Data-driven inversion-based control of nonlinear systems

Data-driven inversion-based control of nonlinear systems

Simone Formentin [*], **Carlo Novara** [**], **Sergio M. Savaresi** [*], **Mario Milanese** [***,**]

[*] *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, via G. Ponzio, 34/5 - 20133 Milano, Italy (e-mail: simone.formentin@polimi.it, sergio.savaresi@polimi.it).*
[**] *Dipartimento di Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi, 24 - 10129 Torino, Italy (e-mail: carlo.novara@polito.it).*
[***] *Modelway srl, via Livorno, 60 - 10144 Torino, Italy (e-mail: mario.milanese@modelway.it).*

Abstract: In this paper, we introduce the Data-Driven Inversion-Based Control (D^2-IBC) method for nonlinear control system design. The method relies on a two degree-of-freedom architecture, with a nonlinear controller and a linear controller running in parallel, and does not require any detailed physical knowledge of the plant to control. Specifically, we use input/output data to synthesize the control action by employing convex optimization tools only. We show the effectiveness of the proposed approach on a simulation example, where the D^2-IBC performance is also compared to that of the Direct FeedbacK (DFK) design approach, a benchmark method for nonlinear controller design from data.

Keywords: data-driven control design; nonlinear systems; convex optimization.

1. INTRODUCTION

One of the most natural ways to force the output of a nonlinear system to follow a given trajectory is to employ a feedforward controller described by the inverse of the system dynamics. When fed by the reference trajectory, the output of such a controller will correspond exactly to the desired input of the system, i.e. the input signal producing an output sequence equal to the reference one. Unfortunately, such an ideal controller is seldom computable and/or applicable in most of the practical cases. The problems can be many, e.g. the system dynamics is not invertible, there are unstable zero dynamics, the model of the plant is not an accurate description of all the underlying dynamics, the plant is affected by noises/disturbances, or other application-specific issues.

The above observations led - for some classes of systems - to the well-known *feedback linearization* approach (see Khalil (2002)), where the objective of the nonlinear controller is milder, i.e. the controller is no longer required to fully invert the system dynamics, but only to linearize it around the current operating point. Nevertheless, such an approach suffers from some critical drawbacks, too. First of all, only input affine systems can typically be dealt with, see Khalil (2002). Secondly, such an approach is still very sensitive to model errors, which may jeopardize the performance but also destabilize the system, when implemented in a real-world setup.

In the last decades, these premises have produced several research directions with the aim to overcome the limits of the above (appealing) approaches for control of nonlinear systems. Among the others, the most common activities in the field are: approximate linearization via feedback (see Guardabassi and Savaresi (2001)), feedforward linearization (see Hagenmeyer and Delaleau (2003)), robust feedback linearization (see Marino and Tomei (1996)), model predictive control (see Mayne et al. (2000)), and data-driven control (see Novara et al. (2013); Radac et al. (2013)).

Unlike the others, a *data-driven approach* does not (intrinsically) require an accurate physical parametrization of the system to control, because the controller is directly computed from experimental measurements. Such a change of perspective obviously offers a great potential against undesired modeling errors, but also offers new theoretical challenges to the systems and control community. In this field, a lot of work has already been done, ranging from the Ziegler and Nichols method (Ziegler and Nichols (1942)) to Virtual Reference Feedback Tuning (VRFT in Campi et al. (2002); Formentin et al. (2012a,b, 2013); Formentin and Karimi (2014)) until the recent data-driven loop-shaping approach (see Formentin and Karimi (2013)). However, only few contributions have focused on nonlinear systems. Among these, *neural networks* have played a major role, see e.g. Yeşildirek and Lewis (1995); Polycarpou (1996). Notwithstanding their evident general applicability, these methods are hard to implement from a practical point of view, due to the lack of criteria for optimal network selection. Since all the related optimization problems are non-convex, there is also no guarantee that the resulting controller corresponds to the optimal one.

The nonlinear version of the *VRFT method* in Campi and Savaresi (2006) represents an effective tool to make the nonlinear system behave like a desired linear reference model, using convex optimization only. Although the idea

behind the method sounds natural and appealing, the approach in the current form is not able to guarantee the stability of the closed-loop system nor a certain level of performance, when the system moves far from the input trajectory of the identification data set. Instead, with the recent *Direct Feedback approach* (DFK Novara et al. (2013)), a stabilizing controller can be computed, which can also guarantee that the norm of the tracking error is bounded, under some assumptions on the identification data set and solving only convex problems. Specifically, the DFK controller aims to be the data-driven counterpart of the model-driven inversion-based controller, but without needing any assumption on the model parametrization. However, the DFK approach may not be suitable for the control of systems described by a regression function non-invertible (non-bijective) with respect to the command input (this limitation is common to many methods, including feedback linearization). Also, DFK is based on full-state feedback and its extension to the output feedback case has not been systematically developed yet.

