined with instrumental variables being

$$C_2(q) = \frac{-0.088q^3 - 0.1348q^2 + 0.05459q}{q^3 - 0.02828q^2 - 0.7526q - 0.2191}.$$

We stress that *no prior information on the unstable zeros has been exploited.*

7.3. Two-cart system

We consider the two-cart system depicted in Fig. 6, whose parameters are reported in Table 2 and are regarded as totally unknown for control design. A normally distributed input force u_k [N] is applied to the carts to collect noisy measures of the position y_k [m] of the second cart m_2 at a sampling rate of $f_s = 10$ [s^{-1}]. We gather a dataset of length $T = 5000$. The measured output is corrupted by an additive, zero-mean white noise v_k with standard deviation equal to 0.16 as in Carè et al. (2019). This dataset is used to tune a first-order reference model, namely

Table 3
Two-carts example: hyper-parameters of the tuning procedure.

n_{a_M}	n_{b_M}	n_{a_C}	n_{b_C}	γ	$\rho_{\mathcal{R}}$	ρ	w	ω	ξ
1	1	2	2	0.1	10	0.1	50	20	200

$$M(q) = \frac{1-\theta_M}{q-\theta_M}, \tag{25}$$

by imposing that

$$0.6 \approxeq e^{-\frac{4.6T_s}{t_{s-}}} \leq \theta_M \leq e^{-\frac{4.6T_s}{t_{s+}}} \approxeq 0.98, \tag{26}$$

to enforce a closed-loop settling time within $[1, 25]$ [s] in the step response. The proposed tuning algorithm is run by setting its hyper-parameters as in Table 3, with $\nu = 4960$ in (6) and input/output data filtered by using the same low-pass filter exploited in Carè et al. (2019) to enforce matching at frequencies lower than 2 [rad/s]. Moreover, we initially consider $\theta_M^{(0)} = \begin{bmatrix} 0.79 & 0.21 \end{bmatrix}'$, and we stop searching for the reference model after 100 iterations have been performed, or when the cost decreases less than 10^{-6} within two successive iterations. According to what is done in Carè et al. (2019), we aim at matching the behavior dictated by the auto-tuned reference model via a discrete-time PID controller C_{θ_C}, namely

$$C(\theta_C, q) = [\theta_C]_1 + [\theta_C]_2 \frac{T_s}{2}\frac{1+q^{-1}}{1-q^{-1}} + [\theta_C]_3 \frac{2}{T_s}\frac{1-q^{-1}}{3-q^{-1}}, \tag{27}$$

where $[\theta_C]_1$, $[\theta_C]_2$ and $[\theta_C]_3$ are the proportional, integral and derivative gains of the controller, respectively. After the selection of the reference model through Algorithm 1, the controller is subsequently refined via the VRFT toolbox (Carè et al., 2019).

The noiseless desired response obtained by considering the same square wave reference used in Carè et al. (2019) is compared to the attained closed-loop one in Fig. 7 over a validation test of length 250 [s]. Clearly, the desired behavior is overall matched, with a short transient in which the actual closed-loop response oscillates. As it can be noticed in Fig. 8, also fixing the reference model a priori as in Carè et al. (2019) leads to some oscillations at set-point changes, along with a significant deterioration of tracking performance due to the longer settling time. This result highlights that the proposed auto-tuning procedure allows one to design data-driven controllers resulting in improved closed-loop tracking performance. The difference between the closed-loops obtained by fixing or auto-tuning the reference model can be further observed by the comparison of their magnitude Bode plots in Fig. 9. Clearly both strategies lead to a reduction in the magnitude peak characterizing the open-loop system (see Fig. 9), with the auto-tuning procedure guaranteeing a broader bandwidth. The CPU time required to assess the cost, constraints and derivatives takes 30 s, approximately, while Algorithm 1 converges in 15 iterations (as shown in Fig. 10).

