Journal of Computational Physics 498 (2024) 112649

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Fast and scalable computation of shape-morphing nonlinear solutions with application to evolutional neural networks

William Anderson, Mohammad Farazmand *

Department of Mathematics, North Carolina State University, 2311 Stinson Drive, Raleigh, NC, 27695-8205, USA

ARTICLE INFO

Keywords:
Partial differential equations
Reduced-order modeling
Spectral methods
Neural networks

ABSTRACT

We develop fast and scalable methods for computing reduced-order nonlinear solutions (RONS). RONS was recently proposed as a framework for reduced-order modeling of time-dependent partial differential equations (PDEs), where the modes depend nonlinearly on a set of time-varying parameters. RONS uses a set of ordinary differential equations (ODEs) for the parameters to optimally evolve the shape of the modes to adapt to the PDE's solution. This method has already proven extremely effective in tackling challenging problems such as advection-dominated flows and high-dimensional PDEs. However, as the number of parameters grow, integrating the RONS equation and even its formation become computationally prohibitive. Here, we develop three separate methods to address these computational bottlenecks: symbolic RONS, collocation RONS and regularized RONS. We demonstrate the efficacy of these methods on two examples: Fokker–Planck equation in high dimensions and the Kuramoto–Sivashinsky equation. In both cases, we observe that the proposed methods lead to several orders of magnitude in speedup and accuracy. Our proposed methods extend the applicability of RONS beyond reduced-order modeling by making it possible to use RONS for accurate numerical solution of linear and nonlinear PDEs. Finally, as a special case of RONS, we discuss its application to problems where the PDE's solution is approximated by a neural network, with the time-dependent parameters being the weights and biases of the network. The RONS equations dictate the optimal evolution of the network's parameters without requiring any training.

1. Introduction

Anderson and Farazmand [1] recently proposed reduced-order nonlinear solutions (RONS) as a new framework for deriving reduced-order models for time-dependent PDEs. RONS considers shape-morphing approximate solutions,

$$\hat{u}(\mathbf{x}, \mathbf{q}(t)) = \sum_{i=1}^{r} \alpha_i(t) u_i(\mathbf{x}, \boldsymbol{\beta}_i(t)), \tag{1}$$

to the PDE which depend nonlinearly on time-varying parameters $\mathbf{q}(t) = \{\alpha_i(t), \boldsymbol{\beta}_i(t)\}_{i=1}^{r}$. This is in contrast to most reduced-order models which consider approximate solutions $\hat{u}(\mathbf{x}, \mathbf{q}(t)) = \sum_i q_i(t) u_i(\mathbf{x})$ as a linear combination of time-independent modes u_i (see [2,3], for reviews). By allowing nonlinear dependence on the parameters, RONS significantly expands the scope of reduced-order

* Corresponding author.
E-mail addresses: wmander3@ncsu.edu (W. Anderson), farazmand@ncsu.edu (M. Farazmand).

https://doi.org/10.1016/j.jcp.2023.112649
Received 21 February 2023; Received in revised form 31 October 2023; Accepted 17 November 2023

Available online 22 November 2023

modeling, resulting in more accurate reduced models capable of tackling challenging problems such as advection-dominated dynamics.

We note that RONS should not be confused with parametric model reduction [2]. In our framework, the reduced-order solution depends on time-varying parameters which are distinct from the governing PDE's parameters such as viscosity coefficients, domain size, etc.

As we review in Section 2, RONS uses a set of ordinary differential equations (ODEs) for the optimal evolution of parameters $\mathbf{q}(t) \in \mathbb{R}^n$ by minimizing the instantaneous error between the dynamics of the reduced-order solution $\hat{u}(\mathbf{x}, \mathbf{q}(t))$ and the true dynamics of the PDE. Furthermore, RONS ensures that the resulting reduced-order model preserves conserved quantities of the PDE.

There are two main computational bottlenecks that may adversely affect the performance of RONS. The main computational cost of RONS comes from evaluating the functional inner products which are required to form the reduced-order equations. More specifically, in order to form the RONS reduced-order equations, $\mathcal{O}(n^2)$ inner products must be evaluated, where n is the number of parameters. Making matters worse, these inner products need to be reevaluated at each time step as the parameter values evolve. To compute these inner products, one can use quadrature, Monte Carlo integration, or symbolic computing. As the number of parameters n grows, all these methods become quickly prohibitive. The second computational cost arises from the stiffness of the RONS equations. As mentioned earlier, RONS equations are a set of ODEs for the evolution of parameters $\mathbf{q}(t)$. As the number of parameters increases, these ODEs can become stiff and therefore very slow to solve using explicit time integration.

In this paper, we develop three separate methods to address these computational bottlenecks, and thus drastically reduce the computational cost of RONS. For the first method, which we call symbolic RONS, we assume that the inner products can be computed symbolically. Exploiting the hidden structure of RONS equations, we reduce the number of required inner product computations to $\mathcal{O}(K^2)$ where $K \ll n$ is an integer independent of n. Furthermore, because this method uses symbolic computation, the inner products do not need to be recomputed during time stepping. As a result, the computational cost remains low even when the number of parameters n is very large. This scalability allows us to go beyond reduced-order modeling and use RONS as a spectral method where the modes (or basis functions) evolve over time through their nonlinear dependence on time-dependent parameters.

The second method, which we refer to as collocation RONS, introduces a collocation point version of RONS in case symbolic computations are not feasible. This method minimizes the discrepancy between the RONS dynamics and the governing PDE on a set of prescribed collocation points. Collocation RONS is applicable to general nonlinear PDEs, does not require inner product evaluations, and therefore does not require any symbolic computation. Furthermore, we show that using Monte Carlo integration to approximate the RONS equations coincides with solving a least squares problem which arises from our collocation point method. However, the system of equations arising from collocation RONS is significantly better conditioned than the Monte Carlo approach, and therefore numerically more stable.

Our third contribution addresses the stiffness of the RONS equations. When the number of model parameters is large, the RONS ODEs can become stiff and therefore slow to solve using explicit time integration schemes. To address this issue, we introduce a regularized version of RONS which is applicable to both symbolic RONS and collocation RONS. Regularized RONS introduces a Tikhonov penalization to the underlying minimization problem. This regularization significantly speeds up the numerical time integration of the RONS equations while insignificantly affecting the accuracy of the solutions.

These three methods expand the scope of RONS beyond reduced-order modeling. Specifically, they allow us to use a large number of modes r to accurately solve PDEs using shape-morphing modes. Being mesh-free, RONS is applicable to PDEs in high spatial dimensions as we demonstrate in section 4.1.

1.1. Related work

Before RONS [1], several previous studies had already considered nonlinear shape-morphing approximate solutions $\hat{u}(\mathbf{x}, \mathbf{q}(t))$ for specific PDEs. For instance, to build reduced-order models for the nonlinear Schrödinger (NLS) equation, several authors have proposed approximating wave packets with either Gaussian or hyperbolic secant envelopes [4–9]. The amplitude, width, and center of the wave packet are controlled by parameters that evolve over time. Refs. [7–9] use the variational Lagrangian formulation of NLS to obtain a set of ODEs for evolving these parameters. As an alternative approach, Adcock et al. [4,5] use the symmetries of NLS to evolve the parameters. Another example appears in fluid dynamics where vortex methods approximate the fluid flow as a superposition of point vortices [10], or their smooth approximations [11,12]. The position, strength, and shape of the vortices are then evolved based on the induced velocity of other vortices.

Although the idea of shape-morphing approximate solutions has been around for decades, the evolution of their shape parameters were determined using ad hoc methods on a case by case basis. RONS proposed a unified framework for evolving these parameters which is applicable to a broad range of PDEs, without relying on the variational structure or symmetries of the PDE.

Interestingly, in the context of evolutional deep neural networks (EDNNs), Du and Zaki [13] simultaneously and independently derived a set of evolution equations similar to RONS [1]. EDNNs approximate solutions of PDEs by evolving weights and biases of a deep neural network over time. Since the network's activation functions are nonlinear, an EDNN depends nonlinearly on its parameters, i.e., weights and biases. As such, EDNNs are a special case of reduced-order nonlinear solutions. Therefore, it is not surprising that the EDNN equations are similar to RONS.

In spite of this similarity, there are some notable differences between EDNN and RONS. Namely, RONS ensures that the reduced-order model respects the conserved quantities of the PDE. EDNN does not guarantee these conservation laws, although they can be easily enforced following the methodology introduced in [1]. On the other hand, Du and Zaki [13] show how various boundary

conditions of the PDE can be embedded into the EDNN framework, an important contribution which was not considered in the development of RONS.

As in RONS, forming the EDNN equations requires the evaluation of certain functional inner products. Du and Zaki [13] approximate these inner products using Monte Carlo integration with uniform sampling. Bruna et al. [14] proposed an adaptive sampling method to estimate the integrals. Their adaptive samples are drawn from a distribution which depends on the approximate solution at any given time. They show that, for PDEs whose solutions are localized in space, adaptive sampling results in more accurate solutions than uniform sampling.

As mentioned earlier, an important feature of RONS is its ability to ensure that the approximate solutions preserve the conserved quantities of the original PDE. There are many studies which consider the same objective; however, the resulting methods are only applicable to a special class of governing equations. For instance, symplectic integrators are specifically designed to preserve the two-form associated with a Hamiltonian system [15–17]. Similarly, Peng and Mohseni [18] developed proper symplectic decomposition (PSD) for Hamiltonian systems to ensures their reduced-order models preserve the Hamiltonian structure of the full-order model. Carlberg et al. [19] propose a finite-volume based method which guarantees preservation of conserved quantities in the reduced model. This method is only applicable to PDEs derived from conservation laws, and hence amenable to finite-volume discretization. In contrast, RONS preserves any finite number of conserved quantities of the PDE without making any restricting assumptions on the structure of the PDE.

Finally, we point out that the method of optimally time-dependent (OTD) modes [20–23] uses an expansion similar to Eq. (1). However, OTD is only applicable to stability analysis of linear or linearized PDEs. In contrast, RONS is applicable for reduced-order modeling and numerical approximation of general nonlinear PDEs.

1.2. Outline

This paper is organized as follows. In section 2, we briefly review the derivation of RONS and its relation to Galerkin-type methods. Section 3 contains our main theoretical results where we develop fast and scalable methods for constructing and solving the RONS equations. Section 4 contains numerical results demonstrating the application of the proposed methods to two different PDEs. We present our concluding remarks in section 5.

2. Set-up and preliminaries

In this section, we present a succinct review of RONS. We refer to Ref. [1] for a more detailed discussion. RONS builds reduced-order models for PDEs of the general form

$$\frac{\partial u}{\partial t} = F(u), \quad u(\mathbf{x},0) = u_0(\mathbf{x}), \tag{2}$$

where $u : D\times\mathbb{R}^+ \to \mathbb{R}^p, (\mathbf{x},t) \mapsto u(\mathbf{x},t)$ is the solution of the PDE, $D \subseteq \mathbb{R}^d$ is the spatial domain, F is a potentially nonlinear differential operator, and u_0 is the initial condition. We assume the solution $u(\cdot,t)$ belongs to a Hilbert space H with the inner product $\langle\cdot,\cdot\rangle_H$ and the induced norm $\|\cdot\|_H$. To simplify the exposition, we assume $p = 1$ hereafter, i.e., $u(\mathbf{x},t) \in \mathbb{R}$. Generalization to $p > 1$ and to complex-valued functions is straightforward [24].

We consider *shape-morphing approximate solutions* $\hat{u}(\mathbf{x},\mathbf{q}(t))$ which depend nonlinearly on a set of time-dependent parameters $\mathbf{q}(t) \in \mathbb{R}^n$. RONS prescribes a set of ODEs to evolve the parameters $\mathbf{q}(t)$ such that the instantaneous error between dynamics of the reduced-order solution $\hat{u}$ and dynamics of the true PDE is minimized. The instantaneous error is defined by

$$\mathcal{J}(\mathbf{q},\dot{\mathbf{q}}) = \frac{1}{2}\|\hat{u}_t - F(\hat{u})\|_H^2, \tag{3}$$

which measures the difference between the rate of change of the approximate solution $\hat{u}_t$ and the rate of change $F(\hat{u})$ dictated by the PDE. Here $\hat{u}_t$ is shorthand for

$$\hat{u}_t(\mathbf{x},\mathbf{q}(t)) = \sum_{i=1}^{n} \frac{\partial \hat{u}}{\partial q_i}(\mathbf{x},\mathbf{q}(t))\dot{q}_i. \tag{4}$$

If the PDE has no conserved quantity, the reduced-order equations are obtained by minimizing (3). However, let's consider the more general case where the PDE has m conserved quantities, $I_k : H \to \mathbb{R}$ with $k \in \{1,2,\cdots,m\}$. Since these quantities are conserved, they must satisfy $I_k(u(\cdot,t)) = I_k(u_0)$ for all $t \geq 0$. It is desirable for the reduced-order model to also preserve these conserved quantities, since otherwise the reduced model may exhibit unphysical behavior [18,25].