In this paper, we propose a novel approach called *Data-Driven Inversion-Based Control* (D^2-IBC). The method is developed within the same mathematical framework of DFK (therefore sharing the same advantages), but without the above limitations and with enhanced performance guarantees, *e.g.* since output feedback is used, this method is applicable also when the state measurements are only partially available.

More specifically, we show how D^2-IBC relies on an architecture composed by a nonlinear controller and a linear controller in parallel, allowing both compensation of nonlinearities and performance boosting. The algorithm is depicted in detail and tested on a benchmark simulation example to show its effectiveness.

The remainder of the paper is as follows. In Section 2, the control design problem is formally stated and the D^2-IBC approach is introduced. The algorithmic details are provided in Section 3, whereas the effectiveness of the strategy on a benchmark simulation example, the Duffing oscillator, is illustrated in Section 4. Some concluding remarks end the paper.

2. THE D^2-IBC APPROACH

2.1 Problem setting

Consider a nonlinear discrete-time SISO system in regression form:

$$\begin{aligned} y_{t+1} &= g\left(\boldsymbol{y}_t, \boldsymbol{u}_t, \boldsymbol{\xi}_t\right) \\ \boldsymbol{y}_t &= (y_t, \ldots, y_{t-n+1}) \\ \boldsymbol{u}_t &= (u_t, \ldots, u_{t-n+1}) \\ \boldsymbol{\xi}_t &= (\xi_t, \ldots, \xi_{t-n+1}) \end{aligned} \tag{1}$$

where $u_t \in U \subset \mathbb{R}$ is the input, $y_t \in \mathbb{R}$ is the output,$\xi_t \in \Xi \subset \mathbb{R}^{n_\xi}$ is a disturbance including both process and measurement noises, and n is the system order. U and Ξ are compact sets. In particular, $U \doteq [\underline{u}, \overline{u}]$ accounts for input saturation.

Suppose that the system (1) is unknown, but a set of measurements is available:

$$\mathcal{D} \doteq \{\tilde{u}_t, \tilde{y}_t\}_{t=1-L}^{0} \tag{2}$$

where $\tilde{u}_t$ and $\tilde{y}_t$ are bounded for all $t = 1-L, \ldots, 0$. The symbol $\sim$ is used to indicate the input and output samples of the data set (2).

Let $\mathcal{Y}^0 \subseteq \mathbb{R}^n$ be a set of initial conditions of interest for the system (1) and, for a given initial condition $\boldsymbol{y}_0 \in \mathcal{Y}^0$, let $\mathcal{Y}(\boldsymbol{y}_0) \subseteq \ell_\infty$ be a set of output sequences of interest.

The aim is to control the system (1) in such a way that, starting from any initial condition $\boldsymbol{y}_0 \in \mathcal{Y}^0$, the system output sequence $\boldsymbol{y} = (y_1, y_2, \ldots)$ tracks any reference sequence $\boldsymbol{r} = (r_1, r_2, \ldots) \in \mathcal{Y}(\boldsymbol{y}_0)$. The set of all solutions of interest is defined as $\mathcal{Y} \doteq \left\{\mathcal{Y}(\boldsymbol{y}_0) : \boldsymbol{y}_0 \in \mathcal{Y}^0\right\}$. The set of all possible disturbance sequences is defined as $\Xi \doteq \{\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots) : \xi_t \in \Xi, \forall t\}$.

To accomplish this task, we use the feedback control structure depicted in Figure 1, where S is the system (1), K^{nl} is a nonlinear controller, K^{lin} is a linear controller, $r_t \in Y$ is the reference, and $Y \subset \mathbb{R}$ is a compact set where the output sequences of interest lie.

K^{nl} is used to stabilize the system (1) around the trajectories of interest, whereas K^{lin} allows us to enhance the tracking performance. As shown in the next Sections 2.2 and 2.3, the design of these two terms is based on system inversion.

