NONLINEAR EMBEDDINGS FOR CONSERVING HAMILTONIANS AND OTHER QUANTITIES WITH NEURAL GALERKIN SCHEMES*

PAUL SCHWERDTNER†, PHILIPP SCHULZE‡, JULES BERMAN†, AND BENJAMIN PEHERSTORFER†

Abstract. This work focuses on the conservation of quantities such as Hamiltonians, mass, and momentum when solution fields of partial differential equations are approximated with nonlinear parametrizations such as deep networks. The proposed approach builds on Neural Galerkin schemes that are based on the Dirac–Frenkel variational principle to train nonlinear parametrizations sequentially in time. We first show that only adding constraints that aim to conserve quantities in continuous time can be insufficient because the nonlinear dependence on the parameters implies that even quantities that are linear in the solution fields become nonlinear in the parameters and thus are challenging to discretize in time. Instead, we propose Neural Galerkin schemes that compute at each time step an explicit embedding onto the manifold of nonlinearly parametrized solution fields to guarantee conservation of quantities. The embeddings can be combined with standard explicit and implicit time integration schemes. Numerical experiments demonstrate that the proposed approach conserves quantities up to machine precision.

Key words. model reduction, Hamiltonian systems, conservation of quantities, structure preservation, deep networks, Neural Galerkin schemes, Dirac–Frenkel variational principle

MSC codes. 65M22, 65P10, 68T07, 70H33

1. Introduction.

Preserving structure and conserving quantities such as Hamiltonians, mass, and momentum in numerical approximations of solution fields governed by time-dependent partial differential equations (PDEs) is important to guarantee physical consistency and help interpretation. This work focuses on conservation of quantities when discretizing solution fields with nonlinear parametrizations such as deep neural networks. The challenge is that the nonlinear dependence on the parameters means that quantities that are linear in the solution fields of the PDEs become nonlinear in the parameters, which leads to a loss of the linear vector-space structure that numerical methods traditionally build on. We show that only adding constraints in continuous-time formulations to keep quantities constant over time in the nonlinearly parametrized solution fields is insufficient. Instead, we propose to compute explicit embeddings onto the manifold of nonlinearly parametrized fields that conserve quantities. Numerical experiments demonstrate that the proposed approach conserves quantities up to machine precision.

Preserving structure and conserving quantities is a mainstay in computational science and engineering. There is a range of works that aim to learn models from data while preserving structure, such as structure-preserving dynamic mode decomposition [4, 47], operator inference [68, 52, 60, 56, 29], methods that learn from frequency-domain data [63, 64, 55, 65, 27, 28], and methods based on deep learning [17, 50, 70]. Another line of work aims to preserve structure in reduced models [54, 16, 12, 24, 1, 22, 36, 39, 34], specifically in the setting of computational fluid dynamics [6, 58, 15, 5, 57, 2, 9, 42, 46, 59]. Closest to our work is [14] that introduces a modified Galerkin system so that reduced solution fields conserve quantities. In all of these works, the

*Submitted to the editors 10/08/2023.

Funding: The authors Schwerdtner, Berman, Peherstorfer were partially supported by National Science Foundation under Grant No. 2046521 and the Office of Naval Research under award N00014-22-1-2728.

†Courant Institute of Mathematical Sciences (corresponding author: paul.schwerdtner@nyu.edu)

‡Technische Universität Berlin

parametrizations are linear.

We focus on nonlinear parametrizations such as neural networks [26], which provide one approach for efficiently constructing reduced models of transport-dominated problems; see [53] for a brief survey. Besides model reduction, nonlinear parametrizations can also help approximating high-dimensional problems [33]. Given that transport-dominated as well as high-dimensional problems are important in science and engineering applications, it is critical to develop numerical methods that can approximate solutions of PDEs with nonlinear parametrizations.

Nonlinear parametrizations can be trained with the Dirac–Frenkel variational principle [19, 23, 37, 43]; see [53] for an overview of nonlinear parametrizations in model reduction. Under assumptions on the parametrization and the equation of interest, structure is preserved in the special case when the parametrization is so rich that the residual and thus the error vanishes, which is leveraged in, e.g., [40]. The work [3] considers nonlinear parametrizations that are tuned towards the solution fields and proposes to add constraints that help conserve quantities in continuous time. Another line of work proposes dynamic low-rank approximations that are structure-preserving such as [21, 20, 51, 35, 48]. The work [13] introduces nonlinear parametrizations that penalize the deviation from symplecticity during training. The work [67] is a major step forward and considers quadratic manifolds to achieve an offline/online decoupling but currently still incurs high online costs because no empirical interpolation is considered.

The work closest to ours is [41], which considers deep autoencoders for parametrizing the latent states and adds constraints in the time-discrete formulation to conserve quantities; however, it requires solving a nonlinear and potentially non-convex optimization problem at each time step with the number of unknowns scaling with the latent state dimension and the number of conserved quantities. In fact, the costs of computing the optimization objective and gradients grow in the dimension of the ambient space rather than the dimension of the latent space. In contrast, our approach is applicable to explicit time integration schemes that lead to linear regression problems in each time step. We also have to solve a system of nonlinear equations at each time step but the number of unknowns grows with the number of conserved quantities only, which typically is orders of magnitude lower than the state dimension. Additionally, we leverage the results of the work [62] that achieves a preservation of the Hamiltonian via weighted schemes and specific nonlinear parametrizations, instead of explicitly computing embeddings. We show that the same specific nonlinear parametrizations lead to the preservation of the Hamiltonian in our setting too.

We build on the Neural Galerkin scheme introduced in [11], which is based on the Dirac–Frenkel variational principle [19, 23, 37, 43] and applies to generic nonlinear parametrizations such as deep networks. We first introduce Neural Galerkin schemes with constraints that conserve quantities in continuous time. We also show that Hamiltonians can be conserved in continuous time with weighted Neural Galerkin schemes and specific nonlinear parametrizations that are separable, even without adding constraints. While systems may conserve quantities in continuous time, the time discretization is delicate because the nonlinear parametrization means that the quantities depend nonlinearly on the parameters and thus no Runge-Kutta integration scheme can exist that conserves such quantities in general [31, Theorem IV.3.3]. Instead, we use nonlinear embeddings to find approximations that are close to the Neural Galerkin solutions and at the same time conserve quantities in discrete time. Importantly, the nonlinear embeddings can be combined with explicit time integration schemes that can be more efficient in the context of nonlinear parametrizations than

implicit schemes [11]. The nonlinear embeddings follow [31] and are numerically computed via an iterative scheme that applies generically to nonlinear parametrizations including deep networks but also to other nonlinear parametrizations used in, e.g., model reduction such as [62]. We stress that our approach conserves quantities but not necessarily structure such as symplecticity. With Burgers', acoustic wave, and shallow water equations, we demonstrate that the proposed scheme conserves quantities such as mass, energy, and Hamiltonians up to machine precision in numerical experiments.

2. Preliminaries. We describe the setup of time-dependent PDEs with conserved quantities and Hamiltonians, discuss Neural Galerkin schemes based on the Dirac–Frenkel variational principle, and provide a problem formulation.

2.1. Setup of time-dependent PDEs. Consider the PDE

$$\partial_t \boldsymbol{u}(t, \boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}(t, \cdot)) \qquad (t, \boldsymbol{x}) \in [0, \infty) \times \mathcal{X}\,, \tag{2.1}$$

$$\boldsymbol{u}(0, \boldsymbol{x}) = \boldsymbol{u}_0(\boldsymbol{x}) \qquad \boldsymbol{x} \in \mathcal{X}\,, \tag{2.2}$$

with the solution field $\boldsymbol{u} : [0, \infty) \times \mathcal{X} \to \mathbb{R}^m$ on the spatial domain $\mathcal{X} \subseteq \mathbb{R}^d$. At each time $t \in [0, \infty)$, the solution field $\boldsymbol{u}(t, \cdot) : \mathcal{X} \to \mathbb{R}^m$ is in a space $\mathcal{U}$ of functions that allow point-wise evaluations. In the following, the space $\mathcal{U}$ is a subspace of the space $\mathcal{L}^2(\mathcal{X})^m$ of square-integrable functions with m outputs with respect to a fully supported measure ν. The right-hand side function $\boldsymbol{f} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^m$ can include partial derivatives of $\boldsymbol{u}$ in the spatial variable $\boldsymbol{x}$. The initial condition is $\boldsymbol{u}_0 \in \mathcal{U}$. In the following, the boundary conditions for equation (2.1) are imposed by seeking solutions in the space $\mathcal{U}$ so that (2.1) is well posed.

2.2. Conserved quantities. Consider now quantities of the form

$$I_i : \mathcal{U} \to \mathbb{R} \quad \boldsymbol{u}(t, \cdot) \mapsto \int_{\mathcal{X}} \kappa_i(\boldsymbol{u}(t, \cdot))(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}), \tag{2.3}$$

where $\kappa_i : \mathcal{U} \to \mathcal{L}^1(\mathcal{X})$ is continuously differentiable for $i = 1, \ldots, n_I$. We categorize a quantity I_i as linear, quadratic or nonlinear depending on whether κ_i is a linear, quadratic or nonlinear function in $\boldsymbol{u}$, respectively. A quantity I_i is called a conserved quantity of the solution field $\boldsymbol{u}$ of equation (2.1) if it remains constant in the sense $I_i(\boldsymbol{u}(t, \cdot)) = I_i(\boldsymbol{u}(0, \cdot))$ for all $t \in [0, \infty)$ [31, Chapter IV]. The quantity I_i is also called first integral, invariant, or constant of motion. Note that I_i can be modified to account for inflow and outflow via a balance term; however, we will not pursue this further here as it would make the quantity depend on time. If the integrals in the conserved quantities (2.3) cannot be computed analytically, we numerically estimate them via Monte Carlo from n_M samples as

$$\hat{I}_i : \mathcal{U} \to \mathbb{R} \quad \boldsymbol{u}(t, \cdot) \mapsto \frac{1}{n_M} \sum_{s=1}^{n_M} \kappa_i(\boldsymbol{u}(t, \cdot))(\boldsymbol{\xi}_s)\,, \quad i = 1, \ldots, n_I\,, \tag{2.4}$$

where $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{n_M} \in \mathcal{X}$. The following methodology extends in a straightforward way to other quadrature schemes.

2.3. Hamiltonian systems. An important example of a conserved quantity arises when the right-hand side $\boldsymbol{f}$ of (2.1) can be written in the Hamiltonian form,

$$\boldsymbol{f}(\cdot, \boldsymbol{v}) = J(\boldsymbol{v}) \frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{v}) \tag{2.5}$$

for all $t \in [0, \infty)$ and $\boldsymbol{v} \in \mathcal{U}$, with the Hamiltonian $H \colon \mathcal{U} \to \mathbb{R}$ and the pointwise skew-adjoint interconnection operator $J \colon \mathcal{U} \to \mathrm{Hom}(\mathcal{U}, \mathcal{Z})$, where $\mathrm{Hom}(\mathcal{U}, \mathcal{Z})$ denotes the space of linear operators from $\mathcal{U}$ to $\mathcal{Z}$. If J satisfies in addition the Jacobi identity, we call it a Hamiltonian operator, see [49, Ch. 7] for more details. The space $\mathcal{Z}$ is a subspace of functions of $\mathcal{L}^2(\mathcal{X})^m$ that allow point-wise evaluations. Additionally, the variational derivative $\frac{\delta H}{\delta \boldsymbol{u}}$ attains values in $\mathcal{U}$. In the following, we consider Hamiltonians H that can be written as

$$H(\boldsymbol{v}) = \int_{\mathcal{X}} h(\boldsymbol{v})(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}) \,, \tag{2.6}$$

with continuously differentiable $h \colon \mathcal{U} \to \mathcal{L}^1(\mathcal{X})$. Analogous to the sampled conserved quantities (2.3), we also introduce the sampled Hamiltonian $\widehat{H}$. The interconnection operator J is pointwise skew-adjoint in the sense that

$$\langle \boldsymbol{q}_1, J(\boldsymbol{v})\boldsymbol{q}_2 \rangle_\nu = -\langle J(\boldsymbol{v})\boldsymbol{q}_1, \boldsymbol{q}_2 \rangle_\nu \tag{2.7}$$

holds for all $\boldsymbol{v}, \boldsymbol{q}_1, \boldsymbol{q}_2 \in \mathcal{U}$, where $\langle \cdot, \cdot \rangle_\nu$ denotes the $\mathcal{L}^2$ inner product corresponding to the measure ν. The pointwise skew-adjoint property implies in particular

$$\langle \boldsymbol{q}, J(\boldsymbol{v})\boldsymbol{q} \rangle_\nu = \frac{1}{2}(\langle \boldsymbol{q}, J(\boldsymbol{v})\boldsymbol{q} \rangle_\nu - \langle J(\boldsymbol{v})\boldsymbol{q}, \boldsymbol{q} \rangle_\nu) = 0 \tag{2.8}$$

for all $\boldsymbol{v}, \boldsymbol{q} \in \mathcal{U}$. The property (2.8) implies that the Hamiltonian is a conserved quantity, which follows from the computation

$$\begin{aligned} \frac{\mathrm{d}H(\boldsymbol{u}(t,\cdot))}{\mathrm{d}t}(t) &= \left\langle \frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{u}(t,\cdot)),\ \partial_t \boldsymbol{u}(t,\cdot) \right\rangle_\nu \\ &= \left\langle \frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{u}(t,\cdot)),\ J(\boldsymbol{u}(t,\cdot)) \frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{u}(t,\cdot)) \right\rangle_\nu = 0. \end{aligned}$$

One example of an equation that can be represented with the structure (2.5) is the inviscid Burgers' equation with periodic boundary conditions in one spatial dimension,

$$J(v)q = -\frac{1}{3}\left(\partial_x(vq) + v\partial_x q\right), \quad H(u) = \frac{1}{2}\|u\|^2_{\mathcal{L}^2(\mathcal{X})} \,, \tag{2.9}$$

which we will discuss in Section 6.2. Another example is the linear wave equation with periodic boundary conditions,

$$J = -\begin{bmatrix} 0 & \partial_x \\ \partial_x & 0 \end{bmatrix}, \quad H(\rho, v) = \frac{1}{2}\int_{\mathcal{X}} \frac{c^2}{\rho_{\mathrm{ref}}}\rho(\boldsymbol{x})^2 + \rho_{\mathrm{ref}} v(\boldsymbol{x})^2 \,\mathrm{d}\boldsymbol{x} \,, \tag{2.10}$$

which we will discuss in Section 6.3.

