IFAC Journal of Systems and Control 35 (2026) 100365

Contents lists available at ScienceDirect

IFAC Journal of Systems and Control

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

Full length article

On continuous-time sparse identification of nonlinear polynomial systems☆

Mazen Alamir

Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France

ARTICLE INFO

Article history:
Received 13 October 2025
Received in revised form 29 December 2025
Accepted 12 January 2026
Available online 15 January 2026

Keywords:
Nonlinear
Continuous-time
Sparse identification
Chain of integrators

ABSTRACT

This paper leverages recent advances in high derivatives reconstruction from noisy-time series and sparse multivariate polynomial identification in order to improve the process of parsimoniously identifying, from a small amount of data, unknown Single-Input/Single-Output nonlinear dynamics of relative degree up to 4. The methodology is illustrated on the Electronic Throttle Controlled automotive system.

1. Introduction

In this paper we are interested in continuous-time identification of unknown Single-Input/Single-Output (SISO) polynomial systems of relative degree $n \in \{1, \dots, 4\}$, namely:

$$y^{(n)} = \mathcal{P}(y, \dot{y}, \dots, y^{(n-1)}, u) \tag{1}$$

where $\mathcal{P}$ is a multivariate polynomial of unknown degree d. Moreover, since the *safe* excitation of unknown systems remains challenging, we are interested in a methodology that enables an accurate identification using small datasets involving a family of short damped excitation signals that are known to be possible to safely apply to the system, initially at rest.

While the restriction applies to the notation and the specific illustrative example, the methodology developed hereafter easily extends to more general MIMO polynomial systems of the form:

For $i \in \{1, \dots, n_y\}$,

$$y_i^{(n_i)} = \mathcal{P}_i\Big(\big\{\{y_j^{(\kappa)}\}_{\kappa=0}^{n_j^{(i)}-1}\big\}_{j=1}^{n_y}, \{u_\ell\}_{\ell=1}^{n_u}\Big) =: \mathcal{P}_i(z_i) \tag{2}$$

using the same tools that are shown hereafter to be efficient in addressing the identification of (1), namely:

- `Tool1`: The recently proposed algorithm[1] for the computation of high derivatives (up to order 4) of noisy time-series (Alamir, 2025b),
- `Tool2`: The scalable `plars` algorithm which addresses the problem of identifying multivariate polynomial relationships (Alamir, 2025c).

Indeed, assuming that for a given n_i, the l.h.s $y_i^{(n_i)}$ of (2) is available (`Tool1`), the problem boils down to identifying the multivariate polynomial relationships (2) that link the `label` $y_i^{(n_i)}$ to the `featuresThe`[2] vector z_i (`Tool2`).

The price associated with the generalization to MIMO systems lies in the dimension of z_i, namely[3]:

$$\texttt{dim}(z_i) = n_u + \sum_{j\in\{1,\dots,n_y\}} n_j^{(i)}$$

as this leads to a multivariate polynomial with a large number of coefficients. For instance, a **9-dimensional** nonlinear system with two inputs and three outputs of relative degree[4] 3 each, might be associated to the following instantiation:

$$\Big\{n_u = 2, n_y = 3, n_j^{(i)} = 3\Big\} \longrightarrow \texttt{dim}(z_i) = 11$$

This leads to 78 coefficients for a polynomial of degree $d = 2$, 1365 coefficients for $d = 4$ and 12,376 coefficients for $d = 6$.

Surprisingly enough, as is shown in Alamir (2025c), such problems can be efficiently solved using standard tools such as the `lassolars` module of the scikit-learn python library (Pedregosa

☆ This article is part of a Special issue entitled: 'IFAC WC 2026 - TC1.1' published in IFAC Journal of Systems and Control.

E-mail address: mazen.alamir@grenoble-inp.fr.

1 This refers to the `python` module `ml-derivatives` that is available through standard `python-pip-install`. see https://mazenalamir.github.io/ml_derivatives/ for more details.

2 couple `(label, features)` borrows the well-known Machine Learning vocabulary where one seeks a map F such that `label=F(features)`.

