Journal of Computational Physics 405 (2020) 109156

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A stable parareal-like method for the second order wave equation

Hieu Nguyen [a,*], Richard Tsai [b,c]

[a] Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, TX 78712, USA
[b] Department of Mathematics and Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, TX 78712, USA
[c] KTH Royal Institute of Technology, Sweden

ARTICLE INFO

Article history:
Received 3 May 2019
Received in revised form 12 October 2019
Accepted 27 November 2019
Available online 4 December 2019

Keywords:
Parallel-in-time
Wave equation
Procrustes problem

ABSTRACT

A new parallel-in-time iterative method is proposed for solving the homogeneous second-order wave equation. The new method involves a coarse scale propagator, allowing for larger time steps, and a fine scale propagator which fully resolves the medium using finer spatial grid and uses shorter time steps. The fine scale propagator is run in parallel for short time intervals. The two propagators are coupled in an iterative way that resembles the standard parareal method [24]. We present a data-driven strategy in which the computed data gathered from each iteration are re-used to stabilize the coupling by minimizing the wave energy residual of the fine and coarse propagated solutions. Several examples, including a wave speed with discontinuities, are provided to demonstrate the effectiveness of the proposed method.

1. Introduction

In this paper, we will focus on the initial value problem of the standard second order wave equation:

$$\begin{aligned} &u_{tt} = c^2(x)\Delta u, \quad x \in [0,1)^d, \ 0 \le t < T, \\ &u(x,0) = u_0(x), \\ &u_t(x,0) = p_0(x). \end{aligned} \tag{1}$$

For boundary conditions, we consider either periodic, absorbing boundary conditions or placing a perfectly matched layer around $[0,1)^d$. The wave speed $c(x)$ is given explicitly and independent of the solution. Our objective is to develop a stable parallel-in-time algorithm for (1).

The wave equation is a physical model for seismic wave and electromagnetic wave in certain simplified setups. It is also used as a test case for developing algorithms that are further generalized to more complicated elastic and electromagnetic wave equations.

Time domain decomposition methods for evolution problems has been of increasing interest in the partial differential equation community due to the increasing number of cores available in modern supercomputers. Despite rapid advance in parallel computer architecture, parallelizing the time evolution of the second order wave equation efficiently is still a

* Corresponding author.
E-mail addresses: hieu@ices.utexas.edu (H. Nguyen), ytsai@math.utexas.edu (R. Tsai).

https://doi.org/10.1016/j.jcp.2019.109156
0021-9991/

challenging problem. One of the time domain decomposition paradigms is parallel-in-time method. The whole time domain $[0, T)$ is partitioned into subintervals for parallel processing. The most relevant algorithm to this paper is the parareal method introduced by Lions, Maday and Turinici [24]. The parareal method combines iteratively two propagators, denoted by $\mathcal{F}v$ the fine propagator and by $\mathcal{C}v$ the coarse propagator. They approximate the solution $v(t_{n+1})$ propagated from $v(t_n)$. The approximate solution at parareal iteration k, denoted as v_n^k, can be described by

$$\begin{aligned} &v_{n+1}^{k+1} = \mathcal{C}v_n^{k+1} + \mathcal{F}v_n^k - \mathcal{C}v_n^k, \\ &v_0^1 = v_0, \quad v_{n+1}^1 = \mathcal{C}v_n^1, \quad k = 1, 2, \ldots, \quad n = 0, 1, \ldots, N. \end{aligned} \tag{2}$$

Note that for each k, $\mathcal{F}v_n^k$ is computed in parallel. For the second order wave equation under consideration, $v(x, t)$ is a vector corresponding to $[u(x, t), u_t(x, t)]$.

Typically, the coarse propagator runs on coarse grid and is cheaper to compute, while the fine propagator runs on finer grid and is assumed to fully resolve the small scales in the problem. The finer propagator is thus more costly to compute.

In [4], it is shown that the parareal method is stable and converges linearly to the serial fine solution if the coarse propagator is smooth and has sufficient dissipation. When certain conditions are met, the parareal method can achieve high fidelity solution within few iterations. Some applications of the parareal method are: plasma turbulence in Tokamak reactor [38,37,36], Navier-Stoke equations [14,40,11], acoustic wave [28], shallow water [20], chemical kinetic [6], molecular dynamics [9], reaction wave [13], neutron diffusion [5,26], lattice Boltzmann equation for laminar flow [29,23,30].

Speed up of wall-clock time is attained when the coarse propagator can be chosen as a spatial coarsening of the fine propagator [35,33] which allows larger coarse time step. Indeed, this coarsening technique provides additional speed up in some applications [25,3,23] because the coarse propagator has less grid points to compute, provided an appropriate grid restriction and interpolation operator. However as shown in [33], considerable coarse grid resolution and accurate interpolation are required in order to make the parareal iteration (2) converge.

The parareal method tends to suffer from slow convergence or instability when applied to hyperbolic problems. Using an oscillatory dynamical system as an example, [1,2,22] pointed out that the phase error between the coarse and fine propagators is the reason for the slow convergence. Analogously for advection problems, the authors in [34] observed that numerical dispersion between the solvers makes the parareal method converge from above and hence causes instability. Intuitively, constructive or destructive interference of two overlapping plane waves depends on their relative phase which is sensitive to the frequency, yet the parareal iterative coupling (2) is point-wise in space and time.