2.4. Neural Galerkin schemes based on the Dirac–Frenkel variational principle. Consider now a parametrization $\hat{\boldsymbol{u}} : \Theta \times \mathcal{X} \to \mathbb{R}^m$ of a solution field $\boldsymbol{u}$, which may depend nonlinearly on a time-dependent parameter vector $\boldsymbol{\theta} : [0, \infty) \to \Theta \subseteq \mathbb{R}^{n_{\boldsymbol{\theta}}}$ of dimension $n_{\boldsymbol{\theta}}$. For example, the function $\hat{\boldsymbol{u}}$ can be a deep network, where the components of the parameter $\boldsymbol{\theta}(t) \in \Theta$ correspond to the weights and biases. We only consider parametrizations that are continuously differentiable in the parameter. Plugging $\hat{\boldsymbol{u}}$ into (2.1) leads to the residual function

$$\boldsymbol{r}_t(\boldsymbol{\theta}(t), \dot{\boldsymbol{\theta}}(t), \boldsymbol{x}) = \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x})^{\mathsf{T}} \dot{\boldsymbol{\theta}}(t) - \boldsymbol{f}(\boldsymbol{x}, \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)) \,. \tag{2.11}$$

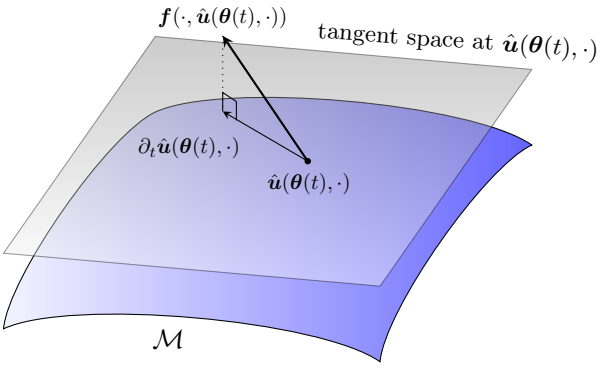


FIG. 1. *Neural Galerkin schemes are based on the Dirac–Frenkel variational principle [19, 23], which defines the time derivative $\dot{\boldsymbol{\theta}}(t)$ of the parameter $\boldsymbol{\theta}(t)$ to be the orthogonal projection of the right-hand side function $\boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))$ onto the tangent space of the manifold $\mathcal{M}$ at the current solution field $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$. The tangent space is spanned by the component functions of the gradient $\nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$.*

The time derivative $\dot{\boldsymbol{\theta}}(t)$ is then determined by following the Dirac–Frenkel variational principle [19, 23, 37, 43] such that

$$\left\langle \partial_{\theta_i} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\ \cdot),\ \boldsymbol{r}_t(\boldsymbol{\theta}(t), \dot{\boldsymbol{\theta}}(t), \cdot) \right\rangle_{\nu} = 0\,, \qquad i = 1, \ldots, n_{\boldsymbol{\theta}}\,, \tag{2.12}$$

where $\partial_{\theta_i} \hat{\boldsymbol{u}}$ is the i-th component function of the gradient $\nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}$ of $\hat{\boldsymbol{u}}$ with respect to the parameter $\boldsymbol{\theta}$. A note on the history of the Dirac–Frenkel variational principle can be found in [40, Section 3.8]. The solution $\dot{\boldsymbol{\theta}}(t)$ can be interpreted as determining an orthogonal projection $\nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)^{\mathsf{T}} \dot{\boldsymbol{\theta}}(t)$ of the right-hand side function $\boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))$ evaluated at $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ onto the tangent space $\mathcal{T}_{\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)} \mathcal{M}$ at the point $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ of the parametrization manifold

$$\mathcal{M} = \{ \hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) \,|\, \boldsymbol{\eta} \in \Theta \subseteq \mathbb{R}^{n_{\boldsymbol{\theta}}} \} \subseteq \mathcal{U}\,, \tag{2.13}$$

which is illustrated in Figure 1. Note that we follow standard terminology in the context of the the Dirac-Frenkel variational principle [43] and use the term manifold for the set $\mathcal{M}$. With tangent space, we refer to the space spanned by the partial derivatives $\partial_{\theta_1} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot), \ldots, \partial_{\theta_{n_{\boldsymbol{\theta}}}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ with respect to the components of the parameter $\boldsymbol{\theta}(t)$.

Conditions (2.12) can be rewritten in matrix form as

$$\boldsymbol{M}(\boldsymbol{\theta}(t)) \dot{\boldsymbol{\theta}}(t) = \boldsymbol{F}(\boldsymbol{\theta}(t)), \tag{2.14}$$

with the matrix $\boldsymbol{M}(\boldsymbol{\theta}(t))$ and vector $\boldsymbol{F}(\boldsymbol{\theta}(t))$ having the following components

$$\boldsymbol{M}_{ij}(\boldsymbol{\theta}(t)) = \left\langle \partial_{\theta_i} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot),\ \partial_{\theta_j} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot) \right\rangle_{\nu}, \quad i, j = 1, \ldots, n_{\boldsymbol{\theta}}\,, \tag{2.15a}$$

$$\boldsymbol{F}_i(\boldsymbol{\theta}(t)) = \left\langle \partial_{\theta_i} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot),\ \boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)) \right\rangle_{\nu}, \quad i = 1, \ldots, n_{\boldsymbol{\theta}}\,. \tag{2.15b}$$

Following [11], we refer to (2.14) as the Neural Galerkin system because the parametrizations that we use in the following are all based on neural networks and (2.12) can be interpreted as a Galerkin projection with the component functions of $\nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}$ as test functions. If the matrix elements (2.15) are not analytically available, then they can be numerically estimated via either quadrature or Monte Carlo methods. We denote the sample-based estimates of $\boldsymbol{M}(\boldsymbol{\theta}(t))$ and vector $\boldsymbol{F}(\boldsymbol{\theta}(t))$ at the sampling points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_S} \in \mathcal{X}$ as

$$\hat{\boldsymbol{M}}(\boldsymbol{\theta}(t)) = \frac{1}{n_S} \sum_{i=1}^{n_S} \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_i) \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_i)^{\mathsf{T}}, \tag{2.16}$$

$$\hat{\boldsymbol{F}}(\boldsymbol{\theta}(t)) = \frac{1}{n_S}\sum_{i=1}^{n_S} \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_i) \boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))(\boldsymbol{x}_i)\,, \tag{2.17}$$

which give rise to the sampled Neural Galerkin system

$$\hat{\boldsymbol{M}}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{F}}(\boldsymbol{\theta}(t))\,. \tag{2.18}$$

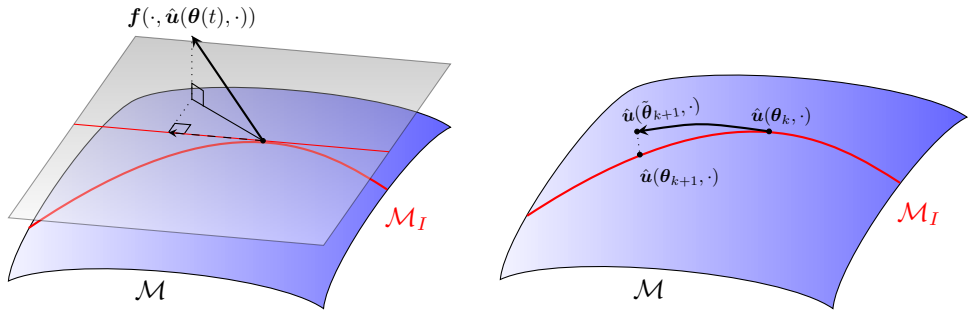
There are several approaches for obtaining efficient Monte Carlo estimates [11, 69], which goes beyond the scope of the present work. In the following, we assume that the system of ordinary differential equations (ODEs) given by the Neural Galerkin system (2.14) and the corresponding sampled system (2.18) have a solution on the whole time interval $[0, \infty)$.

2.5. Problem formulation. Conserving quantities in solutions obtained with Neural Galerkin schemes based on nonlinear parametrizations leads to two challenges. First, a quantity I_i is not necessarily conserved by a Neural Galerkin solution $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ with parameter $\boldsymbol{\theta}(t)$ satisfying (2.14), even in continuous time and without sampling. The reason is that the Galerkin projection formulated in (2.12) only seeks to set the residual orthogonal to a tangent space of $\mathcal{M}$ but ignores any additional constraints given by the quantities $I_1, \ldots, I_{n_I}$.

Second, only adding constraints to the continuous-time Neural Galerkin equations (2.14) or their sampled counterparts (2.18) is insufficient to conserve quantities because of the nonlinear parametrization of the solution field. While linear quantities can be conserved by implicit and explicit Runge-Kutta integrators [66] and quadratic quantities can be conserved by Runge-Kutta integrators if the coefficients satisfy conditions described in [18], arbitrary higher-order polynomial or nonlinear quantities are not conserved by any Runge-Kutta integrators in general; see also the discussion in [31, Chapter IV]. This is important in the context of Neural Galerkin schemes and related methods because the parameter $\boldsymbol{\theta}(t)$ can enter nonlinearly in the parametrization $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$. This means that conserved quantities that are linear in the solution field $\boldsymbol{u}$ of the PDE formulation (2.1) (e.g., the integral in Burgers' equation) depend through the parametrization $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ nonlinearly on the parameter $\boldsymbol{\theta}(t)$, which is the state of the Neural Galerkin system (2.14) and (2.18). Thus, even if the continuous-time Neural Galerkin projection is constrained so that $\dot{\boldsymbol{\theta}}(t)$ conserves the quantities $I_1, \ldots, I_{n_I}$, the conservation is lost after discretizing in time with common integrators such as Runge-Kutta integrators because they typically only conserve linear quantities. In fact, there is no Runge-Kutta method that can conserve all polynomial quantities of degree greater than two [31, Theorem IV.3.3].

3. Conserving quantities in continuous-time Neural Galerkin schemes. We pursue two options for conserving quantities in continuous time. First, in Section 3.1, we conserve quantities via constraints by building on previous work and introducing constrained Neural Galerkin schemes that conserve quantities in continuous time for generic nonlinear parametrizations. Second, in Section 3.2, we enforce conservation via structure in the nonlinear parametrization. In particular, we construct nonlinear parametrizations so that weighted Neural Galerkin schemes preserve Hamiltonians without the need of explicitly adding constraints.

3.1. Adding constraints to Neural Galerkin schemes for conserving quantities in continuous time. The Neural Galerkin scheme is based on projecting the right-hand side onto tangent spaces of the parametrization manifold; see Section 2.4. If the image of the right-hand side function is a subset of the tangent



(a) adding constraints to Neural Galerkin schemes (b) Neural Galerkin with embeddings

FIG. 2. *Only adding constraints to the continuous-time Neural Galerkin system is insufficient to conserve quantities because the constraint can be violated in discrete time due to the nonlinear parametrization of the solution field. In contrast, the proposed Neural Galerkin scheme combines constraints with nonlinear projections to obtain embeddings onto manifolds of functions that conserve quantities in discrete time.*

space, then the residual is zero and thus all quantities that are conserved by the PDE solution field $\boldsymbol{u}$ are also conserved by the continuous-time Neural Galerkin solution $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$. This relation between residual and conserving quantities has been, for instance, used in the context of the Schrödinger equation in [40, Sec. 3] to establish energy and norm conservation when using a Gaussian wave packet for the parametrization. We now consider more general cases where the residual is not necessarily zero and add constraints to the time-continuous Neural Galerkin system (2.14) and its sampled counterpart (2.18) to conserve quantities in continuous time; similar to other methods based on nonlinear parametrizations [41, 3].

