SIAM J. SCI. COMPUT. © 2020 Society for Industrial and Applied Mathematics
Vol. 42, No. 2, pp. A1116–A1146

SNS: A SOLUTION-BASED NONLINEAR SUBSPACE METHOD FOR TIME-DEPENDENT MODEL ORDER REDUCTION*

YOUNGSOO CHOI†, DESHAWN COOMBS‡, AND ROBERT ANDERSON†

Abstract. Several reduced order models have been successfully developed for nonlinear dynamical systems. To achieve a considerable speed-up, a hyper-reduction step is needed to reduce the computational complexity due to nonlinear terms. Many hyper-reduction techniques require the construction of nonlinear term basis, which introduces a computationally expensive offline phase. A novel way of constructing nonlinear term basis within the hyper-reduction process is introduced. In contrast to the traditional hyper-reduction techniques where the collection of nonlinear term snapshots is required, the SNS method avoids collecting the nonlinear term snapshots. Instead, it uses the solution snapshots that are used for building a solution basis, which enables avoiding an extra data compression of nonlinear term snapshots. As a result, the SNS method provides a more efficient offline strategy than the traditional model order reduction techniques, such as the DEIM, GNAT, and ST-GNAT methods. The SNS method is theoretically justified by the conforming subspace condition and the subspace inclusion relation. It is useful for model order reduction of large-scale nonlinear dynamical problems to reduce the offline cost. It is especially useful for ST-GNAT that has shown promising results, such as a good accuracy with a considerable online speed-up for hyperbolic problems in a recent paper by Choi and Carlberg [*SIAM J. Sci. Comput.*, 41 (2019), pp. A26–A58], because ST-GNAT involves an expensive offline cost related to collecting nonlinear term snapshots. Error analysis for the SNS method is presented. Numerical results support that the accuracy of the solution from the SNS method is comparable to the traditional methods and a considerable speed-up (i.e., a factor of two to a hundred) is achieved in the offline phase.

Key words. nonlinear model order reduction, hyper-reduction, nonlinear dynamical system, subspace inclusion, nonlinear term basis, time integrator

AMS subject classifications. 15A23, 35K05, 35N20, 35L65, 65D25, 65D30, 65F15, 65L05, 65L06, 65L60, 65M22

DOI. 10.1137/19M1242963

1. Introduction.

Time-dependent nonlinear problems arise in many important disciplines such as engineering, science, and technologies. They are numerically solved if it is not possible to solve them analytically. Depending on the complexity and size of the governing equations and problem domains, the problems can be computationally expensive to solve. It may take a long time to run one forward simulation even with high performance computing. For example, a simulation of the powder bed fusion additive manufacturing procedure shown in [26] takes a week to finish with 108 cores. Other computationally expensive simulations include the three-dimensional (3D) shocked spherical helium bubble simulation that appeared in [4] and the inertial confinement fusion implosion dynamics simulations that appeared in [1].

*Submitted to the journal's Methods and Algorithms for Scientific Computing section February 5, 2019; accepted for publication (in revised form) December 30, 2019; published electronically April 13, 2020.
https://doi.org/10.1137/19M1242963

Funding: This work was performed at Lawrence Livermore National Laboratory and was supported by the LDRD program (project 43320) and the Engineering Summer Intern program (project 45415). Lawrence Livermore National Laboratory is operated by Lawrence Livermore National Security, LLC, for the U.S. Department of Energy, National Nuclear Security Administration under contract DE-AC52-07NA27344 and LLNL-JRNL-767241.

†Lawrence Livermore National Laboratory, Livermore, CA 94550 (choi15@llnl.gov, anderson110@llnl.gov).
‡Mechanical and Aerospace Department, Syracuse University, Syracuse, NY 13244 (dmcoombs@syr.edu).

The computationally expensive simulations are not desirable in the context of parameter study, design optimization, uncertainty quantification, and inverse problems where several forward simulations are needed. A reduced order model (ROM) can be useful in this context to accelerate the computationally expensive forward simulations with good enough approximate solutions.

We consider projection-based ROMs for nonlinear dynamical systems. Such ROMs include the empirical interpolation method (EIM) [6, 21], the discrete empirical interpolation method (DEIM) [12, 15] and the Gauss–Newton with approximated tensors (GNAT) [10, 11], the best point interpolation method (BPIM) [30], the missing point estimation (MPE) [5], and cubature methods [3, 19, 20, 23]. A hyper-reduction (the term first used in the context of ROMs by Ryckelynck in [32]) is necessary to efficiently reduce the complexity of nonlinear terms for a considerable speed-up compared to a corresponding full order model (FOM). The DEIM and GNAT approaches take discrete nonlinear term snapshots to build a nonlinear term basis. Then, they select a subset of each nonlinear term basis vector to either interpolate or data-fit in a least-squares sense. In this way, they reduce the computational complexity of updating nonlinear terms in an iterative solver for nonlinear problems. The EIM, BPIM, and MPE approaches take a similar hyper-reduction to that in DEIM and GNAT except for the fact that they build nonlinear term basis functions in a continuous space. Whether they work on a discrete or a continuous space, they follow the framework of reconstructing "gappy" data, first introduced in the context of ROMs by [18]. The cubature methods, in contrast, take a different approach. They approximate nonlinear integral as a finite sum of positive scalar weights multiplied by the integrand evaluated at sampled elements. The cubature methods developed in [3, 19, 20] do not require building nonlinear term basis. They solve the nonnegative least-squares (NNLS) problem directly with the nonlinear term snapshots. Recently, Hernandez, Caicedo, and Ferrer in [23] developed the empirical cubature method (ECM) approach that builds nonlinear term basis to solve a smaller NNLS problem. The requirement of building nonlinear term basis in hyper-reduction results in computational cost and storage in addition to solution basis construction. The cost is significant if the corresponding FOMs are large-scale. The large-scale problem requires large additional storage for nonlinear term snapshots and large-scale compression techniques to build a basis. In particular, the cost of the hyper-reduction in the recently developed space-time ROM (i.e., ST-GNAT) [13] is even more significant than the aforementioned spatial ROMs (e.g., EIM, DEIM, GNAT, BPIM, MPE, and ECM). The best nonlinear term snapshots of the ST-GNAT method are obtained from the corresponding space-time ROMs without hyper-reduction (i.e., ST-LSPG) [13], which is not practical for a large-scale problem due to the cumbersome size.[1] Therefore, a novel and efficient way of constructing nonlinear term basis needs to be developed.

This paper shows a practical way of avoiding nonlinear term snapshots for the construction of nonlinear term basis. The idea comes from a simple fact that the nonlinear terms are related with solution snapshots through underlying time integrators. In fact, many time integrators approximate time derivative terms as a linear combination of the solution snapshots. It implies that the nonlinear term snapshots belong to

[1]The full size, implicitly handled by the ST-LSPG approach due to the nonlinear term and corresponding Jacobian updates, is the FOM spatial degrees of freedom multiplied by the number of time steps.

the subspace spanned by the solution snapshots. Furthermore, a subspace needed for nonlinear terms in a hyper-reduction is determined by the range space of the solution basis matrix possibly multiplied by a nonsingular matrix (e.g., the volume matrix). Therefore, the solution snapshots can be used to construct nonlinear term basis. This leads to our proposed method, the solution-based nonlinear subspace (SNS) method, that provides two savings for constructing nonlinear term basis because of

1. no additional collection of nonlinear term snapshots (i.e., storage saving),
2. no additional compression of snapshots (e.g., no additional singular value decomposition (SVD) of nonlinear term snapshots, implying computational cost saving).

The first saving is especially big for GNAT and ST-GNAT because they involve an expensive collection procedure for their best performace.

1.1. Organization of the paper. We start our discussion by describing the time-continuous representation of the FOM in section 2. Section 2 also describes the time-discrete representation of the FOM with one-step Euler time integrators. The subspace inclusion relation between the subspaces spanned by the solution and nonlinear term snapshots is described for the Euler time integrators. Several projection-based ROMs (i.e., the DEIM, GNAT, and ST-GNAT models) are considered in section 3 for the SNS method to be applied. Readers familiar with DEIM, GNAT, and ST-GNAT can skip sections 2 and 3 and start with section 4. The SNS method is described in section 4 and applied to those ROMs. Section 4 also introduces the conforming subspace condition to justify the SNS method. Section 5 shows an error analysis for the SNS method regarding the oblique projection error bound. It analyzes the effect of the nonsingular matrix that is used to form a nonlinear term basis. Section 6 reports numerical results that support benefits of the SNS method and section 7 concludes the paper with a summary and future work. Although the SNS method is mainly illustrated with the Euler time integrators throughout the paper, it is applicable to other time integrators. Appendix A considers several other time integrators and the subspace inclusion relation for each time integrator. The following time integrators are included: the Adams–Bashforth, Adams–Moulton, backward differentiation formula (BDF), and midpoint Runge–Kutta methods.

2. Full order model.

A parameterized nonlinear dynamical system is considered, characterized by a system of nonlinear ordinary differential equations (ODEs), which can be considered as a resultant system from semidiscretization of partial differential equations (PDEs) in space domains

$$\boldsymbol{M}\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, t; \boldsymbol{\mu}), \qquad \boldsymbol{x}(0;\boldsymbol{\mu}) = \boldsymbol{x}^0(\boldsymbol{\mu}), \tag{2.1}$$

where $\boldsymbol{M} \in \mathbb{R}^{N_s\times N_s}$ denotes a nonsingular matrix, $t \in [0, T]$ denotes time with the final time $T \in \mathbb{R}_+$, and $\boldsymbol{x}(t;\boldsymbol{\mu})$ denotes the time-dependent, parameterized state implicitly defined as the solution to problem (2.1) with $\boldsymbol{x} : [0, T]\times\mathcal{D} \to \mathbb{R}^{N_s}$. Further, $\boldsymbol{f} : \mathbb{R}^{N_s} \times [0, T]\times\mathcal{D} \to \mathbb{R}^{N_s}$ with $(\boldsymbol{w}, \tau; \boldsymbol{\nu}) \mapsto \boldsymbol{f}(\boldsymbol{w}, \tau; \boldsymbol{\nu})$ denotes the scaled velocity of $\boldsymbol{Mx}$, which we assume to be nonlinear in at least its first argument. The initial state is denoted by $\boldsymbol{x}^0 : \mathcal{D} \to \mathbb{R}^{N_s}$, and $\boldsymbol{\mu} \in \mathcal{D}$ denotes the parameters with parameter domain $\mathcal{D} \subseteq \mathbb{R}^{n_\mu}$.

A uniform time discretization is assumed throughout the paper, characterized by time step $\Delta t \in \mathbb{R}_+$ and time instances $t^n = t^{n-1} + \Delta t$ for $n \in \mathbb{N}(N_t)$ with $t^0 = 0$, $N_t \in \mathbb{N}$, and $\mathbb{N}(N) := \{1, \ldots, N\}$. To avoid notational clutter, we introduce the following time discretization–related notation: $\boldsymbol{x}_n := \boldsymbol{x}(t^n;\boldsymbol{\mu})$, $\tilde{\boldsymbol{x}}_n := \tilde{\boldsymbol{x}}(t^n;\boldsymbol{\mu})$, $\hat{\boldsymbol{x}}_n := \hat{\boldsymbol{x}}(t^n;\boldsymbol{\mu})$, and $\boldsymbol{f}_n := \boldsymbol{f}(\boldsymbol{x}(t^n;\boldsymbol{\mu}), t^n;\boldsymbol{\mu})$.

Two different types of time discretization methods are considered: explicit and implicit time integrators. For illustration purposes, we mainly consider the forward Euler time integrator for an explicit scheme and the backward Euler time integrator for an implicit scheme. Several other time integrators are shown in Appendix A.

The explicit forward Euler method numerically solves (2.1), by time marching with the following update:

$$\boldsymbol{M}\boldsymbol{x}_n - \boldsymbol{M}\boldsymbol{x}_{n-1} = \Delta t \boldsymbol{f}_{n-1}. \tag{2.2}$$

Equation (2.2) implies the following subspace inclusion:

$$\mathrm{span}\{\boldsymbol{f}_{n-1}\} \subseteq \mathrm{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_n\}.$$

By induction, we conclude the subspace inclusion relation

$$\mathrm{span}\{\boldsymbol{f}_0, \dots, \boldsymbol{f}_{N_t-1}\} \subseteq \mathrm{span}\{\boldsymbol{M}\boldsymbol{x}_0, \dots, \boldsymbol{M}\boldsymbol{x}_{N_t}\}, \tag{2.3}$$

which shows that the span of nonlinear term snapshots is included in the span of $\boldsymbol{M}$-scaled solution snapshots. The residual function with the forward Euler time integrator is defined as

$$\boldsymbol{r}_{\mathrm{FE}}^n(\boldsymbol{x}_n;\boldsymbol{x}_{n-1},\boldsymbol{\mu}) := \boldsymbol{M}(\boldsymbol{x}_n - \boldsymbol{x}_{n-1}) - \Delta t \boldsymbol{f}_{n-1}.$$

The implicit backward Euler (BE) method numerically solves (2.1), by solving the following nonlinear system of equations for $\boldsymbol{x}_n$ at the nth time step:

$$\boldsymbol{M}\boldsymbol{x}_n - \boldsymbol{M}\boldsymbol{x}_{n-1} = \Delta t \boldsymbol{f}_n. \tag{2.4}$$

Equation (2.4) implies the following subspace inclusion:

$$\mathrm{span}\{\boldsymbol{f}_n\} \subseteq \mathrm{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_n\}.$$

By induction, we conclude the following subspace inclusion relation:

$$\mathrm{span}\{\boldsymbol{f}_1, \dots, \boldsymbol{f}_{N_t}\} \subseteq \mathrm{span}\{\boldsymbol{M}\boldsymbol{x}_0, \dots, \boldsymbol{M}\boldsymbol{x}_{N_t}\}, \tag{2.5}$$

which shows that the span of nonlinear term snapshots is included in the span of $\boldsymbol{M}$-scaled solution snapshots. The residual function with the backward Euler time integrator is defined as

$$\boldsymbol{r}_{\mathrm{BE}}^n(\boldsymbol{x}_n;\boldsymbol{x}_{n-1},\boldsymbol{\mu}) := \boldsymbol{M}(\boldsymbol{x}_n - \boldsymbol{x}_{n-1}) - \Delta t \boldsymbol{f}_n. \tag{2.6}$$

3. Projection-based reduced order models.

Projection-based ROMs are considered for nonlinear dynamical systems. Especially, the ones that require building a nonlinear term basis are of interest: the DEIM, GNAT, and ST-GNAT approaches.

3.1. DEIM.

The DEIM approach applies spatial projection using a subspace $\mathcal{S} := \mathrm{span}\{\boldsymbol{\phi}_i\}_{i=1}^{n_s} \subseteq \mathbb{R}^{N_s}$ with $\dim(\mathcal{S}) = n_s \ll N_s$. Using this subspace, it approximates the solution as $\boldsymbol{x} \approx \tilde{\boldsymbol{x}} \in \boldsymbol{x}_0 + \mathcal{S}$ (i.e., in a trial subspace) or equivalently

$$\tilde{\boldsymbol{x}} = \boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}},$$

where $\boldsymbol{\Phi} := [\boldsymbol{\phi}_1 \cdots \boldsymbol{\phi}_{n_s}] \in \mathbb{R}^{N_s \times n_s}$ denotes a basis matrix and $\hat{\boldsymbol{x}} \in \mathbb{R}^{n_s}$ with $\hat{\boldsymbol{x}}_0 = \boldsymbol{0}$ denotes the generalized coordinates. Replacing $\boldsymbol{x}$ with $\tilde{\boldsymbol{x}}$ in (2.1) leads to the following system of equations with reduced number of unknowns:

$$\boldsymbol{M}\boldsymbol{\Phi}\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{f}(\boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}, t; \boldsymbol{\mu}). \tag{3.1}$$

For constructing $\boldsymbol{\Phi}$, proper orthogonal decomposition (POD) is commonly used. POD [7] obtains $\boldsymbol{\Phi}$ from a truncated SVD approximation to a FOM solution snapshot matrix. It is related to principal component analysis in statistical analysis [25] and Karhunen–Loève expansion [28] in stochastic analysis. POD forms a solution snapshot matrix, $\boldsymbol{X} := [\boldsymbol{x}_0^{\boldsymbol{\mu}_1} \quad \cdots \quad \boldsymbol{x}_{N_t}^{\boldsymbol{\mu}_{n_\mu}}] \in \mathbb{R}^{N_s \times n_\mu(N_t+1)}$, where $\boldsymbol{x}_n^{\boldsymbol{\mu}_k}$ is a solution state at nth time step with parameter $\boldsymbol{\mu}_k$ for $n \in \mathbb{N}(N_t)$ and $k \in \mathbb{N}(n_\mu)$. Then, POD computes its thin SVD:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T,$$

where $\boldsymbol{U} \in \mathbb{R}^{N_s \times N_t}$ and $\boldsymbol{V} \in \mathbb{R}^{N_t \times N_t}$ are orthogonal matrices and $\boldsymbol{\Sigma} \in \mathbb{R}^{N_t \times N_t}$ is a diagonal matrix with singular values on its diagonals. Then POD chooses the leading n_s columns of $\boldsymbol{U}$ to set $\boldsymbol{\Phi}$ (i.e., $\boldsymbol{\Phi} = \boldsymbol{U}(:, 1 : n_s)$ in MATLAB notation). The POD basis minimizes $\|\boldsymbol{X} - \boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{X}\|_F^2$ over all $\boldsymbol{\Phi} \in \mathbb{R}^{N_s \times n_s}$ with orthonormal columns, where $\|\boldsymbol{A}\|_F$ denotes the Frobenius norm of a matrix $\boldsymbol{A} \in \mathbb{R}^{I \times J}$, defined as $\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J} a_{ij}}$ with a_{ij} being an (i,j)th element of $\boldsymbol{A}$. Since the objective function does not change if $\boldsymbol{\Phi}$ is postmultiplied by an arbitrary $n_s \times n_s$ orthogonal matrix, the POD procedure seeks the optimal n_s-dimensional subspace that captures the snapshots in the least-squares sense. For more details on POD, we refer to [24, 27].