There have been some attempts to modify the classical parareal method in order to address the slow convergence issue. In [12], the fact that solutions to the wave equation live on a submanifold of constant energy is exploited. In that work, the solutions are projected onto the submanifold to stabilize the parareal iterations. More precisely the algorithm can be presented as

$$u_n^k = P[\mathcal{C}u_{n-1}^k + (\mathcal{F}u_{n-1}^{k-1} - (\mathcal{C}u_{n-1}^{k-1}))],$$

where P denotes the projection onto the constant energy submanifold. However, the projection is obtained by solving nonlinear equations which can be sensitive to the initial guess.

In the so-called Krylov-subspace enhanced parareal methods [15,35], computed solution data is used to construct projection operators, which is used to modify the coarse propagator. To get the projection operator P^k, a set of orthogonal vectors is constructed for the subspace spanned by $(\{u_i^j\}$ for $i = 1, 2 \ldots n, j = 1, 2 \ldots k-1)$. Let S^k be the matrix whose columns are the orthogonal vectors s_ℓ, then $P^k = S^k(S^k)^T$. The enhanced parareal algorithm takes the following form:

$$u_n^k = (\mathcal{C}(I - P^k)u_{n-1}^k + \mathcal{F}P^k u_{n-1}^k) + \mathcal{F}u_{n-1}^{k-1} - (\mathcal{C}(I - P^k)u_{n-1}^{k-1} + \mathcal{F}P^k u_{n-1}^{k-1}),$$

where I is the identity. The enhanced coarse propagator corresponds to

$$\mathcal{C}(I - P^k) + \mathcal{F}P^k.$$

The fine propagation, $\mathcal{F}s_\ell$ for s_ℓ is the orthogonal vector that defines P^k, is precomputed and stored. The precomputation incurs an additional computing cost on top of orthogonalization of the data matrices.

The reduced basis parareal method [10] develops more efficient ways to construct the basis vectors and extends the approach to solve nonlinear equations.

The convergence and stability of these methods are analyzed and demonstrated by numerical examples of constant wave speed media in one and two dimension. However, in these work, the fine and coarse solvers are assumed to work on the same spatial grids and examples of variable wave speed are not presented. In this paper, we will consider the solvers on different spatial grids and present examples with variable wave speed. We also use the computed data, but its usage is very different from [15,35,10], see Section 3.2.

On the other hand, it is known that the slow convergence and instability of the parareal method for hyperbolic problems can be due to some notions of phase errors [2,1] and numerical dispersion [34]. In [1], effective multiscale parareal schemes relying on elaborate phase correction are proposed for a class of highly oscillatory dynamical systems. In [2], we derived convergence theory for a modified parareal scheme applying to linear systems of ordinary differential equations (ODEs).

Additionally, in that work, we investigated a few simple strategies of phase correction systematically and showed that appropriate phase correction could enable the resulting scheme to have superior performance.

In this paper, we propose a new method, based on the idea of θ-parareal scheme [2]. Instead of decomposing the input data as in [15,35], we use the computed data to build an operator, formally denoted as θ, that directly brings the coarse solutions, $\mathcal{C}u$, closer to the fine solutions, $\mathcal{F}u$. In this paper, the θ operators are constructed by minimizing the residual between the fine and coarse solutions in a semi-norm related to the discrete wave energy.

2. Preliminary background

We briefly review the plain parareal method and its properties. In a context of linear evolutionary problem $\dot{u}(t) = Au(t)$ for $t \in \{0, \Delta t, \ldots N\Delta t = T\}$ and $A : \mathbb{R} \mapsto \mathbb{R}$ linear function, let us denote the fine propagator/solver $\mathcal{F}u_n \mapsto u_{n+1}$ and the coarse propagator/solver $\mathcal{C}u_n \mapsto u_{n+1}$. Then the plain parareal iteration $k+1$ can be written as a recurrence relation

$$u_{n+1}^{k+1} = \mathcal{C}u_n^{k+1} + \mathcal{F}u_n^k - \mathcal{C}u_n^k. \tag{3}$$

Starting solution $k = 1$ is the serial coarse solution $u_n^{k=1} = \mathcal{C}^n u_0$. In addition, by rewriting the recurrence relation (3) in a matrix form and manipulating the inverse of Toeplitz structure, an error estimate $e_n^k := |u_n^k - u(t_n)|$ is derived in [2]

$$e_n^{k+1} \leq \|\mathcal{F} - \mathcal{C}\|_\infty \sum_{i=1}^{n-k-1} \|\mathcal{C}\|_\infty^i e_n^k. \tag{4}$$

The first term on the right hand side is equivalent to the local truncation error of the coarse propagator, assuming the fine solver is an exact one. The summation term is bounded above by N for stable schemes, e.g. $\|\mathcal{C}\|_\infty \leq 1$. Above error estimate is equivalent to linear convergence analysis of the parareal method derived in [4,16].