3.1.1. Constrained manifolds. Recall the interpretation that $\dot{\boldsymbol{\theta}}(t)$ corresponds to an orthogonal projection of the right-hand side function $\boldsymbol{f}$ onto a tangent space of $\mathcal{M}$. By adding a constraint to the Neural Galerkin system, we restrict the manifold $\mathcal{M}$ of parametrized functions $\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)$ defined in (2.13) to the manifold $\mathcal{M}_I$ of functions that conserve quantities

$$I_i(\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)) = c_i\,, \qquad i = 1, \ldots, n_I\,, \tag{3.1}$$

for a given vector of constants $\boldsymbol{c} = [c_1, \ldots, c_{n_I}]$. Thus, we obtain $\mathcal{M}_I$ as

$$\mathcal{M}_I = \{\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) : \boldsymbol{\eta} \in \Theta \text{ and } I_i(\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)) = c_i \text{ for } i = 1, \ldots, n_I\}. \tag{3.2}$$

3.1.2. Constrained Neural Galerkin schemes. We now apply the Dirac–Frenkel variational principle with respect to the constrained manifold $\mathcal{M}_I$ instead of $\mathcal{M}$ to determine $\dot{\boldsymbol{\theta}}(t)$; see Figure 2a. The projection onto a tangent space of $\mathcal{M}_I$ is analogous to the projection onto a tangent space of $\mathcal{M}$ and leads to the constrained Neural Galerkin system

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{\theta}(t)) & \boldsymbol{g}(\boldsymbol{\theta}(t)) \\ \boldsymbol{g}(\boldsymbol{\theta}(t))^\mathsf{T} & 0 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}}(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}(\boldsymbol{\theta}(t)) \\ 0 \end{bmatrix}, \tag{3.3}$$

where $\boldsymbol{g}(\boldsymbol{\theta}(t))$ is

$$\boldsymbol{g}(\boldsymbol{\theta}(t)) = \left[\nabla_{\boldsymbol{\theta}} I_1(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)), \ldots, \nabla_{\boldsymbol{\theta}} I_{n_I}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\right]\,. \tag{3.4}$$

System (3.3) is analogous to the system introduced in [3]. The vector $\boldsymbol{\lambda}(t)$ contains the Lagrange multipliers at time t. The system (3.3) has a unique solution $(\dot{\boldsymbol{\theta}}(t), \boldsymbol{\lambda}(t))$ if $\boldsymbol{M}(\boldsymbol{\theta}(t))$ is regular and $\boldsymbol{g}(\boldsymbol{\theta}(t))$ has full column rank [8, Chapter 5]. Moreover, by verifying that the requirements of [38, Thm. 4.13] are satisfied, the initial value problem associated with the system of differential-algebraic equations (3.3) has locally a unique solution, provided that $\boldsymbol{M}$ is pointwise invertible, $\boldsymbol{g}$ has pointwise full column rank, and $\boldsymbol{M}, \boldsymbol{g}, \boldsymbol{F}$ are continuously differentiable. If the constant in the definition (3.2) of the manifold $\mathcal{M}_I$ is set to $\boldsymbol{c} = \left[I_1(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(0), \cdot), \ldots, I_{n_I}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(0), \cdot)\right]$, then a continuous-time Neural Galerkin solution $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)$ with $\boldsymbol{\theta}(t)$ satisfying the constrained Neural Galerkin system (3.3) conserves the quantities $I_1, \ldots, I_{n_I}$.

3.1.3. Sampled constrained Neural Galerkin schemes. Analogously, we can introduce the manifold $\hat{\mathcal{M}}_I$ that is based on the sampled quantities (2.4),

$$\hat{\mathcal{M}}_I = \{\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) : \boldsymbol{\eta} \in \Theta \text{ and } \hat{I}_i(\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)) = c_i \text{ for } i = 1, \ldots, n_I\}\,, \tag{3.5}$$

and derive the sampled constrained Neural Galerkin system as

$$\begin{bmatrix} \hat{\boldsymbol{M}}(\boldsymbol{\theta}(t)) & \hat{\boldsymbol{g}}(\boldsymbol{\theta}(t)) \\ \hat{\boldsymbol{g}}(\boldsymbol{\theta}(t))^\mathsf{T} & 0 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}}(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{F}}(\boldsymbol{\theta}(t)) \\ 0 \end{bmatrix}, \tag{3.6}$$

where $\hat{\boldsymbol{M}}(\boldsymbol{\theta}(t))$ and $\hat{\boldsymbol{F}}(\boldsymbol{\theta}(t))$ are the sampled $\boldsymbol{M}(\boldsymbol{\theta}(t))$ and $\boldsymbol{F}(\boldsymbol{\theta}(t))$, respectively. In $\hat{\boldsymbol{g}}(\boldsymbol{\theta}(t))$, the sampled quantities (2.4) are used as

$$\hat{\boldsymbol{g}}(\boldsymbol{\theta}(t)) = \left[\nabla_{\boldsymbol{\theta}} \hat{I}_1(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot)), \ldots, \nabla_{\boldsymbol{\theta}} \hat{I}_{n_I}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\right]\,, \tag{3.7}$$

so that solutions of the sampled constrained system (3.6) conserve the sampled quantities (2.4).

3.2. Structured nonlinear parametrizations and weighted Neural Galerkin schemes to preserve Hamiltonians in continuous time. We now derive specific nonlinear parametrizations and weighted schemes that preserve Hamiltonians without having to resort to using constraints.

3.2.1. Separable nonlinear parametrizations. To preserve Hamiltonians in time-continuous Neural Galerkin solutions, we consider separable parametrizations that are of the form

$$\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}) = \sum\nolimits_{i=1}^{n_\phi} \boldsymbol{\beta}_i(t) \phi_i(\boldsymbol{x}, \boldsymbol{\alpha}_i(t)) \tag{3.8}$$

with $\boldsymbol{\beta}_i : [0, \infty) \to \mathbb{R}^m$, $\boldsymbol{\alpha}_i : [0, \infty) \to \mathbb{R}^{q_i}$, $\phi_i : \mathcal{X} \times \mathbb{R}^{q_i} \to \mathbb{R}$. The parameters can be combined into

$$\boldsymbol{\theta}(t) = [\boldsymbol{\alpha}_1(t)^\mathsf{T}, \ldots, \boldsymbol{\alpha}_{n_\phi}(t)^\mathsf{T}, \boldsymbol{\beta}_1(t)^\mathsf{T}, \ldots, \boldsymbol{\beta}_{n_\phi}(t)^\mathsf{T}]^\mathsf{T} \in \Theta\,. \tag{3.9}$$

We have $n_{\boldsymbol{\theta}} = n_\phi m + \sum_{i=1}^{n_\phi} q_i$ and $\Theta = \mathbb{R}^{n_{\boldsymbol{\theta}}}$. A parametrization of the form (3.8) is separable because $\boldsymbol{\theta}$ can be separated into the components $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ enters the parametrization linearly. A similar separable parametrization has been used in the context of nonlinear model reduction for finite-dimensional port-Hamiltonian systems in [62]. In the context of deep-network parametrizations, an architecture (3.8) is obtained whenever the last (output) layer of the network is linear without bias. In the following, it will be convenient to introduce the matrix function $\boldsymbol{V} \colon \mathcal{X} \times \mathbb{R}^{\sum_{i=1}^{n_\phi} q_i} \to \mathbb{R}^{m \times n_\phi m}$ via $\boldsymbol{V}(\boldsymbol{x}, \boldsymbol{\alpha}) = [\phi_1(\boldsymbol{x}, \boldsymbol{\alpha}_1), \cdots, \phi_{n_\phi}(\boldsymbol{x}, \boldsymbol{\alpha}_{n_\phi})] \otimes \mathbf{1}_m$, where $\otimes$ denotes the Kronecker product and $\mathbf{1}_m$ the $m \times m$ identity matrix.

3.2.2. Hamiltonians with factorizable structure. We call a Hamiltonian H factorizable if there exist a continuously differentiable function $h\colon \mathbb{R}^m \to \mathbb{R}$ and a point-wise symmetric and positive definite matrix function $\boldsymbol{Q}\colon \mathbb{R}^m \to \mathbb{R}^{m\times m}$ that satisfy

$$H(\boldsymbol{v}) = \int_{\mathcal{X}} h(\boldsymbol{v}(\boldsymbol{x}))\mathrm{d}\nu(\boldsymbol{x})\,, \qquad \text{and} \tag{3.10}$$

$$\left(\frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{v})\right)(\boldsymbol{x}) = \boldsymbol{Q}(\boldsymbol{v}(\boldsymbol{x}))\boldsymbol{v}(\boldsymbol{x}) \quad \text{for all } (\boldsymbol{v},\boldsymbol{x}) \in \mathcal{U}\times\mathcal{X}\,, \tag{3.11}$$

where $\delta H/\delta \boldsymbol{u}$ denotes the variational derivative of H. Hamiltonians are factorizable if they correspond to a squared norm of the state, which is, for instance, the case for the Burgers' and wave equation examples in (2.9)–(2.10). However, the Hamiltonian considered for the shallow water equations in the later Section 6.4 is not factorizable in the sense of (3.10)–(3.11) because it involves the gradient of one of the state variables.

3.2.3. Weighted Neural Galerkin schemes that preserve Hamiltonians in continuous time. We now introduce a Neural Galerkin scheme that conserves Hamiltonians that satisfy (3.11) with a matrix function $\boldsymbol{Q}$. To this end, we follow the Dirac–Frenkel approach as in (2.12) and perform the projection with respect to the weighted inner product given by the function $\boldsymbol{Q}$ of (3.11); see also [62] for a similar approach in the context of model reduction. Consider a separable nonlinear parametrization (3.8) with the parameter $\boldsymbol{\theta}(t)$ given in (3.9) and define $\boldsymbol{M}_{\boldsymbol{Q}}\colon \mathbb{R}^{n_{\boldsymbol{\theta}}} \to \mathbb{R}^{n_{\boldsymbol{\theta}}\times n_{\boldsymbol{\theta}}}$ and $\boldsymbol{F}_{\boldsymbol{Q}}\colon \mathbb{R}^{n_{\boldsymbol{\theta}}} \to \mathbb{R}^{n_{\boldsymbol{\theta}}}$ as

$$\boldsymbol{M}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) = \begin{bmatrix} \boldsymbol{M}^{(11)}(\boldsymbol{\theta}(t)) & \boldsymbol{M}^{(12)}(\boldsymbol{\theta}(t)) \\ \boldsymbol{M}^{(12)}(\boldsymbol{\theta}(t))^{\mathsf{T}} & \boldsymbol{M}^{(22)}(\boldsymbol{\theta}(t)) \end{bmatrix}, \qquad \boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) = \begin{bmatrix} \boldsymbol{F}^{(1)}(\boldsymbol{\theta}(t)) \\ \boldsymbol{F}^{(2)}(\boldsymbol{\theta}(t)) \end{bmatrix}, \tag{3.12}$$

with the blocks defined as

$$\begin{aligned}
\boldsymbol{M}^{(11)}_{ij}(\boldsymbol{\theta}(t)) &= \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))(\boldsymbol{e}_i)\boldsymbol{\beta}(t),\ \partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))(\boldsymbol{e}_j)\boldsymbol{\beta}(t)\rangle_{\nu}\,,\\
\boldsymbol{M}^{(12)}_{i\ell}(\boldsymbol{\theta}(t)) &= \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))(\boldsymbol{e}_i)\boldsymbol{\beta}(t),\ \boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_\ell\rangle_{\nu}\,,\\
\boldsymbol{M}^{(22)}_{k\ell}(\boldsymbol{\theta}(t)) &= \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_k,\ \boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_\ell\rangle_{\nu}\,,\\
\boldsymbol{F}^{(1)}_{i}(\boldsymbol{\theta}(t)) &= \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))(\boldsymbol{e}_i)\boldsymbol{\beta}(t),\\
&\qquad\qquad J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot)\rangle_{\nu}\,,\\
\boldsymbol{F}^{(2)}_{k}(\boldsymbol{\theta}(t)) &= \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_k,\ J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot)\rangle_{\nu}\,,
\end{aligned} \tag{3.13}$$

for $i,j = 1,\ldots,\sum_{s=1}^{n_\phi} q_s$, $k,\ell = 1,\ldots,n_\phi m$, and the ith canonical unit vector $\boldsymbol{e}_i$. Here, we use the notation $\partial_{\boldsymbol{\alpha}}\boldsymbol{V}\colon \mathcal{X}\times\mathbb{R}^{\sum_{i=1}^{n_\phi} q_i} \to \mathrm{Hom}(\mathbb{R}^{\sum_{i=1}^{n_\phi} q_i}, \mathbb{R}^{m\times n_\phi m})$ for the (block) partial derivative of $\boldsymbol{V}$ with respect to $\boldsymbol{\alpha}$. Hence, for given $\boldsymbol{x}\in\mathcal{X}$ and $\boldsymbol{v},\boldsymbol{w}\in\mathbb{R}^{\sum_{i=1}^{n_\phi} q_i}$, $\partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\boldsymbol{x},\boldsymbol{v})$ is a linear mapping from $\mathbb{R}^{\sum_{i=1}^{n_\phi} q_i}$ to $\mathbb{R}^{m\times n_\phi m}$ and $\partial_{\boldsymbol{\alpha}}\boldsymbol{V}(\boldsymbol{x},\boldsymbol{v})(\boldsymbol{w})$ is in $\mathbb{R}^{m\times n_\phi m}$.

The matrix functions $\boldsymbol{M}_{\boldsymbol{Q}}$ and $\boldsymbol{F}_{\boldsymbol{Q}}$ lead to the weighted time-continuous Neural Galerkin system

$$\boldsymbol{M}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t) = \boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t))\,. \tag{3.14}$$

The following proposition states that H is indeed a conserved quantity of (3.14).

PROPOSITION 3.1. *Consider a Hamiltonian PDE of the form (2.1) with right-hand side satisfying (2.5)–(2.7) and let the Hamiltonian be factorizable. Then, any solution of the corresponding time-continuous weighted Neural Galerkin system (3.14) based on the separable parametrization (3.8) preserves the Hamiltonian H in the sense of*

$$\frac{\mathrm{d}H(\boldsymbol{\theta}(t))}{\mathrm{d}t}(t) = 0 \quad \textit{for all } t \in [0, \infty)\,,$$

where we overload the notation to use $H(\boldsymbol{\theta}(t))$ as short-hand notation for $H(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))$.