3 Notice that $n_j^{(i)} = n_i - 1$ for all i by definition.

4 The relative degree of an output is the number of derivation that are required to see at least one of the components of the input explicitly in the expression of the derivative.

https://doi.org/10.1016/j.ifacsc.2026.100365

et al., 2011) in a few minutes if not seconds. Much larger problems involving up to half a million variables can be solved using `plars` (Alamir, 2025c) in a few minutes, suggesting that this is not a real issue.

This being said, for the sake of simplicity of notation, we focus on SISO models given by (1) in the present paper.

The reason we are interested in continuous-time models is threefold.

(1) It enables to totally decouple the identification of the dynamics from the choice of the sampling period for the control design.
(2) More importantly, the degree of the r.h.s of the continuous-time dynamics is lower than that of the discrete-time one-step ahead version, as the latter is the result of multiple composition of the former as is the case in standard Runge–Kutta method for instance. The direct consequence of this fact is that it is an easier task to identify the ODEs than the discrete-time law, provided that the reconstruction of high derivatives is possible despite of the noisy measurement (`Tool1`).
(3) Identifying the nonlinear system in the derivatives space corresponds to a nonlinear coordinates transformation that might, in some cases, approximately admits a linear representation enabling simpler design of the control law. This is because the implicit coordinates change involves the r.h.s of the ODE's. This features is precisely exhibited in the illustrative example used in the present paper.

It is precisely because decent reconstruction of high derivatives has always been viewed as *almost-impossible* that CT-identification was systematically considered as a tricky task (Paduart et al., 2010; Rao & Unbehauen, 2006).

In the next section, a discussion regarding possible alternatives to address the above-defined problem is briefly provided

1.1. State of the art

A relevant way to cluster the set of possible approaches is to examine the mathematical structure that is used to represent the nonlinearity:

- The **Hammerstein** structure (Greblicki, 2002; Jui & Ahmad, 2021) has been widely used in the past where a nonlinear transformation of the control is fed to a linear dynamics. This results in a nonlinear structure that is affine in the state and can be nonlinear only in the control input.

- The **Wiener** structure (Sadeghi & Farrokhi, 2018) where nonlinearity applies to the output of a linear dynamics, the nonlinearity is affine in the control and all the state's components except the output itself which is again a very specific structure.

- Naturally, combined **Wiener–Hammerstein** structures have been studied in Paduart et al. (2012) with polynomial nonlinear part. This leads to the nonlinearity affecting only input and output and not the internal state or, equivalently, the other derivatives of the output as in (1).

- Comprehensively, the recent burst of **Deep NN** enhanced some works that addressed the identification of the two above structures using DNN to express the localized nonlinearity but this does not fundamentally change the properties of the nonlinearity being exclusively concentrated on the input and the output respectively.

- **Fully-polynomial state space models** have been studied in Paduart et al. (2010) and some related papers. The algorithm in Paduart et al. (2010) involves two main steps. In the first a *best* linear model is first identified. This model is then *corrected* by adding virtual inputs that involve a set of multivariate monomials in the state and the control input. The identification of the nonlinear added polynomial expressions is based on the minimization of the norm of the output prediction error which involves the simulation of the nonlinear system for all candidate set of parameters involved in the iterations. Since neither the linear model nor the successive nonlinear iterates are guaranteed to show stable dynamics, this might lead to a fragile optimization process.

This difficulty is shared by all the algorithms that minimize **simulation-based** cost functions. Another difficulties stems from the **non parsimonious** approach being used which might induce the well known *over-fitting* problem unless a large dataset is used which contradicts the problem statement as defined above.

In Biagetti et al. (2019), a Machine Learning approach is proposed that involves the **Particle Bernstein Polynomials** (**PBP**). The output is viewed as a nonlinear (PBP) function of the components of the input trajectory on some modal basis. The nonlinear function is identified via standard ML regression technique after Principal Component Analysis (**PCA**)-based dimensionality reduction. While the approach is discrete-time oriented it can be adapted to continuous-time setting by considering the label to be the higher derivatives of the measured outputs. Notice however that the modal approach needs extensive dataset.

