Computer Methods in Applied Mechanics and Engineering 418 (2024) 116499

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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

Branched Latent Neural Maps

Matteo Salvador [b,c,d,*], Alison Lesley Marsden [a,b,c,d]

[a] Department of Bioengineering, Stanford University, CA, USA
[b] Institute for Computational and Mathematical Engineering, Stanford University, CA, USA
[c] Cardiovascular Institute, Stanford University, CA, USA
[d] Pediatric Cardiology, Stanford University, CA, USA

ARTICLE INFO

Keywords:
Branched Latent Neural Maps
Scientific Machine Learning
Numerical simulations
Cardiac electrophysiology
Congenital heart disease

ABSTRACT

We introduce Branched Latent Neural Maps (BLNMs) to learn finite dimensional input–output maps encoding complex physical processes. A BLNM is defined by a simple and compact feed-forward partially-connected neural network that structurally disentangles inputs with different intrinsic roles, such as the time variable from model parameters of a differential equation, while transferring them into a generic field of interest. BLNMs leverage latent outputs to enhance the learned dynamics and break the curse of dimensionality by showing excellent in-distribution generalization properties with small training datasets and short training times on a single processor. Indeed, their in-distribution generalization error remains comparable regardless of the adopted discretization during the testing phase. Moreover, the partial connections, in place of a fully-connected structure, significantly reduce the number of tunable parameters. We show the capabilities of BLNMs in a challenging test case involving biophysically detailed electrophysiology simulations in a biventricular cardiac model of a pediatric patient with hypoplastic left heart syndrome. The model includes a 1D Purkinje network for fast conduction and a 3D heart-torso geometry. Specifically, we trained BLNMs on 150 in silico generated 12-lead electrocardiograms (ECGs) while spanning 7 model parameters, covering cell-scale, organ-level and electrical dyssynchrony. Although the 12-lead ECGs manifest very fast dynamics with sharp gradients, after automatic hyperparameter tuning the optimal BLNM, trained in less than 3 h on a single CPU, retains just 7 hidden layers and 19 neurons per layer. The resulting mean square error is on the order of 10^{-4} on an independent test dataset comprised of 50 additional electrophysiology simulations. In the online phase, the BLNM allows for 5000x faster real-time simulations of cardiac electrophysiology on a single core standard computer and can be employed to solve inverse problems via global optimization in a few seconds of computational time. This paper provides a novel computational tool to build reliable and efficient reduced-order models for digital twinning in engineering applications. The Julia implementation is publicly available under MIT License at https://github.com/StanfordCBCL/BLNM.jl.

1. Introduction

Learning complex input–output maps behind physical processes in a reliable manner has significant implications in any field of science and engineering. In particular, when these physical processes are described via mechanistic models, the numerical resolution of the underlying differential equations may be challenging and computationally demanding, even for a single instance of model parameters [1,2].

* Corresponding author.
E-mail address: msalvad@stanford.edu (M. Salvador).

https://doi.org/10.1016/j.cma.2023.116499
Received 10 August 2023; Received in revised form 27 September 2023; Accepted 28 September 2023
Available online 9 October 2023

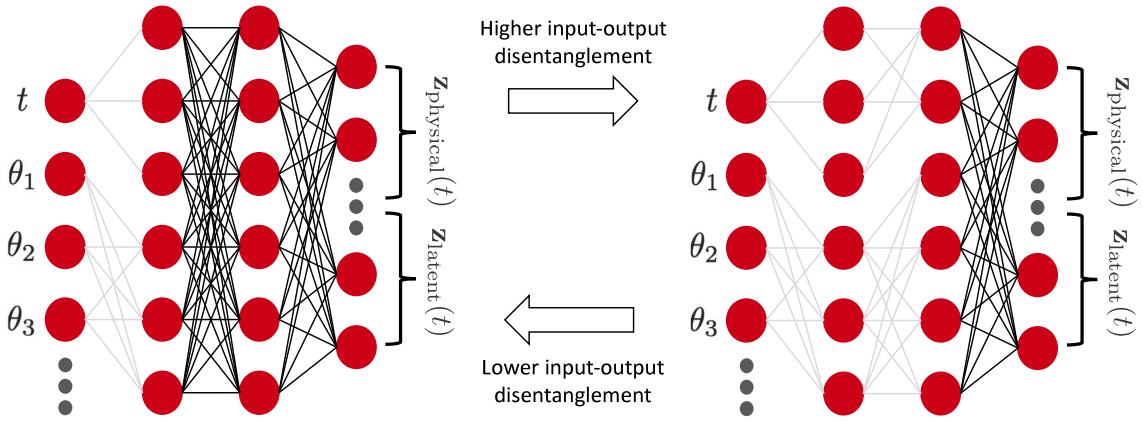

Fig. 1. Sketch of Branched Latent Neural Maps with different disentanglement levels between inputs, involving time variable t and model parameters θ, and outputs, i.e. generic fields of interest, including both physical $\mathbf{z}_{\text{physical}}(t)$ and latent $\mathbf{z}_{\text{latent}}(t)$ temporal quantities. Partial connections are depicted in light gray, whereas full connections are outlined in black.

In the past few years, several methods in the field of model order reduction, partially or entirely based on Neural Networks (NNs), have been proposed to mitigate the high computational cost of physics-based solvers, with the aim of producing accurate and efficient model evaluations for many-query applications [3–8], which involve sensitivity analysis, parameter estimation, forward and inverse uncertainty quantification, and optimization [9–11]. However, many intrusive [1] and non-intrusive [12] reduced-order models either fail or struggle to effectively reproduce phenomena that manifest fast and irregular dynamics while spanning an elaborate solution manifold. In this paper, we propose a novel computational tool which we term Branched Latent Neural Maps (BLNMs) to accurately and efficiently learn generic input–output relationships, even in the presence of sharp features and significant variability. BLNMs are based on feedforward partially-connected NNs [13] to separate the contributions coming from unrelated inputs, such as space and time variables with respect to physics-based scalar parameters. The output of BLNMs is given by relevant scalar or vector fields of interest, as well as additional latent variables, which serve the purpose of enhancing the learned dynamics. The presence of partial connections allows for a significant reduction in the number of tunable parameters while ensuring excellent in-distribution generalization properties during the testing phase, even on different mesh resolutions than those used during the training stage.

Several Machine Learning methods have been recently proposed to tackle cardiac electromechanics while exploiting physics-based knowledge [14–19]. In this paper, we demonstrate the performance of BLNMs in the setting of cardiovascular modeling [20–23] and congenital heart disease [24,25], where multiphysics and multiscale phenomena interact in the context of understudied pathological conditions in the field of computational cardiology. Specifically, we consider a patient-specific heart-torso geometry of a pediatric case with hypoplastic left heart syndrome (HLHS) [26]. We perform biventricular-Purkinje 3D-1D electrophysiology simulations to compute in-silico 12-lead electrocardiograms (ECGs) while spanning cell-scale through tissue-level parameter variability of a biophysically detailed mathematical model of electrophysiology. A BLNM trained on 150 electrophysiology simulations in less than 3 h on a single CPU, endowed with 7 hidden layers and 19 neurons per layer (2398 tunable parameters), retains an approximation error on the order of 10^{-4} on 50 additional unseen 12-lead ECGs by the NN. Moreover, it enables faster than real-time numerical simulations during the online phase, which allows one to accurately and efficiently solve inverse problems. Indeed, this task would be unaffordable using a biophysically detailed electrophysiology model, given the computational cost of these numerical simulations and the amount of queries that are required to solve a nonlinear optimization problem. BLNMs are lightweight, compact, easy to train architectures, able to precisely capture the fast time scales of 12-lead ECGs while spanning cell-to-organ model variability. Moreover, they can be queried in fractions of seconds to generate new predictions. Overall, BLNMs provide a novel computational tool for the generation of accurate and efficient standalone surrogate models that can be applied for digital twinning in computational science.

2. Methods

We describe the methodological details behind BLNMs for time-dependent processes, as well as the mathematical and numerical models adopted for the application of simulated cardiac electrophysiology in a congenital heart disease patient.

2.1. Branched latent neural Maps

Given a generic high-fidelity model $\mathcal{M}_{\text{HF}}$ expressed in terms of an input–output map between model parameters and a time-dependent process, we derive a surrogate model $\mathcal{M}_{\text{BLNM}}$ by building a feedforward partially-connected NN that explores model $\mathcal{M}_{\text{HF}}$ parametric variability while structurally separating the role of time and model parameters. We depict the BLNM architecture in Fig. 1, showing that different levels of disentanglement are allowed, ranging from the first hidden layer to the output layer. This

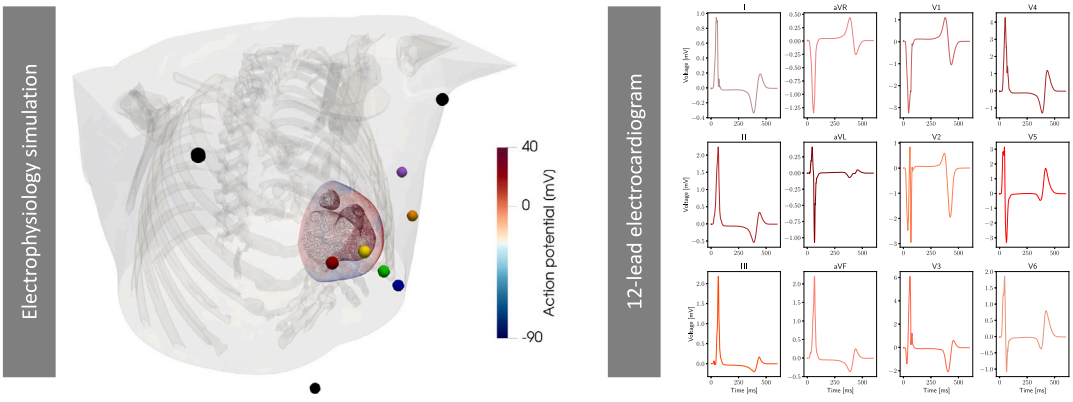

Fig. 2. Example of heart-torso electrophysiology simulation in the patient-specific cardiac model and corresponding 12-lead simulated ECGs.

