Towards direct data-driven model-free design of optimal controllers

Daniela Selvi[1], Dario Piga[2] and Alberto Bemporad[1]

***Abstract*— The most critical step in modern direct data-driven control design approaches, such as virtual reference feedback tuning and non-iterative correlation-based tuning, is the choice of an adequate closed-loop reference model. Indeed, the chosen reference model should reflect the desired closed-loop performance but also be reproducible by the underlying unknown process when in closed loop with the synthesized controller. In this paper, we propose a novel approach to compute, directly from data, an "optimal" reference model along with the corresponding controller. The performance index used to define the optimality of the reference model measures the tracking error and the actuator efforts (as it is typical in performance-driven controllers such as linear-quadratic Gaussian control and model predictive control), along with a term penalizing the expected mismatch between the reference model and the actual closed-loop system. The performance index depends on the variables used to parametrize the reference model and the controller, which are optimized through a suitable combination of particle swarm optimization and virtual reference feedback tuning.**

I. INTRODUCTION

The design of optimal controllers, such as linear-quadratic Gaussian (LQG) and model predictive controllers (MPCs), relies on a dynamical model of the open-loop process. In the case a physical description of the process is not available, system identification algorithms can be employed to train a model from data, which is then used to the design the controller [1]. Designing a controller by first identifying an open-loop model of the process and then synthesizing a controller based on the resulting model would require focusing the attention on reproducing the open-loop behavior of the process first, then selecting a model structure, tune the model parameters, compare the open-loop response of the model on validation data, etc. However, even in the applications where deriving a model from data is neither costly- nor timely-consuming, it remains difficult to decide a-priori the level of accuracy/complexity the model should have to meet the desired closed-loop performance.

As an alternative to model-based control, direct data-driven design methods have been proposed in the literature (see for example [2]-[10] and the references therein). In direct data-driven control, the only information available on the system (namely, data) is directly used to achieve the final control objectives (usually expressed in terms of a stable *reference model* M), thus avoiding the intermediate and approximation step of exploiting this information to derive an open-loop model of the process. We can label these methods as *model-free*, meaning that a model of the plant is not required to synthesize the control law.

The direct approaches sound appealing as only the desired closed-loop reference model has to be specified. However, the choice of an adequate reference model M usually represents a critical step in any direct data-driven method. Indeed, M should reflect the desired closed-loop performance, but it should also take into account the capability of the underlying unknown process to reproduce the desired behavior when the loop is closed with the synthesized controller. This issue is highlighted in [11], and handled through a hierarchical control architecture. The main idea of [11] is to first design, directly from data, an inner controller to match a simple low-performing closed-loop reference model M, which is then used to synthesize an outer model predictive controller to enhance the performance of the inner loop. Although the results in [11] clearly show the benefits of the hierarchical control architecture, choosing a too low-performing reference model for the inner loop might lead to an outer MPC with aggressive control actions, which may not be provided because of physical limitations or might even excite dynamics not described by the inner closed-loop model M.

With the general ambition of synthesizing optimal control laws directly from data in a model-free way, in this paper we provide a heuristic method to determine an optimal (according to some performance index) reference model M achievable by a controller designed through any data-driven design approach, such as the Virtual Reference Feedback Tuning (VRFT) [2] reviewed in Section II. The index used to define the optimality of the reference model M is introduced in Section III-A and it comprises two terms. The first one is the typical cost used in optimal controllers (such as LQG and MPC) and it measures the performance of the model M, defined by the energy of the tracking error and the actuator efforts. The second term measures the expected mismatch between the desired reference model M and the actual closed-loop. The performance index is optimized through particle swarm optimization, and the optimal reference model M is extracted along with the corresponding controller C, as shown in Section III-B. Constraints on input and output signals can also be handled, as discussed in Section III-C. In Section IV the effectiveness of the proposed approach is shown by means of numerical examples, and the achieved

*This work was partially supported by the European Commission under project H2020-SPIRE-636834 "DISIRE - Distributed In-Situ Sensors Integrated into Raw Material and Energy Feedstock" (http://spire2030.eu/disire) and by the H2020-723248 project "DAEDALUS - Distributed control and simulation platform to support an ecosystem of digital automation developers".

[1]Daniela Selvi and Alberto Bemporad are with the IMT School for Advanced Studies Lucca, 55100 Lucca, Italy {daniela.selvi, alberto.bemporad}@imtlucca.it

[2]Dario Piga is with the IDSIA Dalle Molle Institute for Artificial Intelligence, USI-SUPSI, 6928 Manno, Switzerland. dario.piga@supsi.ch

performances are compared to the ones attained by a linear quadratic controller.