Note that (3.1) has more equations than unknowns (i.e., an overdetermined system). It is likely that there is no solution satisfying (3.1). In order to close the system, the Galerkin projection solves the following reduced system with $\hat{\boldsymbol{x}}_0 = \boldsymbol{0}$:

$$\boldsymbol{\Phi}^T\boldsymbol{M}\boldsymbol{\Phi}\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{\Phi}^T\boldsymbol{f}(\boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}, t; \boldsymbol{\mu}). \tag{3.2}$$

Applying a time integrator to (3.2) leads to a fully discretized reduced system, denoted as the reduced OΔE. Note that the reduced OΔE has n_s unknowns and n_s equations. If an implicit time integrator is applied, a Newton–type method can be applied to solve for unknown generalized coordinates each time step. If an explicit time integrator is applied, time marching updates will solve the system. However, we cannot expect any speed-up because the size of the nonlinear term $\boldsymbol{f}$ and its Jacobian, which need to be updated for every Newton step, scales with the FOM size. Thus, to overcome this issue, the DEIM approach applies a hyper-reduction technique. That is, it projects $\boldsymbol{f}$ onto a subspace $\mathscr{F} := \operatorname{span}\{\boldsymbol{\phi}_{f,i}\}_{i=1}^{n_f}$ and approximates $\boldsymbol{f}$ as

$$\boldsymbol{f} \approx \boldsymbol{\Phi}_f\hat{\boldsymbol{f}}, \tag{3.3}$$

where $\boldsymbol{\Phi}_f := [\boldsymbol{\phi}_{f,1}, \ldots, \boldsymbol{\phi}_{f,n_f}] \in \mathbb{R}^{N_s \times n_f}$, $n_f \ll N_s$, denotes the nonlinear term basis matrix and $\hat{\boldsymbol{f}} \in \mathbb{R}^{n_f}$ denotes the generalized coordinates of the nonlinear term. The generalized coordinates, $\hat{\boldsymbol{f}}$, can be determined by the following interpolation:

$$\boldsymbol{Z}^T\boldsymbol{f} = \boldsymbol{Z}^T\boldsymbol{\Phi}_f\hat{\boldsymbol{f}},$$

where $\boldsymbol{Z}^T := [\boldsymbol{e}_{p_1}, \ldots, \boldsymbol{e}_{p_{n_f}}]^T \in \mathbb{R}^{n_f \times N_s}$ is the sampling matrix and $\boldsymbol{e}_{p_i}$ is the p_ith column of the identity matrix $\boldsymbol{I}_{N_s} \in \mathbb{R}^{N_s \times N_s}$. Therefore, (3.3) becomes

$$\boldsymbol{f} \approx \mathscr{P}_{\mathrm{DEIM}} \boldsymbol{f}, \tag{3.4}$$

where $\mathscr{P}_{\mathrm{DEIM}} := \boldsymbol{\Phi}_f (\boldsymbol{Z}^T \boldsymbol{\Phi}_f)^{-1} \boldsymbol{Z}^T \in \mathbb{R}^{n_f \times n_f}$ is the DEIM oblique projection matrix. The DEIM approach does not construct the sampling matrix $\boldsymbol{Z}$. Instead, it maintains the sampling indices $\{p_1, \ldots, p_{n_f}\}$ and corresponding rows of $\boldsymbol{\Phi}_f$ and $\boldsymbol{f}$. This enables DEIM to achieve a speed-up when it is applied to nonlinear problems.

The original DEIM paper [12] constructs the nonlinear term basis $\{\boldsymbol{\phi}_{f,1}, \ldots, \boldsymbol{\phi}_{f,n_f}\}$ by applying another POD on the nonlinear term snapshots[2] obtained from the FOM simulation at every time step. This implies that DEIM requires two separate SVDs and storage for two different snapshots (i.e., solution state and nonlinear term snapshots). Section 4 discusses how to avoid the collection of nonlinear term snapshots and an extra SVD without losing the quality of the hyper-reduction.

The sampling indices (i.e., $\boldsymbol{Z}$) can be found by either a row pivoted LU decomposition [12] or the strong column pivoted rank-revealing QR (sRRQR) decomposition [15, 16]. Depending on the algorithm of selecting the sampling indices, the DEIM projection error bound is determined. For example, the row pivoted LU decomposition in [12] results in the error bound

$$\|\boldsymbol{f} - \mathscr{P}\boldsymbol{f}\|_2 \leq \kappa \|(\boldsymbol{I}_{N_s} - \boldsymbol{\Phi}_f \boldsymbol{\Phi}_f^T)\boldsymbol{f}\|_2,$$

where $\|\cdot\|_2$ denotes ℓ_2 norm of a vector and κ is the condition number of $(\boldsymbol{Z}^T\boldsymbol{\Phi}_f)^{-1}$ and it is bounded by

$$\kappa \leq (1 + \sqrt{2N_s})^{n_f - 1} \|\boldsymbol{\phi}_{f,1}\|_\infty^{-1}. \tag{3.5}$$

On the other hand, the sRRQR factorization in [16] reveals a tighter bound than (3.5):

$$\kappa \leq \sqrt{1 + \eta^2 n_f (N_s - n_f)}, \tag{3.6}$$

where η is a tuning parameter in the sRRQR factorization (i.e., f in Algorithm 4 of [22]).

3.2. GNAT. In contrast to DEIM, the GNAT method takes the Least-Squares Petrov–Galerkin (LSPG) approach. The LSPG method projects a fully discretized solution space onto a trial subspace. That is, it discretizes (2.1) in time domain and replaces $\boldsymbol{x}_n$ with $\tilde{\boldsymbol{x}}_n := \boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}_n$ for $n \in \mathbb{N}(N_t)$ in residual functions defined in section 2 and Appendix A. Here, we consider only implicit time integrators because the LSPG projection is equivalent to the Galerkin projection when an explicit time integrator is used as shown in section 5.1 in [9]. The residual functions for implicit time integrators are defined in (2.6), (A.4), and (A.6) for various time integrators. For example, the residual function with the backward Euler time integrator[3] after the trial subspace projection becomes

[2]The nonlinear term snapshots are $\{\boldsymbol{f}_1, \ldots, \boldsymbol{f}_{N_t}\}$ with the backward Euler time integrator and $\{\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{N_t-1}\}$ with the forward Euler time integrator

[3]Although the backward Euler time integrator is used extensively in the paper as an illustration, many other time integrators introduced in Appendix A can be applied to all the ROM methods addressed in the paper in a straightforward way.

$$\begin{aligned}\tilde{\mathbf{r}}^n_{\mathrm{BE}}(\hat{\boldsymbol{x}}_n;\hat{\boldsymbol{x}}_{n-1},\boldsymbol{\mu}) &:= \boldsymbol{r}^n_{\mathrm{BE}}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n;\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_{n-1},\boldsymbol{\mu})\\ &= \boldsymbol{M}\boldsymbol{\Phi}(\hat{\boldsymbol{x}}_n-\hat{\boldsymbol{x}}_{n-1})-\Delta t\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu}).\end{aligned} \tag{3.7}$$

The basis matrix $\boldsymbol{\Phi}$ can be found by the POD as in the DEIM approach. Note that (3.7) is an overdetermined system. To close the system and solve for the unknown generalized coordinates, $\hat{\boldsymbol{x}}_n$, the LSPG ROM takes the squared norm of the residual vector function and minimizes it at every time step:

$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\boldsymbol{r}^n_{\mathrm{BE}}(\hat{\boldsymbol{v}};\hat{\boldsymbol{x}}_{n-1},\boldsymbol{\mu})\right\|_2^2. \tag{3.8}$$

The Gauss–Newton method with the starting point $\hat{\boldsymbol{x}}_{n-1}$ is applied to solve the minimization problem (3.8) in GNAT. However, as in the DEIM approach, a hyper-reduction is required for a speed-up due to the presence of the nonlinear residual vector function. The GNAT method approximates the nonlinear residual term with gappy POD [18], whose procedure is similar to DEIM, as

$$\boldsymbol{r} \approx \boldsymbol{\Phi}_r\hat{\boldsymbol{r}}, \tag{3.9}$$

where $\boldsymbol{\Phi}_r := [\boldsymbol{\phi}_{r,1},\dots,\boldsymbol{\phi}_{r,n_r}] \in \mathbb{R}^{N_s\times n_r}$, $n_s \le n_r \ll N_s$, denotes the residual basis matrix and $\hat{\boldsymbol{r}} \in \mathbb{R}^{n_r}$ denotes the generalized coordinates of the nonlinear residual term. In contrast to DEIM, the GNAT method solves the following least-squares problem to obtain the generalized coordinates $\hat{\boldsymbol{r}}_n$ at nth time step:

$$\hat{\boldsymbol{r}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_r}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\boldsymbol{Z}^T(\boldsymbol{r}-\boldsymbol{\Phi}_r\hat{\boldsymbol{v}})\right\|_2^2. \tag{3.10}$$

where $\boldsymbol{Z}^T := [\boldsymbol{e}_{p_1},\dots,\boldsymbol{e}_{p_{n_z}}]^T \in \mathbb{R}^{n_z\times N_s}$, $n_s \le n_r \le n_z \ll N_s$, is the sampling matrix and $\boldsymbol{e}_{p_i}$ is the p_ith column of the identity matrix $\boldsymbol{I}_{N_s} \in \mathbb{R}^{N_s\times N_s}$. The solution to (3.10) is given as

$$\hat{\boldsymbol{r}}_n = (\boldsymbol{Z}^T\boldsymbol{\Phi}_r)^\dagger\boldsymbol{Z}^T\boldsymbol{r},$$

where the Moore–Penrose inverse of a matrix $\boldsymbol{A} \in \mathbb{R}^{n_z\times n_r}$ with full column rank is defined as $\boldsymbol{A}^\dagger := (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$. Therefore, (3.9) becomes

$$\boldsymbol{r} \approx \mathscr{P}_{\mathrm{GNAT}}\boldsymbol{r},$$

where $\mathscr{P}_{\mathrm{GNAT}} := \boldsymbol{\Phi}_r(\boldsymbol{Z}^T\boldsymbol{\Phi}_r)^\dagger\boldsymbol{Z}^T$ is the GNAT oblique projection matrix. Note that it has a similar structure to $\mathscr{P}_{\mathrm{DEIM}}$. The GNAT projection matrix has a pseudo-inverse instead of the inverse because it allows $n_r < n_z$. The GNAT method does not construct the sampling matrix $\boldsymbol{Z}$. Instead, it maintains the sampling indices $\{p_1,\dots,p_{n_f}\}$ and corresponding rows of $\boldsymbol{\Phi}_r$ and $\boldsymbol{r}$. This enables GNAT to achieve a speed-up when it is applied to nonlinear problems.

The sampling indices (i.e., $\boldsymbol{Z}$) can be determined by Algorithm 3 of [11] for computational fluid dynamics problems and Algorithm 5 of [10] for other problems. These two algorithms take the greedy procedure to minimize the error in the gappy reconstruction of the POD basis vectors $\boldsymbol{\Phi}_r$. The major difference between these sampling algorithms and the ones for the DEIM method is that these algorithms for the GNAT method allow oversampling (i.e., $n_z > n_r$), resulting in solving least-squares problems in the greedy procedure. These selection algorithms can be viewed as the extension of Algorithm 1 in [12] (i.e., a row pivoted LU decomposition) to the oversampling case. The nonlinaer residual term projection error associated with these sampling algorithms is presented in Appendix D of [11]. That is,

$$\|\boldsymbol{r} - \mathscr{P}_{\mathrm{GNAT}}\boldsymbol{r}\|_2 \leq \|\boldsymbol{R}^{-1}\|_2 \|\boldsymbol{r} - \boldsymbol{\Phi}_r\boldsymbol{\Phi}_r^T\boldsymbol{r}\|_2,$$

where $\boldsymbol{R}$ is the triangular factor from the QR factorization of $\boldsymbol{Z}^T\boldsymbol{\Phi}_r$ (i.e., $\boldsymbol{Z}^T\boldsymbol{\Phi}_r = \boldsymbol{QR}$).

We present another sampling selection algorithm that was not considered in any GNAT papers (e.g., [10, 11]). It uses the sRRQR factorization developed originally in [22] and further utilized in the W-DEIM method of [16] for the case of $n_z \geq n_r$. That is, applying Algorithm 4 of [22] to $\boldsymbol{\Phi}_r^T$ produces an index selection operator $\boldsymbol{Z}$ whose projection error satisfies

$$\|\boldsymbol{r} - \mathscr{P}_{\mathrm{GNAT}}\boldsymbol{r}\|_2 \leq \sqrt{1+\eta^2 n_r(N_s - n_r)}\|\boldsymbol{r} - \boldsymbol{\Phi}_r\boldsymbol{\Phi}_r^T\boldsymbol{r}\|_2. \tag{3.11}$$

This error bound can be obtained by setting the identity matrix as a weight matrix in Theorem 4.8 of [16].

Finally, the GNAT method minimizes the following least-squares problem at every time step, for example, with the backward Euler time integrator:

$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| (\boldsymbol{Z}^T\boldsymbol{\Phi}_r)^\dagger \boldsymbol{Z}^T \boldsymbol{r}_{\mathrm{BE}}^n(\hat{\boldsymbol{v}}; \hat{\boldsymbol{x}}_{n-1}, \boldsymbol{\mu}) \right\|_2^2.$$

The GNAT method applies another POD to a nonlinear residual term snapshot to construct $\boldsymbol{\Phi}_r$. The original GNAT paper [11] collects residual snapshots at every Newton iteration from the LSPG simulations[4] for several reasons:

1. The GNAT method takes LSPG as a reference model (i.e., denoted as Model II in [11]). Its ultimate goal is to achieve the accuracy of Model II.
2. The residual snapshots taken every time step (i.e., at the end of Newton iterations at every time step) of the FOM are small in magnitude.
3. The residual snapshots taken from every Newton step gives information about the path that the Newton iterations in Model II take. GNAT tries to mimic the Newton path that LSPG takes.

Some disadvantages of the original GNAT approach include the following:

1. The GNAT method requires more storage than DEIM to store residual snapshots from every Newton iteration (cf. i.e., the DEIM approach stores only one nonlinear term snapshot per each time step).
2. The GNAT method requires more expensive SVD for nonlinear residual basis construction than the DEIM approach because the number of nonlinear residual snapshots in the GNAT method is larger than the number of nonlinear term snapshots in DEIM.
3. The GNAT method requires the simulations of Model II which are computationally expensive. For a parametric global ROM, it is computationally expensive because it requires as many Model II simulations as there are training points in a given parameter domain.

Section 4 discusses how to avoid the collection of nonlinear term snapshots and the extra SVD without losing the quality of the hyper-reduction.

[4]We denote the LSPG simulation to be the ROM simulation without hyper-reduction. That is, LSPG solves (3.8) without any hyper-reduction if the backward Euler time integrator is used.