disentanglement enables BLNMs to generalize well over different grids during testing even if the training stage is performed on a specific finite dimensional resolution (see Section 3.3). The surrogate model reads:

$$\mathbf{z}(t) = \mathcal{BLNM}(t, \boldsymbol{\theta}; \mathbf{w}) \quad \text{for } t \in [0, T]. \tag{1}$$

This feedforward partially-connected NN is represented by weights and biases $\mathbf{w} \in \mathbb{R}^{N_w}$, and defines a map $\mathcal{BLNM} : \mathbb{R}^{1+N_P} \to \mathbb{R}^{N_z}$ from time t and model parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^{N_P}$ to a state vector $\mathbf{z}(t) = [\mathbf{z}_{\text{physical}}(t), \mathbf{z}_{\text{latent}}(t)]^T$. Indeed, the state vector $\mathbf{z}(t) \in \mathbb{R}^{N_z}$ contains $\mathbf{z}_{\text{physical}}(t)$ physical fields of interest, as well as $\mathbf{z}_{\text{latent}}(t)$ latent temporal variables without a direct physical representation, that enhance the learned dynamics of the BLNM. These non-dimensional latent variables $\mathbf{z}_{\text{latent}}(t)$ are not accounted for in the loss function during the training stage, as in neural differential equations [6,27,28], but enrich the generalization of BLNMs while mapping the whole solution manifold, by selectively and properly acting in areas with steep gradients. During the optimization process of the NN tunable parameters, we minimize the Mean Square Error (MSE), that is:

$$\mathcal{L}(\widetilde{\mathbf{z}}_{\text{physical}}(t), \widetilde{\mathbf{z}}_{\text{obs}}(t); \widehat{\mathbf{w}}) = \arg\min_{\widehat{\mathbf{w}}} \left[\|\widetilde{\mathbf{z}}_{\text{physical}}(t) - \widetilde{\mathbf{z}}_{\text{obs}}(t)\|^2_{\mathrm{L}^2(0,T)} \right], \tag{2}$$

where $\widetilde{\mathbf{z}}_{\text{physical}}(t) \in [-1, 1]^{N_{z_{\text{physical}}}}$ and $\widetilde{\mathbf{z}}_{\text{obs}}(t) \in [-1, 1]^{N_{z_{\text{physical}}}}$ represent model $\mathcal{M}_{\text{BLNM}}$ outputs and observations in non-dimensional form. Time $\widetilde{t} \in [0, 1]$ and model parameters $\widetilde{\boldsymbol{\theta}} \in [-1, 1]^{N_P}$ are also normalized during the training phase of model $\mathcal{M}_{\text{BLNM}}$.

2.2. Cardiac electrophysiology

We reconstruct a heart-torso model of a 7 year old female pediatric patient with HLHS from computerized tomography (CT) images. Images and associated clinical data were obtained under an IRB-approved protocol at Stanford University. In Fig. 2 we show an example of an electrophysiology simulation and in silico derived 12-lead ECGs on this patient-specific geometry.

2.2.1. Mathematical model

We model cardiac electrophysiology in the heart-Purkinje system by considering the biophysically detailed monodomain equation [29,30] coupled with the ten Tusscher-Panfilov ionic model [31], represented here in compact form:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \mathcal{I}_{\text{ion}}(u, \boldsymbol{w}, \boldsymbol{z}) - \nabla \cdot (\boldsymbol{D}_{\text{M}} \nabla u) = \mathcal{I}_{\text{app}}(\mathbf{x}, t) & \text{in } \Omega \times (0, T], \\ (\boldsymbol{D}_{\text{M}} \nabla u) \cdot \mathbf{n} = 0 & \text{on } \partial\Omega \times (0, T], \\ \dfrac{d\boldsymbol{w}}{dt} = \boldsymbol{H}(u, \boldsymbol{w}, \boldsymbol{z}) & \text{in } \Omega \times (0, T], \\ \dfrac{d\boldsymbol{z}}{dt} = \boldsymbol{G}(u, \boldsymbol{w}, \boldsymbol{z}) & \text{in } \Omega \times (0, T], \\ u(\mathbf{x}, 0) = u_0(\mathbf{x}), \ \boldsymbol{w}(\mathbf{x}, 0) = \boldsymbol{w}_0(\mathbf{x}), \ \boldsymbol{z}(\mathbf{x}, 0) = \boldsymbol{z}_0(\mathbf{x}) & \text{in } \Omega. \end{cases} \tag{3}$$

In the following, we denote Eq. (3) as model $\mathcal{M}_{\text{HF}}$. $T = T_{\text{HB}} = 600$ ms corresponds to the final simulation time, given by a single heartbeat. The computational domain $\Omega = \Omega_{\text{purk}} \cup \Omega_{\text{myo}}$ is represented by the one-way coupled 1D Purkinje network and 3D biventricular patient-specific geometry.

Transmembrane potential u describes the propagation of the electric signal at the Purkinje and myocardial level, vector $\boldsymbol{w} = (w_1, \dots, w_M)$ defines the probability density functions of $M = 12$ gating variables, which represent the fraction of open channels across the membrane of a single cardiomyocyte, and vector $\boldsymbol{z} = (z_1, \dots, z_P)$ introduces the concentrations of $P = 6$ relevant ionic species. Among them, sodium Na^+, intracellular calcium Ca^{2+} and potassium K^+ play an important role in the physiological processes [32] dictating heart rhythmicity or sarcomere contractility, and are generally targeted by pharmaceutical therapies [33].

Table 1
Parameter space sampled via latin hypercube for the numerical simulations performed with model $\mathcal{M}_{\text{HF}}$.

Parameter	Description	Range	Units
G_{CaL}	Maximal Ca^{2+} current conductance	[1.99e−5, 7.96e−5]	cm ms^{-1} μF^{-1}
G_{Na}	Maximal Na^{+} current conductance	[7.42, 29.68]	nS pF^{-1}
G_{Kr}	Maximal rapid delayed rectifier current conductance	[0.08, 0.31]	nS pF^{-1}
D_{ani}	Anisotropic conductivity	[0.008298, 0.033192]	mm^2 ms^{-1}
D_{iso}	Isotropic conductivity	[0.002766, 0.011064]	mm^2 ms^{-1}
D_{purk}	Purkinje conductivity	[1.0, 3.5]	mm^2 ms^{-1}
$t_{\text{LV}}^{\text{stim}}$	Purkinje left bundle stimulation time	[0, 100]	ms

Right hand sides $\boldsymbol{H}(u, \boldsymbol{w}, \boldsymbol{z})$ and $\boldsymbol{G}(u, \boldsymbol{w}, \boldsymbol{z})$, which describe the dynamics of the gating variables and ionic concentrations respectively, along with ionic current $I_{\text{ion}}(u, \boldsymbol{w}, \boldsymbol{z})$, derive from the mathematical formulation of the ten Tusscher-Panfilov ionic model [31]. The action potential is triggered in the left and right bundle branches by an external applied current $I_{\text{app}}(\mathbf{x}, t)$.

The diffusion tensor is expressed as $\boldsymbol{D}_{\text{M}} = D_{\text{iso}}\mathbf{I} + D_{\text{ani}}\mathbf{f}_0 \otimes \mathbf{f}_0$ in Ω_{myo} and $\boldsymbol{D}_{\text{M}} = D_{\text{purk}}\mathbf{I}$ in Ω_{purk}, where $\mathbf{f}_0$ expresses the biventricular fiber field [34]. $D_{\text{ani}}, D_{\text{iso}}, D_{\text{purk}} \in \mathbb{R}^{+}$ represent the anisotropic, isotropic and Purkinje conductivities, respectively.

We impose the condition of an electrically isolated domain by prescribing homogeneous Neumann boundary conditions $\partial\Omega$, where $\mathbf{n}$ is the outward unit normal vector to the boundary.

The ECG signals u_{e} are computed in each lead location $\mathbf{x}_{\text{e}}$ following [35]:

$$u_{\text{e}}(\mathbf{x}_{\text{e}}) = -\int_{\Omega} \nabla u \cdot \nabla \frac{1}{\|\mathbf{x} - \mathbf{x}_{\text{e}}\|} dV, \tag{4}$$

where $e = \{V_1, V_2, V_3, V_4, V_5, V_6\}$ and $e = \{LA, RA, F\}$ define 6 precordial leads and 3 limb leads located on the pediatric patient-specific torso model, respectively. From this information, we retrieve 3 bipolar limb leads as:

$$I = LA - RA \quad II = F - RA \quad III = F - LA, \tag{5}$$

and 3 augmented limb leads as:

$$aVL = (I - III)/2 \quad aVR = -(I + II)/2 \quad aVF = (II + III)/2. \tag{6}$$

The set $ECG = \{V_1, V_2, V_3, V_4, V_5, V_6, I, II, III, aVL, aVR, aVF\}$ defines a 12-lead ECG, which is a comprehensive representation of the electrical activity in the heart [30].

In Table 1 we report descriptions, ranges and units for the 7 model parameters that we explore via latin hypercube sampling to generate the dataset of 200 electrophysiology simulations.

