DATA-DRIVEN TIME PARALLELISM VIA FORECASTING*

KEVIN CARLBERG†, LUKAS BRENCHER‡, BERNARD HAASDONK‡, AND ANDREA BARTH‡

Abstract. This work proposes a data-driven method for enabling the efficient, stable time-parallel numerical solution of systems of ordinary differential equations (ODEs). The method assumes that low-dimensional bases that accurately capture the *time evolution* of the dynamical-system state are available. The method adopts the parareal framework for time parallelism, which is defined by an initialization method, a coarse propagator that advances solutions on a coarse time grid, and a fine propagator that operates on an underlying fine time grid. Rather than employing usual approaches for initialization and coarse propagation, we propose novel *data-driven* techniques that leverage the available time-evolution bases. The coarse propagator is defined by a forecast (proposed in [K. Carlberg, J. Ray, and B. van Bloemen Waanders, *Comput. Methods Appl. Mech. Engrg.*, 289 (2015), pp. 79–103]) applied locally within each coarse time interval, which comprises the following steps: (1) apply the fine propagator for a small number of time steps, (2) approximate the state over the entire coarse time interval using gappy proper orthogonal decomposition (POD) with the local time-evolution bases, and (3) select the approximation at the end of the time interval as the propagated state. We also propose both local-forecast and global-forecast initialization techniques. The method is particularly well suited for POD-based reduced-order models (ROMs), as the time-evolution bases can be extracted from readily available data, i.e., the right singular vectors arising during POD computation. In addition to performing analyses related to the method's accuracy, speedup, stability, and convergence, we also numerically demonstrate the method's performance. Here, numerical experiments on ROMs for a nonlinear convection–reaction problem demonstrate the method's ability to realize *near-ideal* speedups.

Key words. time-parallel, parareal, forecasting, gappy proper orthogonal decomposition, data-driven approximation, model reduction

AMS subject classifications. 65B99, 65D30, 65L05A, 65L06, 65L20, 65M12, 65M55, 65Y05

DOI. 10.1137/18M1174362

1. Introduction.

Two emerging trends introduce both challenges and opportunities in computational science: (1) in future extreme-scale architectures, improved wall-time performance must be achieved primarily by exposing additional concurrency, and (2) the rapid increase in the volume of available physical and computational data presents an opportunity to extract useful insights from these data. These trends expose a unique opportunity: integrating extreme-scale simulation with data analytics can positively impact both data-intensive science and extreme-scale computing [37].

This is what this work strives to accomplish: we aim to leverage available *computational data* to improve *concurrency and parallel performance* when simulating

[*] Submitted to the journal's Computational Methods in Science and Engineering section March 13, 2018; accepted for publication (in revised form) March 22, 2019; published electronically May 23, 2019.

http://www.siam.org/journals/sisc/41-3/M117436.html

Funding: The work of the second, third, and fourth authors was supported by the German Research Foundation (DFG) as part of the Cluster of Excellence in Simulation Technology (EXC 310/2) at the University of Stuttgart. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

[†] Sandia National Laboratories, Livermore, CA 94550 (ktcarlb@sandia.gov).

[‡] University of Stuttgart, D-70569 Stuttgart, Germany (lukas.brencher@web.de, haasdonk@mathematik.uni-stuttgart.de, andrea.barth@mathematik.uni-stuttgart.de).

parameterized dynamical systems. More precisely, this work considers numerically solving large-scale systems of parameterized ordinary differential equations (ODEs), which arise in applications ranging from computational fluid dynamics to molecular dynamics. The above trends have particular implications in this context.

1.1. Numerically solving ODEs: Exposing concurrency. First, the sequential nature of numerically solving ODEs (i.e., numerical time integration) typically poses the dominant computational bottleneck, both in strong and weak scaling. In the context of numerically solving ODEs, *strong scaling* is typically achieved through parallelizing "across the system" by increasing the number of processors over which the problem is decomposed *spatially*; this usually associates with parallelizing the linear-system solve occurring within each time step for implicit time integration.[1] However, spatial parallelism saturates: there exists a number of cores beyond which the speedup decreases due to the dominance of latency and bandwidth costs over savings in sequential computation. This maximum number of (useful) cores is proportional to the problem size and defines the minimum wall-time achievable by spatial parallelism alone, even in the presence of unlimited computational resources. This wall-time floor can preclude computational models from being employed in time-critical applications (e.g., model predictive control). In the context of numerically solving ODEs, *weak scaling* is typically achieved by refining the spatial discretization (when the ODE associates with a spatially discretized partial differential equation) as the number of cores used for spatial parallelism increases. However, to prevent time-discretization errors from dominating spatial-discretization errors (and to preserve stability in the case of explicit time integration), spatial refinement typically requires attendant temporal refinement, which leads to an increase in the total number of time steps. This implies poor weak scaling in this case, as the wall-time is proportional to the problem size.

To this end, researchers have developed *time-parallel* methods that "widen the computational front" by exposing parallelism in the temporal dimension. In principle, such approaches can mitigate this bottleneck, as they can decrease the minimum realizable wall-time in the strong-scaling case and can remove the dependence of the runtime on the total number of time steps in the weak-scaling case. Broadly, these techniques can be categorized as iterative methods based on multiple shooting [34, 6, 36, 27, 19], domain decomposition and waveform relaxation [22, 38], and multigrid [26, 28, 31, 18], as well as direct methods [32, 1, 40, 29].

Perhaps the most well-studied and widely adopted time-parallel method is the *parareal* technique [27], which can be interpreted [24, 18] as a deferred/residual-correction scheme, a multiple-shooting method with a finite-difference Jacobian approximation, or as a two-level multigrid method. The parareal method alternates between (1) time integration using a *fine propagator* executed in *parallel* on a non-overlapping decomposition of the time domain, and (2) time integration using a *coarse propagator* executed in *serial* on a coarse time discretization defined by boundaries of the temporal subdomains. The update formula associated with sequential coarse time integration aims to set the discontinuities in the fine solution (occurring at temporal-subdomain boundaries) to zero.

The parareal method converges to the solution computed by the fine propagator; thus the fine propagator is usually chosen to be a typical single-step time integrator (e.g., Runge–Kutta scheme). On the other hand, the coarse propagator can be chosen

[1] If the system of ODEs is nonlinear and Newton's method is applied to solve each system of algebraic equations, the linear-system solve occurs at each Newton iteration within each time step.

somewhat freely; it determines the parallel performance of the parareal method. Desired properties in the coarse propagator include *accuracy* (i.e., it should incur small error with respect to the fine propagator to ensure fast convergence), *low cost* (i.e., its computational complexity should not scale with the underlying fine time discretization), and *stability* (i.e., it should ensure a stable parareal recurrence). A primary research area within time-parallel methods aims to develop coarse propagators that satisfy these properties.

The most commonly used coarse propagator is simply a typical time integrator (which can have a lower-order accuracy than the fine propagator [7]) applied with coarse time steps [27] or an explicit time integrator [33] (where stability may preclude use for large coarse time steps). While straightforward to implement, the coarse time step is typically outside the asymptotic range of convergence for the chosen time integrator, which can hamper accuracy and lead to slow parareal convergence. This approach can be accelerated by additionally coarsening the spatial discretization [21, 20], employing simplified physics models [2, 30, 7], or relaxing solver tolerances [25]. Some authors have also employed reduced-order models (ROMs) constructed "on the fly" (i.e., during the parareal recurrence without any "offline" preprocessing step) [20, 14]. Instead, this work proposes employing *time-evolution data* that may be available to devise an accurate, low-cost, stable coarse propagator. In the following, we describe the source of these data.

1.2. Numerically solving ODEs: Availability of data. It is often the case that *data* are available about the dynamical system of interest. These data can arise (1) from experimental analyses, (2) from numerically solving the system of ODEs over a small time interval, or (3) from simulating the dynamical system for different parameter instances (if the dynamical system is parameterized), for example.

In this work, we assume that data are available that are related to the *time evolution* of the dynamical-system *state*. Such data may be extracted from any of the above sources. For example, these data could be provided from (1) experimental time traces of state variables at different spatial coordinates, (2) a time-domain Fourier transform of the short-time ODE numerical solution, or (3) the singular value decomposition (SVD) of the numerical spatio-temporal solution to the dynamical system at different parameter instances. While we focus primarily on the third data source (see section 5), this is not strictly required for the method to be employed.

1.3. Proposed methodology. The proposed methodology adopts the data-driven forecasting method introduced in [12] to define both the coarse propagator and the initialization used to "seed" the parareal recurrence. Given bases for the time evolution of the dynamical-system state[2] (as discussed in section 1.2), the coarse propagator is defined on a given coarse time interval by a "local forecast" as follows: (1) apply the fine propagator for a small number of time steps, (2) apply gappy proper orthogonal decomposition (POD) [17] with *local* time-evolution bases (with support over the coarse time interval) to generate an approximation of the state over the entire coarse time interval, and (3) select the value of the approximated state at the end of the coarse time interval as the propagated state. For initialization, this "local forecast" can be applied sequentially; alternatively, a "global forecast" can be applied as follows: (1) apply the fine propagator for a small number of time steps at the beginning of the time interval, (2) apply gappy POD with *global* time-evolution bases (with support over the entire time interval) to generate an approximation of the state

[2] In practice, we apply forecasting to a *restriction* of the state.

over the entire time interval, and (3) select the value of the approximated state at the temporal-subdomain boundaries as the initial solution.

The methodology is particular well suited for ROMs for two reasons. *First*, dynamical-system ROMs associate with small-scale ODEs that typically must be integrated over long time intervals. This occurs because ROMs reduce the *spatial complexity* (i.e., the cost of each linear-system solve) of large-scale dynamical systems by reducing the number of degrees of freedom (via projection) and complexity of evaluating nonlinear terms (e.g., via empirical interpolation [5, 13], empirical operator interpolation [16], or gappy POD [10]); however, ROMs generally do not significantly reduce the associated *temporal complexity* (i.e., the number of linear-system solves), which is typically proportional to the spatial dimension of the original large-scale dynamical system. Thus, ROMs suffer from early spatial-parallelism saturation associated with strong scaling as discussed in section 1.1. For example, on a compressible flow problem, the Gauss–Newton with approximated tensors (GNAT) ROM yielded a 438 factor improvement as measured in core-hours, but only a 6.86 wall-time speedup [8]; spatial parallelism was saturated with only 12 cores. The proposed methodology thus can be used in a *complementary manner* to spatial-projection-based ROMs, as it provides a mechanism to accelerate their parallel-in-time numerical integration. *Second*, ROMs already require computational data for their construction. In fact, ROMs based on POD already employ the third data set described in section 1.2; thus, the proposed coarse propagator can be computed “for free” in such contexts (see section 5.2). Here, the required time-evolution bases are easily obtained from the right singular vectors of corresponding snapshot matrices. Finally, we note that while we present the proposed coarse propagator and initialization methods in the parareal context, these techniques could also be applied to alternative time-parallel methods, e.g., the parallel implicit time-integration algorithm (PITA) [19] and multigrid reduction in time (MGRIT) [18].

1.4. Outline and notation.

To summarize, contributions of this work include

- a novel coarse propagator based on local forecasting (section 3.3),
- novel initialization methods based on both local and global forecasting (section 3.4),
- error analysis for the local-forecast coarse propagator (section 4.1) in the general (Theorem 4.1) and ideal (Theorem 4.3) cases,
- speedup analysis for all proposed methods (section 4.2) in the general (Theorems 4.4–4.5) and ideal (Theorems 4.6–4.7) cases,
- stability analysis (section 4.3) of the local-forecast coarse propagator (Lemma 4.9) and the resulting parareal recurrence (Theorem 4.11),
- convergence analysis (section 4.4) of the local-forecast coarse propagator within the parareal recurrence (Corollary 4.12),
- descriptions of how the required method ingredients can be computed via POD (section 5) for parameterized ODEs (section 5.1) and ROMs (section 5.2), and
- numerical experiments (section 6) that both highlight the practical benefits of the proposed methodology and illustrate the theoretical results.

We also provide supplementary materials, wherein section S1 contains all proofs, section S2 describes the dependence of the bound in Lemma 4.9 on the time discretization, section S3 describes theoretical performance improvements when forecasting is also employed for defining the initial guess in the Newton solver, section S4 derives convergence-analysis results, section S5 illustrates a particular case of parameterized linear ODEs in which the proposed method yields an ideal predictive coarse propagator, section S6 provides figures that lend additional insight into the numerical experiments, and section S7 provides a summary of the notation used in this manuscript; the reader may find this to be a useful reference.

In the remainder of this paper, matrices are denoted by capitalized bold letters, vectors by lowercase bold letters, and scalars by lightface letters. The columns of a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ are denoted by $\mathbf{a}_i \in \mathbb{R}^m$, $i \in \mathbb{N}(n)$, with $\mathbb{N}(n) := \{1, \ldots, n\}$ such that $\mathbf{A} := [\mathbf{a}_1 \ \cdots \ \mathbf{a}_n]$. The scalar-valued matrix elements are denoted by $a_{i,j} \in \mathbb{R}$ such that $\mathbf{a}_j := [a_{1,j} \ \cdots \ a_{m,j}]^T$, $j \in \mathbb{N}(n)$; we similarly denote the elements of a vector as $\mathbf{a} := [a_1 \ \cdots \ a_m]^T$. We also define $\mathbb{N}_0(n) := \{0, \ldots, n\}$.

2. Time parallelism and parareal. We consider initial value problems for systems of (possibly nonlinear) ODEs of the form

$$\frac{d}{dt}\mathbf{x}^\star(t) = \mathbf{g}\left(\mathbf{x}^\star; t\right), \quad \mathbf{x}^\star(0) = \mathbf{x}^0, \tag{2.1}$$

where $t \in \mathbb{T} := [0, T_{\text{final}}]$ denotes time with $T_{\text{final}} \in \mathbb{R}_+$ the final time, $\mathbf{x}^\star : \mathbb{T} \to \mathbb{R}^N$ denotes the state implicitly defined as the (exact) solution to problem (2.1), $\mathbf{x}^0 \in \mathbb{R}^N$ denotes the initial state, and $\mathbf{g} : \mathbb{R}^N \times \mathbb{T} \to \mathbb{R}^N$ with $(\boldsymbol{\xi}; t) \mapsto \mathbf{g}(\boldsymbol{\xi}; t)$ denotes the velocity, which may be linear or nonlinear in its first argument. Time-parallel methods constitute one approach to improving wall-time performance when numerically solving such problems. We now introduce the parareal method, which we consider in this work.

First, and without loss of generality, we introduce a uniform *fine* time discretization characterized by time step $h = T_{\text{final}}/m$ and time instances $t^n = nh$, $n \in \mathbb{N}_0(m)$, where $m \in \mathbb{N}$ denotes the number of total time instances beyond the initial time $t^0 = 0$ such that the final time corresponds to $t^m = T_{\text{final}}$. We denote the set of time instances associated with this discretization as $\mathcal{t} := \{t^n\}_{n=0}^m$. We introduce a "fine propagator" $\mathscr{F} : \mathbb{R}^N \times \mathcal{t} \times \mathcal{t} \to \mathbb{R}^N$ with $(\boldsymbol{\xi}; t^i, t^j) \mapsto \mathscr{F}(\boldsymbol{\xi}; t^i, t^j)$ that acts on this discretization and propagates a state $\boldsymbol{\xi} \in \mathbb{R}^N$ defined at time t^i to time t^j with $j \geq i$. This propagator satisfies

$$\mathscr{F}(\boldsymbol{\xi}; t^i, t^k) = \mathscr{F}(\cdot; t^j, t^k) \circ \mathscr{F}(\boldsymbol{\xi}; t^i, t^j), \quad 0 \leq i \leq j \leq k \leq m, \tag{2.2}$$

and typically corresponds to the application of a single-step time integrator (e.g., the Runge–Kutta scheme) to numerically solve problem (2.1). For example, the backward-Euler fine propagator $\mathscr{F}_{\text{BE}}$ implicitly satisfies

$$\mathscr{F}_{\text{BE}}(\boldsymbol{\xi}; t^i, t^{i+1}) - \boldsymbol{\xi} - h\mathbf{g}\left(\mathscr{F}_{\text{BE}}(\boldsymbol{\xi}; t^i, t^{i+1}); t^{i+1}\right) = 0, \quad i \in \mathbb{N}_0(m-1). \tag{2.3}$$

We define

$$\mathbf{x} : t \mapsto \mathscr{F}(\mathbf{x}^0; 0, t), \quad t \in \mathcal{t}, \tag{2.4}$$

as the associated numerical solution with $\mathbf{x} \in (\mathscr{H})^N$, where $\mathscr{H}$ denotes the set of functions from $\mathcal{t}$ to $\mathbb{R}$. Note that (2.2) and (2.4) imply $\mathbf{x}(t^j) = \mathscr{F}(\mathbf{x}(t^i); t^i, t^j)$, $0 \leq i \leq j \leq m$. It is this time-discrete solution $\mathbf{x}$ which we want to approximate with the time-parallel procedure.