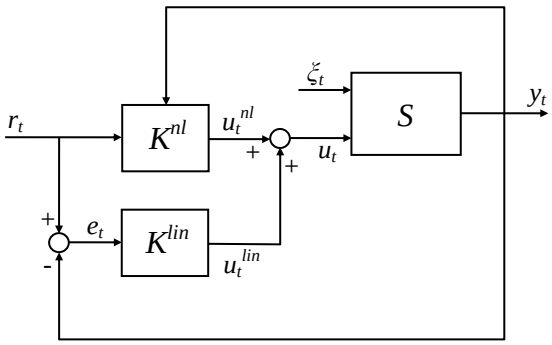

Fig. 1. Feedback control system.

Remark 1. The system (1) is not required to be stable and in general no preliminary stabilizing controllers are needed. The only guideline is to generate the data using input signals for which the system output does not diverge. This can be easily done for many nonlinear systems. Indeed, many systems are characterized by trajectories that are unstable but bounded (a typical feature of chaotic systems). In the presence of unbounded trajectories, for which a suitable input signal can hardly be found, a preliminary stabilizing controller may be required. The preliminary controller can also be a human operator, who is able to drive the system within a bounded domain, see Novara et al. (2013); Fagiano and Novara (2012). □

2.2 Nonlinear controller

The nonlinear controller design is carried out using the Nonlinear Inversion Control (NIC) method, see Novara and Milanese (2014). The first step of this method is to identify from the data (2) a model for the system (1) of the form

$$\begin{aligned} \hat{y}_{t+1} &= f\left(\boldsymbol{y}_t, \boldsymbol{u}_t\right) \equiv f\left(\boldsymbol{q}_t, u_t\right) \\ \boldsymbol{q}_t &= (y_t, \ldots, y_{t-n+1}, u_{t-1}, \ldots, u_{t-n+1}) \end{aligned} \tag{3}$$

where u_t and y_t are the system input and output, and $\hat{y}_t$ is the model output. For simplicity, the model is chosen of the same order as the system but this choice is not necessary: all the results presented in the paper hold also when the

model and system orders are different. Indications on the choice of the model order are given in Section 3.

A parametric structure is taken for the function f:

$$f(\boldsymbol{q}_t, u_t) = \sum_{i=1}^{N} \alpha_i \phi_i(\boldsymbol{q}_t, u_t) \tag{4}$$

where ϕ_i are basis functions and α_i are parameters to be identified. The basis function choice is in general a crucial step, see Sjöberg et al. (1995); Hsu et al. (2006); Novara et al. (2011). In the present approach, polynomial functions are used. The motivations are mainly two: (1) polynomials have been proved to be effective approximators in a huge number of problems; (2) as we will see later, they allow a "fast" controller evaluation. The identification of the parameter vector $\alpha \doteq (\alpha_1, \ldots, \alpha_N)$ can be performed by means of convex optimization, as shown in Section 3.1.

Once a model of the form (3) has been identified, the controller K^{nl} is obtained by its inversion, looking for a command input u_t^{nl} such that the model output at time $t+1$ is "close" to r_{t+1}. This input can be computed solving the following optimization problem:

$$u_t^{nl} = \arg\min_{\mathfrak{u} \in U} J(\mathfrak{u}) \tag{5}$$

The objective function is given by

$$J(\mathfrak{u}) = \frac{1}{\rho_y} \left(r_{t+1} - f(\boldsymbol{q}_t, \mathfrak{u})\right)^2 + \frac{\mu}{\rho_u} \mathfrak{u}^2 \tag{6}$$

where

$$\rho_y \doteq \|(\tilde{y}_{1-L}, \ldots, \tilde{y}_0)\|_2^2 / L, \ \rho_u \doteq \|(\tilde{u}_{1-L}, \ldots, \tilde{u}_0)\|_2^2 / L$$

are normalization constants computed from the data set (2), and $\mu \geq 0$ is a design parameter, allowing us to determine the trade-off between tracking precision and command activity.

Note that the objective function (6) is in general non-convex. Moreover, the optimization problem (5) has to be solved on-line, and this may require a long time compared to the sampling time used in the application of interest. In order to overcome these two relevant problems, a technique is now proposed, allowing a very efficient computation of the optimal command input u_t^{nl}.

Since a polynomial basis function expansion has been considered for f, it follows that the objective function $J(\mathfrak{u})$ is a polynomial in $\mathfrak{u}$. The minima of $J(\mathfrak{u})$ can thus be found considering the roots of its derivative: Define the set

$$U^s \doteq \left(\text{Rroots}\left(\frac{dJ(\mathfrak{u})}{d\mathfrak{u}}\right) \cap U\right) \cup \{\underline{u}, \overline{u}\}$$

where $\text{Rroots}(\cdot)$ denotes the set of all real roots of $\cdot$, and $\underline{u}$ and $\overline{u}$ are the boundaries of U. The optimal command input is given by

$$u_t^{nl} = K^{nl}(r_{t+1}, \boldsymbol{q}_t) \doteq \arg\min_{\mathfrak{u} \in U^s} J(\mathfrak{u}) \tag{7}$$

where it has been considered that U^s depends on the reference r_{t+1} and regressor $\boldsymbol{q}_t$.

