

Combining Data Assimilation and Machine Learning to build data-driven models for unknown long time dynamics - Applications in cardiovascular modeling

Francesco Regazzoni, Dominique Chapelle, Philippe Moireau

▶ **To cite this version:**

Francesco Regazzoni, Dominique Chapelle, Philippe Moireau. Combining Data Assimilation and Machine Learning to build data-driven models for unknown long time dynamics - Applications in cardiovascular modeling. International Journal for Numerical Methods in Biomedical Engineering, 2021, 37 (7), 10.1002/cnm.3471 . hal-03207037v2

HAL Id: hal-03207037
https://inria.hal.science/hal-03207037v2

Submitted on 8 Jun 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

Check for updates

Received: 7 January 2021 | Revised: 12 March 2021 | Accepted: 23 April 2021

DOI: 10.1002/cnm.3471

RESEARCH ARTICLE - FUNDAMENTAL

WILEY

Combining data assimilation and machine learning to build data-driven models for unknown long time dynamics—Applications in cardiovascular modeling

Francesco Regazzoni[1,2,3] | **Dominique Chapelle**[2,3] | **Philippe Moireau**[2,3]

[1]MOX—Mathematics Department, Politecnico di Milano, Milano, Italy

[2]M3DISIM, Institut National de Recherche en Informatique et en Automatique, Palaiseau, France

[3]LMS, Ecole Polytechnique, CNRS, Institut Polytechnique de Paris, Palaiseau, France

Correspondence
Francesco Regazzoni,
MOX—Mathematics Department, Politecnico di Milano, Milano, Lombardia, Italy.
Email: francesco.regazzoni@polimi.it

Abstract

We propose a method to discover differential equations describing the long-term dynamics of phenomena featuring a multiscale behavior in time, starting from measurements taken at the fast-scale. Our methodology is based on a synergetic combination of data assimilation (DA), used to estimate the parameters associated with the known fast-scale dynamics, and machine learning (ML), used to infer the laws underlying the slow-scale dynamics. Specifically, by exploiting the scale separation between the fast and the slow dynamics, we propose a decoupling of time scales that allows to drastically lower the computational burden. Then, we propose a ML algorithm that learns a parametric mathematical model from a collection of time series coming from the phenomenon to be modeled. Moreover, we study the interpretability of the data-driven models obtained within the black-box learning framework proposed in this paper. In particular, we show that every model can be rewritten in infinitely many different equivalent ways, thus making intrinsically ill-posed the problem of learning a parametric differential equation starting from time series. Hence, we propose a strategy that allows to select a unique representative model in each equivalence class, thus enhancing the interpretability of the results. We demonstrate the effectiveness and noise-robustness of the proposed methods through several test cases, in which we reconstruct several differential models starting from time series generated through the models themselves. Finally, we show the results obtained for a test case in the cardiovascular modeling context, which sheds light on a promising field of application of the proposed methods.

KEYWORDS

artificial neural networks, cardiovascular modeling, data assimilation, data-driven modeling, machine learning, multiscale problems

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.
© 2021 The Authors. *International Journal for Numerical Methods in Biomedical Engineering* published by John Wiley & Sons Ltd.

1 | INTRODUCTION

Mathematical models are built upon two types of sources, namely a priori information (physics principles, simplifying assumptions, empirical laws, etc.) and experimental data. The relative weight of the two sources should be properly balanced: the fewer data are available, the stronger a priori assumptions are needed. By moving towards the Big Data era, mathematical models are fed with more and more data, reducing the necessity of strong a priori assumptions. The asymptote of this process is *data-driven modeling*, that is to say the construction of a mathematical model for a given phenomenon solely on the basis of experimental data, as opposed to *physics-based modeling*, that on the other hand makes use of data only to tune the parameters appearing in equations written on the basis of a detailed study of the phenomenon by the modeler. Between pure physics or pure data, hybrid approaches allow to circumvent the uncertainties inherent to physics-based models by making full use of the data available on the system. This is particularly the case when developing estimation strategies of dynamical models,[1,2] with a particular concern in identification—that is, parameter estimation—of such systems.[3,4,5,6] Note that these strategies are also known as data assimilation (DA) approaches in the engineering community since their introduction in environmental sciences in the late 70s.[7]

Data-driven modeling appears particularly advantageous when traditional modeling encounters limitations. A representative case is given by systems whose behavior is well understood and accurately modeled, but for which the model parameters themselves feature a long-term evolution that is difficult to predict because the underlying mechanisms are not fully elucidated. This motivates the idea of resorting to a data-driven approach to apprehend the slow-scale parameter evolution, while still relying on the mathematical model at the fast scale. This is the specific focus of this paper. Examples are given by the long-term remodeling of biological systems and by the development of diseases under the influence of factors such as lifestyle or environmental conditions that are hardly within reach of mathematical modeling. In Section 1.3 we consider an example model inspired by the development of hypertension.

Data-driven modeling poses the problem of finding the laws governing the evolution of a given phenomenon (a natural phenomenon, an engineering process, a system of agents, etc.) starting from its observation. Here comes the challenge of leveraging the information contained in large amounts of data to extract knowledge, by finding the principles hidden in data.

In the past few years, several algorithms have been proposed to automate the process—historically a human prerogative—of discovering the laws hidden in experimental data. In this paper we consider data-driven modeling of time-dependent phenomena. With respect to most machine learning (ML) techniques that are designed to learn the steady-state relationship between two quantities, the introduction of the time variable dramatically increases the complexity of the problem since the object to be learned is not a function, but, typically, a differential equation.

1.1 | Learning time-dependent models from data

Data-driven algorithms capable of building black-box time-dependent models have been developed with the two following goals:

- *Data-driven modeling*: in this case data come from experimental measurements of a given phenomenon that one wants to understand and possibly predict (this is the case considered in this paper).
- *Data-driven model order reduction (MOR)*: in this case data are generated by an high-fidelity model and the goal is to obtain a different model (hopefully with fewer variables and with a lower computational complexity) reproducing—up to some error—the same input–output map.[8,9,10,11]

In most cases, the algorithms developed for one goal are also suited for the other one. Hence, in this section, we review the algorithms proposed in the literature for either goal.

Symbolic regression[12,13,14] is a technique to find a model, written as a differential equation in symbolic form, by means of an iterative process consisting in the generation of candidate symbolic differential equations and in the selection of the best ones.

In Peherstorfer et al.[15] the authors propose a data-driven MOR technique for systems with linear state dependence, based on the idea of inferring the frequency-domain response of the linear system from a collection of trajectories and to apply the Loewner framework[16] to interpolate the transfer function measurements at a collection of interpolation points.

Sparse Identification of Nonlinear Dynamics (SINDy, see Brunton et al[17]) is an algorithm that infers a dynamical system from a collection of measurements on the system—the state, the time derivative of the state and the input at different times—under the assumption that the right-hand side of the model depends on a few combination of predetermined functions of the input and the state (linear combinations, products and trigonometric functions). In Costello and Martin[18] a similar ML technique is employed to predict metabolic pathway dynamics from time series multiomics data. In Wang et al.[19] a nonlinear differential equation is learned from data, by assuming a polynomial form with sparse coefficients by compressive sensing. In Bocquet et al.[20] the authors proposed a method to infer an ODE (ordinary differential equation) representation of a model from time-dependent output data. The right-hand side of the ODE is represented as a linear combination of multivariate polynomials of (at most) quadratic order in each state variable.

In Daniels and Nemenman[21] an S-system (a class of models inspired by the mass-action kinetics) is trained by Bayesian model selection to learn a time-dependent dynamics.

Dynamic mapping Kriging (DMK, see Hernandez and Gallivan[22]) is an extension to the time-dependent case of Kriging[23] or Gaussian process (GP) regression.[24,25] Kriging is a regression method that uses a GP, with a covariance function dependent on hyper-parameters to be tuned from data, as prior for the outcome of a function. DMK consists in performing Kriging on a difference equation, where the independent variables are the state and the input of a system at the current time step, and the dependent variable is the state at the next time step.

In Regazzoni et al.[26] a technique based on artificial neural networks (ANNs) to learn a time-dependent differential equation from a collection of input–output pairs has been proposed. The authors also provided a universal approximation result stating that these ANN models can approximate any time-dependent model with arbitrary accuracy. In Regazzoni et al.[27] this method has been extended by informing the learning machine with some a priori knowledge on the system to learn, moving from a purely black-box framework to a semi-physical (or gray-box) one. A similar ANN model is proposed in Chen et al.,[28] where an efficient algorithm for sensitivities computation is discussed.

Learning algorithms for data-driven discovery of PDEs, either based on ANNs (see, e.g., Raissi et al.[29,30,31]) or GP[32,33] have been proposed. Those methods require the knowledge of the form of the equations and are thus better suited for the identification of parameters. In Raissi et al.[34] the authors proposed an ANN-based algorithm to discover nonlinear time-dependent models from a collection of snapshots of the state of the system, by minimizing the residual of some given multi-step time-stepping scheme. In Zhu et al.[35] the authors proposed an ANN-based method to build a surrogate model, using the physical knowledge as a constraint. In Tartakovsky et al.[36] the method of Raissi et al.[31] has been applied to the identification of the constitutive relationship underlying a PDE, starting from snapshots of the solution.

1.2 | Accounting for inter-individual variability

The above mentioned methods seek a unique model capable of describing the evolution of some quantities, based on a training set composed of observed trajectories. In many applications, however, each trajectory is referred to a different individual of some group (e.g., human beings). Since each individual is different, one should thus look for a *family* of models, rather than a unique model; or, from another perspective, for a single model dependent on some parameters, accounting for inter-individual variability.

Whereas parametric models are widely used in traditional physics-based modeling, the introduction of parameters raises conceptual difficulties when dealing with data-driven modeling since, in a black-box framework, parameters cannot be measured. Moreover, assuming that the learning algorithm is capable of assigning to each training individual a parameter value, then the problem of interpretability comes into play. What is the physical meaning of such learned parameters? The lack of interpretability also hampers the use of the learned models for predictive purposes since, without a clear physical meaning of the parameters, one cannot easily determine the parameters associated with an individual not included in the training set.

In this paper we show how a wise interplay between ML and DA techniques can help to deal with those issues. We define in such a way a framework to perform data-driven modeling of phenomena featuring inter-individual variability, without renouncing to interpretability and predictivity of the learned model.

1.3 | A motivating example: arterial network remodeling

To motivate the approach and to help fix the ideas with the notation, we provide in this section an example that can be treated with the framework proposed in this paper. In Section 3.4 we consider again this example, by showing some numerical results. Specifically, we consider the slow development of diseases related to the remodeling of the arterial network (such as hypertension). Even if the dynamics at the fast scale (order of seconds) of the circulation of blood through the network has been deeply studied and relatively well understood (different mathematical models, from lumped-parameters models[37] to complex three-dimensional fluid–structure interaction models[38,39] are available), the mechanisms driving the long-term evolution of the arterial network are not fully elucidated. Indeed, in spite of some important steps towards the understanding of the determinants of vascular remodeling,[40,41] a comprehensive mathematical model capable of predicting the development and evolution of this phenomenon is still missing.

For the sake of simplicity, we consider a simple model to describe the evolution in the fast-scale dynamics, namely the two-stage Windkessel model (see Figure 1), which reads[37]

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}P_{\mathrm{p}}(t) = \dfrac{P_{\mathrm{d}}(t)-P_{\mathrm{p}}(t)}{R_{\mathrm{p}}C_{\mathrm{p}}} + \dfrac{Q(t)}{C_{\mathrm{p}}} & t\in(0,T], \\ \dfrac{\mathrm{d}}{\mathrm{d}t}P_{\mathrm{d}}(t) = \dfrac{P_{\mathrm{p}}(t)-P_{\mathrm{d}}(t)}{R_{\mathrm{p}}C_{\mathrm{d}}} + \dfrac{P_{\mathrm{vs}}-P_{\mathrm{d}}(t)}{R_{\mathrm{d}}C_{\mathrm{d}}} & t\in(0,T], \\ P_{\mathrm{p}}(0) = P_{\mathrm{p},0}, \\ P_{\mathrm{d}}(0) = P_{\mathrm{d},0}, \end{cases} \tag{1}$$

where we denote by $x(t) = (P_{\mathrm{p}}(t), P_{\mathrm{d}}(t))$ the state of the model, given by the proximal and distal pressures; we denote the parameters of the fast-scale model by $\theta = (R_{\mathrm{p}}, C_{\mathrm{p}}, R_{\mathrm{d}}, C_{\mathrm{d}})$, respectively, the proximal and distal resistance and compliance; $Q(t)$ is the blood flux. We suppose then to be able to measure the proximal pressure at a collection of time instants $(t_1, ..., t_K)$ and we collect all the measurements into the observation vector $z = (P_{\mathrm{p}}(t_1), ..., P_{\mathrm{p}}(t_K))$.

When considering the fast time scale (i.e., the typical time scale of a heartbeat), the value of θ, describing the properties of the arterial network, is fixed. However, if we consider a longer time scale (days and above), θ may evolve, possibly associated with some disease. For instance, hypertension is linked to an increase of the arterial resistance R_d, which requires a higher systolic pressure to preserve the cardiac output. We thus introduce a slow time variable τ and we write $\theta(\tau)$, assuming that the fast-scale parameters are in fact functions of the slow-scale time variable.

We consider the following scenario. Suppose that we have a set of N_P patients, periodically monitored over the long-term horizon at times $(\tau_1 < \tau_2 < ... < \tau_{N_S})$. At each time τ_j (for $j = 1, ..., N_S$), the arterial pressure of the i-th patient (for $i = 1, ..., N_P$) is measured for a few seconds, collecting the observation vector z_j^i. More precisely, we denote the measured observation vector by $\tilde{z}_j^i = z_j^i + \chi_j^i$, where χ_j^i is the measurement error. The measurement $\tilde{z}_j^i$ reflects the value of the fast-scale parameters of the i-th patient at the slow-scale time τ_j, which we define as $\theta_j^i := \theta^i(\tau_j)$, where $\theta^i(\tau)$ denotes the slowly evolving i-th patient's parameters of the fast-scale model.

The goal is to learn a model for the slow-scale evolution of $\theta(\tau)$ from the collection of noisy measurements $\left\{\tilde{z}_j^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$. As mentioned before, all patients are different, and thus inter-individual variability must be taken into account.

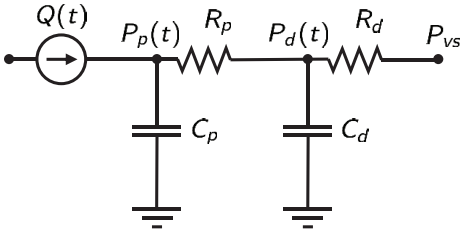


FIGURE 1 Visual representation of the two-stage Windkessel model of Equation (1). In this electrical analogy, current represents blood flux, while voltage represents blood pressure. The heart is represented as a current generator, while the arterial network comprises dissipative and compliant elements, represented by resistances and capacitors, respectively

1.4 | Paper outline

The paper is organized as follows. In Section 2, we state in mathematical terms the problem that we deal with in this paper and we introduce the associated notation. Then, we present the methods that we propose to solve such a problem, based on a decoupling of the time scales. In Section 3, we show the effectiveness and the noise-robustness of the proposed methods by means of several numerical test cases. Finally, in Section 4, we critically discuss the results obtained and we draw our conclusions.

2 | PROBLEM STATEMENT AND PROPOSED METHODS

In this section, motivated by the example of Section 1.3, we state the setting of the problem that we consider in this paper, in which the example of Section 1.3 can be recast, before introducing and analyzing our proposed methods.

We consider a collection of N_P individuals, such that the i-th individual is associated with $\theta^i(\tau) \in \mathcal{P} \subset \mathbb{R}^{N_\theta}$, which evolves with a slow dynamics driven by an equation of the form

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta^i(\tau) = g\left(\theta^i(\tau), \alpha^i\right) & \tau \in (0, \mathcal{T}], \\ \theta^i(0) = \theta_0^i, \end{cases} \tag{2}$$

where $\alpha^i \in \mathcal{A} \subset \mathbb{R}^{N_\alpha}$ are parameters characterizing the i-th individual at the slow-scale, accounting for inter-individual variability. In this paper, we always assume that

$$g \in \mathcal{G} := \left\{ g \in \mathcal{C}^0\left(\mathbb{R}^{N_\theta \times N_\alpha}; \mathbb{R}^{N_\theta}\right), \text{Lipschitz continuous in } \theta \right\},$$

so that the models admit a unique solution for each initial state and parameter.

This slow-scale model is coupled with a fast-scale dynamics, for which $\theta^i(\tau)$ can be considered as a constant. Specifically, we consider a collection of times $(\tau_1 < \tau_2 < \ldots < \tau_{N_S})$ over the long-term horizon, and we write $\theta_j^i := \theta^i(\tau_j)$, for $j = 1, \ldots, N_S$ and $i = 1, \ldots, N_P$. We consider a model for the fast-scale dynamics with the following form

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}x_j^i(t) = f\left(x_j^i(t), \theta_j^i, t\right) & t \in (0, T), \\ x_j^i(0) = x_{j,0}^i, \end{cases} \tag{3}$$

where $x_j^i(t) \in \mathcal{X} \subset \mathbb{R}^{N_x}$ is the fast-scale state. We recall that θ_j^i are considered as constant parameters at this scale (we assume $T \ll \tau_j - \tau_{j-1}$). On this system, we consider an observation process given at the fast scale, namely

$$z_j^i = \int_0^T h\left(x_j^i, t\right) \mathrm{d}t, \tag{4}$$

where $z_j^i \in \mathcal{Z} \subset \mathbb{R}^{N_z}$ and we define the measured observations as $\tilde{z}_j^i = z_j^i + \chi_j^i$, where $\chi_j^i \sim \mathcal{N}(0, W)$ are the measurement errors that we assume to be i.i.d. and Gaussian with covariance matrix W. In the example of Section 1.3, the observation vector contains direct measurements of the first entry of the state x at a collection of discrete times t_k, for $k = 1, \ldots, K$. This can be written in the form of Equation (4) by defining the observation map as $h(x,t) := \sum_{k=1}^K x \cdot e_1 \delta(t - t_k) e_k$, where e_k denotes the k-th element of the canonical basis of $\mathbb{R}^n$ and where δ denotes the Dirac delta function (in fact, given a function $f(t)$, we have $\int_0^T f(t)\delta(t - t_k)\,\mathrm{d}t = f(t_k)$).