Wall-clock complexity of the parareal algorithm is estimated by

$$C_{parareal} = K\Big(\frac{T}{\Delta t} + \frac{T}{n_{CPU}\delta t}\Big). \tag{5}$$

Comparing to the complexity of the serial fine solver $C_{fine} = T/\delta t$, the parareal algorithm is more effective (from the perspective of total wall-clock computing time) if (i) a large number of computing cores, n_{CPU}, are used; (ii) the coarse/fine time stepping ratio is sufficiently large $\Delta t/\delta t \gg 1$; and (iii) the number of iterations, needed to for the desired accuracy K, is small.

The key objective of this paper is to introduce a data-driven strategy to stabilize and improve the efficiency of the parareal iteration.

3. The proposed method

We propose a scheme that takes the general form:

$$\mathbf{u}_{n+1}^{k+1} = \theta_{n+1}^k[\tilde{\mathcal{C}}\mathbf{u}_n^{k+1}] + \tilde{\mathcal{F}}\mathbf{u}_n^k - \theta_{n+1}^k[\tilde{\mathcal{C}}\mathbf{u}_n^k]. \tag{6}$$

Here, $\mathbf{u}_n^k$ denotes the solutions computed on the grid, and it has two component blocks, one corresponds to the wave solution u and the other the time derivative u_t. In this paper, for readability we shall also use $\dot{u}$ to denote the time derivative of u, i.e. $u_t \equiv \dot{u}$. The coarse and fine propagators, $\mathcal{C}$ and $\mathcal{F}$ will operate on different grids, and additional interpolation and restriction operators are needed for coupling the two propagators. Here we use $\tilde{\mathcal{C}}$ and $\tilde{\mathcal{F}}$ to denote the appropriately defined operations to be described in detail in this section.

A family of operators $\theta_n^k[\cdot]$ are constructed such that

$$\theta_{n+1}^k \approx \tilde{\mathcal{F}}\tilde{\mathcal{C}}^{-1} : \tilde{\mathcal{C}}\mathbf{u} \mapsto \tilde{\mathcal{F}}\mathbf{u}.$$

Clearly, direct calculation of $\tilde{\mathcal{F}}\tilde{\mathcal{C}}^{-1}$ is not practical because it undermines time parallelization of the θ-parareal method. Instead, we seek an effective mapping that has similar property as $\tilde{\mathcal{F}}\tilde{\mathcal{C}}^{-1}$ and is constructed from data computed along the parareal iterations.

3.1. Discretizations and data preparation

In this paper, we use the uniform Cartesian grids for the spatial domain and uniform stepping in time. Both the coarse and the fine propagators are defined by the standard second order central difference scheme for the spatial derivatives and velocity Verlet for time marching. The coarse propagator will operate on the coarse grid: $\Delta x \cdot \mathbb{Z}^d \times \Delta t \cdot \mathbb{Z}^+$, and the fine propagator will operate on the fine grid: $\delta x \cdot \mathbb{Z}^d \times \delta t \cdot \mathbb{Z}^+$, for $d = 1$ or 2.

Let $u_n \in \mathbb{R}^{N_{\delta x}}, U_n \in \mathbb{R}^{N_{\Delta x}}$ denote respectively the solutions computed at time $t_n = n\Delta t_{com}$, $n = 1, 2, 3, \ldots N$ on the fine and coarse grids. $N_{\delta x}, N_{\Delta x}$ are the number of grid points for the fine grids and the coarse grids respectively.

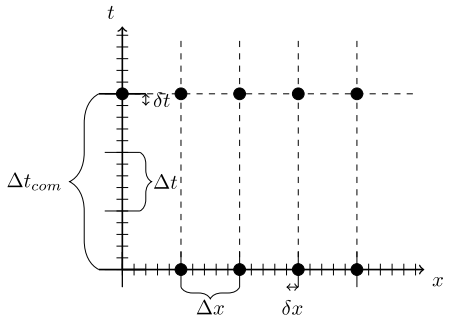

Fig. 1. Discretization diagram. Coarse propagator, $\mathcal{C}$ uses spatial grid size Δx and temporal step size Δt. Fine propagator, $\mathcal{F}$ uses spatial grid size δx and temporal step size δt. These propagators communicate at $(j\Delta x, n\Delta t_{com})$ for $j \in \mathbb{Z}$, $n = 1, 2, 3, \ldots, N$.

These fine grid functions $u \in \mathbb{R}^{N_{\delta x}}$ and coarse grid functions $U \in \mathbb{R}^{N_{\Delta x}}$ are coupled by an interpolation $\mathcal{I} : U \mapsto u$ and a restriction $\mathcal{R} : u \mapsto U$. The accuracy of the interpolation method will influence the stability of parareal iteration, as discussed in Section 6.4. Coarse propagator uses point-wise value of the wave speed $c(j\Delta x)$ and does not involve averaging of the wave speed nor homogenization of the wave equation. The fine and coarse propagators communicate at $n\Delta t_{com}$. The fine propagator uses the step size $\delta t = \Delta t_{com}/m_{\mathcal{F}}$ and the coarse propagator uses $\Delta t = \Delta t_{com}/m_{\mathcal{C}}$, with $m_{\mathcal{F}}, m_{\mathcal{C}} \in \mathbb{N}$ selected according to δx and Δx for stability in the respective time stepping. See Fig. 1.