II. DIRECT CONTROLLER SYNTHESIS

In this section, we formulate the data-driven control design problem and we briefly recall the VRFT approach [2], used in this paper to synthesize a controller C from data to match a given reference model M.

Our goal is to control a process having only a dataset of N input and output samples $(u(0), \dots, u(N-1))$, $(y(0), \dots, y(N-1))$ available, without attempting at finding first a model of the open-loop process and, as we will discuss in Section III, with the ambition of optimizing a certain closed-loop performance index.

In the following, we restrict to the single input single output case ($u, y \in \mathbb{R}$), to a linear reference model defined in terms of a stable discrete-time operator $M(\theta, q)$, and to a linear discrete-time control law

$$u(t) = C(\varphi, q)(r(t) - y(t)) + c_0(\varphi) r(t)$$

where q is the forward shift operator ($q\,u(t) = u(t+1)$), $r(t) \in \mathbb{R}$ is the reference signal to be tracked, $\theta \in \mathbb{R}^{n_\theta}$ collects a parameterization of M (for example, the one expressed as in (8) in Section III-B), and $\varphi \in \mathbb{R}^{n_\varphi}$ the coefficients parameterizing the control law. Note that if $C(\varphi, q)$ is parameterized to contain an integrator block, the feedforward term $c_0(\varphi)$ could be zeroed.

According to the VRFT approach, for a given reference model $M(\theta, q)$ we can introduce the *virtual reference* $r_v(\theta, t)$, defined by the relation

$$y(t) = M(\theta, q) r_v(\theta, t) \tag{1}$$

and the corresponding virtual tracking error

$$e_v(\theta, t) = r_v(\theta, t) - y(t) \tag{2}$$

Note that the definition of the virtual reference $r_v(\theta, t)$ requires to compute the left inverse $M^\dagger(\theta, q)$ of $M(\theta, q)$, i.e., $M^\dagger(\theta, q) M(\theta, q) = 1$. For linear time-invariant or linear parameter-varying (LPV) reference models, the left inverse $M^\dagger(\theta, q)$ can be computed as indicated in [8, Proposition 1].

The controller synthesis problem can be recast as the one of minimizing the difference

$$\epsilon(\theta, \varphi, t) = u(t) - u_v(\theta, \varphi, t)$$

between the collected input data samples $u(t)$ and the virtual input $u_v(\theta, \varphi, t)$ obtained by feeding the virtual error e_v to the controller $C(\varphi, q)$, i.e.,

$$u_v(\theta, \varphi, t) = C(\varphi, q) e_v(\theta, t) + c_0(\varphi) r_v(t). \tag{3}$$

Accordingly, for a given input/output dataset of N samples, we choose the controller parameter vector

$$\varphi^\star(\theta) = \arg\min_\varphi \sum_{t=0}^{N-1} \ell(\epsilon(\theta, \varphi, t)) \tag{4}$$

given some loss function $\ell : \mathbb{R} \to \mathbb{R}$, with $\ell(0) = 0$, $\ell(\epsilon) \geq 0$, $\forall \epsilon \in \mathbb{R}$. For simplicity, we will use $\ell(\epsilon) = \epsilon^2$ in this work.

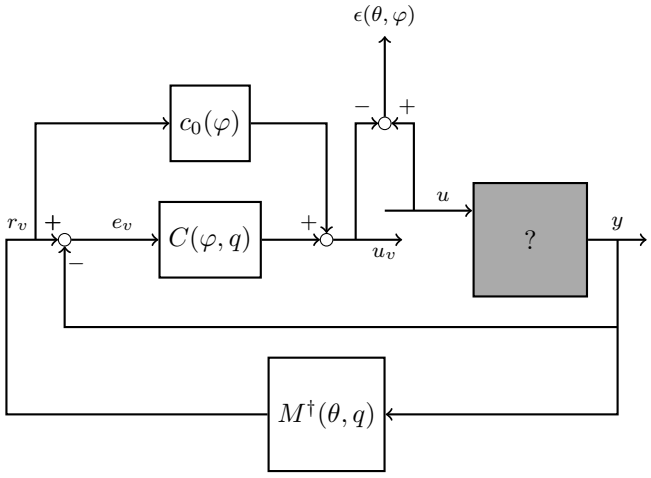

Fig. 1: Virtual reference feedback tuning: a schematic representation. The grey box containing the question mark "?" represents the unknown plant.