7.3.1. Comparative analysis

By exploiting the same parameters in Table 3, we compare the results obtained via our derivative-based approach and the ones attained by exploiting two global optimization methods to solve problem (4), namely Particle Swarm Optimization (The MathWorks, Inc., 2020) (PSO) and GLIS (Bemporad, 2020). Specifically,

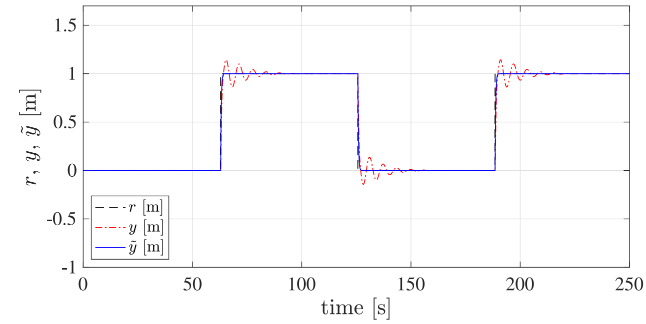


Fig. 7. Two-carts example: desired (blue) *vs* attained (red) response with the auto-tuned reference model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

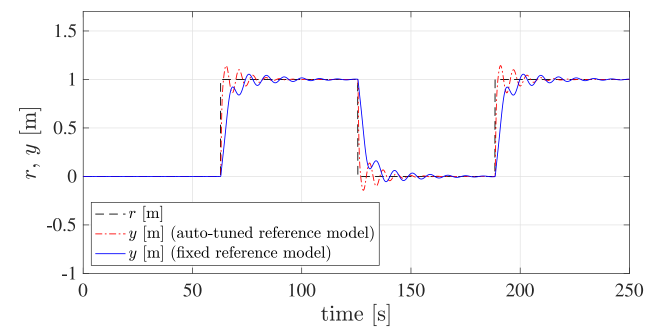


Fig. 8. Two-carts example: attained response with the auto-tuned (red) and the fixed (Carè et al., 2019) (blue) reference model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

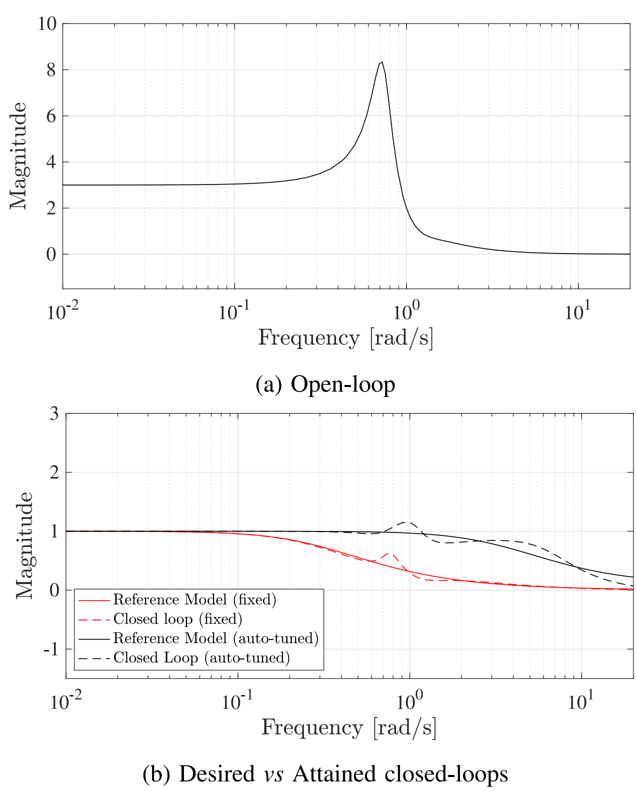


Fig. 9. Two-cart example: Magnitude Bode plots.