Solution involving **Reinforcement learning (RL)** can be imagined where the nonlinear map in (1) would be identified via standard RL episodes. In each episode, a virtual chain of integrators is **simulated** using the current iterate of the DNN representing the r.h.s of (1) and the reward is computed based on the output prediction error. Notice however that being a simulation-based approach, this solution would share the same stability-related issue discussed above leaving aside the non sparse character that would require an important amount of learning data in order to avoid overfitting on the small learning dataset that is mentioned in the problem's statement.

Much more closer to the present work, the use of time derivatives to perform continuous-time **sparse identification** of nonlinear systems has already been proposed in Brunton and Kutz (2022) and the related works among which, a python package (de Silva et al., 2020) is made available. However the framework is based on **first derivatives** of all the state components that are supposed to be fully measured. This assumption *works around* the reconstruction of higher derivatives at the price of a strongly restrictive, if not unrealistic, assumptions.

This quick overview only sketches how vast this topic is and suggests that there is no free-lunch like solution to the problem that outperforms the others in all circumstances. The proposition made in the present contribution adds a slightly different step which builds on recent significant advances in related topics and tools.

This paper is organized as follows: First, some recalls and definitions are proposed in Section 2. Section 3 presents the identification methodology that is illustrated in Section 4 using the automotive ETC as a supporting example. Finally, Section 5 concludes the paper and gives hints for further investigation.

2. Some definitions and recalls

First of all, since this contribution is about multivariate polynomials, some related recalls and notation are necessary.

A multivariate polynomial in $z \in \mathbb{R}^{n_z}$ takes the form:

$$\mathcal{P}(z) = \sum_{i=1}^{n_m} c_i \phi_i(z) \text{ where } \phi_i(z) = \prod_{j=1}^{n_z} z_j^{p_{ij}} \tag{3}$$

where ϕ_i is referred to as the ith monomial of $\mathcal{P}$. The integer n_m refers to the number of monomials used in $\mathcal{P}$. Consequently, a polynomial $\mathcal{P}$ is totally defined by the pair (P, c):

$$P := \left\{ p_{ij} \right\} \in \mathbb{N}^{n_m \times n_z} \quad , \quad c \in \mathbb{R}^{n_m} \tag{4}$$

representing respectively the matrix of monomial powers and the associated coefficients.

The degree d_i of a monomial ϕ_i is defined by $d_i = \sum_{j=1}^{n_z} p_{ij}$. The degree of the polynomial $\mathcal{P}$ is the maximum degree of its monomials with non vanishing coefficients c_i, namely $d = \max_{i=1}^{n_m} \{ d_i \mid c_i \neq 0\}$. Given the dimension n_z of z and the degree d of the polynomial, the number n_m of candidate monomials is given by[5]:

$$n_m = \binom{n_z + d}{d} \tag{5}$$

The framework proposed in this paper is based on the following assumptions:

∃ Safe excitation inputs from rest

Assumption 1. It is assumed that there is a set of input profiles, denoted by $\mathbb{U}_{\text{safe}} \in \mathbb{R}^N$ such that when $\mathbf{u} \in \mathbb{U}_{\text{safe}}$ is applied to Eq. (1), initially at rest, no instability occurs so that the resulting measurement can be *safely* collected for use in the identification.

Notice that $\mathbb{U}_{\text{safe}}$ does not necessarily match the conditions required by standard (linear, dense) identification theory in terms of bandwidth. This is where sparse identification becomes mandatory. Notice that in many cases, the *safety* character of the excitation lies in the fact that the excitation is strongly damped (by design) and that the experiment is stopped as soon as the output (or an estimation of its derivatives) goes beyond some prefixed admissible region. This leads to experiences that do not last the same amount of time.

The next assumption concerns the availability of high-deri vatives estimation algorithm:

∃ an algorithm for Higher derivatives reconstruction

Assumption 2. There exists a self-tuning algorithm $\mathcal{A}_{\text{der}}$, such that for all $n \in \{0, 1, 2, 3, 4\}$ and all time-series $\mathbf{v} \in \mathbb{R}^N$ representing τ-sampled successive values of a smooth signal, the call:

$$(\hat{y}^{(n)}, \sigma^{(n)}) \leftarrow \mathcal{A}_{\text{der}}(\mathbf{v}, n, \tau) \tag{6}$$

provides an estimation $\hat{y}^{(n)} \in \mathbb{R}^N$ of the n-th derivative of the time-series together with a sequence of confidence indicators $\sigma^{(n)} \in \mathbb{R}^N$.