We assume that we have perfect knowledge about the equation driving the fast-scale evolution and that we want to identify a model to describe the slow-scale evolution. Specifically, we assume that we know:

(K1): f, h: the fast-scale evolution and observation laws.

(K2): $\left\{\tilde{z}_j^i\right\}_{j=1,\ldots,N_S}^{i=1,\ldots,N_P}$: the fast-scale measurements.

Whereas we do not know:

(U1): g: the slow-scale evolution law.
(U2): $\{\alpha^i\}^{i=1,...,N_P}$: the slow-scale parameters of individuals.
(U3): $\{\theta_0^i\}^{i=1,...,N_P}$: the slow-scale initial states of individuals, for which we assume a prior distribution $\mathcal{N}(\bar{\theta},\Pi_\theta)$.
(U4): $\left\{x_{j,0}^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$: the fast-scale initial states of individuals, for which we assume a prior distribution $\mathcal{N}(\bar{x},\Pi_x)$.

The goal is to identify the unknown objects (U1)–(U4) from the known ones (K1)–(K2). We notice that this task is situated in an intermediate position between the fields of DA, as we here seek to identify the parameters and the state for a known dynamics, and ML, as we seek to discover the law driving the slow-scale (unknown) dynamics. We remark that the setting considered in this paper differs from that of a standard parameters identification problem, in which one aims to identify unknown parameters of a known model from time series data (see, e.g., Yu et al.[3,4,42]), for two main reasons. First, the observations $\tilde{z}_j^i$ are taken at the fast-scale, while we are interested in inferring knowledge at the slow-scale. Secondly, we assume that we are agnostic of the model whose parameters we aim to identify. As a matter of fact, we aim at simultaneously learning the parameters α^i and the differential Equation (2).

2.1 | General strategy

In this paper we propose to combine ML and DA concepts in order to address the problem presented above. Specifically, concerning the task of learning the law g given the measurements $\left\{\tilde{z}_j^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$, we extend the strategy proposed in Regazzoni et al.[26] Hence, we select a set of candidate laws $\mathcal{G}_n$ (where n indicates that the set is parameterized by a finite number of parameters), such that $\mathcal{G}_n \subset \mathcal{G}$, and we look for the model, inside the family $\mathcal{G}_n$, that best fits the available data. This can be interpreted, by assuming Gaussian distribution of the measurement errors, as the maximum-likelihood estimation of the evolution law g inside the family $\mathcal{G}_n$. This formalism allows to write the problem of identifying/learning the objects (U1)–(U4) in a unified minimization framework, as we show in the next section.

2.1.1 | A unified minimization framework

The maximum likelihood estimation of the unknown objects is found by minimizing the negative log-likelihood

$$\left(\hat{g},\left\{\hat{\alpha}^i\right\}^i,\left\{\hat{\theta}_0^i\right\}^i,\left\{\hat{x}_{j,0}^i\right\}_j^i\right) = \underset{\substack{g\in\mathcal{G}_n\\ \{\alpha^i\}^i\in\mathcal{A}^{N_P}\\ \{\theta_0^i\}^i\in\mathcal{P}^{N_P}\\ \{x_{j,0}^i\}_j^i\in\mathcal{X}^{N_P N_S}}}{\operatorname{argmin}} = \sum_{i=1}^{N_P}\left(\sum_{j=1}^{N_S}\left(\frac{1}{2}\left|z_j^i-\tilde{z}_j^i\right|_{W^{-1}}^2+\frac{1}{2}\left|x_{j,0}^i-\bar{x}\right|_{\Pi_x^{-1}}^2\right)+\frac{1}{2}\left|\theta_0^i-\bar{\theta}\right|_{\Pi_\theta^{-1}}^2\right), \tag{5}$$

such that (2), (3) and (4) hold true. Here and in what follows, given a vector $w\in\mathbb{R}^n$ and a symmetric positive-definite matrix $Q\in\mathbb{R}^{n\times n}$, we denote by $|w|_Q := (w^TQw)^{1/2}$ the energy norm. Moreover, for the sake of brevity, we denote by $\{\cdot\}_j^i$ and $\{\cdot\}^i$ sets indexed by $j = 1, ..., N_S$ and $i = 1, ..., N_P$, where the values taken by the indexes j and i are left implicit.

Problem (5) can be interpreted as that of finding the slow-scale law, the slow-scale parameter for each individual, and the initial state at both scales such that the resulting outputs z_j^i best approximate the measured outputs $\tilde{z}_j^i$. The last two terms of the loss functional can be regarded as regularization terms.

2.1.2 | Decoupling the two scales

Despite its solid foundation, the solution of Problem (5) may be unaffordable in practice, because of the huge number of differential equations that serve as constraints for the minimization problem. Indeed, Problem (5) is constrained by the N_P slow-scale differential Equations (2) and the $N_P \times N_S$ differential Equations (3). Moreover, Problem (5) attempts to simultaneously identify objects related to both the slow scale and the fast scale for each $j = 1, ..., N_S$ and $i = 1, ..., N_P$.

To lower the complexity of Problem (5), we propose to exploit the scale separation between the fast and the slow dynamics, and to split the problem into two steps:

(P1): The first step consists in finding an estimate for θ_j^i, for each $j = 1, ..., N_S$ and $i = 1, ..., N_P$, on the basis of the measurements $\tilde{z}_j^i$, by solving the problem

$$\left(\hat{\theta}_j^i, \hat{x}_{j,0}^i\right) = \underset{\substack{\theta_j^i \in \mathcal{P} \\ x_{j,0}^i \in \mathcal{X}}}{\operatorname{argmin}} \left(\frac{1}{2} \left| z_j^i - \tilde{z}_j^i \right|_{W^{-1}}^2 + \frac{1}{2} \left| x_{j,0}^i - \bar{x} \right|_{\Pi_x^{-1}}^2 + \frac{1}{2} \left| \theta_j^i - \bar{\theta} \right|_{\Pi_\theta^{-1}}^2 \right), \tag{6}$$

such that (3) and (4) hold true.

(P2): Once the estimates $\left\{\hat{\theta}_j^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$ are available, we consider the problem

$$\left(\hat{g}, \left\{\hat{\alpha}^i\right\}^i, \left\{\hat{\theta}_0^i\right\}^i\right) = \underset{\substack{g \in \mathcal{G}_n \\ \left\{\alpha^i\right\}^i \in \mathcal{A}^{N_P} \\ \left\{\theta_0^i\right\}^i \in \mathcal{P}^{N_P}}}{\operatorname{argmin}} \sum_{i=1}^{N_P} \sum_{j=1}^{N_S} \frac{1}{2} \left| \theta^i(\tau_j) - \hat{\theta}_j^i \right|_{\Pi_\theta^{-1}}^2, \tag{7}$$

such that (2) holds true.

We notice that (P1) involves only the fast-scale dynamics, whereas (P2) only the slow-dynamics. Indeed, the two scales have been decoupled. As a consequence, in (P1), each individual and each slow-scale instant τ_j can be treated separately; the problem can be solved for each $j = 1, ..., N_S$ and $i = 1, ..., N_P$ independently of each other. This allows for a possibly parallel solution of (P1). Moreover, differently from Problem (5), in Problems (P1) and (P2) objects related to the fast and slow scales are never simultaneously identified. In other words, the complex Problem (5) has been split into $(N_P N_S + 1)$ simpler problems.

A further effect of splitting Problem (5) into two steps, is that Problems (P1) and (P2) feature different natures. While the former is a purely DA problem, as it consists in the identification of the state and the parameters for System (3), the latter is a learning problem, as the laws governing the slow-scale dynamics need to be discovered. This is a consequence of the decoupling between the fast-scale, driven by a known dynamics, from the slow-scale, whose dynamics is governed by unknown laws.

Several identifications procedures are available in the literature for the solution of problems written in the form of (P1) (see, e.g., Yu et al.[3,4,5,6]). In this paper, we restrict our scope to measurements sampled in time, instead of continuous signals, and we consider a state and parameter sequential estimation strategy[2,6,43] based on the extended Kalman filter (EKF), an extension of the Kalman filter (KF) algorithm to nonlinear systems, performed at each step, of Equation (3).[7,43] We recall that whereas KF is equivalent to least square estimation for linear dynamics, EKF only gives an approximation of the least square estimation (P1) with, however, the benefit of being "on the fly." This is particularly adapted to the fast-scale dynamics where we can typically face real-time monitoring constraints.

2.1.3 | Interpretation of the learned model

Before presenting an algorithm for the solution of Problem (P2) (which is treated in Section 2.2), we deal with the problem of interpreting the solution of Problem (P2) itself. This is instrumental to the presentation of the proposed methods. Therefore, in this section we assume the existence of an algorithm capable of solving Problem (P2) and we speculate about the interpretation of its solution. Solving Problem (P2) serves indeed different purposes, which we list in what follows.

Understanding

First, the function $\hat{g}$ provides insight into the phenomenon under exam. Indeed, $\hat{g}$ yields a description of the underlying laws written in mathematical terms and as such it can provide understanding of the phenomenology. Moreover, an analysis of the features of the function $\hat{g}$ may reveal properties such as states of equilibrium, symmetry properties and

relationships among the variables that may not be immediately identifiable from the observation of the experimental data.

Classification of training individuals

By solving Problem (P2), a (possibly vector-valued) parameter $\hat{\alpha}^i$ is assigned to each individual. However, the physical meaning of such parameters is not clear, as they are learned in a black-box manner. Moreover, the value of such parameters in the solution of Problem (P2) is not unique: given a solution of Problem (P2), one can always find an equivalent solution with different values of the parameters (we will deal extensively with this issue in Section 2.3).

Nonetheless, even if we cannot interpret such parameters in physical terms, we know that they characterize the variability among the individuals. Indeed, one may be tempted to conclude that similar parameters reveal similarities among individuals (that is to say, if $\hat{\alpha}^i \simeq \hat{\alpha}^k$, then the i-th and the k-th individuals are similar). In Section 2.3 we will show under which conditions this conclusion is licit. In that case, the learned parameters $\{\hat{\alpha}^i\}^i$ allow to *classify* the individuals.

For instance, in the example of Section 1.3, the parameter α may describe how the arterial network of a given patient remodels in time, and thus it can be regarded as an indication of the severity of related pathologies.

Predictions

The ultimate aim of mathematical models is making predictions. However, the lack of physical meaning of the parameter α apparently undermines this possibility for the learned model $\hat{g}$. In fact, let us consider a new individual (i.e. not employed to train the model), which we denote as the $(N_P+1)-$th individual; to solve Problem (P2) with the learned model $\hat{g}$, in order to predict the evolution of $\theta^{N_P+1}(\tau)$, one needs to know α^{N_P+1}. However, unlike the parameters of physics-based models that, having a physical meaning, can be measured in practice, the parameters of data-driven models cannot be measured.

This inconvenience can be overcome by combining once again ML with DA (the latter acting this time at the slow-scale). All we need is to observe the $(N_P+1)-$th individual for a (short) interval of time, on the slow-scale, $\tau \in [0, \mathcal{T}^{\text{obs}}]$, with $\mathcal{T}^{\text{obs}} < \mathcal{T}$. During this observation time interval, we collect the observations $\tilde{z}_j^{N_P+1}$, associated with the slow-scale time instants $0 = \tau_1 < \tau_2 < \ldots \tau_{N_S^{\text{obs}}} = \mathcal{T}^{\text{obs}}$, and we find the estimates, by solving (P1), for the associated fast-scale parameters $\hat{\theta}_j^{N_P+1}$. Then, by the slow-scale evolution of such parameters, we estimate the value of $\hat{\alpha}^{N_P+1}$, by solving the following DA problem

$$\left(\hat{\alpha}^{N_P+1}, \hat{\theta}_0^{N_P+1}\right) = \underset{\substack{\alpha^{N_P+1}\in\mathcal{A} \\ \theta_0^{N_P+1}\in\mathcal{P}}}{\operatorname{argmin}} \left(\sum_{j=1}^{N_S^{\text{obs}}} \frac{1}{2} \left| \theta^{N_P+1}(\tau_j) - \hat{\theta}_j^{N_P+1} \right|_{U^{-1}}^2 + \frac{1}{2} \left| \theta_0^{N_P+1} - \bar{\theta} \right|_{\Pi_\theta^{-1}}^2 + \frac{1}{2} \left| \alpha^{N_P+1} - \bar{\alpha} \right|_{\Pi_\alpha^{-1}}^2 \right), \tag{8}$$

subject to the constraint

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta^{N_P+1}(\tau) = \hat{g}\left(\theta^{N_P+1}(\tau), \alpha^{N_P+1}\right), & \tau \in \left(0, \mathcal{T}^{\text{obs}}\right] \\ \theta^{N_P+1}(0) = \theta_0^{N_P+1}. \end{cases} \tag{9}$$

In Equation (8), U denotes the covariance matrix associated with the error introduced during the resolution of the DA Problem (P1). We remark that U can be estimated whenever Problem (P1) is solved, for example, by an algorithm of the Kalman Filter family. Alternatively, assuming that the error introduced by the DA algorithm can be considered as a white noise of magnitude $\sigma \frac{dB(\tau)}{d\tau}$ ($B(\tau)$ being a Wiener process), we set $U = \sigma^2/\Delta\tau\, I$, where I is the $N_\theta \times N_\theta$ identity matrix. On the other hand, $\bar{\alpha}$ and Π_α respectively denote the expected value and the covariance matrix associated with the slow-scale parameter α, estimated from the parameters $\{\alpha^i\}^{i=1,\ldots,N_P}$ learned during the step (P2). Specifically, we set $\bar{\alpha}$ equal to the arithmetic mean of these parameters and Π_α equal to their sample covariance.

We can finally employ the estimated slow-scale parameter $\hat{\alpha}^{N_P+1}$ to predict the evolution of $\theta^{N_P+1}(\tau)$ in the time interval $\left(\mathcal{T}^{\text{obs}}, \mathcal{T}\right]$, by solving

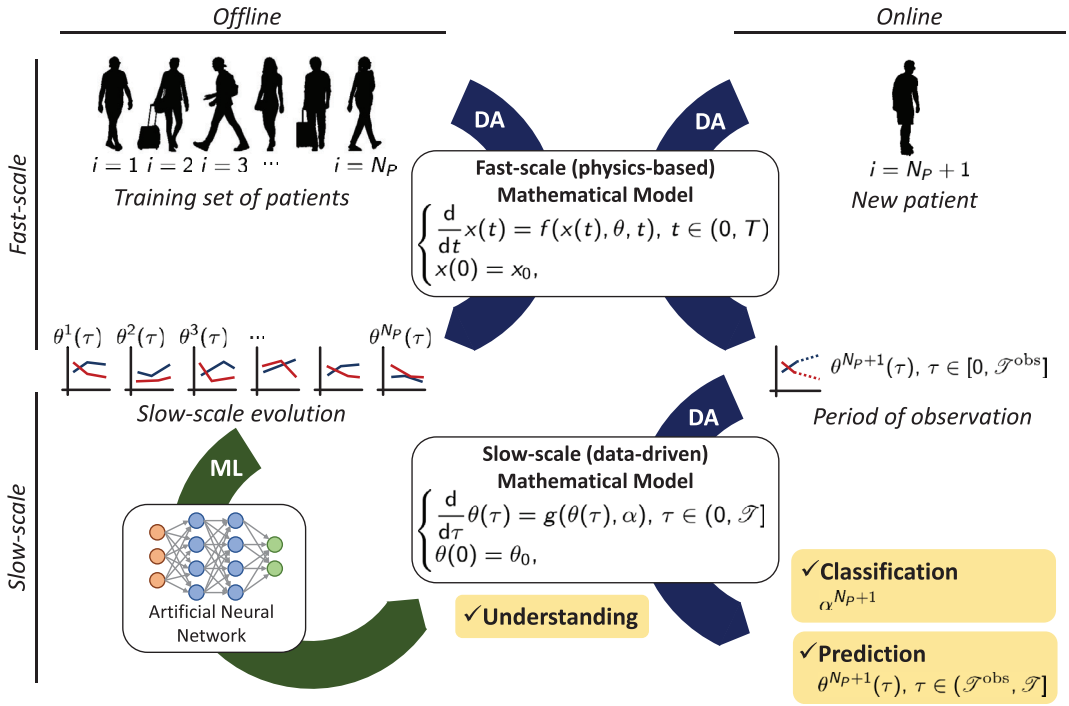


FIGURE 2 Interplay between ML (green arrow) and DA (blue arrows)

$$\begin{cases} \frac{d}{d\tau}\hat{\theta}^{N_P+1}(\tau) = \hat{g}\left(\theta^{N_P+1}(\tau), \hat{\alpha}^{N_P+1}\right), & \tau \in \left(\mathcal{T}^{obs}, \mathcal{T}\right] \\ \hat{\theta}^{N_P+1}\left(\mathcal{T}^{obs}\right) = \hat{\theta}^{N_P+1}_{N_S^{obs}}. \end{cases} \tag{10}$$

To summarize, in a preliminary offline phase, we learn Model (2) from the observation of a set of training individuals, by combining DA at the fast-scale with ML at the slow-scale. Then, in the online phase, we employ DA on both the fast-scale and, for a short interval, on the slow-scale $\left[0, \mathcal{T}^{obs}\right]$ to characterize the features of a new individual, so that the previously learned model can be used to predict the evolution of this individual. The interplay that takes place, at different levels, between DA and ML is summarized in Figure 2.