3.3. Space-time GNAT. The ST-GNAT method takes the space-time LSPG (ST-LSPG) approach. Given a time integrator, one can rewrite a fully discretized system of equations to (2.1) in a space-time form. For example, if the backward Euler time integrator[5] is used for time domain discretization, then the corresponding space-time formulation to (2.4) becomes

$$\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}} = \Delta t\bar{\boldsymbol{f}} + \bar{\boldsymbol{q}}^0, \tag{3.12}$$

where

$$\boldsymbol{A}_{\mathrm{BE}} = \begin{bmatrix} \boldsymbol{M} & & & \\ -\boldsymbol{M} & \boldsymbol{M} & & \\ & \ddots & \ddots & \\ & & -\boldsymbol{M} & \boldsymbol{M} \end{bmatrix}, \quad \bar{\boldsymbol{x}} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{N_t} \end{pmatrix}, \quad \bar{\boldsymbol{f}} = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \vdots \\ \boldsymbol{f}_{N_t} \end{pmatrix}, \quad \bar{\boldsymbol{q}}^0 = \begin{pmatrix} \boldsymbol{M}\boldsymbol{x}^0 \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{pmatrix}. \tag{3.13}$$

Note that $\bar{\boldsymbol{x}}(\boldsymbol{\mu})$ denotes the parameterized space-time state implicitly defined as the solution to the problem (3.12) with $\bar{\boldsymbol{x}} : \mathcal{D} \to \mathbb{R}^{N_s N_t}$ and $\bar{\boldsymbol{x}}(\boldsymbol{\mu}) \in \mathbb{R}^{N_s N_t}$. Further, $\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}; \boldsymbol{\mu}) \in \mathbb{R}^{N_s N_t}$ denotes the space-time nonlinear term that is nonlinear in at least its first argument with $\bar{\boldsymbol{f}} : \mathbb{R}^{N_s N_t} \times \mathcal{D} \to \mathbb{R}^{N_s N_t}$. The space-time residual function $\bar{\boldsymbol{r}} \in \mathbb{R}^{N_s N_t}$ corresponding to (3.12) is defined as

$$\begin{aligned} \bar{\boldsymbol{r}}_{\mathrm{BE}}(\bar{\boldsymbol{x}}; \boldsymbol{\mu}) &:= \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}(\boldsymbol{\mu}) - \Delta t\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}; \boldsymbol{\mu}) - \bar{\boldsymbol{q}}^0(\boldsymbol{\mu}) \\ &= \boldsymbol{0}. \end{aligned} \tag{3.14}$$

To reduce both the spatial and temporal dimensions of the FOM, the ST-LSPG method enforces the approximated numerical solution $\tilde{\boldsymbol{y}} \in \mathbb{R}^{N_s N_t}$ to reside in an affine "space-time trial subspace"

$$\tilde{\boldsymbol{y}} \in \mathcal{ST} \subseteq \mathbb{R}^{N_s N_t},$$

where $\mathcal{ST} := \boldsymbol{1}_{N_t} \otimes \boldsymbol{x}^0(\boldsymbol{\mu}) + \mathrm{span}\{\bar{\boldsymbol{\phi}}_i\}_{i=1}^{n_{st}} \subseteq \mathbb{R}^{N_s N_t}$ with $\dim(\mathcal{ST}) = n_{st} \ll N_s N_t$ and $\boldsymbol{1}_{N_t} \in \mathbb{R}^{N_t}$ whose elements are all one. Here, $\otimes$ denotes the Kronecker product of two matrices and the product of two matrices $\boldsymbol{A} \in \mathbb{R}^{I \times J}$ and $\boldsymbol{B} \in \mathbb{R}^{K \times L}$ is defined as

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1J}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{I1}\boldsymbol{B} & \cdots & a_{IJ}\boldsymbol{B} \end{bmatrix}.$$

Further, $\bar{\boldsymbol{\phi}}_i \in \mathbb{R}^{N_s N_t}$ denotes a space-time basis vector. Thus, the ST-LSPG method approximates the numerical solution as

$$\bar{\boldsymbol{x}}(\boldsymbol{\mu}) \approx \tilde{\boldsymbol{y}}(\boldsymbol{\mu}) = \boldsymbol{1}_{N_t} \otimes \boldsymbol{x}^0(\boldsymbol{\mu}) + \sum_{i=1}^{n_{st}} \bar{\boldsymbol{\phi}}_i \hat{y}_i(\boldsymbol{\mu}), \tag{3.15}$$

where $\hat{y}_i(\boldsymbol{\mu}) \in \mathbb{R}$, $i \in \mathbb{N}(n_{st})$, denotes the generalized coordinate of the ST-LSPG solution.

A space-time residual vector function can now be defined with the generalized coordinates as an argument. Replacing $\bar{\boldsymbol{x}}$ in (3.14) with $\tilde{\boldsymbol{y}}$ gives the residual vector function

$$\bar{\mathbf{r}}_{\mathrm{BE}}(\hat{\boldsymbol{y}}; \boldsymbol{\mu}) := \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}}; \boldsymbol{\mu}) - \bar{\boldsymbol{q}}^0(\boldsymbol{\mu}) + \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0(\boldsymbol{\mu}), \tag{3.16}$$

[5]Here we only consider the backward Euler time integrator for the simplicity of illustration. However, the extension to other linear multistep implicit time integrators is straightforward.

where $\bar{\boldsymbol{x}}^0 := \boldsymbol{1}_{N_t} \otimes \boldsymbol{x}^0$ and $\bar{\boldsymbol{\Phi}} \in \mathbb{R}^{N_s N_t \times n_{st}}$ denotes a space-time basis matrix that is defined as $\bar{\boldsymbol{\Phi}} := \begin{bmatrix} \bar{\boldsymbol{\phi}}_1 & \cdots & \bar{\boldsymbol{\phi}}_{n_{st}} \end{bmatrix}$ and $\hat{\boldsymbol{y}} \in \mathbb{R}^{n_{st}}$ denotes the generalized coordinate vector that is defined as

$$\hat{\boldsymbol{y}} := \begin{bmatrix} \hat{y}_1(\boldsymbol{\mu}) & \cdots & \hat{y}_{n_{st}}(\boldsymbol{\mu}) \end{bmatrix}^T.$$

Note that $\bar{\boldsymbol{q}}^0(\boldsymbol{\mu}) + \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0(\boldsymbol{\mu})$ vanishes. Equation (3.16) becomes

$$\bar{\mathbf{r}}_{\mathrm{BE}}(\hat{\boldsymbol{y}};\boldsymbol{\mu}) := \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu}). \tag{3.17}$$

Reduced space-time residual vector functions for other time integrators can be defined similarly. We denote $\bar{\mathbf{r}}$ as a reduced space-time residual vector function with a generic time integrator.

The space-time basis matrix $\bar{\boldsymbol{\Phi}}$ can be obtained by tensor product of spatial and temporal basis vectors. The spatial basis vectors can be obtained via POD as in the DEIM and GNAT approaches. The temporal basis vectors can be obtained via the following three tensor decompositions described in [13]:

- fixed temporal subspace via T-HOSVD,
- fixed temporal subspace via ST-HOSVD,
- tailored temporal subspace via ST-HOSVD.

The ST-HOSVD method is a more efficient version of T-HOSVD. Thus, we will not consider T-HOSVD. The tailored temporal subspace via ST-HOSVD has an LL1 form that has appeared, for example, in [33, 14]. Therefore, we will refer to it as the LL1 decomposition.

Because of the reduction in spatial and temporal dimension, the space-time residual vector $\bar{\mathbf{r}}$ cannot achieve zero most likely. Thus, the ST-LSPG method minimizes the square norm of $\bar{\mathbf{r}}$ and computes the ST-LSPG solution:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}} \in \mathbb{R}^{n_{st}}}{\arg\min} \|\bar{\mathbf{r}}(\hat{\boldsymbol{v}};\boldsymbol{\mu})\|_2^2. \tag{3.18}$$

The ST-LSPG ROM solves (3.18) without any hyper-reduction. As in the DEIM and GNAT approaches, a hyper-reduction is required for a considerable speed-up due to the presence of the space-time nonlinear residual vector. Therefore, the ST-GNAT method approximates the space-time nonlinear residual terms with gappy POD [18], which in turn requires construction of a space-time residual basis. Similar to the GNAT method, the ST-GNAT method approximates the space-time nonlinear residual term as

$$\bar{\mathbf{r}} \approx \bar{\boldsymbol{\Phi}}_r \hat{\boldsymbol{r}}, \tag{3.19}$$

where $\bar{\boldsymbol{\Phi}}_r := [\bar{\boldsymbol{\phi}}_{r,1}, \ldots, \bar{\boldsymbol{\phi}}_{r,n_r}] \in \mathbb{R}^{N_s N_t \times n_r}$, $n_{st} \le n_r \ll N_s N_t$, denotes the space-time residual basis matrix and $\hat{\boldsymbol{r}} \in \mathbb{R}^{n_r}$ denotes the generalized coordinates of the nonlinear residual term. The ST-GNAT solves the following space-time least-squares problem to obtain the generalized coordinates, $\hat{\boldsymbol{r}}$:

$$\hat{\boldsymbol{r}} = \underset{\hat{\boldsymbol{v}} \in \mathbb{R}^{n_r}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\bar{\boldsymbol{Z}}^T(\bar{\mathbf{r}} - \bar{\boldsymbol{\Phi}}_r\hat{\boldsymbol{v}})\right\|_2^2, \tag{3.20}$$

where $\bar{\boldsymbol{Z}}^T := [\boldsymbol{e}_{p_1}, \ldots, \boldsymbol{e}_{p_{n_z}}]^T \in \mathbb{R}^{n_z \times N_s N_t}$, $n_{st} \le n_r \le n_z \ll N_s N_t$, is the sampling matrix and $\boldsymbol{e}_{p_i}$ is the p_ith column of the identity matrix $\boldsymbol{I}_{N_s N_t} \in \mathbb{R}^{N_s N_t \times N_s N_t}$. The solution to (3.20) is given by

$$\hat{\boldsymbol{r}} = (\bar{\boldsymbol{Z}}^T \bar{\boldsymbol{\Phi}}_r)^\dagger \bar{\boldsymbol{Z}}^T \bar{\mathbf{r}}.$$

Therefore, (3.19) becomes

$$\bar{\mathbf{r}} \approx \mathscr{P}_{\text{ST-GNAT}}\bar{\mathbf{r}}, \tag{3.21}$$

where $\mathscr{P}_{\text{ST-GNAT}} := \bar{\mathbf{\Phi}}_r(\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r)^\dagger\bar{\boldsymbol{Z}}^T$ is the ST-GNAT oblique projection matrix. Note that $\mathscr{P}_{\text{ST-GNAT}}$ has the same structure as $\mathscr{P}_{\text{GNAT}}$. The ST-GNAT method does not construct the sampling matrix $\bar{\boldsymbol{Z}}$. Instead, it maintains the sampling indices $\{p_1,\dots,p_{n_f}\}$ and corresponding rows of $\bar{\mathbf{\Phi}}_r$ and $\bar{\mathbf{r}}$. This enables the ST-GNAT to achieve a speed-up when it is applied to nonlinear problems.

Section 5.3 in [13] discusses three different options to determine the sampling indices (i.e., $\bar{\boldsymbol{Z}}$). However, all these three options are simple variations of Algorithm 3 in [11] and Algorithm 5 in [10]. They all minimize the error in the gappy reconstruction of the POD basis vectors for nonlinear space and time residuals. Therefore, the space-time nonlinear residual projection error due to (3.21) is similar to the ones in the GNAT method. That is,

$$\|\bar{\mathbf{r}} - \bar{\mathbf{\Phi}}_r(\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r)^\dagger\bar{\boldsymbol{Z}}^T\bar{\mathbf{r}}\|_2 \le \|\boldsymbol{R}^{-1}\|_2\|\bar{\mathbf{r}} - \bar{\mathbf{\Phi}}_r\bar{\mathbf{\Phi}}_r^T\bar{\mathbf{r}}\|_2,$$

where $\boldsymbol{R}$ is a triangle matrix from QR factorization of $\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r$ (i.e., $\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r = \boldsymbol{QR}$). On the other hand, one can also apply the sRRQR factorization in Algorithm 4 with a tuning parameter η of [22] to $\bar{\mathbf{\Phi}}_r^T$ to obtain $\bar{\boldsymbol{Z}}$ that is associated with a tighter error bound for the projection error:

$$\|\bar{\mathbf{r}} - \bar{\mathbf{\Phi}}_r(\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r)^\dagger\bar{\boldsymbol{Z}}^T\bar{\mathbf{r}}\|_2 \le \sqrt{1+\eta^2 n_r(N_s-n_r)}\|\bar{\mathbf{r}} - \bar{\mathbf{\Phi}}_r\bar{\mathbf{\Phi}}_r^T\bar{\mathbf{r}}\|_2. \tag{3.22}$$

This error bound can be obtained by setting the identity matrix as a weight matrix in Theorem 4.8 of [16].

Finally, the ST-GNAT solves the following least-squares problem at every time step, for example, with the backward Euler time integrator:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|(\bar{\boldsymbol{Z}}^T\bar{\mathbf{\Phi}}_r)^\dagger\bar{\boldsymbol{Z}}^T\bar{\mathbf{r}}_{\text{BE}}(\hat{\boldsymbol{v}};\boldsymbol{\mu})\right\|_2^2.$$

The original ST-GNAT paper introduces three different ways of collecting space-time residual snapshots, that are in turn used for the space-time residual basis construction (see section 5.2 in [13]). Below is a list of the approaches introduced in [13], explaining advantages and disadvantages of each:

1. *ST-LSPG ROM training iterations.* This approach takes the space-time residual snapshot from every Newton iteration of the ST-LSPG simulations at training points in $\mathscr{D}_{\text{train}}$. This case leads to the number of residual snapshots, $n_{\text{res}} = \sum_{\boldsymbol{\mu}\in\mathscr{D}_{\text{train}}}(k_{\max}(\boldsymbol{\mu})+1)$, where $k_{\max}(\boldsymbol{\mu})$ is the number of Newton iterations taken to solve the ST-LSPG simulation for the training point $\boldsymbol{\mu}$. This approach is not realistic for a large-scale problem because it requires $|\mathscr{D}_{\text{train}}|$ training simulations of the computationally expensive ST-LSPG ROM, where $|\mathscr{D}_{\text{train}}|$ denotes the cardinality of the set $\mathscr{D}_{\text{train}}$. Furthermore, it requires the extra SVD on the residual snapshots, which is not necessary for our proposed method.
2. *Projection of FOM training solutions.* This approach takes the following steps:

(a) Take the FOM state solution at every Newton iteration.
(b) Rearrange them in the space-time form (i.e., $\bar{\boldsymbol{x}}$ in (3.13)).[6]
(c) Project them onto the space-time subspace, $\mathcal{ST}$ (i.e., $\tilde{\boldsymbol{y}} = \bar{\boldsymbol{\Phi}}(\bar{\boldsymbol{\Phi}}^T\bar{\boldsymbol{\Phi}})^{-1}\bar{\boldsymbol{\Phi}}^T (\bar{\boldsymbol{x}} - \bar{\boldsymbol{x}}^0)$).
(d) Compute the corresponding space-time residual (e.g., $\bar{\boldsymbol{r}}(\tilde{\boldsymbol{y}};\boldsymbol{\mu})$ in (3.14) in the case of the backward Euler time integrator).
(e) Use those residuals as residual snapshots.

This approach simply requires n_{res} projections and evaluations of the space-time residual. However, it requires the extra SVD on the residual snapshots, which is not necessary for our proposed method.

3. *Random samples.* This approach generates a random space-time solution state sample (e.g., via Latin hypercube sampling or random sampling from uniform distribution) and computes the corresponding space-time residual (e.g., $\bar{\boldsymbol{r}}(\tilde{\boldsymbol{y}};\boldsymbol{\mu})$ in (3.14) in the case of the backward Euler time integrator). This approach simply requires n_{res} random sample generations and evaluations of the space-time residual. However, random samples are hardly correlated with actual data. Therefore, it is likely to generate poor space-time residual subspace. Furthermore, it requires the extra SVD on the residual snapshots, which is not necessary for our proposed method.