Given $[u^k_{n-1}, \dot{u}^k_{n-1}]$ at t_{n-1}, the fine and coarse propagators are applied to obtain the solutions to define

$$[u_n, \dot{u}_n] := \mathcal{F}[u^k_{n-1}, \dot{u}^k_{n-1}]$$

and

$$[U_n, \dot{U}_n] := \mathcal{C}[\mathcal{R}u^k_{n-1}, \mathcal{R}\dot{u}^k_{n-1}].$$

For readability, we will write in-line vector $[v, w]$ and full vector

$$\begin{bmatrix} v \\ w \end{bmatrix}$$

interchangeably. These solutions are propagated over a coupling time interval $[t_{n-1}, t_n)$. These propagators are expected to approximately preserve the wave energy.

Finally, we will quickly describe the data matrices that will be used to construct the operators θ^k_n. We are interested in using the computed solution data, particularly the gradient of the wavefield u and a weighted momentum of $\dot{u}$. Each column of data matrices is formed by block(s) of the gradients ∇U_n followed by a block of momentum $\dot{U}_n$ of coarse grid solution at n-th coupling time. In practice, the gradient operator, ∇, will be replaced by some numerical approximation ∇_h. Then define the data:

$$\mathsf{F} := \begin{bmatrix} \nabla_h \mathcal{R}u_1 & \nabla_h \mathcal{R}u_2 & \cdots & \nabla_h \mathcal{R}u_N \\ c^{-1}\mathcal{R}\dot{u}_1 & c^{-1}\mathcal{R}\dot{u}_2 & \cdots & c^{-1}\mathcal{R}\dot{u}_N \end{bmatrix}, \tag{7}$$

$$\mathsf{G} := \begin{bmatrix} \nabla_h U_1 & \nabla_h U_2 & \cdots & \nabla_h U_N \\ c^{-1}\dot{U}_1 & c^{-1}\dot{U}_2 & \cdots & c^{-1}\dot{U}_N \end{bmatrix}. \tag{8}$$

Here and for the rest of the paper, $c^{-1}\dot{U}_n$ denotes the component-by-component multiplications of $c^{-1}(x_j)$ and $\dot{U}_n(x_j)$. The same convention is used for $c^{-1}\mathcal{R}\dot{u}_j$.

Now, define the discrete wave energy function as

$$E([U_n, \dot{U}_n]) := \frac{1}{2}\sum_j^{N_\Delta x} |\nabla_h U_n(x_j)|^2 \Delta x^d + \frac{1}{2}\sum_j^{N_\Delta x} c_j^{-2} |\dot{U}_n(x_j)|^2 \Delta x^d. \tag{9}$$

We see that it is equivalent, up to a constant, to the Frobenius norm of the G:

$$\|\mathsf{G}\|_F^2 = \sum_{n=1}^{N} \Big[\sum_j^{N_\Delta x} |\nabla_h U_n(x_j)|^2 + \sum_j^{N_\Delta x} c_j^{-2} |\dot{U}_n(x_j)|^2 \Big] = \frac{2}{\Delta x^d} \sum_{n=1}^{N} E([U_n, \dot{U}_n]). \tag{10}$$

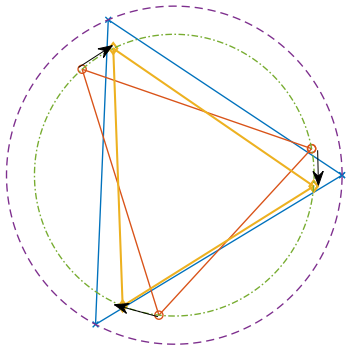

Fig. 2. Example of Procrustes problem in (13). Blue cross points are reference solution, red circle points represents solution to be aligned blue points. Yellow diamond points are corrected solution of red circle points. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3.2. Minimization of coarse-fine solution gaps

For simple plane waves, it is well known that the phase error, not the amplitude difference, between coarse and fine solutions, causes the parareal iteration to converge slowly or diverge [35,22]. If two plane waves are in phase, parareal style updates can effectively correct the amplitude error. For general wave solutions, it is inconvenient to work with the phase notion defined by the plane wave solutions. Instead, we consider the discrete wave energy semi-norm (9) which is induced by the ℓ^2 inner-product of the energy component vectors, i.e. the columns of F, G in (7) and (8). Such inner-product gives us a notion of angle between two wave solutions. The proposed strategy to stabilize the parareal iteration is by minimizing the inner-product between coarse and fine energy component vectors without changing their ℓ^2 norm. Similar strategies of using wave energy to compare wavefields for wave propagation purposes have been used successfully, for example in seismic imaging [31], wavefield approximation by Gaussian beams [39].