2.2.2. Numerical discretization

We perform space discretization of model $\mathcal{M}_{\text{HF}}$ using $\mathbb{P}_1$ Finite Elements. The biventricular tetrahedral mesh is comprised of 933,916 cells and 158,277 DOFs. The average mesh size is $h = 1$ mm. We generate the Purkinje network for both ventricles using the fractal tree and projection algorithm proposed in [36]. We initiate the left and right bundles from the endocardial locations near the atrioventricular node. The left bundle consists of 14,820 elements (14,821 DOFs), whereas the right bundle has 67,456 elements (67,457 DOFs). Following the approach adopted in [37], we use non-Gaussian quadrature rules to recover convergent conduction velocities in the cardiac tissue [38,39]. We consider a transmural variation of ionic conductances to differentiate epicardial, myocardial and endocardial properties according to [31]. For time discretization, we first update the variables of the ionic model and then the transmembrane potential by employing an Implicit-Explicit numerical scheme [20,40,41]. Specifically, in the monodomain equation, the diffusion term is treated implicitly and the ionic term is treated explicitly. Moreover, the ionic current is discretized by means of the Ionic Current Interpolation scheme [42]. We employ a fixed time step $\Delta t = 0.1$ ms. The fiber architecture is prescribed according to the Bayer–Blake–Plank–Trayanova algorithm with $\alpha_{\text{epi}} = -60°$, $\alpha_{\text{endo}} = 60°$, $\beta_{\text{epi}} = 20°$ and $\beta_{\text{endo}} = -20°$ [43].

2.2.3. Integration with branched latent neural Maps

In the present application, BLNMs are used to learn in silico ECGs while spanning relevant parameters of the monodomain equation and ten Tusscher-Panfilov ionic model. The vector $\boldsymbol{\theta}$ corresponds to the 7 model parameters $\boldsymbol{\theta}_{\text{EP}} = [G_{\text{CaL}}, G_{\text{Na}}, G_{\text{Kr}}, D_{\text{ani}}, D_{\text{iso}}, D_{\text{purk}}, t_{\text{LV}}^{\text{stim}}]^T$ reported in Table 1. The vector of physical variables $\mathbf{z}_{\text{physical}}(t)$ contains the $\mathbf{z}_{\text{leads}}(t)$ precordial and limb leads recordings, that is $\mathbf{z}_{\text{leads}}(t) = [V_1(t), V_2(t), V_3(t), V_4(t), V_5(t), V_6(t), LA(t), RA(t), F(t)]^T$. We note that these recordings are considered in their non-dimensional form $\widetilde{\mathbf{z}}_{\text{leads}}(t) \in [-1, 1]^{N_{z_{\text{physical}}}}$ during the training and testing phases. The same holds for time $\widetilde{t} \in [0, 1]$ and model parameters $\widetilde{\boldsymbol{\theta}}_{\text{EP}} \in [-1, 1]^{N_P}$.

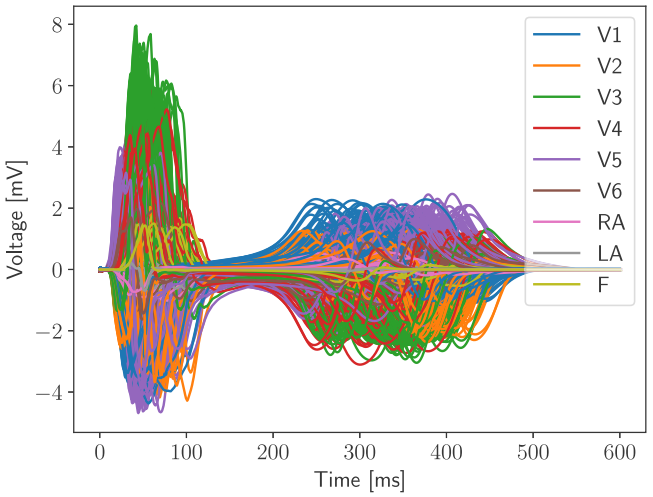

Fig. 3. Full dataset containing 200 in silico precordial and limb leads recordings.

2.3. Parameter estimation

We employ model $\mathcal{M}_{\text{BLNM}}$ in the setting of inverse problems. Specifically, we perform parameter calibration for $\widetilde{\theta}_{\text{EP}} \in [-1,1]^{N_P}$ to match BLNMs physical outputs $\widetilde{\mathbf{z}}_{\text{physical}}(t)$ to observations $\widetilde{\mathbf{z}}_{\text{obs}}(t)$ coming from model $\mathcal{M}_{\text{HF}}$ by minimizing the MSE, all in non-dimensional form, that is:

$$\mathcal{L}(\widetilde{\mathbf{z}}_{\text{physical}}(t), \widetilde{\mathbf{z}}_{\text{obs}}(t)) = \|\widetilde{\mathbf{z}}_{\text{physical}}(t) - \widetilde{\mathbf{z}}_{\text{obs}}(t)\|^2_{\text{L}^2(0,T)}. \tag{7}$$

We randomly initialize $\widetilde{\theta}^{\text{init}}_{EP}$ in the $[-1,1]^{N_P}$ hypercube and we aim to recover model $\mathcal{M}_{\text{HF}}$ parameters $\widetilde{\theta}^{\text{HF}}_{\text{EP}}$. We run a single trial of an Adaptive Differential Evolution algorithm for global optimization [44], which leads to a set of tuned model parameters $\widetilde{\theta}^{\text{DE}}_{\text{EP}}$ via BLNMs.

2.4. Software and hardware

We employ 3D slicer [45] for the manual segmentation of the medical images in order to reconstruct the heart-torso geometry. Meshing of this anatomic model is carried out using the TetGen library available in the SimVascular open-source software [46]. All electrophysiology simulations with model $\mathcal{M}_{\text{HF}}$ are performed using `svFSIplus` [47], a C++ high-performance computing multiphysics and multiscale finite element solver for cardiac and cardiovascular modeling, on 336 cores of the Stanford Research Computing Center. This solver is part of the SimVascular software suite for patient-specific cardiovascular modeling [46]. We train model $\mathcal{M}_{\text{BLNM}}$ by using `BLNM.jl` [48–50], a new, in-house, Julia library for Scientific Machine Learning which is made publicly available under MIT License at https://github.com/StanfordCBCL/BLNM.jl with this work. This public repository also contains the dataset encompassing all the electrophysiology simulations used for the training and testing phases.

3. Results

We report numerical results related to the electrophysiology simulations that were run to generate the training, validation and testing datasets for BLNMs. Then, we explain the technical details behind the automatic BLNM hyperparameter tuning method and show the properties and results associated with model $\mathcal{M}_{\text{BLNM}}$.

3.1. Electrophysiology simulations

We ran 200 numerical simulations on the patient-specific heart-torso model (see Fig. 2) and collected the corresponding simulated 12-lead ECGs. In Fig. 3 we depict the 200 precordial and limb lead sources that are employed for training, validation and testing of the BLNMs. In Fig. 4 we show the corresponding 12-lead ECGs, where the limb leads are algebraically manipulated according to Eqs. (5) and (6). In Fig. 5 we report a representative output from the 3D electrophysiology simulation, namely activation times for 8 random samples from the whole dataset.

We notice that by exploring relevant parameters affecting cardiac function at the cell-level and organ-scale, we are able to generate a broad set of plausible 12-lead ECGs and different patterns in the activation sequence for this pediatric patient. In particular, we remark that the simulated 12-lead ECGs produce sharp gradients during the QRS complex (ventricular depolarization) and T wave propagation (ventricular repolarization). Moreover, they manifest high variability among different instances of the model parameters.