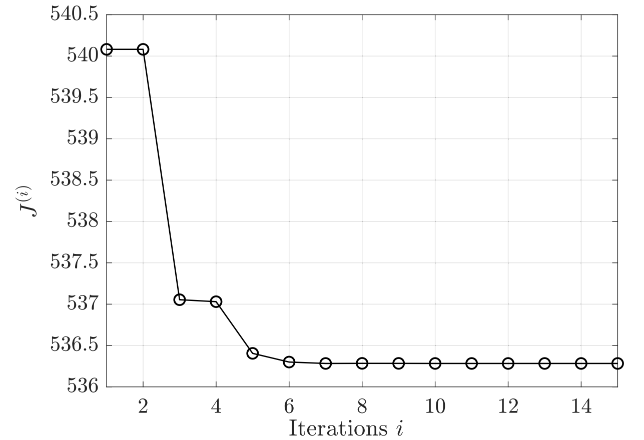


Fig. 10. Two-cart example: cost function $J^{(i)}$ in (24) *vs* iterations of Algorithm 1 during learning..

the PSO algorithm is run up to a maximum of 100 iterations, while the GLIS routine has been carried out by imposing a maximum of 100 function evaluations and by using Gaussian radial basis functions to build the surrogate cost. *All the approaches lead to the selection of the same reference model*, while requiring 15, about 300, and 100 function evaluations, respectively. This result highlights that the proposed method can be competitive with respect to derivative-free optimization methods when auto-tuning a reference model based on the same objective and constraints, while requiring a lower number of function evaluations.

To provide additional insights on the proposed sensitivity-based method, we compare the performance attained with auto-tuned reference models obtained by either applying our approach or the performance-oriented ones introduced in Selvi et al. (2018) and Breschi and Formentin (2021). Note that, the PSO-based method proposed in Selvi et al. (2018) requires the definition of a set-point $\tilde{r}$ to weight the desired closed-loop performance within the training phase, which is instead not required by the BO approach presented in Breschi and Formentin (2021). To assess the tracking performance we use the following indicators

$$\text{RMSE}_y = \sqrt{\frac{1}{T_v}\sum_{k=1}^{T_v}(y_k - r_k)^2}, \quad [m] \tag{28a}$$

$$\text{RMSE}_d = \sqrt{\frac{1}{T_v}\sum_{k=1}^{T_v}(y_k - \tilde{y}_k)^2}, \quad [m] \tag{28b}$$

that are evaluated in closed-loop by considering the reference signal in Fig. 7–8, so that $T_v = 2500$. Table 4 reports the values of these indexes for the different approaches[7].

Clearly, the reference model tuned with the proposed strategy leads to better performance both in terms of target signal and behavior tracking with respect to the one proposed in Selvi et al. (2018). At the same time, it leads to the choice of a desired response as attainable as the one retrieved with the approach presented in Breschi and Formentin (2021), as proven by the values of the RMSE_d index and as expected given the chosen objective function. The results in Table 4 thus prove that the proposed approach can be competitive with respect to alternative strategies in terms of tracking, while resulting in a prudent choice of the reference model. This last feature might

[7] When running the methods described in Selvi et al. (2018) and Breschi and Formentin (2021), $W_y = W_{\Delta u} = 1$ and $W_u = 10$.