As a matter of fact, such an algorithm ($\mathcal{A}_{\text{der}}$) has been recently made freely available through the release of the `ml-derivatives` Python package which implements the framework proposed in Alamir (2025b). Notice that the commonly encountered tricky to determine trade-of between the level of noise and the filtering order of the reconstruction is automatically handled by the algorithm so that only the input arguments $(\mathbf{v}, n, \tau)$ invoked in have to be provided. The limitation to $n \leq 4$ is simply linked to the current version of the implementation and will be shortly extended to higher derivation orders.

The last assumption concerns the algorithm performing sparse identification of multi-variate polynomials:

∃ algorithm for Sparse polynomial identification

Assumption 3. There exists an algorithm $\mathcal{A}_{\text{id}}$ such that, given a features matrix $Z \in \mathbb{R}^{n_s \times n_z}$ involving n_s samples and a corresponding vector of targeted label ℓ, the call:

$$\mathcal{P} := (P, c) \leftarrow \mathcal{A}_{\text{id}}(Z, \ell, \varepsilon) \tag{7}$$

provides the multivariate polynomial $\mathcal{P} := (P, c)$ which is the sparse solution to the least squares problem:

$$\min_{\mathcal{P}} \sum_{j=1}^{n_s} \left\| \ell_j - \mathcal{P}(z_j) \right\|^2 \tag{8}$$

where ℓ_j and z_j stand for the j-th row of ℓ and Z respectively. The optional parameters ε might be used as a threshold defining the relevance of adding more monomials to the current solution.

Such an algorithm $\mathcal{A}_{\text{id}}$ is used hereafter that is proposed in (Alamir (2025a), chapter 10) although the well known `lasso-Lars` of scikit-learn (Pedregosa et al., 2011) could have been used as well provided that the number of candidate monomials defined by (5) remains moderate.

The last definitions we need concerns the metrics of error profiles which are used hereafter in comparing different solutions leading to different error's profiles. More precisely, an error profile $\mathbf{e} \in \mathbb{R}^N_+$ representing the profile of reconstruction error of some time-series $\mathbf{v}$, can be summarized by the following normalized percentiles:

$$p_q\big(\mathbf{e} := |\mathbf{v} - \hat{\mathbf{v}}|\big) := \frac{\texttt{percentile}(|\mathbf{e}|, q)}{\epsilon + \texttt{median}(|\mathbf{v}|)} \tag{9}$$

Given a set of r solutions and their associated error's profiles $\{\mathbf{e}^{(\ell)}\}_{\ell=1}^{r}$, denoting by $p_q^{(\ell)}$ the associated percentiles defined by (9), the selected solution is defined by the formulae:

$$\ell_\star := \arg\min_{\ell} \left\{ p_{95}^{(\ell)} \quad | \quad p_{100}^{(\ell)} \leq \eta \times \left[\min_{\sigma} p_{100}^{(\sigma)} \right] \right\} \tag{10}$$

where $\eta > 1$ is a predefined parameter. For instance, taking $\eta = 1.2$ induces a criterion according to which, the *best* solution is the one that shows the minimum 95-percentiles among all solutions that are not more than 20% worse than the best solution when considering the maximum value of the error.

3. The methodology

The methodology can be described via the following steps:

(1) Generate the datasets. In this step training, validation and test datasets are built using the presumably known excitation protocol with different sets of excitation parameters (see next section).
(2) Generate solutions. Since the degree of the system is not known, one needs to generate solutions for all possible values of n [see (1)] ranging from 1 to 4. For each order ℓ, a solution is generated using the training dataset and the associated error profile $\mathbf{e}^{(\ell)}$ is computed on the validation set.
(3) Choose the best solution. Using (10), the best derivation order $n = \ell_\star$ is selected and the associated $p_q(\mathbf{e}^{(n)})$ is computed on the test datasets in order to assess the generalization power of the identified model.
(4) Test feedback design. In case the results on the test dataset are satisfactory, one disposes of a control-oriented model of the form:

$$y^{(n)} = \mathcal{P}(y^{(0)}, \ldots, y^{(n-1)}, u) \tag{11}$$

[5] This can be computed using the `python, math.comb` module.