To obtain an evolution equation for the parameters $\mathbf{q}(t)$, we solve the constraint optimization problem

$$\begin{aligned} &\min_{\dot{\mathbf{q}}\in\mathbb{R}^n} \mathcal{J}(\mathbf{q},\dot{\mathbf{q}}), \\ &\text{subject to} \quad I_k(\mathbf{q}(t)) = I_k(\mathbf{q}(0)), \quad k = 1,2,...,m, \quad \forall t \geq 0, \end{aligned} \tag{5}$$

where $I_k(\mathbf{q}(t))$ is shorthand for $I_k(\hat{u}(\cdot,\mathbf{q}(t)))$. As shown in [1], the solution to this minimization problem is

$$M(\mathbf{q})\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}) - \sum_{k=1}^{m} \lambda_k \nabla I_k(\mathbf{q}), \tag{6}$$

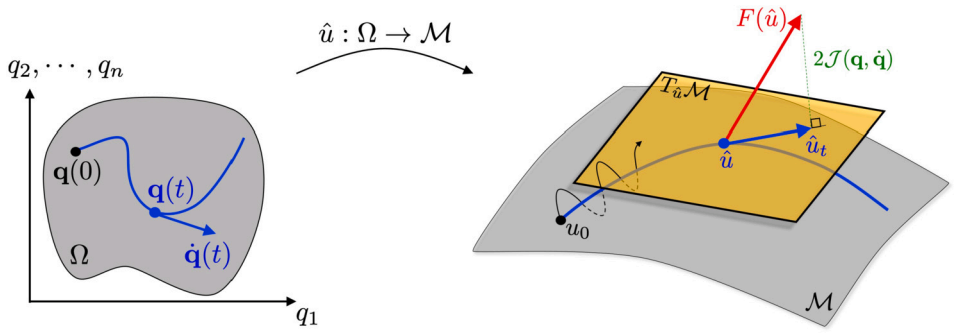

Fig. 1. Geometric illustration of RONS. The shape-morphing approximate solution $\hat{u}$ maps the parameter space Ω to the manifold $\mathcal{M}$. An evolution of the parameters $\mathbf{q}(t)$ is mapped to a curve on $\mathcal{M}$, and the tangent vector to the parameters $\dot{\mathbf{q}}(t)$ is mapped to $\hat{u}_t$ in the tangent space of the manifold.

which we refer to as the RONS equation. Here $M(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite *metric tensor* defined by

$$M_{ij} = \left\langle \frac{\partial \hat{u}}{\partial q_i}, \frac{\partial \hat{u}}{\partial q_j} \right\rangle_H, \quad i,j \in \{1,2,\cdots,n\}. \tag{7}$$

The entries of the right-hand side vector field $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ are given by

$$f_i = \left\langle \frac{\partial \hat{u}}{\partial q_i}, F(\hat{u}) \right\rangle_H, \quad i = 1,2,\cdots,n. \tag{8}$$

The Lagrange multipliers $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_m)^\top$ satisfy the linear system

$$C(\mathbf{q})\boldsymbol{\lambda} = \mathbf{b}(\mathbf{q}), \tag{9}$$

where $C(\mathbf{q}) \in \mathbb{R}^{m\times m}$ is the symmetric positive definite *constraint matrix* defined by

$$C_{ij} = \langle \nabla I_j, M^{-1} \nabla I_i \rangle, \quad i,j \in \{1,2,\cdots,m\}, \tag{10}$$

and the vector $\mathbf{b} = (b_1, b_2, \cdots, b_m)^\top \in \mathbb{R}^m$ is given by

$$b_i = \langle \nabla I_i, M^{-1}\mathbf{f} \rangle, \quad i = 1,2,\cdots,m, \tag{11}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean inner product. The gradients ∇I_i denote the partial derivatives with respect to the components of the parameters $\mathbf{q}$.

If no conserved quantities are enforced, then we must solve the optimization problem (5) without any constraints. The unique minimizer of the unconstrained problem is given by omitting the summation term from Eq. (6), i.e.,

$$M(\mathbf{q})\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}). \tag{12}$$

As we mentioned in section 1.1, equation (12) was derived simultaneously and independently by Du and Zaki [13] in the context of EDNNs. However, the more general equation (6), which ensures the preservation of conserved quantities, was only derived in Ref. [1].

A geometric depiction of RONS in the unconstrained case is shown in Fig. 1. We view the shape-morphing approximate solution $\hat{u}(\cdot, \mathbf{q})$ as a map from the parameters $\mathbf{q}$ to the Hilbert space H where solutions of the PDE lie. The approximate solution $\hat{u}$ maps the set of all viable parameter values $\mathbf{q} \in \Omega \subseteq \mathbb{R}^n$ to an n-dimensional manifold $\mathcal{M} \subset H$. An arbitrary but smooth evolution of parameters $\mathbf{q}(t)$ defines a smooth curve in the set Ω. The tangent vector, or velocity, of this curve is given by $\dot{\mathbf{q}}(t)$. Under the map $\hat{u}$, this curve is mapped onto a curve which lies on the manifold $\mathcal{M}$ in the function space H. The tangent vector $\dot{\mathbf{q}}(t)$ is mapped to the tangent vector $\hat{u}_t$ which lies on the tangent space of the manifold $T_{\hat{u}}\mathcal{M}$. In general, the manifold is not invariant under the dynamics of the governing PDE (2), and therefore $F(\hat{u})$ will not necessarily lie in tangent space. By minimizing (3) with respect to $\dot{\mathbf{q}}(t)$, we find the vector $\hat{u}_t$ which is the orthogonal projection of $F(\hat{u})$ onto $T_{\hat{u}}\mathcal{M}$. In other words, we evolve the approximate solution $\hat{u}$ so that it most closely resembles the expected PDE dynamics.

Conventional Galerkin projection models are a special case of RONS. Consider an approximate solution which depends linearly on the parameters,

$$\hat{u}(\mathbf{x}, \mathbf{q}(t)) = \sum_{i=1}^{n} q_i(t) u_i(\mathbf{x}), \tag{13}$$

where the modes $\{u_i\}_{i=1}^n$ are prescribed orthonormal functions, e.g., proper orthogonal decomposition (POD) modes. In this special case, the unconstrained RONS equation (12) coincides with standard Galerkin projection. More specifically, the metric tensor $M(\mathbf{q})$

becomes the identity matrix and the right-hand side vector is given by $f_i = \langle u_i, F(\hat{u})\rangle_H$. From a geometric standpoint, for the reduced-order solution (13), the manifold $\mathcal{M}$ becomes a flat subspace spanned by the modes $\{u_i\}_{i=1}^n$. We refer to [1] for further details.

3. Fast and scalable computational methods

3.1. Computational bottlenecks

Although RONS has shown great promise for both reduced-order modeling and numerical simulation of PDEs [1,13,14,24], forming and solving the RONS equations can be computationally expensive. In this section, we first outline the main computational bottlenecks associated with RONS and then present our proposed remedies.

The computational cost of RONS equations (6) comes from three main sources:

1. Forming the metric tensor $M(\mathbf{q})$ and the vector field $\mathbf{f}(\mathbf{q})$ (addressed in sections 3.2 and 3.3).
2. Stiffness of the RONS equations (addressed in section 3.4).
3. Inverting the metric tensor. By inverting the metric tensor we refer to any numerical method for solving the linear equation (6) for $\dot{\mathbf{q}}$.

We now describe each of these computational bottlenecks in more detail. To form the metric tensor (7), we need to compute n^2 inner products. Since this matrix is symmetric, the number of independent inner products is in fact $n(n+1)/2$. Additionally, to form the right-hand side vector (8), we need to compute n inner products. Therefore, a total of $n(n+3)/2$ integrals need to be computed. As the number of parameters n grows, this becomes computationally prohibitive. Making matters worse, during time stepping, $\mathbf{q}(t)$ changes and these integrals need to be recomputed at each time step. In section 3.2, we develop a method which drastically reduces the number of inner product computations. We refer to this method as *symbolic RONS*, or *S-RONS* for short.

Symbolic RONS requires the inner products to be symbolically computable. Depending on the choice of the approximate solution $\hat{u}$, this may not be feasible. In section 3.3, we develop a collocation point approach to RONS which does not require any integral or inner product computation and therefore reduces the computational cost of RONS by several orders of magnitude. We refer to this method as *collocation RONS*, or *C-RONS* for short.

The second issue arises from the fact that the RONS equations (6) can be stiff as a set of ODEs. As a result, using explicit schemes for time integration may require exceedingly small time steps. In section 3.4, we propose a regularized version of the optimization problem (5) which alleviate this issue. We refer to the resulting method as the *regularized RONS*, which can be used in conjunction with both S-RONS and C-RONS.

In our experience, the last issue (inverting the metric tensor) does not present a major roadblock. There exist several fast methods for solving large linear systems which can be used for inverting the metric tensor [26]. Therefore, we focus on items 1 and 2 above which constitutes the main computational bottlenecks.

3.2. Symbolic RONS

In this section, we present a method for efficient construction of the metric tensor M and right-hand side vector $\mathbf{f}$ using symbolic computation of the required inner products. We show that only a relatively small number of symbolic computations are required to build the RONS equation, provided that the shape-morphing approximate solution has a specific form and analytical symbolic expressions for the inner products in M and $\mathbf{f}$ can be obtained.

Specifically, we consider shape-morphing approximations of the form,

$$\hat{u}(\mathbf{x}, \mathbf{q}(t)) = \sum_{i=1}^{r} \alpha_i(t)\phi(\mathbf{x}, \boldsymbol{\beta}_i(t)). \tag{14}$$

We refer to $\phi(\mathbf{x}, \boldsymbol{\beta}_i(t))$ as the i-th *shape-morphing mode*. The vector of *shape parameters* $\boldsymbol{\beta}_i \in \mathbb{R}^{K-1}$ controls the shape of the i-th mode; it contains parameters such as length scales and center of the mode. The scalar $\alpha_i(t)$ denotes the mode amplitude. Therefore, parameters of the shape-morphing solution $\hat{u}$ are given by $\mathbf{q} = (\alpha_1, \boldsymbol{\beta}_1^\top, ..., \alpha_r, \boldsymbol{\beta}_r^\top)^\top \in \mathbb{R}^{rK}$, where $n = rK$. Note that the number of shape parameters for each mode $K-1$ is independent of the number of terms r in the sum. This independence plays an important role in the proposed computational method. To ensure that the RONS inner products (7)-(8) are well-defined, we require that $\phi(\mathbf{x}, \boldsymbol{\beta}_i)$ must be differentiable almost everywhere with respect to the shape parameters. Additionally, ϕ and its derivatives with respect to the amplitudes and shape parameters must belong to the underlying Hilbert space H.

As an example, we can consider Gaussian modes ϕ which lead to the approximate solution,

$$\hat{u}(x, \mathbf{q}(t)) = \sum_{i=1}^{r} A_i(t) \exp\left[-\frac{(x - c_i(t))^2}{L_i^2(t)}\right], \tag{15}$$

where $\alpha_i(t) = A_i(t)$ and $\boldsymbol{\beta}_i(t) = (L_i(t), c_i(t))^\top$. Here A_i controls the amplitude of the ith Gaussian, L_i is a length scale that determine the Gaussian's width, and c_i determines the Gaussian's center. We use this one-dimensional Gaussian mixture as an illustrative example throughout this section.

There are many other possible choices of modes ϕ for the approximate solution. For example, we could take the modes to be nonlinear activation functions typically used in neural networks, such as the logistic function or hyperbolic tangent. In this case, Eq. (14) represents a shallow neural network and the shape parameters $\boldsymbol{\beta}_i$ are the weights and biases of the i-th node. Another choice could be wavelet functions, where the shape parameters are dilations and translations.

In order to obtain a reasonable approximation of the governing PDE (2), the function ϕ should have nonzero spatial derivatives up to order of the governing PDE. If this requirement is not met, the highest order derivatives of the governing PDE will have no corresponding term in the RONS equation. As an example, let ϕ be the rectified linear unit (ReLU), where second- and higher-order spatial derivatives are zero almost everywhere. Consequently, when forming $F(\hat{u})$, these higher order derivatives are zero almost everywhere and so they do not contribute to dynamics of the approximate solution when applying RONS.

We now exploit the structure of the shape-morphing approximation (14) to efficiently calculate the inner products in M and $\mathbf{f}$ using symbolic computation. We first discuss how to efficiently build the metric tensor M. Consider arbitrary indices $i, j \in \{1, ..., r\}$. Using these general indices and the approximate solution (14), all entries of M will have the form of one of the following inner products,

$$\mathcal{I}_{\alpha_i\alpha_j} := \left\langle \frac{\partial \hat{u}}{\partial \alpha_i}, \frac{\partial \hat{u}}{\partial \alpha_j} \right\rangle_H, \tag{16a}$$

$$\mathcal{I}_{\alpha_i\beta_{jk}} := \left\langle \frac{\partial \hat{u}}{\partial \alpha_i}, \frac{\partial \hat{u}}{\partial \beta_{jk}} \right\rangle_H, \qquad k \in \{1, ..., K-1\} \tag{16b}$$

$$\mathcal{I}_{\beta_{ik}\beta_{j\ell}} := \left\langle \frac{\partial \hat{u}}{\partial \beta_{ik}}, \frac{\partial \hat{u}}{\partial \beta_{j\ell}} \right\rangle_H, \qquad k, \ell \in \{1, ..., K-1\} \quad , \tag{16c}$$

where β_{ik} denotes the k-th component of $\boldsymbol{\beta}_i = (\beta_{i1}, \beta_{i2}, \cdots, \beta_{i(K-1)})^\top$. The advantage of using symbolic computation for the expressions in equation (16) is that after obtaining closed-form expressions for the inner products, we can build the entire metric tensor through substitution of the appropriate indices i and j. For example, rather than computing $\mathcal{I}_{\alpha_1\alpha_2}$, we simply need to substitute the values of α_1 and α_2 into the already obtained symbolic expression for $\mathcal{I}_{\alpha_i\alpha_j}$. This same principle holds regardless of which indices we choose as i and j.

Note that, after obtaining closed-form expressions the inner products in equation (16), we can evaluate all entries of the metric tensor M regardless of the number of modes r used in the approximate solution. Additionally, we only need to perform the symbolic computations at the initial time and can then substitute the updated parameter values as we march the approximate solution forward in time. Note that, by symmetry of the inner product, we only need to calculate $K(K-1)/2$ inner products for equation (16c). Therefore, there are in total $K(K+1)/2$ terms to be calculated in equation (16). We emphasize that this number is independent of the number of modes r in the shape-morphing approximation (14); it only depends on the number of shape parameters K.