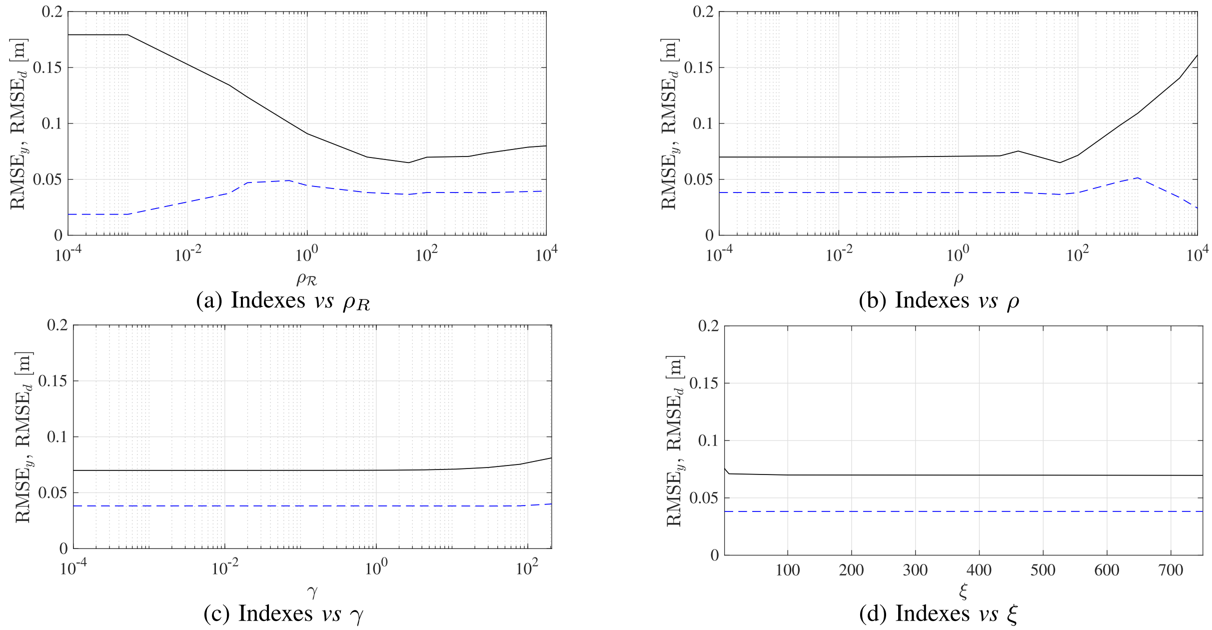


Fig. 11. Two-cart example, sensitivity analysis: RMSE$_y$ (black), RMSE$_d$ (blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Two-cart example: sensitivity-based *vs* methods in Breschi and Formentin (2021) and Selvi et al. (2018).

	Sensitivity-based	BO-based (Breschi & Formentin, 2021)	PSO-based (Selvi et al., 2018)
θ_M	0.6831	0.6004	0.8254
θ_C	$\begin{bmatrix}0.2300\\0.3000\\0.5918\end{bmatrix}$	$\begin{bmatrix}0.2450\\0.4000\\0.7432\end{bmatrix}$	$\begin{bmatrix}0.1690\\0.1400\\0.3257\end{bmatrix}$
RMSE$_y$ [m]	0.070	0.065	0.084
RMSE$_d$ [m]	0.038	0.037	0.060

Table 5
Principal hyper-parameters of the sensitivity-based method.

Parameter name	Brief description
γ	Relative importance weight in (4)
$\rho_{\mathcal{R}}$	Regularization weight in (10)
ρ	Regularization weight in (11b)
w	Length of the window in (22)
ω	Overlapping between windows in (22)
ξ	Steps of BPTT in (23)

entail that the proposed method tends to avoid excessively demanding reference models , which is desirable not to jeopardize the control design phase. Note that, when tuning a continuous-time PID with the Ziegler–Nichols method, we achieve RMSE$_y$ $= 0.197$. Therefore, this classical approach is outperformed by all the data-driven approaches with auto-tuning considered in this comparative analysis.