2.2 | Learning a differential equation from data

In this section we present an algorithm to numerically solve Problem (P2). The algorithm presented in this paper represents an extension of the algorithm presented in Regazzoni et al.,[26] whose goal is learning a (possibly parametric) time-dependent differential equation from a collection of input–output time-series. However, in Regazzoni et al.,[26] the parameters associated with the training data are assumed to be known, while in this paper we need to learn them simultaneously to learning the model. The backbone of the algorithm proposed in this paper is based on that presented in Regazzoni et al.[26] Therefore we recall here the main ideas and we highlight the differences, while we refer to Regazzoni et al.[26] for the details common to both algorithms. The algorithm is based on a ANN-based representation of the function g, which is trained by computing sensitivities through the adjoint method.[44] A similar approach is presented in Chen et al.[28] under the name of Neural ODEs, in which the state equation is discretized by means of a Runge–Kutta method.[45] Our algorithm represents a generalization of this method to some extent, as it learns a parametric differential equation with unknown parameters, rather than assuming that all the samples share the same dynamics.

2.2.1 | Solution strategy

Following,[26] we parameterize g by a finite number of real parameters $\mu \in \mathbb{R}^n$ and we define the set of candidate laws as $\mathcal{G}_n = \{(\theta,\alpha) \mapsto g(\theta,\alpha;\mu) : \mu \in \mathbb{R}^n\}$. For the moment, we do not detail how this parameterization is performed, in order to not restrict ourselves to a specific case. To fix ideas, the reader can think of g as a polynomial in θ and α, where μ are the coefficients of the polynomial.

Problem (P2) can thus be rewritten as the following discrete constrained optimization problem

$$\begin{cases} \min\limits_{\mu\in\mathbb{R}^n} & \sum\limits_{i=1}^{N_P}\sum\limits_{j=1}^{N_S}\frac{1}{2}\left|\theta^i(\tau_j)-\hat{\theta}_j^i\right|^2_{\Pi_\theta^{-1}} \\ & \{\alpha^i\}^i \in \mathcal{A}^{N_P} \\ & \{\theta_0^i\}^i \in \mathcal{P}^{N_P} \\ \text{s.t.} & \frac{\mathrm{d}}{\mathrm{d}\tau}\theta^i(\tau) = g\left(\theta^i(\tau),\alpha^i;\mu\right), \quad \tau\in(0,\mathcal{T}], \quad i=1,...,N_P \\ & \theta^i(0)=\theta_0^i, \quad i=1,...,N_P, \end{cases} \tag{11}$$

where $\left\{\hat{\theta}_j^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$ are given. To derive the gradient of the cost functional $\mathcal{J} = \sum_{i=1}^{N_P}\sum_{j=1}^{N_S}\frac{1}{2}\left|\theta^i(\tau_j)-\hat{\theta}_j^i\right|^2_{\Pi_\theta^{-1}}$ under the constraint given by Equation (2), we introduce a family of Lagrange multipliers $\sigma^i \in \mathcal{C}^0([0,\mathcal{T}];\mathcal{P})$ and we write the Lagrangian associated with Problem (11).

$$\begin{aligned} &\mathscr{L}\left(\mu,\{\alpha^i\}^i,\{\theta_0^i\}^i,\{\theta^i\}^i,\{\sigma^i\}^i\right) \\ &= \sum_{i=1}^{N_P}\sum_{j=1}^{N_S}\frac{1}{2}\left|\theta^i(\tau_j)-\hat{\theta}_j^i\right|^2_{\Pi_\theta^{-1}} - \sum_{i=1}^{N_P}\int_0^{\mathcal{T}}\left(\frac{\mathrm{d}}{\mathrm{d}\tau}\theta^i(\tau)-g(\theta^i(\tau),\alpha^i;\mu)\right)\cdot\sigma^i(\tau)d\tau - \sum_{i=1}^{N_P}\left(\theta^i(0)-\theta_0^i\right)\cdot\sigma^i(0) \\ &= \sum_{i=1}^{N_P}\sum_{j=1}^{N_S}\frac{1}{2}\left|\theta^i(\tau_j)-\hat{\theta}_j^i\right|^2_{\Pi_\theta^{-1}} + \sum_{i=1}^{N_P}\int_0^{\mathcal{T}}\theta^i(\tau)\cdot\frac{\mathrm{d}}{\mathrm{d}\tau}\sigma^i(\tau)d\tau + \sum_{i=1}^{N_P}\int_0^{\mathcal{T}}g\left(\theta^i(\tau),\alpha^i;\mu\right)\cdot\sigma^i(\tau)d\tau - \sum_{i=1}^{N_P}\theta^i(\mathcal{T})\cdot\sigma^i(\mathcal{T}) + \sum_{i=1}^{N_P}\theta_0^i\cdot\sigma^i(0), \end{aligned}$$

where the last inequality is obtained by integrating by parts the time integral. By setting to zero the variation of the Lagrangian with respect to the variables $\{\theta^i\}^i$ we get the adjoint equations, valid for $i = 1, ..., N_P$

$$\begin{cases} -\frac{\mathrm{d}}{\mathrm{d}\tau}\sigma^i(\tau) = \nabla_\theta^T g\left(\theta^i(\tau),\alpha^i;\mu\right) + \sum\limits_{j=1}^{N_S}\Pi_\theta^{-1}\left(\hat{\theta}_j^i-\theta^i(\tau)\right)\delta_{\tau_j}(\tau), \quad \tau\in[0,\mathcal{T}) \\ \sigma^i(\mathcal{T}) = 0, \end{cases} \tag{12}$$

Finally, by computing the variation of the Lagrangian with respect to the design variables we get the gradient of the cost functional with respect to the design variables themselves

$$\nabla_\mu\mathcal{J} = \sum_{i=1}^{N_P}\int_0^{\mathcal{T}}\nabla_\mu^T g\left(\theta^i(\tau),\alpha^i;\mu\right)\sigma^i(\tau)d\tau; \quad \nabla_{\alpha^i}\mathcal{J} = \int_0^{\mathcal{T}}\nabla_\alpha^T g\left(\theta^i(\tau),\alpha^i;\mu\right)\sigma^i(\tau)d\tau; \quad \nabla_{\theta_0^i}\mathcal{J} = \sigma^i(0). \tag{13}$$

Given the gradient of the cost functional, any gradient-based optimization (or training) algorithm can be used to find an approximate solution of Problem (11). To produce the results shown in this paper, we employed the Levenberg–Marquardt algorithm (see, e.g., Nocedal and Wright[46]), which is specifically designed for least-squares problems, coupled with a line-search for the step length, as illustrated in Regazzoni et al.[26]

Following,[26] we discretize the state equations in (11) by means of the Forward Euler scheme. The choice of an explicit scheme is aimed at lowering the computational cost of the training phase. To find the discrete version of the adjoint Equations (12), we write the Lagrangian associated with the fully discretized version of Problem (11) and we proceed as above (see Regazzoni et al.[26] for further details).

2.2.2 | Choice of the space of candidate models

The only missing ingredient to define the algorithm for the numerical solution of Problem (P2) is the definition of the set of candidate laws $\mathcal{G}_n$. This choice is driven by the trade-off between two different desired features: on the one hand, the space $\mathcal{G}_n$ should be rich enough to contain a law that can accurately explain the training data; on the other hand, a too rich space would lead to overfitting, that is to say the learned law would fit very accurately the training data, while featuring bad generalization properties in new cases (see Regazzoni et al.[26] for a detailed discussion on this topic).

A class of function approximators that accomplishes a good trade-off between the accuracy in fitting data and the generalization capability is that of ANNs.[47] As a matter of fact, ANNs are universal approximators in several function spaces, including that of continuous functions (see, e.g., the density results contained in Cybenko[48,49,50]). Moreover, ANNs provide an effective way of tuning the richness of the space $\mathcal{G}_n$ by suitably selecting the number of layers and of neurons.

Hence, we define $\mathcal{G}_n = \{(\theta,\alpha)\mapsto g(\theta,\alpha;\mu) : \mu\in\mathbb{R}^n\}$ where $g(\theta,\alpha;\mu)$ denotes an ANN where the input is given by (θ,α) and $\mu\in\mathbb{R}^n$ represents the vector of ANN parameters, that is, weights and biases.[47] The gradients $\nabla_\theta g(\theta,\alpha;\mu)$, $\nabla_\alpha g(\theta,\alpha;\mu)$ and $\nabla_\mu g(\theta,\alpha;\mu)$, needed in Equations (12) and (13), are computed through the backward-propagation formulas, as given in Yegnanarayana.[47]

2.3 | Learning an interpretable slow-scale dynamics

In Section 2.2 we have presented an algorithm for the solution of Problem (P2). In this section, we discuss the interpretation of the solution of such a problem. We recall that, as mentioned Section 2.1.3, the solution of Problem (P2) serves three different purposes, namely understanding the phenomenon though its mathematical description, classifying individuals and predicting the evolution of new (not yet observed) individuals. The discussion is conducted through two test cases. In all the tests presented in this paper, fully connected ANNs with hyperbolic tangent activation functions are employed.

2.3.1 | Test Case 1: Non-uniqueness of representation of models

In order to assess the possibility of accomplishing the above mentioned purposes, we consider the following idealized situation. We consider only the slow-scale equation, given by

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta(\tau) = g(\theta(\tau),\alpha) \quad \tau\in(0,\mathcal{T}], \\ \theta(0) = \theta_0. \end{cases} \tag{14}$$

Hence, in this section we will simply denote θ as the state and α as the parameter, without specifying the scale on which the dynamics occur. Moreover, we assume that we are in a maximal information situation: namely, we assume that we have a population of individuals (that we denote by Ω) such that all the possible values of initial condition and parameter are covered (i.e., $\{(\theta_0(\omega),\alpha(\omega))\}_{\omega\in\Omega} = \mathcal{P}\times\mathcal{A}$) and that we observe the evolution of $\theta(\tau,\omega)$ for each $\omega\in\Omega$ and $\tau\in[0,\mathcal{T}]$, without noise (we denote the observed data as $\theta^{\mathrm{obs}}(\tau,\omega)$). Finally, we assume that we are able to solve the following problem

$$\left(\hat{g},\{\hat{\alpha}(\omega)\}_{\omega\in\Omega},\{\hat{\theta}_0(\omega)\}_{\omega\in\Omega}\right) = \underset{\substack{g\in\mathcal{G}\\ \{\alpha(\omega)\}_\omega\in\mathcal{A}^\Omega\\ \{\theta_0(\omega)\}_\omega\in\mathcal{P}^\Omega}}{\operatorname{argmin}} \int_0^{\mathcal{T}}\int_{\omega\in\Omega} \frac{1}{2}\left|\theta(\tau,\omega)-\theta^{\mathrm{obs}}(\tau,\omega)\right|^2_{\Pi_\theta^{-1}} d\omega\, d\tau, \tag{15}$$

subject to

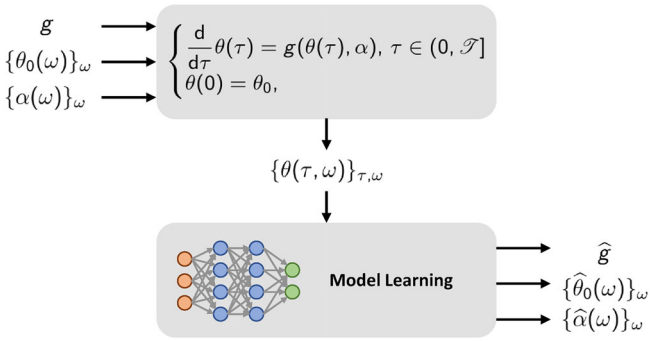


FIGURE 3 Visual representation of the idealized situation considered in Section 2.3

$$\begin{cases} \dfrac{d}{d\tau}\theta(\tau,\omega) = g(\theta(\tau,\omega),\alpha(\omega)) \quad \tau\in(0,\mathcal{T}],\omega\in\Omega \\ \theta(0,\omega) = \theta_0(\omega). \end{cases} \tag{16}$$

More precisely, we assume that we are able to find a triplet $\left(\hat{g},\{\hat{\alpha}(\omega)\}_{\omega\in\Omega},\left\{\hat{\theta}_0(\omega)\right\}_{\omega\in\Omega}\right)$ that is a global minimum of the loss functional of Problem (15). We notice that, indeed, this problem admits global minimizers, as the loss functional is bounded from below by the value zero, which is attained by the triplet $(g,\{\alpha(\omega)\}_{\omega\in\Omega},\{\theta_0(\omega)\}_{\omega\in\Omega})$ itself (namely, the *exact* solution).

The idealized situation above described is visually represented in Figure 3. The triplet $(g,\{\alpha(\omega)\}_{\omega\in\Omega},\{\theta_0(\omega)\}_{\omega\in\Omega})$ generates the collection of data $\left\{\theta^{\text{obs}}(\tau,\omega)\right\}_{\tau\in[0,\mathcal{T}],\omega\in\Omega}$, from which the triplet $\left(\hat{g},\{\hat{\alpha}(\omega)\}_{\omega\in\Omega},\left\{\hat{\theta}_0(\omega)\right\}_{\omega\in\Omega}\right)$ is inferred. Hence, one may tempted to conclude that we have $g\equiv\hat{g}$, $\alpha(\omega)\equiv\hat{\alpha}(\omega)$, $\theta_0(\omega)\equiv\hat{\theta}_0(\omega)$ for any $\omega\in\Omega$. However, while the latter conclusion is clearly true (since $\theta_0(\omega)=\theta(0,\omega)$ is part of the training dataset), the first two conclusions may be wrong. Consider, for instance, the following toy model

$$\begin{cases} \dfrac{d}{d\tau}\theta(\tau) = \alpha\theta(\tau) \quad \tau\in(0,\mathcal{T}], \\ \theta(0) = \theta_0, \end{cases} \tag{17}$$

where we have $g(\theta,\alpha)=\alpha\theta$ and we set $\mathcal{P}=\mathcal{A}=\mathbb{R}$. The data generated by Model (17) are given by $\theta^{\text{obs}}(\tau,\omega)=\theta_0(\omega)e^{\alpha(\omega)\tau}$ for any ω and τ. Consider now the following model

$$\begin{cases} \dfrac{d}{d\tau}\theta(\tau) = \dfrac{1}{2}\tilde{\alpha}\theta(\tau) \quad \tau\in(0,\mathcal{T}], \\ \theta(0) = \tilde{\theta}_0, \end{cases} \tag{18}$$

where we have $\tilde{g}(\theta,\tilde{\alpha})=\frac{1}{2}\tilde{\alpha}\theta$ and $\tilde{\alpha}\in\tilde{\mathcal{A}}=\mathbb{R}$. If we set $\tilde{\alpha}(\omega)=2\alpha(\omega)$ and $\tilde{\theta}_0(\omega)=\theta_0(\omega)$ for any $\omega\in\Omega$, then the data generated by Model (18) are given by $\theta^{\text{obs}}(\tau,\omega)=\tilde{\theta}_0(\omega)e^{\frac{1}{2}\tilde{\alpha}(\omega)\tau}=\theta_0(\omega)e^{\alpha(\omega)\tau}$ for any ω and τ. Therefore, the triplet $\left(\tilde{g},\{\tilde{\alpha}(\omega)\}_{\omega\in\Omega},\left\{\tilde{\theta}_0(\omega)\right\}_{\omega\in\Omega}\right)$ and the triplet $(g,\{\alpha(\omega)\}_{\omega\in\Omega},\{\theta_0(\omega)\}_{\omega\in\Omega})$ generate the same data.

More generally, given any invertible and sufficiently regular function $\psi:\mathcal{A}\to\tilde{\mathcal{A}}$, and by setting $\tilde{g}(\theta,\tilde{\alpha})=g(\theta,\psi^{-1}(\tilde{\alpha}))$, $\tilde{\alpha}(\omega)=\psi(\alpha(\omega))$ and $\tilde{\theta}_0(\omega)=\theta_0(\omega)$ for any $\omega\in\Omega$, the two triplets generate the same data. Hence, the two models g and $\tilde{g}$ are indistinguishable solely on the basis on their output. This entails that Problem (15) (and, thus, Problem (P2)) is intrinsically ill-posed, as it features many solutions. In other terms, even in the idealized case considered in this section, we do not have any guarantee that the learned model $\tilde{g}$ and the learned parameters $\tilde{\alpha}(\omega)$ coincide with the original ones (g and $\alpha(\omega)$). We remark that this is a more subtle issue than the case when different combinations of parameters lead to the same dynamics, which makes the parameters identification problem ill-posed. In the case considered here, indeed, different combinations of *models and parameters* lead to the same dynamics. This makes the problem of learning the model itself ill-posed.

