Automatica 196 (2027) 113334

Contents lists available at ScienceDirect

Automatica

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

Efficient identification of linear, parameter-varying, and nonlinear systems with noise models ☆

Alberto Bemporad [a,*], Roland Tóth [b,c]

[a] IMT School for Advanced Studies Lucca, Lucca, Italy
[b] Control Systems Group, Eindhoven University of Technology, Eindhoven, The Netherlands
[c] Systems and Control Laboratory, HUN-REN Institute for Computer Science and Control, Budapest, Hungary

ARTICLE INFO

Article history:
Received 15 April 2025
Received in revised form 13 May 2026
Accepted 16 August 2026

Keywords:
Nonlinear system identification
Linear parameter varying systems
Machine learning based identification
Deep learning
Noise modeling

ABSTRACT

We present a general system identification procedure capable of estimating a broad spectrum of state-space dynamical models, including *linear time-invariant* (LTI), *linear parameter-varying* (LPV), and nonlinear (NL) dynamics, along with rather general classes of noise models. Similar to the LTI case, we show that for this general class of model structures, including the NL case, the model dynamics can be separated into a *deterministic process* and a *stochastic noise* part, allowing one to seamlessly tune the complexity of the combined model both in terms of nonlinearity and noise modeling. We parameterize the involved nonlinear functional relations by means of *artificial neural-networks* (ANNs), although alternative parametric nonlinear mappings can also be used. To estimate the resulting model structures, we optimize a prediction-error-based criterion using an efficient combination of a constrained quasi-Newton approach and automatic differentiation, achieving training times in the order of seconds compared to existing state-of-the-art ANN methods which may require hours for models of similar complexity. We formally establish the consistency guarantees for the proposed approach and demonstrate its superior estimation accuracy and computational efficiency on several benchmark LTI, LPV, and NL system identification problems.

© 2026 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

System identification, i.e., data-driven modeling, has been an important field of research in control engineering and in statistics, providing reliable approaches for obtaining models of poorly understood systems, filling the gaps where first-principle knowledge is lacking, and providing the machinery for adaptation and data-driven design of controllers (Schoukens & Ljung, 2019). However, going beyond the well-established *linear time-invariant* (LTI) system identification framework (Ljung, 1999) and considering, e.g., *linear time-varying* (LTV) (Niedźwiecki, 2000), *linear parameter-varying* (LPV) (Tóth, 2010), or various *nonlinear* (NL) models (Giri & Bai, 2010), a core question of the resulting identification problems is how to efficiently estimate from data the functional relations involved in these models.

While classical *beyond-LTI* methods used fixed basis-function parameterizations to express nonlinearities, with the recent progress of computational capabilities and machine learning methods, a wide range of learning approaches has been introduced, from *Gaussian process* (GP) regression (Pillonetto et al., 2014) and *support vector machines* (Suykens et al., 2002) to *artificial neural networks* (ANN) based training (see Pillonetto et al., 2025 for a recent overview), including *state-space* (SS) recurrent networks such as LSTMs (Gonzalez & Yu, 2018; Ljung et al., 2020), SS-ANNs (Beintema et al., 2023; Bemporad, 2023b, 2025; Forgione & Piga, 2021; Masti & Bemporad, 2021; Ribeiro et al., 2020; Suykens et al., 1996), etc. In particular, the latter class of methods successfully combines deep learning with state-space model identification concepts, achieving extreme accuracy levels on well-established benchmarks, see Beintema et al. (2023). However, in the quest of solving the underlying function estimation problem with improved accuracy and reliability, many aspects of the classical identification theory have been sacrificed. While deep-learning methods can achieve remarkable results, the related training time when no state-measurements are available is often too high compared to classical identification methods,

☆ This research was supported by the Air Force Office of Scientific Research under award number FA8655-23-1-7061, by The MathWorks Inc., and by the European Research Council (ERC), Advanced Research Grant COMPACT (Grant Agreement No. 101141351). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them. The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Wei Xing Zheng under the direction of Editor Alessandro Chiuso.

* Corresponding author.
E-mail addresses: alberto.bemporad@imtlucca.it (A. Bemporad), r.toth@tue.nl (R. Tóth).

https://doi.org/10.1016/j.automatica.2026.113334
0005-1098/© 2026 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

requiring tens of hours, sometimes days, with the most sophisticated stochastic-gradient methods, like Adam (Kingma & Ba, 2014), to train mid-sized models on training sequences of reasonable length. Using such methods in an identification cycle, which requires iterations on model structure selection, hyperparameter optimization on validation data, and experiment design, is simply impractical.

In addition to computational challenges, most ML-based identification methods neglect or oversimplify process and measurement noise, with few exceptions (Beintema et al., 2023; Ribeiro et al., 2020). The well-developed understanding of process/noise models, one/n-step-ahead predictors, and estimation concepts for LTI systems is largely absent in the NL case, leaving practitioners unclear about how to co-estimate linear and nonlinear components, combine process and noise models, and scale identification to nonlinear settings. For notable contributions addressing nonlinear system identification of black-box autoregressive and state-space models, we refer the reader to the pioneering work (Sjöberg & Raedt, 1997) and its associated symbolic-numeric toolbox (Hjalmarsson & Sjöberg, 2012; Sjöberg & Hjalmarsson, 2009). That line of research also introduced a stepwise identification methodology to incrementally increase the complexity of the models (Sjöberg, 1997, 2000).

1.1. Contribution

To overcome the aforementioned challenges, our first contribution is to provide an extension of the classical process and noise modeling framework of Ljung (1999) to the general NL case, showing that even for general nonlinear stochastic state–space models with innovation noise, the dynamics can be separated into a deterministic process and a stochastic noise model. This allows us to unify a spectrum of model-structure representations from LTI to NL classes, including externally-scheduled and self-scheduled LPV models, enabling us to seamlessly tune the complexity of the model, with model structure selection both in terms of nonlinearity and noise modeling. We show that, in the proposed nonlinear system identification setting, the inverse of the noise model can always be analytically computed and it is beneficial to learn this inverse noise model rather than the direct one.

Our second contribution is to extend the identification methodology recently introduced in Bemporad (2025), which is based on L-BFGS-B optimization (Byrd et al., 1995) and the efficient auto-differentiation framework of JAX (Bradbury et al., 2018), to identify the resulting combination of process and noise models under various LTI, LPV, and NL model classes and ANN parameterizations of the involved nonlinear functional relations. In our approach, the hidden states are directly estimated via forward prediction through the model equations, so that the training problem is reformulated as a nonlinear optimization problem to be solved with respect to the model parameters and the initial hidden state (partially "condensed" problems in which some hidden states are kept as optimization variables are viable alternatives, see, e.g., Forgione and Piga (2020), Bemporad (2023a, Sect. III.B), although they are more amenable to stochastic gradient methods, leading to slower convergence and higher training time). As we demonstrate on various simulation and benchmark examples, the resulting identification method is a highly efficient and unified approach to system identification that can achieve, and even surpass, the accuracy of the state-of-the-art deep-learning-based identification methods, like SUBNET (Beintema et al., 2023), while bringing the total training time from the order of hours to the order of seconds.

Finally, our last contribution is to show consistency of the resulting identification method. Together with formally proving that the dynamics of a nonlinear data-generating system with an innovation type of noise structure can be generally separated to a process model and nonlinear noise model that can be jointly estimated, this makes our approach a statistically and computationally reliable tool for solving identification problems in practice with the same level of simplicity of the existing tools available for the LTI case.

The paper is structured as follows. In Section 2, the considered identification setting is introduced with our separation result. We also discuss the unified parameterization of model structures for capturing the dynamics of the data-generating system and introduce the considered identification problem. This is followed by detailing the proposed identification method for the introduced model structures in Section 4, together with a formal proof of its consistency in Section 5. The estimation accuracy and computational efficiency of the proposed method are demonstrated in Section 6 on several examples and benchmarks. Finally, conclusions about the achieved results are given in Section 7.

1.2. Notation

$\mathbb{R}$ and $\mathbb{Z}$ denote the set of real and integer numbers, respectively, while $\mathbb{Z}^+$ (or $\mathbb{Z}_0^+$) is the set of positive (non-negative) integers. Given a matrix $A \in \mathbb{R}^{n\times m}$, we denote by $\text{vec}(A)$ the nm-dimensional column vector obtained by stacking all columns of A on top of one another. More generally, given p vectors $v_1, \dots, v_p$, then $\text{vec}(v_1, \dots, v_p) = [v_1^\top \ \dots \ v_p^\top]^\top$. Additionally, $\mathbb{I}_{\tau_1}^{\tau_2}$ is the set of integers τ with $\tau_1 \le \tau \le \tau_2$.

2. Identification setting

2.1. Data-generating system

Consider the to-be-identified *discrete-time* system defined in terms of a *state–space* (SS) representation with an *innovation*-type of noise process:

$$w_{k+1} = f_{\text{w}}(w_k, u_k, e_k), \tag{1a}$$

$$y_k = g_{\text{w}}(w_k, u_k) + e_k, \tag{1b}$$

where $k \in \mathbb{Z}$ denotes the discrete-time step, $u_k \in \mathbb{R}^{n_u}$ is a quasi-stationary input signal, $w_k \in \mathbb{R}^{n_{\text{w}}}$ is the state, $y_k \in \mathbb{R}^{n_y}$ is the output, and e_k is an i.i.d. (independent and identically distributed) white noise process with finite variance $\Sigma_{\text{e}} \in \mathbb{R}^{n_y\times n_y}$, independent of u. The functions $f_{\text{w}} : \mathbb{R}^{n_{\text{w}}} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{\text{w}}}$ and $g_{\text{w}} : \mathbb{R}^{n_{\text{w}}} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$, i.e., the state transition and output functions, are considered to be real-valued, deterministic, and possibly nonlinear functions. By assuming various structures for f_{w} and g_{w}, many well-known model structures can be obtained such as *nonlinear output noise* (NOE), *nonlinear auto-regressive with exogenous input* (NARX), *nonlinear auto-regressive with moving average exogenous input* (NARMAX) and *nonlinear Box–Jenkins* (NBJ) (Billings, 2013; Jansson, 2003; Sjöberg & Raedt, 1997). For instance, if f_{w} does not depend on e_k, then a *nonlinear* (NL)-SS model with an OE noise structure is obtained.

Under mild conditions, the dynamics of (1) can be separated into a *deterministic process* and a *stochastic noise* part. Let

$$G_{\text{o}} : \begin{cases} x_{k+1} = f_{\text{x}}(x_k, u_k), \\ y_{\text{o},k} = g_{\text{x}}(x_k, u_k), \end{cases} \tag{2a}$$

correspond to the deterministic process part of (1), with $x_k \in \mathbb{R}^{n_{\text{x}}}$ being the associated state and $f_{\text{x}} : \mathbb{R}^{n_{\text{x}}} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{\text{x}}}$, $g_{\text{x}} : \mathbb{R}^{n_{\text{x}}} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$ being deterministic maps. To ensure an equivalence with (1b), the output $y_{\text{o},k}$ is observed under a stochastic disturbance v_k:

$$y_k = y_{\text{o},k} + v_k, \tag{2b}$$

where v_k is defined by a hidden Markov model:

$$H_o : \begin{cases} z_{k+1} = f_z(z_k, x_k, u_k, e_k), \\ v_k = g_z(z_k, x_k, u_k) + e_k. \end{cases} \tag{2c}$$

In (2c), z_k is a latent stochastic process taking values in $\mathbb{R}^{n_z}$ at each time instant k, while $f_z : \mathbb{R}^{n_z} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_z}$ and $g_z : \mathbb{R}^{n_z} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$ are deterministic and possibly nonlinear functions. In the sequel, we will also treat G_o and H_o as dynamic operators, expressing the signal relations of (2a) and (2c) under left-compact support, i.e., we can write $y_{o,k} = G_o u_k$ and $v_k = H_o(x_k, u_k)e_k$. For the sake of simplicity of notation, we will omit the dependency of H_o on (x_k, u_k) from now on.