Proof. First, we compute the partial derivatives of the Hamiltonian with respect to the parameters in $\boldsymbol{\theta}$. Using the separable structure (3.8), the factorization (3.11), and the definitions of $\boldsymbol{M}^{(12)}, \boldsymbol{M}^{(22)}$ in (3.13), we obtain

$$\nabla_{\boldsymbol{\alpha}} H(\boldsymbol{\eta}) = \boldsymbol{M}^{(12)}(\boldsymbol{\eta})\boldsymbol{\eta}_2, \quad \nabla_{\boldsymbol{\beta}} H(\boldsymbol{\eta}) = \boldsymbol{M}^{(22)}(\boldsymbol{\eta})\boldsymbol{\eta}_2$$

for all $\boldsymbol{\eta} \in \mathbb{R}^{n_{\boldsymbol{\theta}}}$, where $\boldsymbol{\eta}_2 \in \mathbb{R}^{n_\phi m}$ denotes the last block component of $\boldsymbol{\eta}$. In total, this yields

$$\nabla H(\boldsymbol{\eta}) = \boldsymbol{M}_{\boldsymbol{Q}}(\boldsymbol{\eta}) \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\eta} = \boldsymbol{M}_{\boldsymbol{Q}}(\boldsymbol{\eta})^{\mathsf{T}} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\eta}. \tag{3.15}$$

Furthermore, exploiting (3.13), we observe that the right-hand side of (3.14) may be factorized

$$\boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\eta}) = \boldsymbol{J}_{\boldsymbol{Q}}(\boldsymbol{\eta}) \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\eta} \tag{3.16}$$

for all $\boldsymbol{\eta} \in \mathbb{R}^{n_{\boldsymbol{\theta}}}$, where $\boldsymbol{J}_{\boldsymbol{Q}} \colon \mathbb{R}^{n_{\boldsymbol{\theta}}} \to \mathbb{R}^{n_{\boldsymbol{\theta}} \times n_{\boldsymbol{\theta}}}$ is defined via

$$\boldsymbol{J}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) := \begin{bmatrix} 0 & \boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t)) \\ -\boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t))^{\mathsf{T}} & \boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t)) \end{bmatrix},$$

with the blocks

$$\begin{aligned} \boldsymbol{J}^{(12)}_{i,\ell}(\boldsymbol{\theta}(t)) = \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\partial_{\boldsymbol{\alpha}} \boldsymbol{V}(\cdot, \boldsymbol{\alpha}(t))(\boldsymbol{e}_i)\boldsymbol{\beta}(t), \\ J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{V}(\cdot, \boldsymbol{\alpha}(t))\boldsymbol{e}_\ell \rangle_\nu\,, \end{aligned} \tag{3.17}$$

$$\begin{aligned} \boldsymbol{J}^{(22)}_{k,\ell}(\boldsymbol{\theta}(t)) = \langle \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{V}(\cdot, \boldsymbol{\alpha}(t))\boldsymbol{e}_k, \\ J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{V}(\cdot, \boldsymbol{\alpha}(t))\boldsymbol{e}_\ell \rangle_\nu\,, \end{aligned} \tag{3.18}$$

for $i = 1, \ldots, \sum_{s=1}^{n_\phi} q_s$ and $k, \ell = 1, \ldots, n_\phi m$. By exploiting the pointwise skew-adjointness of J, we conclude that $\boldsymbol{J}^{(22)}$ and $\boldsymbol{J}_{\boldsymbol{Q}}$ are pointwise skew-symmetric. Hence, in total we have shown that (3.12)–(3.14) has a port-Hamiltonian structure as in [45] without dissipation or input/output ports. Following [45], we obtain the conservation of the Hamiltonian from the calculation

$$\begin{aligned} \frac{\mathrm{d}H(\boldsymbol{\theta}(t))}{\mathrm{d}t}(t) &= \nabla H(\boldsymbol{\theta}(t))^{\mathsf{T}} \dot{\boldsymbol{\theta}}(t) = \boldsymbol{\theta}(t)^{\mathsf{T}} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{M}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) \dot{\boldsymbol{\theta}}(t) \\ &= \boldsymbol{\theta}(t)^{\mathsf{T}} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) \\ &= \boldsymbol{\theta}(t)^{\mathsf{T}} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{J}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\theta}(t) = 0. \end{aligned}$$

∎

3.2.4. Weighted Neural Galerkin schemes with Monte Carlo approximations. We now show that the sampled quantities are conserved with weighted Neural Galerkin schemes in continuous time even if the integrals occurring in $\boldsymbol{M}_{\boldsymbol{Q}}$ and $\boldsymbol{F}_{\boldsymbol{Q}}$ are replaced by corresponding Monte Carlo estimates. Replacing $\boldsymbol{M}_{\boldsymbol{Q}}$ with its sampled counterpart $\hat{\boldsymbol{M}}_{\boldsymbol{Q}}$ analogous to $\hat{\boldsymbol{M}}$ defined in (2.16) poses no problems in terms of structure preservation. In contrast, the right-hand side term $\boldsymbol{F}_{\boldsymbol{Q}}$ needs more careful treatment because applying Monte Carlo directly to $\boldsymbol{F}_{\boldsymbol{Q}}$ can destroy the property (3.16) that is used in the proof of Proposition 3.1. To avoid loosing property (3.16) in the sampled $\boldsymbol{F}_{\boldsymbol{Q}}$, we first use (3.16) and the skew symmetry of $\boldsymbol{J}_{22}$ defined in (3.18) and write $\boldsymbol{F}_{\boldsymbol{Q}}$ as

$$\begin{aligned}\boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) &= \begin{bmatrix} 0 & \boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t)) \\ -\boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t))^\mathsf{T} & \frac{1}{2}(\boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t)) - \boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t))^\mathsf{T}) \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\theta}(t) \\ &= \begin{bmatrix} \boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t))\boldsymbol{\beta}(t) \\ \frac{1}{2}\boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t))\boldsymbol{\beta}(t) - \frac{1}{2}\boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t))^\mathsf{T}\boldsymbol{\beta}(t) \end{bmatrix}.\end{aligned}$$

Instead of approximating $\boldsymbol{F}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t))$ directly via Monte Carlo estimates, we propose to approximate $\boldsymbol{J}^{(12)}(\boldsymbol{\theta}(t))\boldsymbol{\beta}(t)$ as well as $\boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t))\boldsymbol{\beta}(t)$ and $\boldsymbol{J}^{(22)}(\boldsymbol{\theta}(t))^\mathsf{T}\boldsymbol{\beta}(t)$ separately. The approximations can then be used to assemble the approximation $\hat{\boldsymbol{F}}_{\boldsymbol{Q}}$ of $\boldsymbol{F}_{\boldsymbol{Q}}$ that can be factorized analogous to (3.16). The sampled weighted Neural Galerkin system is

$$\hat{\boldsymbol{M}}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{J}}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{1}_{n_\phi m} \end{bmatrix} \boldsymbol{\theta}(t)\,, \tag{3.19}$$

where the Monte Carlo approximations of $\boldsymbol{M}_{\boldsymbol{Q}}$ and $\boldsymbol{J}_{\boldsymbol{Q}}$ are denoted as $\hat{\boldsymbol{M}}_{\boldsymbol{Q}}, \hat{\boldsymbol{J}}_{\boldsymbol{Q}} \colon \mathbb{R}^{n_{\boldsymbol{\theta}}} \to \mathbb{R}^{n_{\boldsymbol{\theta}} \times n_{\boldsymbol{\theta}}}$, respectively, and defined as

$$\begin{aligned}\hat{\boldsymbol{M}}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) &= \begin{bmatrix} \hat{\boldsymbol{M}}^{(11)}(\boldsymbol{\theta}(t)) & \hat{\boldsymbol{M}}^{(12)}(\boldsymbol{\theta}(t)) \\ \hat{\boldsymbol{M}}^{(12)}(\boldsymbol{\theta}(t))^\mathsf{T} & \hat{\boldsymbol{M}}^{(22)}(\boldsymbol{\theta}(t)) \end{bmatrix}, \\ \hat{\boldsymbol{J}}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t)) &= \begin{bmatrix} 0 & \hat{\boldsymbol{J}}^{(12)}(\boldsymbol{\theta}(t)) \\ -\hat{\boldsymbol{J}}^{(12)}(\boldsymbol{\theta}(t))^\mathsf{T} & \hat{\boldsymbol{J}}^{(22)}(\boldsymbol{\theta}(t)) \end{bmatrix}\end{aligned} \tag{3.20}$$

with the blocks

$$\begin{aligned}\hat{\boldsymbol{M}}^{(11)}_{ij}(\boldsymbol{\theta}(t)) =& \frac{1}{n_S}\sum_{s=1}^{n_S} \boldsymbol{\beta}(t)^\mathsf{T} \left(\partial_{\boldsymbol{\alpha}} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))\boldsymbol{e}_i\right)^\mathsf{T} \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_s))^\mathsf{T} \\ & \cdot \partial_{\boldsymbol{\alpha}} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))(\boldsymbol{e}_j)\boldsymbol{\beta}(t), \\ \hat{\boldsymbol{M}}^{(12)}_{i\ell}(\boldsymbol{\theta}(t)) =& \frac{1}{n_S}\sum_{s=1}^{n_S} \boldsymbol{\beta}(t)^\mathsf{T} \left(\partial_{\boldsymbol{\alpha}} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))\boldsymbol{e}_i\right)^\mathsf{T} \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_s))^\mathsf{T} \\ & \cdot \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))\boldsymbol{e}_\ell, \\ \hat{\boldsymbol{M}}^{(22)}_{k\ell}(\boldsymbol{\theta}(t)) =& \frac{1}{n_S}\sum_{s=1}^{n_S} \boldsymbol{e}_k^\mathsf{T} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))^\mathsf{T} \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_s))^\mathsf{T} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))\boldsymbol{e}_\ell, \\ \hat{\boldsymbol{J}}^{(12)}_{i\ell}(\boldsymbol{\theta}(t)) =& \frac{1}{n_S}\sum_{s=1}^{n_S} \boldsymbol{\beta}(t)^\mathsf{T} \left(\partial_{\boldsymbol{\alpha}} \boldsymbol{V}(\boldsymbol{x}_s, \boldsymbol{\alpha}(t))\boldsymbol{e}_i\right)^\mathsf{T} \boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \boldsymbol{x}_s))^\mathsf{T} \\ & \cdot \left(J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t), \cdot))\boldsymbol{V}(\cdot, \boldsymbol{\alpha}(t))\boldsymbol{e}_\ell\right)(\boldsymbol{x}_s)\,,\end{aligned}$$

$$\begin{aligned}\hat{\boldsymbol{J}}^{(22)}_{k\ell}(\boldsymbol{\theta}(t)) =& \frac{1}{2n_S}\sum_{s=1}^{n_S} \boldsymbol{e}_k^\mathsf{T}\boldsymbol{V}(\boldsymbol{x}_s,\boldsymbol{\alpha}(t))^\mathsf{T}\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\boldsymbol{x}_s))^\mathsf{T} \\ &\qquad \cdot \left(J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_\ell\right)(\boldsymbol{x}_s) \\ &- \frac{1}{2n_S}\sum_{s=1}^{n_S} \boldsymbol{e}_\ell^\mathsf{T}\boldsymbol{V}(\boldsymbol{x}_s,\boldsymbol{\alpha}(t))^\mathsf{T}\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\boldsymbol{x}_s))^\mathsf{T} \\ &\qquad \cdot \left(J(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{Q}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot))\boldsymbol{V}(\cdot,\boldsymbol{\alpha}(t))\boldsymbol{e}_k\right)(\boldsymbol{x}_s)\,,\end{aligned}$$

for $i,j = 1,\ldots,\sum_{s=1}^{n_\phi} q_s$, $k,\ell = 1,\ldots,n_\phi m$.

The following proposition states that the sampled Hamiltonian $\widehat{H}$ is a conserved quantity of the sampled weighted Neural Galerkin system (3.19).

PROPOSITION 3.2. *Let the assumptions of Proposition 3.1 be satisfied and consider the sampled weighted Neural Galerkin system* (3.19)*. Moreover, let the associated Hamiltonian* $\widehat{H}$ *be based on the same sampling points, i.e,* $n_S = n_M$ *and* $\boldsymbol{x}_i = \boldsymbol{\xi}_i$ *for* $i = 1,\ldots,n_S$*. Then, any solution of* (3.19) *satisfies*

$$\frac{\mathrm{d}\widehat{H}(\boldsymbol{\theta}(t))}{\mathrm{d}t}(t) = 0 \quad \textit{for all } t \in [0,\infty)\,,$$

where we again remind the reader that $\widehat{H}(\boldsymbol{\theta}(t))$ *is short-hand notation for* $\widehat{H}(\hat{u}(\boldsymbol{\theta}(t),\cdot))$*.*

Proof. First, we note that the integral form (3.10) of H implies

$$\frac{\delta H}{\delta \boldsymbol{u}}(\boldsymbol{v}) = \nabla h \circ \boldsymbol{v} = \boldsymbol{Q}(\boldsymbol{v})\boldsymbol{v}$$

for all $\boldsymbol{v} \in \mathcal{U}$. Then, by straightforward calculations, we obtain that

$$\nabla\widehat{H}(\boldsymbol{\eta}) = \hat{\boldsymbol{M}}_{\boldsymbol{Q}}(\boldsymbol{\eta})^\mathsf{T}\begin{bmatrix}0 & 0\\ 0 & \mathbf{1}_{n_\phi m}\end{bmatrix}\boldsymbol{\eta}$$

holds for all $\boldsymbol{\eta} \in \mathbb{R}^{n_{\boldsymbol{\theta}}}$, which is the same structure as (3.15) in the proof of Proposition 3.1. Thus, the conservation of $\widehat{H}$ follows from analogous arguments as the ones of the proof of Proposition 3.1. ∎

Remark 3.3. The matrix function $\hat{\boldsymbol{J}}^{(22)}$ is obtained by exploiting the pointwise skew-symmetry of $\boldsymbol{J}^{(22)}$ to ensure the pointwise skew-symmetry of $\hat{\boldsymbol{J}}^{(22)}$. However, this construction of $\hat{\boldsymbol{J}}^{(22)}$ cannot guarantee that (3.19) with given $\boldsymbol{\theta}(t)$ and singular $\hat{\boldsymbol{M}}_{\boldsymbol{Q}}(\boldsymbol{\theta}(t))$ can be satisfied for a $\dot{\boldsymbol{\theta}}(t)$. The same issue applies to the special case of linear parametrizations.