However, *any* black-box parametric systems identification method (or, more generally, any regression method) can be used instead of (4) to determine a parameter vector $\varphi^\star$ such that the virtual input $u_v(\theta, \varphi, t)$ in (3) matches $u(t)$ as much as possible. We may also consider linear-parameter varying (LPV) controllers, as well as nonlinear controllers using different parameterizations of the control laws (e.g., a neural network).

A schematic visualization of the VRFT concept is shown in Figure 1.

III. OPTIMAL REFERENCE MODEL SELECTION

A. Optimality criterion

In the above section we have shown how to design a controller from data for a given reference model $M(\theta, q)$. We need now to define a criterion for selecting a parameter vector $\theta^\star$ that makes the corresponding closed-loop reference model $M(\theta^\star, q)$ "optimal". Let $r(0), \dots, r(N-1)$ be a "representative enough" reference signal, that is of the type we expect the controller will be asked to track once it is deployed. The reference r could be for example a collection of random steps, ramps, sinusoids, square waves, etc.

Optimality of the reference model $M(\theta^\star, q)$ is defined according to the following cost

$$\begin{aligned} J(\theta) = \frac{1}{N} \sum_{t=0}^{N-1} \big\{ & W_y (r(t) - y_p(\theta, t))^2 + W_{\Delta u} \Delta u_p^2(\theta, t) \\ & + W_{\text{fit}} (u(t) - u_v(\theta, t))^2 \big\} \end{aligned} \tag{5}$$

where

$$y_p(\theta, t) = M(\theta, q) r(t) \tag{6a}$$

is the "perfect" output that would result by feeding the reference r to the closed-loop system when the reference model $M(\theta, q)$ is perfectly matched;

and

$$\Delta u_p(\theta, t) = u_p(\theta, t) - u_p(\theta, t-1)$$

are the input increments, with

$$u_p(\theta,t) = C(\varphi^\star(\theta),q)(r(t) - y_p(\theta,t)) + c_0(\varphi^\star(\theta))r(t) \tag{6b}$$

being the input produced by the controller $C(\varphi^\star(\theta),q)$, $c_0(\varphi^\star(\theta))$ synthesized to match $M(\theta,q)$.

The performance index defined in (5) is the typical cost function used in performance-driven control, such as in model predictive control [12]. The first two terms relate to the performance the system would show if the reference model $M(\theta,q)$ were perfectly matched by the closed-loop process. The weights $W_y, W_{\Delta u} > 0$ trade off between reducing the tracking error and reducing the effort of the actuator, and ultimately decide the tradeoff between speed of convergence and robustness. The third term accounts for how well the model $M(\theta,q)$ is matched. Without loss of generality we can set $W_y = 1$, so that only the remaining two weights are left as tuning knobs of the synthesis procedure.

Note that in (5) we could also add a penalty $W_u(u_p(t) - u_r(t))^2$ on tracking errors related to input references. However, usually a static model of the open-loop process is required to generate a consistent input reference $u_r(t)$ from $r(t)$. Here we want to instead restrict ourselves in a completely model-free setting.

The optimal parameter vector $\theta^\star$ is chosen by solving the optimization problem

$$\theta^\star = \arg\,\min_\theta J(\theta) \tag{7}$$

and, correspondingly, the synthesized data-driven optimal controller is $C(\varphi^\star(\theta^\star),q)$, $c_0(\varphi^\star(\theta^\star))$.

The optimization problem (7) is in general nonlinear and nonconvex, as to evaluate $J(\theta)$ it requires an identification procedure (e.g., the VRFT approach in Section II) to get $\varphi^\star(\theta)$ in (6b). Nonetheless it involves a limited number of optimization variables, i.e., the parameters defining the reference model $M(\theta,q)$.

B. Numerical optimization

Particle swarm optimization (PSO) [13] is used in this work, due to the limited number of optimization variables and the desire of finding a global optimizer. Algorithm 1 summarizes the main steps of the PSO-based approach employed to compute the optimal reference model $M(\theta^*,q)$.

In order to use PSO, the reference model $M(\theta,q)$ is parametrized in its zero-pole-gain representation

$$M(\theta,q) = K_c \frac{\prod_i^{n_{zc}} \left(q - z_i^{c,re} \pm j z_i^{c,im}\right) \prod_i^{n_{zr}} (q - z_i^r)}{\prod_i^{n_{pc}} \left(q - p_i^{c,re} \pm j p_i^{c,im}\right) \prod_i^{n_{pr}} (q - p_i^r)} \tag{8}$$