4. Solution-based nonlinear subspace (SNS) method.

Finally, we state our proposed method that avoids collecting nonlinear term snapshots and additional POD for the DEIM and GNAT approaches or additional tensor decomposition for the ST-GNAT method. We propose to use solution snapshots to construct nonlinear term basis in the DEIM, GNAT, and ST-GNAT approaches. A justification for using the solution snapshots comes from the subspace inclusion relation between the subspace spanned by the solution snapshots and the subspace spanned by the nonlinear term snapshots as shown in (2.3) and (2.5) with the forward and backward Euler time integrators.[7]

4.1. DEIM-SNS.

We are going back to (3.1) and replacing the nonlinear term with the approximation in (3.4):

$$\boldsymbol{M}\boldsymbol{\Phi}\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{\Phi}_f(\boldsymbol{Z}^T\boldsymbol{\Phi}_f)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}, t;\boldsymbol{\mu}). \tag{4.1}$$

Equation (4.1) is an overdetermined system, so it is likely that it will not have a solution. However, if there is a solution, then necessary conditions for (4.1) to have a nontrivial solution (i.e., $\hat{\boldsymbol{x}} \neq \boldsymbol{0}$) are $\boldsymbol{f}(\boldsymbol{x}_0, t;\boldsymbol{\mu}) \neq \boldsymbol{0}$ and

$$\operatorname{Range}\boldsymbol{M}\boldsymbol{\Phi} \cap \operatorname{Range}\boldsymbol{\Phi}_f \neq \{\boldsymbol{0}\}.$$

The second condition says that the intersection of Range $\boldsymbol{M}\boldsymbol{\Phi}$ and Range $\boldsymbol{\Phi}_f$ needs to be nontrivial if there is a nontrivial solution to (4.1). Typically, we build $\boldsymbol{\Phi}$ first, using

[6]Note that these are FOM solutions from time marching algorithms in which each time step results in a different number of Newton iterations if implicit time integrators are used. Some time steps take a smaller number of Newton iterations than other time steps. However, in order to rearrange each Newton iterate state in the space-time form, we must have the same number of Newton iterations at each time step. Therefore, for the time steps that have converged with a smaller number of Newton iterations than other time steps, we pad the solution state vectors of the Newton iterations beyond the convergence with the converged solution. This only applies to an implicit time integrator because an explicit time integrator does not require any Newton solve.

[7]Subspace inclusion relations for other time integrators are shown in Appendix A.

the POD approach explained in section 3.1 and Range $\boldsymbol{M\Phi}$ is set by $\boldsymbol{\Phi}$. Therefore, the intersection of Range $\boldsymbol{M\Phi}$ and Range $\boldsymbol{\Phi}_f$ can be controlled by the choice of $\boldsymbol{\Phi}_f$ we made. The larger the intersection of those two range spaces is, the more likely it is that there is a solution to (4.1). Given $\boldsymbol{\Phi}$, the largest subspace intersection it can be is Range $\boldsymbol{M\Phi}$, i.e.,

$$\operatorname{Range}\boldsymbol{M\Phi}\cap\operatorname{Range}\boldsymbol{\Phi}_f=\operatorname{Range}\boldsymbol{M\Phi}.$$

We call this condition the *conforming subspace condition*. The conforming subspace condition leads to two obvious choices for $\boldsymbol{\Phi}_f$:

- The first choice is to ensure Range $\boldsymbol{M\Phi}$ = Range $\boldsymbol{\Phi}_f$. If $\boldsymbol{\Phi}_f=\boldsymbol{M\Phi}$, then the range spaces of the left- and right-hand sides of (4.1) are the same. This leads (4.1) to become

$$\boldsymbol{M\Phi}\frac{d\hat{\boldsymbol{x}}}{dt}=\boldsymbol{M\Phi}(\boldsymbol{Z}^T\boldsymbol{M\Phi})^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.2}$$

  Because (4.2) is an overdetermined system and unlikely to have a solution, applying the Galerkin projection to (4.2) becomes

$$\boldsymbol{\Phi}^T\boldsymbol{M\Phi}\frac{d\hat{\boldsymbol{x}}}{dt}=\boldsymbol{\Phi}^T\boldsymbol{M\Phi}(\boldsymbol{Z}^T\boldsymbol{M\Phi})^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.3}$$

  Assuming $\boldsymbol{\Phi}^T\boldsymbol{M\Phi}$ is invertible, (4.3) becomes

$$\frac{d\hat{\boldsymbol{x}}}{dt}=(\boldsymbol{Z}^T\boldsymbol{M\Phi})^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.4}$$

  For the special case of $\boldsymbol{M}$ being an identity matrix, (4.4) becomes

$$\frac{d\hat{\boldsymbol{x}}}{dt}=(\boldsymbol{Z}^T\boldsymbol{\Phi})^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}).$$

- The second choice is to ensure Range $\boldsymbol{M\Phi}\subset$ Range $\boldsymbol{\Phi}_f$. This can be achieved by taking an extended solution basis, $\boldsymbol{\Phi}_e\in\mathbb{R}^{N_s\times n_e}$ with $n_s<n_e\ll N_s$ and Range $\boldsymbol{\Phi}\subset$ Range $\boldsymbol{\Phi}_e$. Then we set $\boldsymbol{\Phi}_f=\boldsymbol{M\Phi}_e$, which leads to Range $\boldsymbol{M\Phi}\subset$ Range $\boldsymbol{M\Phi}_e$. The obvious choice for $\boldsymbol{\Phi}_e$ is to take a larger truncation of the left singular matrix from SVD of the solution snapshot matrix than $\boldsymbol{\Phi}$. Note that this particular choice of $\boldsymbol{\Phi}_e$ results in the first n_s columns of $\boldsymbol{\Phi}_e$ being the same as $\boldsymbol{\Phi}$ (i.e., $\boldsymbol{\Phi}_e=\begin{bmatrix}\boldsymbol{\Phi} & \boldsymbol{\Phi}_E\end{bmatrix}$ where $\boldsymbol{\Phi}_E\in\mathbb{R}^{N_s\times n_E}$ with $n_E=n_e-n_s$ and $\boldsymbol{\Phi}^T\boldsymbol{\Phi}_E=\boldsymbol{0}\in\mathbb{R}^{n_s\times n_E}$). By setting $\boldsymbol{\Phi}_f=\boldsymbol{M\Phi}_e$, (4.1) becomes

$$\boldsymbol{M\Phi}\frac{d\hat{\boldsymbol{x}}}{dt}=\boldsymbol{M\Phi}_e(\boldsymbol{Z}^T\boldsymbol{M\Phi}_e)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.5}$$

  Because (4.5) is unlikely to have a solution, applying the Galerkin projection to (4.5) becomes

$$\begin{aligned}\boldsymbol{\Phi}^T\boldsymbol{M\Phi}\frac{d\hat{\boldsymbol{x}}}{dt}&=\boldsymbol{\Phi}^T\boldsymbol{M\Phi}_e(\boldsymbol{Z}^T\boldsymbol{M\Phi}_e)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu})\\&=\boldsymbol{\Phi}^T\boldsymbol{M}\begin{bmatrix}\boldsymbol{\Phi} & \boldsymbol{\Phi}_E\end{bmatrix}(\boldsymbol{Z}^T\boldsymbol{M\Phi}_e)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}).\end{aligned} \tag{4.6}$$

  Assuming $\boldsymbol{\Phi}^T\boldsymbol{M\Phi}$ is invertible, (4.6) becomes

$$\frac{d\hat{\boldsymbol{x}}}{dt}=\begin{bmatrix}\boldsymbol{I}_{n_s} & (\boldsymbol{\Phi}^T\boldsymbol{M\Phi})^{-1}(\boldsymbol{\Phi}^T\boldsymbol{M\Phi}_E)\end{bmatrix}(\boldsymbol{Z}^T\boldsymbol{M\Phi}_e)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.7}$$

For the special case of $\boldsymbol{M}$ being an identity matrix, (4.7) becomes

$$\frac{d\hat{\boldsymbol{x}}}{dt} = \begin{bmatrix}\boldsymbol{I}_{n_s} & \boldsymbol{0}\end{bmatrix} (\boldsymbol{Z}^T\boldsymbol{\Phi}_e)^{-1}\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}},t;\boldsymbol{\mu}). \tag{4.8}$$

The DEIM-SNS approach solves either (4.4) or (4.7) depending on the choice of $\boldsymbol{\Phi}_f$ above. Applying a time integrator to (4.4) or (4.8) leads to a reduced OΔE, whose solution, $\hat{\boldsymbol{x}}_\star$, can be lifted to find the full order approximate solution via $\tilde{\boldsymbol{x}}_\star = \boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}_\star$.

Additionally, the subspace inclusion relations[8] show that the subspace spanned by the solution snapshots includes the subspace spanned by the nonlinear term snapshots. This fact further motivates the use of solution snapshots to build a nonlinear term basis. Indeed, numerical experiments show that the solution accuracy obtained by the DEIM-SNS approach is comparable to that obtained by the traditional DEIM approach. For example, see Figures 3 and 4 in section 6.1.1.

The obvious advantage of the DEIM-SNS approach over DEIM is that no additional SVD or eigenvalue decomposition is required, which can save the computational cost of the offline phase.

Remark 4.1. Although the numerical experiments show that the two choices of $\boldsymbol{\Phi}_f$ above give promising results, the error analysis in section 5 shows that the nonlinear term projection error bound increases by the condition number of $\boldsymbol{M}$ (see Theorem 5.2). This is mainly because the orthogonality of $\boldsymbol{\Phi}_f$ is lost when $\boldsymbol{\Phi}_f = \boldsymbol{M\Phi}$ or $\boldsymbol{M\Phi}_e$. This issue is resolved by orthogonalizing $\boldsymbol{\Phi}_f$ (e.g., apply economic QR factorization $\boldsymbol{\Phi}_f = \boldsymbol{QR}$) before using it as nonlinear term basis. Then apply, for example, Algorithm 4 of [22] to the transpose of the orthogonalized one (e.g., $\boldsymbol{Q}^T$) to generate a sampling matrix $\boldsymbol{Z}$. This procedure eliminates the condition number of $\boldsymbol{M}$ in the error bound because (3.6) is valid.

Remark 4.2. Inspired by the weighted inner product space introduced for DEIM in [16], another oblique DEIM-SNS projection is possible. With the weight matrix $\boldsymbol{W} = \boldsymbol{M}^{-T}\boldsymbol{M}^{-1}$, the selection operator $\boldsymbol{S}^T = \boldsymbol{Z}^T\boldsymbol{M}^T$, and the basis $\hat{\boldsymbol{U}} = \boldsymbol{M\Phi}$ according to section 4.2 of [16], the weighted oblique DEIM-SNS projection can be defined as

$$\begin{aligned}\mathscr{P}_{\mathrm{SNS}} &= \hat{\boldsymbol{U}}(\boldsymbol{S}^T\boldsymbol{W}\hat{\boldsymbol{U}})^\dagger\boldsymbol{S}^T\boldsymbol{W}\\ &= \boldsymbol{M\Phi}(\boldsymbol{Z}^T\boldsymbol{\Phi})^\dagger\boldsymbol{Z}^T\boldsymbol{M}^{-1}.\end{aligned}$$

4.2. GNAT-SNS.

The GNAT method needs to build a nonlinear residual term basis, $\boldsymbol{\Phi}_r$, as described in section 3.2. The nonlinear residual term is nothing more than linear combinations of the nonlinear term and the time derivative approximation as in (3.7). Thus, the subspace spanned by the nonlinear term residual snapshots is included in the subspace spanned by the solution snapshots. This motivates the use of the solution snapshots for the construction of a nonlinear residual term basis. Therefore, the same type of the nonlinear term basis in the DEIM-SNS approach can be used to construct the nonlinear residual term basis in the GNAT-SNS method:

- The first choice is to set $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}$. Thus, for example, the GNAT-SNS method solves the following least-squares problem with the backward Euler time integrator:

[8] Equation (2.3) of the forward Euler time integrator, (A.2) of the Adams–Moulton time integrator, and (A.7) of the midpoint Runge–Kutta method show the subspace inclusion relations for explicit time integrators.

$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\boldsymbol{M\Phi}(\boldsymbol{Z}^T\boldsymbol{M\Phi})^\dagger\boldsymbol{Z}^T(\boldsymbol{M\Phi}(\hat{\boldsymbol{v}}-\hat{\boldsymbol{x}}_{n-1})-\Delta t\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu}))\right\|_2^2,$$

which can be manipulated to

(4.9)
$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\boldsymbol{M\Phi}(\hat{\boldsymbol{v}}-\hat{\boldsymbol{x}}_{n-1})-\Delta t\boldsymbol{M\Phi}(\boldsymbol{Z}^T\boldsymbol{M\Phi})^\dagger\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu})\right\|_2^2.$$

Note that the terms in the ℓ_2 norm in (4.9) are very similar to the discretized DEIM residual before Galerkin projection (i.e., applying the backward Euler time integrator to (4.2) gives the terms in the ℓ_2 norm in (4.9)). They are only different by the fact that one is an inverse and the other one is the Moore–Penrose inverse. In fact, if $n_z = n_r$, then (4.9) is equivalent to applying the backward Euler time integrator to (4.2) and minimizing the ℓ_2 norm of the corresponding residual.

For the special case of $\boldsymbol{M}$ being an identity matrix, (4.9) becomes

$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|\hat{\boldsymbol{v}}-\hat{\boldsymbol{x}}_{n-1}-\Delta t(\boldsymbol{Z}^T\boldsymbol{\Phi})^\dagger\boldsymbol{Z}^T\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu})\right\|_2^2. \tag{4.10}$$

- The second choice is to set $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}_e$ where $\boldsymbol{\Phi}_e = \begin{bmatrix}\boldsymbol{\Phi} & \boldsymbol{\Phi}_E\end{bmatrix}$ as in section 4.1. This leads to the following least-squares problem, for example, using the backward Euler time integrator:

(4.11)
$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \ \frac{1}{2}\left\|\boldsymbol{M\Phi}_e(\boldsymbol{Z}^T\boldsymbol{M\Phi}_e)^\dagger\boldsymbol{Z}^T(\boldsymbol{M\Phi}(\hat{\boldsymbol{v}}-\hat{\boldsymbol{x}}_{n-1})-\Delta t\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu}))\right\|_2^2.$$

For the special case of $\boldsymbol{M}$ being an identity matrix, (4.11) becomes

(4.12)
$$\hat{\boldsymbol{x}}_n = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\|(\boldsymbol{Z}^T\boldsymbol{\Phi}_e)^\dagger\boldsymbol{Z}^T(\boldsymbol{\Phi}(\hat{\boldsymbol{v}}-\hat{\boldsymbol{x}}_{n-1})-\Delta t\boldsymbol{f}(\boldsymbol{x}_0+\boldsymbol{\Phi}\hat{\boldsymbol{x}}_n,t;\boldsymbol{\mu}))\right\|_2^2.$$

The GNAT-SNS method solves either (4.9) or (4.11) depending on the choice of $\boldsymbol{\Phi}_r$ above. For the special case of $\boldsymbol{M}$ being an identity matrix, the GNAT-SNS method solves either (4.10) or (4.12) depending on the choice of $\boldsymbol{\Phi}_r$. The reduced solution $\hat{\boldsymbol{x}}_n$ can be lifted to find the full order approximate solution via $\tilde{\boldsymbol{x}}_n = \boldsymbol{x}_0 + \boldsymbol{\Phi}\hat{\boldsymbol{x}}_n$.

Remark 4.3. Similar to Remark 4.1, the error analysis in section 5 shows that the nonlinear residual projection error bound, regarding the two choices of $\boldsymbol{\Phi}_r$ above, increases by the condition number of $\boldsymbol{M}$ (see Theorem 5.3). This is mainly because the orthogonality of $\boldsymbol{\Phi}_r$ is lost when $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}$ or $\boldsymbol{M\Phi}_e$. This issue is resolved by orthogonalizing $\boldsymbol{\Phi}_r$ (e.g., applying economic QR factorization $\boldsymbol{\Phi}_r = \boldsymbol{QR}$) before using it as a nonlinear residual basis. Then apply, for example, Algorithm 4 of [22] to the transpose of orthogonalized one (e.g., $\boldsymbol{Q}^T$) to generate a sampling matrix $\boldsymbol{Z}$. This procedure eliminates the condition number of $\boldsymbol{M}$ in the error bound because (3.11) is valid.