4. Conserving quantities in time-discrete Neural Galerkin approximations. We now consider the time discretization of the constrained or weighted Neural Galerkin systems, which is delicate because the nonlinear dependence of the parametrization $\hat{\boldsymbol{u}}(\boldsymbol{\theta}(t),\cdot)$ on the parameter $\boldsymbol{\theta}(t)$ means that quantities (2.3) become nonlinear in $\boldsymbol{\theta}(t)$ and thus are not conserved by just applying Runge-Kutta integrators; see also the problem formulation in Section 2.5. Building on literature of ODE integrators [30, Chapter VII.2], we propose to use a nonlinear projection method that computes embeddings to conserve quantities in Neural Galerkin solutions in discrete time; see Figure 2b. Importantly, the nonlinear projection approach is applicable with implicit and explicit time integration schemes. Explicit schemes are of especially great interest

in the context of nonlinear parametrizations with Neural Galerkin schemes because they lead to linear least-squares regression problems at each time step whereas implicit schemes lead to non-convex optimization problems at each time step [11, 7]. We also discuss an alternative time discretization scheme which is implicit and based on discrete gradients.

4.1. Neural Galerkin schemes with embeddings for conserving quantities in discrete time. Consider a one-step time integrator applied to the constrained Neural Galerkin system (3.3) or the weighted Neural Galerkin system (3.14) with time-step size $\delta t > 0$. Such an integrator gives rise to a map $\boldsymbol{\Phi}_{\delta t} : \Theta \to \Theta$ that takes a parameter vector $\boldsymbol{\theta}_k \in \Theta$ at time step k and maps it onto $\boldsymbol{\theta}_{k+1} \in \Theta$ at time step $k+1$. Let $\boldsymbol{\theta}_0$ be the parameter of the initial condition $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_0, \cdot)$ and let $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K \in \Theta$ be the parameters obtained with the time integrator at time steps $k = 1, \ldots, K$, which lead to the time-discrete Neural Galerkin solution trajectory $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_1, \cdot), \hat{\boldsymbol{u}}(\boldsymbol{\theta}_2, \cdot), \ldots, \hat{\boldsymbol{u}}(\boldsymbol{\theta}_K, \cdot)$. At each time step k, we want to ensure that the function $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_{k+1}, \cdot)$ at the next time step $k+1$ is on the constrained manifold $\mathcal{M}_I$ if the approximate solution field function $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \cdot)$ at the current time step k is on $\mathcal{M}_I$. To achieve this, at each time step $k = 0, \ldots, K-1$, we evaluate $\boldsymbol{\Phi}_{\delta t}$ at $\boldsymbol{\theta}_k$ to compute $\tilde{\boldsymbol{\theta}}_{k+1}$, which can correspond to a function $\hat{\boldsymbol{u}}(\tilde{\boldsymbol{\theta}}_{k+1}, \cdot)$ that is outside of $\mathcal{M}_I$. We then follow [31, Chapter IV] and embed the function $\hat{\boldsymbol{u}}(\tilde{\boldsymbol{\theta}}_{k+1}, \cdot)$ corresponding to the parameter $\tilde{\boldsymbol{\theta}}_{k+1}$ as a function with parameter $\boldsymbol{\theta}_{k+1}$ onto $\mathcal{M}_I$ by solving for $\boldsymbol{\theta}_{k+1}$ via the nonlinear least-squares problem

$$\min_{\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) \in \mathcal{M}_I} \frac{1}{2} \|\boldsymbol{\eta} - \tilde{\boldsymbol{\theta}}_{k+1}\|_2^2 . \tag{4.1}$$

The embedding step constrains the manifold on which we seek approximations to the PDE solution to $\mathcal{M}_I$, where quantities are conserved. The analogous procedure can be derived for embeddings onto $\hat{\mathcal{M}}_I$ defined in (3.5),

$$\min_{\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) \in \hat{\mathcal{M}}_I} \frac{1}{2} \|\boldsymbol{\eta} - \tilde{\boldsymbol{\theta}}_{k+1}\|_2^2 . \tag{4.2}$$

Note that the operation performed by solving the nonlinear least-squares problem (4.1) is sometimes referred to as nonlinear projection. We use the term embedding to distinguish clearly from the linear projection of the right-hand side onto the tangent space in the Dirac-Frenkel variational principle; see Section 2.4.

4.2. Implicit time discretizations with discrete gradients. Discretizing the second block equation in (3.3) via discrete gradients also leads to Neural Galerkin approximations that conserve quantities [25, 44]. The time integration via discrete gradients is implicit in time and thus can be computationally expensive in the context of Neural Galerkin schemes with nonlinear parametrizations because a potentially non-convex optimization problem has to be solved at each step; see [11, 7]. A discrete gradient [25, 44] of a continuously differentiable function $H \colon \mathbb{R}^n \to \mathbb{R}$ is given by a continuous mapping $\overline{\nabla} H \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ which satisfies

$$\overline{\nabla} H(\boldsymbol{\eta}, \boldsymbol{\eta}) = \nabla H(\boldsymbol{\eta}), \quad \overline{\nabla} H(\boldsymbol{\eta}, \boldsymbol{\eta}')^\mathsf{T} (\boldsymbol{\eta}' - \boldsymbol{\eta}) = H(\boldsymbol{\eta}') - H(\boldsymbol{\eta}) \tag{4.3}$$

for all $\boldsymbol{\eta}, \boldsymbol{\eta}' \in \mathbb{R}^n$. Based on discrete gradients, we discretize the second block equation $g(\boldsymbol{\theta}(t))^\mathsf{T} \dot{\boldsymbol{\theta}}(t) = 0$ in (3.3) by replacing $\dot{\boldsymbol{\theta}}(t)$ by a finite difference approximation with time-step size $\delta t > 0$ and the gradients in (3.4) by corresponding discrete gradients.

For $k = 0, \dots, K-1$, the resulting time-discrete equation is

$$\overline{g}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1})^\mathsf{T} \frac{\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k}{\Delta t} = 0, \tag{4.4}$$

with $\overline{g}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1}) := \left[\overline{\nabla} I_1(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1}), \dots, \overline{\nabla} I_{n_I}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1})\right]$ using the discrete gradients $\overline{\nabla} I_i(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1})$ for $i = 1, \dots, n_I$. By using the second relation in (4.3), we infer that the ith row of (4.4) is

$$0 = \overline{\nabla} I_i(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k+1})^\mathsf{T} \frac{\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k}{\Delta t} = \frac{I_i(\boldsymbol{\theta}_{k+1}) - I_i(\boldsymbol{\theta}_k)}{\Delta t}$$

for $i = 1, \dots, n_I$. Consequently, (4.4) is equivalent to

$$I_1(\boldsymbol{\theta}_{k+1}) = I_1(\boldsymbol{\theta}_k), \quad \dots, \quad I_{n_I}(\boldsymbol{\theta}_{k+1}) = I_{n_I}(\boldsymbol{\theta}_k), \tag{4.5}$$

i.e., in this setting we do not have to construct the discrete gradients explicitly, but we may directly use (4.5). The first equation in (3.3) can be discretized by any time discretization scheme. Similarly, discrete gradients may be also used to ensure conservation in the context of the weighted Neural Galerkin systems (3.14); see [61, App. C] for more details.

5. Computational aspects of Neural Galerkin with embeddings. We now describe computational aspects of constrained Neural Galerkin schemes with nonlinear embeddings for conserving quantities. In particular, we use the specific iterations as in [31] to efficiently perform the embedding onto the constrained manifold. While we focus on constrained systems with embeddings, similar observation holds for the weighted Neural Galerkin schemes and time discretizations with discrete gradients.

5.1. Constrained Neural Galerkin schemes represented as least-squares problems. While the sampled Neural Galerkin system (2.18) introduced in Section 2.4 can be numerically solved directly, the corresponding linear least-squares problem is typically better conditioned. Recall that $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n_S}$ are the sample points used to obtain the sampled Neural Galerkin system (2.18). If the solution field is scalar valued so that $m = 1$ and we use explicit Euler to discretize time with time-step size $\delta t > 0$, then the corresponding sampled least-squares problem at time steps $k = 0, \dots, K-1$ is

$$\min_{\delta\boldsymbol{\theta}_k \in \Theta} \|\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k)\delta\boldsymbol{\theta}_k - \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k)\|_2^2, \tag{5.1}$$

where $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \delta t \delta\boldsymbol{\theta}_k$ and $\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k) \in \mathbb{R}^{n_S \times n_{\boldsymbol{\theta}}}$ and $\hat{\boldsymbol{b}}(\boldsymbol{\theta}_k) \in \mathbb{R}^{n_S}$ are

$$\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k) = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \boldsymbol{x}_1)^\mathsf{T} \\ \vdots \\ \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \boldsymbol{x}_{n_S})^\mathsf{T} \end{bmatrix}, \quad \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k) = \begin{bmatrix} \boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \cdot))(\boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{f}(\cdot, \hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \cdot))(\boldsymbol{x}_{n_S}) \end{bmatrix}. \tag{5.2}$$

We use a singular value decomposition for solving the least-squares problem (5.1) and truncate singular values smaller than a prescribed tolerance of 10^{-5}, so that we obtain the minimal-norm solution in case the matrix $\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k)$ has a rank smaller than $n_{\boldsymbol{\theta}}$. The conservation constraints are added to (5.1) analogous to (3.3), which results in a least-squares problem with equality constraints

$$\min_{\delta\boldsymbol{\theta}_k \in \Theta} \left\| \hat{\boldsymbol{A}}(\boldsymbol{\theta}_k)\delta\boldsymbol{\theta}_k - \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k) \right\|_2^2 \quad \text{s.t. } \hat{\boldsymbol{g}}(\boldsymbol{\theta}_k)^\mathsf{T} \delta\boldsymbol{\theta}_k = 0, \tag{5.3}$$

Algorithm 5.1 Neural Galerkin scheme with embeddings

procedure NGEMBEDDING($\boldsymbol{f}$, δt, K, $\hat{\boldsymbol{u}}$, n_S, $\boldsymbol{\theta}_0$, $I_1, \ldots, I_{n_I}$, n_M)
  for $k = 0, \ldots, K-1$ **do**
    Estimate $\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k)$ and $\hat{\boldsymbol{b}}(\boldsymbol{\theta}_k)$ defined in (5.2) using n_S sample points.
    Estimate $\hat{\boldsymbol{g}}(\boldsymbol{\theta}_k)$ using n_M sample points and quantities $I_1, \ldots, I_{n_I}$.
    Compute $\delta\boldsymbol{\theta}_k$ as solution of (5.3).
    Set $\tilde{\boldsymbol{\theta}}_{k+1} = \boldsymbol{\theta}_k + \delta t \delta\boldsymbol{\theta}_k$.
    Iterate (5.7) to compute $\boldsymbol{\theta}_{k+1}$.
  end for
  Return trajectory $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_0, \cdot), \hat{\boldsymbol{u}}(\boldsymbol{\theta}_1, \cdot), \ldots, \hat{\boldsymbol{u}}(\boldsymbol{\theta}_K, \cdot)$
end procedure

where $\hat{\boldsymbol{g}}$ is defined in (3.7). An analogous least-squares problem with equality constraints can be derived for solution fields with multiple outputs $m > 1$ and other time-integration schemes.

5.2. Computing embeddings onto the constrained manifold. To numerically solve (4.1) with respect to the sampled manifold $\hat{\mathcal{M}}_I$, we introduce the Lagrangian function

$$(\boldsymbol{\eta}, \boldsymbol{\lambda}) \mapsto \frac{1}{2}\|\boldsymbol{\eta} - \tilde{\boldsymbol{\theta}}_{k+1}\|_2^2 + \boldsymbol{\lambda} \cdot \hat{\boldsymbol{m}}(\boldsymbol{\eta}), \tag{5.4}$$

where $\cdot$ denotes the Euclidean inner product and the differentiable function $\hat{\boldsymbol{m}} : \mathbb{R}^{n_{\boldsymbol{\theta}}} \to \mathbb{R}^{n_I}$ is defined as

$$\hat{\boldsymbol{m}}(\boldsymbol{\eta}) = [\hat{I}_1(\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)) - \hat{I}_1(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(0), \cdot)), \ldots, \hat{I}_{n_I}(\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot)) - \hat{I}_{n_I}(\hat{\boldsymbol{u}}(\boldsymbol{\theta}(0), \cdot))]^{\mathsf{T}}. \tag{5.5}$$

Thus, $\hat{\boldsymbol{m}}(\boldsymbol{\eta}) = 0$ implies that $\hat{\boldsymbol{u}}(\boldsymbol{\eta}, \cdot) \in \hat{\mathcal{M}}_I$. The first-order optimality condition leads to the system of equations

$$\boldsymbol{\eta} = \tilde{\boldsymbol{\theta}}_{k+1} + \hat{\boldsymbol{m}}'(\boldsymbol{\eta})^{\mathsf{T}}\boldsymbol{\lambda}, \tag{5.6a}$$

$$0 = \hat{\boldsymbol{m}}(\boldsymbol{\eta}). \tag{5.6b}$$

We follow [31, Section IV.4] and solve (5.6a)–(5.6b) via the iterations

$$\Delta\boldsymbol{\lambda}^{(i)} = -(\hat{\boldsymbol{m}}'(\tilde{\boldsymbol{\theta}}_{k+1})\hat{\boldsymbol{m}}'(\tilde{\boldsymbol{\theta}}_{k+1})^{\mathsf{T}})^{-1}\hat{\boldsymbol{m}}(\tilde{\boldsymbol{\theta}}_{k+1} + \hat{\boldsymbol{m}}'(\tilde{\boldsymbol{\theta}}_{k+1})^{\mathsf{T}}\boldsymbol{\lambda}^{(i)}), \tag{5.7a}$$

$$\boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \Delta\boldsymbol{\lambda}^{(i)}, \tag{5.7b}$$

for $i = 1, 2, 3, \ldots$. Notice that the iterations are over the quantity $\boldsymbol{\lambda}$, which is of dimension n_I. The number of quantities n_I is often much smaller than the number of parameters $n_{\boldsymbol{\theta}}$.