7.3.2. *Sensitivity analysis*

The proposed approach requires the user to select a set of hyper-parameters, summarized in Table 5. We explore the sensitivity of the resulting closed-loop performance with respect to some of them, by alternatively fixing all their values as in Table 3 but one. The attained closed-loop performance is assessed based on the indicators in (28). Fig. 11(a)–11(b) show that the resulting closed-loop performance heavily depends on the choice of the regularization parameters characterizing the costs in (10) and (11b). In particular, an excessive weight on the variations of the virtual reference deteriorates tracking performance. The same consideration holds for the regularization parameters ρ. At the same time, $\rho_{\mathcal{R}}$ should not be excessively small. It is worth to point out that, even if the macro-effect shown in Fig. 11(a)–11(b) are likely to be generalizable to other applications, one should tune $\rho_{\mathcal{R}}$ and ρ every time a new application is considered, *e.g.*, by customizing the open-loop calibration approach proposed in Breschi and Formentin (2020a, Section 6). Concerning the weight γ, Fig. 11(c) clearly indicates that a rather broad range of values of this parameter allows us to retrieve the same reference model. Nonetheless, prioritizing the performance of the reference model over fitting in the design of the data-driven controller (*i.e.*, choosing a high γ), leads to a degradation of tracking performance. Given the role played by γ, it is worth pointing out that its value is likely to be application-dependent and, thus, results might be more sensitive to its choice when other systems and closed-loop requirements are considered. The results reported in Fig. 11(d) instead show that performance is generally insensitive to ξ, in turn indicating that the inputs reconstructed with the controller match quite tightly the actual ones. Note that, in general, the performance might be more sensitive to the choice of ξ, especially when ξ is high. One could thus progressively increase its value, until the matching between the data and the input reconstructed with the learned controller becomes “inaccurate” (according to a user-defined metric, *e.g.*, the one considered in Breschi and Formentin (2020a, Section 6)).

8. Conclusions

In this paper, we have proposed a derivative-based method for feedback control design from input/output data without the need of identifying a model of the process. The approach is flexible, in that it allows imposing constraints on the shape of the desired closed-loop response, and can handle non-minimum phase plants. The presented simulation results have illustrated the effectiveness of the approach in a variety of scenarios, especially as compared to state of the art methods.

Future research will be devoted to extend the approach to multi-input/multi-output systems and to provide formal stability and convergence guarantees. We will also seek to endow the method with structure selection capabilities and to pair it with preference-based global approaches, *e.g.,* see Zhu, Piga, and Bemporad (2022), to auto-tune the hyper-parameters and further refine and customize the reference model and the controller.

References

Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., & Diehl, M. (2019). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, *11*(1), 1–36.

Bemporad, A. (2020). Global optimization via inverse distance weighting and radial basis functions. *Computational Optimization and Applications*, *77*, 571–595, Code available at http://cse.lab.imtlucca.it/~bemporad/glis.

Berberich, J., Köhler, J., Müller, M. A., & Allgöwer, F. (2021). Data-driven model predictive control with stability and robustness guarantees. *IEEE Transactions on Automatic Control*, *66*(4), 1702–1717.

Biegler, L. T., & Zavala, V. M. (2009). Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, *33*(3), 575–582.

Breschi, V., & Formentin, S. (2020a). Direct data-driven design of switching controllers. *International Journal of Robust and Nonlinear Control*, *30*(15), 6042–6072.

Breschi, V., & Formentin, S. (2021). Proper closed-loop specifications for data-driven model-reference control. *IFAC-PapersOnLine*, *54*(9), 46–51, 24th International Symposium on Mathematical Theory of Networks and Systems MTNS 2020.

Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599.

Campestrini, L., Eckhard, D., Gevers, M., & Bazanella, A. S. (2011). Virtual reference feedback tuning for non-minimum phase plants. *Automatica*, *47*(8), 1778–1784.

Campi, M. C., Lecchini, A., & Savaresi, S. M. (2002). Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica*, *38*(8), 1337–1346.

Carè, A., Torricelli, F., Campi, M. C., & Savaresi, S. M. (2019). A toolbox for virtual reference feedback tuning (VRFT). In *2019 18th European control conference* (pp. 4252–4257).

Cerone, V., Abuabiah, M., & Regruto, D. (2020). An H_∞ method to design the reference model in direct data-driven control design. In *2020 59th IEEE conference on decision and control* (pp. 6004–6009).

Formentin, S., Campi, M. C., Carè, A., & Savaresi, S. M. (2019). Deterministic continuous-time virtual reference feedback tuning (VRFT) with application to PID design. *Systems & Control Letters*, *127*, 25–34.