Analogously, we consider a *coarse* time discretization characterized by (uniform) time step $H = T_{\text{final}}/M$ and time instances $T^n = nH$, $n \in \mathbb{N}_0(M)$, where $M \in \mathbb{N}$ denotes the number of coarse time instances beyond the initial time $T^0 = t^0 = 0$ such

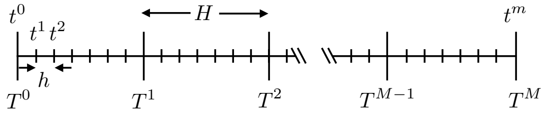


Fig. 1. *Coarse and fine time discretizations.*

that the final time corresponds to $T^M = t^m = T_{\text{final}}$ (see Figure 1). We denote the set of time instances associated with the coarse discretization as $\mathscr{T} := \{T^n\}_{n=0}^M$. Further, we assume that the coarse time step is an integral multiple of the fine time step, i.e., $H = \bar{m}h$ with $\bar{m} \in \mathbb{N}$. This implies that the coarse discretization is nested within the fine discretization $\mathscr{T} \subseteq \boldsymbol{t}$ such that $T^n = t^{\bar{m}n}$, $n \in \mathbb{N}_0(M)$, and $m = \bar{m}M$. We define the set of fine time instances associated with the nth coarse time interval as $\boldsymbol{t}^n := \boldsymbol{t} \cap \left[T^n, T^{n+1}\right] = \{t^i\}_{i=\bar{m}n}^{\bar{m}(n+1)}$, $n \in \mathbb{N}_0(M-1)$.

Denoting by $\mathbf{x}_k^n$ the approximation to $\mathbf{x}(T^n)$ at parareal iteration k, the parareal method first computes an initial guess $\mathbf{x}_0^n$, $n \in \mathbb{N}_0(M-1)$ with $\mathbf{x}_0^0 = \mathbf{x}^0$. Then, for a given parareal iteration k, the method computes

$$\mathbf{x}_{k+1}^{n+1} = \mathscr{G}(\mathbf{x}_{k+1}^n; T^n, T^{n+1}) + \mathscr{F}(\mathbf{x}_k^n; T^n, T^{n+1}) - \mathscr{G}(\mathbf{x}_k^n; T^n, T^{n+1}) \tag{2.5}$$

for $n = k, \ldots, M-1$. This is repeated for parareal iterations $k = 0, \ldots, K$. Here, $\mathbf{x}_{k+1}^k = \mathbf{x}_k^k$ and K is determined by a termination criterion that is satisfied when the solution discontinuities at coarse time instances become sufficiently small. Here, $\mathscr{G} : \mathbb{R}^N \times \mathscr{T} \times \mathscr{T} \to \mathbb{R}^N$ with $(\boldsymbol{\xi}; T^i, T^j) \mapsto \mathscr{G}(\boldsymbol{\xi}; T^i, T^j)$ denotes a "coarse propagator" that propagates a state $\boldsymbol{\xi}$ defined at (coarse) time instance T^i to time instance T^j with $j > i$. In essence, the parareal method alternates serial (inexpensive) *coarse* propagation with parallel (expensive) *fine* propagation; the expectation is that parallelizing the fine propagation can realize wall-time performance improvements. Typically, the coarse propagator is employed to define the initial condition as $\mathbf{x}_0^{n+1} = \mathscr{G}(\mathbf{x}_0^n; T^n, T^{n+1})$. Algorithm 1—which enables alternative initializations—reports the particular parareal algorithm we consider in this work. We note that upon convergence, the algorithm executes an additional update of the final iterate in Steps (22)–(24) to define the ultimate converged solution, as the fine propagation of these states is available from the convergence test and should therefore be used.

Critically, this method exhibits the "finite-termination property"

$$\mathbf{x}_k^n = \mathbf{x}(T^n), \quad n \leq k+1, \tag{2.6}$$

which states that the method will terminate in at most $K = M-1$ parareal iterations. Realizing this "worst-case scenario" implies that the parallelization over time provided no gain over numerically solving (2.1) using the fine propagator in serial.

3. Data-driven time parallelism.

The objective of this work is to devise inputs to Algorithm 1 that leverage the availability of data that inform the *time evolution* of the state. Our two primary points of focus are (1) to devise an initialization method that yields an accurate initial guess, and (2) to develop a coarse propagator that is fast, accurate, and stable. In particular, we aim to improve upon the performance of existing techniques, which generally employ coarse propagators and initialization techniques that do not exploit time-evolution data that may be available.

Our critical assumption is that we have access to *time-evolution bases* $\boldsymbol{\Xi}_j \in \mathbb{V}_a(\mathbb{R}^m)$, $j \in \mathbb{N}(N)$, with $a \leq m$ that describe the *time evolution* of the jth state x_j. Here, $\mathbb{V}_a(\mathbb{R}^m) \subset \mathbb{R}^{m \times a}$ denotes the Stiefel manifold, i.e., the set of all real-valued

Algorithm 1 `parareal_with_initialization`

```
Input: Fine propagator $\mathscr{F}$, coarse propagator $\mathscr{G}$, initialization algorithm initialize,
    initial condition $\mathbf{x}_0^0$, termination tolerance $\epsilon$
Output: Number of parareal iterations $K \leftarrow k$, converged solution
    $(\mathbf{x}_0^1, \ldots, \mathbf{x}_K^{K+1}, \ldots, \mathbf{x}_K^M)$
$k \leftarrow 0$
$(\mathbf{x}_0^1, \ldots, \mathbf{x}_0^M) = \mathtt{initialize}(\mathbf{x}_0^0)$
for $n = 0, \ldots, M-1$ do {parallel fine propagation}
    $\mathbf{f}_0^{n+1} = \mathscr{F}(\mathbf{x}_0^n; T^n, T^{n+1})$
end for
while $\max_{n\in\{k+1,\ldots,M-1\}} \|\mathbf{f}_k^n - \mathbf{x}_k^n\|/\|\mathbf{f}_k^n\| > \epsilon$ do
    if $k = 0$ then {initial-seed coarse propagation}
        for $n = 1, \ldots, M-1$ do {parallel coarse propagation}
            $\mathbf{g}_0^{n+1} = \mathscr{G}(\mathbf{x}_0^n; T^n, T^{n+1})$
        end for
    end if
    $\mathbf{x}_{k+1}^k = \mathbf{x}_k^k$
    for $n = k, \ldots, M-1$ do {serial coarse propagation and correction}
        $\mathbf{g}_{k+1}^{n+1} = \mathscr{G}(\mathbf{x}_{k+1}^n; T^n, T^{n+1})$
        $\mathbf{x}_{k+1}^{n+1} = \mathbf{g}_{k+1}^{n+1} + \mathbf{f}_k^{n+1} - \mathbf{g}_k^{n+1}$
    end for
    for $n = k, \ldots, M-1$ do {parallel fine propagation}
        $\mathbf{f}_{k+1}^{n+1} = \mathscr{F}(\mathbf{x}_{k+1}^n; T^n, T^{n+1})$
    end for
    $k \leftarrow k+1$
end while
for $n = k, \ldots, M-1$ do
    $\mathbf{x}_k^{n+1} \leftarrow \mathbf{f}_k^{n+1}$
end for
```

$m \times a$ matrices with orthonormal columns. Subsequent sections will describe how these bases can be computed in the case of parameterized ODEs (section 5.1) and projection-based ROMs (section 5.2); for now, we simply assume that these bases are available and for ease of notation all have identical dimension a.

3.1. Global forecasting.

We begin by summarizing the data-driven forecasting method proposed in [12]. Given bases $\boldsymbol{\Xi}_j$, $j \in \mathbb{N}(N)$, and a time instance $i \in \mathbb{N}(m)$, the forecasting approach approximates the time evolution of state variable x_j via gappy POD using the basis $\boldsymbol{\Xi}_j$ and the value of x_j at the most recent α time instances. Here, $\alpha \in \mathbb{N}$ with $\alpha \geq a$ denotes the "memory." First, the method computes the gappy POD approximation $\mathbf{w}_j(x_j; t^i)$, defined as

$$\mathbf{w}_j(y; t^i) = \underset{\mathbf{w}\in\mathrm{Ran}(\boldsymbol{\Xi}_j)}{\arg\min} \|\mathbf{Z}_i\mathbf{w} - \mathbf{Z}_i\mathbf{h}(y)\|_2 = \boldsymbol{\Xi}_j[\mathbf{Z}_i\boldsymbol{\Xi}_j]^+\mathbf{Z}_i\mathbf{h}(y), \quad i \in \mathbb{N}_0(m-\alpha),\ y \in \mathscr{H}, \tag{3.1}$$

where the superscript + denotes the Moore–Penrose pseudoinverse, $\mathrm{Ran}(\mathbf{A})$ denotes the range of the matrix $\mathbf{A}$, and $\mathbf{w}_j : \mathscr{H} \times t \to \mathbb{R}^m$. Here, the *sampling matrix* $\mathbf{Z}_i := \begin{bmatrix}\mathbf{e}_{i+1} & \cdots & \mathbf{e}_{i+\alpha}\end{bmatrix}^T \in \{0,1\}^{\alpha\times m}$ extracts entries $i+1$ through $i+\alpha$ of a given vector and $\mathbf{e}_i \in \{0,1\}^m$ denotes the ith canonical unit vector. Further, $\mathbf{h} : \mathscr{H} \to \mathbb{R}^m$

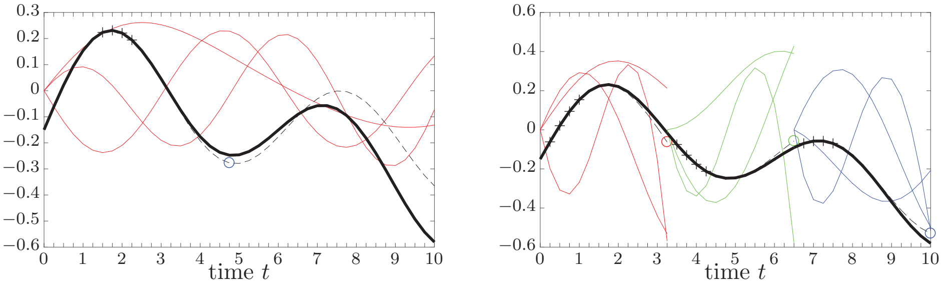
centers and "unrolls" a time-dependent variable according to the time discretization

$$\mathbf{h} : y \mapsto \left[y(t^1) - y(t^0) \ \cdots \ y(t^m) - y(t^0)\right]^T . \tag{3.2}$$

Then, the forecast at a given time instance n, which aims to approximate the value $x_j(t^n)$, is set to $f_j(x_j; t^i, t^n)$, where we have defined the function that forecasts the time-dependent variable y to time t^k using its value at times $t^{i+\ell}$, $\ell \in \mathbb{N}(\alpha)$, as

$$f_j : (y; t^i, t^k) \mapsto y(0) + \mathbf{e}_k^T \mathbf{w}_j(y; t^i), \quad i \in \mathbb{N}_0(m-\alpha),\, k \in \mathbb{N}_0(m),\, y \in \mathscr{H}, \tag{3.3}$$

with $f_j : \mathscr{H} \times t \times t \to \mathbb{R}$. Figure 2(a) illustrates the global-forecasting method graphically.



(a) Global forecast. Here, the time evolution basis vectors (i.e., columns of $\mathbf{\Xi}_j$) are denoted by thin colored lines, the state entry x_j is denoted by a thick black line, the sampled state $\mathbf{Z}_5\mathbf{h}(x_j) + x_j(t^0)$ is denoted by + markers, the forecast $f_j(x_j; t^5, \cdot)$ is plotted as dashed line, and the forecast $f_j(x_j; t^5, t^{19})$ is denoted by a ○ marker.

(b) Local forecast with $M = 3$ coarse time intervals. Here, the time evolution basis vectors (i.e., columns of $\mathbf{\Xi}_j^n$) are denoted by thin colored lines, the state entry x_j is denoted by a thick black line, the sampled state $\mathbf{Z}_0\mathbf{h}^n(x_j) + x_j(T^n)$ is denoted by + markers, and the forecast $f_j^n(x_j; T^n, T^{n+1})$ is denoted by ○ markers.

FIG. 2. *Illustration of global and local forecasting with memory* $\alpha = 4$, *time-evolution basis dimension* $a = 3$, *final time* $T_{\text{final}} = 10$, *and time step* $h = 0.25$.

The approach proposed in [12] employed the forecast $f_j(x_j; t^i, t^n)$, $j \in \mathbb{N}(N)$, as an initial guess for the Newton solver at time t^n for $n > i + \alpha$ obtained after discretizing the ODE associated with a ROM using a linear multistep scheme.[3] Instead, this work considers employing this forecasting strategy to define both the *initialization* and *coarse propagator* as inputs to parareal Algorithm 1. Subsequently, we propose a *local* variant of this global-forecasting method that operates within a single coarse time interval.

3.2. Local forecasting. The proposed *local*-forecasting approach relies on local time-evolution bases $\mathbf{\Xi}_j^n \in \mathbb{V}_a(\mathbb{R}^{\bar{m}})$, $j \in \mathbb{N}(N)$, that inform the time evolution of the jth state x_j over time interval $[T^n, T^{n+1}]$. Given a (global) time-evolution basis $\mathbf{\Xi}_j$, these local bases $\mathbf{\Xi}_j^n$, $n \in \mathbb{N}_0(M-1)$, can be computed via Algorithm 2 as $\mathbf{\Xi}_j^n = \texttt{local\_basis}(\mathbf{\Xi}_j, n, \upsilon)$, where $\upsilon \in [0,1]$ defines a statistical "energy criterion" and we have defined $\mathbf{Z}^n := [\mathbf{e}_{\bar{m}n+1} \ \cdots \ \mathbf{e}_{\bar{m}n+\bar{m}}]^T - \mathbf{1}_{\bar{m}}[\mathbf{e}_{\bar{m}n}]^T \in \{0,1\}^{\bar{m}\times m}$ as the

[3] The method was also generalized to handle Runge–Kutta schemes and second-order ODEs; the only difference in these cases is that the forecast is constructed for the *unknown* variable computed at each time step, which can correspond to the velocity or acceleration depending on the ODE and time integrator.