Symbolic RONS (or S-RONS) can be alternatively described by examining the structure of the metric tensor. The matrix M is composed of blocks $M^{(i,j)} \in \mathbb{R}^{K\times K}$ such that

$$M = \begin{bmatrix} M^{(1,1)} & M^{(1,2)} & \cdots & M^{(1,r)} \\ M^{(2,1)} & M^{(2,2)} & \cdots & M^{(2,r)} \\ \vdots & \vdots & \ddots & \vdots \\ M^{(r,1)} & M^{(r,2)} & \cdots & M^{(r,r)} \end{bmatrix}, \tag{17}$$

where $M^{(i,j)} = (M^{(j,i)})^\top$ since M is symmetric. Each block can be expressed in terms of the inner products (16),

$$M^{(i,j)} = \begin{bmatrix} \mathcal{I}_{\alpha_i\alpha_j} & \mathcal{I}_{\alpha_i\beta_{j1}} & \cdots & \mathcal{I}_{\alpha_i\beta_{j(K-1)}} \\ \mathcal{I}_{\beta_{i1}\alpha_j} & \mathcal{I}_{\beta_{i1}\beta_{j1}} & \cdots & \mathcal{I}_{\beta_{i1}\beta_{j(K-1)}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{I}_{\beta_{i(K-1)}\alpha_j} & \mathcal{I}_{\beta_{i(K-1)}\beta_{j1}} & \cdots & \mathcal{I}_{\beta_{i(K-1)}\beta_{j(K-1)}} \end{bmatrix}. \tag{18}$$

With this labeling, the block $M^{(i,j)}$ represents all of the inner products involving derivatives of the approximate solution with respect to the shape parameters β_{ik} and β_{jk} of the i-th and j-th modes and their respective amplitudes, α_i and α_j. Although there are K^2 entries in $M^{(i,j)}$, we only need to perform symbolic calculations for the $K(K+1)/2$ entries in the lower triangular part of the block. The remaining entries of the matrix, enclosed in a box in (18), are then determined by the symmetry of inner products in (16). In other words, if we have a closed-form expression for an entry $\mathcal{I}_{q_iq_j}$ in the lower triangular part of $M^{(i,j)}$, we obtain the corresponding entry $\mathcal{I}_{q_jq_i}$ in the upper triangular part of the block by simply swapping the values of q_i and q_j in the symbolic expression.

As an example, consider the Gaussian mixture (15). We must symbolically compute six inner products to form the block,

$$M^{(i,j)} = \begin{bmatrix} \left\langle \frac{\partial \hat{u}}{\partial A_i}, \frac{\partial \hat{u}}{\partial A_j} \right\rangle_H & \boxed{\left\langle \frac{\partial \hat{u}}{\partial A_i}, \frac{\partial \hat{u}}{\partial L_j} \right\rangle_H} & \boxed{\left\langle \frac{\partial \hat{u}}{\partial A_i}, \frac{\partial \hat{u}}{\partial c_j} \right\rangle_H} \\ \left\langle \frac{\partial \hat{u}}{\partial L_i}, \frac{\partial \hat{u}}{\partial A_j} \right\rangle_H & \left\langle \frac{\partial \hat{u}}{\partial L_i}, \frac{\partial \hat{u}}{\partial L_j} \right\rangle_H & \boxed{\left\langle \frac{\partial \hat{u}}{\partial L_i}, \frac{\partial \hat{u}}{\partial c_j} \right\rangle_H} \\ \left\langle \frac{\partial \hat{u}}{\partial c_i}, \frac{\partial \hat{u}}{\partial A_j} \right\rangle_H & \left\langle \frac{\partial \hat{u}}{\partial c_i}, \frac{\partial \hat{u}}{\partial L_j} \right\rangle_H & \left\langle \frac{\partial \hat{u}}{\partial c_i}, \frac{\partial \hat{u}}{\partial c_j} \right\rangle_H \end{bmatrix}. \tag{19}$$

Note that the terms enclosed in the box can be evaluated using the lower triangular part of the matrix. For instance, $\langle \partial_{A_i} \hat{u}, \partial_{L_j} \hat{u} \rangle$ is evaluated using the symbolic expression for $\langle \partial_{L_i} \hat{u}, \partial_{A_j} \hat{u} \rangle$ by substituting the values of A_i and L_j instead of A_j and L_i, respectively. Therefore, only 6 symbolic computations are required to form the matrix block (19) and consequently the entire metric tensor M. In comparison, computing the metric tensor by a brute force method, such as quadrature or Monte Carlo methods, would require evaluating $n(n+1)/2 = 3r(3r+1)/2$ integrals, which becomes prohibitive as the number of terms r increases.

The idea for building $\mathbf{f}$ is similar to that of the metric tensor. We again consider a general index $i \in \{1, ..., r\}$ and note all entries of $\mathbf{f}$ will have the form of one of the following inner products:

$$\left\langle \frac{\partial \hat{u}}{\partial \alpha_i}, F(\hat{u}) \right\rangle_H, \quad \left\langle \frac{\partial \hat{u}}{\partial \beta_{ik}}, F(\hat{u}) \right\rangle_H, \quad k = 1, ..., K-1. \tag{20}$$

After using symbolic computation to obtain closed-form expressions for the K inner products in equation (20), we can then build $\mathbf{f}$ through substitution rather than individually calculating each of the $n = rK$ inner products in $\mathbf{f}$.

The vector field $\mathbf{f}$ also has a block structure. We may consider $\mathbf{f}$ as r vectors $\mathbf{f}^{(i)} \in \mathbb{R}^K$ stacked on top of each other so that

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}^{(1)} \\ \vdots \\ \mathbf{f}^{(r)} \end{bmatrix}, \tag{21}$$

where each vector $\mathbf{f}^{(i)}$ is defined by

$$\mathbf{f}^{(i)} = \left[\left\langle \frac{\partial \hat{u}}{\partial \alpha_i}, F(\hat{u}) \right\rangle_H, \quad \left\langle \frac{\partial \hat{u}}{\partial \beta_{i1}}, F(\hat{u}) \right\rangle_H, \quad \cdots, \quad \left\langle \frac{\partial \hat{u}}{\partial \beta_{i(K-1)}}, F(\hat{u}) \right\rangle_H \right]^\top. \tag{22}$$

To evaluate the entire vector $\mathbf{f}$, we only need the symbolic expression for one of the blocks $\mathbf{f}^{(i)}$. Again using the Gaussian mixture as an example, we have

$$\mathbf{f}^{(i)} = \left[\left\langle \frac{\partial \hat{u}}{\partial A_i}, F(\hat{u}) \right\rangle_H, \quad \left\langle \frac{\partial \hat{u}}{\partial L_i}, F(\hat{u}) \right\rangle_H, \quad \left\langle \frac{\partial \hat{u}}{\partial c_i}, F(\hat{u}) \right\rangle_H \right]^\top. \tag{23}$$

Using a general index i we need symbolic expressions for only three integrals to build the vector $\mathbf{f}$ through substitution rather than computing $3r$ integrals.

In summary, building the metric tensor M requires $K(K+1)/2$ symbolic integrations and building the right-hand side vector $\mathbf{f}$ requires K symbolic computations, resulting in only $K(K+3)/2$ symbolic computations to evaluate $n(n+1)$ terms appearing in the RONS equation (12). The above discussion leads to the following theorem.

Theorem 1. *Consider a shape-morphing approximate solution of the form* (14). *Forming the metric tensor M and the right-hand side vector $\mathbf{f}$ in the RONS equation* (6) *requires symbolic calculation of $K(K+3)/2$ inner products. The number of symbolic computations is independent of the number of modes r used in the approximate solution.*

An important implication of Theorem 1 is that the number of modes r can be increased arbitrarily without making the computations prohibitive. As a result, it extends RONS beyond a reduced-order modeling framework, where relatively small number of modes are used, and allows us to use RONS for accurate approximation of the PDE's solutions. More specifically, one can think of (14) as a spectral method with the modes $\phi(\cdot, \boldsymbol{\beta}_i(t))$. In contrast to conventional spectral methods, such as the Fourier spectral method, the modes are allowed to change their shape and position over time to adapt to the solution of the PDE by evolving the shape parameters $\boldsymbol{\beta}_i(t)$. As we show in section 4, this shape-morphing property is specially appealing for advection-dominated problems or high-dimensional PDEs with localized solutions.

The shape-morphing approximate solution (1) can be viewed alternatively as a shallow neural network with r nodes, where $(\alpha_i(t), \boldsymbol{\beta}_i(t))$ are the time-dependent parameters of the i-th node. The network's parameters evolve according to the RONS equation (6), without requiring any data or training. Universal approximation theorems [27–29] ensure that, with an appropriate choice of activation function ϕ, we can approximate the solution u to arbitrary accuracy as long as the number of nodes r is large enough. Here, K is the number of parameters in a single node. Theorem 1 ensures that the RONS equations can be formed efficiently regard-

less of the number of nodes r in the shallow neural network. We point out that this theorem is not immediately applicable to deep evolutional neural networks [13].

Finally, we point out that the summation term in equation (6) involves the Lagrange multipliers λ_k which are obtained as the solution to the linear system (9). Note that this linear system only involves Euclidean inner products and therefore its construction is not computationally expensive.

3.3. Collocation RONS

While S-RONS is efficient, obtaining symbolic expressions for the required inner products may not always be feasible. In this section, we present a new approach to RONS where we only enforce that the approximate solution satisfies the governing PDE on a set of prescribed collocation points. This method is applicable for any choice of the approximate solution $\hat{u}$ and does not require symbolic computing or numerical integration.

We first describe collocation RONS without enforcing any conserved quantities. We define the residual function,

$$R(\mathbf{x}, \mathbf{q}, \dot{\mathbf{q}}) := \hat{u}_t - F(\hat{u}) = \sum_{j=1}^{n} \frac{\partial \hat{u}}{\partial q_j} \dot{q}_j - F(\hat{u}). \tag{24}$$

The residual function R measures the point-wise difference between the rate of change of the approximate solution $\hat{u}_t$ and the dynamics $F(\hat{u})$ dictated by the governing PDE. Previous studies [1,24,14,13] have all sought an evolution of parameters $\mathbf{q}(t)$ which minimizes the norm of the residual function in the underlying Hilbert space by minimizing the cost function (3).

Here we propose a different approach. In its most general form, we assume $R(\cdot, \mathbf{q}, \dot{\mathbf{q}}) : D \to \mathbb{R}$ belongs to a function space V to be specified shortly. For any test function $\phi \in V^*$, we require $\langle \phi, R \rangle = 0$, where $\langle \cdot, \cdot \rangle$ denotes the natural pairing between V and its dual space V^*. For computational purposes, we reduce this problem to its finite-dimensional version. More specifically, as in the Petrov-Galerkin method [30], we choose a finite number of test functions $\{\phi_i\}_{i=1}^{N} \in V^*$ and require that $\langle \phi_i, R \rangle = 0$ for $i = 1, 2, \cdots, N$.

As a special case, we derive a collocation method by assuming that $R(\cdot, \mathbf{q}, \dot{\mathbf{q}})$ is bounded, i.e., $V = L^\infty(D)$. Furthermore, we consider the test functions $\phi_i(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_i)$ for $i = 1, 2, ..., N$, where each $\mathbf{x}_i$ is a collocation point in the spatial domain D. Note that $\phi_i \in L^1(D)$ and the natural pairing implies

$$\langle \phi_i, R \rangle = \int_D \delta(\mathbf{x} - \mathbf{x}_i) R(\mathbf{x}, \mathbf{q}, \dot{\mathbf{q}}) \mathrm{d}\mathbf{x} = R(\mathbf{x}_i, \mathbf{q}, \dot{\mathbf{q}}) = 0, \tag{25}$$

which requires the residual function to vanish at the collocation point $\mathbf{x}_i$. Using definition (24), we obtain

$$\sum_{j=1}^{N} \frac{\partial \hat{u}}{\partial q_j}(\mathbf{x}_i, \mathbf{q}) \dot{q}_j = F(\hat{u})\Big|_{\mathbf{x}=\mathbf{x}_i}, \quad i \in \{1, 2, \cdots, N\}. \tag{26}$$

We write (26) as a system of equations,

$$\widetilde{M}(\mathbf{q}) \dot{\mathbf{q}} = \widetilde{\mathbf{f}}(\mathbf{q}), \tag{27}$$

where the collocation matrix $\widetilde{M}(\mathbf{q}) \in \mathbb{R}^{N \times n}$ is given by

$$\widetilde{M}_{ij}(\mathbf{q}) = \frac{\partial \hat{u}}{\partial q_j}(\mathbf{x}_i, \mathbf{q}), \quad i \in \{1, 2, \cdots, N\}, \quad j \in \{1, 2, \cdots, n\}, \tag{28}$$

and the vector $\widetilde{\mathbf{f}}(\mathbf{q}) \in \mathbb{R}^N$ is defined by

$$\widetilde{f}_i(\mathbf{q}) = F(\hat{u}(\mathbf{x}, \mathbf{q}))\Big|_{\mathbf{x}=\mathbf{x}_i}, \quad i \in \{1, 2, \cdots, N\}. \tag{29}$$

Note that equation (27) requires that the approximate solution $\hat{u}$ satisfies the governing PDE at the collocation points $\mathbf{x}_i$.

We refer to equation (27) as collection RONS, or C-RONS for short. Although this equation resembles the unconstrained RONS equation (12), there are notable differences. First, to form the C-RONS equation, numerical integrations or symbolic computations are not required; we only need point-wise evaluation of known functions in (28) and (29). Second, unlike the metric tensor $M(\mathbf{q}) \in \mathbb{R}^{n \times n}$, the collocation matrix $\widetilde{M}(\mathbf{q})$ is rectangular. Consequently, the linear system (27) may not have a unique solution.

If the number of collocation points is greater than the number of parameters, $N > n$, then the system is overdetermined and a solution may not exist. If the number of collocation points is less than the number of parameters, $N < n$, then the system is underdetermined and there may exist infinitely many solutions to the problem. In either case, we obtain $\dot{\mathbf{q}}$ by solving the least squares problem,

$$\min_{\dot{\mathbf{q}} \in \mathbb{R}^n} \| \widetilde{M} \dot{\mathbf{q}} - \widetilde{\mathbf{f}} \|_2^2, \tag{30}$$

using the Moore-Penrose pseudoinverse of $\widetilde{M}$. Thus, for collocation RONS, the evolution of parameters is given by

$$\dot{\mathbf{q}} = \widetilde{M}^{+}(\mathbf{q}) \widetilde{\mathbf{f}}(\mathbf{q}), \tag{31}$$

where $\widetilde{M}^{+}(\mathbf{q}) \in \mathbb{R}^{n\times N}$ denotes the pseudoinverse of the collocation matrix $\widetilde{M}(\mathbf{q})$. If the C-RONS equation (27) is overdetermined, then the solution (31) is the unique solution to the least-squares problem (30). If the system of equations is underdetermined, then the solution (31) is the solution to the least-squares problem with minimal Euclidean norm [31].

Finally, we note that the least square problem (30) is equivalent to minimizing the residual sum $\sum_{i=1}^{N} |R(\mathbf{x}_i, \mathbf{q}, \dot{\mathbf{q}})|^2$ over all possible $\dot{\mathbf{q}} \in \mathbb{R}^n$. In other words, instead of requiring the residual function to vanish at the collocation points as in (25), we choose $\dot{\mathbf{q}}$ so that the sum of squares of the residual is minimized.