Nonetheless, we remark that, in the previous example, Model (17) and Model (18) are both valid mathematical descriptions of the phenomenon and there is no apparent reason why the former should be preferable to the latter. As a matter of fact, due to the black-box nature of Problem (15), the solution is totally transparent to the specific

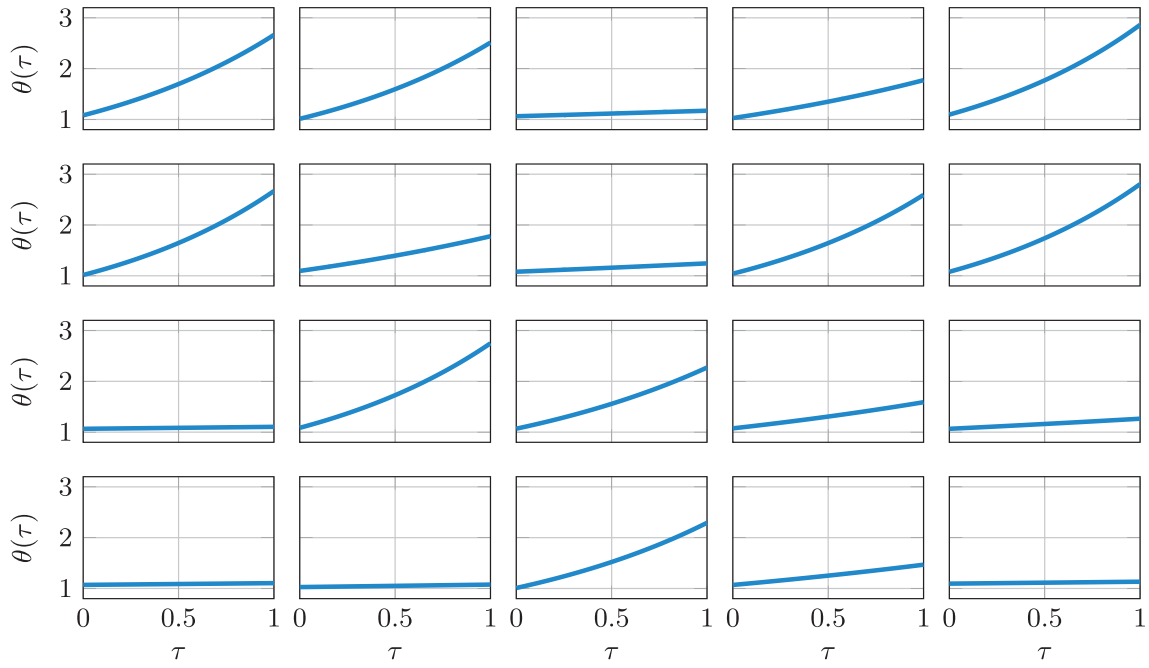


FIGURE 4 Test Case 1: training data. Each plot represents a training individual, characterized by a given pair (θ_0, α)

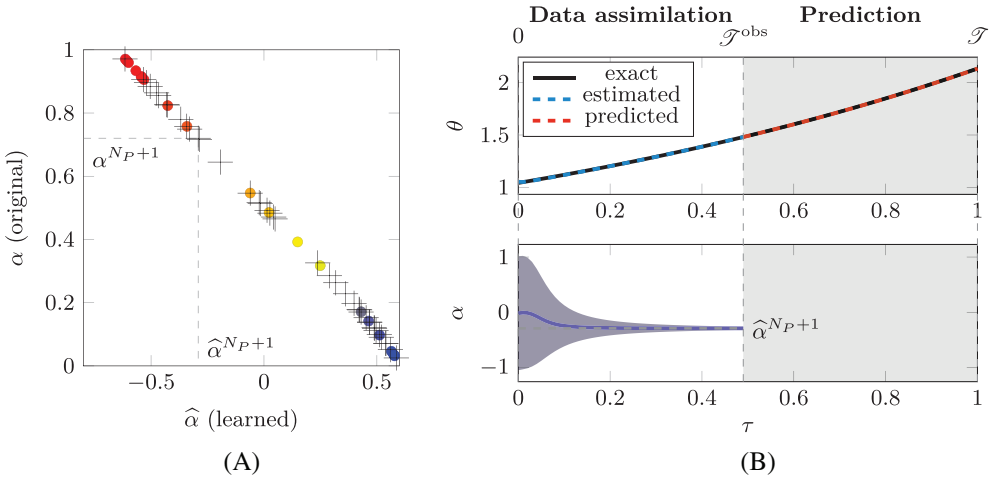


FIGURE 5 Test Case 1. In (A), the colored circles represent the original parameters α^i plotted against the corresponding learned parameters $\hat{\alpha}^i$ for the training individuals ($i = 1, ..., N_P$). Conversely, the black crosses represent the original parameters α^{N_P+k} plotted against the corresponding estimated parameters $\hat{\alpha}^{N_P+k}$ for the testing individuals. In the first row of (B) we show the evolution of $\theta^{N_P+1}(\tau)$ (blue solid line), while the black dashed line represent the value of $\hat{\theta}^{N_P+1}(\tau)$ estimated by means of the EKF algorithm (for $\tau \in [0, \mathcal{T}^{\text{obs}}]$) and its predicted value (for $\tau \in (\mathcal{T}^{\text{obs}}, \mathcal{T}]$). In the second row of (B), we show the evolution of the estimation of the value of $\hat{\alpha}^{N_P+1}$, with the $\pm 3\sigma$ bands, where σ denotes the standard deviation of the estimate

representation of the parameters α. Indeed, the reformulation obtained through the invertible map ψ is associated with a mere change of variables and the underlying model is essentially the same. Hence, we say that model $\tilde{g}$ is a *trivial reformulation* of model g, as it does not represent a substantially different model. In Section 2.3.3, we will give a more rigorous definition of this concept.

To illustrate the practical implications of the existence of trivial reformulations, we consider again the Problem (17), which we denote as Test Case 1. We consider $N_P = 20$ training individuals, for which we randomly generate an initial state $\theta_0^i \in (1, 1.1)$ and a parameter $\alpha^i \in (0, 1)$ by sampling from the two intervals with uniform probability, for $i = 1, ..., N_P$. Then, for each individual, we synthetically generate the evolution $\theta^i(\tau)$ for $\tau \in (0, \mathcal{T}]$ (with $\mathcal{T} = 1$), by numerically

approximating the solution of (17). The obtained transients are represented in Figure 4. Then, we subdivide the time interval $[0,\mathcal{T}]$ into equally distributed time instants $(\tau_1<\tau_2<...<\tau_{N_S})$ with a time step of $\Delta\tau=1e-2$, and we apply the algorithm of Section 2.2 in order to solve Problem (P2), where we set $\hat{\theta}_j^i=\theta^i(\tau_j)$ (i.e., we assume, for simplicity, that the slow-scale states can be estimated without error). Specifically, we train an ANN (endowed with a single hidden layer with three neurons) starting from the collection of observations $\left\{\hat{\theta}_j^i\right\}_{j=1,...,N_S}^{i=1,...,N_P}$, which, in this case, are not affected by noise. In this manner, we obtain a model $\hat{g}$ and a parameter $\hat{\alpha}^i$ for each of the training individuals. We remark that the proposed method is not restricted to single-layer ANNs, but more complex ANN architectures can be employed.

In Figure 5(A) we plot the value of the original parameter α^i against the value of the corresponding learned parameter $\hat{\alpha}^i$, for each of the $N_P=20$ training individuals (colored circles). We notice that, even if we do not have $\hat{\alpha}^i=\alpha^i$, a one-to-one relationship between the two classes of parameters is easily detectable. Hence, we can conclude that the ANN-based model, learned from the data, is based on a different (but, possibly, equivalent) parameterization of the space of parameters.

In order to test the capabilities of the learned model to be used for classification and prediction purposes, we apply the procedure introduced in Section 2.1.3. Specifically, we consider a new individual, for which we randomly generate an initial state $\theta_0^{N_P+1}$ and a parameter α^{N_P+1}. Similarly to what was done for the training individuals, we synthetically generate the evolution of the state $\theta^{N_P+1}(\tau)$, for $\tau\in(0,\mathcal{T}]$ by means of the original model (17). Then, we introduce an intermediate time instant $\mathcal{T}^{\text{obs}}=0.5$ s and we imagine to observe the evolution of the new individual in the interval $\left[0,\mathcal{T}^{\text{obs}}\right]$. In this time interval, we apply the EKF algorithm through the learned model $\hat{g}$: more precisely, we solve Problem (8), in order to estimate the value of the parameter $\hat{\alpha}^{N_P+1}$ and of the state $\theta^{N_P+1}(\tau)$ (see Figure 5(B)). We repeat the same protocol multiple times: we generate a random initial state $\theta_0^{N_P+k}$ and a random parameter α^{N_P+k}, for $k=1$, ..., we generate the corresponding synthetic data, from which, by means of DA, we estimate the values of $\hat{\alpha}^{N_P+k}$. In Figure 5(A), we plot the obtained pairs $\left(\hat{\alpha}^{N_P+k},\alpha^{N_P+k}\right)$. We notice that the estimated values of $\hat{\alpha}$ are compliant with the one-to-one relationship between the parameters α and the parameters $\hat{\alpha}$ that emerged in the training phase. Therefore, since in practical applications one cannot observe α^{N_P+k}, while $\hat{\alpha}^{N_P+k}$ can be estimated, this provides a way of classifying the new individual. Indeed, the mere observation $\hat{\alpha}^{N_P+k}$ allows to conclude that the (N_P+k)-th individual features characteristics similar to those of the individuals with a similar $\hat{\alpha}$.

As mentioned in Section 2.1.3, by exploiting the estimated value of $\hat{\alpha}^{N_P+1}$, the model $\hat{g}$ can be exploited to predict the evolution of $\theta^{N_P+1}(\tau)$ for $\tau\in\left(\mathcal{T}^{\text{obs}},\mathcal{T}\right]$. In Figure 5(B) we report the prediction of $\theta^{N_P+1}(\tau)$ obtained by numerically approximating the solution of Problem (10). We repeat the same protocol for 1000 synthetically generated testing individuals, obtaining an overall normalized L^2 error between the prediction and the exact solutions of $3.8e-3$.

In conclusion, even if the learned model does not coincide with the original one (it is indeed a trivial reformulation of it), thanks to the interplay between ML and DA, it can still be employed to classify individuals and to make predictions. In Section 2.3.4 we will deal with the problem of unequivocally selecting a unique representative model within a class of trivially reformulations of the same underlying model. However, before dealing with this topic, we show that something more subtle than a trivial reformulation may hinder the interpretability of the results.

2.3.2 | Test Case 2: Non-trivial reformulations

Let us consider the following model, which we denote as Test Case 2, where the state is given by the two-dimensional vector $\theta=(\theta_1,\theta_2)\in\mathbb{R}^2$, while the parameter is one-dimensional

$$\begin{cases}\dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_1(\tau)=\alpha\theta_1(\tau) & \tau\in(0,\mathcal{T}],\\ \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_2(\tau)=0 & \tau\in(0,\mathcal{T}],\\ \theta_1(0)=\theta_{1,0},\theta_2(0)=\theta_{2,0}.\end{cases}\tag{19}$$

Test Case 2 is obtained from Test Case 1 by introducing a further state variable, with a trivial dynamics (θ_2 is clearly constant in τ). However, even if the modification is only minimal, the results (shown in Figure 6(A)) are significantly

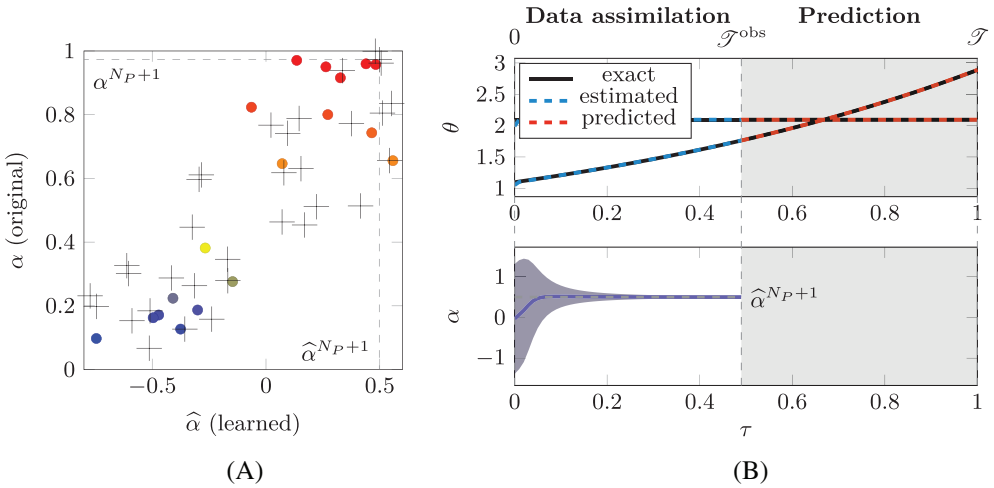


FIGURE 6 Test Case 2: (A) original parameters α^i plotted against the corresponding learned parameters $\hat{\alpha}^i$ and (B) evolution of $\theta^{N_P+1}(\tau)$, of $\hat{\theta}^{N_P+1}(\tau)$ and of the estimation of $\hat{\alpha}^{N_P+1}$. See caption of Figure 5 for the notation

different from those obtained for the Test Case 1 (see Figure 5(A)). Indeed, for Test Case 2 the parameters of the original model g are not related with those of the learned one $\hat{g}$ by a one-to-one relationship.

At first sight, we are tempted to conclude that the algorithm of Section 2.2 failed its goal of finding a good description of the training individuals. However, if we employ the learned model, as for Test Case 1, to predict the evolution of a new individual (by observing it over the time interval $[0, \mathcal{T}^{\text{obs}}]$, where $\mathcal{T}^{\text{obs}} = 0.5$, and by estimating the parameter $\hat{\alpha}^{N_P+1}$ through the EKF algorithm), we still obtain good results (see Figure 6(B)). As a matter of fact, by repeating the above mentioned protocol for 1000 synthetically generated individuals, we obtain an overall relative error of 3.7×10^{-3}.

Since the obtained error is similar to that obtained for Test Case 1, we conclude that also for Test Case 2 the learned model is a faithful mathematical description of the original one. However, the model learned in Test Case 2 cannot be obtained from the original one simply by a change of variable in α. Hence, we say that model $\tilde{g}$ is a *non-trivial reformulation* of model g.

To provide an example that shows how a non-trivial reformulation of Model (19) can be obtained, we consider the following model

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_1(\tau) = (\tilde{\alpha} + \theta_2(\tau))\theta_1(\tau) & \tau \in (0, \mathcal{T}], \\ \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_2(\tau) = 0 & \tau \in (0, \mathcal{T}], \\ \theta_1(0) = \tilde{\theta}_{1,0}, \theta_2(0) = \tilde{\theta}_{2,0}. \end{cases} \tag{20}$$

If we set $\tilde{\alpha}(\omega) = \alpha(\omega) - \theta_{2,0}(\omega)$, $\tilde{\theta}_{1,0}(\omega) = \theta_{1,0}(\omega)$ and $\tilde{\theta}_{2,0}(\omega) = \theta_{2,0}(\omega)$ for any $\omega \in \Omega$, then it follows that Models (19) and (20) generate the same data (i.e., $\theta_1(\tau, \omega) = \theta_{1,0}(\omega)e^{\alpha(\omega)\tau}$ and $\theta_1(\tau, \omega) = 0$).

A less trivial example, where we do not have a constant state as in Model (19), is given by the following model

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_1(\tau) = \alpha\theta_1(\tau) & \tau \in (0, \mathcal{T}], \\ \dfrac{\mathrm{d}}{\mathrm{d}\tau}\theta_2(\tau) = -\alpha\theta_2(\tau) & \tau \in (0, \mathcal{T}], \\ \theta_1(0) = \theta_{1,0}, \theta_2(0) = \theta_{2,0}. \end{cases} \tag{21}$$

A non-trivial reformulation of Model (21) is given by

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}\tau}\theta_1(\tau) = (\tilde{\alpha} + \theta_1(\tau)\theta_2(\tau))\theta_1(\tau) & \tau \in (0, \mathcal{T}], \\ \frac{\mathrm{d}}{\mathrm{d}\tau}\theta_2(\tau) = -(\tilde{\alpha} + \theta_1(\tau)\theta_2(\tau))\theta_2(\tau) & \tau \in (0, \mathcal{T}], \\ \theta_1(0) = \tilde{\theta}_{1,0}, \theta_2(0) = \tilde{\theta}_{2,0}, \end{cases} \tag{22}$$

and by setting $\tilde{\alpha}(\omega) = \alpha(\omega) - \theta_{1,0}(\omega)\theta_{2,0}(\omega)$, $\tilde{\theta}_{1,0}(\omega) = \theta_{1,0}(\omega)$ and $\tilde{\theta}_{2,0}(\omega) = \theta_{2,0}(\omega)$ for any $\omega \in \Omega$. It easy to check that, due to the fact that the product $\theta_1(\tau)\theta_2(\tau)$ is constant in time, the two models produce the same data.

We notice that in both cases (Models (19) and (21)) there is a quantity $C(\theta)$ that is constant in time (we have $C(\theta) = \theta_2$ and $C(\theta) = \theta_1\theta_2$, respectively). This quantity is a constant value characterizing individuals, as much as the parameters α: in both the above considered examples, indeed, the map between $(\alpha, C(\theta_0))$ and $\left(\tilde{\alpha}, C\left(\tilde{\theta}_0\right)\right)$ is one-to-one. Conversely, in the formulations considered above, the constant $C(\theta)$ is embedded into the state, which could be reduced to a single variable. In other words, we are trying to model more than necessary: the state could be reduced to the first variable $\theta_1(\tau)$, and the second variable $\theta_2(\tau)$ can be obtained by solving $C(\theta(\tau)) = C(\theta_0)$. For this reason, we say that a model that admits (respectively, does not admit) a non-trivial reformulation is *non-minimal* (respectively, *minimal*). In Section 2.3.3 we state rigorous definitions of these concepts.

To motivate the importance of model minimality, we recall that the parameter $\hat{\alpha}^{N_P+1}$ estimated through a non-trivial reformulation g of a model $\hat{g}$ is not related to the original parameter α^{N_P+1} by a one-to-one relationship. Hence, it cannot be employed to infer knowledge on the $N_P + 1$ individual (classification purposes) and it hampers the interpretability of the learned model.

2.3.3 | Definition of reformulation

In what follows, we identify a model written in the form of Equation (14) with the triplet $(g, \mathcal{P}, \mathcal{A})$. Moreover, we denote the solution map associated with such a model as $s_g : \mathcal{P} \times \mathcal{A} \times [0, +\infty) \to \mathcal{P}$, defined as $s_g(\theta_0, \alpha, \tau) = \theta(\tau)$, where $\theta(\tau)$ is the solution of Equation (14). We introduce the following definitions.