The nonlinear controller K^{nl} to use in the feedback system of Figure 1 is fully defined by the control law (7). Sufficient conditions under which K^{nl} stabilizes this system are derived in Novara and Formentin (2014) for the sake of space.

Remark 2. The derivative $dJ(\mathfrak{u})/d\mathfrak{u}$ can be computed analytically. Moreover, U^s is composed by a "small" number of elements:

$$\text{card}(U^s) < \deg(J(\mathfrak{u})) + 2$$

where card is the set cardinality and deg indicates the polynomial degree. The evaluation of u_t^{nl} through (7) is thus extremely fast, since it just requires to find the real roots of a polynomial whose analytical expression is known and to compute the objective function for a "small" number of values (see Section 4 for an indicative computation time value). This fact allows a very efficient controller implementation on real-time devices. □

2.3 Linear controller

The design of the linear controller K^{lin} is based on the Virtual Reference Feedback Tuning (VRFT) method (see Campi et al. (2002)), suitably adapted for the present setting. In particular, K^{lin} is parametrized as an extended PID (Proportional Integral Derivative) controller:

$$u_t^{lin}(\theta) = u_{t-1}^{lin}(\theta) + \sum_{i=0}^{n_\theta} \theta_i e_{t-i} \tag{8}$$

where $e_t = r_t - y_t$ is the tracking error, n_θ is the controller order and the θ_i's denote the controller parameters. Note that, for $n_\theta = 1$ and $n_\theta = 2$, the standard PI and PID controller are selected, respectively. The parameters θ_i are identified from the available data by means of convex optimization. See Campi et al. (2002) for more details.

3. D^2-IBC DESIGN

3.1 Model identification and nonlinear controller design

First, a set of polynomial basis functions ϕ_i has to be chosen (see Section 2.2). Then, the following quantities are defined:

$$\tilde{\boldsymbol{y}} \doteq (\tilde{y}_{t_1+1}, \ldots, \tilde{y}_{t_2+1})$$

$$\Phi \doteq \begin{bmatrix} \phi_1(\tilde{\boldsymbol{y}}_{t_1}, \tilde{\boldsymbol{u}}_{t_1}) & \cdots & \phi_N(\tilde{\boldsymbol{y}}_{t_1}, \tilde{\boldsymbol{u}}_{t_1}) \\ \vdots & \ddots & \vdots \\ \phi_1(\tilde{\boldsymbol{y}}_{t_2}, \tilde{\boldsymbol{u}}_{t_2}) & \cdots & \phi_N(\tilde{\boldsymbol{y}}_{t_2}, \tilde{\boldsymbol{u}}_{t_2}) \end{bmatrix}$$

where $t_1 \doteq 1 - L + n$, $t_2 \doteq -1$, and $\tilde{u}_t$ and $\tilde{y}_t$ are the input-output measurements of the data set (2). Consider the set $SC \subset \mathbb{R}^N$, defined as

$$SC(\gamma, \eta, \rho) \doteq \{\beta : |\widetilde{y}_{l+1} - \widetilde{y}_{k+1} + (\boldsymbol{\Phi}_k - \boldsymbol{\Phi}_l)\beta| < \gamma\rho \|\widetilde{\boldsymbol{y}}_l - \widetilde{\boldsymbol{y}}_k\|_\infty + 2\eta\rho, k \in \mathcal{T}, l \in \Upsilon_k\}$$

where $\mathcal{T} \doteq \{t_1, \ldots, t_2\}$, $\boldsymbol{\Phi}_k$ indicates the kth row of Φ, Υ_k is the set of indexes given by

$$\Upsilon_k \doteq \{i : \|\tilde{\boldsymbol{u}}_k - \tilde{\boldsymbol{u}}_i\|_\infty \leq \zeta\}$$

and ζ is the minimum value for which every set Υ_k contains at least two elements. The set SC is defined by a set of linear inequalities in β and is thus convex in β. This set has been introduced in Novara et al. (2013) and is used in the following in order to enforce closed-loop stability.

The parameter vector $\alpha \doteq (\alpha_1, \ldots, \alpha_N)$ of the model defined by (3) and (4) can be identified by means of the following algorithm, completely based on convex optimization. Note that the algorithm is "self-tuning", in the sense that the required parameters are chosen by the algorithm itself, without requiring extensive trial and error procedures.