Now we turn to the problem of enforcing the PDE's conserved quantities in the approximate solution. Note from section 2, that the governing PDE may have a number of conserved quantities I_i for $i = 1, 2, \cdots, m$. We would like to ensure that these quantities are also conserved along the approximate solution $\hat{u}(\cdot, \mathbf{q}(t))$. In other words, we require $I_i(\mathbf{q}(t)) = I_i(\mathbf{q}(0))$ for all times $t \geq 0$. Taking the derivative of this identity with respect to time, we obtain the equivalent set of equations,

$$\langle \nabla I_i(\mathbf{q}), \dot{\mathbf{q}} \rangle = 0, \quad i = 1, 2, \cdots, m. \tag{32}$$

These constraints, together with the C-RONS equation (27), lead to the larger system of equations,

$$\widetilde{M}_c(\mathbf{q})\dot{\mathbf{q}} = \widetilde{\mathbf{f}}_c(\mathbf{q}), \tag{33}$$

where the constrained collocation matrix $\widetilde{M}_c \in \mathbb{R}^{(N+m)\times n}$ and the constrained vector field $\widetilde{\mathbf{f}}_c \in \mathbb{R}^{N+m}$ are defined by

$$\widetilde{M}_c(\mathbf{q}) := \begin{bmatrix} \widetilde{M}(\mathbf{q}) \\ \nabla I_1(\mathbf{q})^\top \\ \nabla I_2(\mathbf{q})^\top \\ \vdots \\ \nabla I_m(\mathbf{q})^\top \end{bmatrix}, \quad \widetilde{\mathbf{f}}_c(\mathbf{q}) := \begin{bmatrix} \widetilde{\mathbf{f}}(\mathbf{q}) \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{34}$$

As before, we solve the linear system (33) using the pseudoinverse to obtain the constrained collocation equation $\dot{\mathbf{q}} = \widetilde{M}_c^{+}\widetilde{\mathbf{f}}_c$.

Remark 1. When the inner products (7)-(8) can be computed symbolically, we prefer to use S-RONS over C-RONS. This is because S-RONS provides several significant benefits over C-RONS or other approximations of the inner products. First, symbolic computation allows us to obtain exact expressions for the inner products, thus avoiding approximation errors which arise from Monte Carlo sampling or quadrature. Second, when using C-RONS or Monte Carlo approximation, the inner products need to be recomputed at every time step. Furthermore, if the PDE's initial condition changes, C-RONS or Monte Carlo approximations need to be recomputed. In contrast, when using S-RONS, symbolic computation of the inner products only needs to be performed once. During time stepping, the analytical symbolic expressions need to be evaluated at the new parameter values, but no additional symbolic computation is required. Similarly, if the initial condition changes, the same symbolic expressions can be reused. S-RONS is also more scalable compared to C-RONS or Monte Carlo sampling. For high-dimensional problems, we require many collocation points or samples to obtain accurate expressions of the inner products which is computationally expensive. S-RONS avoids this issue because we only need to calculate a small number of symbolic expressions that can be reused as we evolve the approximate solution (see Theorem 1). Thus, we use C-RONS only when the inner products cannot be computed symbolically.

3.3.1. Relation between Monte Carlo sampling and C-RONS

Monte Carlo integration has previously been used to approximate the inner products in the RONS equation [14,13]. We will show that, under certain conditions, the evolution of parameters provided by Monte Carlo approximation coincides with C-RONS (30). Although these two methods are mathematically equivalent, C-RONS proves to be numerically more stable.

For the Monte Carlo approximation, one draws a random sample $\{\mathbf{x}_k\}_{k=1}^{N}$ from the spatial domain D to approximate the inner products which appear in equations (7) and (8). More specifically, taking the Hilbert space H to be the space of square-integrable functions $L^2(D)$, the Monte Carlo approximations for the metric tensor M and right-hand side vector $\mathbf{f}$ are given by

$$\begin{aligned} M_{ij}(\mathbf{q}) \approx \bar{M}_{ij}(\mathbf{q}) &:= \frac{|D|}{N}\sum_{k=1}^{N} \frac{\partial \hat{u}}{\partial q_i}(\mathbf{x}_k, \mathbf{q})\frac{\partial \hat{u}}{\partial q_j}(\mathbf{x}_k, \mathbf{q}), && i, j \in \{1, ..., n\}, \\ f_i(\mathbf{q}) \approx \bar{f}_i(\mathbf{q}) &:= \frac{|D|}{N}\sum_{k=1}^{N} \frac{\partial \hat{u}}{\partial q_i}(\mathbf{x}_k, \mathbf{q})F(\hat{u})\Big|_{\mathbf{x}=\mathbf{x}_k}, && i \in \{1, ..., n\}, \end{aligned} \tag{35}$$

where $\bar{M} \in \mathbb{R}^{n\times n}$ and $\bar{\mathbf{f}} \in \mathbb{R}^n$ denote the Monte Carlo approximations and $|D|$ denotes the size of the spatial domain, assuming that it is bounded. Du and Zaki [13] draw the samples $\{\mathbf{x}_k\}_{k=1}^{N}$ from a uniform distribution. Bruna et al. [14] showed that drawing the samples from an adaptive distribution that depends on the approximate solution $\hat{u}$ may lead to more accurate solutions. In either case, the unconstrained RONS (12) with Monte Carlo approximation can be written as

$$\bar{M}(\mathbf{q})\dot{\mathbf{q}} = \bar{\mathbf{f}}(\mathbf{q}). \tag{36}$$

Equation (35) reveals a close relation between C-RONS and the Monte Carlo approximation of RONS. Note that the Monte Carlo approximation of the metric tensor satisfies $\bar{M} = (|D|/N)\widetilde{M}^\top\widetilde{M}$, where $\widetilde{M}$ is the collocation matrix (28). Similarly, the Monte Carlo

approximation of the right-hand side vector field $\mathbf{f}$ satisfies $\widetilde{\mathbf{f}} = (|D|/N)\widetilde{M}^\top \widetilde{\mathbf{f}}$, where $\widetilde{\mathbf{f}}$ is the C-RONS vector field (29). Therefore, the Monte Carlo approximation of RONS (36) can be equivalently written as

$$\widetilde{M}^\top \widetilde{M}\dot{\mathbf{q}} = \widetilde{M}^\top \widetilde{\mathbf{f}}, \tag{37}$$

where $\widetilde{M}$ is the collocation matrix.

The following theorem shows that, if the C-RONS matrix $\widetilde{M}$ has full column rank, then the Monte Carlo approximation of RONS and the C-RONS equation (31) are mathematically equivalent.

Theorem 2. *If the C-RONS matrix $\widetilde{M}$ has full column rank, then the Monte Carlo approximation of RONS* (36) *is equivalent to C-RONS equation* (31)*.*

Proof. First recall that the Monte Carlo approximation (36) is equivalent to equation (37). If $\widetilde{M}$ is full column rank, then $\widetilde{M}^\top \widetilde{M}$ is invertible and therefore we have $\dot{\mathbf{q}} = (\widetilde{M}^\top \widetilde{M})^{-1}\widetilde{M}^\top \widetilde{\mathbf{f}}(\mathbf{q})$. On the other hand, since $\widetilde{M}$ has full column rank, the pseudoinverse of $\widetilde{M}$ is given explicitly by $\widetilde{M}^+ = (\widetilde{M}^\top \widetilde{M})^{-1}\widetilde{M}^\top$. Substituting this expression in C-RONS equation (31), we conclude that the Monte Carlo approximation of RONS and C-RONS lead to the same equation for $\dot{\mathbf{q}}$. □

Remark 2. We note an important distinction between the exact result of Theorem 2 and its numerical implementation. Although Theorem 2 states that C-RONS equation (27) and the Monte Carlo approximation of RONS (37) are equivalent in exact arithmetic, the C-RONS equation is numerically better conditioned than the Monte Carlo approach. To see this, note that $\kappa(\widetilde{M}^\top \widetilde{M}) = [\kappa(\widetilde{M})]^2$, where the condition number is defined as $\kappa(\widetilde{M}) := \sigma_{\max}(\widetilde{M})/\sigma_{\min}(\widetilde{M})$ with $\sigma_{\max}$ and $\sigma_{\min}$ denoting the maximal and minimal nonzero singular values of the matrix, respectively. Thus, it is numerically more stable to solve the C-RONS equation to obtain $\dot{\mathbf{q}}$. In other words, although $\widetilde{M}^+ = (\widetilde{M}^\top \widetilde{M})^{-1}\widetilde{M}^\top$ in exact arithmetic, it is well-known that this formula is numerically sensitive. Instead, we use the singular value decomposition of $\widetilde{M}$ to compute its pseudoinverse which is numerically more stable [31]. In contrast, using the Monte Carlo approximation (37), one must inevitably work with the matrix $\bar{M} \propto \widetilde{M}^\top \widetilde{M}$ which in practice tends to have a significantly larger condition number than $\widetilde{M}$.

Theorem 2 sheds light on the unreasonable effectiveness of the Monte Carlo approximation applied to RONS. As we show in section 4.2 below, using Monte Carlo integration, with a relatively small samples size $N = 128$, captures the behavior of the Kuramoto–Sivashinsky PDE reasonably well. This is surprising because the sample size is too small to accurately approximate the integrals involved in the metric tensor M or the right-hand side vector $\mathbf{f}$ (see section 4.2 for a quantitative comparison). Yet, the approximate solution is reasonably close to a true solution of the Kuramoto–Sivashinsky equation. Theorem 2 shows that this accuracy is not owed to the accuracy of the Monte Carlo approximation; rather it is due to the fact that this approximation, although disguised as a Monte Carlo method, is in fact a collocation method. As such, it minimizes the approximation error at $N = 128$ collocation points. We discuss this point in greater detail in section 4.2.

3.4. Regularized RONS

In sections 3.2 and 3.3, we developed two efficient methods to construct the RONS equations. The next step is to solve the resulting ODEs in order to evolve the parameters $\mathbf{q}(t)$ of the shape-morphing approximation $\hat{u}(\mathbf{x}, \mathbf{q}(t))$. In our experience, when the number of parameters n is large, the RONS equations may become stiff. As a result, using explicit schemes for numerical integration leads to exceedingly small time steps.

To overcome this problem, Refs [13,14] use implicit time integration which unfortunately introduces a different set of issues. Namely, implicit methods require solving a nonlinear system at every time step which adds to the computational cost of RONS. Furthermore, the iterative methods for solving the nonlinear system are not guaranteed to converge [32,33]. In order to address the possible stiffness of RONS equations, while avoiding implicit schemes, we introduce a regularized version of RONS.

More specifically, we add a Tikhonov penalization term [34,35] to the cost function (3), and define the regularized cost function,

$$\hat{\mathcal{J}}(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2}\|\hat{u}_t - F(\hat{u})\|_H^2 + \frac{1}{2}\|\Gamma\dot{\mathbf{q}}\|_2^2, \tag{38}$$

where the full-rank Tikhonov matrix $\Gamma \in \mathbb{R}^{P\times n}$ $(P \geq n)$ is to be specified. We then consider the constrained minimization problem,

$$\begin{aligned} &\min_{\dot{\mathbf{q}}\in\mathbb{R}^n} \ \hat{\mathcal{J}}(\mathbf{q}, \dot{\mathbf{q}}) \\ &\text{subject to} \quad I_k(\mathbf{q}(t)) = I_k(\mathbf{q}(0)), \quad k = 1, 2, ..., m, \quad \forall t \geq 0. \end{aligned} \tag{39}$$

As before, the constraints ensure that the resulting solution $\hat{u}(\mathbf{x}, \mathbf{q}(t))$ conserves the first integrals I_k. The following theorem gives the explicit form of a minimizer to the regularized optimization problem (39).

Theorem 3. *If Γ has full column rank and the constraint gradients $\nabla I_1(\mathbf{q}), \nabla I_2(\mathbf{q}), \cdots, \nabla I_m(\mathbf{q})$ are linearly independent, then the solution to the regularized minimization problem* (39) *satisfies*

$$(M(\mathbf{q})+\Gamma^\top\Gamma)\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}) - \sum_{i=1}^{m} \hat{\lambda}_k \nabla I_k(\mathbf{q}). \tag{40}$$

The Lagrange multipliers $\hat{\boldsymbol{\lambda}} = (\hat{\lambda}_1, ..., \hat{\lambda}_m)^\top$ *are determined through the linear system*

$$\hat{C}(\mathbf{q})\hat{\boldsymbol{\lambda}} = \hat{\mathbf{b}}(\mathbf{q}), \tag{41}$$

where $\hat{C}$ *is the regularized constraint matrix with entries,*

$$\hat{C}_{ij} = \langle \nabla I_j, (M+\Gamma^\top\Gamma)^{-1}\nabla I_i\rangle, \quad i,j\in\{1,2,\cdots,m\}, \tag{42}$$

and the vector $\hat{\mathbf{b}} = (\hat{b}_1, \hat{b}_2, \cdots, \hat{b}_m)^\top \in \mathbb{R}^m$ *is given by*

$$\hat{b}_i = \langle \nabla I_i, (M+\Gamma^\top\Gamma)^{-1}\mathbf{f}\rangle, \quad i = 1,2,\cdots,m. \tag{43}$$

Proof. See appendix A. □

We refer to equation (40) as the *regularized RONS* equation.

Remark 3. In addition to alleviating the stiffness of the RONS equations, the regularization also relaxes the assumptions needed on the shape-morphing solution $\hat{u}$. Note that the metric tensor M is symmetric by definition. It is also positive semi-definite because, for all $\boldsymbol{\xi}\in\mathbb{R}^n$, we have

$$\langle \boldsymbol{\xi}, M\boldsymbol{\xi}\rangle = \sum_{i=1}^{n}\sum_{j=1}^{n}\left\langle \frac{\partial \hat{u}}{\partial q_i}\xi_i, \frac{\partial \hat{u}}{\partial q_j}\xi_j\right\rangle_H = \left\| \sum_{i=1}^{n}\frac{\partial \hat{u}}{\partial q_i}\xi_i\right\|_H^2 \geq 0. \tag{44}$$

In Ref. [1], to ensure that M was positive definite and therefore invertible, we required the assumption that the approximate solution was an immersion (see Lemma 1 of [1]), i.e.,

$$\dim\left(\operatorname{span}\left\{\frac{\partial \hat{u}}{\partial q_1}, \frac{\partial \hat{u}}{\partial q_2}, ..., \frac{\partial \hat{u}}{\partial q_n}\right\}\right) = n. \tag{45}$$

In regularized RONS, we do not require the immersion assumption. Note that, for regularized RONS, we only need the invertibility of $M+\Gamma^\top\Gamma$ which is always guaranteed. This is because M is positive semi-definite, and $\Gamma^\top\Gamma$ is symmetric positive definite. Therefore, $M+\Gamma^\top\Gamma$ is symmetric positive definite and invertible, regardless of whether the approximate solution $\hat{u}$ is an immersion.