Theorem 1 (*Separation of Process and Noise Models*)**.** *A system given by (1) always has a representation in the form of (2a)–(2c) with $n_x, n_z \le n_w$, such that for any sample path realization of $e \in \mathbb{R}^{\mathbb{Z}_0^+}$ and $u \in \mathbb{R}^{\mathbb{Z}_0^+}$ and any initial condition $w_0 \in \mathbb{R}^{n_w}$, there exist $x_0 \in \mathbb{R}^{n_x}$ and $z_0 \in \mathbb{R}^{n_z}$ such that the resulting output trajectories in terms of (1b) and (2b) are equivalent.*

Proof. See Appendix.

If additionally, in (2c), $f_z(0, ., ., 0) \equiv 0$ and $g_z(0, ., .) \equiv 0$ are satisfied, then we call G_o and H_o a *well-posed separation* of the deterministic and stochastic parts of the system. Fulfillment of this condition means that in case the generating noise e_k is zero and the initial condition z_0 is zero, then z_{k+1} remains zero regardless of any variation of x_k or u_k. See the proof of Theorem 1 for such a construction. In the remaining part of the paper and in our examples, we will take this as an assumption without any loss of generality. We also highlight that, due to the nonlinearity of (2c), $\mathbb{E}_{e,z_0}\{v_k\} = \bar{v}_k$ along a given $\{x_\tau\}_{\tau=0}^k$ and $\{u_\tau\}_{\tau=0}^k$ is often nonzero, where the expectation is taken w.r.t. e and an initial probability density function $p_{z,0}$ of z_0.

Note that a separate representation of the deterministic part G_o and stochastic part H_o of the dynamics of (1) can increase the total state dimension $n_x + n_z$ of the combined model compared to n_w of (1). However, this allows one to estimate G_o and H_o separately, and hence to address the deterministic behavior, e.g., in terms of control design, directly, compared to earlier works using (1) as a general model structure, such as Sjöberg and Raedt (1997), where these dynamics are intertwined and difficult to separate a posteriori.

By using the introduced separation, Eqs. (2a)–(2c) define the *data-generating* system depicted in Fig. 1, that we intend to identify from an *input–output* (IO) dataset $\mathcal{D}_N = \{(u_k, y_k)\}_{k=0}^{N-1}$, where $\{y_k\}_{k=0}^{N-1}$ is the response collected from the data-generating system (2a)–(2c) in terms of a sample path realization for a given excitation sequence $\{u_k\}_{k=0}^{N-1}$ and potentially unknown initial states $x_0 \in \mathbb{R}^{n_x}$ and $z_0 \in \mathbb{R}^{n_z}$. To avoid unnecessary clutter, we will not use different notation for random variables such as e_k and their sampled values, but at places where confusion might arise, we will specify which notion is used.

As $e_k = v_k - g_z(z_k, x_k, u_k)$, the inverse of the noise process (2c) can be expressed analytically as

$$H_o^{-1} : \begin{cases} z_{k+1} = f_z(z_k, x_k, u_k, v_k - g_z(z_k, x_k, u_k)), \\ e_k = v_k - g_z(z_k, x_k, u_k). \end{cases} \tag{3}$$

It is easy to show that, for any $z_0 \in \mathbb{R}^{n_z}$ and sequence $\{x_k, u_k\}_{k=0}^{\infty}$, we have $e_k = H_o^{-1}H_o e_k$ and $v_k = H_o H_o^{-1} v_k$ when (3) and (2c) are both initialized at z_0. Moreover, by defining two functions $\tilde{f}_z$ and $\tilde{g}_z$ such that $\tilde{f}_z(z_k, x_k, u_k, v_k) = f_z(z_k, x_k, u_k, v_k - g_z(z_k, x_k, u_k))$ and $g_z = -\tilde{g}_z$, we obtain

$$f_z(z_k, x_k, u_k, e_k) = \tilde{f}_z(z_k, x_k, u_k, e_k - \tilde{g}_z(z_k, x_k, u_k)).$$

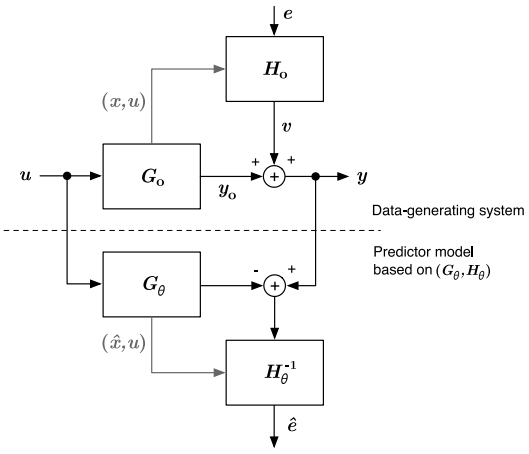


Fig. 1. Separation of the data-generating system into a deterministic and stochastic part in the considered prediction-error-minimization setting.

By exploiting that $v_k = y_k - y_{o,k} = y_k - G_o u_k$, we get

$$e_k = H_o^{-1}(y_k - G_o u_k). \tag{4}$$

Next, using Eqs. (2a)–(2c), we can write

$$y_k = G_o u_k + H_o e_k, \tag{5a}$$
$$= G_o u_k + (H_o - I)e_k + e_k, \tag{5b}$$

and, by substituting (4), we get

$$y_k = G_o u_k + (H_o - I)H_o^{-1}(y_k - G_o u_k) + e_k, \tag{5c}$$
$$= G_o u_k + (I - H_o^{-1})(y_k - G_o u_k) + e_k, \tag{5d}$$

which remarkably holds true as in the LTI case, despite the possible nonlinearity of G_o and H_o, or even regardless of their stability.

Then, following the classical reasoning, under the information set of observed $y^{(k-1)} = \{y_\tau\}_{\tau \le k-1}$ and $u^{(k)} = \{u_\tau\}_{\tau \le k}$, the one-step-ahead prediction of $y(k)$ w.r.t. the ℓ_2 loss is $\hat{y}(k \mid k-1) = \arg\min_{\delta \in \mathbb{R}^{n_y}} \mathbb{E}\{\|y(k) - \delta\|_2^2 \mid u^{(k)}, y^{(k-1)}\}$. Based on (5d), the straightforward application of the expectation operator yields

$$\hat{y}(k \mid k-1) = G_o u_k + (I - H_o^{-1})(y_k - G_o u_k), \tag{6}$$

which can be given in terms of an SS representation as

$$x_{k+1} = f_x(x_k, u_k), \tag{7a}$$
$$z_{k+1} = \tilde{f}_z(z_k, x_k, u_k, y_k - g_x(x_k, u_k)), \tag{7b}$$
$$\hat{y}(k \mid k-1) = g_x(x_k, u_k) - \tilde{g}_z(z_k, x_k, u_k). \tag{7c}$$

To the best of the authors' knowledge, such a general and computable form of the one-step-ahead predictor in the nonlinear case has not been shown in the literature before.

2.2. Model parameterization

Given the dataset $\mathcal{D}_N$, we aim to estimate G_o and H_o in terms of parameterized forms of (2a) and (2c). Specifically, we consider the model G_θ of the process in the following parametric SS form

$$G_\theta : \begin{cases} \hat{x}_{k+1} = f_x(\hat{x}_k, u_k, \theta_x), \\ \hat{y}_k = g_x(\hat{x}_k, u_k, \theta_y), \end{cases} \tag{8a}$$

where $\hat{x}_k \in \mathbb{R}^{\hat{n}_x}$ is the process model state with $\hat{n}_x \ge 1$, not necessarily being equal to (the usually unknown) state dimension n_x of the data-generating system, and $f_x : \mathbb{R}^{\hat{n}_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_{\theta_x}} \to \mathbb{R}^{\hat{n}_x}$, $g_x : \mathbb{R}^{\hat{n}_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_{\theta_y}} \to \mathbb{R}^{n_y}$ are functions parameterized by

vectors θ_x and θ_y, respectively (with a slight abuse of notation, we also use f_x and g_x to characterize the parameterized functions, similar to the notation for the data-generating system). We will specify later different parameterizations and structures of these functions.

To deal with the presence of noise, i.e., to model the discrepancy

$$\hat{v}_k = y_k - \hat{y}_k,$$

with $\hat{v}_k \in \mathbb{R}^{n_y}$, between the measured output y_k and the output $\hat{y}_k$ predicted by the process model, we aim to estimate the inverse of H_o in terms of the parameterized noise model H_θ^{-1}, described by

$$H_\theta^{-1}: \begin{cases} \hat{z}_{k+1} = \tilde{f}_z(\hat{z}_k, \hat{x}_k, u_k, \hat{v}_k, \theta_z), \\ \hat{e}_k = \tilde{g}_z(\hat{z}_k, \hat{x}_k, u_k, \theta_e) + \hat{v}_k, \end{cases} \tag{8b}$$

where $\hat{z}_k \in \mathbb{R}^{\hat{n}_z}$ is the state of the inverse noise model, $\tilde{f}_z : \mathbb{R}^{\hat{n}_z} \times \mathbb{R}^{\hat{n}_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_{\theta_z}} \to \mathbb{R}^{n_z}$ and $\tilde{g}_z : \mathbb{R}^{\hat{n}_z} \times \mathbb{R}^{\hat{n}_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_{\theta_e}} \to \mathbb{R}^{n_y}$ are parameterized by two more vectors θ_z and θ_e, respectively, to be determined together with θ_x and θ_y. Note that, based on (3), the chosen model structure (8b) enables us to easily derive the forward noise model H_θ as

$$H_\theta: \begin{cases} \hat{z}_{k+1} = \tilde{f}_z(\hat{z}_k, \hat{x}_k, u_k, \hat{e}_k - \tilde{g}_z(z_k, \hat{x}_k, u_k, \theta_e), \theta_z), \\ \hat{v}_k = \hat{e}_k - \tilde{g}_z(z_k, \hat{x}_k, u_k, \theta_e), \end{cases} \tag{8c}$$

which gives that $\tilde{f}_z$ and $\tilde{g}_z$ must satisfy the condition $\tilde{g}_z(0, ., ., \theta_e) \equiv 0$ and $\tilde{f}_z(0, ., ., 0, \theta_z) \equiv 0$ for all $\theta_z \in \Theta_z \subseteq \mathbb{R}^{n_{\theta_z}}$ and $\theta_e \in \Theta_e \subseteq \mathbb{R}^{n_{\theta_e}}$ to satisfy the separation constraints. Different ways of parameterizing G_θ and H_θ^{-1} will be detailed in Section 3.

2.3. Identification problem

Based on the given dataset $\mathcal{D}_N$, the objective is to estimate the model parameters $\theta = \text{vec}(\theta_x, \theta_y, \theta_z, \theta_e)$ by minimizing the criterion

$$V_{\mathcal{D}_N}(\theta, \hat{w}_0) = \frac{1}{N} \sum_{k=0}^{N-1} \|\hat{e}_k^{\text{pred}}\|_2^2 \tag{9}$$

where $\hat{e}_k^{\text{pred}} = y_k - \hat{y}_k^{\text{pred}}$ and $\hat{y}_k^{\text{pred}}$ is computed via the prediction model

$$\hat{x}_{k+1} = f_x(\hat{x}_k, u_k, \theta_x), \tag{10a}$$

$$\hat{z}_{k+1} = \tilde{f}_z(\hat{z}_k, \hat{x}_k, u_k, y_k - g_x(\hat{x}_k, u_k, \theta_y), \theta_z), \tag{10b}$$

$$\hat{y}_k^{\text{pred}} = g_x(\hat{x}_k, u_k, \theta_y) - \tilde{g}_z(\hat{z}_k, \hat{x}_k, u_k, \theta_e), \tag{10c}$$

for $k \in \mathbb{I}_0^{N-1}$, with $\hat{w}_0 = \text{vec}(\hat{x}_0, \hat{z}_0)$ as the initial condition, $\hat{w}_0 \in \mathbb{R}^{\hat{n}_w}$, $\hat{n}_w = \hat{n}_x + \hat{n}_z$.