Algorithm 2 `local_basis`

```
Input: time-evolution basis $\boldsymbol{\Xi}_j \in \mathbb{V}_a(\mathbb{R}^m)$, coarse-time-interval index $n \in \mathbb{N}_0(M-1)$,
    energy criterion $\upsilon \in [0,1]$
Output: local time-evolution basis $\boldsymbol{\Xi}_j^n \in \mathbb{V}_a(\mathbb{R}^{\bar{m}})$
1: $\boldsymbol{\Xi}_j^n \leftarrow \mathbf{Z}^n \boldsymbol{\Xi}_j$ {Extract values on $n$th coarse time interval}
2: $(\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{V}) = \texttt{thin\_SVD}(\boldsymbol{\Xi}_j^n)$ {Compute (thin) singular value decomposition}
3: $\boldsymbol{\Xi}_j^n \leftarrow \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_a\end{bmatrix}$, where $a = \min_{i \in \Upsilon(\upsilon)} i$, $\Upsilon(\upsilon) := \{i \geq 1 \mid \sum_{\ell=1}^{i} \sigma_\ell / \sum_{k=1}^{a} \sigma_k \geq \upsilon\}$,
   $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_a)$, $\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_a\end{bmatrix}$. {Truncate}
```

matrix that samples entries associated with the nth coarse time interval and subtracts the initial value on that time interval. Here, $\mathbf{1}_i$ denotes an i-vector of ones. Note that truncation in Step 3 of Algorithm 2 ensures that the local basis $\boldsymbol{\Xi}_j^n$ will have full column rank. Using these local time-evolution bases (which have zero values at the beginning of their respective time intervals), we can define the local forecast using a construction similar to that of section 3.1. In particular, the linear least-squares problem for the locally defined gappy POD approximation becomes

$$\mathbf{w}_j^n(y; t^i) = \underset{\mathbf{w} \in \operatorname{Ran}(\boldsymbol{\Xi}_j^n)}{\arg\min} \|\bar{\mathbf{Z}}_{i-\bar{m}n}\mathbf{w} - \bar{\mathbf{Z}}_{i-\bar{m}n}\mathbf{h}^n(y)\|_2 = \boldsymbol{\Xi}_j^n[\bar{\mathbf{Z}}_{i-\bar{m}n}\boldsymbol{\Xi}_j^n]^+ \bar{\mathbf{Z}}_{i-\bar{m}n}\mathbf{h}^n(y) \tag{3.4}$$

for $t^{i+\ell} \in \boldsymbol{t}^n$, $\forall \ell \in \mathbb{N}(\alpha)$, with $\mathbf{w}_j^n : \mathscr{H}^n \times \boldsymbol{t}^n \rightarrow \mathbb{R}^{\bar{m}}$, where $\mathscr{H}^n$, $n \in \mathbb{N}_0(M-1)$, denotes the set of functions from $\boldsymbol{t}^n$ to $\mathbb{R}$. Here, the sampling matrix $\bar{\mathbf{Z}}_i := \begin{bmatrix}\mathbf{e}_{i+1} & \cdots & \mathbf{e}_{i+\alpha}\end{bmatrix}^T \in \{0,1\}^{\alpha \times \bar{m}}$ extracts entries $i+1$ through $i+\alpha$ of a given vector defined on a local time interval, and the function $\mathbf{h}^n : \mathscr{H}^n \rightarrow \mathbb{R}^{\bar{m}}$ locally centers and unrolls a time-dependent variable over the nth time interval as

$$\mathbf{h}^n : y \mapsto \begin{bmatrix} y(t^{\bar{m}n+1}) - y(T^n) & \cdots & y(T^{n+1}) - y(T^n) \end{bmatrix}^T; \tag{3.5}$$

note that $T^{n+1} = t^{\bar{m}n+\bar{m}}$. Then, the function f_j^n that forecasts a local time-dependent variable to time t^k using the value of the variable at times $t^{i+\ell}$, $\ell \in \mathbb{N}(\alpha)$, can be defined algebraically as

$$f_j^n : (y; t^i, t^k) \mapsto y(T^n) + \mathbf{e}_{k-\bar{m}n}^T \mathbf{w}_j^n(y; t^i), \quad t^k, t^{i+\ell} \in \boldsymbol{t}^n, \ \forall \ell \in \mathbb{N}(\alpha), \tag{3.6}$$

with $f_j^n : \mathscr{H}^n \times \boldsymbol{t}^n \times \boldsymbol{t}^n \rightarrow \mathbb{R}$. Figure 2(b) illustrates the local-forecasting method graphically.

3.3. Coarse propagator: Local forecast.

We aim to employ the local-forecasting approach to construct a data-driven coarse propagator to be used in the parareal Algorithm 1. In particular, we propose constructing a propagator that maps the state evaluated at the *first α fine time instances* of a given coarse time interval to an approximation of the state at the *final time* of the coarse time interval. However, inspired by the multigrid interpretation of parareal, we acknowledge that the role of the coarse propagator is to reduce *large-wavelength errors*; thus we allow the technique to apply this propagation only to a restriction of the state.[4] That is, we set the

[4] Numerical experiments highlight the importance of this (see Figure 10).

coarse propagation of the jth element of a *restricted* time-dependent vector to be the mapping

$$\boldsymbol{y} \mapsto f_j^n(\mathbf{r}_j^T \boldsymbol{y}; T^n, T^{n+1}) : (\mathscr{H}^n)^N \to \mathbb{R}. \tag{3.7}$$

We define $\mathbf{R} := [\mathbf{r}_1 \ \cdots \ \mathbf{r}_{\bar{N}}] \in \mathbb{R}^{N \times \bar{N}}$ with $\bar{N} \in \mathbb{N}(N)$ as a (linear) restriction operator with associated prolongation operator $\mathbf{P} \in \mathbb{R}^{N \times \bar{N}}$ such that $\mathbf{R}^T\mathbf{P} = \mathbf{I}$. Note that the time-evolution bases should therefore be constructed to capture the time evolution of the *restricted* time-dependent variable. Possible choices for the restriction operator include projection onto large-wavelength Fourier modes or onto a set of high-energy POD modes; the latter choice is natural for ROMs and is explored in the numerical experiments.

Introducing a function that maps a vector at the beginning of a coarse time interval to a function over the fine time discretization within that interval, i.e., $\mathpzc{f}^n : \boldsymbol{\xi} \mapsto \mathscr{F}(\boldsymbol{\xi}; T^n, \cdot)$ with $\mathpzc{f}^n : \mathbb{R}^N \to (\mathscr{H}^n)^N$, we define coarse propagation of the jth element of the restricted state on coarse time interval n to be

$$\mathscr{G}_{\mathrm{LF}}{}_j^n : \boldsymbol{\xi} \mapsto f_j^n(\mathbf{r}_j^T \mathpzc{f}^n(\boldsymbol{\xi}); T^n, T^{n+1}), \quad n \in \mathbb{N}_0(M-1), \tag{3.8}$$

with $\mathscr{G}_{\mathrm{LF}}{}_j^n : \mathbb{R}^N \to \mathbb{R}$, which can be expressed algebraically as

$$\mathscr{G}_{\mathrm{LF}}{}_j^n : \boldsymbol{\xi} \mapsto \mathbf{r}_j^T \boldsymbol{\xi} + \sum_{i=1}^{\alpha} \gamma_{ij}^n (\mathbf{r}_j^T \mathscr{F}(\boldsymbol{\xi}; T^n, t^{\bar{m}n+i}) - \mathbf{r}_j^T \boldsymbol{\xi}), \quad n \in \mathbb{N}_0(M-1), \tag{3.9}$$

with $\gamma_{ij}^n := \mathbf{e}_{\bar{m}}^T \boldsymbol{\Xi}_j^n \left[\bar{\mathbf{Z}}_0 \boldsymbol{\Xi}_j^n\right]^+ \mathbf{e}_i \in \mathbb{R}$. We then propose employing a coarse propagator $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$ with

$$\mathscr{G}_{\mathrm{LF}} : (\boldsymbol{\xi}; T^n, T^{n+1}) \mapsto \mathbf{P} \left[\mathscr{G}_{\mathrm{LF}}{}_1^n(\boldsymbol{\xi}) \ \cdots \ \mathscr{G}_{\mathrm{LF}}{}_{\bar{N}}^n(\boldsymbol{\xi})\right]^T, \tag{3.10}$$

which can be expressed algebraically as

$$\mathscr{G}_{\mathrm{LF}} : (\boldsymbol{\xi}; T^n, T^{n+1}) \mapsto \mathbf{P}\mathbf{R}^T \boldsymbol{\xi} + \mathbf{P} \sum_{i=1}^{\alpha} \boldsymbol{\Gamma}_i^n \left[\mathbf{R}^T \mathscr{F}(\boldsymbol{\xi}; T^n, t^{\bar{m}n+i}) - \mathbf{R}^T \boldsymbol{\xi}\right]. \tag{3.11}$$

Here, we have defined $\boldsymbol{\Gamma}_i^n := \mathrm{diag}(\gamma_{i1}^n, \ldots, \gamma_{i\bar{N}}^n) \in \mathbb{R}^{\bar{N} \times \bar{N}}$.

3.4. Initialization: Local and global forecasts. Initialization in Step 2 of Algorithm 1 is typically executed by sequentially applying the coarse propagator, i.e.,

$$\mathbf{x}_0^{n+1} = \mathscr{G}(\mathbf{x}_0^n; T^n, T^{n+1}), \quad n = 0, \ldots, M-1. \tag{3.12}$$

This approach could be applied with the proposed local-forecasting coarse propagator $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$. However, we can also consider an alternative initialization that is both computationally less expensive and more stable (as will be further discussed in Remark 4.6).

In particular, we consider performing initialization by forecasting the state from *the first α time steps* of the first time interval to *all coarse time instances* using the global time-evolution bases $\boldsymbol{\Xi}_j$. That is, we can perform initialization via *global forecasting* as

$$\mathbf{x}_0^{n+1} = \mathscr{G}_{\mathrm{GF}}(\mathbf{x}_0^0; T^{n+1}), \quad n = 0, \ldots, M-1, \tag{3.13}$$

where we have defined

$$\mathscr{G}_{\mathrm{GF}} : (\boldsymbol{\xi}; T^n) \mapsto \mathbf{P}[f_1(\mathbf{r}_1^T \mathscr{f}(\boldsymbol{\xi}); T^0, T^n) \ \cdots \ f_{\bar{N}}(\mathbf{r}_{\bar{N}}^T \mathscr{f}(\boldsymbol{\xi}); T^0, T^n)]^T \tag{3.14}$$

with $\mathscr{G}_{\mathrm{GF}} : \mathbb{R}^N \times \mathscr{T} \to \mathbb{R}^N$, and we have introduced a function that maps a vector at the initial time t^0 to a function over the fine time discretization over the entire time interval, i.e., $\mathscr{f}(\boldsymbol{\xi}) : \boldsymbol{\xi} \mapsto \mathscr{F}(\boldsymbol{\xi}; t^0, \cdot)$ with $\mathscr{f}(\boldsymbol{\xi}) : \mathbb{R}^N \to (\mathscr{H})^N$. This global forecast can be expressed algebraically as

$$\mathscr{G}_{\mathrm{GF}} : (\boldsymbol{\xi}; T^n) \mapsto \mathbf{P}\mathbf{R}^T \boldsymbol{\xi} + \mathbf{P} \sum_{i=1}^{\alpha} \bar{\boldsymbol{\Gamma}}_i^n \left[\mathbf{R}^T \mathscr{F}(\boldsymbol{\xi}; T^0, t^i) - \mathbf{R}^T \boldsymbol{\xi}\right]. \tag{3.15}$$

Here, we have defined $\bar{\boldsymbol{\Gamma}}_i^n := \mathrm{diag}(\bar{\gamma}_{i1}^n, \ldots, \bar{\gamma}_{i\bar{N}}^n) \in \mathbb{R}^{\bar{N} \times \bar{N}}$ and $\bar{\gamma}_{ij}^n := \mathbf{e}_{\bar{m}n}^T \boldsymbol{\Xi}_j \left[\mathbf{Z}_0 \boldsymbol{\Xi}_j\right]^+ \mathbf{e}_i \in \mathbb{R}$.

In principle, the local and global forecasts can employ different values for the memory α and restricted-state dimension $\bar{N}$. However, for notational simplicity, we assume these values are identical.

4. Analysis.

We now analyze the proposed data-driven time-parallel methodology to derive insight into the coarse-propagator error (section 4.1), the method's theoretical speedup (section 4.2), the method's stability (section 4.3), and convergence aspects (section 4.4). All norms in this section refer to the Euclidean norm unless otherwise specified. This section states the final results of the analysis, while section S1 of the supplementary materials contains all proofs.

4.1. Coarse-propagator error analysis.

We first analyze the error of the coarse propagator with respect to the fine propagator.

4.1.1. General case.

We introduce the following assumptions:

A1 The restriction and prolongation operators have counterparts $\mathbf{R}_\perp \in \mathbb{R}^{N \times (N - \bar{N})}$ and $\mathbf{P}_\perp \in \mathbb{R}^{N \times (N-\bar{N})}$, respectively, that satisfy $\boldsymbol{\xi} = \mathbf{P}\mathbf{R}^T\boldsymbol{\xi} + \mathbf{P}_\perp \mathbf{R}_\perp^T \boldsymbol{\xi}\ \forall \boldsymbol{\xi} \in \mathbb{R}^N$.

A2 The prolongation operators are bounded by constants $M_{\mathbf{P}}, M_{\mathbf{P}_\perp} \in \mathbb{R}$, i.e., $\|\mathbf{P}\boldsymbol{\xi}\| \leq M_{\mathbf{P}} \|\boldsymbol{\xi}\|\ \forall \boldsymbol{\xi} \in \mathbb{R}^{\bar{N}}$ and $\|\mathbf{P}_\perp \boldsymbol{\xi}\| \leq M_{\mathbf{P}_\perp} \|\boldsymbol{\xi}\|\ \forall \boldsymbol{\xi} \in \mathbb{R}^{N-\bar{N}}$.

THEOREM 4.1. *If assumptions* A1 *and* A2 *hold, then*

$$\begin{aligned} &\|\mathscr{F}(\boldsymbol{\xi}; T^n, T^{n+1}) - \mathscr{G}_{\mathrm{LF}}(\boldsymbol{\xi}; T^n, T^{n+1})\| \leq M_{\mathbf{P}_\perp} \|\mathbf{R}_\perp^T \mathscr{F}(\boldsymbol{\xi}; T^n, T^{n+1})\| \\ &\qquad + M_{\mathbf{P}} \sum_{j=1}^{\bar{N}} \beta_j^n \left\| (\mathbf{I}_{\bar{m}} - \boldsymbol{\Xi}_j^n [\boldsymbol{\Xi}_j^n]^T) \begin{bmatrix} \mathbf{r}_j^T \mathscr{F}(\boldsymbol{\xi}; T^n, t^{\bar{m}n+1}) - \mathbf{r}_j^T \boldsymbol{\xi} \\ \vdots \\ \mathbf{r}_j^T \mathscr{F}(\boldsymbol{\xi}; T^n, T^{n+1}) - \mathbf{r}_j^T \boldsymbol{\xi} \end{bmatrix} \right\|, \end{aligned} \tag{4.1}$$

where $\beta_j^n := 1/\sigma_{\min}\left(\bar{\mathbf{Z}}_0 \boldsymbol{\Xi}_j^n\right) \geq 1$.[5]

Remark 4.1 (interpolation versus oversampling). As the memory α increases, the inverse stability constants β_j^n in inequality (4.1) monotonically decrease. This occurs because increasing the memory has the effect of appending a row to the matrix $\bar{\mathbf{Z}}_0 \boldsymbol{\Xi}_j^n$, which cannot decrease its minimum singular value. This highlights the stabilizing

[5] Note that $\beta_j^n \geq 1$ because appending a row to a matrix cannot decrease its minimum singular value, and $\sigma_{\min}\left(\boldsymbol{\Xi}_j^n\right) = 1$ because $\boldsymbol{\Xi}_j^n \in \mathbb{V}_a(\mathbb{R}^{\bar{m}})$.

effect of employing a least-squares approach (i.e., gappy POD) as opposed to an interpolation approach (i.e., the (discrete) empirical interpolation method D(EIM)) in the forecast: oversampling can reduce a bound for the error between the fine and coarse propagators.

Remark 4.2 (restriction tradeoff). Increasing the dimension of the restriction operator (i.e., the number of variables included in the local forecast $\bar{N}$) decreases the first term in bound (4.1). However, doing so also increases the second term, as the number of terms in the summation increases. This latter effect is exacerbated when the time evolution of higher-index solution components (i.e., $\mathbf{r}_j^T\boldsymbol{\xi}$ for large j) is not well captured by the associated time-evolution bases (i.e., $\boldsymbol{\Xi}_j^n$ for large j); this can occur, for example, if higher-index solution components associate with high-frequency solution modes, as is the case when the restriction operator associates with a projection onto a low-frequency Fourier or POD (see section 5) basis. These two effects comprise the tradeoff that should be considered when selecting the dimension of the restriction operator $\bar{N}$ in practice.

4.1.2. Ideal case.

We now show that the coarse propagator is exact (i.e., incurs no error with respect to the fine propagator) under the following "ideal conditions":

- A3 The time evolution of the restricted state is an element of the subspace spanned by the time-evolution basis (i.e., $\mathbf{h}(\mathbf{r}_j^T\mathbf{x}) \in \mathrm{Ran}(\boldsymbol{\Xi}_j)$, $j \in \mathbb{N}(\bar{N})$).
- A4 The local bases are constructed with no truncation (i.e., $\upsilon = 1.0$ in Algorithm 2).
- A5 The state lies in the range of the projector $\mathbf{P}\mathbf{R}^T$ at coarse time instances T^n, $n \in \mathbb{N}(M)$, i.e., $\mathbf{P}\mathbf{R}^T\mathbf{x}(T^n) = \mathbf{x}(T^n)$, $n \in \mathbb{N}(M)$.