Formentin, S., Piga, D., Tóth, R., & Savaresi, S. M. (2016). Direct learning of LPV controllers from data. *Automatica*, *65*, 98–110.

Gonçalves da Silva, G. R., Bazanella, A. S., & Campestrini, L. (2019). On the choice of an appropriate reference model for control of multivariable plants. *IEEE Transactions on Control Systems Technology*, *27*(5), 1937–1949.

Hjalmarsson, H., Gevers, M., Gunnarsson, S., & Lequin, O. (1998). Iterative feedback tuning: theory and applications. *IEEE Control Systems Magazine*, *18*(4), 26–41.

Jeon, Younghan, Lee, Minsik, & Choi, Jin Young (2020). Differentiable forward and backward fixed-point iteration layers. arXiv preprint arXiv:2002.02868.

Kergus, P., Olivi, M., Poussot-Vassal, C., & Demourant, F. (2019a). Data-driven reference model selection and application to L-DDC design. arXiv preprint arXiv:1905.04003.

Kergus, P., Olivi, M., Poussot-Vassal, C., & Demourant, F. (2019b). From reference model selection to controller validation: Application to loewner data-driven control. *IEEE Control Systems Letters*, *3*(4), 1008–1013.

Lecchini, A., Campi, M. C., & Savaresi, S. M. (2001). Virtual reference feedback tuning: A new framework for data-based design of PID and linear controllers. *IFAC Proceedings Volumes*, *34*(14), 83–88, IFAC Workshop on Adaptation and Learning in Control and Signal Processing (ALCOSP 2001), Cernobbio-Como, Italy, 29-31 August 2001.

Lecchini, A., & Gevers, M. (2002). On iterative feedback tuning for non-minimum phase plants. In *Proceedings of the 41st IEEE conference on decision and control, 2002, vol. 4* (pp. 4658–4663).

Lee, W. S., Anderson, B. D. O., Mareels, I. M. Y., & Kosut, R. L. (1995). On some key issues in the windsurfer approach to adaptive robust control. *Automatica*, *31*(11), 1619–1636.

Luenberger, D. G., & Ye, Y. (2016). *Linear and nonlinear programming*. Springer.

Nijmeijer, H., & Savaresi, S. M. (1998). On approximate model-reference control of SISO discrete-time nonlinear systems. *Automatica*, *34*(10), 1261–1266.

Ogata, K. (1994). *Discrete-time control systems*. Prentics-Hall, Inc.

Rahmat-Samii, Y., Gies, D., & Robinson, J. (2003). Particle swarm optimization (PSO): a novel paradigm for antenna designs. *URSI Radio Science Bulletin*, *2003*(306), 14–22.

Schoukens, J., & Ljung, L. (2019). Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, *39*(6), 28–99.

Selvi, D., Piga, D., Battistelli, G., & Bemporad, A. (2021). Optimal direct data-driven control with stability guarantees. *European Journal of Control*, *59*, 175–187.

Selvi, D., Piga, D., & Bemporad, A. (2018). Towards direct data-driven model-free design of optimal controllers. In *2018 European control conference* (pp. 2836–2841).

The MathWorks, Inc. (2020). MATLAB optimization toolbox, Natick, Massachusetts, United State.

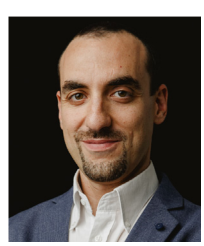
van Meer, M., Breschi, V., Oomen, T., & Formentin, S. (2021). Direct data-driven design of LPV controllers with soft performance specifications. *Journal of the Franklin Institute*, *359*, 816–836.

Werbos, P. J., et al. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560.

Zhu, M., Piga, D., & Bemporad, A. (2022). C-GLISp: Preference-based global optimization under unknown constraints with applications to controller calibration. *IEEE Transactions on Control Systems Technology*, *30*(5), 2176–2187.