4.3. ST-GNAT-SNS. The ST-GNAT method needs to build a space-time nonlinear residual term basis, $\bar{\boldsymbol{\Phi}}_r$, as described in section 3.3. We are going back to (3.17) to find the conforming subspace condition for ST-GNAT-SNS. In order to increase the chance of making the space-time residual function defined in (3.17) zero, the following conforming subspace condition can be made:

$$\text{Range}\,\boldsymbol{A}_{\text{BE}}\bar{\boldsymbol{\Phi}} \cap \text{Range}\,\bar{\boldsymbol{\Phi}}_r = \text{Range}\,\boldsymbol{A}_{\text{BE}}\bar{\boldsymbol{\Phi}}.$$

Therefore, we propose the following bases of the space-time nonlinear residual term:

- The first choice is to set $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}$. Thus, for example, the ST-GNAT-SNS method solves the following least-squares problem with the backward Euler time integrator:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| (\bar{\boldsymbol{Z}}^T\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}})^\dagger \bar{\boldsymbol{Z}}^T(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t\bar{\boldsymbol{f}}(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu}))\right\|_2^2,$$

  which can be rewritten as

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| \hat{\boldsymbol{y}} - \Delta t(\bar{\boldsymbol{Z}}^T\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}})^\dagger \bar{\boldsymbol{Z}}^T\bar{\boldsymbol{f}}(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu})\right\|_2^2.$$

  This is what the ST-GNAT-SNS method solves if $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}$.
- The second choice is to set $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}_e$ where $\bar{\boldsymbol{\Phi}}_e \in \mathbb{R}^{N_sN_t\times n_e}$ with $N_sN_t \gg n_z \geq n_e > n_{st}$ and Range $\bar{\boldsymbol{\Phi}} \subset$ Range $\bar{\boldsymbol{\Phi}}_e$. The obvious choice for $\bar{\boldsymbol{\Phi}}_e$ is to take a larger truncation of the factor matrices from the tensor decomposition (e.g., ST-HOSVD and LL1) of the solution snapshot tensor than $\bar{\boldsymbol{\Phi}}$. Note that this particular choice of $\bar{\boldsymbol{\Phi}}_e$ results in the first n_{st} columns of $\bar{\boldsymbol{\Phi}}_e$ being the same as $\bar{\boldsymbol{\Phi}}$ (i.e., $\bar{\boldsymbol{\Phi}}_e = \begin{bmatrix}\bar{\boldsymbol{\Phi}} & \bar{\boldsymbol{\Phi}}_E\end{bmatrix}$ where $\bar{\boldsymbol{\Phi}}_E \in \mathbb{R}^{N_sN_t\times n_E}$ with $n_E = n_e - n_{st}$). In this case, the ST-GNAT-SNS method solves the following least-squares problem:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| (\bar{\boldsymbol{Z}}^T\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}_e)^\dagger \bar{\boldsymbol{Z}}^T(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t\bar{\boldsymbol{f}}(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu}))\right\|_2^2,$$

In addition to the choices above, we propose the following two choices for the special case of $\boldsymbol{M}$ being an identity matrix:

- The first choice is to set $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}$. Thus, for example, the ST-GNAT-SNS method solves the following least-squares problem with the backward Euler time integrator:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| (\bar{\boldsymbol{Z}}^T\bar{\boldsymbol{\Phi}})^\dagger \bar{\boldsymbol{Z}}^T(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t\bar{\boldsymbol{f}}(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu}))\right\|_2^2.$$

- The second choice is to set $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}_e$. In this case, the ST-GNAT-SNS method solves the following least-squares problem:

$$\hat{\boldsymbol{y}}(\boldsymbol{\mu}) = \underset{\hat{\boldsymbol{v}}\in\mathbb{R}^{n_s}}{\operatorname{argmin}} \quad \frac{1}{2}\left\| (\bar{\boldsymbol{Z}}^T\bar{\boldsymbol{\Phi}}_e)^\dagger \bar{\boldsymbol{Z}}^T(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}} - \Delta t\bar{\boldsymbol{f}}(\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{x}}^0 + \bar{\boldsymbol{\Phi}}\hat{\boldsymbol{y}};\boldsymbol{\mu}))\right\|_2^2.$$

The space-time generalized coordinates, $\hat{\boldsymbol{y}}$, can be lifted to the approximate full space-time solution $\tilde{\boldsymbol{y}}(\boldsymbol{\mu})$ via (3.15).

Remark 4.4. Similar to Remarks 4.1 and 4.3, the error analysis in section 5 shows that the nonlinear space-time residual projection error bound, regarding the first two choices of $\bar{\boldsymbol{\Phi}}_r$ above, increases by the condition number of $\boldsymbol{A}_{\mathrm{BE}}$ (see Theorem 5.4). This is mainly because the orthogonality of $\bar{\boldsymbol{\Phi}}_r$ is lost when $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}$ or $\boldsymbol{A}_{\mathrm{BE}}\bar{\boldsymbol{\Phi}}_e$. This issue is resolved by orthogonalizing $\bar{\boldsymbol{\Phi}}_r$ (e.g., apply economic QR factorization $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{QR}$)[9] before using it as a nonlinear residual basis. Then apply, for example, Algorithm 4 of [22] to the transpose of the orthogonalized one (e.g., $\boldsymbol{Q}^T$) to generate a sampling matrix $\bar{\boldsymbol{Z}}$. This procedure eliminates the condition number of $\boldsymbol{A}_{\mathrm{BE}}$ in the error bound because (3.22) is valid.

[9]This might be challenging because of the size of $\bar{\boldsymbol{\Phi}}_r$. However, a parallel QR factorization can be used to compute QR efficiently for a large matrix, such as [8, 17]

5. Error analysis.

Section 4 introduced the SNS method. If $\boldsymbol{M} = \boldsymbol{I}$, then all the same error analysis presented in section 3 holds by replacing $\boldsymbol{\Phi}_f$, $\boldsymbol{\Phi}_r$ with $\boldsymbol{\Phi}$ or $\boldsymbol{\Phi}_e$ for DEIM-SNS and GNAT-SNS and $\bar{\boldsymbol{\Phi}}_r$ with $\bar{\boldsymbol{\Phi}}$ or $\bar{\boldsymbol{\Phi}}_e$ for ST-GNAT-SNS. However, if $\boldsymbol{M} \neq \boldsymbol{I}$ is a general nonsingular matrix, then the error analysis has to be revisited. The error analysis below is inspired by [12, 15, 16].

LEMMA 5.1. *Let $\boldsymbol{v} \in \mathbb{R}^N$ be an arbitrary vector; $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ is a nonsingular matrix; $\boldsymbol{\Upsilon} \in \mathbb{R}^{N\times n}$ denotes a full rank matrix; $\boldsymbol{Z} \in \mathbb{R}^{N\times s}$, $N > s \geq n$ denotes a sampling matrix, defined in section* 3. *Let $\mathbb{P} \in \mathbb{R}^{N\times N}$ be an oblique projection matrix, defined as $\mathbb{P} := \boldsymbol{K\Upsilon}(\boldsymbol{Z}^T\boldsymbol{K\Upsilon})^\dagger\boldsymbol{Z}^T$; let $\mathbb{P}_\star \in \mathbb{R}^{N\times N}$ be another oblique projection matrix onto* range $\boldsymbol{K\Upsilon}$ *(i.e., $\mathbb{P}_\star := \boldsymbol{K\Upsilon}(\boldsymbol{K\Upsilon})^\dagger$). Then, a projection error bound is given by*

$$\|(\boldsymbol{I}_N - \mathbb{P})\boldsymbol{v}\|_2 \leq \|(\boldsymbol{Z}^T\boldsymbol{Q})^\dagger\|_2\|(\boldsymbol{I}_N - \mathbb{P}_\star)\boldsymbol{v}\|_2, \tag{5.1}$$

where $\boldsymbol{Q}$ is obtained from a QR factorization of $\boldsymbol{K\Upsilon}$, i.e., $\boldsymbol{K\Upsilon} := \boldsymbol{QR}$.

Proof. Note that $\mathbb{P}\mathbb{P}_\star = \mathbb{P}_\star$ is true because $\boldsymbol{Z}^T\boldsymbol{K\Upsilon}$ has full column rank. Thus, $(\boldsymbol{I}_N - \mathbb{P})\mathbb{P}_\star = \boldsymbol{0}$. This leads to

$$(\boldsymbol{I}_N - \mathbb{P})\boldsymbol{v} = (\boldsymbol{I}_N - \mathbb{P})(\boldsymbol{I}_N - \mathbb{P}_\star)\boldsymbol{v}.$$

Because $\mathbb{P} \neq \boldsymbol{0}$, $\mathbb{P} \neq \boldsymbol{I}_N$, it holds that $\|\mathbb{P}\|_2 = \|\boldsymbol{I}_N - \mathbb{P}\|_2$. Note also that $\|\mathbb{P}\|_2 \leq \|(\boldsymbol{Z}^T\boldsymbol{Q})^\dagger\|_2$ is true because $\mathbb{P} = \boldsymbol{Q}(\boldsymbol{Z}^T\boldsymbol{Q})^\dagger\boldsymbol{Z}^T$, which gives (5.1). ∎

THEOREM 5.2. *Let the DEIM-SNS projection matrix be $\mathbb{P} = \boldsymbol{\Phi}_f(\boldsymbol{Z}^T\boldsymbol{\Phi}_f)^{-1}\boldsymbol{Z}^T \in \mathbb{R}^{N_s\times N_s}$ and $\mathbb{P}_\star = \boldsymbol{\Phi}_f(\boldsymbol{\Phi}_f)^\dagger$. If the sampling matrix $\boldsymbol{Z} \in \mathbb{R}^{N_s\times s}$ is constructed using sRRQR factorization of $\boldsymbol{Q}$ or $\boldsymbol{Q}_e$, where $\boldsymbol{Q}$ and $\boldsymbol{Q}_e$ are obtained from QR factorizations of $\boldsymbol{M\Phi}$ and $\boldsymbol{M\Phi}_e$, respectively, with a tuning parameter, η, as in* [16] *(i.e., $s = n_f$) or oversampled (i.e., $N_s \geq s > n_f$), then the DEIM-SNS method with either $\boldsymbol{\Phi}_f = \boldsymbol{M\Phi}$ or $\boldsymbol{\Phi}_f = \boldsymbol{M\Phi}_e$ has a nonlinear term projection error bound for a given vector $\boldsymbol{f} \in \mathbb{R}^{N_s}$:*

$$\|(\boldsymbol{I}_{N_s} - \mathbb{P})\boldsymbol{f}\|_2 \leq \sqrt{1 + \eta^2 n_f(N_s - n_f)}\|(\boldsymbol{I}_{N_s} - \mathbb{P}_\star)\boldsymbol{f}\|_2.$$

Proof. Combining Lemma 5.1 with $\boldsymbol{K} = \boldsymbol{M}$, $\boldsymbol{\Upsilon} = \boldsymbol{\Phi}$, or $\boldsymbol{\Phi}_e$ in section 4.1 and $N = N_s$, and $n = n_f$ in section 3.1 and applying sPPQR factorization of [16] to either $\boldsymbol{Q}$ or $\boldsymbol{Q}_e$ and noting that $\|(\boldsymbol{Z}^T\boldsymbol{Q}_e)^\dagger\|_2 \leq \|(\boldsymbol{Z}^T\boldsymbol{Q})^{-1}\|_2$ proves the bound. ∎

THEOREM 5.3. *Let the GNAT-SNS projection matrix be $\mathbb{P} = \boldsymbol{\Phi}_r(\boldsymbol{Z}^T\boldsymbol{\Phi}_r)^{-1}\boldsymbol{Z}^T \in \mathbb{R}^{N_s\times N_s}$ and $\mathbb{P}_\star = \boldsymbol{\Phi}_r(\boldsymbol{\Phi}_r)^\dagger$. Set the sampling matrix $\boldsymbol{Z}$. If the sampling matrix $\boldsymbol{Z} \in \mathbb{R}^{N_s\times s}$ is constructed using sRRQR factorization of $\boldsymbol{Q}$ or $\boldsymbol{Q}_e$, where $\boldsymbol{Q}$ and $\boldsymbol{Q}_e$ are obtained from QR factorizations of $\boldsymbol{M\Phi}$ and $\boldsymbol{M\Phi}_e$, respectively, with a tuning parameter, η, as in* [16] *(i.e., $s = n_r$) or oversampled (i.e., $N_s \geq s > n_r$), then the GNAT-SNS method with either $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}$ or $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}_e$ has a nonlinear residual term projection error bound for a given vector $\boldsymbol{r} \in \mathbb{R}^{N_s}$:*

$$\|(\boldsymbol{I}_{N_s} - \mathbb{P})\boldsymbol{r}\|_2 \leq \sqrt{1 + \eta^2 n_r(N_s - n_r)}\|(\boldsymbol{I}_{N_s} - \mathbb{P}_\star)\boldsymbol{r}\|_2.$$

Proof. Combining Lemma 5.1 with $\boldsymbol{K} = \boldsymbol{M}$, $\boldsymbol{\Upsilon} = \boldsymbol{\Phi}$, or $\boldsymbol{\Phi}_e$ in section 4.2 and $N = N_s$, and $n = n_r$ in section 3.2 and applying sPPQR factorization of [16] to either $\boldsymbol{Q}$ or $\boldsymbol{Q}_e$ and noting that $\|(\boldsymbol{Z}^T\boldsymbol{Q}_e)^\dagger\|_2 \leq \|(\boldsymbol{Z}^T\boldsymbol{Q})^{-1}\|_2$ proves the bound. ∎

THEOREM 5.4. *Let the ST-GNAT-SNS projection matrix be $\mathbb{P}=\bar{\boldsymbol{\Phi}}_r(\bar{\boldsymbol{Z}}^T\bar{\boldsymbol{\Phi}}_r)^{-1}\bar{\boldsymbol{Z}}^T \in \mathbb{R}^{N_sN_t\times N_sN_t}$ and $\mathbb{P}_\star = \bar{\boldsymbol{\Phi}}_r(\bar{\boldsymbol{\Phi}}_r)^\dagger$. If the sampling matrix $\bar{\boldsymbol{Z}}^T \in \mathbb{R}^{N_sN_t\times s}$ is constructed using sRRQR factorization of $\bar{\boldsymbol{Q}}$ or $\bar{\boldsymbol{Q}}_e$, where $\bar{\boldsymbol{Q}}$ and $\bar{\boldsymbol{Q}}_e$ are obtained from QR* *factorizations of $\boldsymbol{A}_{BE}\bar{\boldsymbol{\Phi}}$ and $\boldsymbol{A}_{BE}\bar{\boldsymbol{\Phi}}_e$, respectively, with a tuning parameter, η, as in* [16] *(i.e., $s = n_r$) or oversampled (i.e., $N_sN_t \geq s > n_r$), then the ST-GNAT-SNS method with either $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{BE}\bar{\boldsymbol{\Phi}}$ or $\bar{\boldsymbol{\Phi}}_r = \boldsymbol{A}_{BE}\bar{\boldsymbol{\Phi}}_e$ has a nonlinear space-time residual term projection error bound for a given vector $\bar{\boldsymbol{r}} \in \mathbb{R}^{N_sN_t}$:*

$$\|(\boldsymbol{I}_{N_sN_t} - \mathbb{P})\bar{\boldsymbol{r}}\|_2 \leq \sqrt{1+\eta^2 n_r(N_sN_t - n_r)}\|(\boldsymbol{I}_{N_sN_t} - \mathbb{P}_\star)\bar{\boldsymbol{r}}\|_2.$$

Proof. Combining Lemma 5.1 with $\boldsymbol{K} = \boldsymbol{A}_{\mathrm{BE}}$, $\boldsymbol{\Upsilon} = \bar{\boldsymbol{\Phi}}$, or $\bar{\boldsymbol{\Phi}}_e$ in section 4.3 and $N = N_sN_t$, and $n = n_r$ in section 3.3 and applying sPPQR factorization to either $\bar{\boldsymbol{Q}}$ or $\bar{\boldsymbol{Q}}_e$ and noting that $\|(\bar{\boldsymbol{Z}}^T\bar{\boldsymbol{Q}}_e)^\dagger\|_2 \leq \|(\bar{\boldsymbol{Z}}^T\bar{\boldsymbol{Q}})^{-1}\|_2$ proves the bound. ◻

6. Numerical results.

In this section, we demonstrate that the SNS methods reduce the offline computational time without losing accuracy of the DEIM, GNAT, and ST-GNAT approaches. The focus of the numerical experiments is not to show the accuracy and speed-up of all the model order reduction techniques considered in the paper. For the benefits of the DEIM, GNAT, and ST-GNAT methods in terms of accuracy and speed-up, we refer to their original papers [12, 11, 13]. For the DEIM method, which does not require collecting the nonlinear term snapshots from simulations other than the corresponding FOM, the only offline computational time reduction comes from the fact that the SNS methods require only one compression of the only solution snapshots instead of two compressions that are required in the DEIM method. Therefore, the compression computational time reduction due to the DEIM-SNS method is about a factor of two (e.g., see Figures 3(c) and 4(c)). For the GNAT and ST-GNAT methods, on the other hand, the offline computational time reduction due to the SNS methods is large because the GNAT and ST-GNAT require collecting nonlinear residual term snapshots from their corresponding LSPG and ST-LSPG simulations for the best performance, but the SNS methods do not require that. Therefore, the numerical experiments show that the offline computational time reduction is from a factor of three to a hundred (e.g., see Figures 8, 9, 11, 14(b), and 16).