We can similarly apply Tikhonov regularization to C-RONS. Recall the least squares problem (30) which arises for the collocation point method, and consider its regularized counterpart,

$$\begin{aligned} &\min_{\dot{\mathbf{q}}\in\mathbb{R}^n} \ \|\widetilde{M}\dot{\mathbf{q}} - \widetilde{\mathbf{f}}\|_2^2 + \|\Gamma\dot{\mathbf{q}}\|_2^2 \\ &\text{subject to} \quad I_k(\mathbf{q}(t)) = I_k(\mathbf{q}(0)), \quad k = 1,2,...,m, \quad \forall t\geq 0. \end{aligned} \tag{46}$$

The following theorem gives an explicit expression to the solution of this optimization problem.

Theorem 4. *If* Γ *has full column rank and the constraint gradients* $\nabla I_1(\mathbf{q}), \nabla I_2(\mathbf{q}), \cdots, \nabla I_m(\mathbf{q})$ *are linearly independent, the minimizer to the constrained optimization problem* (46) *satisfies*

$$(\widetilde{M}^\top\widetilde{M}+\Gamma^\top\Gamma)\dot{\mathbf{q}} = \widetilde{M}^\top\widetilde{\mathbf{f}} - \sum_{i=1}^{m}\widetilde{\lambda}_k\nabla I_k(\mathbf{q}), \tag{47}$$

which we refer to as the regularized C-RONS equation. The Lagrange multipliers $\widetilde{\boldsymbol{\lambda}} = (\widetilde{\lambda}_1, ..., \widetilde{\lambda}_m)^\top$ *are determined through the linear system*

$$\widetilde{C}(\mathbf{q})\widetilde{\boldsymbol{\lambda}} = \widetilde{\mathbf{b}}(\mathbf{q}), \tag{48}$$

where the regularized constraint matrix $\widetilde{C}$ *has entries,*

$$\widetilde{C}_{ij} = \langle \nabla I_j, (\widetilde{M}^\top\widetilde{M}+\Gamma^\top\Gamma)^{-1}\nabla I_i\rangle, \quad i,j\in\{1,2,\cdots,m\}, \tag{49}$$

and the vector $\widetilde{\mathbf{b}} = (\widetilde{b}_1, \widetilde{b}_2, \cdots, \widetilde{b}_m)^\top \in \mathbb{R}^m$ *is given by*

$$\widetilde{b}_i = \langle \nabla I_i, (\widetilde{M}^\top\widetilde{M}+\Gamma^\top\Gamma)^{-1}\widetilde{M}^\top\widetilde{\mathbf{f}}\rangle, \quad i = 1,2,\cdots,m. \tag{50}$$

Proof. The proof of this theorem is very similar to that of Theorem 3 and therefore is omitted here for brevity. □

As before, invertibility of $(\widetilde{M}^\top \widetilde{M} + \Gamma^\top \Gamma)$ is guaranteed by the fact that $\widetilde{M}^\top \widetilde{M}$ is symmetric, positive semi-definite and $\Gamma^\top \Gamma$ is positive definite. In the numerical examples presented in section 4, we take the matrix Γ to be a multiple of the identity matrix so that $\Gamma^\top \Gamma = \alpha I$, where $\alpha > 0$ is a prescribed regularization parameter.

There exist several methods in the literature for choosing the optimal value of the regularization parameter α such as the L-curve method [34,36], generalized cross-validation [37–39], the quasi-optimality criterion [40–42], and an iterative scheme which converges to an optimal regularization parameter [43]. Applying these methods require repeated solves of the linear equation (40) for S-RONS or (47) for C-RONS. Since these systems are not excessively large, the computational cost of solving the linear system is insignificant compared to the primary computational bottlenecks which are forming the RONS equations and its subsequent time integration. For example, the largest approximate solution we consider in this manuscript has 300 parameters (see section 4.1.2). This means that the linear system size is 300 (the S-RONS metric tensor is a 300×300 matrix), which is easily solvable with negligible computational cost.

4. Numerical results

In this section we present two numerical examples: the Fokker–Planck equation and the Kuramoto–Sivashinsky equation. The Fokker–Planck equation demonstrates the computational advantages of using symbolic RONS as introduced in section 3.2. The Kuramoto–Sivashinsky equation demonstrates the benefits of using the collocation point method (section 3.3) over Monte Carlo integration. In both numerical examples, we also discuss the advantages of regularization as described in section 3.4. We carried out our computations on a 2019 Macbook Pro with a 1.7 GHz Quad-Core Intel Core i7 processor. Time integration for the Kuramoto–Sivashinsky equation was carried out using an explicit adaptive Runge-Kutta scheme, i.e., Matlab's `ode45` [44]. For the Fokker–Planck equation, we used an explicit adaptive multi-step solver, i.e., Matlab's `ode113` [45].

4.1. Fokker–Planck equation

In this section we consider the Fokker–Planck equation in eight dimensions. We demonstrate that applying RONS with symbolic computation provides solutions which are several orders of magnitude more accurate and faster than the adaptive Monte Carlo sampling technique used in [14]. We also demonstrate the importance of enforcing conserved quantities to obtain accurate approximate solutions of the Fokker–Planck equation.

Following Bruna et al. [14], we consider d interacting particles whose motion is governed by the system of stochastic differential equations (SDEs),

$$\mathrm{d}X_i = g(t, X_i)\mathrm{d}t + \sum_{j=1}^{d} K(X_i, X_j)\mathrm{d}t + \sqrt{2\nu}\mathrm{d}W_i, \quad i = 1, 2, ..., d. \tag{51}$$

Here $X_i(t)$ denotes the position of the i-th particle at time t, $g : [0, \infty) \times \mathbb{R} \to \mathbb{R}$ is a forcing term, $K : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ describes the pairwise interactions between particles, ν is a positive diffusion constant, and W_i is a standard Wiener process. The Fokker–Planck equation is a deterministic PDE which describes the evolution of the probability density function (PDF), $p(\mathbf{x}, t)$, for the location of the particles. The Fokker–Planck equation associated with (51) reads

$$\frac{\partial p}{\partial t} = \sum_{i=1}^{d} -\frac{\partial}{\partial x_i}\left[\left(g(t, x_i) + \sum_{j=1}^{d} K(x_i, x_j)\right)p\right] + \nu \frac{\partial^2 p}{\partial x_i^2}. \tag{52}$$

As the spatial dimension d grows, solving the Fokker–Planck equation using conventional discretization methods becomes prohibitive [46–48]. Alternatively, one may seek to approximate the density $p(\mathbf{x}, t)$ using Monte Carlo simulations of the original SDE (51). This also becomes prohibitively expensive in higher dimensions since exceedingly large samples are required. Here, we use RONS to directly approximates $p(\mathbf{x}, t)$, bypassing the need for any Monte Carlo simulations of the SDE or spatial discretization of the PDE.

As in [14], we choose the functions g and K to be

$$g(t, x_i) = a(t) - x_i, \quad K(x_i, x_j) = \frac{\alpha}{d}(x_j - x_i), \tag{53}$$

which correspond to particles in a harmonic trap centered in each spatial coordinate at $a(t)$ while the particles also attract each other. A significant advantage of this choice is that we can obtain analytical expressions for the mean and covariance of each particle to use as a benchmark for our approximate solutions. Taking the expected value of the SDE (51) with our choices of g and K given in (53), we obtain the following expressions for the mean $\bar{X}_i = \mathbb{E}[X_i]$ of each particle

$$\dot{\bar{X}}_i = a(t) - \bar{X}_i + \frac{\alpha}{d}\sum_{j=1}^{d}(\bar{X}_j - \bar{X}_i), \quad i = 1, 2, ..., d. \tag{54}$$

Similarly, we have the following expressions for entries of the matrix $\Sigma_{ij} = \mathbb{E}[X_i X_j]$,

$$\dot{\Sigma}_{ij} = a(t)(\bar{X}_j + \bar{X}_i) - 2(1 + \alpha)\Sigma_{ij} + \frac{\alpha}{d}\sum_{l=1}^{d}(\Sigma_{lj} + \Sigma_{li}) + 2\nu\delta_{ij}, \qquad i, j \in \{1, 2, ..., d\}, \tag{55}$$

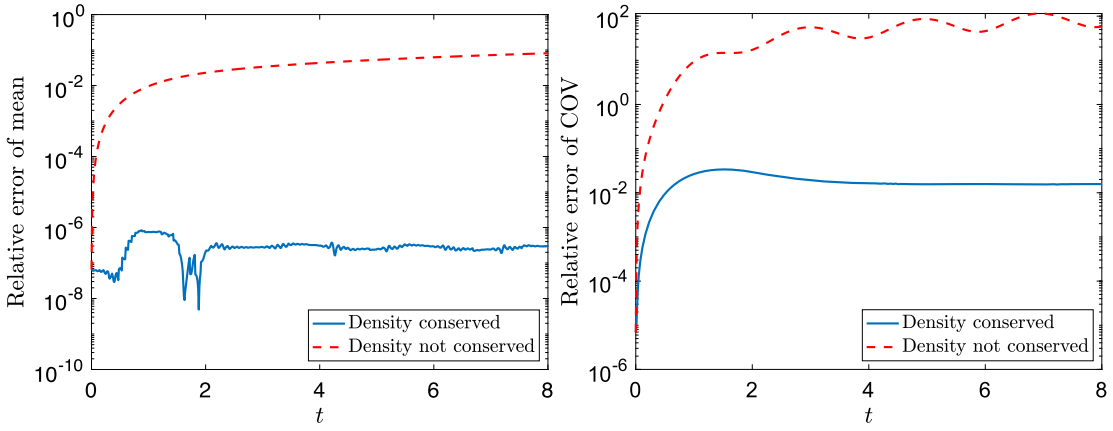

Fig. 2. Relative error for mean and covariance of the S-RONS solution applied to the Fokker–Planck equation for the harmonic trap using two Gaussians in the approximate solution ($r = 2$). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

where δ_{ij} denotes the Kronecker delta. The covariance matrix $\Sigma_{ij} - \bar{X}_i \bar{X}_j$ can then be computed using the solutions to equations (54) and (55).

We consider the same initial condition and parameter values as in Ref. [14]. More specifically, we take the Gaussian initial condition,

$$p(\mathbf{x}, 0) = (2\pi)^{-d/2} \det(\Sigma)^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right], \tag{56}$$

where the initial mean $\boldsymbol{\mu} \in \mathbb{R}^d$ is given by $\mu_i = 0.9 + 2.1(i-1)/(d-1)$ for $i = 1, \ldots, d$, and the initial covariance $\Sigma \in \mathbb{R}^{d \times d}$ is the diagonal matrix $\Sigma = \mathrm{diag}(0.1, 0.1, \ldots, 0.1)$. The remaining parameters are given by $a(t) = 1.25(\sin(\pi t) + 1.5)$, $\alpha = 0.25$, $d = 8$, and $\nu = 0.01$.

We approximate the solution of the Fokker–Planck equation (52) using symbolic RONS as described in section 3.2. For the shape-morphing approximate solution (14), we choose

$$\hat{p}(\mathbf{x}, \mathbf{q}(t)) = \sum_{i=1}^{r} A_i^2(t) \exp\left[-w_i^2(t) |\mathbf{x} - \mathbf{c}_i(t)|^2 \right], \tag{57}$$

where the mode function ϕ is a Gaussian, the shape parameters are $\boldsymbol{\beta}_i = (w_i, \mathbf{c}_i) \in \mathbb{R}^9$ and the amplitudes are $\alpha_i = A_i^2$. We square the amplitudes to ensure that the approximate PDF $\hat{p}$ is non-negative. The wights $w_i(t)$ control the standard deviation of each Gaussian since $|w_i|^{-1}$ is proportional to the standard deviation of the i-th Gaussian. Finally, the vector $\mathbf{c}_i(t) \in \mathbb{R}^d$ determines the i-th Gaussian's mode. Therefore, the parameters of the approximate solution are $\mathbf{q} = \{A_i, w_i, \mathbf{c}_i\}_{i=1}^r$, resulting in a total of $n = r(d+2)$ parameters.

Note that since the solution p is a PDF, its integral over the entire domain $\mathbb{R}^d$ must be equal to one for all times. This constitutes a conserved quantity for the Fokker–Planck equation which can be easily enforced in RONS. We ensure that the total probability of the approximate solution $\hat{p}$ is unity by enforcing the conserved quantity,

$$I_1(\mathbf{q}(t)) := \int_{\mathbb{R}^d} \hat{p}(\mathbf{x}, \mathbf{q}(t)) \, \mathrm{d}\mathbf{x} = \pi^4 \sum_{i=1}^{r} \frac{A_i^2(t)}{w_i^8(t)} = 1, \tag{58}$$

for all $t \geq 0$.

Assuming that the Hilbert space H is the space of square integrable functions over $\mathbb{R}^d$, we use the symbolic RONS with the Gaussian approximate solution (57) to form the RONS equation. Since our initial condition $p(\mathbf{x}, 0)$ is a Gaussian and the approximate solution is a sum of Gaussians, there are infinitely many choices of parameter $\mathbf{q}(0)$ with which the approximate solution can exactly represent the initial condition. We choose to represent the initial condition by giving all Gaussians in the approximate solution $\hat{p}$ the same mean and covariance as the initial condition, and then equally distributing the amplitude of the initial condition between each of the r Gaussians in the approximate solution. More explicitly, we choose initial parameter values $A_i^2(0) = (2\pi \times 0.1)^{-4} r^{-1}$, $w_i^2(0) = (2 \times 0.1)^{-1}$, and $\mathbf{c}_i(0) = \boldsymbol{\mu}$. With this choice of initial parameters, the metric tensor M is not invertible at the initial time and so we use the Moore-Penrose pseudoinverse M^+ when solving the RONS equation (6).