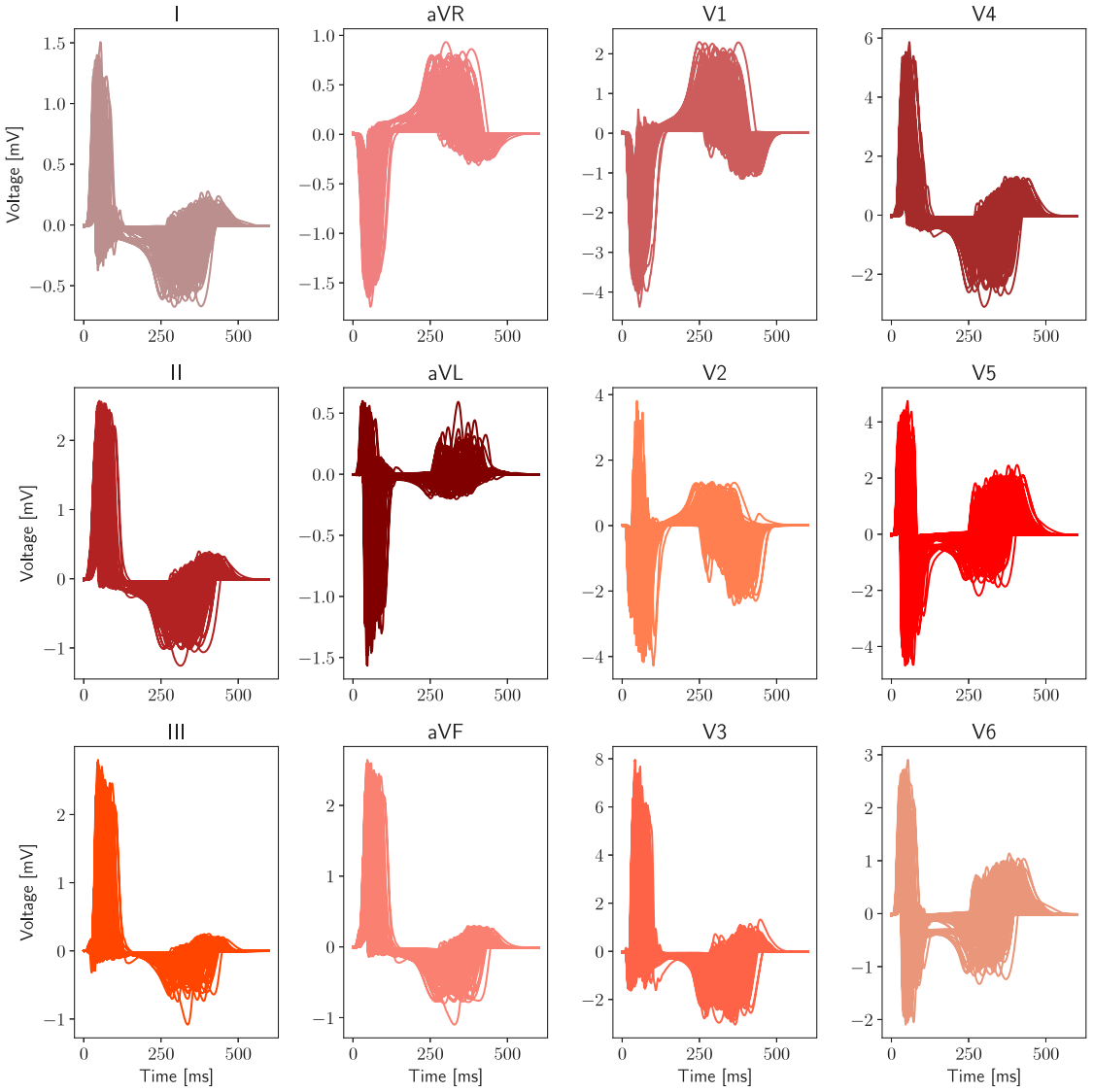

Fig. 4. Full dataset containing 200 simulated 12-lead ECGs.

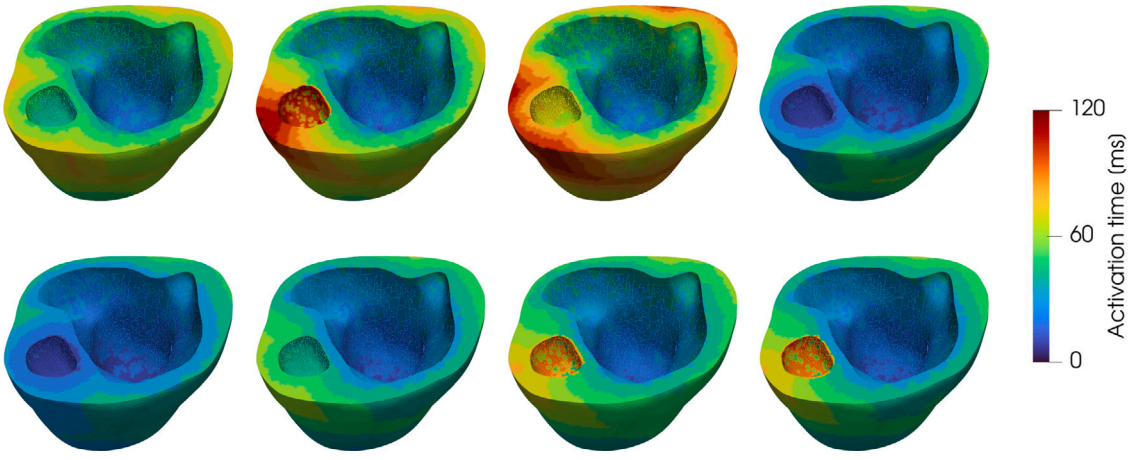

Fig. 5. Simulated activation times in 8 different electrophysiology simulations that are randomly extracted from the full dataset.

Table 2
Hyperparameters ranges and selected values for the final training stage.

BLNM	Hyperparameters				Trainable parameters
	Layers	Neurons	Number of states	Disentanglement level	# parameters
Tuning	{1 ... 8}	{10 ... 30}	{9 ... 12}	{1 ... N_{layers}}	
Final	7	19	10	2	2398

Table 3
Summary of the computational times and resources to generate the electrophysiology simulations with model $\mathcal{M}_{\text{HF}}$ and to train model $\mathcal{M}_{\text{BLNM}}$. We always tune NN parameters with the BFGS optimizer, by employing either 5 cores or serial execution on an Intel(R) Core(TM) i7-8700 3.20 GHz CPU. We sample in silico 12-lead ECGs with a fixed time step $\Delta t = 5.0$ ms.

Task	Computational resources	Execution time
Dataset generation using $\mathcal{M}_{\text{HF}}$ (200 simulations)	336 cores	1 day
$\mathcal{M}_{\text{BLNM}}$ hyperparameters tuning (50 confs, 10,000 iters)	5 cores	20 h
$\mathcal{M}_{\text{BLNM}}$ final training (50,000 iters)	1 core	2 h and 30 min

Table 4
MSE and computational times associated with different training dataset with increasing complexity for the optimal NN architecture (7 layers, 19 neurons per layer, 10 states, 2 disentanglement level). We use a fixed time step $\Delta t = 5.0$ ms. We employ 1 core of a standard computer endowed with an Intel(R) Core(TM) i7-8700 3.20 GHz CPU.

Number of simulations	Training loss (MSE)	Testing loss (MSE)	Training time
50	0.000599	0.018932	50 min
100	0.000293	0.000589	1 h and 45 min
150	0.000340	0.000454	2 h and 30 min

3.2. Hyperparameter tuning

We perform hyperparameter tuning by employing K-fold ($K = 5$) cross validation over 150 electrophysiology simulations. We consider a hypercube as a search space for the number of layers, number of neurons, number of states N_z and disentanglement level in the BLNM structure. Given the limited dimension of the search space, we employ 50 instances of latin hypercube sampling and select the configuration providing the lowest MSE. The Julia implementation is based on `Hyperopt.jl` [51], a package to perform parallel hyperparameter optimization. The different NNs associated with each K-fold are simultaneously trained via Message Passing Interface (MPI) on 5 physical cores of a standard workstation computer. We also exploit Hyper-Threading over 7 additional virtual cores with Open Multi-Processing (OpenMP) to speed-up computations. For each configuration of hyperparameters, we sample the dataset with a fixed time step of $\Delta t = 5.0$ ms and we perform 10,000 iterations of the second-order BFGS optimizer [52]. In Table 2 we report the initial hyperparameter ranges for tuning and the final optimized values. In Table 3 we detail the computational times and resources that we employ to generate electrophysiology simulations and to train NNs. Generating the dataset of biophysically detailed and anatomically accurate electrophysiology simulations and reaching the final BLNM configuration require less than 2 days of computation time. Each electrophysiology simulation runs in approximately 10 min but requires hundreds of cores to achieve this performance. On the other hand, training a single NN defining a BLNM requires 10 min to 3 h on a single CPU depending to the specific architecture.

3.3. Branched latent neural Maps

We showcase the features of BLNMs by means of different test cases. In Table 4 we analyze the influence of the training set size on the computational times and MSE. We consider the optimal NN architecture obtained from hyperparameters tuning (see Section 3.2). We notice that the total training time scales linearly with the dimensionality of the dataset. Moreover, the training costs on a single CPU are quite modest, approximately ranging from 1 to 3 h. The training MSE is small, on the order of 10^{-4}, and comparable, regardless of the number of electrophysiology simulations. On the other hand, the testing loss drops to $6 \cdot 10^{-4}$ with 100 numerical simulations. Given the sharp temporal dynamics of 12-lead ECGs, the number and ranges of model parameters covered by model $\mathcal{M}_{\text{HF}}$, BLNMs provide excellent in-distribution generalization properties with a relatively small amount of training data, especially when compared to the significant variability and complex dynamics encompassed by the dataset.

In Table 5 and Fig. 6 we study the effect of different testing time steps on the BLNM prediction accuracy. We see that the MSE remains approximately the same on finer and coarser meshes with respect to the fixed time step used for training, that is $\Delta t = 5.0$ ms. This means that BLNMs appear to show little sensitivity to time discretization even if the training stage is performed on a specific finite dimensional representation of the encoded physical process.

In Fig. 7 we compare the BLNM predictions with the ground truth for 5 randomly selected testing samples. BLNMs manifest good agreement with observations, even in presence of sharp peaks and gradients during the QRS complex and T wave propagation. In Table 6 we see the impact of varying the total number of states on the resulting MSEs. Adding latent outputs to the 9 physical outputs representing precordial and limb lead recordings allows us to significantly reduce both training and testing errors. Specifically,

Table 5
Testing errors associated with different sampling time steps on in silico 12-lead ECGs for the optimal NN architecture (7 layers, 19 neurons per layer, 10 states, 2 disentanglement level). We consider 50,000 BFGS iterations and 150 electrophysiology simulations for the training stage.

Training time step [ms]	Testing time step [ms]	Training loss (MSE)	Testing loss (MSE)
5.0	0.1	0.000348	0.000459
5.0	1.0	0.000348	0.000458
5.0	5.0	0.000340	0.000454
5.0	10.0	0.000337	0.000452
5.0	20.0	0.000333	0.000445

Table 6
Training and testing errors associated with different number of states on in silico 12-lead ECGs for the optimal NN architecture (7 layers, 19 neurons per layer, 2 disentanglement level). We consider 50,000 BFGS iterations and 150 electrophysiology simulations for the training stage, with $\Delta t = 5.0$ ms.