with n_{zc}, n_{zcr}, n_{pc}, n_{pr} denoting the number of complex conjugate zeros, real zeros, complex conjugate poles and real poles, respectively, and $\{z_i^{c,re}, z_i^{c,im}\}_{i=1}^{n_{zc}}$, $\{z_i^r\}_{i=1}^{n_{zr}}$, $\left\{p_i^{c,re}, p_i^{c,im}\right\}_{i=1}^{n_{pc}}$, $\{p_i^r\}_{i=1}^{n_{pr}}$ stacked in the parameter vector θ (Step 2.1.1). The variable K_c is not optimized and it acts as a normalization constant to enforce $M(\theta,1) = 1$ (unitary static gain for stable M) (Step 2.1.2). For each particle θ^i, the controller parameters $\varphi^\star(\theta^i)$ are estimated through the VRFT approach presented in Section II (Step 2.1.3). In order to enforce stability of the reference model $M(\theta^i,q)$ and of the left inverse $M^\dagger(\theta^i,q)$ (needed to compute the virtual reference $r_v(\theta,t)$), a barrier function $b : \mathbb{R} \to \mathbb{R}$ penalizing the violation of the stability conditions:

$$\begin{aligned}
h_j^p(\theta) &:= (p_j^{c,re})^2 + (p_j^{c,im})^2 - 1 < 0, \quad j = 1,\dots,n_{pc} \\
h_j^p(\theta) &:= |p_j^r|^2 - 1 < 0, \quad j = n_{pc}+1,\dots,n_{pc}+n_{pr} \\
h_l^z(\theta) &:= (z_l^{c,re})^2 + (z_l^{c,im})^2 - 1 < 0, \quad l = 1,\dots,n_{zc} \\
h_l^z(\theta) &:= |z_l^r|^2 - 1 < 0, \quad l = n_{zc}+1,\dots,n_{zc}+n_{zr}
\end{aligned} \tag{9}$$

is added to the computation of the performance index $J(\theta^i)$ (Step 2.1.4). Among many possible choices, the following piecewise-polynomial barrier function b has been used:

$$b(h) = \begin{cases} 0 & \text{if} \quad h < 0 \\ \sqrt{k}h & \text{if} \quad 0 \le h < 1 \\ \sqrt{k}h^2 & \text{if} \quad h \ge 1 \end{cases} \tag{10}$$

where k denotes the current PSO iteration.

The global best particle θ^* is computed as the particle providing the minimum value of the performance indexes $J(\theta^i)$, $i = 1,\dots,N_{\text{part}}$ (Step 3). Finally, particles' position is updated based on common rules in PSO (Step 4). Note that, instead of using barrier functions, the new particles can be projected onto the set defined by constraints (9). The algorithm is iterated until the maximum number of iterations is reached.

C. Constraint handling

The data-driven direct controller synthesis formulation described above can be also extended to take into account constraints on input and output variables.

One way is to add bounds on $u_p(\theta,t)$ in (7) for all $t = 0,\dots,N-1$ to limit the input range. Output constraints can be enforced on $y_p(\theta,t)$ in a similar way. Clearly, only the inputs/outputs corresponding to the data set generated by $r(0),\dots,r(N-1)$ will be guaranteed to satisfy the constraints. Alternatively, the control law $C(\varphi,q)$ can be parametrized in a different way, for example using basis functions that include saturations, such as sigmoids, to enforce input constraints.

Another approach is to keep $C(\varphi,q)$ a linear discrete-time controller, motivated by the fact that for small signals, assuming the underlying process is a smooth one, the behavior of the closed-loop system can be approximated as a linear one. The idea is then to synthesize a control law that is optimal for small signals using the approach described above, and then design an MPC controller on top of the closed-loop system to handle constraints, as done in [11]. Alternatively, one can extend the synthesized linear controller $C(\varphi^\star(\theta^\star),q)$ to an MPC controller by using the approach described in [14].

IV. SIMULATION RESULTS

We show the effectiveness of the proposed approach in two examples.

Algorithm 1 Particle swarm optimization for reference model selection

Input: number of particles N_{part}, maximum number of iterations $k_{\max}$; positive weights W_y, W_{Δ_u}, W_{fit}; barrier function b.