5.3. Neural Galerkin schemes with embeddings. Algorithm 5.1 summarizes the Neural Galerkin schemes with embeddings when explicit Euler is used to discretize time. We stress that other time integration schemes can be used in an analogous way.

5.3.1. Description of algorithm. The inputs to Algorithm 5.1 are the right-hand side function $\boldsymbol{f}$ of the PDE (2.1), time-step size δt, number of time steps K, parametrization $\hat{\boldsymbol{u}}$, number of sampling points n_S, parameter $\boldsymbol{\theta}_0$ corresponding to the

initial condition $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_0, \cdot)$, the quantities $I_1, \ldots, I_{n_I}$ and the number of sampling points n_M for the sampled quantities $\hat{I}_1, \ldots, \hat{I}_{n_I}$. The algorithm iterates over the time steps $k = 0, \ldots, K-1$. In each iteration, the least-squares problem (5.2) and the sampled gradients (3.7) are assembled. Notice that the sampled $\hat{\boldsymbol{g}}$ is used, which is based on the sampled quantities $\hat{I}_1, \ldots, \hat{I}_{n_I}$ with n_M sampling points. Then, the update $\delta\boldsymbol{\theta}_k$ is computed by solving the least-squares problem (5.3). In line 7, the vector $\tilde{\boldsymbol{\theta}}_{k+1} = \boldsymbol{\theta}_k + \delta t \delta\boldsymbol{\theta}_k$ is projected onto $\boldsymbol{\theta}_{k+1}$ so that the solution field $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_{k+1}, \cdot)$ lies on the constrained manifold. After K time steps, the trajectory of Neural Galerkin approximations $\hat{\boldsymbol{u}}(\boldsymbol{\theta}_0, \cdot), \ldots, \hat{\boldsymbol{u}}(\boldsymbol{\theta}_K)$ is returned.

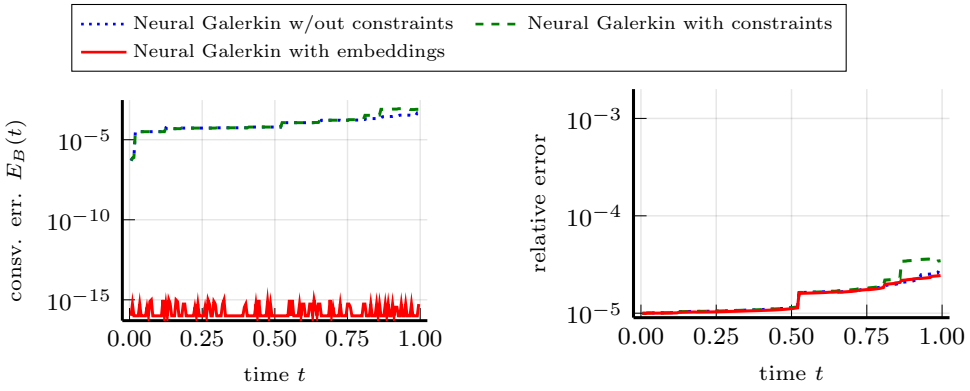
5.3.2. Computational costs. The costs of a time step of Algorithm 5.1 is dominated by estimating the sampled $\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k), \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k), \hat{\boldsymbol{g}}(\boldsymbol{\theta}_k)$ and subsequently solving the least-squares problem (5.3) to obtain $\delta\boldsymbol{\theta}_k$. The costs of computing the sampled $\hat{\boldsymbol{A}}(\boldsymbol{\theta}_k), \hat{\boldsymbol{b}}(\boldsymbol{\theta}_k)$ scale with the number of sampling points n_S. For each sample, the parametrization $\hat{\boldsymbol{u}}$, its gradient $\nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{u}}$, and the right-hand side function $\boldsymbol{f}$ are evaluated. The costs of these evaluations depend on the parametrization. If a fully connected deep network with ℓ layers and p nodes is used, then the costs scale as $\mathcal{O}(\ell p^2)$. The costs of computing $\hat{\boldsymbol{g}}(\boldsymbol{\theta}_k)$ are dominated by the quadrature in (2.4), which depends on the number of sampling points n_M. The functions $\kappa_1, \ldots, \kappa_{n_I}$ that define $I_1, \ldots, I_{n_I}$ as shown in (2.4) are typically cheap to evaluate. The costs of solving the least-squares problem (5.3) scales as $\mathcal{O}(n_S {n_{\boldsymbol{\theta}}}^2)$. In summary, the dominating costs of a time step scale as $\mathcal{O}(n_S \ell p^2 + n_M + n_S {n_{\boldsymbol{\theta}}}^2)$, which provides only a crude estimate as these costs critically depend on the right-hand side function $\boldsymbol{f}$. We refer to [7] for work on reducing the costs per time step. The projection (5.7) computed in line 7 of Algorithm 5.1 typically requires only few iterations and thus incurs negligible costs in our numerical experiments.

6. Numerical experiments. We now demonstrate Neural Galerkin with embeddings with numerical experiments. First, we consider the inviscid Burgers' equation (2.9) to demonstrate the interplay between the constrained Neural Galerkin system and the nonlinear embedding. Second, we illustrate with the acoustic wave equation (2.10) how the number of sampling points n_M influences the preservation of the Hamiltonian (2.10). Third, we consider the shallow water equations in a two-dimensional spatial domain with the total energy as conserved quantity. The implementation is based on `jax`, which is a Python library for automatic differentiation [10]. All floating point computations are carried out in double precision. The implementation is available online github.com/Algopaul/ng_embeddings.

6.1. Experimental setup. In each experiment, we compare approximations to reference solutions $\boldsymbol{u}^{\text{ref}}$ obtained with spectral methods. Time is discretized in all examples with the explicit fourth-order Runge-Kutta method [32, Table 1.2, l]. For a solution trajectory with parameters $\{\boldsymbol{\theta}_k\}_{k=0}^K$, $K \in \mathbb{N}$, at time step $\{t_k\}_{k=0}^K$, we compute the relative error of the solution field as

$$E_r(t_k) = \sum_{i=1}^{n_E} \|\hat{\boldsymbol{u}}(\boldsymbol{\theta}_k, \boldsymbol{x}_i^{\text{test}}) - \boldsymbol{u}^{\text{ref}}(t_k, \boldsymbol{x}_i^{\text{test}})\| \Big/ \sum_{i=1}^{n_E} \|\boldsymbol{u}^{\text{ref}}(t_k, \boldsymbol{x}_i^{\text{test}})\|, \tag{6.1}$$

using n_E equidistantly sampled test points $\boldsymbol{x}_1^{\text{test}}, \ldots, \boldsymbol{x}_{n_E}^{\text{test}}$ in the respective domains. We also report the error in conserving the quantities of interest $t \mapsto \|\hat{I}^{\text{test}}(t) - \hat{I}^{\text{test}}(0)\|$ for the different solution trajectories, where we use n_E equidistantly sampled test points in the respective domain to estimate the quantity, which we denote as $\hat{I}^{\text{test}}$.



(a) conservation error E_B from (6.2)

(b) relative error

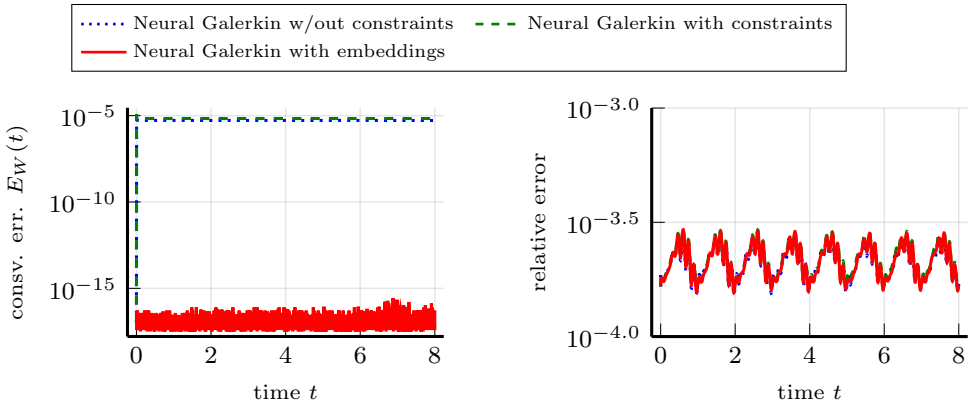
FIG. 3. *Burgers' equation: Nonlinear parametrization such as deep networks used in Neural Galerkin schemes imply that even linear quantities become nonlinear in the parameters. Thus, adding a constraint to the Neural Galerkin scheme is insufficient for conserving quantities with explicit Runge-Kutta schemes. In contrast, the proposed Neural Galerkin scheme with embeddings computes projections onto constrained manifolds at each time step to conserve quantities even in discrete time.*

We ensure that the n_E points used during the evaluation are different from the n_S points used to construct the least-squares problem (5.2) and the n_M points used to estimate the quantities during time integration.

6.2. Burger's equation with conservation of mass. We consider the inviscid Burgers' equation (2.9) in the spatial domain $[-1, 1)$ with periodic boundary conditions and conserved quantity $I_{\text{mass}} = \int_{-1}^{1} u \, \mathrm{d}x$ in the time interval $[0, 1]$. We parametrize the solution field with a fully connected feed-forward deep network that has three hidden layers of width ten with sinusoidal activation functions, which leads to $n_{\boldsymbol{\theta}} = 241$ network weights. The first layer imposes periodicity as in [7]. The output layer is linear. It is important to note that the conserved quantity is linear in the solution function u but nonlinear in the parameter $\boldsymbol{\theta}(t)$ of the parametrization. We use $n_S = 200$ equidistant sample points to construct the least-squares problem (5.1) and $n_M = 200$ equidistant points to estimate the conserved quantity and its derivative (3.7). The time-step size is $\delta t = 5 \cdot 10^{-3}$. We now compare Neural Galerkin without constraints (2.18), Neural Galerkin with constraints (3.6) that conserves quantities in continuous time, and Neural Galerkin with embeddings that is described in Algorithm 5.1 and that combines (3.6) with the embeddings (4.2) to conserve quantities also in discrete time. Figure 3a shows the error in conserving the quantity $\hat{I}^{\text{test}}_{\text{mass}}$, estimated with $n_E = 400$, which we define as

$$E_B(t) = |\hat{I}^{\text{test}}_{\text{mass}}(t) - \hat{I}^{\text{test}}_{\text{mass}}(0)|. \tag{6.2}$$

Because we use a deep network which is a nonlinear parametrization, the quantity is not conserved when just adding a constraint. In contrast, the proposed Neural Galerkin scheme with embeddings conserves the sampled quantity because it performs an explicit embedding after each time step. Figure 3b shows the relative error computed using (6.1) with $n_E = 400$. For the reference solution, we have used 200 Fourier modes and a time step size of 10^{-3}.