Number of states	Training loss (MSE)	Testing loss (MSE)
9	0.000758	0.097660
10	0.000340	0.000454
11	0.000358	0.000754

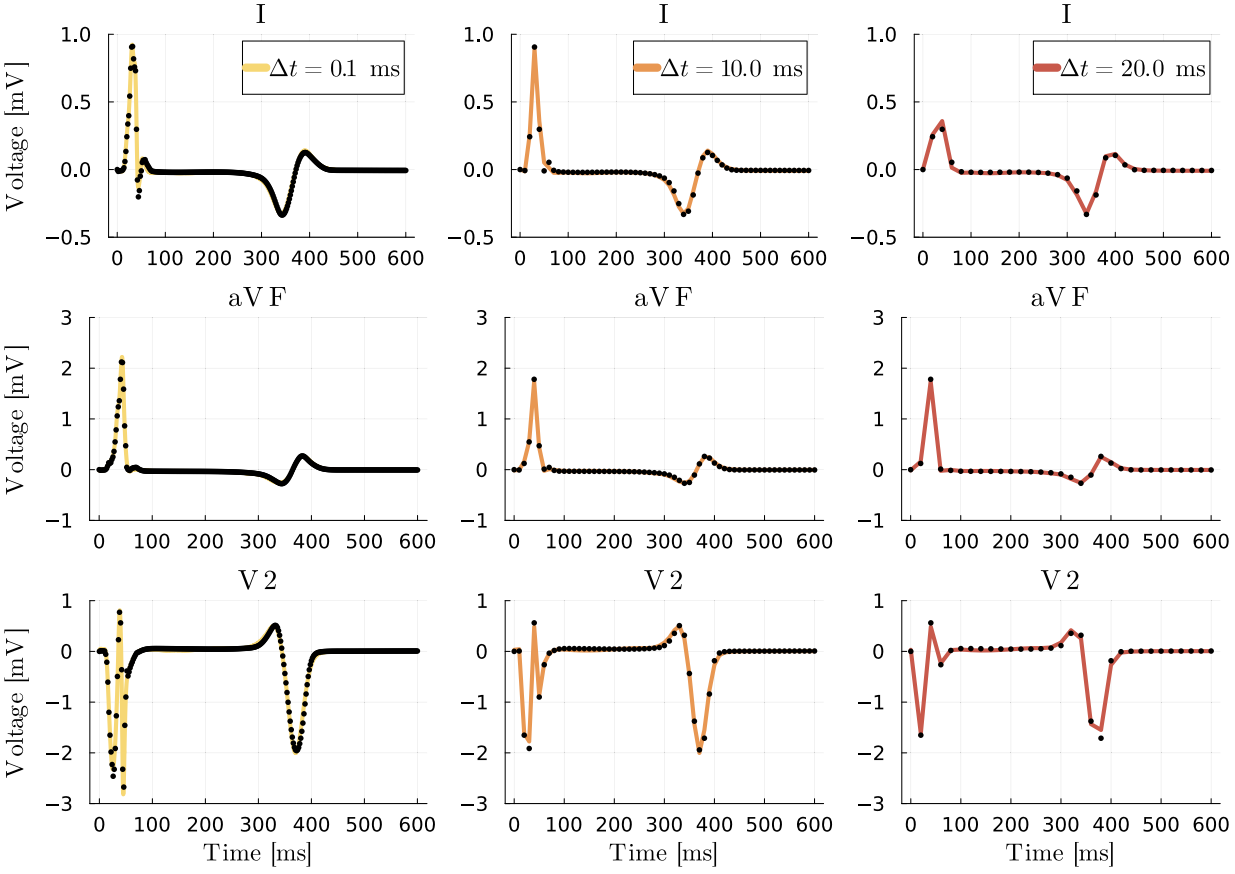

Fig. 6. BLNM predictions (solid) and ground truth (points) for 1 randomly selected 12-lead ECGs in the testing set. Different colors represent different testing time steps, namely 0.1, 10.0 and 20.0 ms (left to right), respectively. We show the time evolution of three relevant leads, i.e. I, aVF and V_2. We employ $\Delta t = 5.0$ ms for training. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

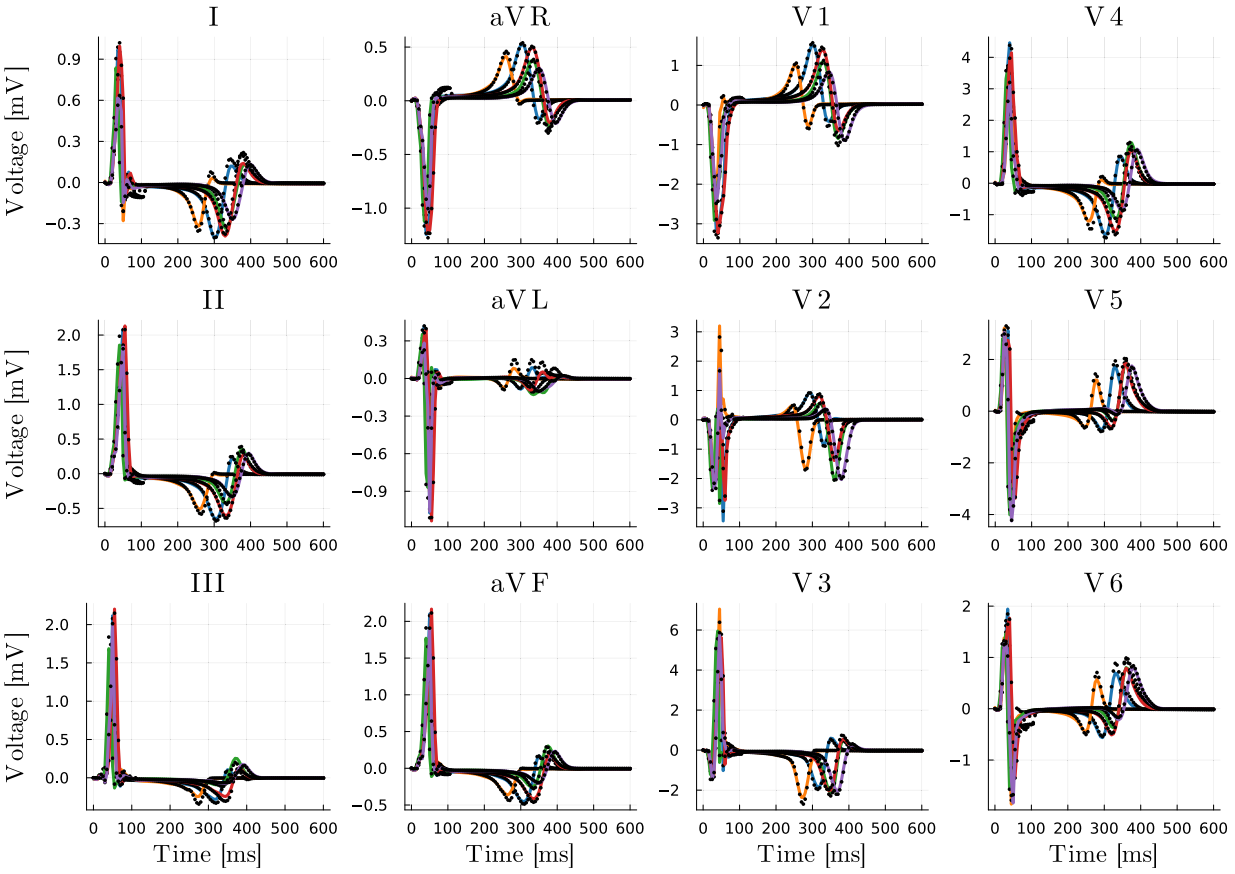

Fig. 7. BLNM predictions (solid) and ground truth (points) for 5 randomly selected 12-lead ECGs in the testing set.

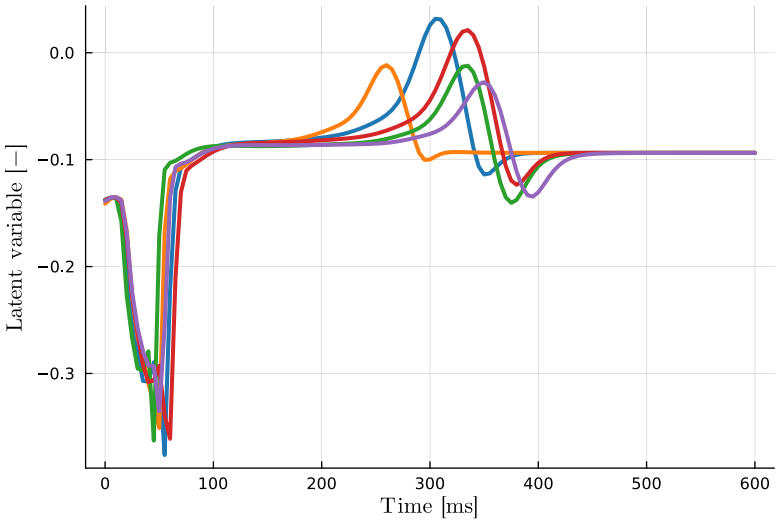

Fig. 8. Time evolution of BLNM latent variable for 5 randomly selected 12-lead ECGs in the testing set.