Definition 1. We say that a function $\tilde{A} : \mathcal{P} \times \mathcal{A} \to \tilde{\mathcal{A}}$ is:

- *trivial*, if it is constant in its first argument; *non-trivial*, otherwise;
- *non-pathological*, if $\left\{\tilde{A}(\theta_0, \alpha) : \alpha \in \mathcal{A}\right\} = \tilde{\mathcal{A}}$ for any $\theta_0 \in \mathcal{P}$; *pathological*, otherwise.

Definition 2. We say that $\left(\tilde{g}, \mathcal{P}, \tilde{\mathcal{A}}\right)$ is a *reformulation* of $(g, \mathcal{P}, \mathcal{A})$ if there exists a non-pathological function $\tilde{A} : \mathcal{P} \times \mathcal{A} \to \tilde{\mathcal{A}}$ such that:

$$g\left(s_g(\theta_0, \alpha, \tau), \alpha\right) = \tilde{g}\left(s_g(\theta_0, \alpha, \tau), \tilde{A}(\theta_0, \alpha)\right) \quad \forall \theta_0 \in \mathcal{P}, \alpha \in \mathcal{A}, \tau \geq 0. \tag{23}$$

Moreover, we say that $\left(\tilde{g}, \mathcal{P}, \tilde{\mathcal{A}}\right)$ is a

- *non-trivial reformulation* of $(g, \mathcal{P}, \mathcal{A})$, if (23) holds for some non-trivial non-pathological function $\tilde{A}$;
- *trivial reformulation* if all the non-pathological functions $\tilde{A}$ satisfying (23) are trivial.

Definition 3. A model $(g, \mathcal{P}, \mathcal{A})$ is minimal if it does not admit any non-trivial reformulation.

We remark that the hypothesis that the function $\tilde{A}$ be non-pathological is needed to avoid pathological situations, such as the case when a reformulation if obtained by a mere piece-wise redefinition of the coefficient. Moreover, we remark that the definition of reformulation is motivated by the following result, for which the proof is given in A.

Proposition 1. Let $\left(\tilde{g},\mathcal{P},\tilde{\mathcal{A}}\right)$ be a reformulation of $(g,\mathcal{P},\mathcal{A})$ through the map $\tilde{A}:\mathcal{P}\times\mathcal{A}\to\tilde{\mathcal{A}}$. Then we have

$$s_g(\theta_0,\alpha,\tau)=s_{\tilde{g}}\left(\theta_0,\tilde{A}(\theta_0,\alpha),\tau\right)\quad\forall\theta_0\in\mathcal{P},\alpha\in\mathcal{A},\tau\geq 0. \tag{24}$$

We remark that minimality is an intrinsic property of models: it is not affected, for instance, by invertible transformations of either the inputs, the outputs or the parameters (such as scaling or nondimensionalization). Even if, as shown in Test Case 2, the methods proposed in this paper can be applied to any model (minimal and non-minimal ones), working with minimal models enhances the quantity of information that can be extracted from the learned model. This leads to the issue of devising tests to check minimality of models. Clearly, since the original model is not known in practical applications, the minimality of the model to be learned cannot in practice be analytically checked through the Definition 3. In the case that some a priori knowledge about the physical phenomenon is available, minimality could be deduced by means of physics-based considerations. The development of a posteriori numerical tests aimed at checking model minimality will be the subject of future works.

We recall an important concept, that is *identifiability of models*.[51,52]

Definition 4. Two parameters $\alpha_1,\alpha_2\in\mathcal{A}$ are said *indistinguishable* for the model $(g,\mathcal{P},\mathcal{A})$ if

$$\exists\theta_0\in\mathcal{P}\quad\forall\tau\geq 0\qquad s_g(\theta_0,\alpha_1,\tau)=s_g(\theta_0,\alpha_2,\tau). \tag{25}$$

Moreover, we define $I(\alpha)=\{\alpha^*$ s. t. α and α^* are indistinguishable$\}$.

Definition 5. We say that a model $(g,\mathcal{P},\mathcal{A})$ is identifiable if $I(\alpha)=\{\alpha\}$ for any $\alpha\in\mathcal{A}$, that is

$$\forall\theta_0\in\mathcal{P},\alpha_1\neq\alpha_2\in\mathcal{A}\quad\exists\tau^*>0\quad s_g(\theta_0,\alpha_1,\tau^*)\neq s_g(\theta_0,\alpha_2,\tau^*) \tag{26}$$

In fact, if a model is identifiable, then the map from the parameter-trajectory map $\alpha\mapsto\{s_g(\theta_0,\alpha,\tau)\}_{\tau\geq 0}$ is one-to-one, for each $\theta_0\in\mathcal{P}$. Conversely, if a model is not identifiable, then, for some choices of the initial condition $\theta_0\in\mathcal{P}$ the DA assimilation problem of identifying the parameter α from the trajectory $\{s_g(\theta_0,\alpha,\tau)\}_{\tau\geq 0}$ is ill-posed. Hence, in this paper we assume that we work with identifiable models. We remark that for linear models identifiability can be assessed by standard criteria (e.g., by studying the spectral properties of the observability matrix[43]). Nonetheless, in this paper we consider the more general case of nonlinear parametric dynamical models.

2.3.4 | Uniqueness of representation of models

Clearly, every model (even the minimal ones) admit trivial reformulations. Indeed, the parameter α is never measured and so it is subject to changes of variables. This is linked to the black-box structure of the model learning Problem (P2). As a consequence, the solution of Problem (P2) is not unique. In order to transform such a problem into a problem with a unique solution, we need to select a representative model inside each class of trivial reformulations. In this section, we suggest a strategy to accomplish this goal. Specifically, we restrict the space of candidate models $\mathcal{G}_n$ by imposing some a priori constraints on the dependence of g on α.

Let us suppose that there exists a function, $\Gamma\{g(\cdot,\alpha)\}(\alpha)$, expressed as a combination of g and of its derivatives in θ (of any order) evaluated at some given points of $\mathcal{P}$, invertible in α. Then, we perform learning by restricting the search space to functions g satisfying the constraint $\Gamma\{g\}(\alpha)=\alpha$. In what follows we will show that, in this way, the solution of the model learning problem is unique.

This strategy can be interpreted as that of giving a physical meaning to the parameter α, which would be otherwise a black-box object. As a matter of fact, the constraint $\Gamma\{g\}(\alpha)=\alpha$ provides a definition of the parameter α. Let us consider the following example.

Example 1. Select some $\theta^* \in \mathcal{P}$, and set $\Gamma\{g\}(\alpha) = g(\theta^*, \alpha)$. In this case, performing optimization under the constraint $g(\theta^*, \alpha) = \alpha$ is equivalent to defining the parameter as the time rate of change of the state at θ^*.

The effectiveness of the proposed strategy is supported by the following proposition (the proof is given in A).

Proposition 2. Let $(g, \mathcal{P}, \mathcal{A})$ be an identifiable model. Suppose that there exists an operator $\Gamma\{g(\cdot, \alpha)\}(\alpha)$, which acts on g as a function of θ, which is injective in α. Then, there exists a unique trivial reformulation of $(g, \mathcal{P}, \mathcal{A})$ (that we denote by $\left(\tilde{g}, \mathcal{P}, \tilde{\mathcal{A}}\right)$) such that $\Gamma\{\tilde{g}\}(\tilde{\alpha}) = \tilde{\alpha}$ for any $\tilde{\alpha} \in \tilde{\mathcal{A}}$.

From an implementative viewpoint, the constraint can be imposed in two alternative ways. The first (that we denote as *weak constraint*) consists in adding to the loss function of Problem (P2) the following penalization term

$$\frac{w^2}{2} \int_{\mathcal{A}} |\alpha - \Gamma\{g\}(\alpha)|^2 d\alpha, \tag{27}$$

where w denotes a weighting constant. From a numerical viewpoint, we evaluate the integral by Latin Hypercube Sampling[53] of the parameters space $\mathcal{A}$. When computing the gradients of the loss function (see Equation 13), the gradient of the penalization term with respect to μ is added to $\nabla_\mu \mathcal{J}$.

The second strategy (that we denote as *strong constraint*) consists in a manipulation of the ANN architecture, such that the constraint is exactly satisfied. Let us consider for instance Example 1. In this case, we define the space $\mathcal{G}_n$ as

$$\mathcal{G}_n = \{(\theta, \alpha) \mapsto \mathscr{H}(\theta, \alpha; \mu) - \mathscr{H}(\theta^*, \alpha; \mu) + \alpha : \mu \in \mathbb{R}^n\},$$

where $\mathscr{H}(\theta, \alpha; \mu)$ denotes an ANN, whose input is given by (θ, α) and where $\mu \in \mathbb{R}^n$ represents the vector of ANN weights and biases. In this manner, the constraint $\alpha = \Gamma\{g\}(\alpha) = g(\theta^*, \alpha)$ for any $\alpha \in \mathcal{A}$ is automatically satisfied by any function $g \in \mathcal{G}_n$. We notice that, in this case, the last layer of biases can be removed from the ANN since its action in canceled by construction.

We remark that, while the weak constraint strategy is always feasible, the strong constraint feasibility depends on the specific form of the function Γ.

3 | NUMERICAL ILLUSTRATIONS

In this section, we present the results of several numerical test cases, showing the effectiveness and the noise-robustness of the proposed methods. All the results shown in this paper were produced within the MATLAB library `model-learning` (publicly available in a GitHub repository1), which was suitably extended with the implementation of the methods proposed here.

3.1 | Test Case 1: Gaining a unique solution

In order to assess the effectiveness of the strategy proposed in Section 2.3.4 to identify a unique representative model among a class of trivial reformulations, we consider again Test Case 1, already considered in Section 2.3.1 (see Equation (17)). Since this model is minimal (in the sense of Definition 3), when learning from its data we obtain one of its infinitely many trivial reformulations. As a matter of fact, by running the learning algorithm several times, with different random initialization of the ANN parameters, we obtain a different model every time, as testified by Figure 7(A)–(C), which show, for three different models learned from the same dataset, the values of the original parameters α^i against the value of the corresponding learned parameters $\hat{\alpha}^i$. Even if in each case an underlying one-to-one relationship linking α to $\hat{\alpha}$ can be easily detected, this relationship is different in each case.

In order to obtain a unique solution, we consider the constraint $\Gamma\{\tilde{g}\}(\tilde{\alpha}) = \tilde{\alpha}$ for $\Gamma\{\tilde{g}\}(\tilde{\alpha}) = \tilde{g}(\theta^*, \tilde{\alpha})$, by setting $\theta^* = 1.1$. By Proposition 2, we know that there is a unique model $\left(\tilde{g}, \mathcal{P}, \tilde{\mathcal{A}}\right)$, trivial reformulation of the model of Equation (17), such that $\tilde{g}(\theta^*, \tilde{\alpha}) = \tilde{\alpha}$. It is easy to check that this model has a right-hand side defined as $\tilde{g}(\theta, \tilde{\alpha}) = \tilde{\alpha}\theta/\theta^*$ and that the parameters of such model are related to those of the original one by the relationship $\tilde{\alpha}(\omega) = \psi(\alpha(\omega)) = \theta^* \alpha(\omega)$.

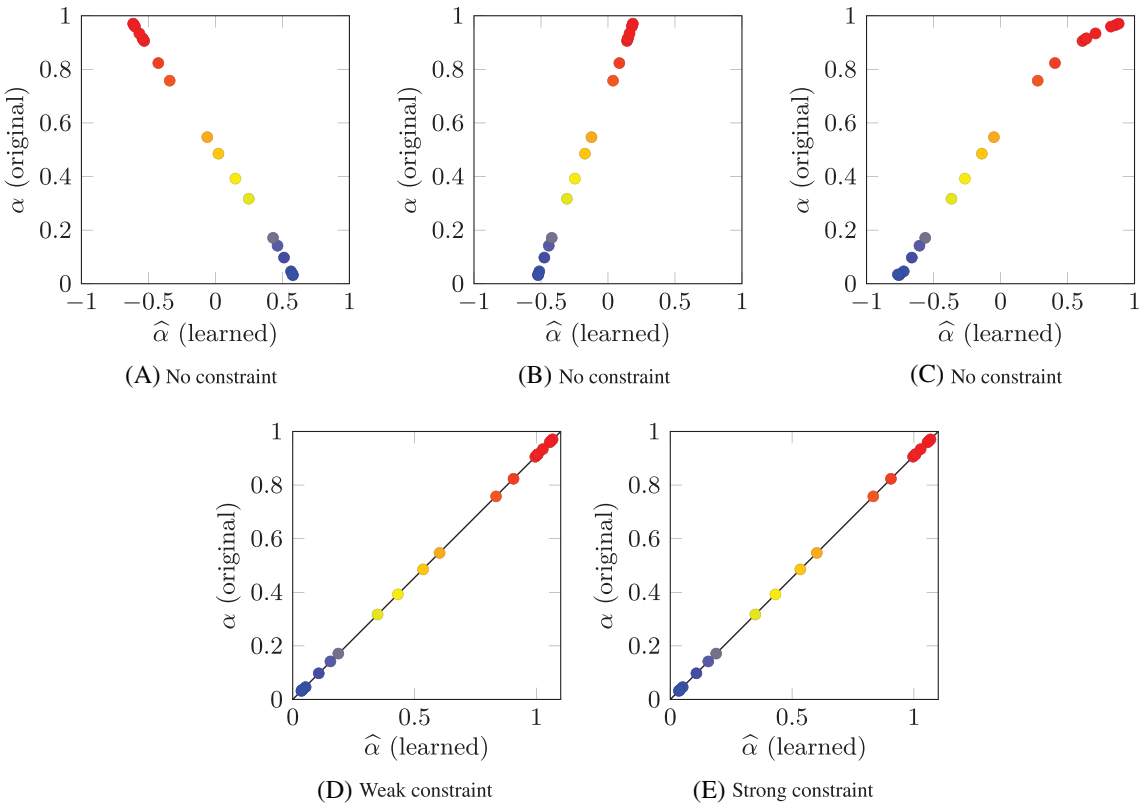


FIGURE 7 Test Case 1: original parameters α^i plotted against the corresponding learned parameters $\hat{\alpha}^i$ for the training individuals ($i = 1, ..., N_P$). The models considered in (A), (B) and (C) are obtained with the non-constrained strategy, the model of (D) and (E) are obtained with the weak and the strong strategies, respectively. In (D) and (E), the black line represents the analytical solution $\alpha = \psi^{-1}(\tilde{\alpha})$

We perform several runs of the learning algorithm of Section 2.2 with different random initialization of the ANN parameters, both with the weak and with the strong constraint approach. Unlike the non-constrained case, this time (with both approaches) we obtain the same model in every run (up to numerical errors). In Figure 7(D),(E) we show the α^i versus $\hat{\alpha}^i$ plot obtained with the weak and the strong constraint strategies, respectively. In both figures we superimpose (black line) the relationship $\alpha = \psi^{-1}(\tilde{\alpha}) = \tilde{\alpha}/\theta^*$, corresponding to the analytical exact solution of the learning problem. The remarkably good agreement between the analytical solution and the learned parameters demonstrates the performance of the learning algorithm and of the strategy proposed in Section 2.3.4 to select a unique representative model.

As noticed in Section 2.3.1, the original and the learned model right-hand sides are related by the relationship $g(\theta,\alpha) = \tilde{g}(\theta,\Gamma\{g\}(\alpha)) = \tilde{g}(\theta,\theta^*\alpha)$ for any $\theta \in \mathcal{P}$ and $\alpha \in \mathcal{A}$. In order to check that the learned models are consistent with this relationship, we show in Figure 8 a comparison between the functions $g(\theta,\alpha)$ and $\hat{g}(\theta,\Gamma\{g\}(\alpha))$ (where $\hat{g}$ denotes the learned model) in the portion of the state-parameter plane $\mathcal{P} \times \mathcal{A}$ spanned by the training dataset. We can see that, with both the weak and the strong constraint strategies, the results are consistent with the analytical solution, except for the region of the state-parameter plane corresponding to low α and large θ. The reason for this discrepancy lies in the fact that, when α^i is low, the state of the i-th individual never reaches the region where θ^i is large. Hence, the training dataset does not contain any information about the behavior of the phenomenon in this region of the state-parameter plane and we cannot expect the data-driven apparatus to learn anything that is associated with this region (*extrapolation* effect). Nonetheless, if the training dataset contains a sufficiently representative collection of the possible behaviors of the phenomenon to be modeled, spanning the values of parameters and initial conditions of interest and the time horizon of interest, even if the learned model is not reliable outside the state-parameter region spanned by the training dataset, in practical applications the complementary of this region is never reached. Hence, in this case, the risks associated with extrapolation do not represent a threat. However, it is important to be aware of this limitation. A good practice would be to check a posteriori, when the learned model is used for prediction purposes, that the state-parameter pair does not move "too far" from the region spanned by the training dataset (i.e., $\left\{\left(\hat{\theta}^i_j, \hat{\alpha}^i\right)\right\}^{i=1,...,N_P}_{j=1,...,N_S}$).