Another algorithm will be derived later in this section for the design of the linear controller.

Algorithm 1 (Design of K^{nl})

(1) Initialization: choose "low" model order (e.g., $n = 1$) and polynomial degree (e.g., 2); choose a precision level η_0 (e.g., $\eta_0 = 0.05 \left\|\tilde{\boldsymbol{y}}\right\|_\infty$).
(2) Construct the vector $\tilde{\boldsymbol{y}}$ and the matrix Φ as indicated above.
(3) Let $\eta \doteq \max(\eta_0, \eta_1)$, where
$$\eta_1 = \min_{\beta \in \mathbb{R}^N} \left\|\tilde{\boldsymbol{y}} - \boldsymbol{\Phi}\beta\right\|_\infty .$$
(4) Consider the optimization problem
$$\begin{array}{l} \alpha = \arg \min_{\beta \in \mathbb{R}^N} \left\|\beta\right\|_1 \\ \text{subject to} \\ (a)\ \beta \in SC(1, \eta, \rho) \\ (b)\ \left\|\tilde{\boldsymbol{y}} - \boldsymbol{\Phi}\beta\right\|_\infty \leq \eta\rho \end{array} \tag{9}$$
where ρ is a real number slightly larger than 1 (e.g., $\rho = 1 + \delta\rho$, $\delta\rho = 0.05$).
If the optimization problem (9) is feasible, solve it, return α and stop.
Else, increase the model order n and go to step 2. The order should be increased up to a maximum value n_{max}, chosen on the basis of some rough knowledge on the order of system to control (this kind of knowledge is available in most practical cases).
(5) If $n = n_{max}$ and (9) is not feasible, repeat steps 2-4 for increasing polynomial degree d. The degree should be increased up to a maximum value d_{max}. From our experience, d_{max} can be a value in the range $[4, 10]$. Note that this choice is not critical thanks to ℓ_1 norm minimization in (9), which penalizes models with a large number of basis functions.
(6) If $n = n_{max}$, $d = d_{max}$ and (9) is not feasible, repeat steps 1-5 for $\rho = 1 + 2\delta\rho, 1 + 3\delta\rho, \ldots$.

The algorithm allows the achievement of three important features:

(1) ***Closed-loop stability***. As proven in Novara et al. (2013), under reasonable conditions, constraint (a) ensures that the function $\Delta \doteq g - f$ has a Lipschitz constant γ_y non larger than 1, as $L \to \infty$. On the other hand, a theorem in Novara and Formentin (2014) shows that having this constant smaller than 1 is a key condition for closed-loop stability. The proposed approach is thus able to ensure closed-loop stability when the number of data is sufficiently large.
(2) ***"Small" tracking error***. Constraint (b) is aimed at providing a model with a "small" prediction error (this error, evaluated on the design data set, is given by $\left\|\tilde{\mathbf{y}} - \boldsymbol{\Phi}\alpha\right\|_\infty$). As shown in Novara and Formentin (2014), reducing this error allows us to obtain a "small" tracking error. Note that there is a trade-off between stability and tracking performance: In step 6, ρ is increased until the stability condition is met. However, increasing ρ causes an increase of the prediction error and, consequently, of the tracking error.
(3) ***Model sparsity***. In step 4, the ℓ_1 norm of the coefficient vector β is minimized, leading to a sparse coefficient vector α, i.e. a vector with a "small" number of non-zero elements, Fuchs (2005); Donoho et al. (2006); Candes et al. (2006); Tropp (2006). Sparsity is important to ensure a low complexity model, limiting at the same time well known issues such as over-fitting and the curse of dimensionality. Sparsity allows also an efficient implementation of the model/controller on real-time processors, which may have limited memory and computation capacities.

Once a model has been identified, the nonlinear controller K^{nl} is obtained by its inversion, as explained in Section 2.2. The weight μ in (6) can be chosen on the basis of the desired trade-off between inversion precision and command input activity.