(a) error in conserving Hamiltonian

(b) relative error in solution fields

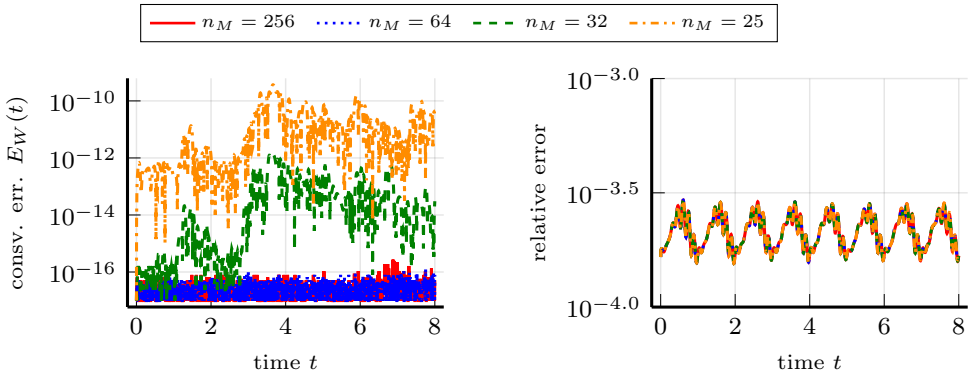
FIG. 4. *Acoustic wave equation: The proposed Neural Galerkin scheme with embeddings preserves the Hamiltonian at test points to machine precision.*

6.3. Acoustic wave equation with conservation of Hamiltonian. We present results for the acoustic wave equation (2.10) with its Hamiltonian H_{wave} as conserved quantity. We have periodic boundary conditions on the spatial domain $\mathcal{X} = [-1, 1)$ with end time $T = 8$ and initial condition $\rho(0, x) = \mathrm{e}^{-9x^2}, v(0, x) = 0$. We set $\rho_{\text{ref}} = c = 1$. The parametrization is a fully connected feed-forward network with two hidden layers of width ten followed by a hidden layer of width 20, each with sinusoidal activation functions, a periodic input layer as in [7] of width ten, and a linear output layer of width two to account for ρ and v, which leads to $n_{\boldsymbol{\theta}} = 392$ network weights. We use $n_S = 256$ sampling points distributed equidistantly in $\mathcal{X}$ to assemble the least-squares problem (5.1) and the same $n_M = 256$ sampling points to estimate the Hamiltonian and its derivative (3.7). We discretize time using fourth-order Runge-Kutta with a time-step size of $2^{-8} \approx 4 \times 10^{-3}$. For the reference solution, we have used 256 Fourier modes and fourth-order explicit Runge-Kutta with a time-step of 10^{-3}. Figure 4a shows the error in conserving the Hamiltonian. We denote the error in $\hat{H}_{\text{wave}}$ by

$$E_W(t) = |\hat{H}^{\text{test}}_{\text{wave}}(t) - \hat{H}^{\text{test}}_{\text{wave}}(0)|, \tag{6.3}$$

where we set $n_E = 512$. The results show that our scheme based on embeddings preserves the sampled Hamiltonian at test points whereas the other schemes lead to solutions with large variations in the Hamiltonian. Figure 4b shows the relative error (6.1) estimated with $n_E = 512$ samples. The relative error of all three schemes is comparable in this example. In Figure 5, we demonstrate that we can use fewer samples n_M for the estimation of $\hat{H}_{\text{wave}}$ during the time integration and still achieve conservation at test points $\hat{H}^{\text{test}}_{\text{wave}}$ to machine precision. In fact, in Figure 5, it is shown the sampled Hamiltonian at the test points is conserved to machine precision even when choosing $n_M = 64$ and still to 10^{-10} for $n_M = 25$.

6.4. Shallow water waves in two spatial dimensions with energy conservation. We now consider the shallow water equations in a two-dimensional domain with periodic boundary conditions. We follow a similar setup as the one introduced in [35, Section 8].



(a) error in conserving Hamiltonian

(b) relative error in solution field

FIG. 5. *Acoustic wave equation: The proposed Neural Galerkin scheme with embeddings preserves the Hamiltonian at a fine grid of test points even when a low number of sampling points n_M is used.*

6.4.1. Setup. The governing equations are

$$\begin{aligned} \partial_t h + \nabla \cdot (h \nabla \phi) &= 0 \quad \text{in } [0,6] \times \mathcal{X}, \\ \partial_t \phi + \frac{1}{2} \|\nabla \phi\|^2 + h &= 0 \quad \text{in } [0,6] \times \mathcal{X}, \end{aligned} \tag{6.4}$$

with initial conditions

$$\begin{aligned} h(0, \boldsymbol{x}) &= 1 + 0.33 \mathrm{e}^{-1.7\|\boldsymbol{x}\|^2} \quad \text{in } \mathcal{X}, \\ \phi(0, \boldsymbol{x}) &= 0 \quad \text{in } \mathcal{X}, \end{aligned} \tag{6.5}$$

where $\mathcal{X} = [-4, 4)^2$, and $h, \phi : [0,6] \times \mathcal{X} \to \mathbb{R}$ denote the height and the potential field of the fluid. A conserved quantity of (6.4) is the energy

$$I_{(\mathrm{e})}(h, \phi) = \frac{1}{2} \int_{\mathcal{X}} h \|\nabla \phi\|^2 + h^2 \mathrm{d}\boldsymbol{x}. \tag{6.6}$$

We scale the equations above such that h and ϕ are close in magnitude and centered around 0 to avoid numerical issues,

$$\begin{aligned} \partial_t \widetilde{h} + \nabla \cdot ((\widetilde{h} + 1) \nabla \phi) &= 0, \quad \text{in } [0,6] \times \mathcal{X}, \\ \partial_t \widetilde{\phi} + \frac{1}{2} \left\| \nabla \widetilde{\phi} \right\|^2 + \widetilde{h} &= 0, \quad \text{in } [0,6] \times \mathcal{X}\,. \end{aligned} \tag{6.7}$$

The corresponding initial conditions are

$$\begin{aligned} \widetilde{h}(0, \boldsymbol{x}) &= 0.33 \mathrm{e}^{-1.7\|\boldsymbol{x}\|^2}, \quad \text{in } \mathcal{X}, \\ \widetilde{\phi}(0, \boldsymbol{x}) &= 0, \quad \text{in } \mathcal{X}, \end{aligned} \tag{6.8}$$

where $\widetilde{h} = h - 1$ and $\widetilde{\phi} = \phi + t$. Note that in the second equation, we did not add the constant 1 to prevent $\widetilde{\phi}$ from diverging numerically. Our reference solution is obtained with a spectral method with 300 degrees of freedom in each spatial dimension and a time-step size 10^{-3}.

6.4.2. Results. The parametrization is a deep network with the same structure as in Section 6.3, which leads to $n_{\boldsymbol{\theta}} = 402$ network weights for the two-dimensional input domain. The time-step size is $\delta t = 2 \times 10^{-3}$ and the time discretization scheme is fourth-order explicit Runge-Kutta. We take 200 points in each spatial direction to assemble the least-squares problem (5.1) and for estimating the conserved quantity (6.6) and its derivative (3.7). We compare Neural Galerkin without constraints, Neural Galerkin with constraints, and the proposed scheme with embeddings. Figure 6 and 7 show the solution field and the point-wise error of the fluid height h at times $t = 5$ and $t = 6$, respectively. The approximation obtained with embeddings avoids the oscillations that are present in the approximations obtained with other schemes that ignore conservation of quantities. The error in conserving the energy is shown in Figure 8a, which we measure as $E_S(t) = |\hat{I}^{\text{test}}_{(e)}(t) - \hat{I}^{\text{test}}_{(e)}(0)|$, for $n_E = 90\,000$ equidistantly sampled test points. Figure 8b shows the relative error (6.1) with $n_E = 90\,000$ test samples. The Neural Galerkin scheme with embeddings achieves the lowest relative error in the solution fields, which is in agreement with the plots shown in Figure 6 and 7 where the variants without energy conservation lead to oscillations in the approximate solution fields. In Figure 9 we study the effect of the the number of samples n_M used to estimate the energy. The results show that with 100 points in each direction so that $n_M = 10\,000$, the Neural Galerkin scheme with embeddings conserves the energy at the test points up to machine precision. The error in the energy conservation grows to about 10^{-5} only when there are only 25 sampling points in each spatial direction, which corresponds to a total of $n_M = 625$ points. Figure 9b shows how the relative error (6.1) in the solution fields depends on the number of sampling points n_M. It can be seen that conserving the energy more accurately with more sampling points leads to lower relative errors in the solution fields in this example. If only 25 sampling points in each spatial dimension are used, then the error increases by about one order of magnitude compared to when the energy is conserved to machine precision.

7. Conclusions. Preserving structure and conserving quantities in nonlinear approximations of PDE solutions is delicate because the nonlinear dependence on the parameter implies that even linear quantities in the solution fields can become nonlinear in the parameters. Thus, just adding constraints to time-continuous formulations is insufficient with standard Runge-Kutta time integrators. While one can resort to implicit methods such as discrete gradients, it is desirable to have explicit time integration schemes when nonlinear parametrizations are used because they require solving systems of linear equations at each time step in typical cases rather than systems of nonlinear equations as with implicit integrators. The proposed Neural Galerkin schemes compute explicit embeddings on manifolds of parametrizations that conserve quantities, which can be computed efficiently and are applicable with explicit time integrators and generic nonlinear parametrizations such as deep networks.

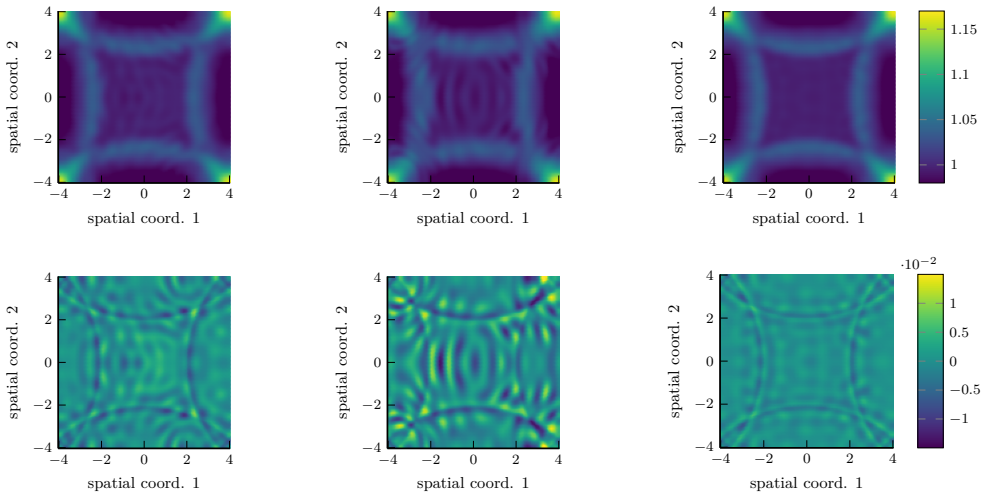
REFERENCES

[1] B. M. Afkham and J. S. Hesthaven, *Structure preserving model reduction of parametric Hamiltonian systems*, SIAM J. Sci. Comput., 39 (2017), pp. A2616–A2644.

[2] B. M. Afkham, N. Ripamonti, Q. Wang, and J. S. Hesthaven, *Conservative Model Order Reduction for Fluid Flow*, Springer International Publishing, Cham, 2020, pp. 67–99.

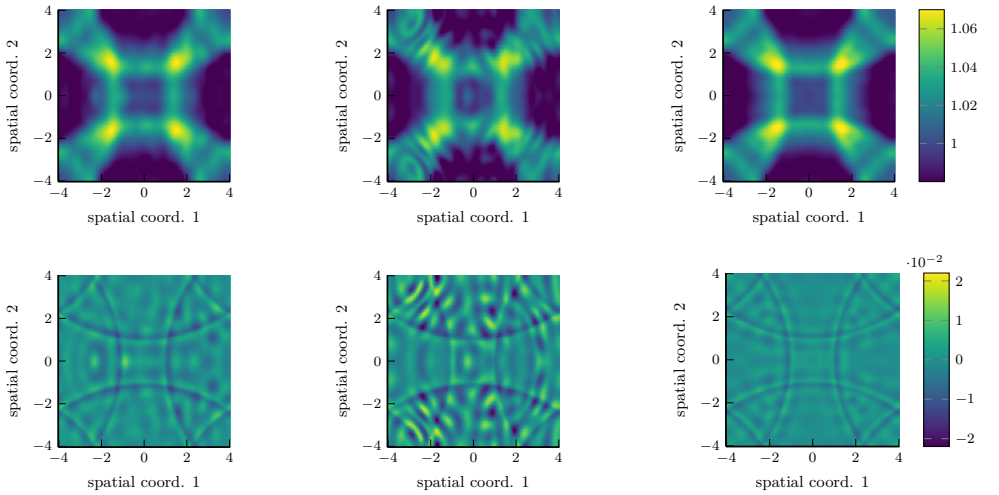
[3] W. Anderson and M. Farazmand, *Evolution of nonlinear reduced-order solutions for PDEs with conserved quantities*, SIAM J. on Sci. Comput., 44 (2022), pp. A176–A197.

[4] P. J. Baddoo, B. Herrmann, B. J. McKeon, N. J. Kutz, and S. L. Brunton, *Physics-informed dynamic mode decomposition*, Proc. R. Soc. A: Math. Phys. Eng. Sci., 479 (2023),



(a) Neural Galerkin (b) Neural Galerkin with constraints (c) Neural Galerkin with embeddings

FIG. 6. *Shallow water: The proposed Neural Galerkin scheme with embeddings conserves energy over time and avoids oscillations in the solution field, which represents the height of the fluid. Time is* $t = 5$.



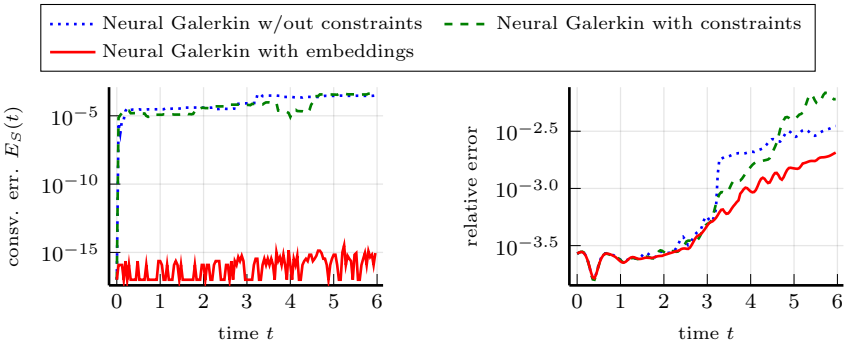
(a) Neural Galerkin (b) Neural Galerkin with constraints (c) Neural Galerkin with embeddings

FIG. 7. *Shallow water: The plots of the solution field of* h *obtained with Neural Galerkin and embeddings indicates that the conservation of energy avoids the oscillations that are present in the approximations obtained with the other schemes that ignore energy conservation. Time is* $t = 6$.

p. 20220576.