Intuitively, we emphasize that assumption A3 holds if and only if the time-evolution basis $\boldsymbol{\Xi}_j$ can *exactly* represent the time evolution of the restricted state $\mathbf{r}_j^T\mathbf{x}$ for all $j \in \mathbb{N}(\bar{N})$.

LEMMA 4.2 (local-subspace condition). *If assumptions* A3 *and* A4 *hold, then* $\mathbf{h}^n(\mathbf{r}_j^T\mathbf{x}) \in \mathrm{Ran}(\boldsymbol{\Xi}_j^n),\ j \in \mathbb{N}(N),\ n \in \mathbb{N}_0(M-1)$.

THEOREM 4.3 (exact coarse propagator). *Under assumptions* A3, A4, *and* A5, *the coarse propagator is exact when applied to the state, i.e.,* $\mathscr{G}_{\mathrm{LF}}(\mathbf{x}(T^n); T^n, T^{n+1}) = \mathscr{F}(\mathbf{x}(T^n); T^n, T^{n+1}) = \mathbf{x}(T^{n+1}),\ n \in \mathbb{N}_0(M-1)$.

4.2. Speedup analysis.

This section analyzes the theoretical speedup of the method under various conditions. Section 4.2.1 provides the theoretical speedup of the methodology achieved for a given number of parareal iterations when both the local forecast (Theorem 4.4) and global forecast (Theorem 4.5) are employed for initialization. Section 4.2.2 derives theoretical speedups for the method under "ideal conditions" for both the local-forecast (Theorem 4.6) and global-forecast (Theorem 4.7) initializations. Section S3 of the supplementary materials shows that the proposed method can produce super-ideal theoretical speedups when the forecast is also employed for providing initial guesses for the Newton solver in the case of implicit fine propagators and nonlinear dynamical systems. We note that while the ultimate objective is to assess the cost incurred to achieve a given error level, this analysis is difficult to perform a priori for general dynamical systems; thus, we assess this effect numerically in section 6.[6]

Each theoretical result employs a subset of the following assumptions:

[6] In particular, we refer the reader to Figures 6(c)–6(d), 9(c)–9(d), and 11.

A6 Initialization in Step 2 of Algorithm 1 is computed via local forecasting (i.e., (3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$).

A7 Initialization in Step 2 of Algorithm 1 is computed via global forecasting (i.e., (3.13)).

A8 The wall-time incurred by computing time advancement with the fine propagator $\mathscr{F}(\mathbf{x}(t^m); t^m, t^{m+1})$ dominates all other costs and parallel overhead.

Further, all speedup results assume that the number of processors is equal to the number of coarse time intervals M.

4.2.1. General case.

Theorem 4.4 (*speedup*: local-forecast initialization). *If assumptions* A6 *and* A8 *hold, then the proposed method (which employs the local forecast for initialization and coarse propagation) upon convergence in K parareal iterations realizes a speedup of*

$$S_{\mathrm{LF\text{-}LF}}(K) := \frac{m}{\left[\alpha(M - \frac{K}{2}) + \bar{m}\right](K+1) - \alpha(K+1)}. \tag{4.2}$$

Theorem 4.5 (*speedup*: global-forecast initialization). *If assumptions* A7 *and* A8 *hold, then the proposed method (which employs the global forecast for initialization and the local forecast for coarse propagation) upon convergence in K parareal iterations realizes a speedup of*

$$S_{\mathrm{GF\text{-}LF}}(K) := \frac{m}{\alpha(1 + \mathbf{1}_{K>0}) + \alpha K(M - \frac{1}{2}(1+K)) + (K+1)\bar{m} - \alpha K}. \tag{4.3}$$

Here, the indicator function is defined as $\mathbf{1}_A = 1$ if A is true, while $\mathbf{1}_A = 0$ otherwise.

Remark 4.3 (memory tradeoff: iteration count and speedup). Equations (4.2) and (4.3) demonstrate that increasing the memory α can reduce the speedup of the methodology, assuming the number of iterations K needed for convergence is constant. However, as discussed in Remark 4.1, increasing the memory also leads to a nonincreasing bound for the error between coarse and fine propagators. As such, it may lead to a decreasing error bound; because improved coarse-propagator accuracy can promote convergence (see Remark 4.7), increasing the memory can thus reduce the number of total parareal iterations K. These two effects constitute the tradeoff that should be considered when selecting the memory α in practice.

Remark 4.4 (reuse of sampled state). We note that the α applications of the fine propagator employed by the local-forecast coarse propagator to sample the restricted state can be reused during the subsequent fine propagation; this leads to speedup improvements as manifested in terms $-\alpha(K+1)$ and $-\alpha K$ in the denominators of (4.2) and (4.3), respectively. This is also an important aspect of the practical implementation of the local-forecast coarse propagator.

4.2.2. Ideal case.

In this section, we derive theoretical speedups for the method under "ideal conditions," i.e., when Assumption A3 holds.

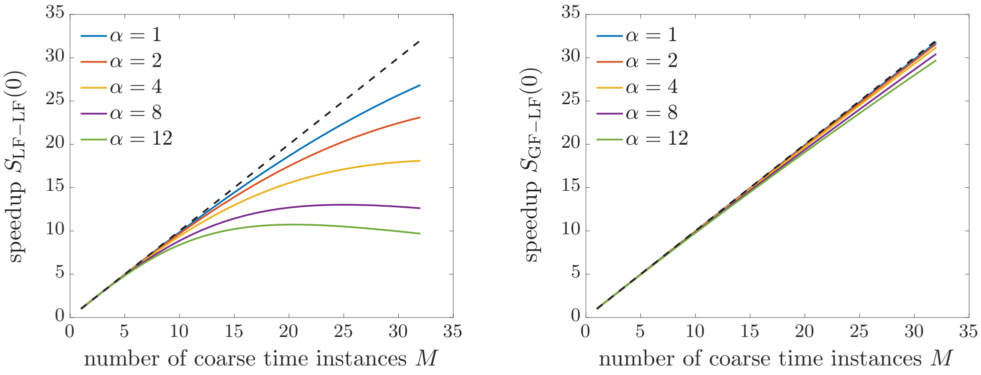
Theorem 4.6 (*ideal-conditions speedup*: local-forecast initialization). *If assumptions* A3, A4, A5, *and* A6 *hold, then the proposed method converges after initialization (i.e., $K = 0$ in Algorithm* 1*). Further, if assumption* A8 *holds, then the method realizes a speedup of $S_{\mathrm{LF\text{-}LF}}(0) = m/((M-1)\alpha + \bar{m})$.*

Figure 3(a) provides a visualization of this theoretical speedup for specific values of method parameters. First, note that the "serial bottleneck" of time evolution is apparent from this result: the speedup degrades as the number of coarse time instances M increases. This is due to the requirement of computing α fine propagations in serial across coarse time intervals for this initialization method. Second, note that the memory α has an appreciable effect on the speedup; keeping this value as low as possible without compromising convergence is thus desirable.

THEOREM 4.7 (*ideal-conditions speedup*: global-forecast initialization). *If assumptions* A3, A5, *and* A7 *hold, then the proposed method converges after parareal initialization (i.e., $K = 0$ in Algorithm* 1*). Further, if assumption* A8 *holds, then the method realizes a speedup of*

$$S_{\text{GF-LF}}(0) = \frac{m}{\alpha + \bar{m}}. \tag{4.4}$$

Figure 3(b) visualizes this theoretical speedup in the case of global-forecast initialization. As compared with local-forecast initialization, note that the theoretical speedup realizable by the global forecast is much closer to ideal. Further, it is more stable, as discussed in Remark 4.6.



(a) Local-forecast initialization

(b) Global-forecast initialization

FIG. 3. *Ideal-conditions speedup. Plot corresponds to a strong-scaling study with $m = 5000$ fine time instances, setting the number of processors equal to the number of coarse time instances M.*

To illustrate the full potential of the proposed approach, section S3 of the supplementary materials demonstrates that super-ideal speedups can be realized when the proposed methodology is combined with the method presented in [12] for defining Newton-solver initial guesses via forecasting.

4.3. Stability analysis. We begin by providing a general statement for stability of the parareal recurrence; we then derive specific quantities needed to demonstrate stability when the proposed forecast is employed as a coarse propagator. These results employ a subset of the following assumptions:

A9 The fine propagator is stable;[7] i.e., there exists a constant $C_{\mathcal{F}}$ such that $\|\mathcal{F}(\boldsymbol{\xi}; t^i, t^j)\| \leq (1 + C_{\mathcal{F}}(t^j - t^i))\|\boldsymbol{\xi}\| \ \forall \boldsymbol{\xi} \in \mathbb{R}^N$.

A10 The restriction operators and prolongation operator counterparts are bounded by constants $M_\perp$, $M_{\mathbf{r}_j} \in \mathbb{R}$, i.e., $\|\mathbf{P}_\perp \mathbf{R}_\perp^T \boldsymbol{\xi}\| \leq M_\perp \|\boldsymbol{\xi}\| \ \forall \boldsymbol{\xi} \in \mathbb{R}^N$ and $\|\mathbf{r}_j^T \boldsymbol{\xi}\| \leq M_{\mathbf{r}_j} \|\boldsymbol{\xi}\| \ \forall \boldsymbol{\xi} \in \mathbb{R}^N$.

[7] Note that this assumption implies that $\boldsymbol{\xi} = 0$ is an equilibrium point. The following analysis also holds when $\|\boldsymbol{\xi}\|$ is replaced by $\|\boldsymbol{\xi} - \boldsymbol{\xi}_e\|$ with $\boldsymbol{\xi}_e$ an equilibrium point. The same applies to the bound in (4.5).

The following lemma follows some elements of the stability analysis performed in [14].

Lemma 4.8 (general parareal stability). *If constants α_A and C_A exist such that the coarse propagator can be bounded as*

$$\|\mathscr{G}(\boldsymbol{\xi}; T^n, T^{n+1})\| \leq \alpha_A(1 + C_A H)\|\boldsymbol{\xi}\| \tag{4.5}$$

and constants α_B and C_B exist such that the difference between the coarse and fine propagators can be bounded as

$$\|\mathscr{F}(\boldsymbol{\xi}; T^n, T^{n+1}) - \mathscr{G}(\boldsymbol{\xi}; T^n, T^{n+1})\| \leq \alpha_B(1 + C_B H)\|\boldsymbol{\xi}\|, \tag{4.6}$$

then the parareal recurrence (2.5) *is stable, as it satisfies*

$$\begin{aligned} \|\mathbf{x}_k^n\| \leq& (\alpha_A)^n \exp(C_A n H)\|\mathbf{x}^0\| \\ &+ \sum_{j=1}^{n} (\alpha_A)^{n-j} \alpha_B \exp(((n-j)C_A + C_B)H)\|\mathbf{x}_{k-1}^{j-1}\|, \quad n \in \mathbb{N}(M),\ k \in \mathbb{N}(n), \end{aligned} \tag{4.7}$$

$$\|\mathbf{x}_k^n\| \leq \sum_{j=0}^{n} \binom{n}{j} (\alpha_A)^j (\alpha_B)^{n-j} \exp\left((jC_A + (n-j)C_B)\, H\right) \|\mathbf{x}^0\|, \quad n \in \mathbb{N}(M),\ k = n. \tag{4.8}$$

We now derive the quantities α_A, C_A, α_B, and C_B from Lemma 4.8 that are specific to the proposed coarse propagator $\mathscr{G}_{\mathrm{LF}}$.

Lemma 4.9 (stability of proposed coarse propagator). *Under assumptions* A2*,* A9*, and* A10

$$\|\mathscr{G}_{\mathrm{LF}}(\boldsymbol{\xi}; T^n, T^{n+1})\| \leq \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N})} M_{\mathbf{r}_j} \alpha_j^n \left(1 + C_j^n C_{\mathscr{F}} H\right) \|\boldsymbol{\xi}\|, \tag{4.9}$$

where we have defined $\alpha_j^n := \left(\lambda_j^n + \kappa_j^n \sqrt{\alpha}\right) \geq 1$, $C_j^n := \frac{(\alpha/\bar{m})\kappa_j^n \sqrt{\alpha}}{\lambda_j^n + \kappa_j^n \sqrt{\alpha}} \leq 1$, $\lambda_j^n := |1 - \sum_{i=1}^{\alpha} \gamma_{ij}^n|$ and $\kappa_j^n := \sqrt{\sum_{i=1}^{\alpha} (\gamma_{ij}^n)^2}$. Hence, we have demonstrated stability in the sense of (4.5)*.*

Remark 4.5 (bound dependence on discretization). We note that for a fixed coarse time step H and time-sampling fraction $\alpha/\bar{m}$, the stability constants α_j^n and C_j^n that appear in bound (4.9) approach constant values as the fine time step h approaches zero, rendering the resulting bound insensitive to the underlying time discretization. Section S2 of the supplementary materials provides a detailed description and numerical investigation of this effect.

Remark 4.6 (superior stability of global-forecasting initialization). We now consider the implications of Lemma 4.9 in terms of the two initialization methods proposed in section 3.4. The first proposal involved applying the local forecast for initialization, i.e., computing the initial values $\mathbf{x}_0^n$, $n \in \mathbb{N}(M)$, via (3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$. Applying inequality (4.9) to (3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$ leads to the following stability result for the computed initial values: $\|\mathbf{x}_0^n\| \leq (\upsilon)^n \|\mathbf{x}^0\|$, $n \in \mathbb{N}(M)$. Here, $\upsilon := \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N}), n \in \mathbb{N}_0(M-1)} M_{\mathbf{r}_j} \alpha_j^n \left(1 + C_j^n C_{\mathscr{F}} H\right)$; that is, the stability factor associated with local-forecast initialization grows exponentially in the number of coarse time instances n. This phenomenon can be interpreted as follows: small errors in a local forecast can be amplified by subsequent local forecasts, as these are performed sequentially.

On the other hand, by comparing (3.13) and (3.10), one can note that global-forecast initialization (3.13) is equivalent to applying the local forecast with global time-evolution bases $\boldsymbol{\Xi}_j$ over a time interval $T^{n+1} - T^0$. Thus, the stability of the global-forecast initialization can be derived directly from inequality (4.9) applied with these modifications as

$$\|\mathbf{x}_0^{n+1}\| \leq \bar{\upsilon}\|\mathbf{x}^0\|, \quad n \in \mathbb{N}_0(M-1). \tag{4.10}$$

Here, we have defined

$$\bar{\upsilon} := \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N}), n \in \mathbb{N}_0(M-1)} M_{\mathbf{r}_j} \bar{\alpha}_j^n \left(1 + \bar{C}_j^n C_{\mathscr{F}} H\right),$$

$\bar{\alpha}_j^n := \left(\bar{\lambda}_j^n + \bar{\kappa}_j^n \sqrt{\alpha}\right) \geq 1$, $\bar{C}_j^n := \frac{(\alpha/\bar{m})\bar{\kappa}_j^n\sqrt{\alpha}}{\bar{\lambda}_j^n + \bar{\kappa}_j^n\sqrt{\alpha}} \leq 1$, $\bar{\lambda}_j^n := |1 - \sum_{i=1}^{\alpha} \bar{\gamma}_{ij}^n|$, and $\bar{\kappa}_j^n := \sqrt{\sum_{i=1}^{\alpha} (\bar{\gamma}_{ij}^n)^2}$. Inequality (4.10) shows that the stability factor associated with global-forecast initialization does not grow with the number of coarse time instances; it depends on the coarse time instance only through the quantities $\bar{\lambda}_j^n$ and $\bar{\kappa}_j^n$, which should not grow with n. This phenomenon can be interpreted as follows: small forecasting errors cannot be amplified, as a single forecast is employed for the entire time interval.