3.2 Linear controller design

The main idea to identify parameters of the linear controller K^{lin} is to rewrite (8) as

$$u_t^{lin}(\theta) = \sum_{i=0}^{n_\theta} \theta_i \psi_t^i \tag{10}$$

where

$$\psi_t^i = \psi_{t-1}^i + e_{t-i}^v, \quad i = 0, \ldots, n_\theta. \tag{11}$$

The two forms (10) and (8) are equivalent because the only difference between them is that the integral action and the Finite-Impulse-Response (FIR) filter - built with θ and the past values of the virtual error - are switched. Since the system is linear, the filters can obviously commute without changing the dynamical properties of the controller.
By defining

$$\vartheta \doteq (\vartheta_0, \ldots, \vartheta_{n_\theta})$$

$$\Psi \doteq \begin{bmatrix} \psi_{1-L}^0 & \cdots & \psi_0^0 \\ \vdots & \ddots & \vdots \\ \psi_{1-L}^{n_\theta} & \cdots & \psi_0^{n_\theta} \end{bmatrix}$$

we introduce the following optimization problem:

$$\theta = \arg \min_{\vartheta \in \mathbb{R}^{n_\theta}} \left\| \tilde{\boldsymbol{\delta u}} - \vartheta\Psi \right\|_2^2, \tag{12}$$

which is a simple least squares problem (and therefore convex).
A linear controller of the form (8) can then be computed, given K^{nl}, according to the following algorithm.

Algorithm 2 (Design of K^{lin})

Initialization: choose the controller order n_θ and a reference model M as explained in Campi et al. (2002).
(1) Construct the vector of measurements $\tilde{\boldsymbol{\delta u}}$ based on K^{nl} and the matrix Ψ as indicated above.
(2) Compute the controller parameters θ as in (12).

The design of the linear controller is based on some user defined parameters, i.e. the parameters defining M and n_θ. Smart and simple choices for these parameters are $n_\theta = 1, 2$ (PI or PID controller) and M as a low order model with dynamics comparable to the mean time response of the model f and the same pure delay. In the simulation section, we will show the practical effectiveness of such choices.
In Novara and Formentin (2014), we provide sufficient conditions for closed-loop stability of the proposed scheme, as well as a rigorous motivation for employing such an architecture.

4. EXAMPLE: CONTROL OF THE DUFFING OSCILLATOR

The Duffing system is a second-order damped oscillator with nonlinear spring, described by the following differential equations:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\alpha_1 x_1 - \alpha_2 x_1^3 - \beta x_2 + u \\ y &= x_1 + \xi \end{aligned} \tag{13}$$

where $x = (x_1, x_2)$ is the system state (x_1 and x_2 are the oscillator position and velocity, respectively), u is the input, y is the output, and ξ is a zero-mean Gaussian noise having a noise-to-signal standard deviation ratio of 0.03. The following values of the parameters have been considered: $\alpha_1 = -1$, $\alpha_2 = 1$, $\beta = 0.2$. For these parameter values and for certain choices of the input signal, this system exhibits a chaotic behavior, and this makes control design a particularly challenging problem.

A simulation of the Duffing system (13) having duration 400 s has been performed, using the input signal $u(\tau) = 0.3\sin(\tau) + \xi^u(\tau)$, where τ here denotes the continuous time and $\xi^u(\tau)$ is a white Gaussian noise with zero mean and standard deviation 0.2. Notice that the selected input signal is persistently exciting in the sense of Ljung (1999). This is mandatory, as the design of the linear controller in D^2-IBC has been reformulated as a pure linear system identification problem, and therefore linear identifiability conditions must be satisfied.

A set of $L = 4000$ data have been collected from this simulation with a sampling period $T_s = 0.1$ s:

$$\mathcal{D} \doteq \{\tilde{u}_t, \tilde{y}_t\}_{t=-1999}^{0}$$

where $\tilde{u}_t = u(T_s t)$ are the measurements of the input and $\tilde{y}_t = u(T_s t)$ are the measurements of the output (t denotes the discrete time).

A nonlinear controller K^{nl} has been designed following the approach of Sections 2.2 and 3.1. The basis functions have been generated as products of univariate polynomials with degree 2, yielding a set of 28 functions with maximum degree 4. Then, Algorithm 1 has been run, choosing $\mu = 0.001$ and $\rho = 1.05$. The following parameter values have been produced by the algorithm: $n = 2$, $\eta = 0.18$, $\gamma_y = 0.29$. The constant Γ_y has also been estimated as shown in Section 3.1 and the value 0.05 has been obtained, indicating a good stabilizing capability. The algorithm selected 23 of the initial 28 basis functions. Note that the sparsification properties of the algorithm may be even more important in situations where the regressor is of higher dimensions, leading to problems with hundreds or thousands basis functions.

A PID linear controller K^{lin} has been designed according to the approach of Sections (2.3) and (3.2), posing $\boldsymbol{\delta\tilde{u}} = \boldsymbol{\tilde{u}} - \boldsymbol{\tilde{u}}^{nl}$, where $\boldsymbol{\tilde{u}}^{nl}$ is the output sequence provided by the controller K^{nl}. The reference model M as been chosen as a first order system with a unitary steady-state gain and a bandwidth of 10 rad/s.