Denote the j-th column of F and G by f_j and g_j respectively. We consider the following optimization problem:

$$\min_{\Omega\in\mathbb{R}^{(d+1)N_{\Delta x}\times(d+1)N_{\Delta x}}} \sum_{j=1}^{N} ||f_j - \Omega g_j||_2^2, \quad \text{s.t. } \Omega\Omega^T = \Omega^T\Omega = I. \tag{11}$$

Recall that the elements in the columns of F and G consist of the spatial gradients and weighted time derivatives of the solutions on the respective fine and coarse grid, and that the ℓ^2 norm corresponds to the discrete wave energy (9). Therefore, we look for a unitary matrix so that the discrete wave energy of the corrected coarse solutions is the same as before correction. Intuitively the correction operator aligns the phase (in the above sense) of the coarse solution to fine solution for each t_n. It is similar to the local phase-alignment procedure in [1] as depicted in Fig. 2. Indeed, from each term in the summation

$$||f_j - \Omega g_j||_2^2 = ||f_j||_2^2 + ||g_j||_2^2 - 2(f_j, \Omega g_j), \tag{12}$$

the minimization can be interpreted as minimizing the sum of the angles between the columns on the data matrices. Thus, we shall refer to Ω as the *phase corrector*.

The minimization problem (11) is equivalent to the "Procrustes Problem" [17]:

$$\min_{\Omega\in\mathbb{R}^{(d+1)N_{\Delta x}\times(d+1)N_{\Delta x}}} \|\mathsf{F} - \Omega\mathsf{G}\|_F^2, \quad s.t. \quad \Omega\Omega^T = \Omega^T\Omega = I, \tag{13}$$

where $||\cdot||_F$ denotes the Frobenius norm of a matrix. An in-depth review of the Procrustes problem can be found in [18]. Its variants have been instrumental to multidimensional statistical analysis, rigid body motion simulation, satellite tracking and machine learning [42,32,19].

3.3. Solution to the optimization problem

The optimization problem (11) can be solved in a couple of different ways. One of them is to use the singular value decomposition (SVD) of the correlation matrix

$$\mathsf{M} := \mathsf{F}\mathsf{G}^T = \sum_{j=1}^{n} \nabla_h \mathcal{R} u_j \otimes \nabla_h U_j + c^{-1}\mathcal{R}\dot{u}_j \otimes c^{-1}\dot{U}_j. \tag{14}$$

If matrix M has full rank, the minimizer of (11) is uniquely

$$\Omega_* = \mathsf{X}\mathsf{Y}^T, \tag{15}$$

where X, Y are the left and right singular vectors of $\mathsf{M} = \mathsf{X}\Sigma\mathsf{Y}^T$. Correspondingly, the minimum residual is

$$r_{min}^2 = \|\mathsf{F}\|_F^2 + \|\mathsf{G}\|_F^2 - 2\,\mathrm{trace}(\Sigma).$$

Fig. 2 illustrates the Procrustes problem and its solution in a simple setup in $\mathbb{R}^2$.

3.3.1. Low rank approximation of Ω_*

We now consider a low rank approximation of Ω_* for computational efficiency. Since the number of time slices is usually much smaller than the number of (coarse) spatial grid nodes, i.e. $N \ll (d+1)N_{\Delta x}$, we can factorize the data matrices using the reduced QR factorization. Denote the factorizations by $\mathsf{F} = \mathsf{Q}_F\mathsf{R}_F$ and $\mathsf{G} = \mathsf{Q}_G\mathsf{R}_G$, where

$$\mathsf{Q}_F, \mathsf{Q}_G \in \mathbb{R}^{(d+1)N_{\Delta x}\times N}, \quad \mathsf{R}_F, \mathsf{R}_G \in \mathbb{R}^{N\times N}.$$

With the singular value decomposition of the smaller system $\mathsf{R}_F\mathsf{R}_G^T = \mathsf{X}_F\Sigma\mathsf{Y}_G^T$, the correlation matrix can be factored into

$$\mathsf{M} = \mathsf{Q}_F\mathsf{X}_F\Sigma\mathsf{Y}_G^T\mathsf{Q}_G^T.$$

The last relation shows that

$$\mathrm{rank}(\mathsf{M}) = \mathrm{rank}(\mathsf{R}_F\mathsf{R}_G^T) = \min(\mathrm{rank}(\mathsf{F}), \mathrm{rank}(\mathsf{G})).$$

Hence we can use the factorization of the smaller $N \times N$ matrix $\mathsf{R}_F\mathsf{R}_G^T$ to obtain

$$\Omega_* = (\mathsf{Q}_F\mathsf{X}_F)(\mathsf{Q}_G\mathsf{Y}_G)^T. \tag{16}$$

By setting a tolerance to singular values in Σ, there are s singular values such that $\sigma_s \geq tol$ remained. As the result, we only need to store s number of columns in $\mathsf{Q}_F, \mathsf{Q}_G$, and the truncated phase corrector becomes

$$\Omega_* := \Big(\mathsf{Q}_F(:, 1:s)\mathsf{X}_F(1:s, 1:s)\Big)\Big(\mathsf{Q}_G(:, 1:s)\mathsf{Y}_G(1:s, 1:s)\Big)^T.$$

3.3.2. Enriching the phase corrector Ω_*

After every parareal iteration, more data becomes available. We can use this data to enrich the phase corrector. Define

$$\mathsf{M}^{k+1} = \mathsf{M}^k + \mathsf{F}^k(\mathsf{G}^k)^T.$$

The singular value decomposition of $M^{k+1} = \tilde{\mathsf{U}}\tilde{\mathsf{S}}\tilde{\mathsf{V}}^T$ can be updated using that of $M^k = \mathsf{U}\mathsf{S}\mathsf{V}^T$, see [7]. We summarize the update procedure is Algorithm 1.