4.1.1. Symbolic RONS without regularization

First, we study solutions to RONS with only two modes ($r = 2$) in the Gaussian approximate solution (57). In this case, the resulting ODEs are not stiff and therefore regularization is not necessary. We first demonstrate the importance of enforcing conserved quantities in the reduced-order model. Fig. 2 shows the results both with and without enforcing that the total probability of the approximate solution remains constant; see equation (58). When enforcing constant total probability, RONS captures the true mean

Table 1
Comparison of computational time and accuracy for harmonic trap example using the adaptive sampling approach [14] and S-RONS. Two Gaussians are used in approximate solution for each simulation. Note that the symbolic computation only needs to be performed once; changing the initial condition or Fokker–Planck parameters does not require additional symbolic computation.

Harmonic Trap ($r = 2$)	Symbolic computation	Time integration	Relative error of mean	Relative error of covariance
Adaptive Sampling	none	189.1 minutes	$\approx 4 \times 10^{-3}$	≈ 2
Symbolic RONS	13.7 minutes	0.31 seconds	$\approx 3 \times 10^{-7}$	$\approx 10^{-2}$

with a relative error on the order of 10^{-6} and the covariance is captured with relative error of approximately 10^{-2} (solid blue curves in Fig. 2). In contrast, if the conservation of probability is not enforced, the relative error of the mean increases to about 10^{-1} and the covariance error reaches 10^{2} (dashed red curves in Fig. 2). Therefore, enforcing the conserved quantity (58) results in approximate solutions which are 4 to 5 orders of magnitude more accurate.

In addition to providing accurate solutions, the time integration for the RONS simulation takes only 0.31 seconds when using 2 Gaussians in the approximate solution. If we were to instead simulate many realizations of the SDE to approximate the PDF, the total computational time would be significantly higher.

As mentioned earlier, the Fokker–Planck equation (52) was also solved in [14], where they used an adaptive Monte Carlo sampling to estimate the inner products in RONS. In Table 1, we compare the computational time and accuracy between the adaptive sampling approach of [14] and our symbolic RONS, where both methods use two Gaussians in the approximate solution (57). For the adaptive sampling method, we use the code that was made publicly available by Bruna et al. [49]. This code uses a backwards Euler scheme for time integration together with stochastic gradient descent to solve the nonlinear system at every timestep, whereas our RONS simulations are integrated in time using an explicit scheme. As shown in Table 1, we see that RONS returns significantly faster and more accurate solutions than the adaptive sampling approach.

In particular, time integration using adaptive sampling takes approximately 189 minutes (more than 3 hours). The main computational cost comes from sampling and subsequent evaluation of the inner products, which has to be repeated at each time step. In contrast, time integration using symbolic RONS only takes 0.31 seconds since no sampling is required and the symbolic computations do not need to be repeated at every time step. There is the one-time cost of computing the integrals symbolically for RONS which takes approximately 13.7 minutes. However, the symbolic expressions for the RONS equation are obtained, we can integrate the equations from any initial condition without having to recompute the inner products. In other words, the solution from a different initial condition $p(\mathbf{x}, 0)$ can be obtained in approximately 0.31 seconds. In contrast, adaptive Monte Carlo simulations need to be repeated for every initial condition and therefore the numerical integration would again take hours if we were to change the initial condition.

In addition to being faster, symbolic RONS is also more accurate. As shown in Table 1, the mean of the solution is computed four orders of magnitude more accurately when using symbolic RONS compared to adaptive sampling. Moreover, the estimated covariance is two orders of magnitude more accurate when using symbolic RONS. The higher accuracy of symbolic RONS is not surprising since the inner products are computed exactly, whereas relatively large errors are accrued when adaptive Monte Carlo sampling is used.

4.1.2. Regularized symbolic RONS

Next we consider the effect of increasing the number of modes r. In particular, we consider the approximate solution (57) with $r = 30$ modes. As the number of parameters increase, the RONS equation becomes stiff, due to the fact that the metric tensor has a condition number in the order of 10^{20}. To address this issue, Bruna et al. [14] use an implicit time integration scheme. As mentioned in section 3.4, an alternative approach is to use regularization. Here, we use regularized symbolic RONS with the regularization parameter $\alpha = 10^{-3}$ and compare our results to the adaptive sampling method of [14] with their implicit time integrator [49]. Regularization lowers the condition number to around 10^{7}, which allows us to use an explicit time integrator and obtain more accurate results.

Fig. 3 shows the relative error of the mean and covariance when applying RONS to the Fokker–Planck PDE with $r = 30$ Gaussians in the approximate solution. The regularized symbolic RONS approximation matches the analytical solution well. As the solution evolves, the relative error of the mean settles around 10^{-6}. We see a similar behavior in the approximate solution's covariance, where the relative error settles around 3×10^{-4} as the solution evolves.

There is a short time period around $t = 0$ where the relative errors increase. This transient increase coincides with the time needed for the particles to settle in the harmonic trap. Initially the particles travel from their initial condition, but after a short time they settle in the harmonic trap and oscillate there. After this trapping, it becomes easier for the approximate solution to capture the true solution and therefore the relative error decreases. Note that this initial growth was absent when using only $r = 2$ modes, where no regularization was required. This demonstrates the fact that, although regularized RONS speeds up the time integration, it can lead to a deterioration of the accuracy. Nonetheless, the error is relatively small. In fact, as shown in Fig. 3, regularized symbolic RONS is about three orders of magnitude more accurate than adaptive sampling with implicit time integration.

In Table 2, we compare the computational time and accuracy of the adaptive sampling approach of [14] and regularized symbolic RONS. As in the case of 2 Gaussians, the 30-mode approximation using symbolic RONS significantly outperforms the results from adaptive sampling in both computational speed and accuracy. In particular, time integration using symbolic RONS takes slightly over

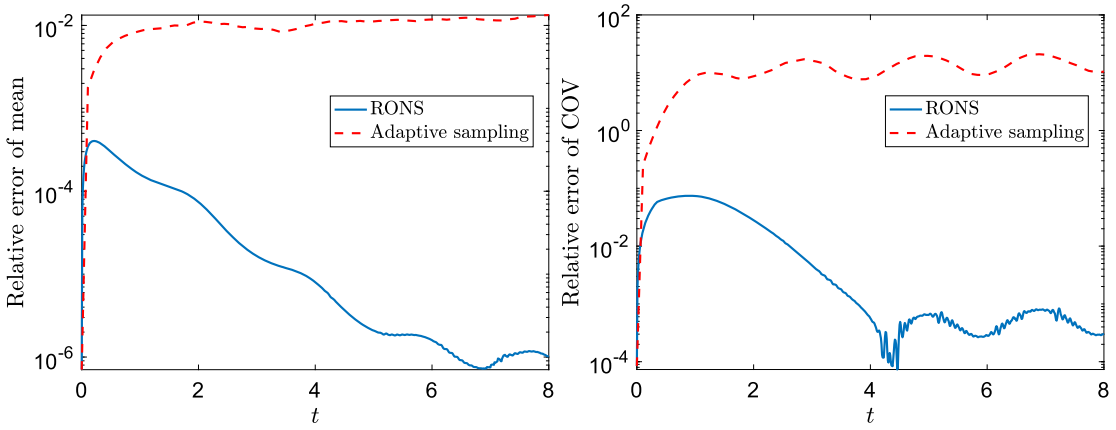

Fig. 3. Relative error for mean and covariance of the S-RONS solution applied to the Fokker–Planck equation corresponding to the harmonic trap. For comparison, the same relative errors are shown for the adaptive sampling method [14]. Thirty Gaussians are used in the approximate solution ($r = 30$).

Table 2
Comparison of computational speed and accuracy for the harmonic trap with $r = 30$ modes. We compare the adaptive sampling approach [14] to our proposed method of regularized symbolic RONS.

Harmonic Trap ($r = 30$)	Symbolic computation	Time integration	Relative error of mean	Relative error of covariance
Adaptive Sampling	none	24.4 hours	$\approx 10^{-2}$	≈ 10
Symbolic RONS	0 minutes	64.2 minutes	$\approx 10^{-6}$	$\approx 3 \times 10^{-4}$

one hour, whereas adaptive sampling takes over 24 hours and yet yields lower accuracy. The high computational cost of adaptive sampling is attributed to the fact that RONS inner products must be reevaluated at every time step. Furthermore, since an implicit time integration scheme is used, a nonlinear equation needs to be solved at every time step which adds to the computational cost. In contrast, regularized RONS uses an explicit scheme which does not require solving a nonlinear equation.

We emphasize that the computational cost of symbolic integration is independent of the number of modes r when using symbolic RONS as described in section 3.2. More specifically, the symbolic expressions from $r = 2$ modes can be used to evaluate all RONS terms when $r = 30$, without requiring additional symbolic computation, thus the zero symbolic computational time reported in Table 2.

We conclude this section by remarking that S-RONS is computationally feasible for this problem because of the method developed in section 3.2. The Gaussian approximate solution (57) has $r = 30$ modes with $K = 10$ parameters in each mode, so that a brute force approach to RONS would have required symbolic computation of $300(300 + 3)/2 = 45,450$ integrals. In contrast, S-RONS requires symbolic computation of only $10(10 + 3)/2 = 65$ integrals (see Theorem 1).

4.2. Kuramoto–Sivashinsky equation

In this section, we consider the Kuramoto–Sivashinsky (KS) equation and approximate its solution with a shallow neural network with hyperbolic tangent activation functions. In this case, obtaining symbolic expressions for the RONS equation is not possible. Therefore, we use collocation RONS as described in section 3.3. The KS equation was also solved in [13] using a Monte Carlo method to approximate the RONS equations. We compare our results with this Monte Carlo approach.

The Kuramoto–Sivashinsky equation is given by

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^4 u}{\partial x^4}, \quad u(x,0) = u_0(x), \tag{59}$$

where the solution $u(x,t)$ is assumed to have periodic boundary conditions over the domain $x \in [-\ell, \ell]$. We consider the same set-up used in [13]. In particular, we set $\ell = 10$ and consider the initial condition

$$u_0(x) = -\sin\left(\frac{\pi x}{\ell}\right), \quad x \in [-\ell, \ell]. \tag{60}$$

The corresponding solutions of the KS equation are known to exhibit chaotic behavior [50–53]. As the ground truth, we use direct numerical simulations (DNS) using a Fourier pseudo-spectral method with 2^7 modes. Time integration is carried out using a fourth-order Runge–Kutta scheme with adaptive time-stepping [44]. Integrating the solution to time $t = 40$ using this DNS method takes 0.6 seconds. The Fourier pseudo-spectral method in one dimension is extremely efficient. Our objective in this section is not to outperform this method. Instead, our goal is to compare our results against the Monte Carlo method of [13] and show that C-RONS

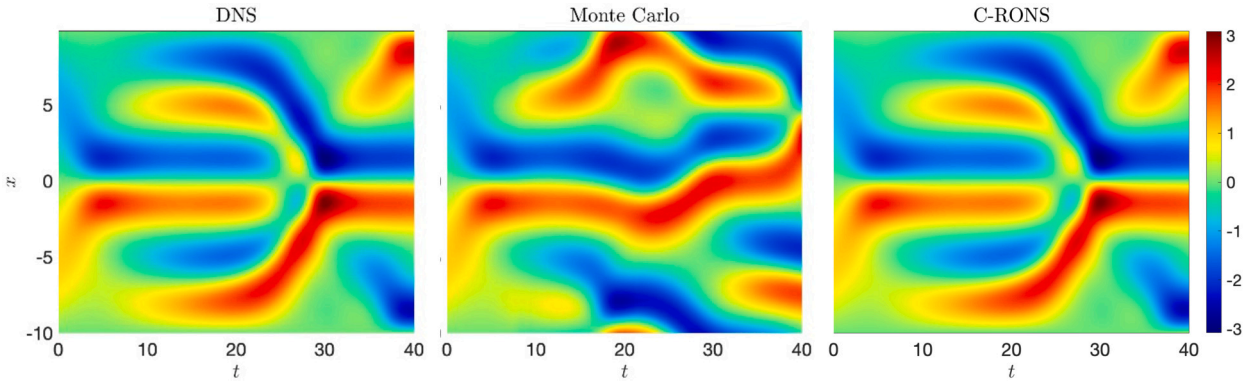

Fig. 4. Simulation of Kuramoto–Sivashinsky equation using direct numerical simulation (left panel), Monte Carlo integration to approximate the RONS equation (middle panel), and regularized collocation RONS (right panel).

Table 3
Computational time and accuracy for the Kuramoto–Sivashinsky equation using Monte Carlo integration and C-RONS.

Kuramoto–Sivashinsky	Monte Carlo	Regularized Monte Carlo	Regularized C-RONS
Computational time	222.3 minutes	118.3 seconds	38.6 seconds
Relative error	1.34	9.3×10^{-2}	3.3×10^{-2}

together with regularization significantly reduces the computational time and simultaneously increases the accuracy compared to the Monte Carlo method.

For the KS equation, the choice of appropriate approximate solution is not as clear as in the Fokker–Planck example. Motivated by architectures typically used in neural networks, we choose an approximate solution which is a shallow neural network with hyperbolic tangent activation function,

$$\hat{u}(x,\mathbf{q}) = \sum_{i=1}^{r} A_i(t) \tanh\left(w_i(t)s_i(x) + d_i(t)\right), \tag{61}$$

where

$$s_i(x) = \sin\left(\frac{\pi x}{\ell} + c_i(t)\right), \tag{62}$$

is a nonlinear coordinate transformation to ensure that the approximate solution $\hat{u}$ satisfies the periodic boundary conditions over the domain $[-\ell,\ell]$. This is a common technique discussed further in [54,13,55]. The approximate solution (61) can be thought of as a neural network with a single hidden layer and r nodes. Each node contains the amplitude $A_i(t)$, the weight $w_i(t)$, and the biases $c_i(t)$ and $d_i(t)$. These form the parameters of the approximate solution, $\mathbf{q} = \{A_i, w_i, c_i, d_i\}_{i=1}^{r}$, where the shape parameters comprise $\boldsymbol{\beta}_i = (w_i, c_i, d_i)$. Typically these parameters would be obtained by a training process. However, as Du and Zaki [13] observe, no training is required; the parameters can be evolved using the RONS equation (12).