The D^2-IBC control scheme of Figure 1 has then been implemented, where S is the system (13), K^{nl} and K^{lin} are the designed controllers, and ξ_t is a Gaussian noise affecting the output measurements, having zero-mean and a noise-to-signal standard deviation ratio of 0.03. A testing simulation of the control system with duration 800 s has been performed, using zero initial conditions and a reference signal r_t generated as a sequence of random steps, filtered by a second-order filter with a cutoff frequency of 2 rad/s (this filter has been inserted in order to ensure not too abrupt variations). In Figure 2, the output of the D^2-IBC control system is compared to the reference.

	D^2-IBC	DFK
$\overline{RMS}$	0.015	0.021

Table 1. Average RMS tracking errors.

Then, a Monte Carlo (MC) simulation has been carried out, where this data-generation-control-design-and-testing procedure has been repeated 100 times. For each trial, the tracking performance has been evaluated by means of the Root Mean Square tracking error

$$RMS \doteq \sqrt{\frac{1}{8000}\sum_{t=1}^{8000}(r_t - y_t)^2}.$$

The average $\overline{RMS}$ error obtained in the MC simulation is reported in Table 1.

For comparison, a similar MC simulation has been performed using the DFK control design approach of Novara et al. (2013). The average $\overline{RMS}$ tracking error obtained by the DFK method in the MC simulation is also reported in Table 1.

For both the D^2-IBC and DFK approaches, a simulation of the closed-loop system has been performed where $r_t = 0$, $\forall t$ and ξ_t is a step disturbance of amplitude 0.5. The output signals obtained in these simulations are shown in Figure 3.

From these results, it can be concluded that both the D^2-IBC and DFK controllers are able to (1) ensure a very accurate tracking, even in the presence of quite significant measurement noises; (2) reject/attenuate strong step disturbances. Thanks to the presence of the linear controller, the D^2-IBC controllers ensure a zero steady-state tracking error for step references (with null noise) and full asymptotic rejection of step disturbances. Similar features are not guaranteed by the DFK controllers, which anyway yields quite satisfactory results. Note also that an output feedback structure is used for the D^2-IBC controllers, whereas the DFK controllers uses a full-state feedback scheme.

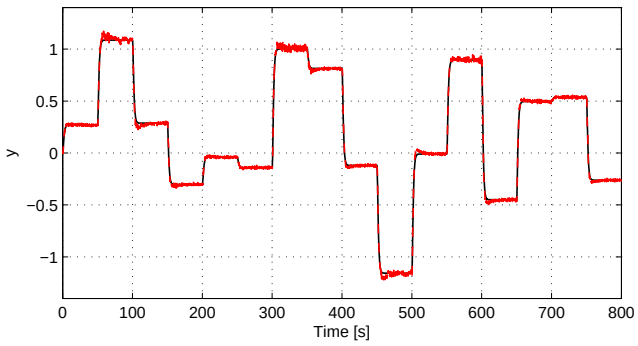

Fig. 2. Tracking performance of a D^2-IBC control system. Continuous (black) line: reference. Dashed (red) line: actual output.

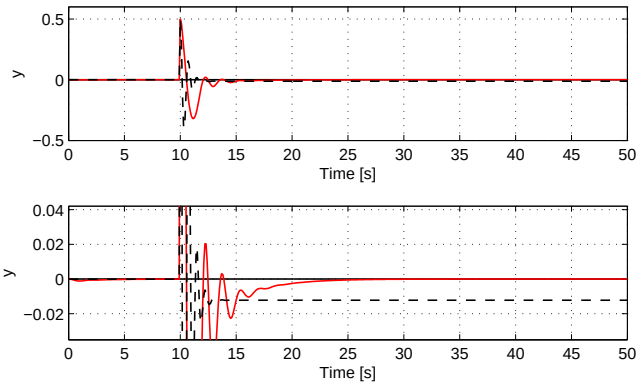

Fig. 3. Above: disturbance rejection of a D^2-IBC (continuous-red) controller and a DFK (dashed-black) controller. Below: same figure, with zoomed y axis.

5. CONCLUSIONS

In this paper, we introduced and analyzed the D^2-IBC approach for control of nonlinear systems. In such an approach, only convex optimization problems need to be solved to compute the controllers, the final laws are easy to implement, due to the sparsity property, and a large class of nonlinear systems can be dealt with. The effectiveness of the approach has been shown on a benchmark simulation example.