The objective (10) corresponds to the traditional *prediction-error-minimization* (PEM) setting and aims to minimize the error of the one-step-ahead predictor of the combined model structure in terms of (6) w.r.t. the observed output samples $\{y_k\}_{k=0}^{N-1}$. Furthermore, as we will see in Section 5, (9) together with (10) also have favorable properties to show consistency of the resulting estimator.

Note that the classical PEM problem introduced w.r.t. to LTI models, such as in Ljung (1999), is the same as the minimization of (9) in case we substitute linear models (see (11) later) into (10). One particular difference is that H_θ^{-1} is directly parameterized in (10), compared to traditional PEM structures where H_θ is parameterized and H_θ^{-1} is computed. While this bears no significance in the LTI case, in the general nonlinear case it leads to a simpler optimization problem, with the added benefit that the resulting H_θ^{-1} is the natural form for filtering, which is the primary use of an estimated noise model. In fact, one of the significant contributions of the paper is the extension of the LTI formulation of the one-step-ahead predictor, which uses a separated process model G_θ and noise model H_θ, to the nonlinear case, and the demonstration that these models can be parameterized separately and that the identification can be formulated as a nonlinear optimization problem.

3. Model structures

Next, we discuss various parameterizations of G_θ and H_θ^{-1} that can be used to estimate linear, parameter-varying, and nonlinear models.

3.1. LTI models

To restrict the model structure to only express LTI dynamics, we take G_θ in the following form

$$G_\theta: \begin{cases} \hat{x}_{k+1} = \overbrace{A_x\hat{x}_k + B_x u_k}^{f_x(\hat{x}_k, u_k, \theta_x)}, \\ \hat{y}_k = \underbrace{C_x\hat{x}_k + D_x u_k}_{g_x(\hat{x}_k, u_k, \theta_y)}, \end{cases} \tag{11a}$$

and parameterize H_θ^{-1} as

$$H_\theta^{-1}: \begin{cases} \hat{z}_{k+1} = \overbrace{A_z\hat{z}_k + B_z\hat{v}_k}^{\tilde{f}_z(\hat{z}_k, \hat{x}_k, u_k, \hat{v}_k, \theta_z)}, \\ \hat{e}_k = \underbrace{C_z\hat{z}_k}_{\tilde{g}_z(\hat{z}_k, \hat{x}_k, u_k, \theta_e)} + \hat{v}_k, \end{cases} \tag{11b}$$

for which $\theta_x = \text{vec}([A_x\ B_x])$, $\theta_y = \text{vec}([C_x\ D_x])$, $\theta_z = \text{vec}([A_z\ B_z])$, $\theta_e = \text{vec}(C_z)$. Note that, by appropriately restricting (A_z, B_z, C_z) further, well known LTI model structures can be realized, from ARX to BJ.[1]

3.2. LPV models

To estimate the system dynamics in LPV form, G_θ and H_θ^{-1} are formulated as

$$G_\theta: \begin{cases} \hat{x}_{k+1} = A_x(p_k)\hat{x}_k + B_x(p_k)u_k, \\ \hat{y}_k = C_x(p_k)\hat{x}_k + D_x(p_k)u_k, \end{cases} \tag{12a}$$

$$H_\theta^{-1}: \begin{cases} \hat{z}_{k+1} = A_z(p_k)\hat{z}_k + B_z(p_k)\hat{v}_k, \\ \hat{e}_k = C_z(p_k)\hat{z}_k + \hat{v}_k, \end{cases} \tag{12b}$$

where $p_k \in \mathbb{R}^{n_p}$ is the scheduling variable. In the so called *self-scheduled* case (see Verhoek et al., 2022 for a detailed discussion on LPV modeling and identification), $p_k = \psi(x_k, u_k, \theta_\psi)$, with $\psi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_{\theta_\psi}} \to \mathbb{R}^{n_p}$ being a parameterized, generally nonlinear function, called the *scheduling map*. In the alternative *externally-scheduled* case, p_k is considered as a given exogenous signal, which is part of the input $\tilde{u}_k = \text{vec}(u_k, p_k)$ received by the system, and is included in the dataset $\mathcal{D}_N$. By introducing,

$$M_x(p_k) = \begin{bmatrix} A_x(p_k) & B_x(p_k) \\ C_x(p_k) & D_x(p_k) \end{bmatrix}, \quad M_z(p_k) = \begin{bmatrix} A_z(p_k) & B_z(p_k) \\ C_z(p_k) & 0 \end{bmatrix},$$

we parameterize the LPV model as

$$M_x(p_k) = f_{M_x}(p_k, \theta_{M_x}), \ M_z(p_k) = f_{M_z}(p_k, \theta_{M_z}), \tag{13}$$

where f_{M_x} and f_{M_z} are multi-layer *feedforward neural networks* (FNNs) with $L - 1 \geq 0$ hidden layers and a linear output

[1] For ARX and ARMAX models, an observer canonical realization form can be used to separate the states associated with the autoregressive and moving average dynamics, see Ljung (1999, Ch. 4); OE models require setting $(A_z, B_z, C_z) = 0$; BJ corresponds to an unrestricted parameterization of (A_z, B_z, C_z).

layer having n_p inputs and $(n_x + n_y)(n_x + n_u)$ and $n_z^2 + 2n_y n_z$ outputs, respectively, and parameterized by θ_{M_x} and θ_{M_z}, containing the weight and bias terms of the layers.[2] Also, these FNNs can also contain a linear bypass for more efficient modeling of linearly dominant functions. Note that different nonlinear parametric functions can be used here in alternative to FNNs, including parametric physics-based models.

A special case of (13) when f_{M_x} and f_{M_z} only consist of a linear layer, i.e., $L = 1$:

$$M_x(p_k) = M_{x,0} + \sum_{i=1}^{n_p} p_{k,i} M_{x,i} \tag{14}$$

and similarly for $M_z(p_k)$, corresponding to LPV models that depend *affinely* on p_k. The class of LPV state-space models with affine dependence has paramount importance in control engineering, due to the wide range of convex analysis and controller synthesis methods that assume the representation of the system in such a form.

In the self-scheduled setting, the function ψ defining p_k is taken as a multi-layer FNN with weights and biases collected in θ_ψ, while restricting f_{M_x} and f_{M_z} to the linear expression (14). This is without loss of generality: due to the flexibility we have in choosing function ψ, given an arbitrary FNN representation of f_{M_x} and f_{M_z}, we can interpret its last linear layer as in (14), set p_k as the value of the neurons at the second last nonlinear layer, and call ψ the remaining layers. The parameter vector to learn is therefore $\theta = \text{vec}([M_{x,0}, \ldots, M_{x,n_p}], [M_{z,0}, \ldots, M_{z,n_p}], \theta_\psi)$. Note that, with reference to the general model structure (8), in this case, θ_x, θ_y, θ_z, and θ_e share the components of θ_ψ.

In the externally scheduled case, p_k is given, therefore there is no need to estimate ψ. Instead, we take f_{M_x} and f_{M_z} as general FNNs, with $\theta = \text{vec}(\theta_{M_x}, \theta_{M_z})$ as the overall parameter vector, and θ_x, θ_y, θ_z, θ_e defined accordingly with possibly shared components.

3.3. General nonlinear models

In the general case of black-box nonlinear models, G_θ and H_θ^{-1} are considered in the form of (8a) and (8b), corresponding to *recurrent neural network* (RNN) models when $f_x, g_x, \tilde{f}_z, \tilde{g}_z$ are taken as multi-layered FNNs with linear output layer and a possible linear bypass. The weights and bias terms of these networks are collected in θ_x, θ_y, θ_z, and θ_e, respectively

Note that, while in the LTI and LPV cases, the considered model structures naturally satisfy the separation constraint, in the general NL case, we need to enforce $\tilde{g}_z(0, ., ., \theta_e) \equiv 0$ and $\tilde{f}_z(0, ., ., 0, \theta_z) \equiv 0$ for all $\theta_z \in \Theta_z \subseteq \mathbb{R}^{n_{\theta_z}}$.

3.4. Identifiability

The introduced parameterization of the considered model structures is generally not *identifiable* under the PEM cost function (9) in the classical sense. This means that there exist many parameter values in the considered parameterization of the state and output equations for which the corresponding one-step-ahead predictor (10), which is used in (9) to characterize model performance, is indistinguishable (Ljung, 1999). This is due to three causes: (*i*) non-identifiability of state–space representations, (*ii*) non-uniqueness of multi-layer ANN parameterizations, and (*iii*) non-uniqueness of the separation of the process and noise parts.

Note that (*i*) is due to the well-known fact that any invertible state transformation leads to an equivalent state–space representation of a system irrespective of whether the dynamics are linear, parameter-varying, or nonlinear. However, this represents no issue in system identification as, e.g., by the use of regularization and reliable bootstrapping for initialization, optimization schemes reliably converge to solutions that are close to the equivalence manifold representing the global optima. In terms of (*iii*), the introduced separation is also non-unique as there exists multiple representations of the considered splitting of the behavior, but this can be handled in the same spirit as (*i*). For (*ii*), it is well-known that advantageous approximation capabilities of ANNs are rooted in their multilayer structure, even if they provide non-unique descriptions of the to-be-represented functions. In machine learning, this is efficiently handled by various methods such as Adam and other stochastic-gradient schemes, see Goodfellow et al. (2016).

Hence, it is quite standard that the considered model structures are not globally identifiable, not even locally; however, for the proposed identification approach discussed in the next section, we will see that this is not an obstacle to obtaining reliable performance in practice, nor, as shown in Section 5, to establishing a consistency proof w.r.t. the equivalence class of state-space representations describing the same system behavior, i.e., the same one-step-ahead predictor.

4. Identification method

Next, we discuss the proposed identification method to estimate a model according to the parameterized model structure (8a) and (8b) based on a given data set $\mathcal{D}_N$ and under the considered criterion (9). We provide a general method that can be applied to any of the model classes described in Section 3.