1. **populate** particle swarm θ^i, $i = 1, \dots, N_{\text{part}}$ with random initial values under constraints (9);
2. **for** $k = 1, \dots, k_{\max}$ **do**

2.1. **for** $i = 1, \dots, N_{\text{part}}$ **do**

2.1.1. **parametrize** $M(\theta^i, q)$ as in (8);

2.1.2. **set** K_c such that $M(\theta^i, 1) = 1$;

2.1.3. **compute** $C(\varphi^\star(\theta^i), q)$, $c_0(\varphi^\star(\theta^i))$ through VRFT;

2.1.4. **set** the fitness function

$$J(\theta^i) = \frac{1}{N}\sum_{t=0}^{N-1} W_y(r(t) - y_p(\theta^i, t))^2 + W_{\Delta u}\Delta u_p^2(\theta^i, t) + W_{\text{fit}}(u(t) - u_v(\theta^i, t))^2 + \sum_{j=1}^{n_{pc}+n_{pr}} b(h_j^p(\theta^i)) + \sum_{l=1}^{n_{zc}+n_{zr}} b(h_l^z(\theta^i)) \tag{11}$$

2.2. **end for**;

3. **choose** the best particle θ^* based on the computed fitness functions $J(\theta^i)$, $i = 1, \dots, N_{\text{part}}$;
4. **update** particles' position as in [13, Algorithm 1];
5. **end for**;
6. **end**.

Output: Best particle θ^*, reference model parameters θ^*, controller parameters $\varphi^\star(\theta^*)$.

A. Example 1

We consider data generated by the linear time-invariant asymptotically stable process

$$G(q) = \frac{q - 0.4}{q^2 + 0.15\,q - 0.325}. \tag{12}$$

The transfer function $G(q)$ is supposed to be unknown for the design of the data-driven controller, it is only used for data collection and simulation. An open-loop experiment is performed to collect a dataset of $N = 5000$ input and output samples. The process is excited with a white Gaussian noise input signal $u(t) \sim \mathcal{N}(0, 1)$ and the output signal is corrupted by a white Gaussian noise disturbance $\zeta(t) \sim \mathcal{N}(0, 10^{-4})$. The sampling time is 0.1 s. The reference signal $r(t)$ used to design the controller, that is to construct the cost $J(\theta)$ in (5), is a pseudo-random binary signal taking values in $\{-1, 1\}$. The values of the weights are $W_y = 1$, $W_{\Delta u} = 1.5$, and $W_{\text{fit}} = 50$. Algorithm 1 is executed for a maximum number $k_{\max} = 100$ of iterations, with $N_{\text{part}} = 10$ particles and piecewise polynomial barrier function b as in (10). Each reference model $M(\theta^i, q)$ in Step 2.1.1 is parametrized as in (8), with $n_{zr} = 2$, $n_{pr} = 3$, and $n_{zc} = n_{pc} = 0$. For a given $M(\theta^i, q)$, a third-order linear controller $C(\varphi, q)$ is synthesized through the VRFT approach (Step 2.1.3), with feedforward term $c_0(\varphi) = 0$. The closed-loop performance achieved by the synthesized controller $C(\varphi^*(\theta^\star), q)$ are compared with the performance obtained by the optimal linear quadratic Gaussian (LQG) controller $C_{\text{LQG}}(q)$. The controller, which clearly requires the knowledge of the open-loop process model (12), is based on the extended model $G(q)\frac{q}{1-q}$ and the same weights W_y, W_{Δ_u} used for the data-driven synthesis. The performance of the designed controllers is evaluated by the following cumulated cost

$$J_{cl} = \frac{1}{\tau}\sum_{t=0}^{\tau-1}\left\{W_y(\overline{r}(t) - y(t))^2 + W_{\Delta u}\Delta u^2(t)\right\} \tag{13}$$

where $\tau = 110$, $\overline{r}(t)$ is the reference signal to be tracked, $y(t)$ is the output of (12), and $\Delta u(t) = u(t) - u(t-1)$ is the input increment of the signal $u(t)$ generated by the controller. In the closed-loop tests we consider the signal $\overline{r}(t)$, composed of steps and sine waves, depicted in Fig. 2, along with the output responses obtained when the loop is closed with the data-driven controller $C(\varphi^\star(\theta^\star), q)$ and the model-based optimal controller $C_{\text{LQG}}(q)$. The time interval $[2.4,\ 3.7]\,s$ is zoomed in Fig. 3. Fig. 4 shows the corresponding input increments. It is apparent that the two controllers achieve almost the same closed-loop behavior, which confirms that Algorithm 1 successfully synthesized an almost optimal controller directly from data and without an open-loop model. More quantitatively, the data-driven controller $C(\varphi^\star(\theta^\star), q)$ scores $J_{cl} = 0.3855$, while the model-based controller $C_{\text{LQG}}(q)$ scores $J_{cl} = 0.3805$.