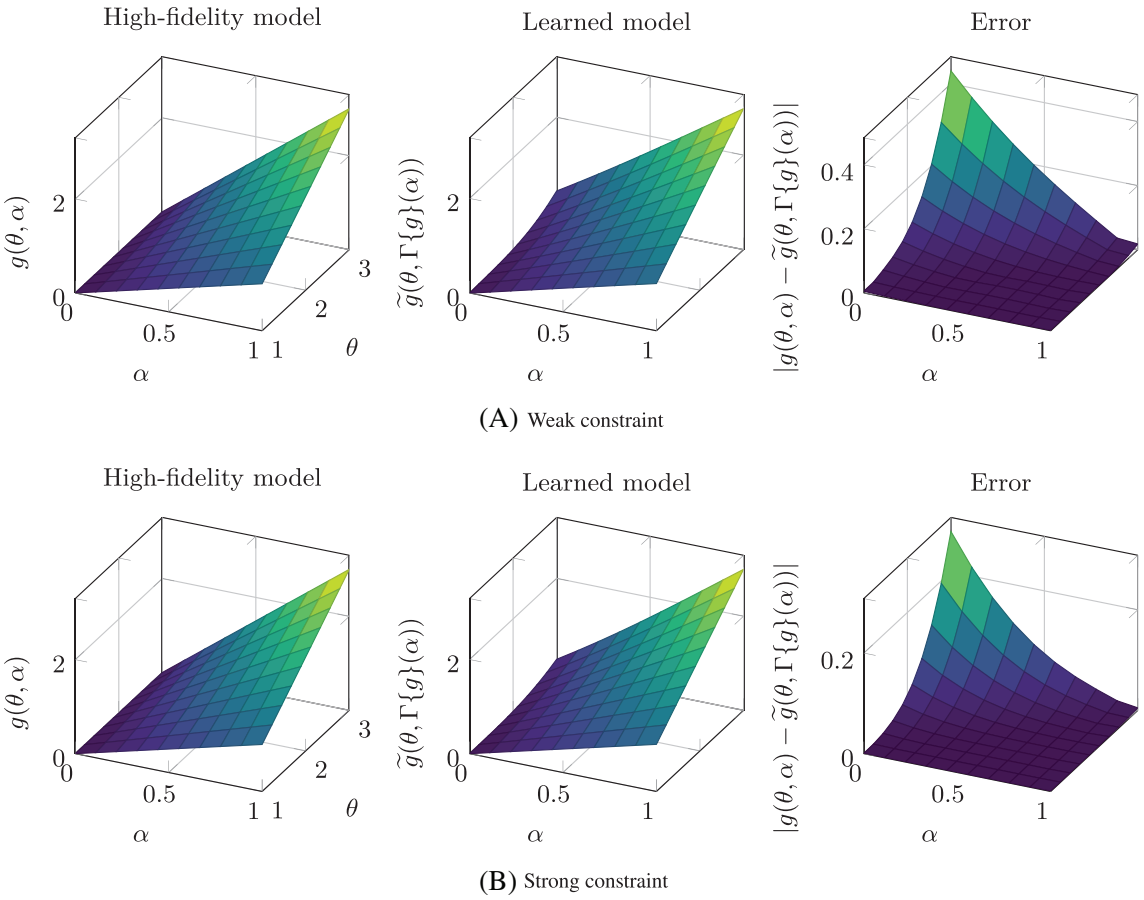


FIGURE 8 Test Case 1: comparison between the original model right-hand side $g(\theta,\alpha)$ with the learned model right-hand side, mapped back to the state-parameter plane of the original model (i.e., $\tilde{g}(\theta,\Gamma\{g\}(\alpha))$). The last column shows the absolute error between the two functions. (A) Weak constraint. (B) Strong constraint

3.2 | Test Case 1: Noise robustness of the learning algorithm

In the numerical tests considered so far, we have considered the case when measurements of the variables $\theta^i(\tau)$ are available without error. However, in practical applications, measurements are never error-free. Moreover, in case the variables $\hat{\theta}_j^i$ are obtained as results of a DA algorithm (as in the framework proposed in Section 2.1), their values are affected by the error introduced by the DA procedure. Hence, it is of utmost importance to investigate how much the proposed learning algorithm is robust to noise.

With this aim, we consider again Test Case 1, this time by assuming that the available measurements are affected by noise. Specifically, we set $\hat{\theta}_j^i = \theta^i(\tau_j) + \varepsilon_j^i$, where the observation errors $\varepsilon_j^i \sim \mathcal{N}(0,U)$ are independent and identically distributed. By assuming that the stochastic variables ε_j^i are the result of a discrete sampling of the white noise $\sigma\frac{dB(\tau)}{d\tau}$ (where $B(\tau)$ is a Wiener process and σ is the noise magnitude), we set $U = \sigma^2/\Delta\tau$. In particular, we set $\sigma = \sigma_{\text{offline}}$ to generate the training data (offline noise) and $\sigma = \sigma_{\text{online}}$ to generate the data to test the learned model (online noise). In all the test cases presented in this paper, we assumed the noise magnitude as given. A possible future development could be using suitable algorithms to automatically estimate the noise magnitude (see, e.g., Bavdekar et al.[54]).

We add a random noise to the dataset used in Sections 2.3.1 and 3.1 for Test Case 1, with different magnitudes of σ_{offline}, and we run the learning algorithm. For the sake of brevity in this section we only consider the strong constraint strategy, but similar results are obtained with the weak constraint strategy too. In Figure 9(A) we show the α^i versus $\hat{\alpha}^i$ plot obtained for σ_{offline} ranging between $10^{-2}\text{s}^{1/2}$ and $10^{-5}\text{s}^{1/2}$. We notice that, as the noise decreases, the accuracy in the attribution of the parameters $\hat{\alpha}^i$ increases.

To quantitatively assess the noise-robustness of the method, we generate 1000 random test individuals and we add a random noise (with different levels of magnitude σ_{online}), we apply the EKF algorithm in the time interval $\left[0, T^{\text{obs}}\right]$, in

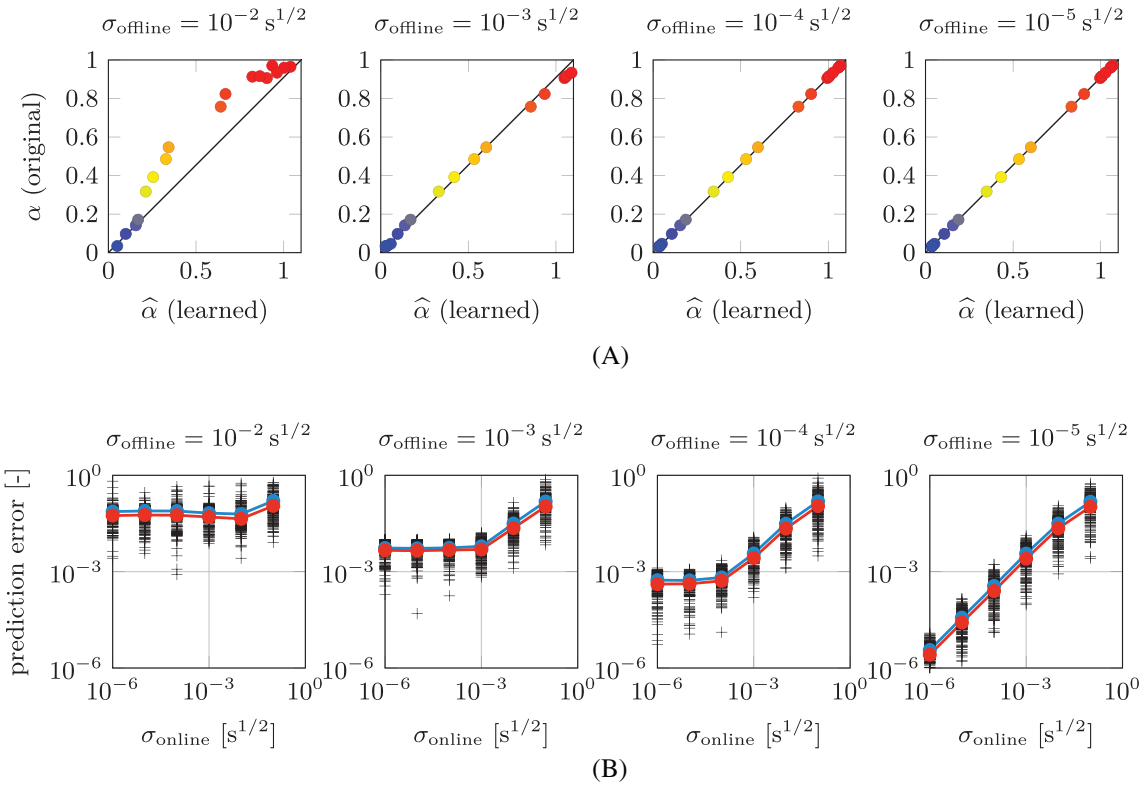


FIGURE 9 Test Case 1: (A) original parameters α^i plotted against the corresponding learned parameters $\hat{\alpha}^i$ for the training individuals ($i = 1, ..., N_P$), for different noise magnitudes σ_{offline} and (B) overall L^2 relative prediction errors obtained for different values of σ_{offline} and σ_{online}. In (B), each black cross represent a testing individual, the blue curve represent the arithmetic mean, the red curve the geometric mean

order to estimate the parameters $\hat{\alpha}^i$ associated with those individuals, and finally we predict their evolution in the interval $\left(\mathcal{T}^{\text{obs}}, \mathcal{T}\right]$. In Figure 8 we show the overall normalized L^2 error between the predictions and the exact solutions, for the different values of σ_{offline} and σ_{online}. We notice that, when σ_{online} is large compared to σ_{offline}, the prediction error has a trend which is proportional to σ_{online}. However, for low values of σ_{online}, the prediction error saturates to a level that depends on σ_{offline}. Indeed, the learned model $\hat{g}$ is an approximate surrogate for the laws governing the phenomenon (represented in this example by the original model g). As such, the learned model is affected by a *model error*, that is to say it is endowed with a limited capability of describing and predicting the behavior of the phenomenon. In conclusion, when $\sigma_{\text{online}} \gg \sigma_{\text{offline}}$, the error introduced by the online noise prevails over the model error, which explains the linear trend between σ_{online} and the prediction error. Conversely, when $\sigma_{\text{online}} < \sigma_{\text{offline}}$, the model error comes into play and the decreasing trend of the prediction error ends.

We notice that in most practical applications, measurements are available with the same quality in the online and the offline stage. Hence, we repeat the test performed in Figure 9(B), by setting $\sigma_{\text{online}} = \sigma_{\text{offline}}$, that is by simultaneously decreasing the offline and the online error. In order to provide a benchmark value for the errors obtained in this manner, we perform the same test, this time by employing the original model g itself in the DA and in the prediction stage. The prediction errors so obtained are reported in Figure 10. Such errors represent the minimum error that can be attained for a given level of noise. Indeed, they are obtained in the idealized case of absence of any model error (they are only affected by the online noise). In Figure 9 we also report the prediction errors obtained by replacing the ground-truth model g with its data-driven surrogates. The closeness of the error curve obtained with the ANN-based model to that obtained with the original one demonstrates that the effect of the model error introduced by replacing the original model with its surrogate is small compared to the error due to the noise. Hence, we conclude that the learned model can be reliably used to interpret data and to make predictions for the phenomenon.

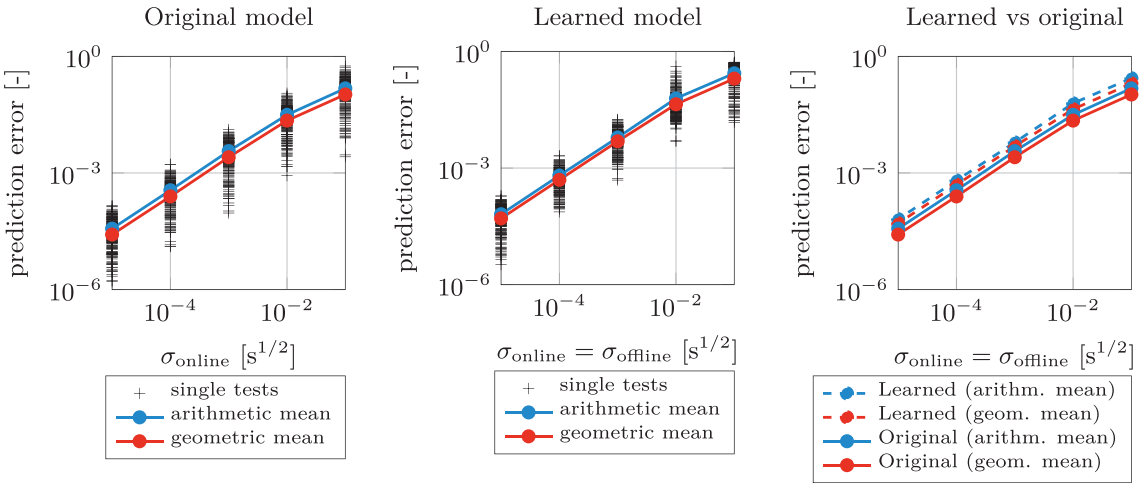


FIGURE 10 Test Case 1: overall L^2 relative prediction errors for different values of σ_{online} obtained by employing the original model (left) and the learned models (center), by setting $\sigma_{\text{offline}} = \sigma_{\text{online}}$ (we use the same conventions of Figure 9(B)). Finally, we show a comparison of the arithmetic and the geometric means of the prediction errors, for the original and the learned models (right)

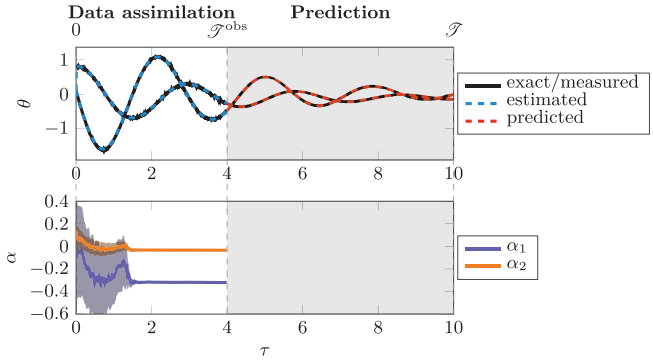


FIGURE 11 Test Case 3: evolution of $\theta^{N_P+1}(\tau)$, of $\hat{\theta}^{N_P+1}(\tau)$ and of the estimation of $\hat{\alpha}^{N_P+1}$. See caption of Figure 5 for the notation

3.3 | Test Case 3: Nonlinear second-order dynamics

In order to test the capabilities of the proposed method to deal with more complex test cases than the linear first-order dynamics considered in the previous sections, we consider the following second-order nonlinear model

$$\begin{cases} \dfrac{d^2}{d\tau^2}\omega(\tau) + \beta \sin(\omega(\tau) - \alpha_1) + \alpha_2 \dfrac{d}{d\tau}\omega(\tau) = 0 \quad \tau \in (0, \mathcal{T}], \\ \omega(0) = \omega_0, \\ \dfrac{d}{d\tau}\omega(0) = \psi_0, \end{cases} \tag{28}$$

describing the dynamics of a nonlinear oscillator. The state of the model is given by $\theta(\tau) = \left(\omega(\tau), \frac{d}{d\tau}\omega(\tau)\right)$. The two parameters α_1 and α_2 represent the steady-state value of the variable ω and the damping coefficient, respectively, while $\beta = 5\ \text{s}^{-2}$ is a constant. We generate $N_P = 20$ training samples, by randomly sampling the state space $\theta_0 \in [-1, 1] \times [-0.1\ \text{s}^{-1}, 0.1\ \text{s}^{-1}]$ and the parameters space $\alpha \in [-0.2, 0.2] \times [0.5\ \text{s}^{-1}, 1.5\ \text{s}^{-1}]$ and by setting $\mathcal{T} = 10$ s. For the time discretization we set $\Delta\tau = 1 \cdot 10^{-2}$ s.

As in Section 3.2, we build several data-driven models with different values of σ_{offline}. For each model, we set a single hidden layer with eight neurons. Then, we randomly generate several test samples; we add noise to the model output; we perform DA in the time interval $\left[0, \mathcal{T}^{\text{obs}}\right]$, for $\mathcal{T}^{\text{obs}} = 4$ s; finally, we predict the evolution in the time interval $\left(\mathcal{T}^{\text{obs}}, \mathcal{T}\right]$, for $\mathcal{T} = 10$ s.

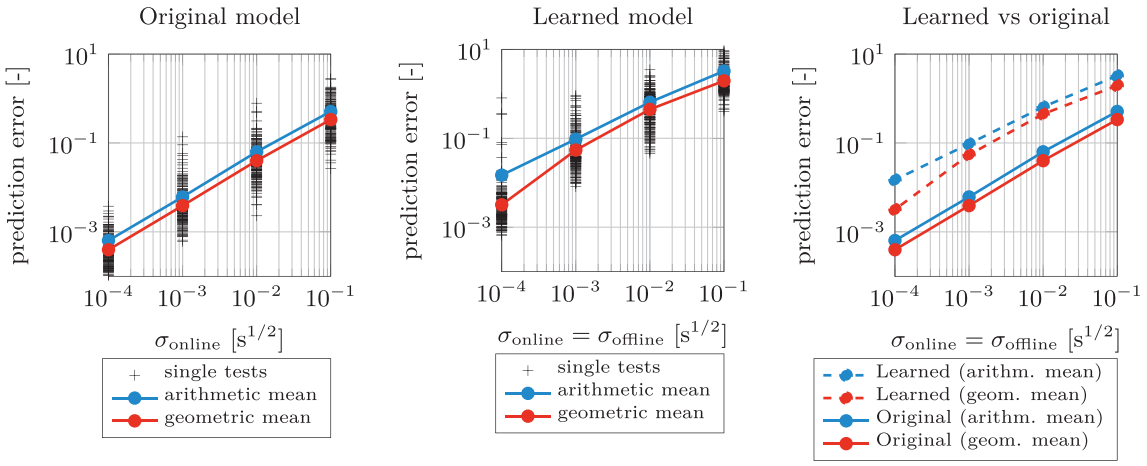


FIGURE 12 Test Case 3: overall L^2 relative prediction errors for different values of σ_{online} obtained by employing the original model (left) and the learned models (center), by setting $\sigma_{\text{offline}} = \sigma_{\text{online}}$ (we use the same conventions of Figure 9(B)). Finally, we show a comparison of the arithmetic and the geometric means of the prediction errors, for the original and the learned models (right)

We report in Figure 11 the predicted evolution of the system state for a random sample, in the case $\sigma_{\text{offline}} = \sigma_{\text{online}} = 10^{-4}\ \text{s}^{1/2}$. Then, in Figure 12, we compare the overall L^2 relative prediction errors obtained, for different noise levels, by employing the data-driven models and the exact model of Equation (28). Even if the errors obtained with the data-driven models are larger than those obtained with the exact model, compared to Test Case 1, the error still converges to zero when the noise decreases.