Algorithm 1: Update SVD of the current correlation matrix $\mathsf{M}^{k+1} \equiv \tilde{\mathsf{U}}\tilde{\mathsf{S}}\tilde{\mathsf{V}}^T = \mathsf{U}\mathsf{S}\mathsf{V}^T + \mathsf{F}\mathsf{G}^T$, where $\mathsf{U}\mathsf{S}\mathsf{V}^T = \mathsf{M}^k$.

$[\tilde{\mathsf{U}}, \tilde{\mathsf{S}}, \tilde{\mathsf{V}}] \leftarrow$ UpdateSVD(U, S, V, F, G, *tol*) :
if S *is empty* **then**
  $\mathsf{Q}_F\mathsf{R}_F = \mathtt{truncatedqr}(\mathsf{F})$
  $\mathsf{Q}_G\mathsf{R}_G = \mathtt{truncatedqr}(\mathsf{G})$
  $\mathsf{X}_r\Sigma Y_r^T = \mathtt{svd}(\mathsf{R}_F\mathsf{R}_G^T)$
  $rankM = \mathtt{sum}(\mathtt{diag}(\Sigma)/\mathtt{max}(\mathtt{diag}(\Sigma)) > tol)$
  $\tilde{U} = \mathsf{Q}_F\mathsf{X}_r(:, 1:rankM)$
  $\tilde{V} = \mathsf{Q}_G\mathsf{Y}_r(:, 1:rankM)$
  $\tilde{S} = \Sigma(:, 1:rankM)(:, 1:rankM)$
else
  $\mathsf{U}_F = \mathsf{U}^T\mathsf{F}$
  $\mathsf{V}_G = \mathsf{V}^T\mathsf{G}$
  $\mathsf{Q}_F\mathsf{R}_F = \mathtt{truncatedqr}(\mathsf{F} - \mathsf{U}\mathsf{U}_F)$
  $\mathsf{Q}_G\mathsf{R}_G = \mathtt{truncatedqr}(\mathsf{G} - \mathsf{V}\mathsf{V}_G)$
  $\mathsf{H} = [\mathsf{U}_F; \mathsf{R}_F][\mathsf{V}_G; \mathsf{R}_G]^T + [\mathsf{S}\ \ 0; 0\ \ 0]$
  $X_h\Sigma Y_h = \mathtt{svd}(\mathsf{H})$
  $rankM = \mathtt{sum}(\mathtt{diag}(\Sigma)/\mathtt{max}(\mathtt{diag}(\Sigma)) > tol)$
  $\tilde{U} = [\mathsf{U}\ \ \mathsf{Q}_F]\mathsf{X}_h(:, 1:rankM)$
  $\tilde{V} = [\mathsf{V}\ \ \mathsf{Q}_G]\mathsf{Y}_h(:, 1:rankM)$
  $\tilde{S} = \Sigma(:, 1:rankM)(:, 1:rankM)$

3.4. Reconstruction of wavefield from the gradient

After correcting the energy components, i.e. the gradients and the weighted time derivatives, of the coarse solutions, it is necessary to reconstruct the wavefield pair from the corrected energy components. In other words, we denote $[q, p]$ as the corrected energy components of a wavefield pair $[w, \dot{w}]$

$$\begin{bmatrix} q \\ p \end{bmatrix} \equiv \Omega_* \Lambda \begin{bmatrix} w \\ \dot{w} \end{bmatrix} := \Omega_* \begin{bmatrix} \nabla_h w \\ c^{-1}\dot{w} \end{bmatrix}, \tag{17}$$

where the mapping $\Lambda : [w, \dot{w}] \mapsto [\nabla_h w, c^{-1}\dot{w}]$ takes function to wave energy components. Then we want to deduce the corrected wavefield pair $[v, \dot{v}]$ such that

$$\begin{bmatrix} \nabla v \\ c^{-1}\dot{v} \end{bmatrix} \simeq \begin{bmatrix} q \\ p \end{bmatrix}.$$

It is straightforward to find the latter component $\dot{v} = cp$. For the displacement component v, we use the spectral property of differentiation $\texttt{fft}\{\nabla v\} = i\boldsymbol{\xi}\,\texttt{fft}\{v\}$ to recover its the Fourier modes as follow

$$\texttt{fft}\{v\} = \begin{cases} -i(\boldsymbol{\xi} \cdot \texttt{fft}\{q\})|\boldsymbol{\xi}|^{-2} & \text{for } |\boldsymbol{\xi}| \neq 0, \\ \sum_j^{N_{\Delta x}} w(x_j) & \text{for } |\boldsymbol{\xi}| = 0. \end{cases} \tag{18}$$

We denote this mapping from energy component to wavefield component as $\Lambda^\dagger : [\nabla v, c^{-1}\dot{v}] \mapsto [v, \dot{v}]$. In particular, when the gradient is approximated by Fourier method, this reconstruction is an identity.