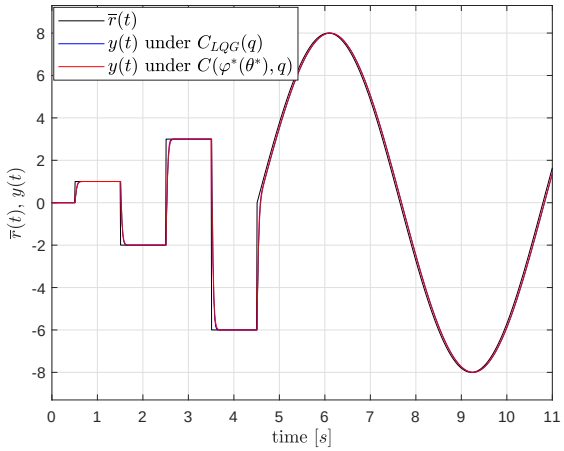

Fig. 2: Example 1. Closed-loop experiment: reference signal $\overline{r}(t)$ (black); output under data-driven controller $C(\varphi^\star(\theta^\star), q)$ (red), and the model-based optimal controller $C_{\text{LQG}}(q)$ (blue).

B. Example 2

We want to test our synthesis approach on a simple nonlinear process. To this end, we consider the nonlinear Wiener process obtained by cascading the linear transfer

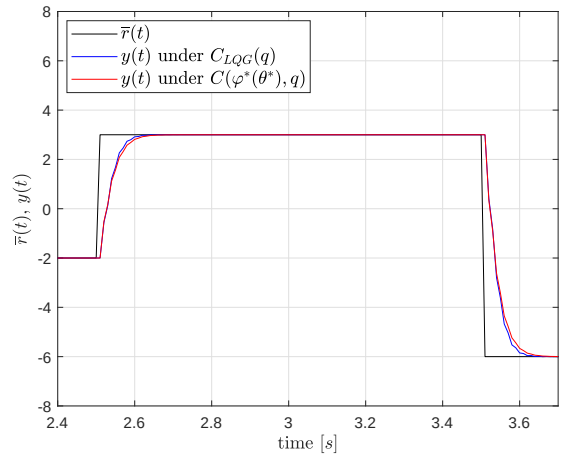

Fig. 3: Example 1. Detail of Fig. 2 in the time interval $[2.4,\ 3.7]\ s$.

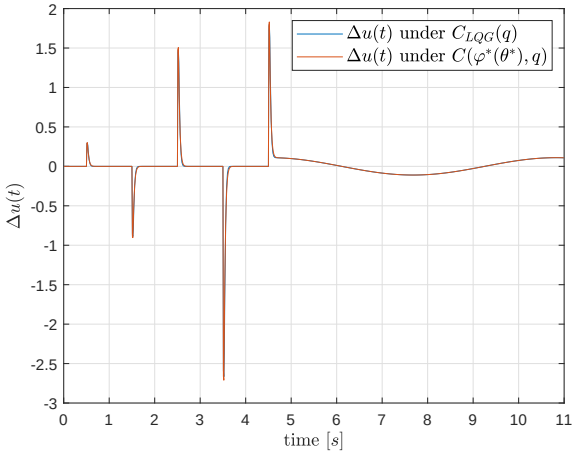

Fig. 4: Example 1. Closed-loop experiment: input increments under data-driven controller $C(\varphi^\star(\theta^\star),q)$ (red), and controller $C_{\mathrm{LQG}}(q)$ (blue).

function $G(q)$ in (12) with the static nonlinear function $f:\mathbb{R}\to\mathbb{R}$,

$$f(y_L(t)) = |y_L(t)|\ \arctan(y_L(t))$$

where $y_L(t) = G(q)u(t)$. As in Example 1, we perform an open-loop experiment by exciting the process with a white Gaussian noise $u(t)\sim\mathcal{N}(0,1)$ and collect a dataset of $N = 5000$ input/output samples. The measured output signal is $y(t) = f(y_L(t)) + \zeta(t)$, where the measurement noise $\zeta(t)\sim\mathcal{N}(0,10^{-4})$. The sampling time is 0.1 s. Algorithm 1 is executed with the same settings as in Example 1. The performance obtained by the resulting controller $C(\varphi^*(\theta^\star),q)$ is compared with the one achieved by a linear quadratic controller $C_{\mathrm{LQG}}(q)$ designed based on a linear model of the process $\hat{G}(q)$ estimated from $N_f = 3000$ samples in the available dataset. The best fit rate (BFR)

$$\mathrm{BFR} = 100\left(1-\frac{\|y-\hat{y}\|}{\|y-y_m\|}\right)\%\qquad(14)$$

of the model is 81%, where in (14) $\hat{y}$ is the output trajectory obtained by simulating $\hat{G}(q)$ in open-loop on inputs from the validation data-set of the remaining $N-N_f$ samples, y the corresponding output data-set, and y_m the average of y.