We consider three different problems: a 2D nonlinear diffusion problem is solved in section 6.1, a parameterized 1D Burgers' equation is solved in section 6.2, and a parameterized quasi-1D Euler equation is solved in section 6.3. The performance of the DEIM and DEIM-SNS approaches is compared in section 6.1. The performance of the GNAT and GNAT-SNS methods is compared in sections 6.2 and 6.3. The performance of the ST-GNAT and ST-GNAT-SNS methods is compared in sections 6.2 and 6.3.

The following greedy algorithms for constructing sample indices are used for each method:

- Algorithm 1 in [12] with the DEIM and DEIM-SNS approaches,
- Algorithm 3 in [11] with the GNAT and GNAT-SNS methods,
- Algorithms 1 and 2 in [13] with the ST-GNAT and ST-GNAT-SNS methods.

Procedure identifier 1 in Table 1 of [11] is used for residual snapshot-collection procedures for GNAT. The accuracy of any ROM solution $\tilde{\boldsymbol{x}}(\cdot;\boldsymbol{\mu})$ is assessed from its mean squared state-space error:

$$\text{relative error} = \sqrt{\sum_{n=1}^{N_t} \|\tilde{\boldsymbol{x}}(t^n;\boldsymbol{\mu}) - \boldsymbol{x}(t^n;\boldsymbol{\mu})\|_2^2} \Bigg/ \sqrt{\sum_{n=1}^{N_t} \|\boldsymbol{x}(t^n;\boldsymbol{\mu})\|_2^2}\ .$$

We measure the computational offline cost in terms of the wall time. All timings with the GNAT, ST-GNAT, GNAT-SNS, and ST-GNAT-SNS methods are obtained by performing calculations on an Intel Core i7 CPU @ 2.5 GHz, 16 GB 1600 MHz DDR3 using the modified version of `MORTestbed`[34] in MATLAB. All the timings with the DEIM and DEIM-SNS approaches are obtained by performing calculations on the Quartz cluster at Lawrence Livermore National Laboratory using MFEM-based ROM house code [2]. Quartz has 2634 nodes, each with an Intel Xeon E5-2695 with 36 cores operating at 2.1 GHz and 128 GB RAM memory. The time units of Figures 8, 9, 11, and 16 are seconds.

6.1. Nonlinear diffusion equation. We now consider a parameterized 2D nonlinear diffusion equation associated with the problem of time-dependent nonlinear heat conduction. The problem corresponds to the following initial boundary value problem on the unit square and $t \in [0, T]$:

$$\frac{\partial u}{\partial t} = \nabla \cdot (\kappa + \alpha u)\nabla u \quad \forall (x, y) \in D = [0, 1]m \times [0, 1]m, \quad \forall t \in [0, T], \tag{6.1}$$

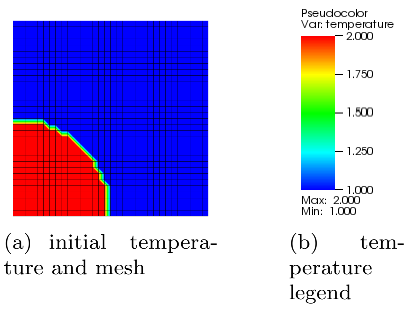
where $u((x, y), t) \in H^1(D)$ denotes the space and time-dependent temperature function with $u : D \times [0, 1] \to \mathbb{R}$ implicitly defined as the solution to (6.1). The diffusivity depends linearly on u with coefficients, $\kappa = 0.5\ m^2/s$ and $\alpha = 0.01\ m^2/(s \cdot K)$. Zero temperature gradient boundaries are employed and the simulation is initialized by a step function defined on a quarter circle given by

$$\frac{\partial u}{\partial n} = 0 \quad \text{on} \quad \Gamma = \{(x, y) | x \in \{0, 1\}, y \in \{0, 1\}\},$$

$$u(x, y, 0; \mu) = \begin{cases} 2 & \text{if } x^2 + y^2 \le 0.5^2, \\ 1 & \text{if otherwise.} \end{cases}$$

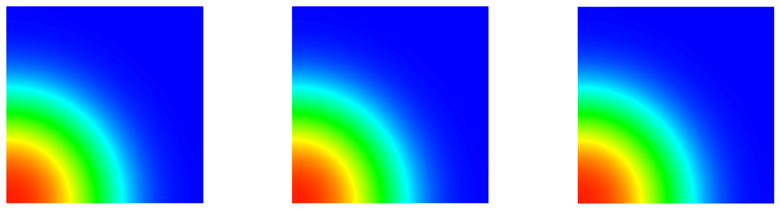
After applying a linear finite-element spatial discretization with $N_s = 1089$ (32 elements at each side; see Figure 1 for the mesh). Equation (6.1) leads to an initial-value ODE problem consistent with (2.1) with $\boldsymbol{M}$ being a volume matrix. For time discretization, the forward and backward Euler schemes are applied with a uniform time step. The solution basis dimension is set $n_s = 20$.

6.1.1. DEIM versus DEIM-SNS. The DEIM and DEIM-SNS approaches are compared numerically. For this parabolic problem, the DEIM and DEIM-SNS approaches try to reproduce the solution of the corresponding high fidelity model with



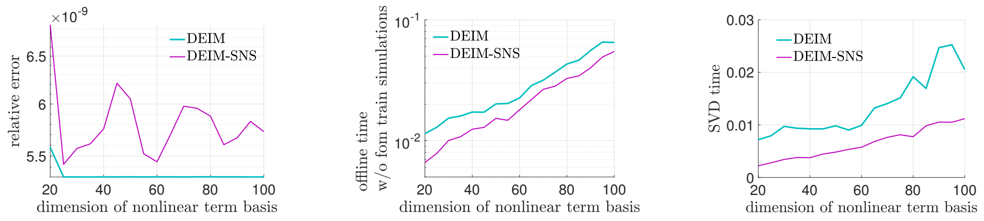
(a) initial temperature and mesh (b) temperature legend

Fig. 1. *Initial temperature distribution, mesh, and legend.*



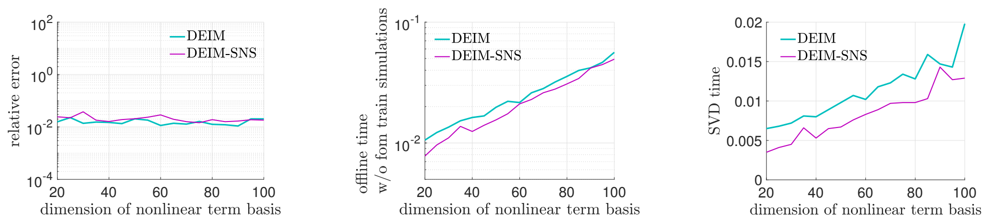
(a) FOM temperature (b) DEIM temperature (c) DEIM-SNS temperature

FIG. 2. *Temperature distribution using the forward Euler time integrator. For the DEIM and DEIM-SNS approaches, $n_s = 20$ and $n_r = 20$ are used.*



(a) relative error (b) offline time (c) SVD time

FIG. 3. *Relative errors and offline time with the forward Euler time integrator, a number of time steps ($N_t = 100$).*



(a) relative error (b) offline time (c) SVD time

FIG. 4. *Relative errors and offline time with the backward Euler time integrator, a number of time steps ($N_t = 100$).*

the same problem parameter settings. For a parametric case, where the DEIM and DEIM-SNS approaches are trained with a number of sample points in a parameter space and are used to predict the solution of a new parameter point, is considered for the hyperbolic problems in sections 6.2 and 6.3. Figures 2, 3, and 4 are generated by setting $T = 0.01$ s and $\Delta t = 1.0 \times 10^{-4}$ s, leading to $N_t = 100$. The relative error and the offline time are plotted as the dimension of nonlinear term basis n_f increases. For DEIM-SNS, $\boldsymbol{\Phi}_f = \boldsymbol{M}\boldsymbol{\Phi}$ is used if $n_f = n_s$, while $\boldsymbol{\Phi}_f = \boldsymbol{M}\boldsymbol{\Phi}_e$ is used for $n_f > n_s$, where $\boldsymbol{M}$ is a volume matrix.

In Figure 3(a), the relative errors of both the DEIM and DEIM-SNS approaches are plotted as the dimension of nonlinear term basis increases from 20 to 100 by 5 with the forward Euler time integrator. The figure shows that DEIM-SNS is comparable to DEIM in terms of accuracy. Note that the order of relative errors with both DEIM and DEIM-SNS is 10^{-9} for the whole range of the nonlinear term basis dimension considered here. This implies that setting the dimension of the nonlinear term basis n_f as small as the dimension of the solution basis n_s is sufficient to achieve good accuracy.

Figure 3(b) shows the offline times required by DEIM and DEIM-SNS. The offline time of the DEIM approach includes the time of two SVDs for the solution and nonlinear term bases construction and the time of constructing sample indices. The offline time of DEIM-SNS includes the time of *one* SVD for the solution basis and nonlinear term basis and the time of constructing sample indices. Because DEIM-SNS requires only one SVD, the offline time of the DEIM-SNS approach is less than that of the DEIM approach. This fact is shown more clearly in Figure 3(c), which shows the SVD times only, where DEIM-SNS achieves a speed-up of around two with respect to DEIM. This excludes the time of constructing sample indices from Figure 3(b).

In Figure 4(a), the relative errors of both the DEIM and DEIM-SNS approaches are plotted as the dimension of the nonlinear term basis increases from 20 to 100 by 5 with the backward Euler time integrator. The figure shows that the DEIM-SNS approach is comparable to the DEIM approach in terms of accuracy. The order of relative errors of both DEIM and DEIM-SNS is 10^{-2} for the whole range of the nonlinear term basis dimensions considered here.

Figure 4(b) shows the offline time required by DEIM and DEIM-SNS. The offline time of the DEIM approach includes the time of two SVDs for the solution and nonlinear term bases construction and the time of constructing sample indices. The offline time of the DEIM-SNS approach includes the time of *one* SVD for the solution and nonlinear term bases construction and the time of constructing sample indices. Because DEIM-SNS requires only one SVD, the offline time of DEIM-SNS is less than that of DEIM. This fact is shown more clearly in Figure 4(c), which shows the SVD times only. This excludes the time of constructing sample indices from Figure 4(b).

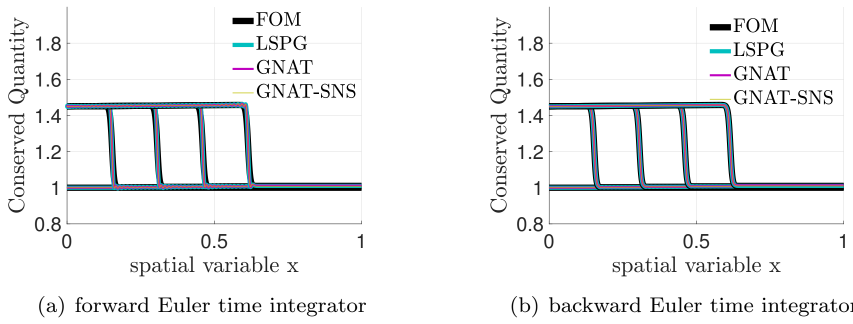
6.2. Parameterized 1D Burgers' equation. We first consider the parameterized inviscid Burgers' equation described in [31], which corresponds to the following initial boundary value problem for $x \in [0,1]$ m and $t \in [0,T]$ with $T = 0.5$ s:

$$\begin{aligned} \frac{\partial w(x,t;\mu)}{\partial t} + \frac{\partial f(w(x,t;\mu))}{\partial x} &= 0.02e^{\mu_2 x} \quad \forall x \in [0,1], \quad \forall t \in [0,T], \\ w(0,t;\boldsymbol{\mu}) &= \mu_1 \quad \forall t \in [0,T], \\ w(x,0) &= 1 \quad \forall x \in [0,1], \end{aligned} \tag{6.2}$$

where $w : [0,1] \times [0,T] \times \mathscr{D} \to \mathbb{R}$ is a conserved quantity and the $n_\mu = 2$ parameters comprise the left boundary value and source-term coefficient with $\boldsymbol{\mu} \equiv (\mu_1, \mu_2) \in \mathscr{D} = [1.2, 1.5] \times [0.02, 0.025]$.

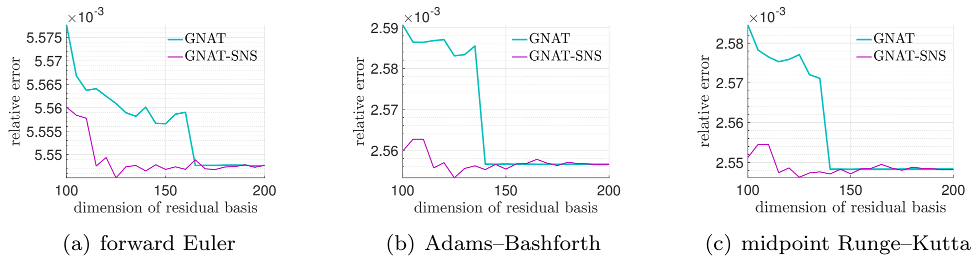
After applying Godunov's scheme for spatial discretization, equations (6.2) lead to a parameterized initial-value ODE problem consistent with (2.1) with $\boldsymbol{M}$ being an identity matrix. For this problem, all ROMs employ a training set $\mathscr{D}_{\text{train}} = \{1.2, 1.3, 1.4, 1.5\} \times \{0.02, 0.025\}$ such that $n_{\text{train}} = 8$ at which the FOM is solved. Then, the target parameter, $\boldsymbol{\mu} = (1.45, 0.0201)$, is pursued.

6.2.1. GNAT-SNS versus GNAT. For the spatial ROMs the domain is discretized with 1000 control volumes, for $N_s = 1000$ spatial degrees of freedom. We employ $N_t = 2000$, leading to a uniform time step of $\Delta t = 2.5 \times 10^{-4}$. The solution basis dimension of $n_s = 100$ is used. The relative error and the offline time are plotted



(a) forward Euler time integrator (b) backward Euler time integrator

FIG. 5. *Solution snapshots at* $t \in \{0, 0.125, 0.25, 0.375, 0.5\}$*,* $n_r = 100$*,* $n_z = 300$*.*



(a) forward Euler (b) Adams–Bashforth (c) midpoint Runge–Kutta

FIG. 6. *Relative errors with respect to FOM solution for explicit time integrators.*

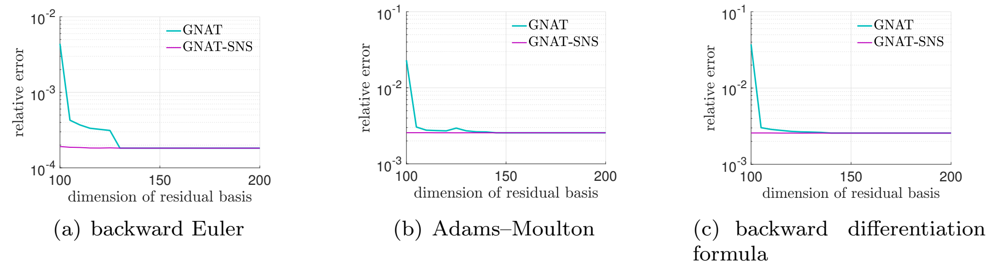
as the dimension of nonlinear residual term basis n_r increases. For the GNAT-SNS method, $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}$ is set if $n_r = n_s$, while $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}_e$ is set for $n_r > n_s$.

Figure 5 compares the solution snapshots of several methods: the LSPG, GNAT, and GNAT-SNS methods with the FOM solution snapshots. Figure 5(a) is generated with the forward Euler time integrator, while Figure 5(b) is generated with the backward Euler time integrator. All the methods are able to generate almost the same solutions as the FOM solutions.

In Figure 6, the relative errors of both the GNAT and GNAT-SNS methods are plotted as the dimension of the residual basis (n_r) increases from 100 to 200 by 5 with a fixed number of sample indices, $n_z = 300$, with the three different explicit time integrators: the forward Euler, the Adams–Bashforth, and the midpoint Runge–Kutta time integrators. The figures show that the GNAT-SNS method is comparable to the GNAT method in terms of accuracy; the order of relative errors is 10^{-3} for the whole range of the residual basis dimensions considered here.