Lemma 4.10 (stability of difference between fine and proposed coarse propagators). *If assumptions* A1*,* A2*,* A9*, and* A10 *hold, then*

$$\begin{aligned} &\|\mathscr{F}(\boldsymbol{\xi}; T^n, T^{n+1}) - \mathscr{G}_{\mathrm{LF}}(\boldsymbol{\xi}; T^n, T^{n+1})\| \\ &\qquad \leq \left(M_{\perp} + \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N})} M_{\mathbf{r}_j} (\alpha_j^n + 1)\right)(1 + C_{\mathscr{F}} H)\, \|\boldsymbol{\xi}\|. \end{aligned} \tag{4.11}$$

Theorem 4.11 (parareal stability with proposed coarse propagator). *Under assumptions* A1*,* A2*,* A9*, and* A10*, employing the proposed coarse propagator in the parareal recurrence (i.e.,* $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$ *in* (2.5)*) yields a stable recurrence, as the iterates satisfy* (4.7) *and* (4.8) *with*

$$\alpha_A = \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N}), n \in \mathbb{N}_0(M-1)} M_{\mathbf{r}_j} \alpha_j^n, \quad C_A = C_{\mathscr{F}} \max_{j \in \mathbb{N}(\bar{N}), n \in \mathbb{N}_0(M-1)} C_j^n, \tag{4.12}$$

$$\alpha_B = \left(M_{\perp} + \sqrt{\bar{N}} M_{\mathbf{P}} \max_{j \in \mathbb{N}(\bar{N})} M_{\mathbf{r}_j} (\alpha_j^n + 1)\right), \quad C_B = C_{\mathscr{F}}. \tag{4.13}$$

Remark 4.5 shows that, for a fixed coarse time step H and time-sampling fraction $\alpha/\bar{m}$, the stability constants α_j^n and C_j^n approach constant values as the time step h approaches zero. As all other quantities in the stability bounds (4.7)–(4.8) (with coefficients specified in Theorem 4.11) are independent of the underlying fine time discretization (i.e., h and $\bar{m}$), we know the stability result is not sensitive to the selected fine time step if it is taken to be sufficiently small.

4.4. Convergence analysis. Recall that the proposed approach merely defines alternative techniques for initialization and coarse propagation for the parareal method. Thus, one might expect that existing convergence results derived for the parareal methods still hold in the present context. However, this is not always the case, as many existing results assume that the coarse propagator corresponds to a time integrator with a known order of accuracy [27, 3, 4, 23]; the local-forecast coarse propagator cannot be straightforwardly assigned such an order of accuracy, as its error is bounded by an expression that does not explicitly depend on the coarse time step H (see Theorem 4.1).

Instead, we can make use of existing convergence results that require only a *general definition* of the coarse propagator. One such example is the convergence analysis of [24], which assumes a fixed coarse time step H and assesses convergence as the number of parareal iterations increases. Theorem S4.1 in section S4 of the supplementary materials provides this result.

We now show how those results can be extended to the proposed initialization and coarse propagator. We require the following assumptions:

A11 Problem (2.1) is scalar and linear, i.e., $N = 1$ and $g : (\xi; t) \mapsto a\xi$ with $a \in \mathbb{R}$.

A12 The same local basis is employed for every coarse time interval, i.e., $\boldsymbol{\Xi}^i = \boldsymbol{\Xi}^j = \boldsymbol{\Xi} \in \mathbb{V}_a(\mathbb{R}^{\bar{m}})$, $i, j \in \mathbb{N}_0(M-1)$.

A13 The forecast satisfies the inequality $|1 + \sum_{i=1}^{\alpha} \gamma_i[(\bar{a}_{\mathscr{F}})^i - 1]| < 1$ with $\gamma_i := \mathbf{e}_{\bar{m}}^T \boldsymbol{\Xi} \left[\bar{\mathbf{Z}}_0 \boldsymbol{\Xi}\right]^+ \mathbf{e}_i \in \mathbb{R}$, $i \in \mathbb{N}(\alpha)$.

Corollary 4.12 (superlinear parareal convergence using the proposed coarse propagator). *Under assumptions* A11 *and* A12*, the proposed coarse propagator is linear and satisfies* $\mathscr{G}_{\mathrm{LF}} : (\xi; T^n, T^{n+1}) \mapsto \bar{a}_{LF}\xi$ *with* $\bar{a}_{LF} := 1 + \sum_{i=1}^{\alpha} \gamma_i[(\bar{a}_{\mathscr{F}})^i - 1]$*. Further, the error in the parareal recurrence executed with the proposed coarse propagator* $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$ *satisfies*

$$\max_{n\in\mathbb{N}(M)} |x(T^n) - x_k^n| \leq (\rho)^k \|(\mathbf{M}(\bar{a}_{LF}))^k\|_\infty \max_{n\in\mathbb{N}(M)} |x(T^n) - x_0^n|, \tag{4.14}$$

where $\rho := |(\bar{a}_{\mathscr{F}})^{\bar{m}} - \bar{a}_{LF}| = |(\bar{a}_{\mathscr{F}})^{\bar{m}} - 1 - \sum_{i=1}^{\alpha} \gamma_i[(\bar{a}_{\mathscr{F}})^i - 1]|$ *and* $\mathbf{M} \in \mathbb{R}^{M\times M}$ *is a Toeplitz matrix whose elements are defined by the value of the elements in the first column, which are*

$$m_{i,1}(\beta) = \begin{cases} 0, & i = 1, \\ \beta^{i-2}, & i \in \{2, \ldots, M\}. \end{cases} \tag{4.15}$$

If assumption A13 *additionally holds, then the recurrence converges superlinearly as*

$$\max_{n\in\mathbb{N}(M)} |x(T^n) - x_k^n| \leq \frac{(\rho)^k \left(\prod_{j=1}^{k}(M-j)\right)}{k!} \max_{n\in\mathbb{N}(M)} |x(T^n) - x_0^n|. \tag{4.16}$$

Section S4 in the supplementary materials provides the proof.

Remark 4.7 (role of accuracy in convergence). Inequalities (4.14)–(4.16) demonstrate the effect of coarse-propagation and initial-seed accuracy on convergence. In particular, the term ρ represents the error the coarse propagator incurs with respect to the fine propagator; this is precisely the quantity we aim to minimize with the proposed coarse propagator. In fact, Theorem 4.1 bounds this error, and Theorem 4.3 demonstrates that this error is zero under "ideal conditions." Further, the error incurred by the initial seed appears as $\max_{n\in\mathbb{N}(M)} |x(T^n) - x_0^n|$ in these results. This is the term we aim to minimize by applying the proposed local- and global-forecasting methods for initialization; this quantity is also zero under the "ideal conditions" stated in Theorem 4.3.

Algorithm 3 `pod`

```
Input: training parameter instances $\{\bar{\boldsymbol{\mu}}_i\}_{i=1}^{N_{\text{train}}} \subset \mathcal{D}$, $\upsilon \in [0,1]$
Output: POD state basis $\mathbf{U} \in \mathbb{V}_q(\mathbb{R}^N)$, POD time-evolution bases $\mathbf{V}_j \in \mathbb{V}_{N_{\text{train}}}(\mathbb{R}^m)$, $j = 1,\ldots,q$
for $i = 1,\ldots,N_{\text{train}}$ do {collect snapshots}
    Numerically solve (5.1) with $\boldsymbol{\mu} \leftarrow \bar{\boldsymbol{\mu}}_i$ to obtain snapshots
    $\mathbf{X}_i := [\mathbf{x}(t^1,\bar{\boldsymbol{\mu}}_i) - \mathbf{x}^0(\bar{\boldsymbol{\mu}}_i) \ \cdots \ \mathbf{x}(t^m,\bar{\boldsymbol{\mu}}_i) - \mathbf{x}^0(\bar{\boldsymbol{\mu}}_i)] \in \mathbb{R}^{N\times m}$
end for
$(\mathbf{U},\boldsymbol{\Sigma},\mathbf{V}) =$ thin_SVD($[\mathbf{X}_1 \ \cdots \ \mathbf{X}_{N_{\text{train}}}]$) {Compute (thin) singular value decomposition}
$\mathbf{U} \leftarrow \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_q\end{bmatrix}$, where $q = \min_{i\in\Upsilon(\upsilon)} i$, $\Upsilon(\upsilon) := \{i \mid \sum_{j=1}^{i}\sigma_j/\sum_{k=1}^{mN_{\text{train}}}\sigma_k \geq \upsilon\}$,
$\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1,\ldots,\sigma_{mN_{\text{train}}})$. {Compute truncated state basis}
for $j = 1,\ldots,q$ do {Extract temporal bases from right singular vectors}
    $(\mathbf{Q}_j,\mathbf{R}_j) =$ thin_QR($[\mathbf{v}_j^1 \ \cdots \ \mathbf{v}_j^{N_{\text{train}}}]$) with $\mathbf{v}_j^i := [v_{j,m(i-1)+1} \ \cdots \ v_{j,mi}]^T$ {Compute (thin) QR factorization}
    $\mathbf{V}_j \leftarrow \mathbf{Q}_j$
end for
```

5. Computing forecasting ingredients via SVD/POD.

We now describe how the three ingredients that define the proposed methodology—the time-evolution bases $\boldsymbol{\Xi}_j$, $j \in \mathbb{N}(N)$, the restriction operator $\mathbf{R}$, and the prolongation operator $\mathbf{P}$—can be computed using POD. Section 5.1 describes this for the case of parameterized ODEs, while section 5.2 specializes this for the case of POD-based ROMs.

5.1. Parameterized ODEs.

We first introduce a parameterized variant of the governing initial-value ODE problem (2.1):

$$\frac{d}{dt}\mathbf{x}^\star(t,\boldsymbol{\mu}) = \mathbf{g}\left(\mathbf{x}^\star;t,\boldsymbol{\mu}\right), \quad \mathbf{x}^\star(0,\boldsymbol{\mu}) = \mathbf{x}^0(\boldsymbol{\mu}), \tag{5.1}$$

where $\boldsymbol{\mu} \in \mathcal{D} \subset \mathbb{R}^p$ denotes the parameters, $\mathbf{x}^\star : \mathbb{T}\times\mathcal{D} \to \mathbb{R}^N$ denotes the (parameterized) state implicitly defined as the exact solution to problem (5.1), $\mathbf{g} : \mathbb{R}^N \times \mathbb{T} \times \mathcal{D} \to \mathbb{R}^N$ with $(\boldsymbol{\xi};t,\boldsymbol{\mu}) \mapsto \mathbf{g}(\boldsymbol{\xi};t,\boldsymbol{\mu})$ denotes the velocity, and $\mathbf{x}^0 : \mathcal{D} \to \mathbb{R}^N$ denotes the initial state. Analogously to (2.4), we define $\mathbf{x}(\cdot,\boldsymbol{\mu}) : t \mapsto \mathcal{F}(\mathbf{x}^0(\boldsymbol{\mu});0,t)$ for $t \in \mathcal{t}$ as the associated numerical solution with $\mathbf{x}(\cdot,\boldsymbol{\mu}) \in (\mathcal{H})^N$.

The ingredients required for the proposed methodology can be computed in a data-driven manner via the POD method by executing the following steps:

1. Given training parameter instances $\{\bar{\boldsymbol{\mu}}_i\}_{i=1}^{N_{\text{train}}} \subset \mathcal{D}$ and energy criterion $\upsilon \in [0,1]$, execute Algorithm 3 to obtain POD state basis $\mathbf{U} \in \mathbb{V}_q(\mathbb{R}^N)$ and POD time-evolution bases $\mathbf{V}_j \in \mathbb{V}_{N_{\text{train}}}(\mathbb{R}^m)$, $j \in \mathbb{N}(q)$.
2. Set the forecasting time-evolution bases equal to the POD time-evolution bases $\boldsymbol{\Xi}_j \leftarrow \mathbf{V}_j$, $j \in \mathbb{N}(q)$. Note that $\bar{N} = q$ and $a = N_{\text{train}}$.
3. Define the restriction and prolongation operators as $\mathbf{R} \leftarrow \mathbf{U}$ and $\mathbf{P} \leftarrow \mathbf{U}$, respectively.

This approach is sensible, as numerous studies have shown that POD tends to truncate solution modes associated with high-frequency spatial behavior; in many cases, these also associate with high-frequency temporal behavior [9]. Thus, in such cases, the resulting restriction operator will ensure that forecasting is applied only to the long-temporal-wavelength solution components. We note that this approach is equivalent to computing "tailored" temporal subspaces [15] via the sequentially truncated high-order SVD [39]. Section S5 of the supplementary materials demonstrates conditions under which the proposed approach yields an ideal predictive coarse propagator.

5.2. POD-based ROM.

Projection-based model reduction aims to reduce the cost of numerically solving (5.1) by reducing the dimensionality of the governing equations. To achieve this, these techniques employ a "trial basis" $\mathbf{\Phi} \in \mathbb{R}_\star^{N\times\hat{N}}$ with reduced state dimension $\hat{N} \leq N$ and subsequently approximate the state as $\tilde{\mathbf{x}}^\star : (t, \boldsymbol{\mu}) \mapsto \mathbf{x}^0(\boldsymbol{\mu}) + \mathbf{\Phi}\hat{\mathbf{x}}^\star(t,\boldsymbol{\mu})$. Here, $\mathbb{R}_\star^{m\times n}$ denotes the set of full-column-rank $m \times n$ real-valued matrices (i.e., the noncompact Stiefel manifold), and the reduced state $\hat{\mathbf{x}}^\star : \mathbb{T} \times \mathbb{R}^p \to \mathbb{R}^{\hat{N}}$ satisfies

$$\frac{d}{dt}\hat{\mathbf{x}}^\star(t,\boldsymbol{\mu}) = \hat{\mathbf{g}}\left(\hat{\mathbf{x}}^\star; t, \boldsymbol{\mu}\right), \quad \hat{\mathbf{x}}^\star(0,\boldsymbol{\mu}) = \mathbf{0}, \tag{5.2}$$

where $\hat{\mathbf{g}} : (\hat{\boldsymbol{\xi}}; t, \boldsymbol{\mu}) \mapsto (\mathbf{\Psi}(\hat{\boldsymbol{\xi}}; t, \boldsymbol{\mu})^T\mathbf{\Phi})^{-1}\mathbf{\Psi}(\hat{\boldsymbol{\xi}}; t, \boldsymbol{\mu})^T\mathbf{g}\left(\mathbf{x}^0(\boldsymbol{\mu}) + \mathbf{\Phi}\hat{\boldsymbol{\xi}}; t, \boldsymbol{\mu}\right)$ denotes the reduced velocity and $\mathbf{\Psi} : \mathbb{R}^{\hat{N}} \times \mathbb{T} \times \mathscr{D} \to \mathbb{R}_\star^{N\times\hat{N}}$ denotes the "test basis." Note that (5.2) enforces the ODE residual $\frac{d}{dt}\tilde{\mathbf{x}}^\star(t,\boldsymbol{\mu}) - \mathbf{g}\left(\tilde{\mathbf{x}}^\star(t,\boldsymbol{\mu}); t, \boldsymbol{\mu}\right)$ to be orthogonal to $\mathrm{Ran}(\mathbf{\Psi}\left(\hat{\mathbf{x}}; t, \boldsymbol{\mu}\right))$. The test basis can be set equal to the trial basis (i.e., $\mathbf{\Psi}\left(\hat{\mathbf{x}}; t, \boldsymbol{\mu}\right) = \mathbf{\Phi}$)—which is referred to as Galerkin projection—or can be chosen to minimize the discrete residual arising after time discretization (e.g., $\mathbf{\Psi}\left(\hat{\mathbf{x}}; t, \boldsymbol{\mu}\right) = [\alpha_0\mathbf{I} - h\beta_0\partial\mathbf{g}/\partial\boldsymbol{\xi}(\mathbf{x}^0 + \mathbf{\Phi}\hat{\mathbf{x}}; t, \boldsymbol{\mu})]\mathbf{\Phi}$ for linear multistep schemes, where α_0 and β_0 are coefficients for a given scheme), which is referred to as least-squares Petrov–Galerkin projection [10, 11, 9], for example. Again, we define $\hat{\mathbf{x}}(\cdot,\boldsymbol{\mu}) : t \mapsto \mathscr{F}(\hat{\mathbf{x}}^0(\boldsymbol{\mu}); 0, t)$ as the associated numerical solution with $\hat{\mathbf{x}}(\cdot,\boldsymbol{\mu}) \in (\mathscr{H})^{\hat{N}}$.

When the trial basis $\mathbf{\Phi}$ is computed via POD, both the trial basis and the proposed method's ingredients can be computed by executing the following steps:

1. Given training parameter instances $\{\bar{\boldsymbol{\mu}}_i\}_{i=1}^{N_{\mathrm{train}}} \subset \mathscr{D}$ and energy criterion $\upsilon \in [0,1]$, execute Algorithm 3 to obtain POD state basis $\mathbf{U} \in \mathbb{V}_q(\mathbb{R}^N)$ and POD time-evolution bases $\mathbf{V}_j \in \mathbb{V}_{N_{\mathrm{train}}}(\mathbb{R}^m)$, $j \in \mathbb{N}(q)$.
2. Set the trial basis equal to the POD state basis $\mathbf{\Phi} \leftarrow \mathbf{U}$; note that $\hat{N} = q$.
3. Set the forecasting time-evolution bases equal to the truncated POD time evolution bases such that only the first $\bar{N}$ (with $\bar{N} \leq q = \hat{N}$) POD modes are employed for forecasting: $\mathbf{\Xi}_j \leftarrow \mathbf{V}_j$, $j \in \mathbb{N}(\bar{N})$. Note that $a = N_{\mathrm{train}}$.
4. Define the restriction and prolongation operators as $\mathbf{R} \leftarrow [\mathbf{e}_1 \ \cdots \ \mathbf{e}_{\bar{N}}] \in \{0,1\}^{\hat{N}\times\bar{N}}$ and $\mathbf{P} \leftarrow [\mathbf{e}_1 \ \cdots \ \mathbf{e}_{\bar{N}}] \in \{0,1\}^{\hat{N}\times\bar{N}}$, respectively.