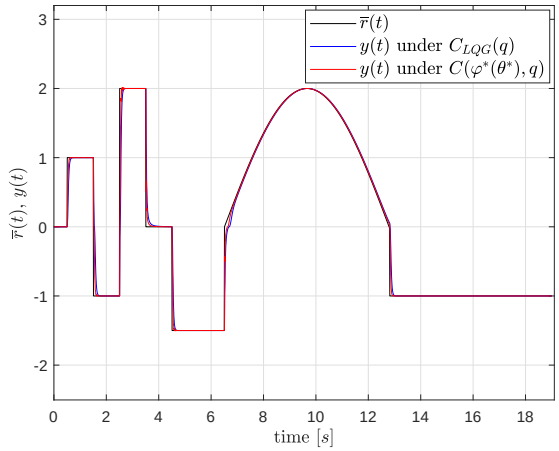

Fig. 5: Example 2. Closed-loop experiment: reference signal $\overline{r}(t)$ (black); output under data-driven controller $C(\varphi^\star(\theta^\star),q)$ (red), and under controller $C_{\mathrm{LQG}}(q)$ (blue).

As in Example 1, the reference $\overline{r}(t)$ is composed of steps and sinusoids, as shown in Fig. 5, which also displays the closed-loop output responses when $C(\varphi^*(\theta^\star),q)$ and $C_{\mathrm{LQG}}(q)$, respectively, are in feedback with the given nonlinear process. A zoom of the output signal in the interval $[0,2]$ s is provided in Fig. 6 for clarity. The input increments generated by the two controllers are shown in Fig. 7. While the closed-loop behavior of the two controllers looks quite similar, $C(\varphi^*(\theta^\star),q)$ slightly outperforms $C_{\mathrm{LQG}}(q)$, as $C(\varphi^*(\theta^\star),q)$ scores $J_{cl} = 0.02963$ while the model-based LQG controller $C_{\mathrm{LQG}}(q)$ scores $J_{cl} = 0.03909$ over the same horizon $\tau = 190$ of steps.

The weight W_{fit} in (5) is a key tuning parameter of our method. In order to analyze its effects, we consider different choices for W_{fit} between 10^{-9} and $10^{1}1$, while keeping $W_y = 1$ and $W_{\Delta u} = 1.5$ fixed. Fig. 8 shows the closed-loop performance J_{cl} (13) as a function of W_{fit}. As expected, performance deteriorates when W_{fit} is too small (the reference model is loosely matched) or too high (too little emphasis on tracking performance). In this particular example very small and large values of W_{fit} lead to a very similar value of the cumulated cost J_{cl}, although the time evolution of $y(t)$ and $\Delta u(t)$ (not reported here) are considerably different.

V. CONCLUSIONS

This paper has presented a novel approach towards direct data-driven *model-free* design of optimal controllers. We

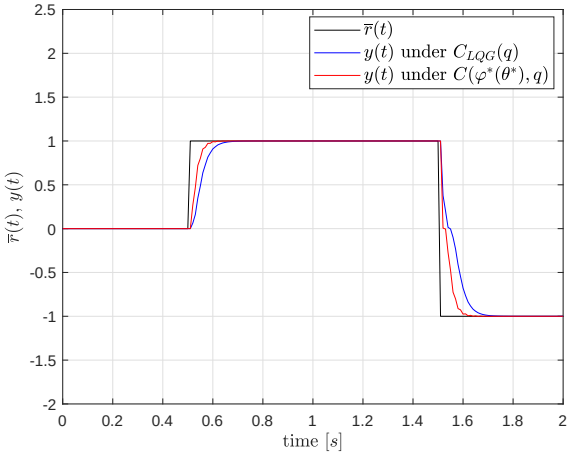

Fig. 6: Example 2. Detail of Fig. 5 in the time interval $[0,\, 2]\, s$.

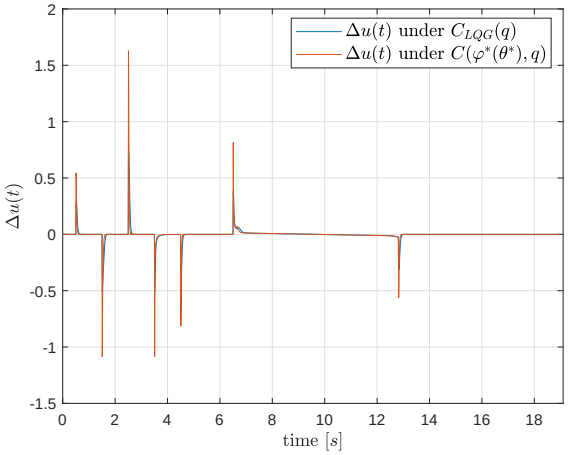

Fig. 7: Example 2. Closed-loop experiment: input increments under data-driven controller $C(\varphi^{\star}(\theta^{\star}), q)$ (red), and under controller $C_{\mathrm{LQG}}(q)$ (blue).