Table 1
ETC-system's parameters values.

Parameter	Value	Parameter	Value
$p_1 = N_m$	4	$p_7 = K_t$	0.1045
$p_2 = J_m$	0.0004	$p_8 = R_p$	0.0015
$p_3 = J_g$	0.005	$p_9 = R_{af}$	0.002
$p_4 = b_m$	0.03	$p_{10} = L_a$	0.003
$p_5 = b_t$	3.4×10^{-3}	$p_{11} = K_b$	0.1051
$p_6 = K_{sp}$	0.4316	$p_{12} = R_a$	1.9
K_g	100		

for which a candidate dynamic observer might be given by:

$$\hat{y}^{(n)} = \mathcal{P}(\hat{y}^{(0)}, \dots, \hat{y}^{(n-1)}, u) + L(y - \hat{y}^{(0)}) \tag{12}$$

where the observer gain matrix L is computed for the linear system represented by the chain of n integrators with known leading term. Using the so estimated quantities, any control design is eligible to yield a dynamic output feedback of the form:

$$u = K(\hat{y}^{(0)}, \dots, \hat{y}^{(n-1)}, y_{\text{ref}}) \tag{13}$$

Notice that the present paper is mainly concerned with the first three steps leading to the identification of the continuous-time model of the dynamic system. In particular, constrained nonlinear model predictive control can be designed for the control task while a moving horizon state estimator, high gain or sliding-mode nonlinear state estimators can be implemented. Nevertheless and for the sake of completeness, a concrete instantiation is provided for the illustrative example discussed in the next section.

4. Experiments

4.1. The unknown system to be identified

In order to validate the framework proposed in the previous section, let us consider the automotive Electronic Throttle Control (**ETC**) system given by Conatser et al. (2004):

$$\dot{x}_1 = x_2 \tag{14a}$$

$$\dot{x}_2 = \frac{1}{N_m^2 J_m + J_g}\Big[\phi(x,p) + N_m K_t x_3\Big] \tag{14b}$$

$$\dot{x}_3 = \frac{1}{L_a}\Big[-N_m K_b x_2 - R_a x_3 + K_g u\Big] \tag{14c}$$

where $x := (\theta, \dot{\theta}, e_a)$, in which θ stands for the angle of the admission device while e_a is the electromotor torque induced by the current $u = i_a$ serving as the control input. The vector $p = (N_m, J_m, J_g, \dots)$ gathers all the parameters involved in (14) (see Table 1 for the values).

The nonlinear map $\phi(x,p)$ appearing in (14b) is given by:

$$\phi(x,p) := -K_{sp}(x_1 - \pi/2) - (N_m^2 b_m + b_t)x_2 - \\ - 2P_{\text{atm}}(\pi - x_1)R_p^2 R_{af}\cos^2(x_1), \tag{15}$$

The measured quantities are the angular position $y = \theta$ and the control input u.

4.2. The safe excitation input profiles

The following parametrized set of damped excitation signals is considered for the generation of the datasets:

$$u(t) = \texttt{Sat}_{-\bar{u}}^{+\bar{u}}\Big(e^{-\mu t} \times \sum_{i=1}^{n_B} \alpha_i \sin(\omega_i t + \psi_i)\Big) \tag{16}$$

in which (α_i, ψ_i) are randomly generated in $[-\bar{u}, +\bar{u}] \times [0, 2\pi]$, $\omega_i = i \times \omega_{\max}/n_B$ with $\omega_{\max} = 10$, $\mu = 4$, $\bar{u} = 5$ and $n_B = 20$. Typical randomly generated profiles are shown in Fig. 1.

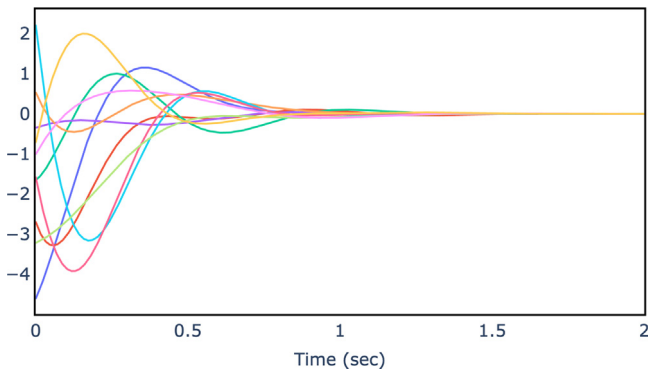

Fig. 1. Typical randomly drawn u-excitation profiles according to (16).