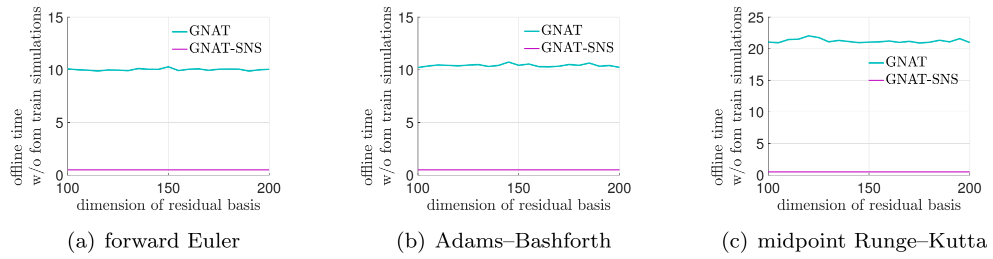
In Figure 7, the relative errors of both the GNAT and GNAT-SNS methods are plotted as the dimension of residual basis (n_r) increases from 100 to 200 by 5 with a fixed number of sample indices, $n_z = 300$, with the three different implicit time integrators: the backward Euler, the Adams–Moulton, and the BDF time integrators. The figures show that the GNAT-SNS method produces results with better accuracy than the GNAT method when the residual basis dimensions are between 100 and 120 (i.e., $n_r \approx n_s$). In fact, the GNAT-SNS achieves an accuracy as good as the LSPG can achieve. We are not sure why this is so at this time, but it would be interesting to investigate the cause of it in future work.

Figure 8 shows the offline time of the GNAT and GNAT-SNS methods for the three different explicit time integrators. The offline time of the GNAT method includes the time of two SVDs for the solution and residual bases construction, the time



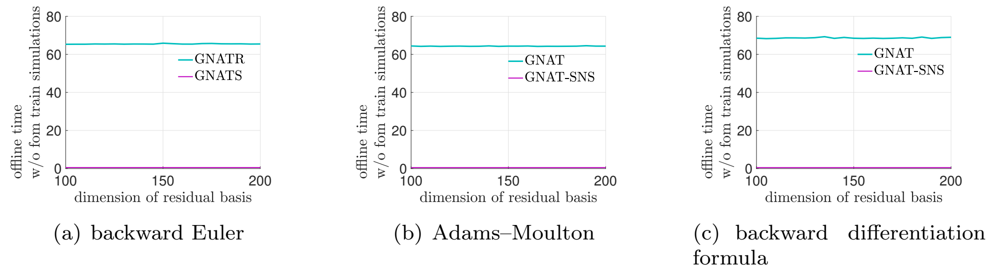
(a) backward Euler (b) Adams–Moulton (c) backward differentiation formula

FIG. 7. *Relative errors with respect to FOM solution for implicit time integrators.*



(a) forward Euler (b) Adams–Bashforth (c) midpoint Runge–Kutta

FIG. 8. *Offline time for explicit time integrators.*



(a) backward Euler (b) Adams–Moulton (c) backward differentiation formula

FIG. 9. *Offline time for implicit time integrators.*

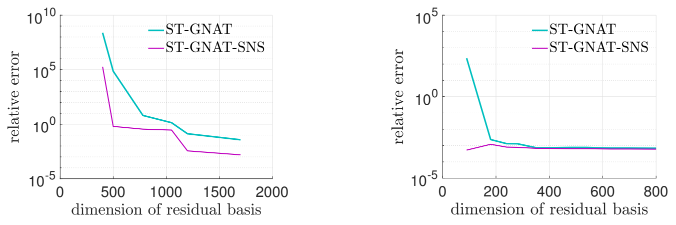
of the training LSPG simulations for collecting residual snapshots, and the time of constructing sample indices. The offline time of the GNAT-SNS method includes the time of *one* SVD for the solution and residual bases and the time of constructing sample indices. Because the GNAT-SNS method does not need to solve the training LSPG simulations for collecting residual snapshots and it requires only one SVD, the offline time of the GNAT-SNS is less than that of the GNAT method. Here, we obtain the offline computational time speed-up, a factor of 20 to 40 with the SNS methods.

Figure 9 shows the offline time of the GNAT and GNAT-SNS methods for the three different implicit time integrators. Because the GNAT-SNS method does not need to solve the training LSPG simulations for collecting residual snapshots and it requires only one SVD, the offline time of the GNAT-SNS method is less than that of the GNAT method. Here, we obtain the offline computational time speed-up, an order of 100 with the SNS methods.

6.2.2. ST-GNAT-SNS versus ST-GNAT. For the space-time ROMs, the domain is discretized with 100 control volumes, for $N_s = 100$ spatial degrees of freedom. The time discretization used in the space-time ROMs is the backward Euler time integrator. We employ $N_t = 2\,000$, leading to a uniform time step of $\Delta t = 2.5 \times 10^{-4}$ s. The description for the various ways of collecting the space-time residual snapshots are shown in sections 3.3 and 4.3. We use *ST-LSPG ROM training iterations* to collect the ST-GNAT residual basis snapshots. The relative error and the offline time are plotted as the dimension of nonlinear residual term basis n_r increases. For the ST-GNAT-SNS method, we set $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}$ if $n_{st} = n_r$, while $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}_e$ is used if $n_{st} < n_r$.

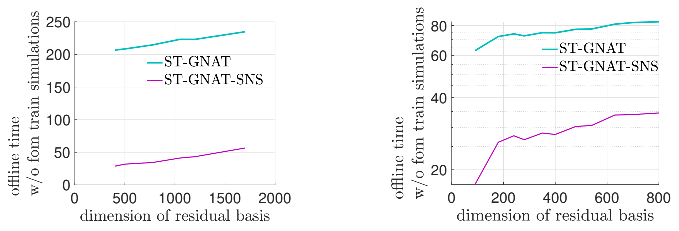
Figure 10 shows the relative error for the two different space-time basis generation methods, namely two different tensor decompositions: ST-HOSVD and LL1. For each case, the dimension of residual basis n_r varies. For the ST-HOSVD, we use $n_{st} = 400$ and $n_r \in \{400,\ 500,\ 780,\ 1050,\ 1200,\ 1700\}$. For the LL1, we use $n_{st} = 90$ and $n_r \in \{90,\ 180,\ 240,\ 280,\ 350,\ 400,\ 480,\ 540,\ 630,\ 700,\ 800\}$. For all the cases, the ST-GNAT-SNS method is comparable to the ST-GNAT method.

Figure 11 shows the offline time of the ST-GNAT and ST-GNAT-SNS methods. The offline time of the ST-GNAT method includes the time of two tensor decompositions (e.g., ST-HOSVD or LL1) for the solution and residual bases construction, the time of the training ST-LSPG simulations for collecting residual snapshots, and the time of constructing sample indices. The offline time of the ST-GNAT-SNS method includes the time of *one* tensor decomposition for the solution and residual bases and the time of constructing sample indices. Because the ST-GNAT-SNS does not need to solve the training ST-LSPG simulations for collecting residual snapshots and it only requires one tensor decomposition, the offline time of the ST-GNAT-SNS method is less than that of the ST-GNAT method. Here, we obtain the offline computational time speed-up, a factor of three to seven with the SNS methods.



(a) ST-HOSVD (b) LL1

FIG. 10. *Relative errors of the space-time ROMs for solving Burgers' equation.*



(a) ST-HOSVD (b) LL1

FIG. 11. *Offline time of the space-time ROMs for solving Burgers' equation.*

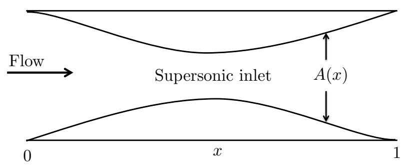


FIG. 12. *Quasi-1D Euler. Schematic figures of converging-diverging nozzle.*

6.3. Parameterized quasi-1D Euler equation. We now consider a parameterized quasi-1D Euler equation associated with modeling inviscid compressible flow in a 1D converging-diverging nozzle with a continuously varying cross-sectional area [29, Chapter 13]; Figure 12 depicts the problem geometry.

The governing system of nonlinear partial differential equations is

$$\frac{\partial \boldsymbol{w}}{\partial t} + \frac{1}{A}\frac{\partial(\boldsymbol{f}(\boldsymbol{w})A)}{\partial x} = \boldsymbol{q}(\boldsymbol{w}) \quad \forall x \in [0,1] \text{ m}, \quad \forall t \in [0,T],$$

where $T = 0.6$ s and

$$\boldsymbol{w} = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix}, \quad \boldsymbol{f}(\boldsymbol{w}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (e+p)u \end{bmatrix}, \quad \boldsymbol{q}(\boldsymbol{w}) = \begin{bmatrix} 0 \\ \frac{p}{A}\frac{\partial A}{\partial x} \\ 0 \end{bmatrix},$$

$$p = (\gamma - 1)\rho\epsilon, \quad \epsilon = \frac{e}{\rho} - \frac{u^2}{2}, \quad A = A(x).$$

Here, ρ denotes density, u denotes velocity, p denotes pressure, ϵ denotes potential energy per unit mass, e denotes total energy density, γ denotes the specific heat ratio, and A denotes the converging-diverging nozzle cross-sectional area. We employ a specific heat ratio of $\gamma = 1.3$, a specific gas constant of $R = 355.4$ m^2/s^2/K, a total temperature of $T_t = 300$ K, and a total pressure of $p_t = 10^6$ N/m^2. The cross-sectional area $A(x)$ is determined by a cubic spline interpolation over the points $(x, A(x)) \in \{(0, 0.2), (0.25, 0.173), (0.5, 0.17), (0.75, 0.173), (1, 0.2)\}$, which results in

$$A(x) = \begin{cases} -0.288x^3 + 0.4080x^2 - 0.1920x + 0.2, & x \in [0, 0.25) \text{ m}, \\ -0.288(x-0.25)^3 + 0.1920(x-0.25)^2 - 0.0420(x-0.25) + 0.1730, & x \in [0.25, 0.5) \text{ m}, \\ 0.288(x-0.5)^3 - 0.0240(x-0.5)^2 + 0.17, & x \in [0.5, 0.75) \text{ m}, \\ 0.288(x-0.75)^3 + 0.1920(x-0.75)^2 + 0.0420(x-0.75) + 0.1730, & x \in [0.75, 1] \text{ m}. \end{cases}$$

A perfect gas is assumed that obeys the ideal gas law (i.e., $p = \rho RT$). The initial flow field is created in several steps. First, the following isentropic relations are used to generate a zero pressure-gradient flow field:

$$M(x) = \frac{M_m A_m}{A(x)} \left(\frac{1 + \frac{\gamma-1}{2}M(x)^2}{1 + \frac{\gamma-1}{2}M_m^2} \right)^{\frac{\gamma+1}{2(\gamma-1)}}, \quad p(x) = p_t \left(1 + \frac{\gamma-1}{2}M(x)^2\right)^{\frac{-\gamma}{\gamma-1}}$$

$$T(x) = T_t \left(1 + \frac{\gamma-1}{2}M(x)^2\right)^{-1}, \quad \rho(x) = \frac{p(x)}{RT(x)}, \quad c(x) = \sqrt{\gamma\frac{p(x)}{\rho(x)}}, \quad u(x) = M(x)c(x),$$

where a subscript m indicates the flow quantity at $x = 0.5$ m, and M denotes the Mach number. Then, a shock is placed at $x = 0.85$ m of the flow field. The jump relations for a stationary shock and the perfect gas equation of state are used to derive the velocity across the shock u_2, which satisfies the quadratic equation

$$\left(\frac{1}{2} - \frac{\gamma}{\gamma-1}\right)u_2^2 + \frac{\gamma}{\gamma-1}\frac{n}{m}u_2 - h = 0. \tag{6.3}$$

Here, $m := \rho_2 u_2 = \rho_1 u_1$, $n := \rho_2 u_2^2 + p_2 = \rho_1 u_1^2 + p_1$, $h := (e_2 + p_2)/\rho_2 = (e_1 + p_1)/\rho_1$, and subscripts 1 and 2 denote a flow quantity to the left and to the right of the shock, respectively. A solution u_2 of (6.3) is chosen to result in a discontinuity (i.e., shock). Finally, the exit pressure is increased to a factor P_{exit} of its original value in order to generate transient dynamics.

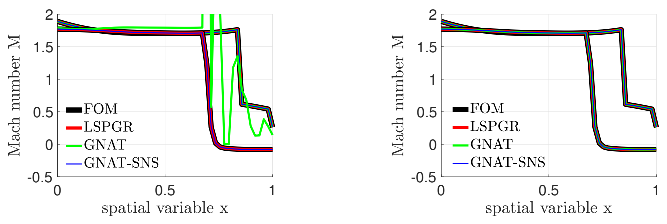
Applying a finite-volume spatial discretization with 50 equally spaced control volumes and fully implicit boundary conditions leads to a parameterized system of nonlinear ODEs consistent with (2.1) with $N_s = 150$ spatial degrees of freedom. The Roe flux difference vector splitting method is used to compute the flux at each intercell face [29, Chapter 9]. For time discretization, we apply the backward Euler scheme and a uniform time step of $\Delta t = 0.001$ s, leading to $N_t = 600$.

For this problem, we use the following two parameters: the pressure factor $\mu_1 = P_{\text{exit}}$ and the Mach number at the middle of the nozzle $\mu_2 = M_m$. All ROMs employ a training set at which the FOM is solved of $\mathscr{D}_{\text{train}} = \{1.7 + 0.01i\}_{i=0}^{3} \times \{1.7, 1.72\}$ such that $n_{\text{train}} = 8$. Then the target parameter, $\boldsymbol{\mu} = (1.7225, 1.705)$, is pursued.

6.3.1. GNAT-SNS versus GNAT. The solution basis dimension of $n_s = 30$ is used. The relative error and the offline time are plotted as the dimension of nonlinear residual term basis n_r increases. For the GNAT-SNS method, $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}$ is used if $n_r = n_s$, while $\boldsymbol{\Phi}_r = \boldsymbol{M\Phi}_e$ is used if $n_r > n_s$.

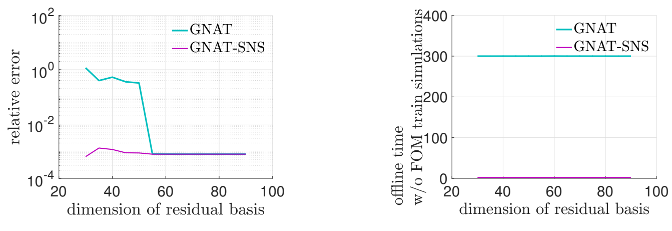
Figure 13 compares the solution snapshots of several methods: the LSPG, GNAT, and GNAT-SNS methods with the FOM solution snapshots. Figure 13(a) is generated with $n_r = 30$ and $n_z = 90$, while Figure 13(b) is generated with $n_r = 60$ and $n_z = 90$. All the methods are able to generate almost the same solutions as the FOM solutions except for the GNAT method with $n_s = n_r = 30$. Surprisingly, the GNAT-SNS method does not suffer when $n_s = n_r$ as shown in Figure 7 of the Burgers' example. Again, we do not know why the GNAT-SNS achieves accuracy as good as that of the LSPG method in this particular example.

In Figure 14(a), the relative errors of both the GNAT and GNAT-SNS methods are plotted as the dimension of residual basis increases from 30 to 90 by 5 with a fixed number of sample indices, 90. The figures show that the GNAT-SNS method is comparable to the GNAT method in terms of accuracy; the order of relative errors of the GNAT-SNS method is 10^{-3} for the whole range of the residual basis dimensions considered here. On the other hand, the relative errors of the GNAT method are bigger than those of the GNAT-SNS method when the residual basis dimensions are between 30 and 55. Figure 14(b) shows the offline time of the GNAT and GNAT-SNS methods. The offline time of the GNAT method includes the time of two SVDs for the solution and residual bases construction, the time of the training LSPG simulations



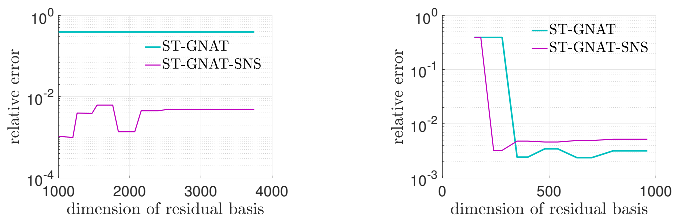
(a) $n_r = 30$ (b) $n_r = 60$

FIG. 13. *Solution snapshots at $t \in \{0, 0.6\}$, $n_z = 90$ with the backward Euler time integrator.*



(a) relative error (b) offline time

FIG. 14. *Relative errors and offline time with the backward Euler time integrator.*



(a) ST-HOSVD (b) LL1

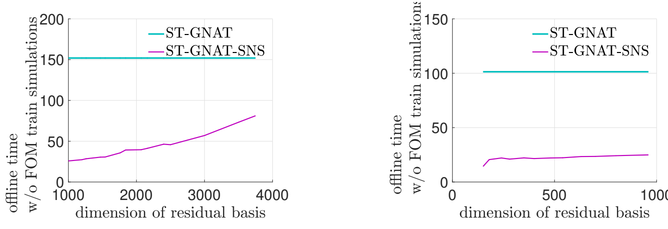
FIG. 15. *Relative errors and offline time with the backward Euler time integrator.*

for collecting residual snapshots, and the time of constructing sample indices. The offline time of the GNAT-SNS method includes the time of *one* SVD for the solution and residual bases construction and the time of constructing sample indices. Because the GNAT-SNS method does not need to solve the training LSPG simulations for collecting residual snapshots and it requires only one SVD, the offline time of the GNAT-SNS method is less than that of the GNAT method. Here, we get a factor of 130 speed-up in the offline computational time with the GNAT-SNS method.