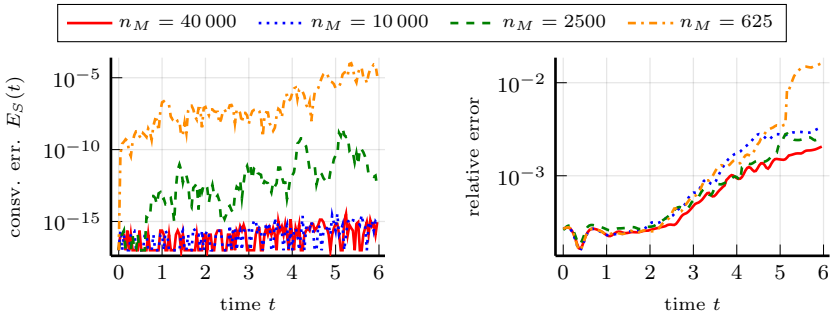
[5] M. BALAJEWICZ, I. TEZAUR, AND E. DOWELL, *Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations*, Journal of Computational Physics, 321 (2016), pp. 224–241.

[6] M. F. BARONE, I. KALASHNIKOVA, D. J. SEGALMAN, AND H. K. THORNQUIST, *Stable Galerkin reduced order models for linearized compressible flow*, J. Comput. Phys., 228 (2009),



(a) error in conserving energy (b) relative error in solution fields

FIG. 8. *Shallow water: The proposed Neural Galerkin scheme with embeddings conserves the sampled energy at test points up to machine precision.*



(a) error in conserving energy (b) relative error in solution fields

FIG. 9. *Shallow water: The energy is conserved up to machine precision at the test points when* $n_M = 10\,000$ *samples (100 in each spatial direction) are used. The error increases up to* 10^{-5} *when only* $n_M = 625$ *samples are used (25 in each direction).*

pp. 1932–1946.

[7] J. BERMAN AND B. PEHERSTORFER, *Randomized sparse Neural Galerkin schemes for solving evolution equations with deep networks*, in Advances in Neural Information Processing Systems, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds., vol. 36, Curran Associates, Inc., 2023, pp. 4097–4114.

[8] Å. BJÖRCK, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics, 1996.

[9] W. M. BOON AND A. FUMAGALLI, *A reduced basis method for Darcy flow systems that ensures local mass conservation by using exact discrete complexes*, J. Sci. Comput., 94 (2023), p. 64.

[10] J. BRADBURY, R. FROSTIG, P. HAWKINS, M. J. JOHNSON, C. LEARY, D. MACLAURIN, G. NECULA, A. PASZKE, J. VANDERPLAS, S. WANDERMAN-MILNE, AND Q. ZHANG, *JAX: composable transformations of Python+NumPy programs*, 2018, http://github.com/google/jax.

[11] J. BRUNA, B. PEHERSTORFER, AND E. VANDEN-EIJNDEN, *Neural Galerkin scheme with active learning for high-dimensional evolution equations*, arXiv, 2203.01360 (2022).

[12] P. BUCHFINK, A. BHATT, AND B. HAASDONK, *Symplectic model order reduction with non-orthonormal bases*, Mathematical and Computational Applications, 24 (2019).

[13] P. BUCHFINK, S. GLAS, AND B. HAASDONK, *Symplectic model reduction of Hamiltonian systems on nonlinear manifolds and approximation with weakly symplectic autoencoder*, SIAM Journal on Scientific Computing, 45 (2023), pp. A289–A311.

[14] K. CARLBERG, Y. CHOI, AND S. SARGSYAN, *Conservative model reduction for finite-volume models*, J. Comput. Phys., 371 (2018), pp. 280–314.

[15] J. Chan, *Entropy stable reduced order modeling of nonlinear conservation laws*, J. Comput. Phys., 423 (2020), p. 109789.
[16] S. Chaturantabut, C. Beattie, and S. Gugercin, *Structure-preserving model reduction for nonlinear port-Hamiltonian systems*, SIAM Journal on Scientific Computing, 38 (2016), pp. B837–B865.
[17] Z. Chen, J. Zhang, M. Arjovsky, and L. Bottou, *Symplectic recurrent neural networks*, in International Conference on Learning Representations, 2020.
[18] G. J. Cooper, *Stability of Runge-Kutta methods for trajectory problems*, IMA J. Numer. Anal., 7 (1987), pp. 1–13.
[19] P. A. M. Dirac, *Note on exchange phenomena in the Thomas atom*, Mathematical Proceedings of the Cambridge Philosophical Society, 26 (1930), pp. 376–385.
[20] L. Einkemmer, J. Hu, and Y. Wang, *An asymptotic-preserving dynamical low-rank method for the multi-scale multi-dimensional linear transport equation*, Journal of Computational Physics, 439 (2021), p. 110353.
[21] L. Einkemmer, A. Ostermann, and C. Scalone, *A robust and conservative dynamical low-rank algorithm*, Journal of Computational Physics, 484 (2023), p. 112060.
[22] C. Farhat, T. Chapman, and P. Avery, *Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models*, International Journal for Numerical Methods in Engineering, 102 (2015), pp. 1077–1110.
[23] J. Frenkel, *Wave Mechanics, Advanced General Theory*, Clarendon Press, Oxford, 1934.
[24] Y. Gong, Q. Wang, and Z. Wang, *Structure-preserving Galerkin POD reduced-order modeling of Hamiltonian systems*, Computer Methods in Applied Mechanics and Engineering, 315 (2017), pp. 780–798.
[25] O. Gonzalez, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.
[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. http://www.deeplearningbook.org.
[27] I. V. Gosea, S. Gugercin, and S. W. R. Werner, *Structured barycentric forms for interpolation-based data-driven reduced modeling of second-order systems*, ArXiv preprint 2303.12576v1, 2023.
[28] P. Goyal, B. Peherstorfer, and P. Benner, *Rank-minimizing and structured model inference*, SIAM Journal on Scientific Computing, (2024). (accepted).
[29] A. Gruber and I. Tezaur, *Canonical and noncanonical Hamiltonian operator inference*, ArXiv preprint 2304.06262v2, 2023.
[30] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, Springer Berlin Heidelberg, 1996.
[31] E. Hairer, G. Wanner, and C. Lubich, *Geometric Numerical Integration, Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer Berlin, Heidelberg, 2006.
[32] E. Hairer, G. Wanner, and S. P. Nørsett, *Solving Ordinary Differential Equations I*, Springer Berlin Heidelberg, 1993.
[33] J. Han, A. Jentzen, and W. E, *Solving high-dimensional partial differential equations using deep learning*, Proceedings of the National Academy of Sciences, 115 (2018), pp. 8505–8510.
[34] J. S. Hesthaven and C. Pagliantini, *Structure-preserving reduced basis methods for Poisson systems*, Math. Comp., 90 (2021), pp. 1701–1740.
[35] J. S. Hesthaven, C. Pagliantini, and N. Ripamonti, *Rank-adaptive structure-preserving model order reduction of Hamiltonian systems*, ESAIM: M2AN, 56 (2022), pp. 617–650.
[36] I. Kalashnikova, B. van Bloemen Waanders, S. Arunajatesan, and M. Barone, *Stabilization of projection-based reduced order models for linear time-invariant systems via optimization-based eigenvalue reassignment*, Computer Methods in Applied Mechanics and Engineering, 272 (2014), pp. 251–270.
[37] P. Kramer and M. Saraceno, *Geometry of the time-dependent variational principle in quantum mechanics*, in Lecture Notes in Physics, vol. 140, Springer, 1981.
[38] P. Kunkel and V. Mehrmann, *Differential-Algebraic Equations: Analysis and Numerical Solution*, EMS Publishing House, Zürich, 2006.
[39] S. Lall, P. Krysl, and J. E. Marsden, *Structure-preserving model reduction for mechanical systems*, vol. 184, 2003, pp. 304–318. Complexity and nonlinearity in physical systems (Tucson, AZ, 2001).
[40] C. Lasser and C. Lubich, *Computing quantum dynamics in the semiclassical regime*, Acta Numerica, 29 (2020), pp. 229–401.
[41] K. Lee and K. T. Carlberg, *Deep conservation: A latent-dynamics model for exact satis-*

faction of physical conservation laws, Proceedings of the AAAI Conference on Artificial Intelligence, 35 (2021), pp. 277–285.

[42] B. Liljegren-Sailer and N. Marheineke, *On snapshot-based model reduction under compatibility conditions for a nonlinear flow problem on networks*, J. Sci. Comput., 92 (2022), p. 62.

[43] C. Lubich, *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*, EMS Press, 2008.

[44] R. I. McLachlan, G. R. W. Quispel, and N. Robidoux, *Geometric integration using discrete gradients*, Phil. Trans. R. Soc. Lond., 357 (1999), pp. 1021–1045.

[45] V. Mehrmann and R. Morandin, *Structure-preserving discretization for port-Hamiltonian descriptor systems*, in Proceedings of the 58th IEEE Conference on Decision and Control, Nice, France, 2019, pp. 6863–6868.

[46] M. Mohebujjaman, L. G. Rebholz, and T. Iliescu, *Physically constrained data-driven correction for reduced-order modeling of fluid flows*, Int. J. Numer. Methods Fluids, 89 (2019), pp. 103–122.

[47] R. Morandin, J. Nicodemus, and B. Unger, *Port-Hamiltonian dynamic mode decomposition*, SIAM J. Sci. Comput., 45 (2023), pp. A1690–A1710.

[48] E. Musharbash, F. Nobile, and E. Vidličková, *Symplectic dynamical low rank approximation of wave equations with random parameters*, BIT, 60 (2020), pp. 1153–1201.

[49] P. J. Olver, *Applications of Lie Groups to Differential Equations*, Springer New York, USA, second ed., 1993.

[50] K. Otness, A. Gjoka, J. Bruna, D. Panozzo, B. Peherstorfer, T. Schneider, and D. Zorin, *An extensible benchmark suite for learning to simulate physical systems*, in Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1), 2021.

[51] C. Pagliantini, *Dynamical reduced basis methods for Hamiltonian systems*, Numerische Mathematik, 148 (2021), pp. 409–448.

[52] B. Peherstorfer, *Sampling low-dimensional Markovian dynamics for pre-asymptotically recovering reduced models from data with operator inference*, SIAM Journal on Scientific Computing, 42 (2020), pp. A3489–A3515.

[53] B. Peherstorfer, *Breaking the Kolmogorov barrier with nonlinear model reduction*, Notices of the American Mathematical Society, 69 (2022), pp. 725–733.

[54] L. Peng and K. Mohseni, *Symplectic model reduction of Hamiltonian systems*, SIAM J. Sci. Comput., 38 (2016), pp. A1–A27.

[55] I. Pontes Duff, P. Goyal, and P. Benner, *Data-driven identification of Rayleigh-damped second-order systems*, in Realization and Model Reduction of Dynamical Systems, C. Beattie, P. Benner, M. Embree, S. Gugercin, and S. Lefteriu, eds., Springer, Cham, Switzerland, 2022, pp. 255–272.

[56] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, *Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems*, Physica D: Nonlinear Phenomena, Volume 406 (2020).

[57] H. K. E. Rosenberger and B. Sanderse, *Momentum-conserving ROMs for the incompressible Navier-Stokes equations*, arXiv, (2022).

[58] C. W. Rowley, T. Colonius, and R. M. Murray, *Model reduction for compressible flows using POD and Galerkin projection*, Physica D: Nonlinear Phenomena, 189 (2004), pp. 115–129.

[59] B. Sanderse, *Non-linearly stable reduced-order models for incompressible flow with energy-conserving finite volume methods*, J. Comput. Phys., 421 (2020), p. 109736.

[60] N. Sawant, B. Kramer, and B. Peherstorfer, *Physics-informed regularization and structure preservation for learning stable reduced models from data with operator inference*, Comput. Methods Appl. Mech. Engrg., 404 (2023), p. 115836.

[61] P. Schulze, *Energy-based model reduction of transport-dominated phenomena*, PhD thesis, Technische Universität Berlin, Germany, 2023.

[62] P. Schulze, *Structure-preserving model reduction for port-Hamiltonian systems based on separable nonlinear approximation ansatzes*, Front. Appl. Math. Stat., 9 (2023), p. 1160250.

[63] P. Schulze and B. Unger, *Data-driven interpolation of dynamical systems with delay*, Systems Control Lett., 97 (2016), pp. 125–131.

[64] P. Schulze, B. Unger, C. Beattie, and S. Gugercin, *Data-driven structured realization*, Linear Algebra Appl., 537 (2018), pp. 250–286.

[65] P. Schwerdtner and M. Voigt, *Sobmor: Structured optimization-based model order reduction*, SIAM J. Sci. Comput., 45 (2023), pp. A502–A529.

[66] L. Shampine, *Conservation laws and the numerical solution of ODEs*, Computers & Mathe-

matics with Applications, 12 (1986), pp. 1287–1296.

[67] H. Sharma, H. Mu, P. Buchfink, R. Geelen, S. Glas, and B. Kramer, *Symplectic model reduction of Hamiltonian systems using data-driven quadratic manifolds*, ArXiv preprint 2305.15490v2, 2023.

[68] H. Sharma, Z. Wang, and B. Kramer, *Hamiltonian operator inference: Physics-preserving learning of reduced-order models for canonical Hamiltonian systems*, Phys. D, 431 (2022), p. 133122.

[69] Y. Wen, E. Vanden-Eijnden, and B. Peherstorfer, *Coupling parameter and particle dynamics for adaptive sampling in Neural Galerkin schemes*, Physica D: Nonlinear Phenomena, 462 (2024), p. 134129.

[70] X. Zhang, T. Cheng, and L. Ju, *Implicit form neural network for learning scalar hyperbolic conservation laws*, in Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference, PMLR, 2022, pp. 1082–1098.