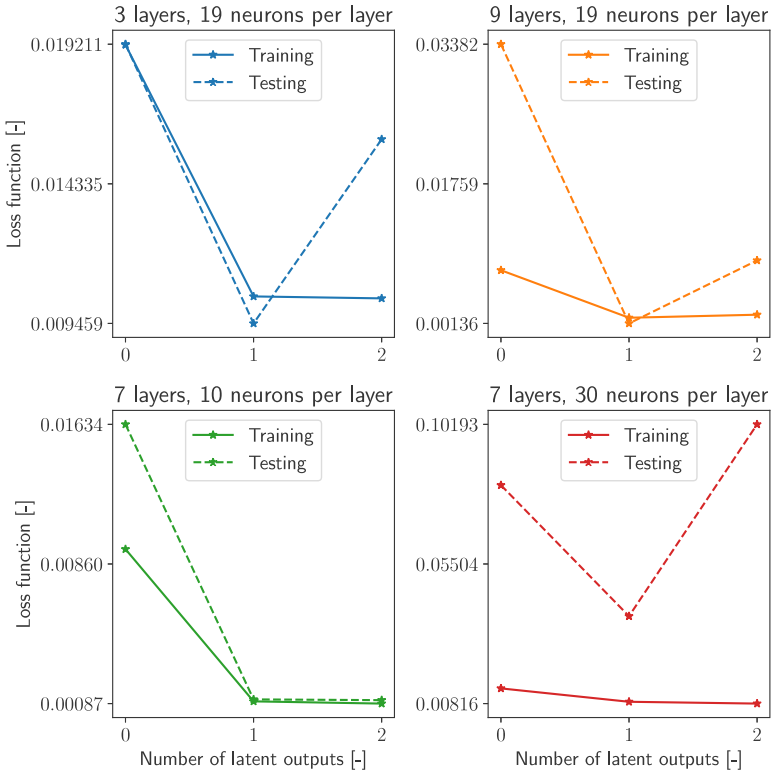

Fig. 9. Training and testing errors vs. number of latent outputs associated with four different NN architectures. We consider 50,000 BFGS iterations, 150 and 50 electrophysiology simulations for training and testing, respectively, with $\Delta t = 5.0$ ms.

the training error is approximately halved, whereas the testing error is reduced by two orders of magnitude. This means that the dynamics of 12-lead ECGs can be reproduced more accurately in the presence of a suitable number of latent variables. In particular, from Fig. 8 we notice that the additional latent variable selected by the hyperparameter tuning process enhances the BLNM learned dynamics by selectively acting on the QRS complex, that is ventricular depolarization, and T wave, that is ventricular repolarization. Similar considerations hold even for sub-optimal NN architectures. In Fig. 9 we depict the training and testing errors with respect to the number of latent outputs by considering four different BLNMs with smaller/higher number of layers and/or neurons per layer than the optimal set of hyperparameters. We see that adding one latent output always entails a significant reduction in both loss functions. On the other hand, two latent outputs contribute to a small reduction of the training error while sometimes leading to overfitting. This means that a single latent output is sufficient to capture the required additional features for this specific application.

We also train a standard feedforward fully-connected NN with 9 physical outputs, i.e. without latent outputs, 7 layers and 19 neurons per layer, that is the optimal configuration for BLNMs. This NN accounts for 2631 trainable parameters. We employ the BFGS optimizer and we perform 50,000 epochs over the usual 150 electrophysiology simulations, sampled with $\Delta t = 5.0$ ms. The training error is $7 \cdot 10^{-3}$ while the testing error is 3.1. This shows that BLNMs outperform standard NNs in terms of training and testing errors while considering less tunable parameters and shorter training times. Moreover, the standard NN does not generalize well on different discretization due to the high MSE reported for the testing loss.

Furthermore, we quantitatively compare BLNMs against latent neural differential equations [28,53]. We perform hyperparameter tuning using the same ranges reported in Table 2, except for the disentanglement level, which is not present given the feedforward fully-connected structure of the NN in this framework. Following the approach of BLNMs, we employ the BFGS optimizer and we perform 10,000 epochs, sampling the training and validation sets with a fixed time step $\Delta t = 5.0$ ms. We discretize the latent neural differential equations in time using the forward Euler method, by considering $\Delta t = 5.0$ ms. The optimal configuration of hyperparameters found during K-fold ($K = 5$), which is given by 7 layers, 26 neurons per layer and 9 states (i.e no latent variables), has a validation loss that is equal to 54.3. Indeed, we notice that latent neural differential equations fail to capture the QRS complex and the T wave, which are the most important features of 12-lead ECGs.

3.4. Parameter estimation

We employ the final BLNM to perform parameter calibration against the testing set, which is comprised of 50 electrophysiology simulations. In Fig. 10 we report the box plots showing the distribution of the errors, given by the absolute difference between each

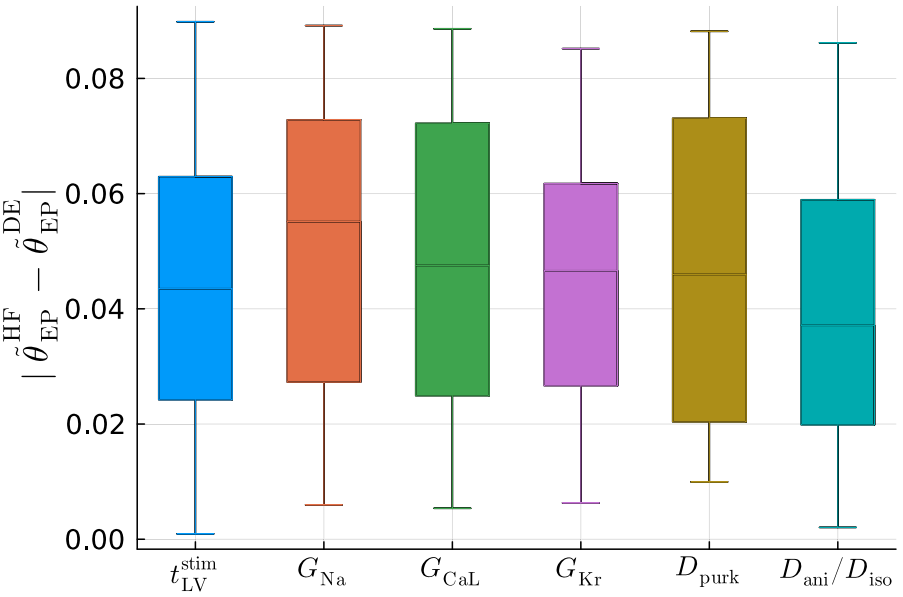

Fig. 10. Box plots showing the distribution of the errors for all model parameters.

parameter $\widetilde{\theta}^{\mathrm{HF}}_{\mathrm{EP}}$ from model $\mathcal{M}_{\mathrm{HF}}$ and each estimated parameter $\widetilde{\theta}^{\mathrm{DE}}_{\mathrm{EP}}$ with model $\mathcal{M}_{\mathrm{BLNM}}$, in non-dimensional form. We notice that all the errors are small and lay within the $[0, 0.09]$ range. This is possible given the small approximation error ($\sim 10^{-4}$) provided by the BLNM with respect to the high-fidelity electrophysiology simulations. We show that BLNMs can be used to match unseen observations coming from model $\mathcal{M}_{\mathrm{HF}}$, while also retrieving all 7 cell-to-organ model parameters. Performing a single instance of global optimization requires 7 s of computations in serial execution on an Intel(R) Core(TM) i7-8700 3.20 GHz CPU.

4. Discussion

Many efforts in the Scientific Machine Learning community are devoted to learning or mapping physical processes, within a certain range of variability, by means of NNs. This can be performed by either learning the time [27,28,53–55], space [56,57] and space–time [6–8,58,59] dynamics via different forms of neural differential equations or by mapping the whole solution manifold with physics-informed or data-driven neural maps [4,5,60,61]. These involve the use of feedforward fully-connected, recurrent, convolutional or graph neural networks, as well as encoders and decoders based on these architectures.

BLNMs blend and share mathematical properties coming from both classes of numerical methods. Indeed, this novel neural map encodes the whole output of interest by spanning model variability in a supervised fashion, while structurally disentangling inputs of different nature, such as time and model parameters of a differential equation. The level of separation between different categories of inputs can be properly tuned, ranging from the first hidden layer to the outputs of a feedforward partially-connected NN. BLNMs are simple, lightweight architectures, easy and fast to train, that effectively reproduce challenging processes with sharp gradients and fast dynamics in complex solution manifolds.

While autoencoders generally exploit latent variables between the encoder and the decoder in order to perform dimensionality reduction [62–64], BLNMs are endowed with additional latent outputs that act in specific regions of the simulated process to locally enhance the learned dynamics. This principle is similar to what is done in latent/augmented neural differential equations [6,27,28], where the NN defines a novel set of differential equations encoding the dynamics of a specific state vector, which contains both physical and latent variables. The latter are generally not considered in the loss function, as in BLNMs, but allow one to find better dynamics for the physical variables that are targeted during the optimization process. BLNMs exploit these latent variables as a lifting in the output dimension in order to better map the whole solution manifold directly, without passing them from a system of differential equations. This enables faster training than neural differential equations, as we do not have to replicate the NN structure over different time steps and we do not need to compute gradients over a chain of NNs during backpropagation. BLNMs require backpropagation over a single NN, where the presence of partial connections significantly reduces the number of tunable parameters with respect to latent neural differential equations. Similar considerations hold for the online inference process, which can be carried out with BLNMs by simply querying the NN without solving one or multiple differential equations. This provides a speed-up in the testing phase of BLNMs in comparison to neural differential equations. Moreover, latent neural differential equations generally struggle to reproduce sharp and irregular features, as we showed in Section 3.3.