4.1. Combined plant and noise model learning

To address the posed identification problem in terms of the minimization of (9) with respect to $\theta \in \mathbb{R}^{n_\theta}$, which collects all the free model parameters, and $\hat{w}_0$, which collects the initial states of the plant and noise models, we solve the following optimization problem

$$\min_{\theta, \hat{w}_0} \; V_{\mathcal{D}_N}(\theta, \hat{w}_0) + R(\theta, \hat{w}_0), \tag{15}$$

$$\text{s.t.} \begin{cases} \hat{x}_{k+1} &= f_x(\hat{x}_k, u_k, \theta_x), \\ \hat{z}_{k+1} &= \tilde{f}_z(\hat{z}_k, \hat{x}_k, u_k, y_k - g_x(\hat{x}_k, u_k, \theta_y), \theta_z), \\ \hat{e}_k^{\text{pred}} &= y_k - g_x(\hat{x}_k, u_k, \theta_y) + \tilde{g}_z(\hat{z}_k, \hat{x}_k, u_k, \theta_e), \end{cases}$$

where $R(\theta, \hat{w}_0)$ is a regularization function. The term

$$R(\theta, \hat{w}_0) = \frac{\rho_\theta}{2}\|\theta\|_2^2 + \tau\|\theta\|_1 + \frac{\rho_w}{2}\|\hat{w}_0\|_2^2, \tag{16}$$

widely used in machine learning, includes an *elastic-net* type of regularization on the model parameters θ, where $\rho_\theta > 0$ and $\tau \geq 0$ are the weights defining the ℓ_2 and ℓ_1 penalties, respectively, and $\rho_w > 0$ relates to the ℓ_2-regularization of the initial state $\hat{w}_0$. Both penalties limit the magnitude of the optimization variables (θ and $\hat{w}_0$) and mitigate the risk of overfitting; they handle the problem of non-identifiability by making the solution unique (minimum norm solution among all possible equivalent realizations); the ℓ_1 penalty promotes sparsity in θ to lower model complexity; the quadratic terms also improve the conditioning of (15) by adding positive multiples of the identity to the Hessian

[2] In the multi-layer ANN f_x, parameterized by θ_x, each hidden layer is composed by m activation functions $\phi : \mathbb{R} \to \mathbb{R}$ in the form of $s_{i,j} = \phi(\sum_{l=1}^{m_{i-1}} \theta_{x,i,j,l} s_{i-1,l} + \theta_{b,i,j})$, where $s_i = \text{col}(s_{i,1}, \ldots, s_{i,m_i})$ is the latent variable representing the output of layer $1 \leq i \leq q$. Here, col(.) denotes composition of a column vector. For f_x with q hidden-layers and linear input and output layers, this means $f_x(\hat{x}_k, u_k, \theta_x) = \theta_{x,q+1} s_q(k) + \theta_{x,0} s_0(k) + \theta_{b,q+1}$ and $s_0(k) = \text{col}(\hat{x}_k, u_k)$ where $\theta_{x,0} s_0(k)$ is the linear bypass. The other FNN functions are similarly parameterized. For more details, see (cf. Bemporad, 2023b, Sect. 2).

matrix. For more on elastic-net regularization, see Zou and Hastie (2005).

In this paper, we adopt the optimization approach described in Bemporad (2025) based on handling ℓ_1 penalties by splitting $\theta = \theta_+ - \theta_-, \theta_+, \theta_- \geq 0$, and bound-constrained limited-memory *Broyden–Fletcher–Goldfarb–Shanno* (L-BFGS-B) optimization (Byrd et al., 1995) to solve (15). To better escape local minima, a fixed number of Adam (Kingma & Ba, 2014) gradient-descent steps is used to warm-start the L-BFGS-B optimization (see Bemporad, 2025, Table 3 for the benefit of using Adam as a warm start). We use *automatic differentiation* (AD) (Griewank & Walther, 2008) to compute the gradients of the objective function with respect to $(\theta, \hat{w}_0)$. We rely on the efficient general-purpose AD implementations available in JAX (Bradbury et al., 2018) for differentiating recurrent functions, although other AD tools specialized for nonlinear system identification could be also used, e.g., Donati et al. (2025). In Section 6, we will demonstrate that our optimization toolchain, extended from Bemporad (2025) for the proposed model classes, gives a computationally far more efficient training of the considered ANN-based state–space models than existing nonlinear identification methods using stochastic gradient methods, e.g., Adam, directly applied on various forms of ANN models.

4.2. Model structure selection

In (16), we can also include a *group-lasso* penalty

$$\tau_g \sum_{i=1}^{n_g} \|\theta_{g,i}\|_2 \tag{17}$$

where $\tau_g > 0$ is the regularization weight and $\theta_{g,i}$ represents groups of parameters selected from θ and $\hat{w}_0$ that are related to a specific state dimension of x or z, or elements of the scheduling variable p. For example, in case of a self-scheduled LPV model, $\theta_{g,i}$ can collect the parameters associated with the ith dimension of x, containing the ith rows and columns of the matrices $\{A_{x,j}\}_{j=0}^{n_p}$, the ith rows of $\{B_{x,j}\}_{j=0}^{n_p}$, the ith columns of $\{C_{x,j}\}_{j=0}^{n_p}$ and the ith column of the input layer weights of ψ and the ith element of $\hat{w}_0$. The additional penalty (17) can be used, in principle, to provide automated model structure selection (appropriate choice of $\hat{n}_x$, $\hat{n}_z$, arguments of the scheduling map, groups of neurons in the NN layers, etc.) if the corresponding parameters can be zeroed without increasing the prediction cost. See also Bemporad (2025) and Mulagaleti and Bemporad (2026) for further examples on group-lasso regularization applied to system identification problems of nonlinear state-space models.

A simple implementation of model-structure selection based on group-lasso regularization involves: (*i*) training a model-parameter vector $\hat{\theta}_g$ with $\tau_g > 0$; (*ii*) removing superfluous optimization variables by zeroing the entries in $\hat{\theta}_g$ whose absolute value is below a certain threshold $\epsilon_g > 0$; (*iii*) re-estimating the remaining model parameters with $\tau_g = 0$. The process is repeated with different choices of the hyperparameters τ_g and ϵ_g until an accurate enough, but low-complexity, model is found. This process can be also automated by n-fold cross-validation or choice of τ_g based on more sophisticated AIC/BIC arguments (Rojas et al., 2015). As an alternative, suggested in Candès et al. (2008), an efficient process to recover lower-dimensional models is to weight the parameters appearing in the group-lasso term with the inverse of the corresponding values estimated at the previous estimation step and keep iterating until convergence is achieved.

4.3. Initialization

Since problem (15) is nonconvex, it is beneficial to apply bootstrapping as pointed out in Sjöberg (1997, 2000). For our proposed identification process, first training a deterministic model as in (8a) is used by minimizing $\|y_k - \hat{y}_k\|_2^2 = \|\hat{v}_k\|_2^2$ to get a warm start for θ_x, θ_y, and then the combined plant and noise models (8a) and (8b) are estimated by minimizing (9).

To train NN functions, we initialize the bias terms to zero and the weights with values drawn from the normal distribution with zero mean. For the latter, a slightly more advanced initialization in the form of the Xavier method (Glorot & Bengio, 2010) can also be applied. In case of LTI and LPV models, the elements of the additional matrices can be drawn from a normal distribution or other known initialization schemes for state–space model identification can be followed, e.g., by pre-estimating models via subspace identification in the LTI case.

5. Consistency analysis

In this section, we prove the consistency of the proposed identification scheme, which corresponds to the minimization of (15) under no regularization, i.e., $R(\theta, \hat{w}_0) = 0$. In fact, we will show that the resulting model estimates will converge to an equivalent representation of the system in the form (2a)–(2a) if the number of data points in the available data set $\mathcal{D}_N$ tends to infinity, i.e., $N \to \infty$. Our derivations will also highlight the additional conditions the data-generating system, the available data, and the considered model structures need to fulfill to obtain such guarantees.

5.1. Convergence

First we show convergence of the model estimates relying on the results in Ljung (1978). For this, we require the data generating system to satisfy the following stability condition:

Condition 1 (*Stability of the Data-generating System*)**.** *The data-generating system (2a)–(2c) has the property that, for any $\delta > 0$, there exist constants $c \in [0, \infty)$ and $\lambda \in [0, 1)$ such that*

$$\mathbb{E}_e\{\|y_k - \tilde{y}_k\|_2^4\} < c\lambda^{k-k_o}, \quad \forall k \geq k_o, \tag{18}$$

under any $k_o \geq 0$, $w_o, \tilde{w}_o \in \mathbb{R}^{n_x+n_z}$ with $\|w_o - \tilde{w}_o\|_2 < \delta$, and $\{(u_\ell, e_\ell)\}_{\ell=0}^{k} \in \mathcal{S}_{[0,\infty]}$, where $\mathcal{S}_{[0,\infty]}$ denotes the σ-algebra generated by the random variables $\{(u_\ell, e_\ell)\}_{\ell=0}^{\infty}$, and y_k and $\tilde{y}_k$ satisfy (2a)–(2c) with the same (u_k, e_k), but with initial conditions $\mathrm{vec}(x_{k_o}, z_{k_o}) = w_o$ *and* $\mathrm{vec}(\tilde{x}_{k_o}, \tilde{z}_{k_o}) = \tilde{w}_o$.

This means that both G_o and H_o are assumed to be stable. Note that if the data-generating system is not stable in the above considered incremental sense, then the one-step-ahead predictor (6) is also not stable. This makes the convergence analysis ill posed, as asymptotically the squared prediction error will be infinite in an arbitrary small neighborhood of the parameter vector corresponding to the true system. Furthermore, Condition 1 is also reasonable from a practical point of view as, without stability of (2a)–(2c), it is difficult to obtain open-loop measurement data, requiring closed-loop identification with a stabilizing controller. The latter introduces correlation between u and e, which would require theoretical extension of closed-loop results from the LTI case, which are beyond the scope of this paper.

To identify (2a)–(2c), let the *model structure* $M_\theta = (G_\theta, H_\theta)$, defined by (8a)–(8c), be chosen in terms of the discussed cases in Section 3, corresponding to LTI, LPV with external or internal scheduling, or fully NL model structures with NN parameterization of the underlying nonlinear functions. Let θ be restricted to

a compact set $\Theta \subset \mathbb{R}^{n_\theta}$, giving the *model set* $\mathcal{M} = \{M_\theta \mid \theta \in \Theta\}$. For a given initial condition $\hat{w}_0 \in \mathbb{R}^{\hat{n}_w}$, Eq. (10) defines the following *1-step-ahead predictor* of M_θ

$$\hat{y}_k^{\text{pred}} = m_k^{\text{pred}}(\{y_\ell\}_{\ell=0}^{k-1}, \{u_\ell\}_{\ell=0}^{k}, \theta, \hat{w}_0). \tag{19}$$

Without loss of generality, we will take the assumption that m_k^{pred} is differentiable w.r.t. θ and $\hat{w}_0$ everywhere on an open neighborhood $\breve{\Theta}$ of Θ and $\mathbb{R}^{\hat{n}_w}$, respectively. Note that this is a technical condition, as only such parameterizations are considered for which automated differentiation is available as discussed in Section 4.1. Furthermore, to conduct our analysis, we require (19) to be stable w.r.t. perturbations of the dataset, as formalized by the following condition:

Condition 2 (*Predictor Stability*)**.** *Given a $\delta \geq 0$, there exist constants $c \in [0, \infty)$ and $\lambda \in [0, 1)$ such that, for any $\theta \in \breve{\Theta}$, $\hat{w}_0 \in \mathbb{R}^{\hat{n}_w}$ with $\|\hat{w}_0\|_2 < \delta$, and for any $\{(y_\ell, u_\ell)\}_{\ell=0}^{k}$, $\{(\tilde{y}_\ell, \tilde{u}_\ell)\}_{\ell=0}^{k} \in \mathbb{R}^{(n_y+n_u)\times(k+1)}$, the predictors*

$$\hat{y}_k^{\text{pred}} = m_k^{\text{pred}}(\{y_\ell\}_{\ell=0}^{k-1}, \{u_\ell\}_{\ell=0}^{k}, \theta, \hat{w}_0),$$
$$\tilde{y}_k^{\text{pred}} = m_k^{\text{pred}}(\{\tilde{y}_\ell\}_{\ell=0}^{k-1}, \{\tilde{u}_\ell\}_{\ell=0}^{k}, \theta, \hat{w}_0),$$

satisfy

$$\|\hat{y}_k^{\text{pred}} - \tilde{y}_k^{\text{pred}}\|_2 \leq c\gamma(k), \quad \forall k \geq 0, \tag{20}$$

with $\gamma(k) = \sum_{\ell=0}^{k} \lambda^{k-\ell} \left(\|u_\ell - \tilde{u}_\ell\|_2 + \|y_\ell - \tilde{y}_\ell\|_2\right)$ and $\|m_k^{\text{pred}}(\{0\}_{\ell=\hat{w}_0}^{k-1}, \{0\}_{\ell=0}^{k}, \theta, \hat{w}_0)\|_2 \leq c$. Furthermore, (20) is also satisfied by $\frac{\partial}{\partial\theta} m_k^{\text{pred}}$ and $\frac{\partial}{\partial\hat{w}_0} m_k^{\text{pred}}$.