Future work will be devoted to the multivariable extension of the D^2-IBC method and to some real-world test applications.

REFERENCES

M.C. Campi and S.M. Savaresi. Direct nonlinear control design: The virtual reference feedback tuning (VRFT) approach. *IEEE Transactions on Automatic Control*, 51 (1):14–27, 2006.

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

E.J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. on Information Theory*, 52(2):489 – 509, feb. 2006.

D.L. Donoho, M. Elad, and V.N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Trans. on Information Theory*, 52(1):6 – 18, jan. 2006.

L. Fagiano and C. Novara. Learning controllers from data: control of a power kite used for wind energy conversion. *http://lorenzofagiano.altervista.org/movies/LC_kite_LD.mp4*, 2012.

S. Formentin and A. Karimi. A data-driven approach to mixed-sensitivity control with application to an active suspension system. *IEEE Transactions on Industrial Informatics*, 9(4):2293–2300, 2013.

S. Formentin and A. Karimi. Enhancing statistical performance of data-driven controller tuning via L2-regularization. *Automatica*, 50(5):1514–1520, 2014.

S. Formentin, M. Corno, S.M. Savaresi, and L. Del Re. Direct data-driven control of linear time-delay systems. *Asian Journal of Control*, 14(3):652–663, 2012a.

S. Formentin, S.M. Savaresi, and L. Del Re. Non-iterative direct data-driven controller tuning for multivariable systems: theory and application. *IET Control Theory & Applications*, 6(9):1250–1257, 2012b.

S. Formentin, A. Karimi, and S.M. Savaresi. Optimal input design for direct data-driven tuning of model-reference controllers. *Automatica*, 49(6):1874–1882, 2013.

J.J. Fuchs. Recovery of exact sparse representations in the presence of bounded noise. *IEEE Transactions on Information Theory*, 51(10):3601 –3608, oct. 2005. ISSN 0018-9448. doi: 10.1109/TIT.2005.855614.

G. Guardabassi and S.M. Savaresi. Approximate linearization via feedbackan overview. *Automatica*, 37(1):1–15, 2001.

V. Hagenmeyer and E. Delaleau. Exact feedforward linearization based on differential flatness. *International Journal of Control*, 76(6):537–556, 2003.

K. Hsu, C. Novara, T. Vincent, M. Milanese, and K. Poolla. Parametric and nonparametric curve fitting. *Automatica*, 42/11:1869–1873, 2006.

H. Khalil. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, 2002.

L. Ljung. *System identification: theory for the user*. Prentice Hall, Upper Saddle River, N.J., 1999.

R. Marino and P. Tomei. *Nonlinear control design: geometric, adaptive and robust*. Prentice Hall International (UK) Ltd., 1996.

D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

C. Novara and S. Formentin. Data-driven controller design for nonlinear systems: a two degrees of freedom architecture. *arXiv*, 2014.

C. Novara and M. Milanese. Control of nonlinear systems: a model inversion approach. *arXiv*, 2014.

C. Novara, T. Vincent, K. Hsu, M. Milanese, and K. Poolla. Parametric identification of structured nonlinear systems. *Automatica*, 47(4):711 – 721, 2011. ISSN 0005-1098. doi: 10.1016/j.automatica.2011.01.063.

C. Novara, L. Fagiano, and M. Milanese. Direct feedback control design for nonlinear systems. *Automatica*, 49(4): 849–860, 2013. doi: 10.1016/j.automatica.2013.01.002.

M.M. Polycarpou. Stable adaptive neural control scheme for nonlinear systems. *IEEE Transactions on Automatic Control*, 41(3):447–451, 1996.

M.B. Radac, R.E. Precup, E.M. Petriu, S. Preitl, and C.A. Dragos. Data-driven reference trajectory tracking algorithm and experimental validation. *IEEE Transactions on Industrial Informatics*, 9(4):2327–2336, 2013.

J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B.Delyon, P. Glorennec, H. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31:1691–1723, 1995.

J.A. Tropp. Just relax: convex programming methods for identifying sparse signals in noise. *IEEE Transactions on Information Theory*, 52(3):1030 –1051, mar. 2006. ISSN 0018-9448. doi: 10.1109/TIT.2005.864420.

A. Yeşildirek and F.L. Lewis. Feedback linearization using neural networks. *Automatica*, 31(11):1659–1664, 1995.

J.G. Ziegler and N.B. Nichols. Optimum settings for automatic controllers. *trans. ASME*, 64(11), 1942.