Remark 1. As the underlying assumption is that the system's equations are totally unknown, it is impossible to derive a theoretical design of the excitation signals and their bandwidth. Nevertheless, the response time and hence the bandwidth of the system is generally known for the system's designers or practitioners. This enables to determine the maximum pulsation ω_{max} that is used in (16). Moreover, it is possible to monitor whether the gathered information is *sufficient* by observing the difference between the identified model before and after adding a supplementary experience. As the identification is sparse, the variation of the identified model should progressively decrease.

4.3. Building working datasets

A set of 100 damped excitation scenarios such as the ones depicted in Fig. 1 have been generated and the resulting measurement profiles collected to constitute the working dataset. Only 25 of these scenarios have been used in the training step while the remaining 75 have been used in the test. With a sampling period of $\tau = 0.003$, the complete dataset involved 100,000 rows (samples) among which, 25,000 samples are used in training while 75,000 are used to check the quality of the model on unseen test data.

A noise-to-signal ratio of 3% has been applied to the measurement profiles leading to the typical noisy measurement profiles shown in Fig. 2. More precisely, the measurement y_m fed to the algorithm is obtained from the original noise-free measurement y using:

$$y_m := y + \bar{\eta} \times \texttt{percentile}(abs(y), 99) \times \nu \tag{17}$$

where $\bar{\eta} = 0.03$ while ν is a normalized unitary white noise.

4.4. Fitting polynomial models

Tables 2 and 3 show the fitting errors (defined by (9) in which $\epsilon = 10^{-12}$ is used) respectively when using noise-free or noisy measurements. Each table shows the error statistics for different model's order involved in (1). These statistics clearly show that the third order model $n = 3$ is the only appropriate ones, which is expected from our knowledge of the hidden ODEs generating the data.

The results also show the rather nice robustness of the approach to the presence of decently high level of noise (see Fig. 2) except for a significant increase of the maximum absolute error which nevertheless remains lower that 80% of the median value of the third derivative's amplitude. This is obviously a sort of incompressible instant-wise error due to the presence of measurement noise and does not necessarily mean that the model is

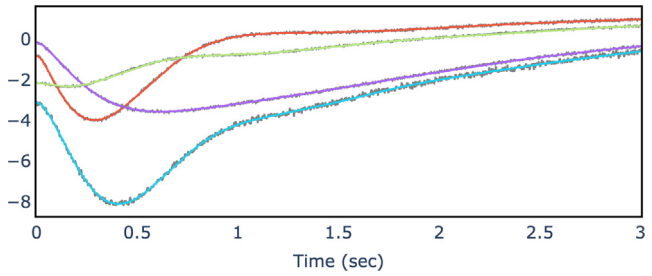

Fig. 2. Typical noisy measurements used in the training dataset to fit the sparse polynomial models. The plot shows the measured outputs corresponding to different randomly sampled excitation scenarios.

Table 2
Noise-free measurements. Normalized percentiles of the absolute identification errors on the train data for different candidate model's orders $n \in \{1, 2, 3, 4\}$. $d = 3$.

q	n=1	n=2	n=3	n=4
50%	0.36	1.12	**0.01**	0.25
80%	0.92	1.77	**0.01**	0.56
90%	1.75	2.81	**0.03**	0.88
95%	3.57	4.82	**0.05**	1.54
98%	7.31	8.03	**0.08**	2.63
99%	9.28	10.96	**0.10**	3.30
100%	18.69	26.92	**0.17**	7.76

Table 3
Noisy measurements. Normalized percentiles of the absolute identification errors on the train data for different candidate model's orders $n \in \{1, 2, 3, 4\}$, $d = 3$.