3.4 | Test Case 4: A synthetic test case of hypertension evolution

As a final test case (Test Case 3) we consider the example introduced in Section 1.3 as a motivation for the methods proposed in this paper. In this test case, the fast-scale phenomenon is the blood circulation, which we here describe by means of the two-stage Windkessel model of Equation (1). The evolution of the fast-scale variables $x(t) = (P_{\text{p}}(t), P_{\text{d}}(t))$ is conditioned by the four parameters $\theta = (R_{\text{p}}, C_{\text{p}}, R_{\text{d}}, C_{\text{d}})$, which evolve on a slower time scale τ. In order to test the proposed methods, we synthetically generate data that we use in place of experimental measurements. With this aim, we consider the following model for the slow scale evolution of the parameters θ

$$\begin{cases} \dfrac{\text{d}}{\text{d}\tau} R_{\text{p}}(\tau) = 0 & \tau \in (0, \mathcal{T}], \\ \dfrac{\text{d}}{\text{d}\tau} C_{\text{p}}(\tau) = 0 & \tau \in (0, \mathcal{T}], \\ \dfrac{\text{d}}{\text{d}\tau} R_{\text{d}}(\tau) = \alpha R_{\text{d}} & \tau \in (0, \mathcal{T}], \\ \dfrac{\text{d}}{\text{d}\tau} C_{\text{d}}(\tau) = -\alpha C_{\text{d}} & \tau \in (0, \mathcal{T}], \\ R_{\text{p}}(0) = R_{\text{p},0}, C_{\text{p}}(0) = C_{\text{p},0}, R_{\text{d}}(0) = R_{\text{d},0}, C_{\text{d}}(0) = C_{\text{d},0}. \end{cases} \tag{29}$$

Model (29) describes the evolution of hypertension due to a progressive increase of the distal arterial resistance R_{d}, while the distal arterial capacitance C_{d} accordingly decreases so that the characteristic time constant of the arterial system (corresponding to the product $R_{\text{d}}C_{\text{d}}$) is preserved. On the other hand, the properties of the proximal arterial system (i.e., R_{p} and C_{p}) are not subject to long-term variations. The slow-scale coefficient $\alpha \in \mathcal{A} := [0, 1 \times 10^{-2}\text{week}^{-1}]$ represents the disease severity (more precisely, we have $\alpha = \log(2)/\mathcal{T}_{\text{double}}$, where $\mathcal{T}_{\text{double}}$ is the doubling time of R_{d}).

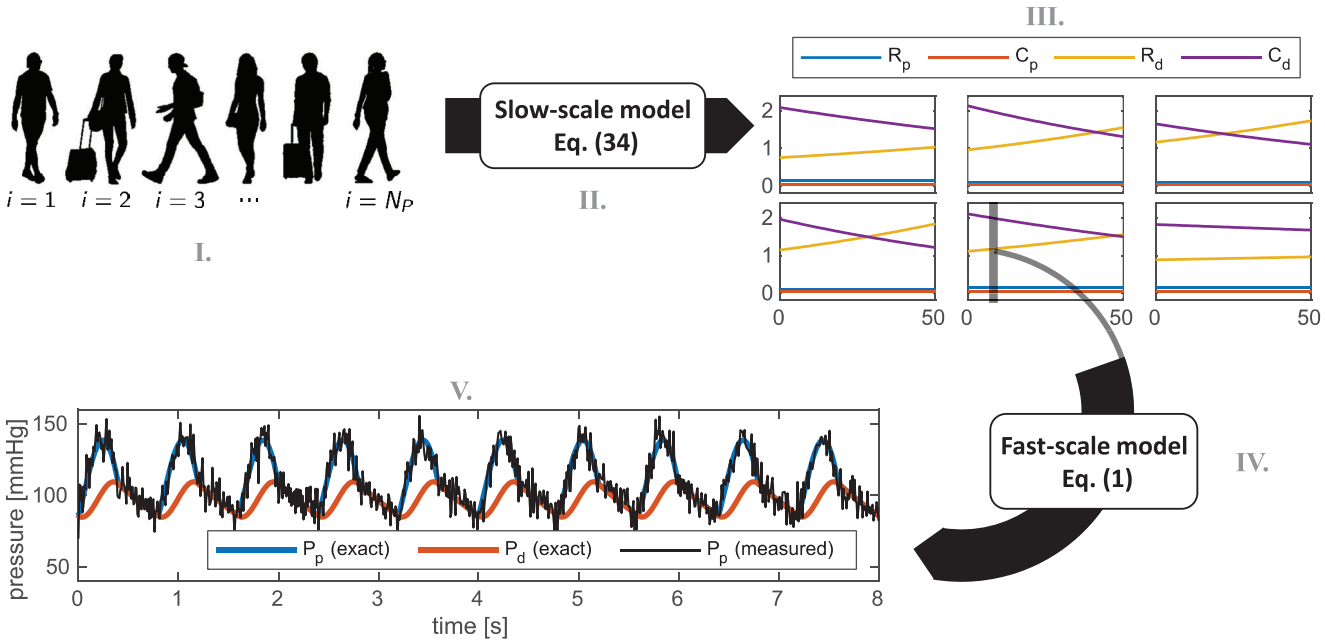


FIGURE 13 Test Case 4: visual representation of the five steps followed to generate synthetic data (see Section 3.4.1)

Parameter	Value	Parameter	Value	Units
R_p^{min}	8.1757×10^{-2}	R_p^{max}	1.7326×10^{-1}	mmHg s mL^{-1}
C_p^{min}	3.1197×10^{-2}	C_p^{max}	3.1464×10^{-2}	mL mmHg^{-1}
R_d^{min}	6.9006×10^{-1}	R_d^{max}	1.1599	mmHg s mL^{-1}
C_d^{min}	1.1851	C_d^{max}	2.1731	mL mmHg^{-1}

TABLE 1 Test Case 4: Lower and upper bounds of the initial values of the slow-scale variables

3.4.1 | Synthetic data generation

We generate the data needed to test the methods proposed in this paper alongside the following steps, visually represented in Figure 13.

1. We generate 100 synthetic patients, each one characterized by an initial state $\theta_0 = (R_{p,0}, C_{p,0}, R_{d,0}, C_{d,0})$ and by a degree of disease severity α. Specifically, the i-th patient initial state θ_0^i and α^i are randomly selected by sampling with uniform probability the sets $\left[R_p^{min}, R_p^{max}\right] \times \left[C_p^{min}, C_p^{max}\right] \times \left[R_d^{min}, R_d^{max}\right] \times \left[C_d^{min}, C_d^{max}\right]$ (see Table 1) and $\mathcal{A}$.
2. For each patient, we solve the slow-time model of Equation (29), obtaining $\theta^i(\tau)$ for $\tau \in (0, \mathcal{T}]$, with $\mathcal{T} = 50$ weeks.
3. We suppose that we monitor each patient with a period of $\Delta\tau = 1$ week. Hence, we subdivide the interval $(0, \mathcal{T}]$ into the time instants $0 = \tau_1 < \tau_2 < ... < \tau_{N_S} = \mathcal{T}$, such that $\tau_{j+1} = \tau_j + \Delta\tau$.
4. At each time τ_j (for $j = 1, ..., N_S$) and for each patient i, we simulate the fast-scale patient circulation, by solving the Windkessel model (1) by setting $\theta = \theta^i(\tau_j)$. We simulate 10 heartbeats, in order to let the dynamical system reach a limit cycle, and then we record the result of the following 10 heartbeats. In this manner, we obtain z_j^i, namely the observation associated with the i-th patient at the j-th discrete slow-time. We recall that, in this test case, the observation vector z collects the measurements of the proximal arterial pressure at a discrete collection of time instants $(t_1, ..., t_K)$, with $t_{k+1} - t_k = \Delta t = 1e-2$ (i.e., we have $z = (P_p(t_1), ..., P_p(t_K))$.
5. In order to mimic the measurement error that unavoidably comes with any experimental measurement, we define the measured observation as $\tilde{z}_j^i = z_j^i + \chi_j^i$, where $\chi_j^i \sim \mathcal{N}(0, W)$ are independent and identically distributed observation errors. Similarly to what we have done in Section 3.2, we assume that the observation errors derive from a white

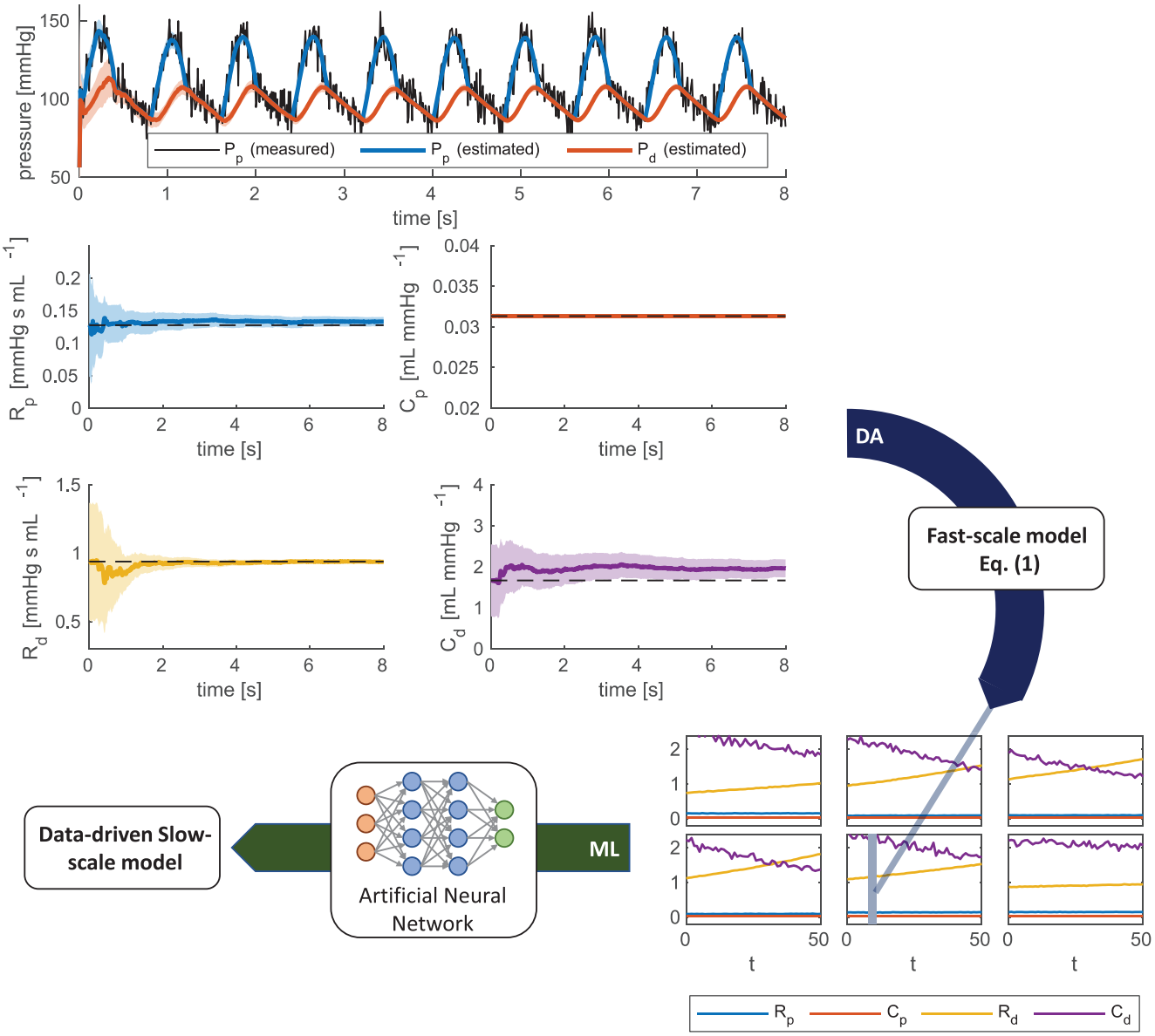


FIGURE 14 Test Case 4: visual representation of the procedure followed to generate a data-driven model for hypertension development (see Section 3.4.2). In the top part of the figure we represent in black the noisy measurement of P_p associated with a given patient i and a to given slow-time instant τ_j. The colored line represent the time evolution of estimated values of the system state (P_p and P_d) and of the four parameters (R_p, C_p, R_d and C_d). The shadowed areas represent the $\pm 3\sigma$ bands, where σ denotes the standard deviation of the estimate. By combining the estimated values of the parameters obtained for all patients and for all the slow-time instants, we obtain an estimate of their slow-scale evolution, represented in the bottom-right part of the figure

noise of magnitude $\rho = 0.75\,\text{mmHgs}^{-1/2}$. Hence, we set $W = \rho^2/\Delta t = (7.5\,\text{mmHg})^2$. This means that any pointwise measurement is affected by a relative error of the order of 10%.

Among the quantities generated through the above mentioned steps, the unique information available to the learning algorithm consists in the noisy observations $\left\{\tilde{z}_j^i\right\}_j^i$. The goal is that of climbing back those steps, in order to obtain the initial state of each patient, their degree of disease severity, and the laws governing the slow-scale arterial network remodeling. Although the data employed for this test case are synthetically generated, the assumptions made here are consistent with an hypothetical clinical use of the proposed procedure; indeed this type of measurements can be available, for patients with specific indications, with the accuracy assumed above.[55]

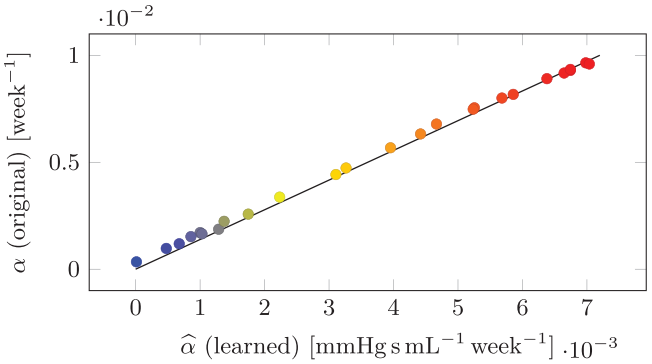


FIGURE 15 Test Case 4: original parameters α^i plotted against the corresponding learned parameters $\hat{\alpha}^i$ for the training individuals ($i = 1, ..., N_P$)

3.4.2 | Data-driven model for hypertension development

Starting from the noisy observation $\left\{\tilde{z}_j^i\right\}_j^i$, we build a data-driven model by following the procedure proposed in Section 2.1 and visually represented in Figure 14.

Specifically, we select $N_P = 25$ training patients. For each of them and for each slow-scale time instant τ_j, we consider the noisy observation $\tilde{z}_j^i$, collecting the proximal arterial pressure measurements along 10 heartbeats. By solving Problem (P1) by means of the EKF algorithm, we obtain an estimate of the slow-scale parameter vector for the i-th patient at time τ_j, denoted by $\hat{\theta}_j^i$. By collecting the estimated parameters for all i and j, we obtain the slow-scale training dataset, a subset of which is displayed in Figure 14.

Next, we employ this dataset to train a data-driven model. Specifically, since hypertension is related to an increase of the peripheral system resistance, we focus on the variable R_d. Hence, we solve Problem (P2) for $\theta(\tau) \coloneqq R_\mathrm{d}(\tau)$ by means of the algorithm proposed in Section 2.2. In order to guarantee the uniqueness of the solution for Problem (P2) and to enhance the interpretability of the results, we adopt the constraint of Example 1, for $\theta^* = R_\mathrm{d}^* \coloneqq 0.75$ mmHg s mL^{-1}, by following the strong constraint strategy. We set $N_\alpha = 1$, and we train an ANN with a single three-neurons hidden layer. In this manner, we obtain the data-driven model $\hat{g}$ and the slow-scale parameters $\left\{\hat{\alpha}^i\right\}^i$.

In order to a posteriori check the quality of the obtained results, in Figure 15 we show the learned parameters $\hat{\alpha}^i$, plotted against the ground-truth parameters of the original model α^i, used to the generate the data. We can see that, thanks to the proposed methodologies, we have successfully classified patients, by ranking them according to the disease severity.

Then, we consider the 75 synthetic patients generated in Section 3.4.1 but not included in the training set, in order to check the capability of the learned model to interpret unseen data and to make predictions. Hence, we apply the EKF algorithm in the time interval $\left[0, \mathcal{T}^\mathrm{obs}\right]$, with $\mathcal{T}^\mathrm{obs} = 25$ weeks, to estimate the value of the parameter $\hat{\alpha}$ for the testing patients. Then, we use the estimated parameter to predict the future evolution of the patient state in the time interval $\left(\mathcal{T}^\mathrm{obs}, \mathcal{T}\right]$. In this manner, we obtain an overall relative L^2 error between the prediction and the exact solutions equal to 1.06×10^{-2}.

The computations were run on a standard laptop (single core Intel i7-6500U). The execution of the EKF algorithm took nearly 140 ms for each run, for a total of nearly 7 s for each patient (as we have $N_S = 50$ checkpoints per patient). Therefore, preprocessing the 25 patients used to train the ANN required less than 3 min. The ANN training took instead 160 s. In conclusion, the whole offline phase took nearly 5 min. Concerning the online phase, for each patient the EKF algorithm needs to be run on the fast scale model up to $\mathcal{T}^\mathrm{obs}$, taking nearly 3.5 s; then the EKF algorithm is run on the slow scale model, which is then used to forecast the evolution of the patient. These steps only took 20 ms for each patient.