Note that without the stability of the predictor in terms of Condition 2, any asymptotic analysis is ill posed as when $k \to \infty$, $\hat{y}_k^{\text{pred}}$ becomes unbounded, making the limit of the parameter estimate non-existing. This assumption also implies that the process model G_θ and the inverse noise model H_θ^{-1} are stable for any $\theta \in \breve{\Theta}$. While this is a technical condition required for analysis, similarly to the classical setting (Ljung, 1999), it is not enforced in the considered model structures, although it is possible to do so by applying the parameterization in Revay et al. (2024).

Theorem 2 (*Convergence*)**.** *Given the data-generating system (2a)–(2c) satisfying Condition 1 with a quasi-stationary u independent of the white noise process e and the model set $\mathcal{M}$ defined by (8a)–(8c), satisfying Condition 2. Then,*

$$\sup_{\theta\in\Theta, \hat{w}_0\in\mathbb{R}^{\hat{n}_w}} \left\| V_{\mathcal{D}_N}(\theta, \hat{w}_0) - \mathbb{E}_e\{V_{\mathcal{D}_N}(\theta, \hat{w}_0)\} \right\|_2 \to 0, \tag{21}$$

with probability 1 as $N \to \infty$.

Proof. The identification criterion (9) satisfies Condition C1 in Ljung (1978), hence the proof of Ljung (1978, Lemma 3.1) applies for the considered case. ■

5.2. Consistency

In order to show consistency, we need to assume that the data-generating system (G_o, H_o) belongs to the chosen model set (G_θ, H_θ). This means that for the considered model set $\mathcal{M}$ defined by (8a)–(8c) under $\theta \in \Theta$, there exists a $\theta_o \in \Theta$ such that the one-step-ahead predictor m_k^{pred} associated with M_{θ_o} and the one-step-ahead predictor (6) associated with the data-generating system (G_o, H_o), defined by (2a)–(2c), correspond to equivalent state-space representations. However, as state-space representations are not unique, we can define $\Theta_o \subset \Theta$ which contains all $\theta_o \in \Theta$ that defines an equivalent model with the data-generating system (G_o, H_o) in the one-step-ahead prediction sense. If $\Theta_o = \varnothing$, then the model set is not rich enough to capture (G_o, H_o) and consistency cannot be investigated.

We need the following condition to hold for the data $\mathcal{D}_N$:

Condition 3 (*Persistence of Excitation*)**.** *Given the model set $\mathcal{M} = \{M_\theta \mid \theta \in \Theta\}$, the input sequence $\{u_\ell\}_{\ell=0}^{N-1}$ in $\mathcal{D}_N$ generated by (G_o, H_o) is weakly persistently exciting, which means that for all pairs of parameterizations given by $(\theta_1, \hat{w}_1) \in \Theta \times \mathbb{R}^{\hat{n}_w}$ and $(\theta_2, \hat{w}_2) \in \Theta \times \mathbb{R}^{\hat{n}_w}$ for which the function mapping is unequal, i.e., $V_{(\cdot)}(\theta_1, \hat{w}_1) \neq V_{(\cdot)}(\theta_2, \hat{w}_2)$, we have*

$$V_{\mathcal{D}_N}(\theta_1, \hat{w}_1) \neq V_{\mathcal{D}_N}(\theta_2, \hat{w}_2) \tag{22}$$

with probability 1.

The notion of informativity (based on which persistency of excitation is defined) of a data set implies that the data is such that, if it is used in computing the identification cost $V_{\mathcal{D}_N}$, it will reveal the differences between the parameter vector choices, see Ljung (1999). As our model structure is not identifiable in general, a reasonable expectation is that equivalence classes in the parameterization can be distinguished based on the data, i.e., we have a set of parameter vectors that correspond to the same one-step-ahead predictor. The weak persistency of excitation, i.e., Condition 3 expresses this. In the LTI case, Condition 3 is a standard assumption in most of the asymptotic analysis literature (Ljung, 1999) and it can be easily checked for general black-box models via standard persistency of excitation tests using the power spectrum of the data. For the general NL case, such computable tests do not exist and it is beyond the scope of the current paper to establish them.

To show consistency, we also require that any element of Θ_o and the corresponding initial condition $\hat{w}_o$, have minimal asymptotic cost in terms of $\lim_{N\to\infty} V_{\mathcal{D}_N}(\theta, \hat{w}_0)$. Due to the mean squared loss function and since $\mathbb{E}_e\{V_{\mathcal{D}_N}(\theta, \hat{w}_0)\}$ exists as shown in Theorem 2, this trivially holds. For a detailed proof see Ljung (1978). Then, the following result holds:

Theorem 3 (*Consistency*)**.** *Under the conditions of Theorem 2 and Condition 3,*

$$\lim_{N\to\infty} \hat{\theta}_N \in \Theta_o \tag{23}$$

with probability 1, where

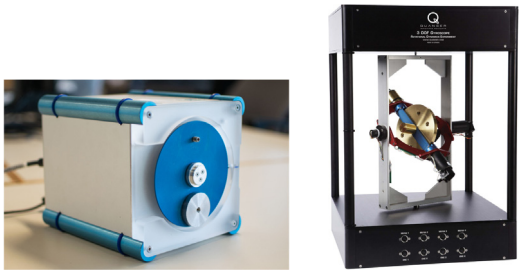
$$(\hat{\theta}_N, \hat{w}_N) = \arg\min_{\theta\in\Theta, w\in\mathbb{R}^{\hat{n}_w}} V_{\mathcal{D}_N}(\theta, w). \tag{24}$$

Proof. The proof follows from Lemma 4.1 in Ljung (1978) as the mean-squared loss function (9) fulfills Condition (4.4) in Ljung (1978). ■

Note that while Theorem 3 shows consistency of the proposed identification approach in terms of asymptotic convergence to equivalent representations of (G_o, H_o), the estimates $(\hat{\theta}_N, \hat{w}_N)$ correspond to a global minimum of $V_{\mathcal{D}_N}$ in terms of (24). As the latter is a nonlinear optimization problem, a global minimum might not be obtained in practice, despite the sophisticated optimization toolchain explained in Section 4.1.

6. Examples

In this section we demonstrate the estimation performance and computational efficiency of the proposed identification scheme on two benchmark systems, depicted in Fig. 2, utilizing simulation data. Estimation of LTI, LPV and general NL models will be accomplished, showcasing the flexibility of the overall system identification toolchain.



(a) Unbalanced disk (b) CMG system

Fig. 2. Considered benchmark systems, whose simulation model is used for identification.

6.1. Implementation and computational aspects

The proposed identification approach is implemented with the `jax-sysid` package (Bemporad, 2025) in Python 3.11 and run on a MacBook Pro with Apple M4 Max processor equipped with 16 CPU cores. Given an initialization of the optimization parameters $(\theta, \hat{w}_0)$, we report the CPU time spent on solving Problem (15) on a single core.

In all the examples, unless specified differently, we set: the regularization term (16) to $R(\theta, \hat{w}_0) = \frac{\rho_\theta}{2}\|\theta\|_2^2 + \frac{\rho_w}{2}\|\hat{w}_0\|_2^2$ with $\rho_\theta = 2 \cdot 10^{-4}$ and $\rho_w = 2 \cdot 10^{-8}$; if Adam is used for warm start, we run 10^3 iterations with parameters $\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$; L-BFGS-B optimization is run for at most 10^4 iterations; the results shown in the tables are obtained by repeating the identification procedure 100 times from different random initial guesses, selecting the model providing the highest *best fit rate* (BFR) of the predicted and simulated model response on test data in terms of (10), defined as

$$\text{BFR} = \max\left\{1 - \sqrt{\frac{\sum_{k=0}^{N-1}\|y_k - y_k^{\text{pred}}\|_2^2}{\sum_{k=0}^{N-1}\|y_k - \bar{y}\|_2^2}}, 0\right\}$$

for prediction, with $\bar{y}$ being the sample mean of the measured y, while for simulation, y_k^{pred} is substituted with y_k^{sim}, resulting when (10) is run with $\tilde{f}_z$ and $\tilde{g}_z$ set to 0. Unless specified differently, the coefficients of the deterministic model (8a) are not used as a warm start when training the combined model.

For each example, we identify the combination of plant (8a) and (inverse) noise model (8b) in the form (10) by solving (15), denoted as "`combined`" in the reported tables. For comparison, we also train a purely deterministic plant model ((8a) only) by setting $\tilde{f}_z$ and $\tilde{g}_z$ to 0 in (15), denoted as "`plant only`" in the tables. Note that (10) corresponds to the simulation error when only a plant model is trained, realizing an OE noise-based model structure. After identifying a model, the initial state is reconstructed from the dataset as described in Bemporad (2025, Section III.C) to compute output predictions. The tables also show the maximum achievable performance, denoted as "`true`". We remark that results under "`true`" mean that the original representation (G_o, H_o) of the data-generating system has been used as a model to calculate (*i*) the simulation performance in terms of subtracting the noise-free response $y_{o,k}$ of the process model G_o from the measured output y, which gives back the true v_k as the error in the BFR calculation; (*ii*) the prediction performance, which is simply $e_k = H_o^{-1}v_k$, meaning that the original noise sequence is used as the prediction error in the BFR calculations. The CPU time reported in the tables is referred to the training of the best model.

6.2. Unbalanced disk

6.2.1. Data-generating system

The unbalanced disk system, depicted in Fig. 2(a), consists of a vertically mounted disk with a lumped mass m at distance l from the center, actuated by an input voltage u. Due to the gravitational force exerted on the lumped mass, the system exhibits nonlinear dynamics, which can be represented by the following differential equation:

$$\ddot{\alpha} = -\frac{1}{\tau}\dot{\alpha} + \frac{K_m}{\tau}u - \frac{mgl}{J}\sin(\alpha), \tag{25}$$

where α is the disk angular position, τ is the lumped back EMF constant, K_m is the motor constant, J is the complete disk inertia, and g is the acceleration due to gravity. These physical parameters are given in den Boef et al. (2021). To generate discrete-time data, under *zero-order-hold* (ZOH) actuation, (25) is discretized via the Euler approach with sampling time $T_s = 0.01$s. The measured output is the angular position α corrupted by an additive noise process v_k, which will vary in the different scenarios that we will consider.

6.2.2. LTI case

To test our approach in the ideal case of data generated by a linear system, we linearize (25) around $(\dot{\alpha}, \alpha) = (0, 0)$ and discretize the resulting linear dynamics, obtaining

$$x_{k+1} = A_x x_k + B_x u_k, \tag{26a}$$
$$y_{o,k} = C_x x_k, \tag{26b}$$

with

$$A_{x,0} = \begin{bmatrix} 1 - \frac{T_s}{\tau} & 0 \\ T_s & 1 \end{bmatrix},\ A_{x,1} = \begin{bmatrix} 0 & -\frac{Mg\ell T_s}{J} \\ 0 & 0 \end{bmatrix},$$
$$A_x = A_{x,0} + A_{x,1},\ B_x = \begin{bmatrix} \frac{T_s K_m}{\tau} \\ 0 \end{bmatrix},\ C_x = \begin{bmatrix} 0 & 1 \end{bmatrix}, \tag{26c}$$

completed with the BJ noise dynamics

$$z_{k+1} = (A_{z,0} + A_{z,1})z_k + (B_{z,0} + B_{z,1})e_k, \tag{26d}$$
$$v_k = z_k + e_k, \tag{26e}$$
$$y_k = y_{o,k} + v_k, \tag{26f}$$

where $A_{z,0} = \frac{277}{300}$, $A_{z,1} = \frac{2}{30}$, $B_{z,0} = \frac{41}{150}$, $B_{z,1} = \frac{-11}{60}$. Using (26), independent training and test datasets, $N = 2000$ sample long each, are generated by using $u_k \sim \mathcal{N}(0, 1)$ and $e_k \sim \mathcal{N}(0, 3.75 \cdot 10^{-3})$, giving a *signal-to-noise ratio* (SNR) of about 10 dB on both sets. Using our proposed approach, we run L-BFGS-B, without the need of Adam warm starting, and identify an LTI plant model with $\hat{n}_x = 2$ states and $\hat{n}_z = 1$ noise-model state, corresponding to the minimal state dimensions required to describe the model dynamics in terms of (26). For comparison purposes, the subspace method `n4sid` and the gradient-based PEM-SS approach `ssest` from the System Identification Toolbox in MATLAB (Ljung, 2026) are also applied on the datasets with default settings.