q	n=1	n=2	n=3	n=4
50%	0.45	1.32	**0.01**	0.49
80%	1.11	2.01	**0.04**	1.04
90%	1.89	3.26	**0.06**	1.79
95%	3.58	5.52	**0.08**	2.75
98%	7.34	8.99	**0.11**	4.78
99%	9.21	11.78	**0.12**	6.16
100%	18.41	28.33	**0.82**	37.01

Table 4
Noisy measurements. Normalized percentiles of errors on **test dataset** using the model's order $n = 3$, $d = 3$.

q	50%	80%	90%	95%	98%	99%	100%
Error	**0.01**	**0.03**	**0.04**	**0.06**	**0.12**	**0.22**	**1.46**

erroneous as the latter should capture only the raw signal and not the noisy one.

Now selecting the best model (the one with order $n = 3$), one can check its generalization power by examining the same statistics of error on the unseen samples included in the test dataset. The results are shown in Table 4 which demonstrates a good generalization power as the order of magnitude of the error remains practically the same up to 99% of the data while a noticeable increase in the maximum value.

The same results are shown in Table 5 in the case where a polynomial of degree 1 is used to fit the data. Despite of a noticeably higher error statistics compared to the ones obtained with a polynomial of degree 3 (Table 4), the error's statistics remain quite small and makes this model eligible to be used in the design of the feedback control. Notice however that even if the linear polynomial is used to design the control law, this still induces a nonlinear control as the second derivatives $y^{(2)} = \dot{x}_2$, given by the r.h.s of (14b), is obviously a nonlinear function of the state.

Table 5
Noisy measurements. Normalized percentiles of errors on **test dataset** using the model's order $n = 3$, $d = 1$.

q	50%	80%	90%	95%	98%	99%	100%
Error	0.03	0.07	0.09	0.13	0.24	0.36	1.82

4.5. Feedback design

The control design is based on a very specific version of NMPC that is tailored to this specific problem of regulating a chain of 3 integrators. This can be summarized by the following two steps:

(1) At each decision instant k, compute a trajectory that is compatible with the estimated values $\hat{y}^{(i)}(k)$, $i = 0, \ldots, 2$ and the targeted desired value y_{ref}. This is done by identifying the coefficients α_i of the following time-profile:

$$\bar{y} = \Psi(t, \alpha) := \sum_{j=1}^{4} \alpha_i \times e^{-\sigma j t} \tag{18}$$

through the solution of the following linear system of equations in the unknown α:

$$\begin{bmatrix} \Psi(0,\alpha) \\ \dot{\Psi}(0,\alpha) \\ \ddot{\Psi}(0,\alpha) \\ \Psi(\infty,\alpha) \end{bmatrix} = \begin{bmatrix} \hat{y}_k^{(0)} \\ \hat{y}_k^{(1)} \\ \hat{y}_k^{(2)} \\ y_{\text{ref}} \end{bmatrix} =: \begin{bmatrix} \hat{\xi}_k \\ y_{\text{ref}} \end{bmatrix} \tag{19}$$

where ξ is used to denote the state vector of the system in the derivatives-based coordinates while $\hat{\xi}_k$ denotes the observer based estimation of ξ_k. The solution of this system leads to the $\hat{\xi}_k$-dependent solution denoted by $\alpha(\hat{\xi}_k)$.

(2) The so obtained solution $\alpha(\hat{\xi}_k)$ when used in (18) enables to compute a *desired* third derivative $\Psi^{(3)}\Big(0, \alpha(\hat{\xi}_k)\Big)$ that one can try to achieve using the appropriate control u and the identified polynomial given by (11), namely:

$$u^\star(\hat{\xi}_k) = \arg\min_{v \in \mathbb{U}} \left| \Psi^{(3)}\Big(0, \alpha(\hat{\xi}_k)\Big) - \mathcal{P}(\hat{\xi}_k, v) \right| \tag{20}$$

In order to get a real-time implementable control for this sensitive system (that needs very small sampling periods), this process is not necessarily performed at each sampling period (set here to $\tau = 800$ μsec) as this might lead to a computation time exceeding τ. Rather, we block the control over an integer number κ of sampling periods. Moreover, the polynomial of order 1 (Table 5) is used for the sake of reducing the computation time and getting simpler design. Closed-loop simulation is shown in Fig. 3 for a 5 s scenario involving step and ramp like changes in the reference angle with a blocking parameter of $\kappa = 3$. The control saturation level is set to $\bar{u} = 0.15$ (using higher actions lead to larger chattering around the reference). This induces a real-life compatible feedback (the total simulation time of the 5 s scenario is 1.3 s).