There does not exist a choice of parameters such that the sine wave initial condition u_0 can be exactly represented with our choice of approximate solution (61). To determine the initial parameter values $\mathbf{q}(0)$, we perform a least-squares fitting of the approximate solution to the initial condition, i.e., we set

$$\mathbf{q}(0) = \underset{\mathbf{q}\in\mathbb{R}^n}{\operatorname{argmin}} \|\hat{u}(\cdot,\mathbf{q}) - u_0\|_{L^2}^2. \tag{63}$$

We solve this optimization problem once at the initial time to obtain the parameters $\mathbf{q}(0)$. The corresponding approximation error is less than 2×10^{-7}.

We then evolve the approximate solution (61) with 10 modes ($r = 10$) using collocation RONS method of section 3.3. We compare our results with the Monte Carlo approximation of the inner products as proposed in [13]. For collocation RONS we use 2^7 equidistant collocation points. For the Monte Carlo approach, we use 2^7 samples uniformly distributed throughout the domain $[-\ell,\ell]$. When applying collocation RONS, we use Tikhonov regularization as described in section 3.4, with a regularization parameter value of $\alpha = 10^{-5}$. Both methods, collocation RONS and the Monte Carlo approach, use Matlab's `ode45` for time integration.

In Fig. 4, we compare the approximate solutions produced by collocation RONS and the Monte Carlo approach. Collocation RONS is in excellent agreement with the DNS solution, whereas the Monte Carlo approach quickly diverges from the DNS solution after approximately 10 time units. Table 3 compares the computational time of these two methods. The Monte Carlo approach takes 222.3 minutes to run while the regularized collocation RONS takes only 38.6 seconds.

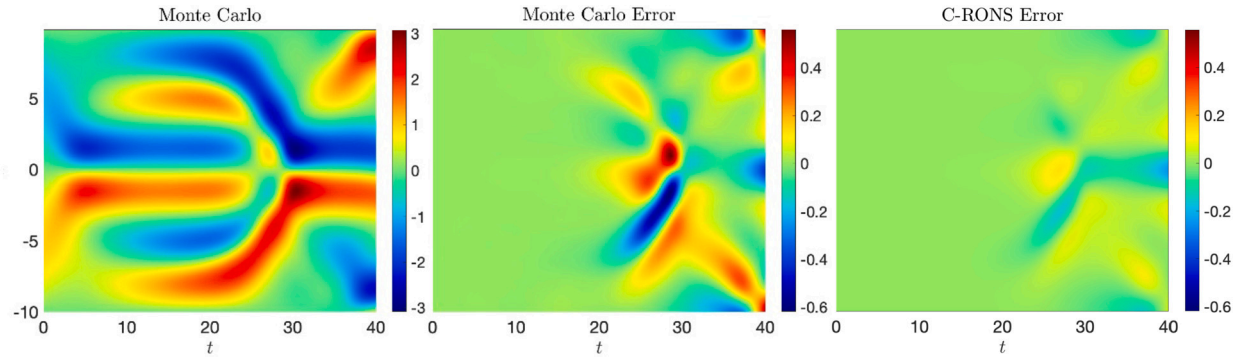

Fig. 5. Simulation of Kuramoto–Sivashinsky equation using regularized Monte Carlo method (left panel). Error of the regularized Monte Carlo method (middle panel) as compared to the DNS solution. Error of regularized C-RONS is shown for comparison (right panel).

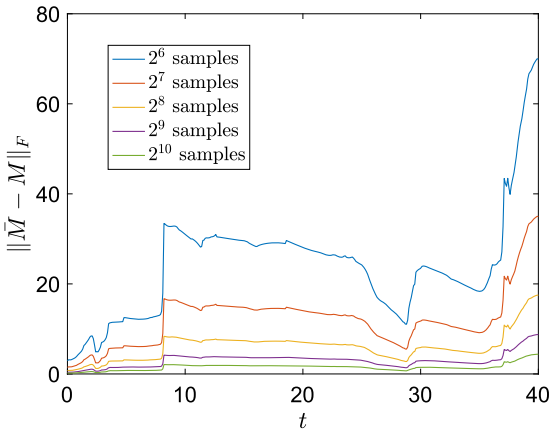

Fig. 6. Frobenius error of the Monte Carlo approximation $\bar{M}$ to the metric tensor for the Kuramoto–Sivashinsky equation when using increasing number of samples. Samples are all uniformly distributed on the domain $x \in [-10, 10]$ and the true value of M is approximated by using trapezoidal rule with 10^4 samples.

The Monte Carlo method is significantly slower mainly because of the stiffness of the resulting ODEs due to the high condition number of the metric tensor $\bar{M}$. Without regularization, the condition number of $\bar{M}$ is at least 10^{16}, whereas regularized C-RONS lowers this condition number to $10^8 - 10^9$. To deal with stiffness of the Monte Carlo method, we also regularize the Monte Carlo approach by applying Tikhonov regularization to (36). As shown in Table 3, regularization significantly reduces the computational time of the Monte Carlo method from 222.3 minutes to 118.3 seconds. Although this is still 3 times slower than regularized C-RONS, Tikhonov regularization greatly reduces the computational cost of the Monte Carlo method.

Interestingly, regularization also increases the accuracy of the Monte Carlo method. In Fig. 5, we show the error between regularized Monte Carlo method and the DNS solution. For comparison, we also show the error for regularized C-RONS. The Monte Carlo approximation uses the regularization parameter $\alpha = 10^{-8}$ since larger values of α led to numerical results which deviated significantly from the DNS. Comparing Figs. 4 and 5, we first note that regularization greatly improves the accuracy of the Monte Carlo approach. However, the solution obtained by regularized C-RONS still yields lower errors and only takes a third of the computational time of regularized Monte Carlo (see Table 3). This is largely due to the poor conditioning of the matrix $\bar{M}$ which appears in the Monte Carlo approximation (see Remark 2).

The approximation errors for both methods grow over time, which is expected for a chaotic system as any error in approximating the initial condition will grow as the solution evolves. Even for short timescales, it is surprising that the Monte Carlo method is able to approximate the solution given that only 128 samples are used. In fact, the Monte Carlo integration (35) with 128 samples is quite inaccurate. For instance, Fig. 6 shows the Frobenius error between the Monte Carlo approximation $\bar{M}$ and the true metric tensor M. For 128 samples, the error start around 2 and grows to approximately 18 in less than 10 time units. As a result, there is considerable error in the Monte Carlo approximation of the RONS equations when only 128 samples are used. Yet, the regularized Monte Carlo method returns a rather accurate solution as shown in Fig. 5. Theorem 2 is the key to resolving this seeming paradox. The RONS equation approximated by Monte Carlo integration is equivalent to collocation RONS. Therefore, the Monte Carlo approximation is in fact minimizing the residual function (24) at the sampled points. This allows the Monte Carlo approach to obtain an accurate solution with a small sample size, despite being inaccurate as an integration method.

The choice of hyperbolic tangent as the activation function for the approximate solution (61) is somewhat arbitrary. For example, we can alternatively use the logistic function to define the approximate solution,

$$\hat{u}(x,\mathbf{q}(t)) = \sum_{i=1}^{r} \frac{A_i(t)}{1+\exp[-w_i(t)s_i(x)+d_i(t)]}. \tag{64}$$

To evolve the parameters (A_i, w_i, d_i), we apply C-RONS using $r = 10$ modes, $N = 2^7$ collection points uniformly distributed throughout the domain, and regularization parameter $\alpha = 10^{-5}$. We observe that the logistic activation function performs slightly worse than the hyperbolic tangent activation function. Time integration of the ODEs takes 155.2 seconds, whereas with hyperbolic tangent it took 38.6 seconds. The relative error over the space-time domain increases to 5.8×10^{-2} from 3.3×10^{-2}. Of course, these results would change with the regularization parameter. For example, when using $\alpha = 5 \times 10^{-5}$ with approximate solution (64), the time integration of the ODEs takes 101.6 seconds and the relative error decreases to 2.2×10^{-2}.

We conclude this section by noting that Du and Zaki [13] solved the same KS equation using a deep neural network with 4 layers, 20 nodes per layer, and 10^3 samples to approximate the RONS integrals. In spite of their network being wider and deeper, the accuracy of our shallow neural network is better. For instance, at time $t = 20$, our relative error is approximately 6×10^{-3}, whereas their relative error is slightly above 10^{-2} (see figure 10 in [13]). We attribute the poorer performance of the deep neural network to the higher condition number of the metric tensor as discussed in section 3.3.1 which makes time integration more challenging.

5. Conclusions

Despite being in its infancy, RONS has already emerged as an effective method both for reduced-order modeling [1,24] and for solving PDEs with neural networks without requiring any training [13,14]. However, brute force construction of the RONS equations requires the evaluation of $\mathcal{O}(n^2)$ integrals, where n denotes the number of time-dependent parameters in the approximate solution. Therefore, this approach becomes computationally prohibitive when a large number of parameters are required to accurately approximate the solution of the PDE. Making matters worse, the resulting ODEs tend to become stiff as the number of parameters grows. Here, we developed three methods to address these computational bottlenecks: symbolic RONS, collocation RONS, and regularized RONS.

Using symbolic computing and exploiting the structure of the RONS equations, symbolic RONS (or S-RONS, for short) drastically reduces the computational cost from $\mathcal{O}(n^2)$ to $\mathcal{O}(K^2)$ where $K \ll n$ is independent of the number of parameters n. Furthermore, since the equations are constructed symbolically, the S-RONS integrals do not need to be recomputed during time stepping; rather they can be evaluated by direct substitution of the updated parameters into the symbolic expressions. Applying S-RONS to the Fokker–Planck equation, we obtained 14-23 times speedup in comparison to the adaptive sampling method of [14]. In addition, the accuracy of the solutions increased by several orders of magnitude.

We also developed collocation RONS (or C-RONS, for short) in case symbolic computation of the integrals are not feasible. Rather than minimizing the error between evolution of the approximate solution and dynamics of the governing PDE over the entire spatial domain, C-RONS minimizes this error over a set of prescribed collocation points. Since this method does not require any symbolic computation, it is applicable to any choice of the approximate solution and any form of the PDE. We also proved that, in exact arithmetic, C-RONS is equivalent to the Monte Carlo method proposed in [13]. However, from a numerical standpoint, C-RONS is significantly better conditioned than the Monte Carlo approximation and thus numerically more stable. Applying C-RONS to the Kuramoto–Sivashinsky PDE, we observed a 300 times speedup in the computation, while simultaneously reducing the error by two orders of magnitude. Although here we only considered equidistant collocation points, choosing them on an unstructured grid or even an adaptive grid is certainly a possibility.

The RONS equations take the form of a system of nonlinear ODEs which evolve the parameters of the approximate solution. These ODEs tend to become stiff as the number of parameters n increases. As a result, one either has to take exceedingly small time steps or use implicit time integration schemes. To address this issue, we introduced regularized versions of S-RONS and C-RONS that allow fast time integration even with explicit schemes. Regularized RONS adds Tikhonov penalization to the underlying minimization problem such that the resulting ODEs are not stiff. Applying this regularization to both Fokker–Planck and Kuramoto–Sivashinsky equations led to significant speedup without adversely affecting the accuracy of the solutions.

Here we chose the regularization parameter α in an ad hoc manner. Our preliminary experimentation (not shown here) indicates that commonly used methods for determining a regularization parameter fail when applied to RONS. For instance, generalized cross-validation [37–39] returned regularization parameters which are too small and consequently led to slow time integration of the RONS equations. Conversely, the L-curve method [34,36] and quasi-optimality criterion [40] returned regularization parameters which are too large and led to inaccurate approximate solutions. Therefore, tailor-made methods for identifying the optimal regularization parameter for RONS are needed.

Another open problem is related to the number of terms r in the approximation solution (1). In the present work, we chose the number of required terms in an ad hoc manner. A systematic method for choosing r requires error estimates for a given PDE. Since the approximate solution depends nonlinearly on its parameters, classical error estimates are not immediately applicable. Therefore, the derivation of such estimates remains an interesting open problem.

The computational methods developed here pave the way for RONS to be used as a shape-morphing spectral method for efficient numerical solution of nonlinear PDEs. In contrast to existing spectral methods where the modes are static in time, the RONS-based spectral methods will allow the modes to change shape and adapt to the solution of the PDE. As a result, these methods will be specially suitable for solving PDEs with localized features (e.g., sharp gradients or shocks) and for advection-dominated PDEs. Future work will explore this avenue by determining the appropriate choice of the shape-morphing modes and carrying out error analysis of the resulting spectral method.

Funding

This work was supported by the National Science Foundation through the award DMS-2208541.

CRediT authorship contribution statement

William Anderson: Investigation, Numerical simulations, Writing the first draft of the paper. **Mohammad Farazmand:** Conceptualization and supervision, Investigation, Funding acquisition, Reviewing and editing the paper.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Appendix A. Proof of Theorem 3

We first note that by taking a time derivative, we can write the constraints in (39) as

$$\frac{\mathrm{d}}{\mathrm{d}t} I_k(\mathbf{q}(t)) = \langle \nabla I_k(\mathbf{q}), \dot{\mathbf{q}} \rangle = 0, \quad k = 1, 2, ..., m. \tag{65}$$

Introducing the Lagrange multipliers $\hat{\boldsymbol{\lambda}} = (\hat{\lambda}_1, ..., \hat{\lambda}_m)^\top \in \mathbb{R}^m$, we define the augmented cost function

$$\hat{\mathcal{J}}_c(\mathbf{q}, \dot{\mathbf{q}}, \hat{\boldsymbol{\lambda}}) := \hat{\mathcal{J}}(\mathbf{q}, \dot{\mathbf{q}}) + \sum_{k=1}^{m} \hat{\lambda}_k \langle \nabla I_k(\mathbf{q}), \dot{\mathbf{q}} \rangle. \tag{66}$$

If a solution to the constrained optimization problem (39) exists, the partial derivatives of $\hat{\mathcal{J}}_c$ with respect to q_i and $\hat{\lambda}_k$ must vanish at the minimizer. This yields

$$\nabla_{\dot{\mathbf{q}}} \hat{\mathcal{J}} + \sum_{k=1}^{m} \hat{\lambda}_k \nabla I_k = 0, \tag{67a}$$

$$\langle \nabla I_1(\mathbf{q}), \dot{\mathbf{q}} \rangle = \langle \nabla I_2(\mathbf{q}), \dot{\mathbf{q}} \rangle = ... = \langle \nabla I_m(\mathbf{q}), \dot{\mathbf{q}} \rangle = 0. \tag{67b}$$

We already know (see [1], Theorem 1) that our original cost functional satisfies $\nabla_{\dot{\mathbf{q}}} \mathcal{J} = M(\mathbf{q})\dot{\mathbf{q}} - \mathbf{f}(\mathbf{q})$. Similarly, we can calculate the gradient of our regularized cost functional to obtain $\nabla_{\dot{\mathbf{q}}} \hat{\mathcal{J}} = (M(\mathbf{q}) + \Gamma^\top \Gamma)\dot{\mathbf{q}} - \mathbf{f}(\mathbf{q})$.