The results are summarized in Table 1, where the column "type" refers to whether the BFR calculations are based on the predicted or simulated response of the model. The labels `(s)` and `(p)` for `ssest` indicate if the estimation is performed with simulation or prediction focus, respectively. Note that to capture only the process dynamics, $\hat{n}_x = 2$ is sufficient, while to capture also the noise dynamics $\hat{n}_x = 3$ is required in an innovation form used by the subspace and PEM-SS methods in MATLAB, hence estimation is performed with the required optimal state dimensions for these methods, showing the obtained results in Table 1. Based on our results, `n4sid` with its default options fails to find a good model in this case, while `ssest` achieves worse

Table 1
Identification results for the linearized unbalanced disk system (26) with BJ noise dynamics.

	$\hat{n}_x$	$\hat{n}_z$	BFR train.	BFR test	type	time
`true`	2	1	72.12% 77.66%	68.13% 72.85%	sim pred	– –
`plant only`	2	0	72.03%	68.08%	sim	0.13 s
`combined`	2 2	1 1	72.02% 77.67%	68.08% 72.84%	sim pred	0.34 s
`n4sid (s)`	2	–	0.33%	0.56%	sim	0.15 s
`n4sid (p)`	3	-	61.21% 75.63%	54.49% 70.46%	sim pred	0.11 s
`ssest (s)`	2 3	– –	1.33% 72.04%	1.37% 68.07%	sim sim	0.47 s 0.26 s
`ssest (p)`	3	-	64.23% 76.31%	58.02% 71.43%	sim pred	0.20 s

results in terms of estimation of the process part only ($n_x = 2$) than our proposed `plant only` approach with comparable computation time, while it shows similar results when adding a third state ($n_x = 3$) to our `combined` approach. This might be attributed to numerical problems in the subspace initialization of `ssest` in the $n_x = 2$ case.

6.2.3. Externally scheduled LPV case

As discussed in den Boef et al. (2021), we convert the nonlinear dynamics of the unbalanced disk system to an LPV form that we discretize, giving the LPV plant dynamics combined with a scheduling-dependent noise process in the following form:

$$x_{k+1} = \overbrace{(A_{x,0} + A_{x,1}p_k)}^{A_x(p_k)} x_k + B_x u_k, \tag{27a}$$

$$z_{k+1} = (A_{z,0} + A_{z,1}p_k)z_k + (B_{z,0} + B_{z,1}p_k)e_k, \tag{27b}$$

$$y_k = C_x x_k + \underbrace{z_k + e_k}_{v_k}, \tag{27c}$$

with $A_{x,0}, \ldots, C_x$ and $A_{z,0}, \ldots, B_{z,1}$ defined as in (26c). We assume that the scheduling signal p_k is known and exogenously generated, $p_k = (1 - p_{\text{mag}}) + p_{\text{mag}} n_k$, where $p_{\text{mag}} = 0.25$ and n_k is a random binary signal with bandwidth $\frac{0.05}{2T_s}$ Hz, while the input u and the generating noise e are independent white noise sequences with $u_k \sim \mathcal{N}(0, 1)$ and $e_k \sim \mathcal{N}(0, 3.75 \cdot 10^{-3})$ corresponding to an SNR of 10 dB. Under these conditions, a training and test dataset are generated independently with $N = 2000$ samples each. Using the proposed identification approach, we identified models with $\hat{n}_x = 2$ and $\hat{n}_z = 1$, while all matrix functions in (12a) and (12b) are assumed to depend affinely on p_k. For comparison, we used the LPVcore Toolbox for Matlab (den Boef et al., 2021) to identify an LPV state–space model of the same order using the subspace method `lpvidss` with a past window of 6 and the gradient-based PEM-SS method `lpvssest`.

Table 2 shows the results obtained by running the L-BFGS-B optimization without Adam warm starting. As it can be seen, the estimation performance of the newly proposed approach is slightly better than gradient-based PEM in LPVcore and, as expected, achieves better results than the subspace method, due to the direct minimization of the ℓ_2-loss on the prediction error. In terms of computation time, the subspace approach is faster due to the short past window taken, while the newly proposed method, implemented in Python with JAX, converges faster than `lpvssest` in Matlab.

Table 2
Identification results for the LPV form of the unbalanced disk system (27) with LPV-BJ noise dynamics and external scheduling.

	$\hat{n}_x$	$\hat{n}_z$	sched.	BFR train.	BFR test	type	time
`true`	2	1	ext.	70.69% 73.63%	72.61% 74.36%	sim pred	– –
`plant only`	2	0	ext.	70.84%	71.94%	sim	3.81 s
`combined`	2	1	ext. ext.	70.82% 73.62%	71.81% 74.19%	sim pred	5.15 s
`lpvidss`	3	-	ext.	1.33% 67.45%	1.37% 66.81%	sim pred	1.99 s
`lpvssest`	3	-	ext.	69.80% 73.37%	69.48% 72.39%	sim pred	21.78 s

Table 3
Identification results for the unbalanced disk system with input noise and using self-scheduled LPV models.

	$\hat{n}_x$	$\hat{n}_z$	sched.	BFR train.	BFR test	type	noise	time
`true`	2	1	self	70.02% 75.11%	76.04% 78.89%	sim pred	LPV LPV	– –
`plant only`	2	0	self	70.17%	75.08%	sim	–	8.56 s
`combined`	2	1	self	70.21% 75.16%	75.06% 78.29%	sim pred	LTI LTI	10.81 s
`combined`	2	1	self	70.22% 75.17%	75.15% 78.43%	sim pred	LPV LPV	15.46 s

6.2.4. Self-scheduled LPV case

We now identify a self-scheduled LPV model with $\hat{n}_x = 2$ and $\hat{n}_z = 1$, while all matrix functions in (12a) and (12b) are assumed to depend on p_k in terms of affine dependence. To generate data, (27) has been simulated with $p_k = \text{sinc}(x_1(k)) = \frac{\sin(\alpha(k))}{\alpha(k)}$, corresponding to the original nonlinear dynamics of the unbalanced disk system. Under the same data generation with $N = 2000$, $u_k \sim \mathcal{N}(0, 1)$, $e_k \sim \mathcal{N}(0, 3.75 \cdot 10^{-3})$, the obtained estimation and test data sets had an SNR of 21 dB. For the scheduling map ψ, a feedforward ANN with two hidden layers is used, with 8 nodes in each layer and *sigmoid* (first layer) and *swish* (second layer) activation functions. In total, the plant model has 121 parameters to learn, while the combined plant and noise model has 124 (LTI noise model) and 127 (LPV noise model) parameters. The results achieved using 1000 Adam iterations to warm start the L-BFGS optimization are summarized in Table 3, where the reported time includes the entire time required for training each model, including the warm-starting phase. As we can see, including the disturbance model ($\hat{n}_z = 1$) is beneficial, and, moreover, a simple LPV noise model provides slightly better results than an LTI noise model.

For the case of identification of a self-scheduled LPV model (`combined`), the sample variance of $\hat{v}_k$ and $\hat{e}_k$ on test data are, respectively, $\text{Var}\{\hat{v}_k\} = 5.9959 \times 10^{-3}$, $\text{Var}\{\hat{e}_k\} = 4.3418 \times 10^{-3}$. shows the one-step-ahead predictions obtained by the `combined` model with LTI noise model for the first 200 samples of the test data. It is apparent how the noise model has tried to capture the acceleration noise. The plot also shows the evolution $\hat{v}_k$ and $\hat{e}_k$, while their power spectral density is shown in Fig. 3(b), where it can be observed that the reconstructed noise $\hat{e}_k$ has less low-frequency content than the estimate $\hat{v}_k$ of the output error $v_k = y_k - y_{o,k}$.

6.2.5. Nonlinear case

For the data set generated in Section 6.2.4, we identify a nonlinear state–space plant model (8a) in which $f_x = [x_1 + \frac{1}{10}x_2 \; f_{x2}]'$ and f_{x2} is a two-layer ANN with n_1 and n_2 neurons,

Table 4
Identification results for the altered unbalanced disk system with input noise and using NL models.

	$\hat{n}_x$	$\hat{n}_z$	n_1	n_2	n_θ	BFR train.	BFR test	type	time
`true`	2	1				70.02% 75.11%	76.04% 78.89%	sim pred	– –
`plant only`	2	0	3	3	41	70.90%	73.25%	sim	8.15 s
`combined`	2	1	3	3	44	70.16% 75.18%	74.79% 78.08%	sim pred	9.36 s
`plant only`	2	0	8	8	141	70.96%	70.74%	sim	16.54 s
`combined`	2	1	8	8	144	70.51% 75.29%	74.94% 78.30%	sim pred	11.74 s
`plant only`	2	0	10	10	195	71.15%	69.14%	sim	19.29 s
`combined`	2	1	10	10	198	70.67% 75.34%	74.01% 77.65%	sim pred	22.24 s
`plant only`	2	0	15	15	365	71.26%	67.85%	sim	25.89 s
`combined`	2	1	15	15	368	70.91% 75.46%	73.27% 76.86%	sim pred	25.13 s

swish activation function, and linear bypass of x and u in all layers, while f_y is a tunable linear function of x. The noise model is a first-order LTI system ($\hat{n}_z = 1$). The results obtained for different values of n_1, n_2 are summarized in Table 4. We can see that the proposed approach can be easily used to scale up the model estimation to the NL dynamics, and can reliably capture the underlying behavior of the system. Note that the simulation performance on training data of `plant only` increases with n_1 and n_2, at the price of a decrease in the test performance, due to overfitting. However, it is apparent the benefit of using the noise model ($\hat{n}_z = 1$), as performance significantly improves on test data. Note that the simulation performance on training data of `plant only` is slightly better than `true`; this is possibly due to the extra degree of freedom the training algorithm has in optimizing also the initial state $\hat{x}_0$.

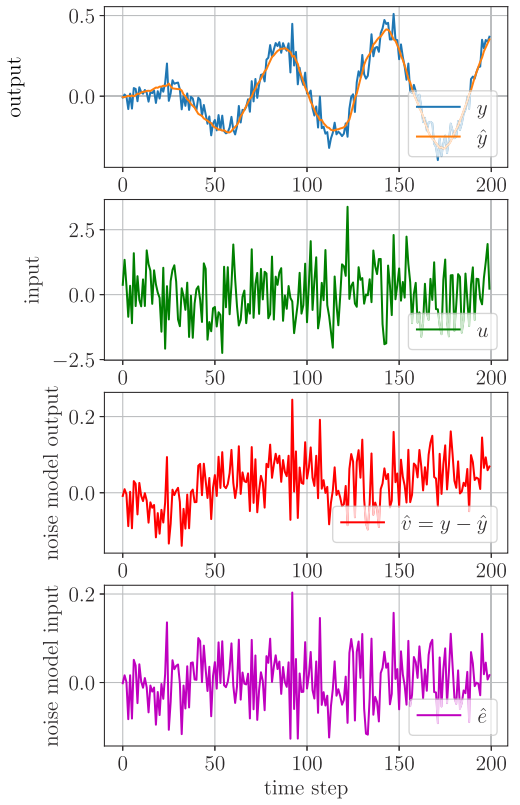
6.3. Control moment gyroscope

We consider a high-fidelity NL simulator of the 3 degrees-of-freedom *Control Moment Gyroscope* (CMG) depicted in Fig. 2(b).