Daniele Masti was born in Siena, Italy in 1993. Daniele Masti received the Bachelor's degree in computer and information engineering from the University of Siena, Italy, in 2015, the Master's degree in electric and automation engineering in 2018 from the University of Florence, Italy and the Ph.D. in systems science in 2021 at IMT School for Advanced Studies Lucca, Italy. From 2022, he is a researcher in cyber security at IMT School for Advanced Studies in Lucca, Italy. His main research interest lies at the border between control theory and machine learning, with the overall aim of bridging the gap between the two, and network security.

Valentina Breschi received her B.Sc. in Electronic and Telecommunication Engineering and her M.Sc. in Electrical and Automation Engineering from the University of Florence (Italy) in 2011 and 2014, respectively. She received her Ph.D. in Control Systems from IMT School for Advanced Studies Lucca (Italy) in 2018. From January to July 2017, she was a visiting scholar at the Department of Aerospace Engineering, University of Michigan (United States). From 2018 to 2020, she was a post-doctoral researcher at Politecnico di Milano (Italy), where she then held a fixed-term position as a junior assistant professor from 2020 to 2023. In 2023, she joined the Department of Electrical Engineering at Eindhoven University of Technology (Netherlands) as an Assistant Professor. Her main research interests include data-driven control, system identification, with a focus on piecewise affine and switching systems, collaborative learning, and human-centered policy design, focusing on mobility systems.



Simone Formentin was born in Legnano (MI), Italy, in 1984. He received his B.Sc. and M.Sc. degrees cum laude in Automation and Control Engineering from Politecnico di Milano, Italy, in 2006 and 2008, respectively. In 2012, he obtained his Ph.D. degree cum laude in Information Technology within a joint program between Politecnico di Milano and Johannes Kepler University of Linz, Austria. After that, he held two post-doctoral appointments at the Swiss Federal Institute of Technology of Lausanne (EPFL), Switzerland and the University of Bergamo, Italy, respectively. Since 2014, he has been with Politecnico di Milano, first as an assistant professor, then as an associate professor. He is the chair of the IEEE TC on System Identification and Adaptive Control, the social media representative of the IFAC TC on Robust Control and a member of the IFAC TC on Modeling, Identification and Signal Processing. He is an Associate Editor of Automatica and the European Journal of Control. His research interests include system identification and data-driven control with a focus on automotive and financial applications.

Alberto Bemporad received his Master's degree cum laude in Electrical Engineering in 1993 and his Ph.D. in Control Engineering in 1997 from the University of Florence, Italy. In 1996/97 he was with the Center for Robotics and Automation, Department of Systems Science and Mathematics, Washington University, St. Louis. In 1997–1999 he held a postdoctoral position at the Automatic Control Laboratory, ETH Zurich, Switzerland, where he collaborated as a Senior Researcher until 2002. In 1999–2009 he was with the Department of Information Engineering of the University of Siena, Italy, becoming an Associate Professor in 2005. In 2010–2011 he was with the Department of Mechanical and Structural Engineering of the University of Trento, Italy. Since 2011 he is Full Professor at the IMT School for Advanced Studies Lucca, Italy, where he served as the Director of the institute in 2012–2015. He spent visiting periods at Stanford University, University of Michigan, and Zhejiang University. In 2011 he co-founded ODYS S.r.l., a company specialized in developing model predictive control systems for industrial production. He has published more than 400 papers in the areas of model predictive control, hybrid systems, optimization, automotive control, and is the co-inventor of 21 patents. He is author or coauthor of various software packages for model predictive control design and implementation, including the Model Predictive Control Toolbox (The Mathworks, Inc.) and the Hybrid Toolbox for MATLAB. He was an Associate Editor of the IEEE Transactions on Automatic Control during 2001–2004 and Chair of the Technical Committee on Hybrid Systems of the IEEE Control Systems Society in 2002–2010. He received the IFAC High-Impact Paper Award for the 2011–14 triennial, the IEEE CSS Transition to Practice Award in 2019, and the 2021 SAE Environmental Excellence in Transportation Award. He is an IEEE Fellow since 2010.