5. Conclusion and future work

In this paper, a framework for the identification of continuous-time nonlinear system is proposed that leverages some recent advances in the high derivative computation and multivariate polynomial identification. Undergoing investigations concern the application of this framework to a set of benchmark including MIMO systems. Moreover, a freely available python module is under preparation for potential interested users.

Declaration of competing interest

There is no conflict of interest regarding this paper.

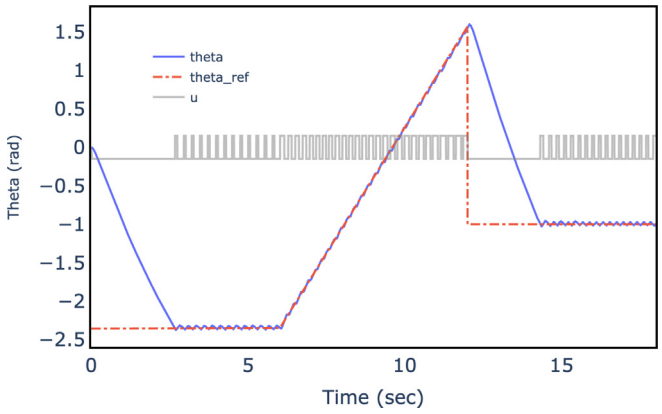

Fig. 3. Behaviour of the closed-loop control using a sampling period of $\tau = 800$ μsec and a blocking order $\kappa = 3$.

Data availability

Data will be made available on request.

References

Alamir, M. (2025a). *Nonlinear control of uncertain systems. Conventional and learning-based alternatives with Python*. Springer-Nature (under press).

Alamir, M. (2025b). On reconstructing high derivatives of noisy time-series with confidence intervals. *Automatica*, *180*, Article 112469.

Alamir, M. (2025c). Scalable sparse identification of nonlinear relationships. In *Nonlinear control of uncertain systems: conventional and learning-based alternatives with python*. Springer-Nature.

Biagetti, G., Crippa, P., Falaschetti, L., & Turchetti, C. (2019). A machine learning approach to the identification of dynamical nonlinear systems. In *2019 27th European signal processing conference* (pp. 1–5). http://dx.doi.org/10.23919/EUSIPCO.2019.8902539.

Brunton, S. L., & Kutz, J. N. (2022). *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press.

Conatser, R., Wagner, J., Ganta, S., & Walker, I. (2004). Diagnosis of automotive electronic throttle control systems. *Control Engineering Practice*, *12*(1), 23–30.

de Silva, B. M., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J. N., & Brunton, S. L. (2020). Pysindy: a python package for the sparse identification of nonlinear dynamics from data. arXiv preprint arXiv:2004.08424.

Greblicki, W. (2002). Continuous-time Hammerstein system identification. *IEEE Transactions on Automatic Control*, *47*(8), 1363–1366.

Jui, J. J., & Ahmad, M. A. (2021). A hybrid metaheuristic algorithm for identification of continuous-time Hammerstein systems. *Applied Mathematical Modelling*, *95*, 339–360.

Paduart, J., Lauwers, L., Pintelon, R., & Schoukens, J. (2012). Identification of a Wiener–Hammerstein system using the polynomial nonlinear state space approach. *Control Engineering Practice*, *20*(11), 1133–1139.

Paduart, J., Lauwers, L., Swevers, J., Smolders, K., Schoukens, J., & Pintelon, R. (2010). Identification of nonlinear systems using polynomial nonlinear state space models. *Automatica*, *46*(4), 647–656.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rao, G. P., & Unbehauen, H. (2006). Identification of continuous-time systems. *IEE Proceedings-Control Theory and Applications*, *153*(2), 185–220.

Sadeghi, M., & Farrokhi, M. (2018). Online identification of non-linear dynamic systems by Wiener model using subspace method and neural networks. *Transactions of the Institute of Measurement and Control*, *40*(15), 4211–4221.