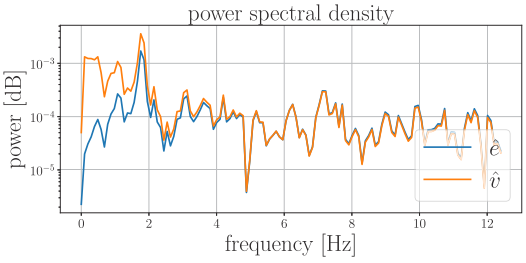
6.3.1. Data-generating system

The system consists of a flywheel (with angular position q_1), mounted inside an inner blue gimbal (with angular position q_2), which in turn is mounted inside an outer red gimbal (with angular position q_3). The entire structure is supported by a rectangular silver frame (with angular position q_4) that can rotate around its vertical axis of symmetry. The gimbals can rotate freely. The setup is equipped with four DC motors and encoders, actuating and measuring the position of the flywheel and the gimbals, respectively. The nonlinear system is highly complex due to the involved rotational dynamics.

In this study, we use the high-fidelity CMG simulation model (Bloemers & Tóth, 2019) as the data-generating system, which is implemented in C and interfaced to MATLAB. We consider a scenario in which the red gimbal is locked ($q_3 \equiv 0$), the blue gimbal (q_2) is actuated with motor current i_2 (treated as the manipulated input), while $\dot{q}_4$ is the measured output, corrupted by a white noise signal $e_k \sim \mathcal{N}(0, \sigma_e^2)$ with variance $\sigma_e^2 = 2.2 \cdot 10^{-5}$. The angular velocity $\dot{q}_1$ of the flywheel is controlled independently to follow a random multi-level-reference signal with amplitude in [30, 50] rad/s and a dwell-time between 4 and 8 s.



(a) First 200 samples of one-step-ahead predictions



(b) Power spectral density of the reconstructed noise $\hat{e}_k$ and $\hat{v}_k$ at low frequencies

Fig. 3. Prediction results of the self-scheduled LPV model with LTI noise model identified for the unbalanced disk.

The CMG is simulated in continuous time with fixed-step Runge–Kutta 4 under ZOH, sampling the results every $T_s = 0.01$ s to obtain the training and test data sets. By analyzing the first-principle dynamics of the system (see, e.g., Bloemers & Tóth, 2019), it can be noted that, due to the independent control of the flywheel and the locked status of q_3, the angular velocity $\dot{q}_1$ can be regarded as an external input to a model describing the motion of q_2 and q_4. Hence, in the considered scenarios, $\dot{q}_1$ is treated either as an additional input signal (LTI, self-scheduled LPV, and NL case) or a scheduling variable (externally-scheduled LPV case).

6.3.2. LTI case

Using the simulator with the configuration described in Section 6.3.1, a training dataset of size $N = 10^4$ and test dataset of size $N = 3 \cdot 10^4$ were generated by exciting i_2 with the experiment design detailed in Verhoek et al. (2022, Sec. IV.B), giving an SNR of 35 dB on both data sets. We identify LTI models of the nonlinear dynamics with inputs $u_1 = i_2$, $u_2 = \dot{q}_1$.

Table 5
Identification results for the CMG system (Section 6.3) with LTI models.

	$\hat{n}_x$	$\hat{n}_z$	BFR train.	BFR test	type	time
`true`	5	–	98.16%	98.19%	sim	–
`plant only`	8	0	35.99%	25.28%	sim	3.50 s
`combined`	8 8	2 2	34.46% 97.17%	29.91% 97.12%	sim pred	8.73 s
`n4sid`	8	0	29.72%	20.98%	sim	0.85 s
`n4sid`	10	0	34.76%	22.45%	sim	1.05 s
`ssest (s)`	8	0	35.48%	24.70%	sim	2.05 s
`ssest (p)`	10	0	33.51%	26.88%	sim	3.06 s

Using model-structure selection, the plant model order $\hat{n}_x = 8$ was established as a good choice while $\hat{n}_z = 2$ was taken to test overfitting as, in this case, the data-generating system has an OE noise structure, meaning that the estimation cannot benefit from a noise model. For comparison purposes, `n4sid` with focus on simulation and `ssest` with focus on simulation (`s`) and prediction (`p`) were also executed on the data set using $\hat{n}_x = 8$ (corresponding to `plant only`) and $\hat{n}_x = 10$ (corresponding to `combined`) as model order. The results, obtained by running L-BFGS without Adam warm start and $\rho_\theta = 2 \cdot 10^{-9}$, are shown in Table 5.

As can be seen from the table, the proposed approach results in a better fit on the training data than `n4sid` or `ssest` when only a plant model is estimated, as well as on test data, except for `ssest (p)`. When both a plant and a noise model are estimated, the best fit on test data is achieved by the proposed method, without excessively sacrificing the fit on training data.

6.3.3. LPV case

We use the same simulation setting to identify LPV models. The plant model has $\hat{n}_x = 5$ states and affine dependence on p, while the noise model is taken LTI with $\hat{n}_z = 2$ states. To make a fair comparison with results previously obtained on this dataset in Verhoek et al. (2022), for the self-scheduled LPV case we choose $n_p = 3$ and parameterize the scheduling map ψ as a feedforward ANN with two hidden layers of 5 neurons each with a *swish* activation function. The plant and the combined plant + noise models have 281 and 289 parameters, respectively. For the externally-scheduled case, the scheduling variable p_k is taken from measured data by setting $p = \psi(x,u) = [\dot{q}_1 \ \sin(q_2) \ \cos(q_2)]^\top$ and the plant and the combined plant + noise models have 185 and 193 parameters, respectively.

For initialization of the proposed identification scheme, we use a bootstrapping process by first training an LTI plant model, use it to initialize the training of an LPV plant model (`plant only`), which is in turn used to initialize the training of the full model including a noise model (`combined`). Note again that, due to the OE noise structure of the data-generating system, the noise model is taken only to test overfitting by the proposed estimation scheme. The results obtained by training the models, with $\rho_w = 2 \cdot 10^{-4}$ and $\rho_\theta = 2 \cdot 10^{-8}$ (externally-scheduled case), and $\rho_\theta = 2 \cdot 10^{-3}$ (self-scheduled case), are summarized in Table 6. For comparison purposes, the table also shows the results obtained in Verhoek et al. (2022) using the input–output model structure based PEM approach `lpvoe` with a 5$^{\text{th}}$ order LPV-OE input–output model ($n_a = n_b = 5$) and the gradient-based PEM-SS method `lpvssest` with $\hat{n}_x = 5$ and no noise model estimation implemented in LPVcore. Next to this, the LPV variant of the SUBNET method has been also applied to identify a self-scheduled LPV model with $\hat{n}_x = 5$ and $n_p = 3$ using a scheduling map parameterization with 2 hidden layers, each with 64 neurons, tanh activation functions, and a linear bypass.

Table 6
Identification results for the CMG system (Section 6.3) with LPV models.

	$\hat{n}_x$	$\hat{n}_z$	sched.	BFR train.	BFR test	type	time
`true`	5	–	–	98.16%	98.19%	sim	–
`plant only`	5	0	self ext.	97.61% 98.14%	96.50% 98.13%	sim	47.54 s 22.65 s
`combined`	5	2	self	81.96% 97.56%	83.78% 97.64%	sim pred	67.19 s
`combined`	5	2	ext.	98.12% 98.11%	98.11% 98.10%	sim pred	31.61 s
`lpvoe`	5	0	ext.	93.32%	89.96%	sim	11.12 s
`lpvssest`	5	0	ext.	93.62% 94.91%	90.84% 93.22%	sim pred	348 s
SUBNET	5	0	self. ext.	97.28% 97.25%	96.40% 97.01%	sim	≈10 h ≈10 h

As we can see from Table 6, the proposed approach with only process model estimation achieves better results than the state-of-the-art SUBNET approach, both for the self and externally scheduled cases. Actually, it comes nearly to the performance of the true model (theoretical maximum) in the externally scheduled case and accomplishes the estimation in a fraction (≈0.2%) of the required training time of SUBNET, which is a remarkable achievement. Note that no other approach is currently capable of combined estimation of the scheduling map and the LPV model than the proposed method and the SUBNET approach. When compared to traditional LPV approaches working with a user defined scheduling map such as `lpvoe` and `lpvssest`, it is obvious that co-estimation of the scheduling map and the plant model gives clear advantages for both the proposed method and SUBNET.

When a noise model is added, in the self-scheduled case the estimated model results in worse simulation performance compared to the `plant only` case. However, in case of external scheduling, the performance is close to that of the `plant only` case and the noise model is consistently estimated to be approximately zero.

6.3.4. Nonlinear case

Finally, we identify a fully nonlinear model using the same simulator and configuration for the CMG system described above. We let f_x be the sum of a linear function $Ax_k + Bu_k$ and a two-layer FNN with 10 neurons each and f_y the sum of a linear function Cx_k and a two-layer FNN with 5 neurons each, both with *swish* activation functions, with $\hat{n}_x = 5$ states, and set $\rho_\theta = 2 \cdot 10^{-3}$, $\rho_w = 2 \cdot 10^{-4}$. The noise model is LTI with $\hat{n}_z = 2$ states. Again, the same bootstrapping process is used for initialization as discussed in Section 6.3.3. We freeze the model matrices A, B, C and train the remaining 311 parameters of the deterministic plant model by setting $\rho_w = 2 \cdot 10^{-3}$. After training the plant model, this is frozen and the remaining 8 parameters of the noise model are trained to get the combined plant+noise model. The results obtained by training the models from 500 initial guesses, with $\rho_w = 2 \cdot 10^{-4}$, are summarized in Table 7. We can see that there is a slight improvement on training data with respect to the LPV self-scheduled structure when only estimating a process model, and an improvement on both training and test data in the case a combined model (`combined`) is used.

7. Conclusions

This paper has introduced a novel general identification approach that is based on a theoretically well-founded separate parameterization of the deterministic process and stochastic noise dynamics in the model structure. This not only allows the extension of the estimation concept of the LTI framework to general

Table 7
Identification results for the CMG system (Section 6.3) with NL models.

	$\hat{n}_x$	$\hat{n}_z$	BFR train.	BFR test	type	time
true	5	–	98.16%	98.19%	sim	–
plant only	5	0	96.75%	96.12%	sim	42.10 s
combined	5	2	96.66%	96.12%	sim	47.78 s
	5	2	97.82%	97.84%	pred	

nonlinear problems, but also the treatment of the estimation of a whole spectrum of process and noise models from LTI, to LPV and NL dynamics and various noise scenarios encoded by the noise model, by seamlessly tuning the complexity of the model estimates in both of these aspects.

Thanks to the use of efficient L-BFGS nonlinear optimization and auto-differentiation libraries, the proposed method has been demonstrated to exhibit high computational efficiency in estimating nonlinear dynamics through ANN-based parameterization of the functional relationships, compared to other (deep)-learning-based ANN methods. Besides theoretical guarantees of asymptotic consistency, the introduced approach has been shown to provide equivalent and often superior quality of fit compared to many existing state-of-the-art methods for LTI, LPV, and NL identification, with limited needs of hyperparameter tuning. We believe that our proposed work opens the door to a new generation of system identification methods, capable of handling a broad spectrum of model classes while maintaining the simplicity and computational efficiency users are familiar with within the LTI framework.