Proposition 3.1. *Suppose the gradient of function $v(x)$ is estimated by spectral method $\nabla_h v \equiv \texttt{ifft}\{i\boldsymbol{\xi}\,\texttt{fft}\{v\}\}$, then*

$$\Lambda^\dagger \Lambda \begin{bmatrix} v \\ \dot{v} \end{bmatrix} = \Lambda^\dagger \begin{bmatrix} \texttt{ifft}\{i\boldsymbol{\xi}\,\texttt{fft}\{v\}\} \\ c^{-1}\dot{v} \end{bmatrix} = \begin{bmatrix} v \\ \dot{v} \end{bmatrix}. \tag{19}$$

Proof. Let

$$\begin{bmatrix} w \\ \dot{w} \end{bmatrix} = \Lambda^\dagger \Lambda \begin{bmatrix} v \\ \dot{v} \end{bmatrix}$$

Since Λ maps function to energy components we have

$$\Lambda^\dagger \Lambda \begin{bmatrix} v \\ \dot{v} \end{bmatrix} = \Lambda^\dagger \begin{bmatrix} \nabla_h v \\ c^{-1}\dot{v} \end{bmatrix}$$

By construction of $\Lambda^\dagger$, for nonzero wavenumber $|\boldsymbol{\xi}| \neq 0$

$$\texttt{fft}\{w\} = -i\boldsymbol{\xi} \cdot \texttt{fft}\{\nabla_h v\}|\boldsymbol{\xi}|^{-2}.$$

Here the gradient is approximated using spectral method then

$$\begin{aligned} \texttt{fft}\{w\} &= -i\boldsymbol{\xi} \cdot \{i\boldsymbol{\xi}\,\texttt{fft}\{v\}\}|\boldsymbol{\xi}|^{-2} \\ &= \texttt{fft}\{v\}. \end{aligned} \tag{20}$$

And for zero wavenumber $|\boldsymbol{\xi}| = 0$, $\texttt{fft}\{w\} = \sum_j v(x_j) = \texttt{fft}\{v\}$. Thus, $w = v$ while the second energy component $\dot{w} = cc^{-1}\dot{v} = \dot{v}$. This concludes that the mapping $\Lambda^\dagger\Lambda$ is equal to identity. □

If the gradient is approximated by a central finite difference of $2m$-order instead of the spectral method, for one dimensional setting equation (20) in the proof above becomes

$$\begin{aligned} \texttt{fft}\{w\} &= -i\xi[i\Delta x^{-1} \sum_{j=1}^{m} (e^{ij\xi\Delta x} - e^{-ij\xi\Delta x})\beta_j\, \texttt{fft}\{v\}]|\xi|^{-2} \\ &= 2(\xi\Delta x)^{-1} \sum_{j=1}^{m} \sin(j\xi\Delta x)\beta_j \texttt{fft}\{v\}, \end{aligned}$$

where β_j are appropriate coefficients of the difference stencil. When the spatial grid is small enough $\xi\Delta x \ll 1$, above expression is approximately

$$\begin{aligned}\texttt{fft}\{w\} &= 2(\xi\Delta x)^{-1}\sum_{j=1}^{m}(j\xi\Delta x - \frac{1}{3!}(j\xi\Delta x)^3 + \mathcal{O}(j\xi\Delta x)^5)\beta_j\ \texttt{fft}\{v\}\\ &= 2\sum_{j=1}^{m}(j - \frac{1}{3!}j^3(\xi\Delta x)^2 + \mathcal{O}j^5(\xi\Delta x)^4)\beta_j\ \texttt{fft}\{v\}.\end{aligned}$$

Particularly for the second order central difference $m = 1$, we would have $\beta_1 = 1/2$, then

$$\texttt{fft}\{w\} = \text{sinc}(\xi\Delta x)\texttt{fft}\{v\},$$

which says that $|\texttt{fft}\{w\}| \le |\texttt{fft}\{v\}|$ because $\text{sinc}(\xi\Delta x) \le 1$.

In practice, we observe that the algorithm does not require spectral approximation of the gradient, but instead $\|\Lambda^\dagger\Lambda\|_2 \le 1$ is necessary for stability of the method. When central finite difference is utilized, it is well known that the modified wavenumber is less than $|\boldsymbol{\xi}|$, hence central difference satisfies the requirement $\|\Lambda^\dagger\Lambda\|_2 \le 1$. Algorithm 2 summarizes above procedure.

Algorithm 2: Reconstruct function from the gradient.