have described the approach in a linear time-invariant framework for simplicity of notation, but it can be immediately generalized to linear parameter-varying (LPV) structures using the LPV-VRFT method of [8] to design LPV controllers and to invert LPV reference models. Moreover, nonlinear basis functions can be used in synthesizing the controller. The approach has also the benefit of providing, as a by-product, the optimal reference model M corresponding to the synthesized controller. Hence, in the spirit of the hierarchical architecture in [11], M can employed to design a reference governor, such as a model predictive controller, in order to handle input and output constraints. Further research activities will address the issue of data-driven selection of the optimal controller structure, as well as of the inclusion of a data-based stability criterion, like the one in [10], to guarantee robust stability for handling the mismatch between

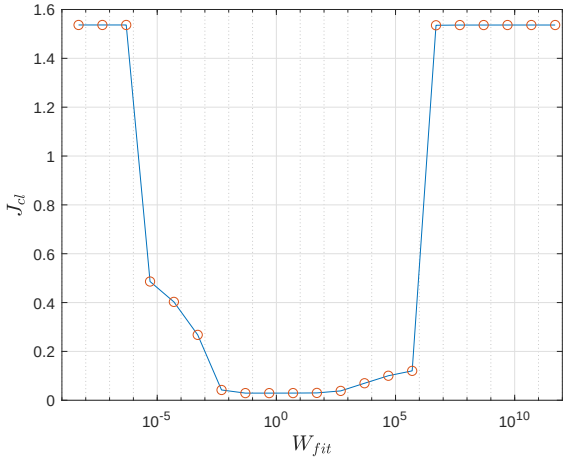

Fig. 8: Example 2. Closed-loop performance index J_{cl} vs weight W_{fit}.

the computed reference model and the actual closed-loop system.

References

[1] L. Ljung, *System identification: theory for the user*. Prentice-Hall Englewood Cliffs, NJ, 1999.

[2] M. Campi, A. Lecchini, and S. Savaresi, "Virtual reference feedback tuning: a direct method for the design of feedback controllers," *Automatica*, vol. 38, pp. 1337–1346, 2002.

[3] A. Sala and A. Esparza, "Extensions to virtual reference feedback tuning: a direct method for the design of feedback controllers," *Automatica*, vol. 41, no. 8, pp. 1473–1476, 2005.

[4] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, "Iterative feedback tuning: theory and applications," *IEEE control systems*, vol. 18, no. 4, pp. 26–41, 1998.

[5] K. van Heusden, A. Karimi, and D. Bonvin, "Data-driven model reference control with asymptotically guaranteed stability," *International Journal of Adaptive Control and Signal Processing*, vol. 25, no. 4, pp. 331–351, 2011.

[6] A. Bazanella, L. Campestrini, and D. Eckhard, *Data-Driven Controller Design: The $\mathcal{H}_2$ Approach*. Springer, 2011.

[7] S. Formentin, A. Karimi, and S. Savaresi, "Optimal input design for direct data-driven tuning of model-reference controllers," *Automatica*, vol. 49, no. 6, pp. 1874–1882, 2013.

[8] S. Formentin, D. Piga, R. Tóth, and S. M. Saveresi, "Direct learning of LPV controllers from data," *Automatica*, vol. 65, pp. 98–110, 2016.

[9] M. Tanaskovic, L. Fagiano, C. Novara, and M. Morari, "Data-driven control of nonlinear systems: An on-line direct approach," *Automatica*, vol. 75, pp. 1–10, 2017.

[10] G. Battistelli, D. Mari, D. Selvi, and P. Tesi, "Direct control design via controller unfalsification," *International Journal of Robust and Nonlinear Control*, doi: 10.1002/rnc.3778.

[11] D. Piga, S. Formentin, and A. Bemporad, "Direct data-driven control of constrained systems," *IEEE Transactions on Control Systems Technology*, doi: 10.1109/TCST.2017.2702118.

[12] A. Bemporad, N. Ricker, and J. Owen, "Model predictive control – New tools for design and evaluation," in *Proc. American Control Conference*, Boston, MA, 2004, pp. 5622–5627.

[13] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[14] S. Di Cairano and A. Bemporad, "Model predictive control tuning by controller matching," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 185–190, 2010.