On the other hand, while recent computational tools based on neural differential equations or deep neural operators enable space–time extrapolation [6,7,65,66], learning the input–output map via BLNMs currently allows for excellent in-distribution generalization only. Indeed, while testing BLNMs for out-of-distribution generalization, i.e. by considering model parameters outside the training

range and longer simulation times, we notice that they provide reasonable approximations only in the neighborhood that is right outside the training range and fail to perform time extrapolation. In particular, after the maximum training time, BLNMs provide the trivial zero solution followed by a divergent behavior. Future studies should aim to improve the performance of BLNMs for out-of-distribution generalization.

Another important feature of BLNMs is that they present a comparable performance among different discretizations during the testing phase. This last property is also shared by neural operators [60], which learn maps between infinite dimensional function spaces. Nevertheless, BLNMs focus on a specific finite dimensional grid during the training stage and are able to generalize over different resolutions during the testing phase. Moreover, if compared to BLNMs, neural operators of different categories, such as Fourier, low-rank, graph-based or deep operators, necessitate a more complex structure within the layers of the NN, which increases training and testing times [4,67].

BLNMs present several differences with respect to both physics-informed neural networks (PINNs) [61] and associated recent extensions [68–71]. While both BLNMs and PINNs share a data-driven term in the loss function, the former method encodes latent outputs that enhance the learned dynamics but does not enforce any physics-based knowledge, while the latter focuses on physical outputs only but also incorporates a physics-driven loss function based on the strong form of differential equations. BLNMs focus on a specific mesh during training and present similar generalization errors over both coarser and finer grids during testing. On the other hand, PINNs are mesh-less and require a suitable distribution of training points in the parameter space in order to generalize well during the testing phase. BLNMs present a partially connected structure, that allows for reduced complexity (i.e. number of trainable parameters) while structuring separating flows of information coming from inputs that are intrinsically different, whereas PINNs are normally based on fully-connected NNs. Both methods can potentially handle different sets of inputs and outputs, such as space and time variables, scalar and vector fields from parameterized differential equations, model-based or geometrically-based parameters. Furthermore, in this specific application for cardiac electrophysiology, the 12-lead ECGs are obtained by a space integral over the gradient of the transmembrane potential coming from the monodomain equation (see Eq. (4)), which makes a direct use of PINNs unfeasible because the physics-based part cannot be incorporated as the residual of a differential equation written in strong form. On the other hand, BLNMs can properly handle scenarios for model discovery or when the mathematical formulation cannot be seamlessly enforced in the loss function.

All the aforementioned aspects characterizing BLNMs are demonstrated on a challenging real-world application in the field of cardiac modeling. Specifically, a reduced-order model of in silico 12-lead ECGs spanning 7 cell-to-organ model parameters is learned from biophysically detailed and anatomically accurate electrophysiology simulations on a patient-specific heart-torso geometry of a pediatric patient with HLHS, a complex form of congenital heart disease. BLNMs accurately reproduce the outputs of this high-fidelity electrophysiology model and can be readily employed in many-query applications, such as robust and global parameter estimation.

5. Conclusions

We introduced BLNMs, a novel computational tool for arbitrary functional mapping. BLNMs structurally disentangles inputs with different intrinsic roles, such as time and model parameters, by means of feedforward partially-connected NNs. These partial connections can be propagated from the first hidden layer throughout the outputs according to the chosen disentanglement level. Furthermore, BLNMs may be endowed with latent variables in the output space, which enhance the learned dynamics of the neural map.

The novelties of this work reside both in the methods and their application to congenital heart disease, which is understudied in the field of computational cardiology. Indeed, we apply BLNMs in a challenging test case, that is learning the 12-lead ECGs of a pediatric patient with HLHS by covering a large range of 7 significant cell-to-organ model parameters. We demonstrate that BLNMs retain a small number of tunable parameters while accurately encoding complex, irregular and highly variable dynamics. Moreover, thanks to the efficient Julia implementation, leveraging different NN libraries and optimization tools, these neural maps can be trained in a fast manner even on a single CPU. BLNMs require small training datasets and do not degrade in accuracy when tested on a different discretization than the one used for training. Furthermore, they can be effectively employed for parameter estimation, as demonstrated using the whole testing set of the high-fidelity numerical simulations. This parameter calibration process can be carried out within a few seconds, i.e. almost in real-time, on a single core standard computer, by considering global optimization in the parameter space. In future works, we aim at using BLNMs to match patient-specific data with numerical simulations by also leveraging computational tools from global sensitivity analysis and robust parameter estimation with uncertainty quantification. Moreover, we would incorporate geometrical features within BLNMs, so that we can cover anatomical variability and we do not need to re-train the NN on every new patient.

Finally, although we showcased and tested BLNMs in a specific application involving time processes only, this paper paves the way for several extensions of the presented approach to space–time processes, while also structurally disentangling different sets of parameters, such as the ones describing geometric variability from scalar and vector values related to a single geometry. Furthermore, integrating a physics-based loss or a multifidelity approach, as recently proposed in the framework of deep operator networks [72], may improve the performance and generalization of BLNMs, especially for multiscale and multiphysics problems with known physical laws and properties.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my data/code in the paper.

Acknowledgments

This project has been funded by the NSF SSI, United States grant 1663671, CDSE, United States grant 2105345 and NIH, United States grants R01EB029362, R01LM013120. We acknowledge Additional Ventures Foundation, United States, Stanford Cardiovascular Institute, United States and the Vera Moulton Wall Center for pulmonary vascular disease at Stanford University, United States. We thank Dr. Fanwei Kong for the segmentation and mesh generation of the patient-specific heart-torso model.

References

[1] A. Quarteroni, A. Manzoni, F. Negri, Reduced Basis Methods for Partial Differential Equations. An Introduction, Vol. 92, Springer, 2016.
[2] A. Quarteroni, Numerical Models for Differential Problems, Springer, 2017.
[3] S. Fresca, L. Dede', A. Manzoni, A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs, J. Sci. Comput. 87 (2021).
[4] L. Lu, P. Jin, G. Pang, Z. Zhang, G.E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nat. Mach. Intell. 3 (2021) 218–229.
[5] B. Raonic, R. Molinaro, T. Rohner, S. Mishra, E. de Bezenac, Convolutional neural operators, in: ICLR Workshop on Physics for Machine Learning, 2023.
[6] F. Regazzoni, S. Pagani, M. Salvador, L. Dede', A. Quarteroni, Latent Dynamics Networks (LDNets): learning the intrinsic dynamics of spatio-temporal processes, 2023, arXiv:2305.00094.
[7] P.R. Vlachas, G. Arampatzis, C. Uhler, P. Koumoutsakos, Multiscale simulations of complex systems by learning their effective dynamics, Nat. Mach. Intell. 4 (2022) 359–366.
[8] L. Pegolotti, M.R. Pfaller, N.L. Rubio, K. Ding, R.B. Brufau, E. Darve, A.L. Marsden, Learning reduced-order models for cardiovascular simulations with graph neural networks, 2023, arXiv:2303.07310.
[9] C.M. Fleeter, G. Geraci, D.E. Schiavazzi, A.M. Kahn, A.L. Marsden, Multilevel and multifidelity uncertainty quantification for cardiovascular hemodynamics, Comput. Methods Appl. Mech. Engrg. 365 (2020) 113030.
[10] J.S. Tran, D.E. Schiavazzi, A.B. Ramachandra, A.M. Kahn, A.L. Marsden, Automated tuning for parameter identification and uncertainty quantification in multi-scale coronary simulations, Comput. & Fluids 142 (2017) 128—138.
[11] M. Salvador, F. Regazzoni, L. Dede', A. Quarteroni, Fast and robust parameter estimation with uncertainty quantification for the cardiac function, Comput. Methods Programs Biomed. 231 (2023) 107402.
[12] J.S. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, J. Comput. Phys. 363 (2018) 55–78.
[13] S. Kang, C. Isik, Partially connected feedforward neural networks structured by input types, IEEE Trans. Neural Netw. 16 (2005) 175–184.
[14] L. Cicci, S. Fresca, A. Manzoni, A. Quarteroni, Efficient approximation of cardiac mechanics through reduced order modeling with deep learning-based operator approximation, 2022, arXiv:2202.03904.
[15] S. Fresca, A. Manzoni, L. Dede', A. Quarteroni, Deep learning-based reduced order models in cardiac electrophysiology, PLOS ONE 15 (2020) 1–32.
[16] S. Pagani, A. Manzoni, Enabling forward uncertainty quantification and sensitivity analysis in cardiac electrophysiology by reduced order modeling and machine learning, Int. J. Numer. Methods Biomed. Eng. 37 (2021) e3450.
[17] R. Tenderini, S. Pagani, A. Quarteroni, S. Deparis, PDE-aware deep learning for inverse problems in cardiac electrophysiology, SIAM J. Sci. Comput. 44 (2022) B605–B639.
[18] F. Regazzoni, L. Dede', A. Quarteroni, Machine learning of multiscale active force generation models for the efficient simulation of cardiac electromechanics, Comput. Methods Appl. Mech. Engrg. 370 (2020) 113268.
[19] F. Regazzoni, M. Salvador, L. Dede', A. Quarteroni, A machine learning method for real-time numerical simulations of cardiac electromechanics, Comput. Methods Appl. Mech. Engrg. 393 (2022) 114825.
[20] M. Fedele, R. Piersanti, F. Regazzoni, M. Salvador, P.C. Africa, M. Bucelli, A. Zingaro, L. Dede', A. Quarteroni, A comprehensive and biophysically detailed computational model of the whole human heart electromechanics, Comput. Methods Appl. Mech. Engrg. 410 (2023) 115983.
[21] M. Strocchi, C.M. Augustin, M.A.F. Gsell, et al., A publicly available virtual cohort of four-chamber heart meshes for cardiac electro-mechanics simulations, PLOS ONE 15 (2020) 1–26.
[22] M. Peirlinck, F. Sahli Costabal, J. Yao, et al., Precision medicine in human heart modeling, Biomech. Model. Mechanobiol. 20 (2021) 803–831.
[23] M. Pfaller, J. Hörmann, M. Weigl, et al., The importance of the pericardium for cardiac biomechanics: from physiology to computational modeling, Biomech. Model. Mechanobiol. 18 (2019) 503–529.
[24] A.L. Marsden, J.A. Feinstein, Computational modeling and engineering in pediatric and congenital heart disease, Curr. Opin. Pediatr. 27 (2015) 587–596.
[25] I.E. Vignon-Clementel, A.L. Marsden, J.A. Feinstein, A primer on computational simulation in congenital heart disease for the clinician, Prog. Pediatr. Cardiol. 30 (1) (2010) 3–13.
[26] J.A. Feinstein, D.W. Benson, A.M. Dubin, et al., Hypoplastic left heart syndrome: Current considerations and expectations, J. Am. Coll. Cardiol. 59 (2012) S1–S42.
[27] E. Dupont, A. Doucet, Y.W. Teh, Augmented neural ODEs, 2019, arXiv:1904.01681.
[28] F. Regazzoni, L. Dede', A. Quarteroni, Machine learning for fast and reliable solution of time-dependent differential equations, J. Comput. Phys. 397 (2019) 108852.
[29] A. Quarteroni, L. Dede', A. Manzoni, C. Vergara, Mathematical Modelling of the Human Cardiovascular System: Data, Numerical Approximation, Clinical Applications, Cambridge University Press, 2019.
[30] P. Colli Franzone, L. Pavarino, S. Scacchi, Mathematical Cardiac Electrophysiology, Vol. 13, Springer, 2014.
[31] K.H. ten Tusscher, A.V. Panfilov, Alternans and spiral breakup in a human ventricular tissue model, Am. J. Physiol. Heart Circ. Physiol. 291 (2006) 1088–1100.
[32] D.C. Bartos, E. Grandi, C.M. Ripplinger, Ion channels in the heart, in: Comprehensive Physiology, 2015, pp. 1423–1464.
[33] T. Brennan, M. Fink, B. Rodriguez, Multiscale modelling of drug-induced effects on cardiac electrophysiological activity, Eur. J. Pharm. Sci. 36 (2009) 62–77.
[34] R. Piersanti, P.C. Africa, M. Fedele, et al., Modeling cardiac muscle fibers in ventricular and atrial electrophysiology simulations, Comput. Methods Appl. Mech. Engrg. 373 (2021) 113468.
[35] F. Sahli Costabal, J. Yao, E. Kuhl, Predicting drug-induced arrhythmias by multiscale modeling, Int. J. Numer. Methods Biomed. Eng. 34 (5) (2018) e2964.
[36] F. Sahli Costabal, D.E. Hurtado, E. Kuhl, Generating Purkinje networks in the human heart, J. Biomech. 49 (2016) 2455–2465.