Acknowledgments

The authors would like to thank Gerben Beintema, Maarten Schoukens, and Jonas Sjöberg for fruitful discussions on the results.

Appendix. Proof of Theorem 1

By taking $w_k = [\, x_k^\top \;\; z_k^\top \,]^\top$ and $f_w = [\, f_x^\top \;\; f_z^\top \,]^\top$ with $g_w = [\, g_x^\top \;\; g_z^\top \,]^\top$, (2a)–(2c) can always be written in the form of (1). For the reverse statement, define a state vector x_k with $n_x = n_w$ components and its state-update function $f_x : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ as

$$x_{k+1} = f_x(x_k, u_k) \triangleq f_w(x_k, u_k, 0), \tag{A.1}$$

with $x_0 = w_0$. Then, it follows that

$$w_{k+1} - x_{k+1} = f_w(w_k, u_k, e_k) - f_x(x_k, u_k) \tag{A.2}$$

for all $k \in \mathbb{Z}_0^+$. Now, let us define the state $z_k \triangleq w_k - x_k$ of the inverse noise model, with $z_0 = 0$ and $n_z = n_w$, and its state-update function $f_z : \mathbb{R}^{n_z} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_z}$ as

$$\begin{aligned} z_{k+1} &= f_z(z_k, x_k, u_k, e_k) \\ &\triangleq f_w(z_k + x_k, u_k, e_k) - f_w(x_k, u_k, 0). \end{aligned} \tag{A.3}$$

Similarly, take

$$y_{o,k} = g_x(x_k, u_k) \triangleq g_w(x_k, u_k) \tag{A.4}$$

and define the output function $g_z(z_k, x_k, u_k) \triangleq g_w(z_k + x_k, u_k) - g_w(x_k, u_k)$ such that

$$\begin{aligned} v_k &= y_k - y_{o,k} \\ &= g_w(z_k + x_k, u_k) + e_k - g_w(x_k, u_k) \\ &= g_z(z_k, x_k, u_k) + e_k. \end{aligned} \tag{A.5}$$

Note that $f_z(0, ., ., 0) \equiv 0$, as by (A.3),

$$f_z(0, x_k, u_k, 0) = f_w(0 + x_k, u_k, 0) - f_w(x_k, u_k, 0) = 0, \tag{A.6}$$

and $g_z(0, ., .) \equiv 0$, as by (A.5), $g_w(0 + x_k, u_k) - g_w(x_k, u_k) = 0$.

This concludes the proof with $n_x = n_z = n_w$. Note that such a state construction can result in a non-minimal state realization for G_o and H_o. Hence, the corresponding realizations might be further reduced by finding invertible state-transformations with $n_x, n_z \le n_w$ such that the input/output maps corresponding to G_o and H_o remain unchanged. ■

References

Beintema, G. I., Schoukens, M., & Tóth, R. (2023). Deep subspace encoders for nonlinear system identification. *Automatica*, *156*, Article 111210.

Bemporad, A. (2023a). Recurrent neural network training with convex loss and regularization functions by extended Kalman filtering. *IEEE Transactions on Automatic Control*, *68*(9), 5661–5668.

Bemporad, A. (2023b). Training recurrent neural networks by sequential least squares and the alternating direction method of multipliers. *Automatica*, *156*, Article 111183.

Bemporad, A. (2025). An L-BFGS-B approach for linear and nonlinear system identification under ℓ_1- and group-lasso regularization. *IEEE Transactions on Automatic Control*, *70*(7), 4857–4864.

Billings, S. A. (2013). *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Wiley.

Bloemers, T., & Tóth, R. (2019). *Equations of motion of a control moment gyroscope: Technical report*, Eindhoven University of Technology.

den Boef, P., Cox, P. B., & Tóth, R. (2021). LPVcore: MATLAB toolbox for LPV modelling, identification and control of non-linear systems. In *Proc. of the 19th IFAC symposium system identification: learning models for decision and control* (pp. 385–390). Padova, Italy.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208.

Candès, E. J., Wakin, M. B., & Boyd, S. P. (2008). Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, *14*(5), 877–905.

Donati, C., Mammarella, M., Dabbene, F., Novara, C., & Lagoa, C. (2025). A scalable, gradient-stable approach to multi-step, nonlinear system identification using first-order methods. In *Proc. of the 11th IFAC symposium on robust control design* (pp. 37–42).

Forgione, M., & Piga, D. (2020). Model structures and fitting criteria for system identification with neural networks. In *Proc. of the 14th IEEE international conference on application of information and communication technologies* (pp. 1–6). Tashkent, Uzbekistan.

Forgione, M., & Piga, D. (2021). dynoNet: A neural network architecture for learning dynamical systems. *International Journal of Adaptive Control and Signal Processing*, *35*(4), 612–626.

Giri, F., & Bai, E.-W. (2010). Lecture notes in control and information sciences. In *Block-oriented nonlinear system identification*, Springer-Germany.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proc. of the 13th int. conference on artificial intelligence and statistics* (pp. 249–256). JMLR Workshop and Conference Proceedings.

Gonzalez, J., & Yu, W. (2018). Non-linear system modeling using LSTM neural networks. In *Proc. of the 2nd IFAC conference on modelling, identification and control of nonlinear systems* (pp. 485–489).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Griewank, Andreas, & Walther, Andrea (2008). *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM.

Hjalmarsson, H., & Sjöberg, J. (2012). A Mathematica toolbox for signals, models and identification. *IFAC Proceedings Volumes*, *45*(16), 1541–1546.

Jansson, M. (2003). Subspace identification and ARX modeling. *IFAC Proceedings Volumes*, *36*(16), 1585–1590.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Ljung, L. (1978). Convergence analysis of parametric identification methods. *IEEE Transactions on Automatic Control*, *23*(5), 770–783.

Ljung, L. (1999). *System Identification, theory for the user* (2nd ed.). Prentice-Hall.

Ljung, L. (2026). *System identification toolbox for MATLAB*. The Mathworks, Inc., https://www.mathworks.com/help/ident.

Ljung, L., Andersson, C., Tiels, K., & Schön, T. B. (2020). Deep learning and system identification. In *Proc. of the 21th IFAC world congress* (pp. 1175–1181).

Masti, D., & Bemporad, A. (2021). Learning nonlinear state–space models using autoencoders. *Automatica*, *129*, Article 109666.

Mulagaleti, S. K., & Bemporad, A. (2026). Combined learning of linear parameter-varying models and robust control invariant sets. *IEEE Transactions on Automatic Control*, in press.
Niedźwiecki, M. (2000). *Identification of time-varying processes*. John Wiley and Sons.
Pillonetto, G., Aravkin, A., Gedon, D., Ljung, L., Ribeiro, A. H., & Schön, T. B. (2025). Deep networks for system identification: A survey. *Automatica*, *171*, Article 111907.
Pillonetto, G., Dinuzzo, F., Chen, T., Nicolao, G. De, & Ljung, L. (2014). Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, *50*(3), 657–682.
Revay, Max, Wang, Ruigang, & Manchester, Ian R. (2024). Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness. *IEEE Transactions on Automatic Control*, *69*(5), 2855–2870.
Ribeiro, A. H., Tiels, K., Umenberger, J., Schön, T. B., & Aguirre, L. A. (2020). On the smoothness of nonlinear system identification. *Automatica*, *121*, Article 109158.
Rojas, C., Tóth, R., & Hjalmarsson, H. (2015). Sparse estimation of polynomial and rational dynamic models. *IEEE Transactions on Automatic Control*, *59*, 2962–2977.
Schoukens, J., & Ljung, L. (2019). Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, *39*(6), 28–99.
Sjöberg, J. (1997). On estimation of nonlinear black-box models: how to obtain a good initialization. In *Proc. IEEE workshop on neural networks for signal processing* (pp. 72–81). Amelia Island Plantation, FL, USA.
Sjöberg, J. (2000). A nonlinear grey-box example using a stepwise system identification approach. In *Proc. of the 11th IFAC symposium on system identification*. Santa Barbara, CA, USA.
Sjöberg, J., & Hjalmarsson, H. (2009). A system, signals and identification toolbox in Mathematica with symbolic capabilities. *IFAC Proceedings Volumes*, *42*(10), 747–751.
Sjöberg, J., & Raedt, P. De (1997). Nonlinear system identification: a software concept and examples. *IFAC Proceedings Volumes*, *30*(11), 657–662.
Suykens, J. A. K., Gestel, T. Van, Brabanter, J. De, Moor, B. De, & Vandewalle, J. (2002). *Least squares support vector machines*. World Scientific.
Suykens, J. A. K., Vandewalle, J. P. L., & Moor, B. De (1996). Nonlinear system identification using neural networks. In *Artificial neural networks for modelling and control of non-linear systems*. Springer.
Tóth, R. (2010). Modeling and identification of linear parameter-varying systems. In *Lecture notes in control and information sciences, vol. 403*, Heidelberg: Springer.
Verhoek, C., Beintema, G. I., Haesaert, S., Schoukens, M., & Tóth, R. (2022). Deep-learning-based identification of LPV models for nonlinear systems. In *Proc. of the 61st IEEE conference on decision and control* (pp. 3274–3280). Cancun, Mexico.
Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *67*(2), 301–320.



Alberto Bemporad received his M.S. degree cum laude in Electrical Engineering (1993) and his Ph.D. in Control Engineering (1997) from the University of Florence, Italy. He held research positions at Washington University in St. Louis (1996–1997) and ETH Zurich (1997–2002). He served at the University of Siena (1999–2009) and at the University of Trento (2010–2011). Since 2011, he has been Full Professor at the IMT School for Advanced Studies Lucca, where he was Director from 2012 to 2015. He co-founded ODYS S.r.l. in 2011, a company specialized in industrial model predictive control systems, and has held visiting appointments at Stanford University, the University of Michigan, and Zhejiang University. He has authored over 400 publications and 21 patents in model predictive control, hybrid systems, numerical optimization, nonlinear system identification, and automotive control, and co-developed the Model Predictive Control Toolbox (MathWorks) for MATLAB. He was an Associate Editor of the IEEE Transactions on Automatic Control during 2001–2004 and Chair of the IEEE CSS Technical Committee on Hybrid Systems during 2002–2010. He is an IEEE Fellow (since 2010) and IFAC Fellow (since 2025). His awards include the IFAC High-Impact Paper Award (2011–14), IEEE CSS Transition to Practice Award (2019), SAE Environmental Excellence in Transportation Award (2021), the Beale–Orchard-Hays Prize (2024), and an ERC Advanced Grant (2024).



Roland Tóth received his Ph.D. degree with Cum Laude distinction at the Delft University of Technology (TUDelft) in 2008. He was a post-doctoral researcher at TUDelft in 2009 and at the Berkeley Center for Control and Identification, University of California in 2010. He held a position at TUDelft in 2011-12, then he joined to the Control Systems (CS) Group at the Eindhoven University of Technology (TU/e). Currently, he is a Full Professor at the CS Group, TU/e and a Senior Researcher at the Systems and Control Laboratory, HUN-REN Institute for Computer Science and Control (SZTAKI) in Budapest, Hungary. He is Senior Editor of the IEEE Transactions on Control Systems Technology and Associate Editor of Automatica. His research interests are in identification and control of linear parameter-varying (LPV) and nonlinear systems, developing data-driven and machine learning methods with performance and stability guarantees for modeling and control, model predictive control and behavioral system theory. On the application side, his research focuses on advancing reliability and performance of precision mechatronics and autonomous robots/vehicles with nonlinear, LPV and learning-based motion control. He is an IEEE Fellow (since 2026). He has received the TUDelft Young Researcher Fellowship Award in 2010, the VENI award of The Netherlands Organization for Scientific Research in 2011, the Starting Grant of the European Research Council in 2016 and the DCRG Fellowship of Mathworks in 2022.