Remark 5.1 (negligible additional cost and effective use of right singular vectors). Steps 1–2 above are already required when the trial basis is computed via POD. Thus, in this case, the ingredients required for the proposed method can be obtained with negligible additional computational cost, as the dominant costs in Steps 1–4 above are incurred in Step 1. Thus, one can interpret the proposed methodology as providing a technique to *effectively use the right singular vectors*, which are already available for POD-based ROMs after computing the SVD in Step 4 of Algorithm 3.

Remark 5.2 (general ROMs). When the trial basis is not computed via POD, the approach described in section 5.1 can be employed, as the ROM ODE (5.2) has the same structure as the parameterized ODE (5.1). In this case, the snapshot collection required in Step 1 incurs a small computational cost, as Steps 1–3 of Algorithm 3 entail numerically solving only the ROM ODE (5.2) at parameter instances $\{\bar{\boldsymbol{\mu}}_i\}_{i=1}^{N_{\mathrm{train}}}$.

6. Numerical experiments.

This section compares the performance of several choices for parareal initialization and coarse propagation in the context of model reduction applied to a parameterized Burgers' equation. Here, the backward-Euler scheme is employed as the time integrator that defines the fine propagator; that is, we employ $\mathscr{F} \leftarrow \mathscr{F}_{\mathrm{BE}}$. In particular, we consider the following.

- Four methods for performing initialization in Step 2 of Algorithm 1:

(BE) the backward-Euler scheme ((3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{BE}}$), where the coarse propagator $\mathscr{G}_{\mathrm{BE}}$ is first-order accurate and implicitly satisfies

$$\mathscr{G}_{\mathrm{BE}}(\boldsymbol{\xi}; T^n, T^{n+1}) - \boldsymbol{\xi} - H\mathbf{g}\left(\mathscr{G}_{\mathrm{BE}}(\boldsymbol{\xi}; T^n, T^{n+1}); T^{n+1}\right) = 0,$$

(CN) the Crank–Nicolson scheme ((3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{CN}}$), where the coarse propagator $\mathscr{G}_{\mathrm{CN}}$ is second-order accurate and implicitly satisfies

$$\mathscr{G}_{\mathrm{CN}}(\boldsymbol{\xi}; T^n, T^{n+1}) - \boldsymbol{\xi} - \frac{1}{2}\left(H\mathbf{g}\left(\mathscr{G}_{\mathrm{CN}}(\boldsymbol{\xi}; T^n, T^{n+1}); T^{n+1}\right) + H\mathbf{g}\left(\boldsymbol{\xi}; T^n\right)\right) = 0,$$

(LF) local forecasting ((3.12) with $\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$), and

(GF) global forecasting (3.13).

- Three coarse propagators:

(BE) the backward-Euler scheme ($\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{BE}}$),

(CN) the Crank–Nicolson scheme ($\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{CN}}$), and

(LF) local forecasting ($\mathscr{G} \leftarrow \mathscr{G}_{\mathrm{LF}}$).

We refer to method *i-j* as the method where initialization is carried out with method i and coarse propagation with method j; for example, method GF-LF performs initialization using the global forecast and employs the local-forecasting coarse propagator.

6.1. Parameterized Burgers' equation and model reduction.

We now describe the parameterized Burgers' equation as described in [35], which corresponds to the following parameterized initial boundary value problem for $(x, \tau) \in [0, 100] \times [0, 25]$:

$$\frac{\partial u(x,\tau)}{\partial \tau} + \frac{1}{2}\frac{\partial\left(u^2(x,\tau)\right)}{\partial x} = 0.02e^{p_2 x} \tag{6.1}$$

with $u(0,\tau) = p_1 \ \forall \tau \in [0, 25]$, $u(x, 0) = 1 \ \forall x \in [0, 100]$, where the parameter domain corresponds to $\boldsymbol{\mu} = (p_1, p_2) \in \mathscr{D} = [1.5, 2.0] \times [0.02, 0.025]$.

After applying Godunov's scheme for spatial discretization with 500 control volumes, (6.1) and the boundary and initial conditions lead to a parameterized initial-value ODE problem consistent with (5.1) with $N = 500$. As described previously, we employ the backward-Euler scheme for time discretization using uniform fine time steps $h = 0.1$, which leads to $m = 250$ fine time instances. Unless stated otherwise, we set a parareal termination tolerance $\epsilon = 5 \times 10^{-3}$ in Algorithm 1.

We compare the time-parallel methods in the POD-based reduced-order-modeling context as discussed in section 5.2. Here, we employ $N_{\mathrm{train}} = 4$ randomly selected training points $\bar{\boldsymbol{\mu}}_1 = (1.5331, 0.0249)$, $\bar{\boldsymbol{\mu}}_2 = (1.6880, 0.0223)$, $\bar{\boldsymbol{\mu}}_3 = (1.9656, 0.0209)$, and $\bar{\boldsymbol{\mu}}_4 = (1.8000, 0.0232)$. We choose a reduced-state dimension[8] of $\hat{N} = q = 100$ and use the least-squares Petrov–Galerkin (LSPG) ROM [10], which—for the backward-Euler case—corresponds to a test basis of $\boldsymbol{\Psi}(\hat{\mathbf{x}}; t, \boldsymbol{\mu}) = [\mathbf{I} - h\partial\mathbf{g}/\partial\boldsymbol{\xi}(\mathbf{x}^0 + \boldsymbol{\Phi}\hat{\mathbf{x}}; t, \boldsymbol{\mu})]\boldsymbol{\Phi}$. Note that we do not consider hyperreduction, as this is not necessary to demonstrate

[8] Note that instead of specifying the energy criterion υ as suggested in section 5.2, we directly specify the reduced-state dimension $\hat{N}$.

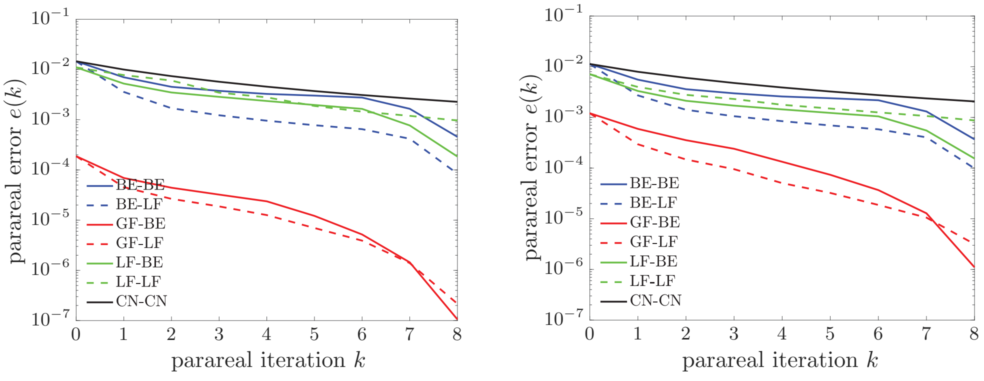
the merits of the proposed initialization technique and coarse propagator. During the experiments, we assess the performance of the ROMs and time-parallel methods at a set of $N_{\mathrm{online}} = 2$ randomly selected online parameter instances $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$ and $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$; that is, we numerically solve the reduced initial-value ODE problem (5.2) for $\boldsymbol{\mu} \in \{\boldsymbol{\mu}_i^\star\}_{i=1}^{N_{\mathrm{online}}}$. During the experiments, we vary the number of restricted states $\bar{N}$ and the forecast memory α.

6.2. Comparison of initialization and coarse-propagation methods. We first compare the performance of multiple combinations of initialization methods and coarse propagators. To achieve this, we set the number of coarse time instances to $M = 10$ and employ a parareal tolerance of $\epsilon = 0$; this ensures that the parareal method will execute (the maximum value of) $K = M - 1 = 9$ parareal iterations in Algorithm 1, thereby allowing us to analyze the complete convergence behavior of all methods. For both forecasting methods, we employ a memory of $\alpha = 8$; we employ a restricted-state dimension $\bar{N} = 8$ for the local forecast (i.e., we forecast only the first 8 POD modes) and a restricted-state dimension $\bar{N} = 50$ for the global forecast.

Figure 4 reports these results, where the time-parallel error at parareal iteration k is defined as

$$e(k) := \max_{n \in \{k+1,\dots,M-1\}} \|\mathscr{F}(\mathbf{x}_k^{n-1}; T^{n-1}, T^n) - \mathbf{x}_k^n\| / \|\mathscr{F}(\mathbf{x}_k^{n-1}; T^{n-1}, T^n)\|, \tag{6.2}$$

which is a measure of the normalized residual that the parareal method is aiming to set to zero [24].[9] The figure highlights two important trends. First, the results empirically support the theoretical result discussed in Remark 4.6: namely, the global-forecast initialization exhibits superior stability properties to the local-forecast initialization. In both online parameter instances, global-forecast initialization produces a very small initial error, while local-forecast initialization produces a larger initial error despite its use of the same time-evolution data; backward-Euler and Crank–Nicolson initialization produces a slightly smaller initial error than the local forecast. Second, note that the local-forecast propagator outperforms the backward-Euler coarse propagator when either the backward-Euler or global-forecasting initializations are employed.



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

FIG. 4. Comparison of initialization and coarse-propagation methods. *Convergence of six methods.*

To gain additional insight into the convergence properties of the methods, Figure S3 in section S6 of the supplementary materials reports the convergence of the error

[9] In Algorithm 1, this corresponds to the tolerance that appears in Step 6.

in the 51st entry of the state vector over parareal iterations for online parameter instance $\boldsymbol{\mu}_2^\star$. In the remainder of the numerical experiments, we limit our focus to the typical parareal methods BE-BE and CN-CN, as well as the most promising proposed data-driven method GF-LF.

6.3. Ideal case. This section assesses performance of the method under the "ideal case," i.e., when assumptions A3–A5 are satisfied as discussed in sections 4.1.2 and 4.2.2. Here, we ensure these conditions are met by repeating the training for each online point (i.e., $N_{\text{train}} = 1$ with both the training point set equal to the online point) and employing $\bar{N} = \hat{N}$ for both the local and global forecasts. Recall that under these conditions, the coarse propagator is exact (Theorem 4.3), and the GF-LF method should converge after parareal initialization (hence require $K = 0$ parareal iterations) and produce speedups given by (4.4) (Theorem 4.7). Note that while these conditions are "ideal" for the proposed methodology, they are not ideal for typical parareal methods BE-BE or CN-CN. We assess memories of $\alpha = 1, 2, 4, 6$ (set to be equal for the local and global forecasts) and employ a termination tolerance of $\epsilon = 5 \times 10^{-4}$ in this section only.

In the remaining experiments, we report the theoretical speedups derived in section 4.2 due to the lack of reliability in timings obtained with our MATLAB implementation.[10] Here, the speedup for method GF-LF is provided by (4.3), and the speedup for methods BE-BE and CN-CN[11] are provided by the following theorem, whose proof can be found in section S1 of the supplementary materials.

Theorem 6.1 (*speedup*: fine propagator as coarse propagator). *If the same time integrator is used for both the coarse and fine propagator and assumption* A8 *holds, then the parareal method realizes a speedup of*

$$S_{\mathit{fine}}(K) := \frac{m}{(M + \bar{m} - \frac{1}{2}K)(K+1)}. \tag{6.3}$$

Figures 5(a)–5(b) report a strong-scaling study wherein the number of parareal iterations required for convergence when the number of coarse time instances M increases (and the coarse time step $H = T_{\text{final}}/M$ undergoes an attendant decrease). As expected, in all cases, the proposed GF-LF method converges in the minimum number of parareal iterations (i.e., $K = 0$). In contrast, the BE-BE and CN-CN methods converge in the *worst-case* number of iterations (i.e., $K = M - 1$) for $M \leq 6$ in both cases; this occurs because these cases correspond to relatively large coarse time steps H, which degrades the accuracy of the backward-Euler and Crank–Nicolson schemes. The number of parareal iterations needed for convergence in the BE-BE and CN-CN cases decreases as the number of coarse time instances M increases; this can be attributed to the decreasing coarse time step H, which improves the accuracy of the time integrators.

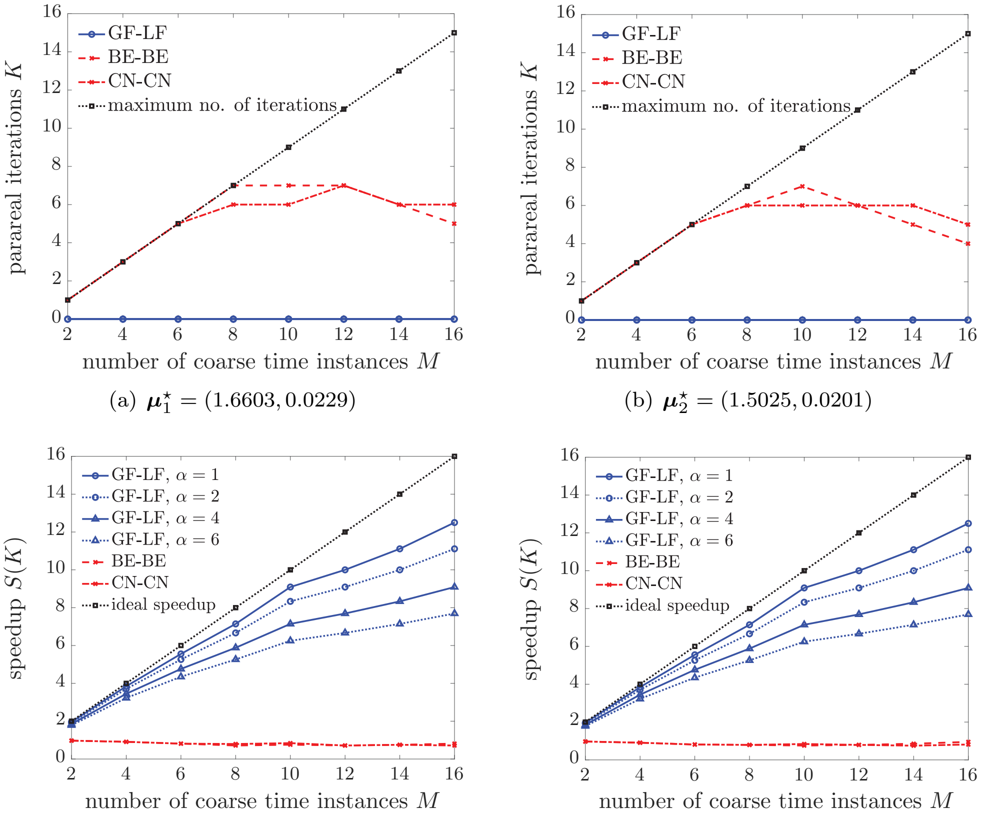
Figures 5(c)–5(d) report the theoretical speedups of these methods under these ideal conditions. Here, the reported values correspond to (6.3) for the BE-BE and CN-CN methods and to (4.3) for the GF-LF method. As expected, the proposed technique yields near-ideal theoretical speedups, while the typical approaches produce modest speedups due to their slow convergence on this difficult problem. Note that increasing

[10] Future work will entail implementation in a "production" computational-mechanics code and assessment of the method in a parallel computing environment.

[11] Because each method is characterized by only one implicit stage, we assume that the cost of Crank–Nicolson is the same as that of backward Euler; the additional explicit stage for Crank–Nicolson introduces negligible additional cost.

the memory degrades speedup in this case, as all values for the memory ensure an exact initial solution in the ideal case; thus, employing a small memory does not degrade convergence here.