6.3.2. ST-GNAT-SNS versus ST-GNAT. The relative error and the offline time of both the ST-GNAT and ST-GNAT-SNS methods are plotted as the dimension of nonlinear residual term basis n_r increases. For the ST-GNAT-SNS method, we set $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}$ if $n_{st} = n_r$, while $\bar{\boldsymbol{\Phi}}_r = \bar{\boldsymbol{\Phi}}_e$ is used if $n_{st} < n_r$.

In Figure 15, the relative errors of both the ST-GNAT and ST-GNAT-SNS methods are shown for two different tensor decompositions: ST-HOSVD and LL1. For the ST-HOSVD, we use $n_{st} = 1000$ and $n_r \in \{1000, 1200, 1260, 1470, 1540, 1760, 1840, 2070, 2160, 2400, 2500, 3000, 3500, 3750\}$. For the LL1 decomposition, we use $n_{st} = 150$ and $n_r \in \{150, 180, 240, 280, 350, 400, 480, 540, 630, 700, 800, 960\}$. For the ST-HOSVD, the ST-GNAT-SNS method achieves two orders of magnitude better accuracy than the ST-GNAT method. For the LL1 decomposition, the ST-GNAT-SNS method generates results as good as the ST-GNAT method.

Figure 16 shows the offline time of the ST-GNAT and ST-GNAT-SNS methods. The offline time of the ST-GNAT method includes the time of two tensor decompositions (e.g., ST-HOSVD or LL1) for the solution and residual bases construction, the time of the training ST-LSPG simulations for collecting residual snapshots, and the time of constructing sample indices. The offline time of the ST-GNAT-SNS method includes the time of *one* tensor decomposition for the solution and residual bases construction and the time of constructing sample indices. Because the ST-GNAT-



(a) ST-HOSVD (b) LL1

FIG. 16. *Offline time of the space-time ROMs for solving Euler equation.*

SNS does not need to solve the training ST-LSPG simulations for collecting residual snapshots and it only requires one tensor decomposition, the offline time of the ST-GNAT-SNS method is less than that of the ST-GNAT method. Here, we get a factor of six to seven speed-up in the offline computational time with the SNS methods.

7. Conclusion.

We have introduced a new way of constructing nonlinear term basis using solution snapshots to construct a projection-based ROM for solving a nonlinear dynamical system of equations, which is characterized by an ODE. Our proposed method, the SNS method, is theoretically justified by the conforming subspace condition and the subspace inclusion relation.

The two main advantages of the SNS method over the traditional hyper-reduction methods considered here are (1) the avoidance of collecting nonlinear term snapshots and (2) the avoidance of the second data compression process. These advantages result in the offline computational time reduction, which is demonstrated in numerical experiments. The benefits of the SNS method are more vivid when it is compared with GNAT and ST-GNAT than DEIM because the GNAT and ST-GNAT methods require collecting the nonlinear residual snapshots from the corrsponding LSPG and ST-LSPG simulations. These benefits are also demonstrated in numerical experiments, where parametric GNAT and ST-GNAT are compared with the SNS methods. There, we have shown that a considerable speed-up in the offline computational time is achieved by the SNS methods. Also, the error analysis contributes to the theoretical insight about the effects of the volume matrix $\boldsymbol{M}$ on the oblique projection of the SNS method, revealing that the conditioner number of $\boldsymbol{M}$ can affect the error due to the oblique projection. This issue can be addressed by a preorthogonalization process.

In numerical experiments, we observe that the SNS methods produce a more accurate solution than the DEIM, GNAT, and ST-GNAT methods with a smaller number of nonlinear term basis vectors, especially for the GNAT-SNS method. The reason for this attractive feature of the GNAT-SNS method will be investigated in future work.

Appendix A. Subspace inclusion relation.

The forward and backward Euler time integrators and their corresponding subspace inclusion relations are shown in section 2. Here, we show several other time integrators and corresponding subspace inclusion relation between the subspace spanned by the nonlinear term snapshots and the subspace spanned by the solution snapshots.

A.1. The Adams–Bashforth methods.

The second order Adams–Bashforth method numerically solves (2.1) by solving the following nonlinear system of equations for $\boldsymbol{x}_n$ at the nth time step:

$$\boldsymbol{M}\boldsymbol{x}_n - \boldsymbol{M}\boldsymbol{x}_{n-1} = \Delta t\left(\frac{3}{2}\boldsymbol{f}_{n-1} - \frac{1}{2}\boldsymbol{f}_{n-2}\right). \tag{A.1}$$

Equation (A.1) implies the following subspace inclusions:

$$\text{span}\{\boldsymbol{f}_{n-2}, \boldsymbol{f}_{n-1}\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_n\}.$$

By induction, we conclude that

$$\text{span}\{\boldsymbol{f}_0, \dots, \boldsymbol{f}_{N_t-1}\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_0, \dots, \boldsymbol{M}\boldsymbol{x}_{N_t}\}, \tag{A.2}$$

which shows that the span of nonlinear term snapshots is a subspace of the span of $\boldsymbol{M}$-scaled solution snapshots. The residual function of the second Adams–Bashforth method is defined as

$$\boldsymbol{r}^n_{\text{AB}}(\boldsymbol{x}_n; \boldsymbol{x}_{n-1}, \boldsymbol{\mu}) := \boldsymbol{M}(\boldsymbol{x}_n - \boldsymbol{x}_{n-1}) - \Delta t\left(\frac{3}{2}\boldsymbol{f}_{n-1} - \frac{1}{2}\boldsymbol{f}_{n-2}\right).$$

A.2. The Adams–Moulton methods. The second order Adams–Moulton method numerically solves (2.1) by solving the following nonlinear system of equations for $\boldsymbol{x}_n$ at the nth time step:

$$\boldsymbol{M}\boldsymbol{x}_n - \boldsymbol{M}\boldsymbol{x}_{n-1} = \frac{1}{2}\Delta t(\boldsymbol{f}_n + \boldsymbol{f}_{n-1}). \tag{A.3}$$

Equation (A.3) implies the following subspace inclusions:

$$\text{span}\{\boldsymbol{f}_{n-1}, \boldsymbol{f}_n\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_n\}.$$

By induction, we conclude that

$$\text{span}\{\boldsymbol{f}_0, \dots, \boldsymbol{f}_{N_t}\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_0, \dots, \boldsymbol{M}\boldsymbol{x}_{N_t}\},$$

which shows that the span of nonlinear term snapshots is a subspace of the span of $\boldsymbol{M}$-scaled solution snapshots. The residual function of the second Adams–Moulton method is defined as

$$\boldsymbol{r}^n_{\text{AM}}(\boldsymbol{x}_n; \boldsymbol{x}_{n-1}, \boldsymbol{\mu}) := \boldsymbol{M}(\boldsymbol{x}_n - \boldsymbol{x}_{n-1}) - \Delta t\frac{1}{2}(\boldsymbol{f}_n + \boldsymbol{f}_{n-1}). \tag{A.4}$$

A.3. The backward differentiation formulas. The two-step BDF numerically solves (2.1) by solving the following nonlinear system of equations for $\boldsymbol{x}_n$ at the nth time step:

$$\boldsymbol{M}\boldsymbol{x}_n - \frac{4}{3}\boldsymbol{M}\boldsymbol{x}_{n-1} + \frac{1}{3}\boldsymbol{M}\boldsymbol{x}_{n-2} = \frac{2}{3}\Delta t\boldsymbol{f}_n. \tag{A.5}$$

Equation (A.5) implies the following subspace inclusions:

$$\text{span}\{\boldsymbol{f}_n\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_{n-2}, \boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_n\}.$$

By induction, we conclude that

$$\text{span}\{\boldsymbol{f}_1, \dots, \boldsymbol{f}_{N_t-1}\} \subseteq \text{span}\{\boldsymbol{M}\boldsymbol{x}_0, \dots, \boldsymbol{M}\boldsymbol{x}_{N_t}\},$$

which shows that the span of nonlinear term snapshots is a subspace of the span of $\boldsymbol{M}$-scaled solution snapshots. The residual function of the two-step BDF method is defined as

$$\boldsymbol{r}^n_{\text{BDF}}(\boldsymbol{x}_n; \boldsymbol{x}_{n-1}, \boldsymbol{x}_{n-2}, \boldsymbol{\mu}) := \boldsymbol{M}\left(\boldsymbol{x}_n - \frac{4}{3}\boldsymbol{x}_{n-1} + \frac{1}{3}\boldsymbol{x}_{n-2}\right) - \frac{2}{3}\Delta t\boldsymbol{f}_n. \tag{A.6}$$

A.4. The midpoint Runge–Kutta method. The midpoint method, a two-stage Runge–Kutta method, takes the following two stages to advance at the nth time step of (2.1):

$$\begin{aligned}\boldsymbol{M}\boldsymbol{x}_{n-\frac{1}{2}} &= \boldsymbol{M}\boldsymbol{x}_{n-1} + \frac{\Delta t}{2}\boldsymbol{f}_{n-1},\\ \boldsymbol{M}\boldsymbol{x}_{n} &= \boldsymbol{M}\boldsymbol{x}_{n-1} + \Delta t\boldsymbol{f}_{n-\frac{1}{2}}.\end{aligned}$$

These lead to the following two subspace inclusion relations:

$$\begin{aligned}\operatorname{span}\{\boldsymbol{f}_{n-1}\} &\subseteq \operatorname{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_{n-\frac{1}{2}}\},\\ \operatorname{span}\{\boldsymbol{f}_{n-\frac{1}{2}}\} &\subseteq \operatorname{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_{n}\}.\end{aligned}$$

These in turn lead to the following subspace inclusion relation:

$$\operatorname{span}\{\boldsymbol{f}_{n-1}, \boldsymbol{f}_{n-\frac{1}{2}}\} \subseteq \operatorname{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_{n-\frac{1}{2}}, \boldsymbol{M}\boldsymbol{x}_{n}\}.$$

By induction, we conclude that

$$\begin{aligned}&\operatorname{span}\{\boldsymbol{f}_{0}, \boldsymbol{f}_{\frac{1}{2}}, \ldots, \boldsymbol{f}_{N_t-1}, \boldsymbol{f}_{N_t-\frac{1}{2}}\}\\ &\quad\subseteq \operatorname{span}\{\boldsymbol{M}\boldsymbol{x}_{n-1}, \boldsymbol{M}\boldsymbol{x}_{n-\frac{1}{2}}, \boldsymbol{M}\boldsymbol{x}_{n}, \ldots, \boldsymbol{M}\boldsymbol{x}_{N_t-1}, \boldsymbol{M}\boldsymbol{x}_{N_t-\frac{1}{2}}, \boldsymbol{M}\boldsymbol{x}_{N_t}\}.\end{aligned} \tag{A.7}$$

Acknowledgment. The authors acknowledge the helpful comments provided by the anonymous reviewers.

REFERENCES

[1] *BLAST: High-order Finite Element Hydrodynamics*, https://computation.llnl.gov/projects/blast/icf-like-implosion.

[2] *MFEM: Modular Finite Element Methods Library*, https://doi.org/10.11578/dc.20171025.1248.

[3] S. S. An, T. Kim, and D. L. James, *Optimizing cubature for efficient integration of subspace deformations*, ACM Trans. Graphics, 27 (2008), 165.

[4] R. W. Anderson, V. A. Dobrev, T. V. Kolev, R. N. Rieben, and V. Z. Tomov, *High-order multi-material ale hydrodynamics*, SIAM J. Sci. Comput., 40 (2018), pp. B32–B58.

[5] P. Astrid, S. Weiland, K. Willcox, and T. Backx, *Missing point estimation in models described by proper orthogonal decomposition*, IEEE Trans. Automat. Control, 53 (2008), pp. 2237–2251.

[6] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, *An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations*, C. R. Math., 339 (2004), pp. 667–672.

[7] G. Berkooz, P. Holmes, and J. L. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annu. Rev. Fluid Mech., 25 (1993), pp. 539–575.

[8] A. Buttari, J. Langou, J. Kurzak, and J. Dongarra, *Parallel tiled QR factorization for multicore architectures*, Concurrency Comput. Practice Experience, 20 (2008), pp. 1573–1590.

[9] K. Carlberg, M. Barone, and H. Antil, *Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction*, J. Comput. Phys., 330 (2017), pp. 693–734.

[10] K. Carlberg, C. Bou-Mosleh, and C. Farhat, *Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations*, Internat. J. Numer. Methods Engrg., 86 (2011), pp. 155–181.

[11] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, *The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows*, J. Comput. Phys., 242 (2013), pp. 623–647.

[12] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764.

[13] Y. Choi and K. Carlberg, *Space-time least-squares Petrov–Galerkin projection for nonlinear model reduction*, SIAM J. Sci. Comput., 41 (2019), pp. A26–A58.

[14] L. De Lathauwer, *Blind separation of exponential polynomials and the decomposition of a tensor in rank-($l\_r, l\_r, 1$) terms*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1451–1474.

[15] Z. Drmac and S. Gugercin, *A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions*, SIAM J. Sci. Comput., 38 (2016), pp. A631–A648.

[16] Z. Drmac and A. K. Saibaba, *The discrete empirical interpolation method: Canonical structure and formulation in weighted inner product spaces*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 1152–1180.

[17] E. Elmroth and F. Gustavson, *High-performance library software for QR factorization*, in International Workshop on Applied Parallel Computing, Lecture Notes in Comput. Sci. 1947, Springer, New York, 2000, pp. 53–63.

[18] R. Everson and L. Sirovich, *Karhunen–Loeve procedure for gappy data*, JOSA A, 12 (1995), pp. 1657–1664.

[19] C. Farhat, P. Avery, T. Chapman, and J. Cortial, *Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency*, Internat. J. Numer. Methods Engrg., 98 (2014), pp. 625–662.

[20] C. Farhat, T. Chapman, and P. Avery, *Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models*, Internat. J. Numer. Methods Engrg., 102 (2015), pp. 1077–1110.

[21] M. A. Grepl, Y. Maday, N. C. Nguyen, and A. T. Patera, *Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations*, ESAIM Math. Model. Numer. Anal., 41 (2007), pp. 575–605.

[22] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.

[23] J. A. Hernandez, M. A. Caicedo, and A. Ferrer, *Dimensional hyper-reduction of nonlinear finite element models via empirical cubature*, Comput. Methods Appl. Mech. Engrg., 313 (2017), pp. 687–722.

[24] M. Hinze and S. Volkwein, *Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: Error estimates and suboptimal control*, in Dimension Reduction of Large-Scale Systems, Lect. Notes Comput. Sci. Eng. 45, Springer, New York, 2005, pp. 261–306.

[25] H. Hotelling, *Analysis of a complex of statistical variables into principal components*, J. Educational Psychology, 24 (1933), pp. 417–441.

[26] S. A. Khairallah and A. Anderson, *Mesoscopic simulation model of selective laser melting of stainless steel powder*, J. Materials Process. Technol., 214 (2014), pp. 2627–2636.

[27] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics*, SIAM J. Numer. Anal., 40 (2002), pp. 492–515.

[28] M. Loeve, *Probability Theory*, Van Nostrand, New York, 1955.

[29] R. MacCormack, *Numerical Computation of Compressible Viscous Flow*, Tech. report, Lecture notes for AA214b and AA214c, Stanford University, Stanford, CA, 2007.

[30] N. C. Nguyen and J. Peraire, *An efficient reduced-order modeling approach for non-linear parametrized partial differential equations*, Internat. J. Numer. Methods Engrg., 76 (2008), pp. 27–55.

[31] M. J. Rewieński, *A Trajectory Piecewise-Linear Approach to Model Order Reduction of Nonlinear Dynamical Systems*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2003.

[32] D. Ryckelynck, *A priori hyperreduction method: An adaptive approach*, J. Comput. Phys., 202 (2005), pp. 346–366.

[33] L. Sorber, M. Van Barel, and L. De Lathauwer, *Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-($l\_r, l\_r, 1$) terms, and a new generalization*, SIAM J. Optim., 23 (2013), pp. 695–720.

[34] M. J. Zahr, K. Carlberg, D. Amsallem, and C. Farhat, *Comparison of Model Reduction Techniques on High-Fidelity Linear and Nonlinear Electrical, Mechanical, and Biological Systems*, University of California, Berkeley, 2010.