We note that the matrix $(M(\mathbf{q}) + \Gamma^\top \Gamma)$ is symmetric positive definite. This is because the metric tensor M is symmetric positive semi-definite, and $\Gamma^\top \Gamma$ is symmetric positive definite due to the assumption that Γ is full column rank. Therefore, $(M(\mathbf{q}) + \Gamma^\top \Gamma)$ is symmetric positive definite and thus invertible. For notational convenience, we define the regularized metric tensor $\hat{M}(\mathbf{q}) := (M(\mathbf{q}) + \Gamma^\top \Gamma)$.

Using the fact that $\hat{M}$ is invertible, equation (67a) yields

$$\dot{\mathbf{q}} = \hat{M}^{-1}(\mathbf{q}) \left[\mathbf{f}(\mathbf{q}) - \sum_{k=1}^{m} \hat{\lambda}_k \nabla I_k(\mathbf{q}) \right]. \tag{68}$$

Substituting this expression into (67b), we obtain m equations

$$\sum_{k=1}^{m} \hat{\lambda}_k \langle \nabla I_i, \hat{M}^{-1}(\mathbf{q}) \nabla I_k \rangle = \langle \nabla I_i, \hat{M}^{-1}(\mathbf{q}) \mathbf{f} \rangle, \quad i = 1, 2, ..., m. \tag{69}$$

Equation (69) can be written as the system of equations $\hat{C}\hat{\boldsymbol{\lambda}} = \hat{\mathbf{b}}$, where $\hat{C}$ is the *regularized constraint matrix* with entries given by

$$\hat{C}_{ij} = \langle \nabla I_j, \hat{M}^{-1} \nabla I_i \rangle, \quad i, j \in \{1, 2, \cdots, m\}, \tag{70}$$

and the vector $\hat{\mathbf{b}} = (\hat{b}_1, \hat{b}_2, \cdots, \hat{b}_m)^\top \in \mathbb{R}^m$ is given by

$$\hat{b}_i = \langle \nabla I_i, \hat{M}^{-1} \mathbf{f} \rangle, \quad i = 1, 2, \cdots, m. \tag{71}$$

The matrix $\hat{C}$ is symmetric positive definite, provided that the constraint gradients $\nabla I_1(\mathbf{q}), \nabla I_2(\mathbf{q}), \cdots, \nabla I_m(\mathbf{q})$ are linearly independent (see [1], Lemma 2). Thus, the Lagrange multipliers $\hat{\boldsymbol{\lambda}}$ are the uniquely determined by $\hat{\boldsymbol{\lambda}} = \hat{C}^{-1}\hat{\mathbf{b}}$. Therefore, $\dot{\mathbf{q}}$ must satisfy equation (68) where the Lagrange multipliers are determined by solving the system of equations $\hat{C}\hat{\boldsymbol{\lambda}} = \hat{\mathbf{b}}$.

References

[1] W. Anderson, M. Farazmand, Evolution of nonlinear reduced-order solutions for PDEs with conserved quantities, SIAM J. Sci. Comput. 44 (2022) A176–A197.
[2] P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems, SIAM Rev. 57 (4) (2015) 483–531, https://doi.org/10.1137/130932715.
[3] C.W. Rowley, S.T.M. Dawson, Model reduction for flow analysis and control, Annu. Rev. Fluid Mech. 49 (1) (2017) 387–417, https://doi.org/10.1146/annurev-fluid-010816-060042.
[4] T.A.A. Adcock, R.H. Gibbs, P.H. Taylor, The nonlinear evolution and approximate scaling of directionally spread wave groups on deep water, Proc. R. Soc. A 468 (2145) (2012) 2704–2721, https://doi.org/10.1098/rspa.2012.0029.
[5] T.A.A. Adcock, P.H. Taylor, Focusing of unidirectional wave groups on deep water: an approximate nonlinear Schrödinger equation-based model, Proc. R. Soc. A 465 (2110) (2009) 3083–3102, https://doi.org/10.1098/rspa.2009.0224.
[6] W. Cousins, T.P. Sapsis, Unsteady evolution of localized unidirectional deep-water wave groups, Phys. Rev. E 91 (6) (2015) 063204.
[7] V.M. Pérez-García, H. Michinel, J.I. Cirac, M. Lewenstein, P. Zoller, Low energy excitations of a Bose-Einstein condensate: a time-dependent variational analysis, Phys. Rev. Lett. 77 (1996) 5320–5323, https://doi.org/10.1103/PhysRevLett.77.5320.
[8] V.P. Ruban, Anomalous wave as a result of the collision of two wave groups on the sea surface, JETP Lett. 102 (10) (2015) 650–654.
[9] V.P. Ruban, Gaussian variational ansatz in the problem of anomalous sea waves: comparison with direct numerical simulation, J. Exp. Theor. Phys. 120 (5) (2015) 925–932.
[10] P.K. Newton, The N-Vortex Problem: Analytical Techniques, Applied Mathematical Sciences, vol. 145, Springer, 2001.
[11] J.T. Beale, A. Majda, High order accurate vortex methods with explicit velocity kernels, J. Comput. Phys. 58 (2) (1985) 188–208.
[12] G.-H. Cottet, P.D. Koumoutsakos, Vortex Methods: Theory and Practice, Cambridge University Press, 2000.
[13] Y. Du, T.A. Zaki, Evolutional deep neural network, Phys. Rev. E 104 (2021) 045303.
[14] J. Bruna, B. Peherstorfer, E. Vanden-Eijnden, Neural Galerkin Scheme with Active Learning for High-Dimensional Evolution Equations, 2022.
[15] T.J. Bridges, S. Reich, Numerical methods for Hamiltonian PDEs, J. Phys. A, Math. Gen. 39 (19) (2006) 5287, https://doi.org/10.1088/0305-4470/39/19/S02.
[16] R.I. McLachlan, G.R.W. Quispel, Geometric integrators for ODEs, J. Phys. A, Math. Gen. 39 (19) (2006) 5251, https://doi.org/10.1088/0305-4470/39/19/S01.
[17] P. Cifani, M. Viviani, E. Luesink, K. Modin, B.J. Geurts, Casimir preserving spectrum of two-dimensional turbulence, Phys. Rev. Fluids 7 (2022) L082601, https://doi.org/10.1103/PhysRevFluids.7.L082601.
[18] L. Peng, K. Mohseni, Symplectic model reduction of Hamiltonian systems, SIAM J. Sci. Comput. 38 (1) (2016) A1–A27, https://doi.org/10.1137/140978922.
[19] K. Carlberg, Y. Choi, S. Sargsyan, Conservative model reduction for finite-volume models, J. Comput. Phys. (ISSN 0021-9991) 371 (2018) 280–314.
[20] H. Babaee, T.P. Sapsis, A minimization principle for the description of modes associated with finite-time instabilities, Proc. R. Soc. A 472 (2186) (2016).
[21] H. Babaee, M. Farazmand, G. Haller, T.P. Sapsis, Reduced-order description of transient instabilities and computation of finite-time Lyapunov exponents, Chaos 27 (6) (2017) 063103.
[22] M. Farazmand, T.P. Sapsis, Dynamical indicators for the prediction of bursting phenomena in high-dimensional systems, Phys. Rev. E 94 (2016) 032212, https://doi.org/10.1103/PhysRevE.94.032212.
[23] M. Donello, M.H. Carpenter, H. Babaee, Computing sensitivities in evolutionary systems: a real-time reduced order modeling strategy, SIAM J. Sci. Comput. 44 (1) (2022) A128–A149, https://doi.org/10.1137/20M1388565.
[24] W. Anderson, M. Farazmand, Shape-morphing reduced-order models for nonlinear Schrödinger equations, Nonlinear Dyn. 108 (2022) 2889–2902, https://doi.org/10.1007/s11071-022-07448-w.
[25] A.J. Majda, Y. Yuan, Fundamental limitations of ad hoc linear and quadratic multi-level regression models for physical systems, Discrete Contin. Dyn. Syst., Ser. B (ISSN 1531-3492) 17 (4) (2012) 1333–1363, https://doi.org/10.3934/dcdsb.2012.17.1333, http://aimsciences.org//article/id/b2ad36d0-8c41-494b-8908-7ebea4aa23a9.
[26] P.E. Gill, W. Murray, M.H. Wright, Numerical Linear Algebra and Optimization, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021.
[27] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (4) (1989) 303–314.
[28] K. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Netw. 2 (3) (1989) 183–192.
[29] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 3 (2) (1991) 246–257, https://doi.org/10.1162/neco.1991.3.2.246.
[30] G.E. Karniadakis, S. Sherwin, Spectral/hp Element Methods for Computational Fluid Dynamics, Oxford University Press, Oxford, UK, 2005.
[31] I.C.F. Ipsen, Numerical Matrix Analysis, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2009.
[32] P.S. Keller, Chaotic behavior of Newton's method, Real Anal. Exch. 18 (2) (1992) 490–507.
[33] M. Farazmand, An adjoint-based approach for finding invariant solutions of Navier-Stokes equations, J. Fluid Mech. 795 (2016) 278–312, https://doi.org/10.1017/jfm.2016.203.
[34] D. Calvetti, S. Morigi, L. Reichel, F. Sgallari, Tikhonov regularization and the l-curve for large discrete ill-posed problems, J. Comput. Appl. Math. 123 (1) (2000) 423–446.
[35] G.H. Golub, P.C. Hansen, D.P. O'Leary, Tikhonov regularization and total least squares, SIAM J. Matrix Anal. Appl. 21 (1) (1999) 185–194.
[36] P.C. Hansen, The l-Curve and Its Use in the Numerical Treatment of Inverse Problems, 1999.
[37] P. Craven, G. Wahba, Smoothing noisy data with spline functions, Numer. Math. 31 (4) (1978) 377–403.
[38] G.H. Golub, M. Heath, G. Wahba, Generalized cross-validation as a method for choosing a good ridge parameter, Technometrics (ISSN 0040-1706) 21 (2) (1979) 215–223.
[39] C.R. Vogel, Computational Methods for Inverse Problems, Society for Industrial and Applied Mathematics, 2002.
[40] Frank Bauer, Stefan Kindermann, The quasi-optimality criterion for classical inverse problems, Inverse Probl. 24 (3) (2008) 035002, https://doi.org/10.1088/0266-5611/24/3/035002.
[41] F. Bauer, M. Reiß, Regularization independent of the noise level: an analysis of quasi-optimality, Inverse Probl. 24 (5) (2008) 055009, https://doi.org/10.1088/0266-5611/24/5/055009.
[42] A.N. Tikhonov, V.B. Glasko, Use of the regularization method in non-linear problems, USSR Comput. Math. Math. Phys. (ISSN 0041-5553) 5 (3) (1965) 93–107, https://doi.org/10.1016/0041-5553(65)90150-3.
[43] K. Ito, B. Jin, T. Takeuchi, A regularization parameter for nonsmooth Tikhonov regularization, SIAM J. Sci. Comput. 33 (3) (2011) 1415–1438, https://doi.org/10.1137/100790756.
[44] J.R. Dormand, P.J. Prince, A family of embedded Runge–Kutta formulae, J. Comput. Appl. Math. 6 (1) (1980) 19–26.
[45] L.F. Shampine, M.W. Reichelt, The Matlab ode suite, SIAM J. Sci. Comput. 18 (1) (1997) 1–22, https://doi.org/10.1137/S1064827594276424.
[46] J. Sirignano, K. Spiliopoulos, DGM: a deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1339–1364, https://doi.org/10.1016/j.jcp.2018.08.029.
[47] J. Han, A. Jentzen , W. E, Solving high-dimensional partial differential equations using deep learning, Proc. Natl. Acad. Sci. 115 (34) (2018) 8505–8510, https://doi.org/10.1073/pnas.1718942115.
[48] W. Anderson, M. Farazmand, Fisher information and shape-morphing modes for solving the Fokker–Planck equation in higher dimensions, arXiv:2306.03749, 2023, https://doi.org/10.48550/arXiv.2306.03749.

[49] J. Bruna, B. Peherstorfer, E. Vanden-Eijnden, Github repository: neural Galerkin with active learning for high-dimensional evolution equations, https://github.com/pehersto/ng, 2022.
[50] J.M. Hyman, B. Nicolaenko, The Kuramoto–Sivashinsky equation: a bridge between PDE's and dynamical systems, Phys. D: Nonlinear Phenom. (ISSN 0167-2789) 18 (1) (1986) 113–126.
[51] I.G. Kevrekidis, B. Nicolaenko, J.C. Scovel, Back in the saddle again: a computer assisted study of the Kuramoto–Sivashinsky equation, SIAM J. Appl. Math. 50 (3) (1990) 760–790, https://doi.org/10.1137/0150045.
[52] P. Cvitanović, R.L. Davidchack, E. Siminos, On the state space geometry of the Kuramoto–Sivashinsky flow in a periodic domain, SIAM J. Appl. Dyn. Syst. 9 (1) (2010) 1–33, https://doi.org/10.1137/070705623.
[53] J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach, Phys. Rev. Lett. 120 (2018) 024102.
[54] S. Dong, N. Ni, A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks, J. Comput. Phys. 435 (2021) 110242.
[55] A. Yazdani, L. Lu, M. Raissi, G.E. Karniadakis, Systems biology informed deep learning for inferring parameters and hidden dynamics, PLoS Comput. Biol. 16 (11) (2020) 1–19.