Finally, Figure 6 reports parareal convergence for these methods for $M = 10$. As expected, the proposed GF-LF method produces a (near) zero error after initialization; on the other hand, the typical BE-BE and CN-CN methods exhibit relatively slow convergence. Figures 6(c)–6(d) show that only the proposed GF-LF method can satisfy the prescribed tolerance while yielding relative wall-times significantly less than 1.0.



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

(c) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

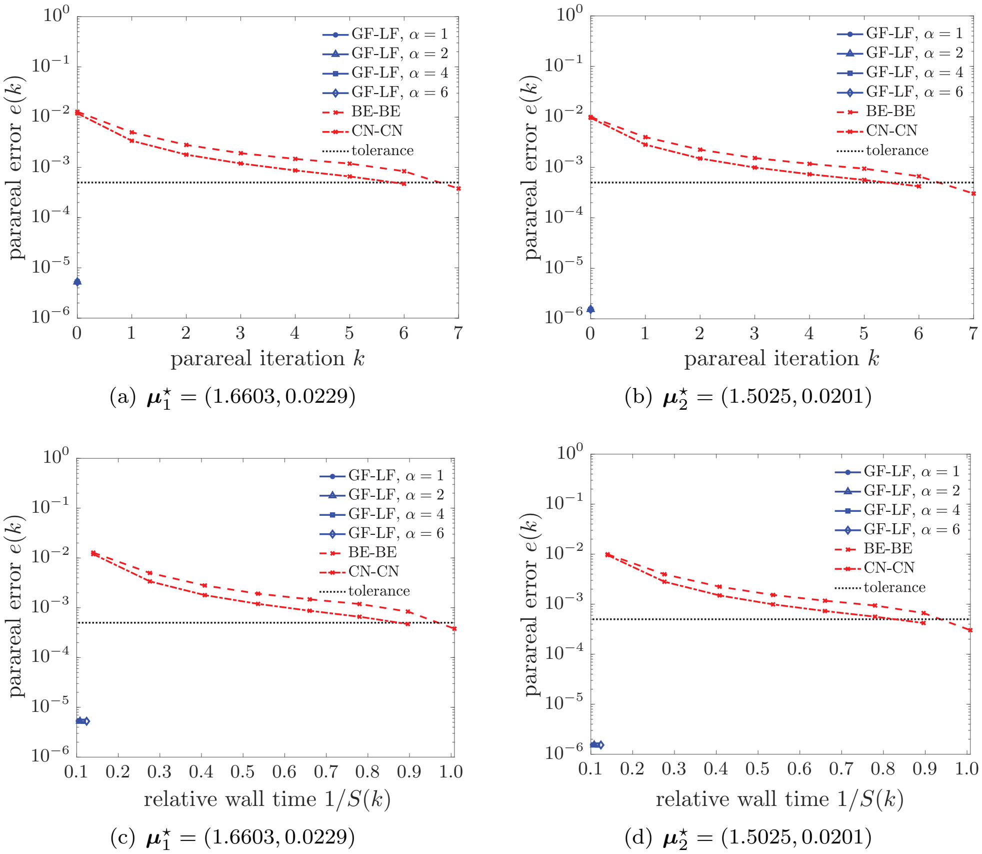
(d) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

FIG. 5. Ideal case. *Strong-scaling study reporting the number of parareal iterations K required for convergence and resulting speedups. Theoretical speedups are computed via* (6.3) *for BE-BE and CN-CN, and* (4.3) *for GF-LF, and convergence plots.*

6.4. Predictive case. We now return to the original problem setup with $N_{\text{train}} = 4$ training points and $N_{\text{online}} = 2$ online points. Here, the "ideal case" assumptions A3–A5 no longer hold. To assess the accuracy of the coarse propagator in this predictive scenario, Figure 7 reports the relative projection error

$$\varepsilon_j(\boldsymbol{\mu}) := \|(\mathbf{I}_m - \boldsymbol{\Xi}_j[\boldsymbol{\Xi}_j]^T)\mathbf{h}(\hat{x}_j(\cdot,\boldsymbol{\mu}))\| / \|\mathbf{h}(\hat{x}_j(\cdot,\boldsymbol{\mu}))\|, \tag{6.4}$$

$j \in \mathbb{N}(\hat{N})$, which measures the ability of the temporal bases $\boldsymbol{\Xi}_j$ to capture the time evolution of the reduced states. Note that this is a global variant of the quantity



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

(c) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(d) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

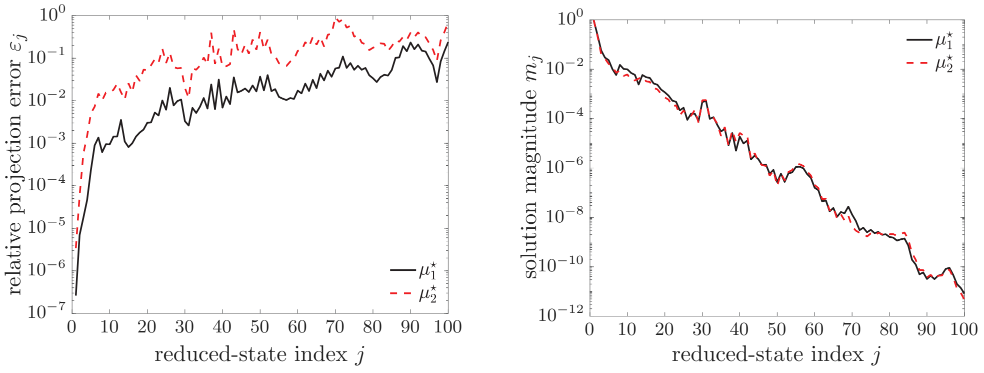
FIG. 6. Ideal case. *Convergence results for $M = 10$. Theoretical speedups computed via* (6.3) *for BE-BE and CN-CN, and* (4.3) *for GF-LF. Note that the bottom row of plots reports the same data as the top row, except the abscissa corresponds to the relative wall time rather than the parareal iteration.*

that appears in the coarse-propagator error bound in Theorem 4.1 and measures the extent to which assumption A3 is violated. Further, note that $\varepsilon_j = 0$, $j \in \mathbb{N}(\hat{N})$, for the ideal case. This figure also reports the relative magnitude of each reduced state

$$m_j(\boldsymbol{\mu}) := \|\mathbf{h}(\hat{x}_j(\cdot, \boldsymbol{\mu}))\| / \|\mathbf{h}(\hat{\mathbf{x}}(\cdot, \boldsymbol{\mu}))\|, \quad j \in \mathbb{N}(\hat{N}). \tag{6.5}$$

Figure 7 shows that the temporal bases are more accurate (i.e., yield smaller projection errors) for online point $\boldsymbol{\mu}_1^\star$ than for $\boldsymbol{\mu}_2^\star$; this suggests that the method should perform better (i.e., converge in fewer parareal iterations) for the first online point. Thus, we can interpret $\boldsymbol{\mu}_1^\star$ and $\boldsymbol{\mu}_2^\star$ as providing increasingly difficult scenarios for the proposed method in which the time-evolution bases are increasingly inaccurate. In addition, the figure shows an inverse relationship between the projection error and the solution magnitude. This is intuitive: the time-evolution bases are able to accurately capture the time-evolution of the dominant (low-index) reduced states, while the "noisy" (high-index) reduced states yield large projection errors. Section 6.5 explores this effect further.

Figure 8 reports a strong-scaling study, wherein the proposed method employs the parameter values employed in section 6.2; i.e., both forecasting methods employ



(a) Projection error ε_j, $j \in \mathbb{N}(\hat{N})$ (b) Solution magnitude m_j, $j \in \mathbb{N}(\hat{N})$

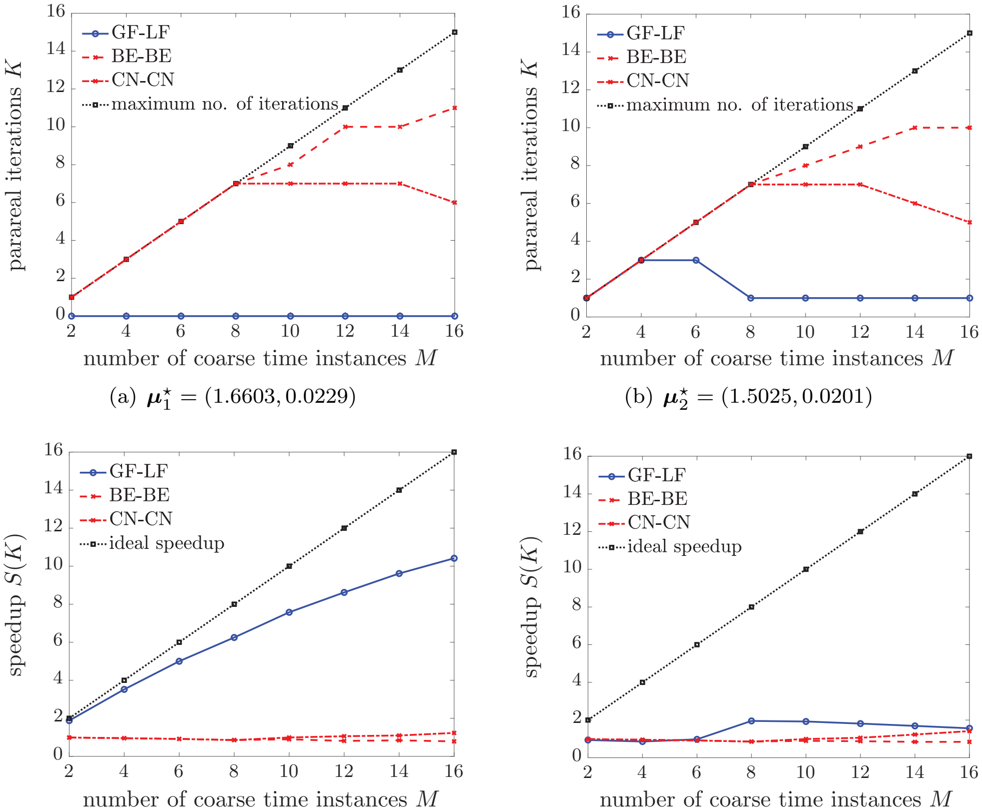
FIG. 7. Predictive case. *Projection error and solution magnitude for online points* $\boldsymbol{\mu}_1^\star$ *and* $\boldsymbol{\mu}_2^\star$.

a memory of $\alpha = 8$, the local forecast employs a restricted-state dimension $\bar{N} = 8$, and the global forecast employs a restricted-state dimension $\bar{N} = 50$. Figures 8(a)–8(b) report the dependence of the number of parareal iterations on the number of coarse time instances M for this case. Similar to the ideal case, the proposed GF-LF method converges in considerably fewer iterations than the BE-BE and CN-CN methods; in fact it converges in the minimum number of iterations $K = 0$ for $\boldsymbol{\mu}_1^\star$. Also, the proposed GF-LF method exhibits better performance for $\boldsymbol{\mu}_1^\star$ than $\boldsymbol{\mu}_2^\star$ as was suggested by the projection errors in Figure 7. As before, the BE-BE and CN-CN methods converge in the *worst-case* number of iterations (i.e., $K = M - 1$) for $M \leq 8$ for both online points. However, for $M \geq 9$, the CN-CN method converges in fewer iterations than the BE-BE method, likely due to its higher-order accuracy. Figures 8(c)–8(d) report the theoretical speedups of both methods under these ideal conditions. Again, the proposed technique yields better speedups compared with the typical methods, which is apparent for $\boldsymbol{\mu}_1^\star$ in particular.

Finally, Figure 9 reports parareal convergence for these methods for $M = 10$. The proposed GF-LF method produces a small error after initialization; for $\boldsymbol{\mu}_1^\star$, the error is smaller than the specified threshold for convergence. In contrast, the BE-BE and CN-CN methods exhibit relatively slow convergence with CN-CN converging faster, likely due to its higher-order accuracy. Figures 9(c)–9(d) show that only the proposed GF-LF method can satisfy the prescribed tolerance while yielding relative wall times significantly less than 1.0.

These promising results suggest that the proposed GF-LF method can deliver significant performance improvements over standard parareal techniques, even when ideal conditions do not hold. We note that numerical results obtained for $\hat{N} = 50$ (i.e., a less accurate ROM) reproduce exactly the results reported in Figure 8, which correspond to $\hat{N} = 100$. This reflects the fact that the proposed method's performance is not directly tied to the accuracy of the ROM; rather, it depends on the ability of the time-evolution bases to capture the time evolution of the reduced states as discussed above.

6.5. Parameter study. We now assess the dependence of the proposed GF-LF method on its parameters, namely the number of restricted variables $\bar{N}$ and the memory α. We again consider the original problem setup with $N_{\text{train}} = 4$ training points and $N_{\text{online}} = 2$ online points, wherein the "ideal case" assumptions A3–A5 do



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

(c) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

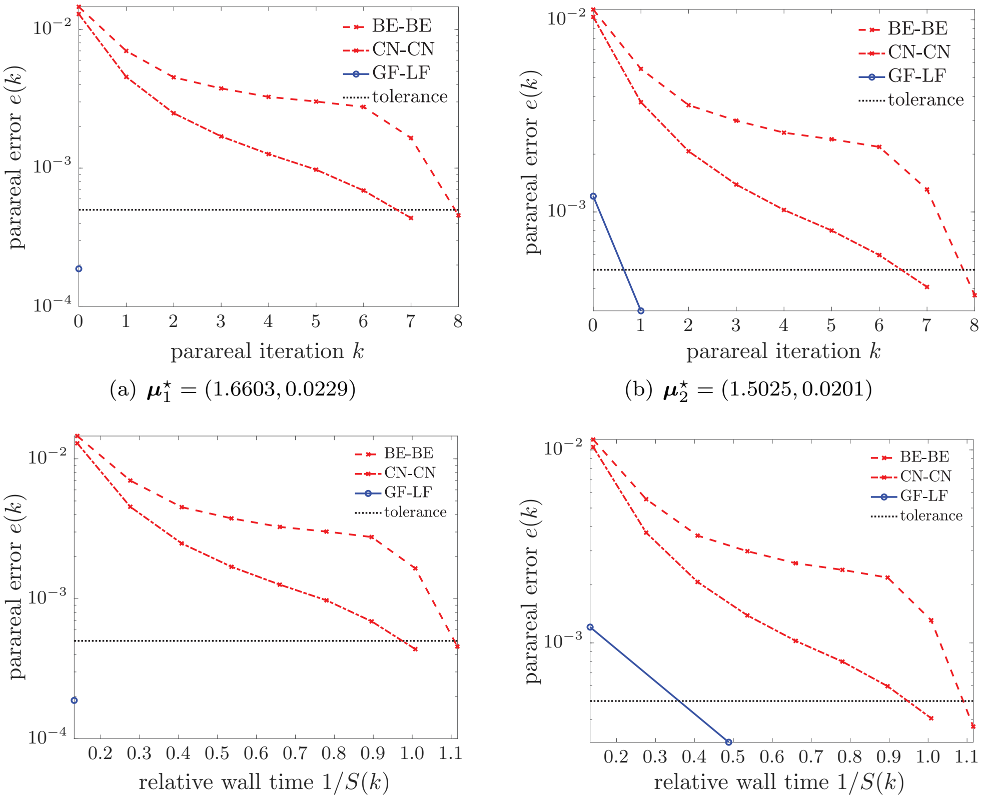
(d) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

FIG. 8. Predictive case. *Strong-scaling study reporting the number of parareal iterations K required for convergence and resulting speedups. Theoretical speedups are computed via* (6.3) *for BE-BE and CN-CN, and* (4.3) *for GF-LF, and convergence plots.*

not hold.

We first consider the effect of the restricted-state dimension $\bar{N}$ when performing global-forecast initialization. Table 1 reports these results; the memory is set to $\alpha = 8$. Note that for both online parameter instances, the parareal error after initialization $e(0)$ decreases monotonically as the restricted-state dimension $\bar{N}$ increases, and is constant to within three digits for $\bar{N} \geq 50$; this is an artifact of the intrinsic stability of the global forecast as discussed in Remark 4.6. Thus, in the remaining numerical experiments, we employ $\bar{N} = 50$ for the global forecast.

We next assess the effect of the restricted-state dimension $\bar{N}$ on the local forecast. Recall from Figure 7 that there is an inverse relationship between projection error and solution magnitude. In particular, low-index reduced states have large solution magnitudes and yield low projection errors; high-index reduced states comprise "noise" that cannot be accurately forecasted due to their high projection errors. To gain additional insight into this, Figure S4 in section S6 of the supplementary materials plots the global temporal bases $\boldsymbol{\Xi}_j$ associated with different (restricted) solution components. Figure 10 reports convergence of the method for $\alpha = 8$ for a range of values for $\bar{N}$. These results show precisely what we expect: the best performance is obtained (roughly) for an intermediate value of $\bar{N} = 8$.