[37] O.Z. Tikenogullari, M. Peirlinck, H. Chubb, A.M. Dubin, E. Kuhl, A.L. Marsden, Effects of cardiac growth on electrical dyssynchrony in the single ventricle patient, Comput. Methods Biomech. Biomed. Eng. (2023) 1–17.
[38] S. Pezzuto, J. Hake, J. Sundnes, Space-discretization error analysis and stabilization schemes for conduction velocity in cardiac electrophysiology, Int. J. Numer. Methods Biomed. Eng. 32 (2016) e02762.
[39] L.A. Woodworth, B. Cansiz, M. Kaliske, Balancing conduction velocity error in cardiac electrophysiology using a modified quadrature approach, Int. J. Numer. Methods Biomed. Eng. 38 (2022) e3589.
[40] F. Regazzoni, M. Salvador, P.C. Africa, M. Fedele, L. Dede', A. Quarteroni, A cardiac electromechanical model coupled with a lumped-parameter model for closed-loop blood circulation, J. Comput. Phys. 457 (2022) 111083.
[41] R. Piersanti, F. Regazzoni, M. Salvador, et al., 3D-0D closed-loop model for the simulation of cardiac biventricular electromechanics, Comput. Methods Appl. Mech. Engrg. 391 (2022) 114607.
[42] S. Krishnamoorthi, M. Sarkar, W.S. Klug, Numerical quadrature and operator splitting in finite element methods for cardiac electrophysiology, Int. J. Numer. Methods Biomed. Eng. 29 (2013) 1243–1266.
[43] J.D. Bayer, R.C. Blake, G. Plank, N. Trayanova, A novel rule-based algorithm for assigning myocardial fiber orientation to computational heart models, Ann. Biomed. Eng. 40 (2012) 2243–2254.
[44] J. Zhang, A.C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[45] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, et al., 3D Slicer as an image computing platform for the Quantitative Imaging Network, Magn. Reson. Imaging 30 (9) (2012) 1323–1341.
[46] A. Updegrove, N.M. Wilson, J. Merkow, H. Lan, A.L. Marsden, S.C. Shadden, SimVascular: An open source pipeline for cardiovascular simulation, Ann. Biomed. Eng. 45 (2017) 525–541.
[47] C. Zhu, V. Vedula, D. Parker, N. Wilson, S. Shadden, A. Marsden, svFSI: A multiphysics package for integrated cardiac modeling, J. Open Source Softw. 7 (2022) 4118.
[48] M. Innes, Flux: Elegant machine learning with Julia, J. Open Source Softw. (2018).
[49] C. Rackauckas, Q. Nie, Differentialequations.jl–a performant and feature-rich ecosystem for solving differential equations in julia, J. Open Res. Softw. 5 (1) (2017) 15.
[50] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, Universal differential equations for scientific machine learning, 2020, arXiv:2001.04385.
[51] F. Bagge Carlson, Hyperopt.jl: Hyperparameter optimization in Julia, 2018.
[52] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep Learning, Vol. 1, MIT press, Cambridge, 2016, 2.
[53] R.T.Q. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud, Neural ordinary differential equations, 2019, arXiv:1806.07366.
[54] Y. Rubanova, R.T.Q. Chen, D.K. Duvenaud, Latent ordinary differential equations for irregularly-sampled time series, in: Advances in Neural Information Processing Systems, Vol. 32, Curran Associates, Inc., 2019.
[55] P. Kidger, J. Morrill, J. Foster, T. Lyons, Neural controlled differential equations for irregular time series, Adv. Neural Inf. Process. Syst. (2020).
[56] F. Regazzoni, S. Pagani, A. Quarteroni, Universal Solution Manifold Networks (USM-Nets): Non-intrusive mesh-free surrogate models for problems in variable domains, J. Biomech. Eng. 144 (12) (2022) 121004.
[57] F. Pichi, B. Moya, J. Hesthaven, A graph convolutional autoencoder approach to model order reduction for parametrized PDEs, 2023, arXiv:2305.08573.
[58] R.M. Hasani, M. Lechner, A. Amini, D. Rus, R. Grosu, Liquid time-constant networks, in: AAAI Conference on Artificial Intelligence, 2020.
[59] I. Kičića, P.R. Vlachas, G. Arampatzis, M. Chatzimanolakis, L. Guibasc, P. Koumoutsakos, Adaptive learning of effective dynamics: Adaptive real-time, online modeling for complex systems, 2023, arXiv:2304.01732.
[60] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Learning maps between function spaces with applications to PDEs, J. Mach. Learn. Res. 24 (2023) 1–97.
[61] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[62] S. Fresca, A. Manzoni, POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, Comput. Methods Appl. Mech. Engrg. 388 (2022) 114181.
[63] F. Romor, G. Stabile, G. Rozza, Non-linear manifold reduced-order models with convolutional autoencoders and reduced over-collocation method, J. Sci. Comput. 94 (2023) 74.
[64] A. Solera-Rico, C.S. Vila, M.A. Gómez, Y. Wang, A. Almashjary, S.T.M. Dawson, R. Vinuesa, β-Variational autoencoders and transformers for reduced-order modelling of fluid flows, 2023, arXiv:2304.03571.
[65] F. Fatone, S. Fresca, A. Manzoni, Long-time prediction of nonlinear parametrized dynamical systems by deep learning-based reduced order models, 2022, arXiv:2201.10215.
[66] M. Zhu, H. Zhang, A. Jiao, G.E. Karniadakis, L. Lu, Reliable extrapolation of deep neural operators informed by physics or sparse observations, Comput. Methods Appl. Mech. Engrg. 412 (2023) 116064.
[67] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, 2021, arXiv:2010.08895.
[68] S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics–Informed neural networks: Where we are and what's next, J. Sci. Comput. 92 (3) (2022).
[69] S. Manikkan, B. Srinivasan, Transfer physics informed neural network: a new framework for distributed physics informed neural networks via parameter sharing, Eng. Comput. 39 (2023) 2961–2988.
[70] G. Pang, M. D'Elia, M. Parks, G.E. Karniadakis, nPINNs: Nonlocal physics-informed neural networks for a parametrized nonlocal universal Laplacian operator. Algorithms and applications, J. Comput. Phys. 422 (2020) 109760.
[71] M. Penwarden, S. Zhe, A. Narayan, R.M. Kirby, A metalearning approach for Physics-Informed Neural Networks (PINNs): Application to parameterized PDEs, J. Comput. Phys. 477 (2023) 111912.
[72] A.A. Howard, M. Perego, G.E. Karniadakis, P. Stinis, Multifidelity deep operator networks for data-driven and physics-informed problems, J. Comput. Phys. (2023) 112462.