Finally, we notice that thanks to the application of the constraint of Example 1, the otherwise black-box parameter $\hat{\alpha}$ is endowed with a physical meaning. In particular, it owns a measure unit (arterial resistance per time unit) and it corresponds to the time rate of change of arterial resistance when its value is equal to R_d^*. This allows to provide a practical definition of the parameter: its numerical value (expressed as mmHg s mL^{-1} week^{-1}) can be estimated as the

increase of arterial resistance occurring in 1 week, when the patient has resistance equal to R_d^*. Thanks to the enhanced interpretability, this will facilitate the use of such data-driven models in clinical practice.

4 | CONCLUSIONS

In this paper we considered the problem of discovering differential equations describing the long-term evolution of phenomena featuring two temporal scales, starting from experimental measurements related to the fast-scale dynamics. We proposed to split the problem into two steps. The first step deals uniquely with the fast scale, and consists in estimating the fast-scale parameters by means of DA methods. Then, in the second step, we employ the reconstructed slow-scale evolution of these parameters to build a data-driven model for the dynamics occurring at this slow time scale.

With this aim, we proposed a ML algorithm that builds a data-driven model starting from time series data of a phenomenon featuring inter-individual variability. In practice, this algorithm discovers a differential equation, depending on one parameter or more, that approximates the time evolution of the individuals belonging to the training set. Moreover, this algorithm also learns the most likely value of such parameters for each of the training individual. We remark that this algorithm can be easily generalized to the case when the slow-scale phenomenon features a time-dependent input, similarly to what was done in Regazzoni et al.[26]

Then, we investigated the issue of interpretability of the learned model. In particular, we have highlighted that the problem of learning from time series a differential equation featuring one parameter or more never has a unique solution. Indeed, since the parameters are not directly measured, each model is associated with an equivalence class of models that can be obtained from the original one by an invertible transformation of the parameters into a new parameterization of the parameters themselves (we call such models *trivial reformulations* of the original one). Motivated by this, we have proposed a strategy to select a unique representative model inside each class of trivial reformulations of a given model. This is obtained by adding a constraint to the candidate class of models, either in weak or in strong form.

Moreover, we have shown that models can be split into two categories: minimal and non-minimal. Models belonging to the second class feature a disadvantage when one tries to build a data-driven model for the underlying phenomenon. In fact, non-minimal models admit, besides trivial reformulations, what we call *non-trivial reformulations*. Non-trivial reformulations are mathematical models, written as parametric differential equations, that can produce the same output of the original model, but that cannot be obtained from it simply by an invertible transformation of the parameters. For this reason, it is preferable to work with minimal models. Although minimality is an intrinsic property of a model, we recall that a non-minimal model can typically be reduced to a minimal one by removing variables that can be obtained as a combination of other variables. In other words, one should try not to learn a model with more variables than necessary; otherwise the dynamics would occur on trivial manifolds in the state space, leading to the existence of non-trivial reformulations. The development of criteria that allow to check the minimality of a model will be the subject of future works.

Once the training phase (offline phase) is accomplished, the learned model serves multiple purposes. First, it provides *understanding* of the phenomenon to be modeled. Secondly, by performing DA through the learned model, one can estimate the values of the parameters associated with an individual not included in the training set (online phase). In case the model is minimal, then the parameters obtained in this manner allow to *classify* the individual, as their values may be used to detect patients that have similar features. Thirdly, once an estimate of the parameters associated with an unseen individual is available, then the learned model can be used to *predict* the future evolution of such individual. We remark that, while classification is only possible when working with minimal models, the learned model can be employed to predict the future evolution of unseen individuals both when working with minimal and with non-minimal models.

Finally, we considered several test cases to show the effectiveness of the proposed methods and algorithms. Specifically, we synthetically generated time series by numerically approximating the solution of various differential models, including linear and nonlinear, first-order and second-order dynamics. Then, we employed our algorithm to learn parametric differential equations describing the measured data. We showed that, by suitably employing DA on the learned models in order to estimate the associated parameters, the future evolution of individuals not included in the training set can be accurately predicted. Moreover, numerical tests demonstrate that, when the noise added in the online and in the offline phases decreases, the prediction accuracy increases. Finally, we have shown that the strategy proposed to make the model learning problem well-posed succeeds in selecting a unique model among the equivalence class of

trivial reformulations of the original model, thus enhancing the interpretability of the learned model. Moreover, this strategy also provides a physical meaning to the parameters associated with the learned model, which would otherwise be a black-box object without any clear physical interpretation.

ORCID

Francesco Regazzoni https://orcid.org/0000-0002-4207-1400

ENDNOTE

[1] https://github.com/FrancescoRegazzoni/model-learning.

REFERENCES

1. Bensoussan A. *Estimation and Control of Dynamical Systems*. Springer; 2018:48.
2. Haykin S. *Kalman Filtering and Neural Networks*. John Wiley & Sons; 2004:47.
3. Yu W, Chen G, Cao J, Lü J, Parlitz U. Parameter identification of dynamical systems from time series. *Phys Rev E*. 2007;75(6):067201.
4. Xue H, Miao H, Wu H. Sieve estimation of constant and time-varying coefficients in nonlinear ordinary differential equation models by considering both numerical error and measurement error. *Ann Stat*. 2010;38(4):2351.
5. Ramsay J, Hooker G, Campbell D, Cao J. Parameter estimation for differential equations: a generalized smoothing approach series B statistical methodology; 2007.
6. Chapelle D, Fragu M, Mallet V, Moireau P. Fundamental principles of data assimilation underlying the Verdandi library: applications to biophysical model personalization within euHeart. *Med Biol Eng Comput*. 2013;51(11):1221-1233. https://doi.org/10.1007/s11517-012-0969-6.
7. Asch M, Bocquet M, Nodet M. *Data Assimilation: Methods, Algorithms, and Applications. Fundamentals of Algorithms*. SIAM; 2016.
8. Antoulas AC. *Approximation of Large-Scale Dynamical Systems*. SIAM; 2005:6.
9. Peherstorfer B, Willcox K. Dynamic data-driven reduced-order models. *Comput Methods Appl Mech Eng*. 2015;291:21-41.
10. Benner P, Mehrmann V, Sorensen DC. *Dimension Reduction of Large-Scale Systems*. Springer; 2005:35.
11. Benner P, Gugercin S, Willcox K. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev*. 2015;57(4):483-531.
12. Bongard J, Lipson H. Automated reverse engineering of nonlinear dynamical systems. *Proc Natl Acad Sci USA*. 2007;104(24):9943-9948.
13. Schmidt M, Lipson H. Distilling free-form natural laws from experimental data. *Science*. 2009;324(5923):81-85.
14. Schmidt MD, Vallabhajosyula RR, Jenkins JW, et al. Automated refinement and inference of analytical models for metabolic networks. *Phys Biol*. 2011;8(5):055011.
15. Peherstorfer B, Gugercin S, Willcox K. Data-driven reduced model construction with time-domain Loewner models. *SIAM J Sci Comput*. 2017;39(5):A2152-A2178.
16. Löwner K. Über monotone matrixfunktionen. *Mathematische Zeitschrift*. 1934;38(1):177-216.
17. Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc Natl Acad Sci USA*. 2016;113(15):3932-3937.
18. Costello Z, Martin HG. A machine learning approach to predict metabolic pathway dynamics from time-series multiomics data. *NPJ Syst Biol Appl*. 2018;4(1):19.
19. Wang W, Yang R, Lai Y, Kovanis V, Grebogi C. Predicting catastrophes in nonlinear dynamical systems by compressive sensing. *Phys Rev Lett*. 2011;106(15):154101.
20. Bocquet M, Brajard J, Carrassi A, Bertino L. Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models. *Nonlinear Processes Geophys*. 2019;26(3):143-162.
21. Daniels BC, Nemenman I. Efficient inference of parsimonious phenomenological models of cellular dynamics using systems and alternating regression. *PLoS One*. 2015;10(3):e0119821.
22. Hernandez AF, Gallivan MG. An exploratory study of discrete time state-space models using Kriging. In: IEEE; 2008:3993-3998.
23. Krige DG. A statistical approach to some basic mine valuation problems on the Witwatersrand. *J South Afr Inst Mining Metallurgy*. 1951;52(6):119-139.
24. Menafoglio A, Secchi P, Dalla RM. A universal Kriging predictor for spatially dependent functional data of a Hilbert space. *Electron J Stat*. 2013;7:2209-2240.
25. Rasmussen CE. *Gaussian Processes in Machine Learning*. Springer; 2004:63-71.
26. Regazzoni F, Dedè L, Quarteroni A. Machine learning for fast and reliable solution of time-dependent differential equations. *J Comput Phys*. 2019;397:108852.
27. Regazzoni F, Dedè L, Quarteroni A. Machine learning of multiscale active force generation models for the efficient simulation of cardiac electromechanics. *Comput Methods Appl Mech Eng*. 2020;370:113268.
28. Chen RT, Rubanova Y, Bettencourt J, Duvenaud D. Neural ordinary differential equations. arXiv preprint arXiv:1806.07366; 2018.
29. Raissi M, Perdikaris P, Karniadakis GE. Physics informed deep learning (Part I): data-driven solutions of nonlinear partial differential equations. arXiv preprint arXiv:1711.10561; 2017.

30. Raissi M, Perdikaris P, Karniadakis GE. Physics informed deep learning (Part II): data-driven discovery of nonlinear partial differential equations. arXiv preprint arXiv:1711.10566; 2017.
31. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys*. 2019;378:686-707.
32. Raissi M, Perdikaris P, Karniadakis GE. Machine learning of linear differential equations using Gaussian processes. *J Comput Phys*. 2017;348:683-693.
33. Raissi M, Karniadakis GE. Hidden physics models: machine learning of nonlinear partial differential equations. *J Comput Phys*. 2018; 357:125-141.
34. Raissi M, Perdikaris P, Karniadakis GE. Multistep neural networks for data-driven discovery of nonlinear dynamical systems. arXiv preprint arXiv:1801.01236; 2018.
35. Zhu Y, Zabaras N, Koutsourelakis P, Perdikaris P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *J Comput Phys*. 2019;394:56-81. https://doi.org/10.1016/j.jcp.2019.05.024.
36. Tartakovsky AM, Ortiz CM, Perdikaris P, Tartakovsky GD, Barajas-Solano D. Learning parameters and constitutive relationships with physics informed deep neural networks. arXiv preprint arXiv:1808.03398; 2018.
37. Westerhof N, Lankhaar JW, Westerhof BE. The arterial Windkessel. *Med Biol Eng Comput*. 2009;47(2):131-141.
38. Formaggia L, Gerbeau J, Nobile F, Quarteroni A. On the coupling of 3D and 1D Navier–Stokes equations for flow problems in compliant vessels. *Comput Methods Appl Mech Eng*. 2001;191(6–7):561-582.
39. Formaggia L, Quarteroni A, Veneziani A. *Cardiovascular Mathematics: Modeling and Simulation of the Circulatory System*. Springer Science & Business Media; 2010:1.
40. Friedman MH, Deters OJ, Mark FF, Bargeron CB, Hutchins GM. Arterial geometry affects hemodynamics: a potential risk factor for atherosclerosis. *Atherosclerosis*. 1983;46(2):225-231.
41. Nerem RM. Vascular fluid mechanics, the arterial wall, and atherosclerosis. *J Biomech Eng*. 1992;114(3):274-282. https://doi.org/10.1115/1.2891384.
42. Liang H, Miao H, Wu H. Estimation of constant and time-varying dynamic parameters of HIV infection in a nonlinear differential equation model. *Ann Appl Stat*. 2010;4(1):460.
43. Simon D. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience; 2006.
44. Pontryagin LS. *Mathematical Theory of Optimal Processes*. Routledge; 1962.
45. Kutta W. Beitrag zur naherungsweisen integration totaler differentialgleichungen. *Z Math Phys*. 1901;46:435-453.
46. Nocedal J, Wright S. *Numerical optimization*. 2nd ed. New York, NY: Springer Science & Business Media; 2006.
47. Yegnanarayana B. *Artificial Neural Networks*. PHI Learning Pvt. Ltd.; 2009.
48. Cybenko G. Approximation by superpositions of a sigmoidal function. *Math Control Sig Syst*. 1989;2(4):303-314.
49. Siegel JW, Xu J. On the approximation properties of neural networks. arXiv preprint arXiv:1904.02311; 2019.
50. He J, Li L, Xu J, Zheng C. Relu deep neural networks and linear finite elements. arXiv preprint arXiv:1807.03973; 2018.
51. Hermann R, Krener A. Nonlinear controllability and observability. *IEEE Trans Autom Control*. 1977;22(5):728-740.
52. Kou SR, Elliott DL, Tarn TJ. Observability of nonlinear systems. *Inform Control*. 1973;22(1):89-99.
53. McKay MD, Beckman RJ, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Dent Tech*. 2000;42(1):55-61.
54. Bavdekar VA, Deshpande AP, Patwardhan SC. Identification of process and measurement noise covariance for state and parameter estimation using extended Kalman filter. *J Proc Control*. 2011;21(4):585-601.
55. Gall AL, Vallée F, Pushparajah K, et al. Monitoring of cardiovascular physiology augmented by a patient-specific biomechanical model during general anesthesia. A proof of concept study. *PLoS One*. 2020;15(5):1-19.

How to cite this article: Regazzoni F, Chapelle D, Moireau P. Combining data assimilation and machine learning to build data-driven models for unknown long time dynamics—Applications in cardiovascular modeling. *Int J Numer Meth Biomed Engng*. 2021;e3471. https://doi.org/10.1002/cnm.3471

APPENDIX: A PROOFS OF THE PRESENTED RESULTS

We give in this appendix the Proofs of Propositions 1 and 2.

Proof of Proposition 1. Let us consider a generic $\theta_0 \in \mathcal{P}$ and $\alpha \in \mathcal{A}$. Let us denote $\theta(\tau) := s_g(\theta_0, \alpha, \tau)$. In virtue of Equation (23), we have

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\theta(\tau) = g(\theta(\tau),\alpha) = \tilde{g}\big(\theta(\tau),\tilde{A}(\theta_0,\alpha)\big), \quad \tau \geq 0. \tag{A1}$$

By the uniqueness of the solution of this ODE system, if follows that $s_{\tilde{g}}\big(\theta_0,\tilde{A}(\theta_0,\alpha),\tau\big) = \theta(\tau)$ for any $\tau \geq 0$.

Proof of Proposition 2. Let us define

$$\tilde{A}(\theta_0,\alpha) \coloneqq \Gamma\{g\}(\alpha), \quad \tilde{g}(\theta,\tilde{\alpha}) \coloneqq g\big(\theta,\Gamma\{g\}^{-1}(\tilde{\alpha})\big) \quad \forall \theta \in \mathcal{P}, \alpha \in \mathcal{A} \tag{A2}$$

and let $\tilde{\mathcal{A}}$ be the image of $\tilde{A}$. Clearly, the model $\left(\tilde{g},\mathcal{P},\tilde{\mathcal{A}}\right)$ is a trivial reformulation of $(g,\mathcal{P},\mathcal{A})$.

Let us suppose that $(g,\mathcal{P},\mathcal{A})$ admits another trivial reformulation, which we denote by $\left(\bar{g},\mathcal{P},\bar{\mathcal{A}}\right)$, satisfying $\Gamma\{\bar{g}\}(\bar{\alpha}) = \bar{\alpha}$ for any $\bar{\alpha} \in \bar{\mathcal{A}}$. Then, by Equation (23), there exists a function $B : \mathcal{A} \to \bar{\mathcal{A}}$ such that

$$g(\theta,\alpha) = \bar{g}(\theta,B(\alpha)) \quad \forall \theta \in \mathcal{P}, \alpha \in \mathcal{A}. \tag{A3}$$

By definition of identifiability, there exists a function h such that, for any $\theta_0 \in \mathcal{P}$ and $\alpha \in \mathcal{A}$

$$\alpha = h\Big(\big\{s_g(\theta_0,\alpha,\tau)\big\}_{\tau\geq 0}\Big) = h\Big(\big\{s_{\bar{g}}(\theta_0,B(\alpha),\tau)\big\}_{\tau\geq 0}\Big),$$

where the second equality follows from Proposition 1. The function B is thus invertible (in particular, we have $B^{-1}(\bar{\alpha}) = h\Big(\big\{s_{\bar{g}}(\theta_0,\bar{\alpha},\tau)\big\}_{\tau\geq 0}\Big)$).

By combining Equation (A2) with Equation (A3), we obtain

$$\tilde{g}(\theta,\tilde{\alpha}) = \bar{g}\big(\theta,B\big(\Gamma\{g\}^{-1}(\tilde{\alpha})\big)\big) \quad \forall \theta \in \mathcal{P}, \tilde{\alpha} \in \tilde{\mathcal{A}}. \tag{A4}$$

Therefore, we have $\tilde{\alpha} = \Gamma\{\tilde{g}\}(\tilde{\alpha}) = \Gamma\{\bar{g}\}\big(B\big(\Gamma\{g\}^{-1}(\tilde{\alpha})\big)\big) = B\big(\Gamma\{g\}^{-1}(\tilde{\alpha})\big)$, for each $\tilde{\alpha} \in \tilde{\mathcal{A}}$. By applying the function $B^{-1}(\cdot)$ at both sides, we get $B^{-1}(\tilde{\alpha}) = \Gamma\{g\}^{-1}(\tilde{\alpha})$, which entails $B(\cdot) \equiv \Gamma\{g\}(\cdot)$. Moreover, the sets $\bar{\mathcal{A}}$ and $\tilde{\mathcal{A}}$ coincide, being defined as the image of the maps $B(\cdot)$ and $\Gamma\{g\}(\cdot)$, respectively. Finally, by setting $B\big(\Gamma\{g\}^{-1}(\tilde{\alpha})\big) = \tilde{\alpha}$ in Equation (A4), we get $\tilde{g} \equiv \bar{g}$.