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

(c) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(d) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

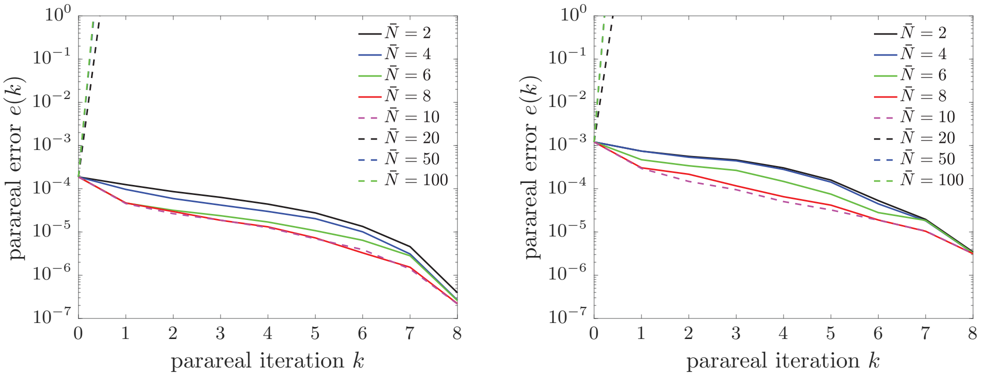
FIG. 9. Predictive case. *Convergence results for $M = 10$. Theoretical speedups computed via (6.3) for BE-BE and CN-CN, and (4.3) for GF-LF. Note that the bottom row of plots reports the same data as the top row, except the abscissa corresponds to the relative wall time rather than the parareal iteration.*

TABLE 1

Parameter study. *Value of the parareal error after global-forecast initialization $e(0)$ for different values of $\bar{N}$ employed by the global forecast. Here the memory is set to $\alpha = 8$.*

Restricted-state dimension $\bar{N}$	6	8	10	15	20	30	50	75	100
Initialization error $e(0) \times 10^4$ for $\boldsymbol{\mu}_1^\star$	251	188	111	46.7	8.57	3.88	1.88	1.88	1.88
Initialization error $e(0) \times 10^3$ for $\boldsymbol{\mu}_2^\star$	18.2	13.6	7.31	3.35	1.27	1.23	1.21	1.21	1.21

Next, Figure 11 reports performance of the method for fixed values of $\bar{N} = 8$ for the local forecast and $\bar{N} = 50$ for the global forecast, and for a range of values for the memory α, which is set to be equal for the local and global forecasts. First, note that interpolation, which corresponds to $\alpha = a = 4$, yields the worst performance in terms of error at a given iteration. This supports the theoretical results discussed in Remark 4.1: oversampling (i.e., employing $\alpha > a$) produces a stabilizing effect. In this case, the value of the memory leading to best overall performance (in terms of accuracy) is $\alpha = 8$. Note that employing the smallest value for the memory yields



(a) $\boldsymbol{\mu}_1^{\star} = (1.6603, 0.0229)$ (b) $\boldsymbol{\mu}_2^{\star} = (1.5025, 0.0201)$

Fig. 10. Parameter study. *Convergence plots for the GF-LF method for a range of values for $\bar{N}$ employed by the local forecast, a memory of $\alpha = 8$, and a restricted-state dimension of $\bar{N} = 50$ for the global forecast. Note that a value of $\bar{N} = 8$ for the local forecast yields roughly the best overall performance.*

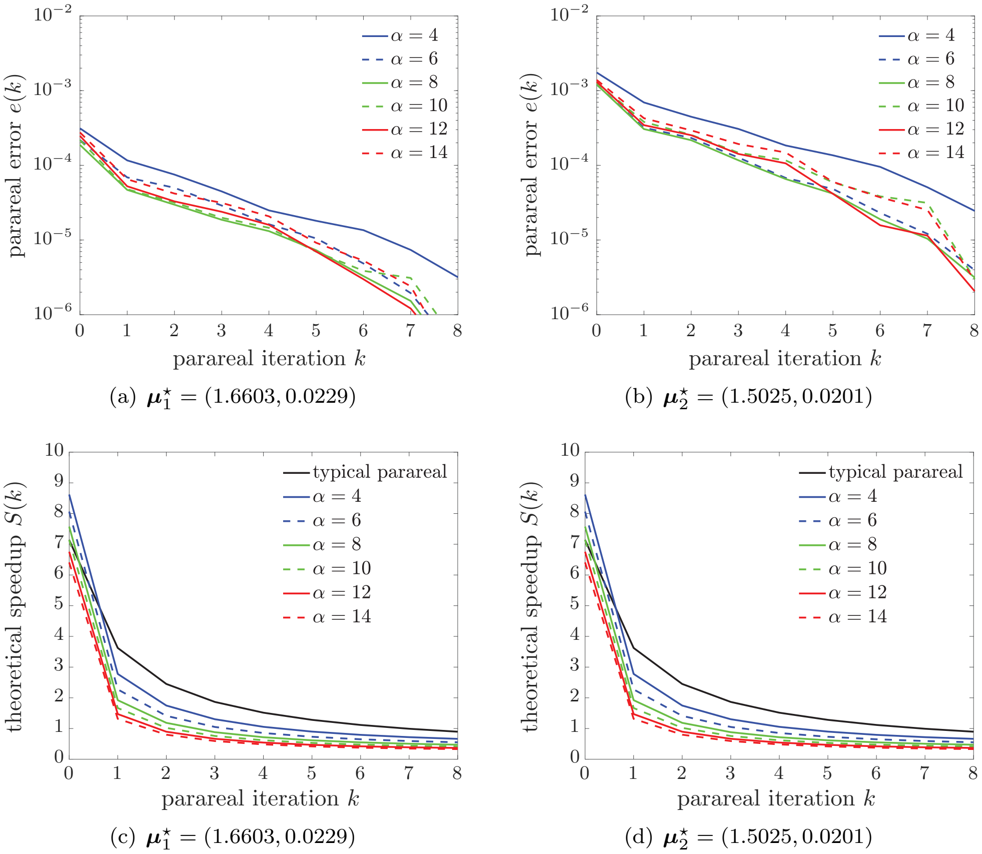
the best theoretical speedups if the method were to converge in the same number of parareal iterations for all values of the memory. This illustrates the tradeoff discussed in Remark 4.3: increasing the memory α reduces the speedup for a fixed number of iterations needed for convergence; yet, doing so can also decrease the bound for the error between coarse and fine propagators, which promotes convergence.

7. Conclusions. This work presented a novel data-driven method for time parallelism. We applied both local and global forecasting to define initialization methods, as well as local forecasting to define the coarse propagator. We performed analysis demonstrating the method's accuracy, speedup, and stability. Key theoretical results include the following:

- The error between the local-forecast coarse propagator and the fine propagator can be bounded by a readily interpretable quantity (Theorem 4.1).
- Ideal conditions exist under which the local-forecast coarse propagator is equal to the fine propagator (Theorem 4.3).
- The parareal recurrence is stable with the local-forecast coarse propagator (Theorem 4.11) with constants that are independent of the time discretization (Remark 4.5).
- Existing convergence results for the parareal recurrence hold with the proposed coarse propagator, and superlinear convergence can be obtained under certain conditions (Corollary 4.12).

Key results corroborated by both theoretical analysis and numerical experiments include the following:

- Global-forecast initialization is more stable (Remark 4.6) and produces a more accurate solution (Figure 4) than the local-forecast initialization.
- Local-forecast coarse propagation is nearly always more accurate than backward-Euler coarse propagation, regardless of initialization (Figure 4).
- Under ideal conditions, the proposed method converges after parareal initialization, i.e., for $K = 0$ in Algorithm 1 (Theorems 4.6–4.7 and Figures 5(a)–5(b)), and can realize near-ideal speedups (Figures 3 and 5(c)–5(d)).
- Increasing the memory α can improve coarse-propagation accuracy but incurs



(a) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(b) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

(c) $\boldsymbol{\mu}_1^\star = (1.6603, 0.0229)$

(d) $\boldsymbol{\mu}_2^\star = (1.5025, 0.0201)$

Fig. 11. Parameter study. *Convergence plots for the GF-LF method for a range of values for α and values of $\bar{N} = 8$ for the local forecast and $\bar{N} = 50$ for the global forecast. Note that a value of $\alpha = 8$ yields the best overall performance. Theoretical speedups for typical parareal correspond to $S_{\mathit{fine}}(k)$ defined in (6.3), while speedups for the proposed GF-LF method correspond to $S_{\text{GF-LF}}(k)$ defined in (4.3), and are assessed for a range of values for α and a value of $\bar{N} = 8$.*

a larger cost (Remark 4.1 and Figure 11).

- Increasing the number of variables included in the local forecast $\bar{N}$ has two competing effects: it can improve the forecast accuracy but can incur error if the additional variables are difficult to forecast, e.g., associate with high-frequency temporal content (Remark 4.2 and Figures 7 and 10).

Finally, numerical experiments show that in all (predictive) cases where ideal conditions do not hold, global-forecast initialization and local-forecast coarse propagation outperform backward-Euler initialization and coarse propagation (Figures 4 and 8).

Future work involves applying the proposed method in parallel computing environments with realistic timings, applying the method to parameterized full-order ODEs (i.e., not reduced-order models), and assessing the viability of alternative data sources to produce low-dimensional time-evolution bases.

Acknowledgments. Kevin Carlberg acknowledges Julien Cortial for insightful initial conversations, as well as Yvon Maday for providing useful feedback.

REFERENCES

[1] A. Axelsson and J. Verwer, *Boundary value techniques for initial value problems in ordinary differential equations*, Math. Comp., 45 (1985), pp. 153–171.

[2] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah, *Parallel-in-time molecular-dynamics simulations*, Phys. Rev. E, 66 (2002), 057701.

[3] G. Bal, *On the convergence and the stability of the parareal algorithm to solve partial differential equations*, in Domain Decomposition Methods in Science and Engineering, Springer, New York, 2005, pp. 425–432.

[4] G. Bal and Y. Maday, *A "parareal" time discretization for non-linear PDEs with application to the pricing of an American put*, in Recent Developments in Domain Decomposition Methods, Springer, New York, 2002, pp. 189–202.

[5] 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. Acad. Sci. Paris, 339 (2004), pp. 667–672.

[6] A. Bellen and M. Zennaro, *Parallel algorithms for initial-value problems for difference and differential equations*, J. Comput. Appl. Math., 25 (1989), pp. 341–350.

[7] A. Blouza, L. Boudin, and S. M. Kaber, *Parallel in time algorithms with reduction methods for solving chemical kinetics*, Commun. Appl. Math. Comput. Sci., 5 (2011), pp. 241–263.

[8] K. Carlberg, *Model Reduction of Nonlinear Mechanical Systems via Optimal Projection and Tensor Approximation*, Ph.D. thesis, Stanford University, Stanford, CA, 2011.

[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. Farhat, and C. Bou-Mosleh, *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] K. Carlberg, J. Ray, and B. van Bloemen Waanders, *Decreasing the temporal complexity for nonlinear, implicit reduced-order models by forecasting*, Comput. Methods Appl. Mech. Engrg., 289 (2015), pp. 79–103.

[13] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764, https://doi.org/10.1137/090766498.

[14] F. Chen, J. S. Hesthaven, and X. Zhu, *On the use of reduced basis methods to accelerate and stabilize the parareal method*, in Reduced Order Methods for Modeling and Computational Reduction, Springer, New York, 2014, pp. 187–214.

[15] Y. Choi and K. Carlberg, *Space–Time Least-Squares Petrov–Galerkin Projection for Nonlinear Model Reduction*, preprint, https://arxiv.org/abs/1703.04560, 2017.

[16] M. Drohmann, B. Haasdonk, and M. Ohlberger, *Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation*, SIAM J. Sci. Comput., 34 (2012), pp. A937–A969, https://doi.org/10.1137/10081157X.

[17] R. Everson and L. Sirovich, *Karhunen–Loève procedure for gappy data*, J. Optical Soc. Amer. A, 12 (1995), pp. 1657–1664.

[18] R. D. Falgout, S. Freidhoff, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder, *Parallel time integration with multigrid*, SIAM J. Sci. Comput., 36 (2014), pp. C635–C661, https://doi.org/10.1137/130944230.

[19] C. Farhat and M. Chandesris, *Time-decomposed parallel time-integrators: Theory and feasibility studies for fluid, structure, and fluid–structure applications*, Internat. J. Numer. Methods Engrg., 58 (2003), pp. 1397–1434.

[20] C. Farhat, J. Cortial, C. Dastillung, and H. Bavestrello, *Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses*, Internat. J. Numer. Methods Engrg., 67 (2006), pp. 697–724.

[21] P. F. Fischer, F. Hecht, and Y. Maday, *A parareal in time semi-implicit approximation of the Navier–Stokes equations*, in Domain Decomposition Methods in Science and Engineering, Springer, New York, 2005, pp. 433–440.

[22] M. J. Gander, *Overlapping Schwarz for parabolic problems*, in Proceedings of the 9th International Conference on Domain Decomposition Methods (Boulder, CO, 1997), AMS, Providence, RI, 1998, pp. 425–431.

[23] M. J. Gander and E. Hairer, *Nonlinear convergence analysis for the parareal algorithm*, in Domain Decomposition Methods in Science and Engineering XVII, Springer, Berlin, 2008, pp. 45–56.

[24] M. J. Gander and S. Vandewalle, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput., 29 (2007), pp. 556–578, https://doi.org/10.1137/05064607X.

[25] D. Guibert and D. Tromeur-Dervout, *Adaptive parareal for systems of ODEs*, in Domain Decomposition Methods in Science and Engineering XVI, Springer, Berlin, 2007, pp. 587–594.

[26] W. Hackbusch, *Parabolic multigrid methods*, in Computing Methods in Applied Sciences and Engineering, VI (Versailles, France, 1983), R. Glowinski and J. Lions, eds., North–Holland, Amsterdam, 1984, pp. 189–197.

[27] J. Lions, Y. Maday, and G. Turinici, *A "parareal" in time discretization of PDE's*, C. R. Acad. Sci. Paris Sér. I Math., 332 (2001), pp. 661–668 (in French).

[28] C. Lubich and A. Ostermann, *Multigrid dynamic iteration for parabolic equations*, BIT, 27 (1987), pp. 216–234.

[29] Y. Maday and E. M. Rønquist, *Parallelization in time through tensor-product space–time solvers*, C. R. Math. Acad. Sci. Paris, 346 (2008), pp. 113–118.

[30] Y. Maday and G. Turinici, *Parallel in time algorithms for quantum control: Parareal time discretization scheme*, Internat. J. Quantum Chem., 93 (2003), pp. 223–228.

[31] M. Minion, *A hybrid parareal spectral deferred corrections method*, Commun. Appl. Math. Comput. Sci., 5 (2011), pp. 265–301.

[32] W. L. Miranker and W. Liniger, *Parallel methods for the numerical integration of ordinary differential equations*, Math. Comp., 21 (1967), pp. 303–320.

[33] A. S. Nielsen, *Feasibility Study of the Parareal Algorithm*, Masters thesis, Technical University of Denmark, DTU Informatic, Kongens Lyngby, Denmark, 2012.

[34] J. Nievergelt, *Parallel methods for integrating ordinary differential equations*, Comm. ACM, 7 (1964), pp. 731–733.

[35] M. J. Rewienski, *A Trajectory Piecewise-Linear Approach to Model Order Reduction of Nonlinear Dynamical Systems*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[36] P. Saha, J. Stadel, and S. Tremaine, *A parallel integration method for solar system dynamics*, Astron. J., 114 (1997), pp. 409–415.

[37] D. A. D. Subcommittee, *Synergistic Challenges in Data-Intensive Science and Exascale Computing*, DOE Office of Science, Washington, DC, 2013.

[38] S. Vandewalle, *Parallel Multigrid Waveform Relaxation for Parabolic Problems*, Vieweg+Teubner Verlag, Berlin, 1993.

[39] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, *A new truncation strategy for the higher-order singular value decomposition*, SIAM J. Sci. Comput., 34 (2012), pp. A1027–A1052, https://doi.org/10.1137/110836067.

[40] D. E. Womble, *A time-stepping algorithm for parallel computers*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 824–837, https://doi.org/10.1137/0911049.