$w \leftarrow \text{grad2func}(\nabla_h w, \sum w)$:
$\hat{p} = \texttt{fft}(\nabla_h w)$
$\hat{q}(|\boldsymbol{\xi}| \neq 0) = -i\boldsymbol{\xi}\cdot\hat{p}|\boldsymbol{\xi}|^{-2}$
$\hat{q}(|\boldsymbol{\xi}| = 0) = \sum w$
$w = \texttt{ifft}(\hat{q})$

3.5. The proposed algorithm

The proposed algorithm couples the fine and the coarse propagators at times $n\Delta t_{com}, n = 1, 2, \ldots, N$ over the fine grid (the spatial grid that the fine solutions are defined). However, it is important to note that *the phase corrections are applied on the coarse grid*. If the two grids are not identical, an interpolation is needed. We denote the interpolation operator by $\mathcal{I}$. Furthermore, denote the mappings between the wavefield $[v, \dot{v}]$ and its energy components $[\nabla v, c^{-1}\dot{v}]$ by $\Lambda : [v, \dot{v}] \mapsto [\nabla v, c^{-1}\dot{v}]$ and $\Lambda^\dagger : [\nabla v, c^{-1}\dot{v}] \mapsto [v, \dot{v}]$. With these notations, the θ operator after k iterations can be written as

$$\theta^k[v, \dot{v}] = \mathcal{I}\Lambda^\dagger\boldsymbol{\Omega}_*^k\Lambda[v, \dot{v}].$$

Here we use $\boldsymbol{\Omega}_*^k$ to denote the phase corrector derived from the data matrix M^k.

Finally, our new algorithm can be written compactly as in θ-parareal form

$$\begin{bmatrix} u_{n+1}^{k+1} \\ \dot{u}_{n+1}^{k+1}\end{bmatrix} = \theta^k\mathcal{C}\begin{bmatrix}\mathcal{R}u_n^{k+1}\\ \mathcal{R}\dot{u}_n^{k+1}\end{bmatrix} + \mathcal{F}\begin{bmatrix}u_n^k\\ \dot{u}_n^k\end{bmatrix} - \theta^k\mathcal{C}\begin{bmatrix}\mathcal{R}u_n^k\\ \mathcal{R}\dot{u}_n^k\end{bmatrix}. \tag{21}$$

Algorithm 3 describes the new scheme in a pseudo-code form with more details.

Similar to the Krylov subspace method [15,10], our method requires orthogonalization of data matrices, but they are formed in a different way. In this paper, the data matrices are the multiplication of the wave energy components of the fine data and the coarse data as in equation (14). Then the phase correctors are constructed from the singular value decomposition of the data matrices. In [15,10], the data matrices, consisting of computed solutions, are orthogonalized to form projection operators. In contrast, our phase correctors are not projections, but they effectively induce translation of the coarse solutions on constant energy submanifolds.

4. Complexity analysis

There are three parts to our implementation: parallel fine propagator computation, construction of $\boldsymbol{\Omega}^*$ and the serial coarse updates. We assume that (i) no spatial domain decomposition, i.e. whole domain on a single core, (ii) standard QR complexity, i.e. no multithreading, (iii) communication between nodes and other tasks negligible.

In each iteration, the wall clock complexity for the parallel fine and coarse computations is in the order of

$$\frac{1}{n_{CPU}}\Big(\frac{T}{\delta t}N_{\delta x} + \frac{T}{\Delta t}N_{\Delta x}\Big)(d+1)$$

where n_{CPU} is the number of cores, $N_{\delta x}, N_{\Delta x}$ are respectively the total number of fine and coarse grid points. The complexity of serial coarse update in an iteration is

$$\frac{T}{\Delta t}(d+1)N_{\Delta x}.$$

Algorithm 3: The proposed algorithm.

Initialization: $[u_n^{k=1}, \dot{u}_n^{k=1}] = \mathcal{IC}[\mathcal{R}u_{n-1}^{k=1}, \mathcal{R}\dot{u}_{n-1}^{k=1}]$
$\Sigma = [\,]$, $\mathsf{X} = [\,]$, $\mathsf{Y} = [\,]$
while *tolerance not meet and* $k \le K$ **do**
  par**for** $n = 2 \to N$ **do**
    $[v_n, \dot{v}_n] = \mathcal{F}[u_{n-1}^k, \dot{u}_{n-1}^k]$
    $[U_n, \dot{U}_n] = \mathcal{C}[\mathcal{R}u_{n-1}^k, \mathcal{R}\dot{u}_{n-1}^k]$
  end

  Solve the orthogonal Procrustes problem:
  $\mathsf{F} = [\nabla_h \mathcal{R} v_n, \mathcal{R} c^{-1} \dot{v}_n]$
  $\mathsf{G} = [\nabla_h U_n, c^{-1}\dot{U}_n]$
  $[\mathsf{X}, \Sigma, \mathsf{Y}] = \texttt{UpdateSVD}(\mathsf{X}, \Sigma, \mathsf{Y}, \mathsf{F}, \mathsf{G}, tol)$

  for $n = 2 \to N$ **do**
    $[w, \dot{w}] = \mathcal{C}[\mathcal{R}u_{n-1}^{k+1}, \mathcal{R}\dot{u}_{n-1}^{k+1}]$
    $[q, p] = \mathsf{X}\mathsf{Y}^T[\nabla_h w, c^{-1}\dot{w}]$
    $[\tilde{q}, \tilde{p}] = \mathsf{X}\mathsf{Y}^T[\nabla_h U_n, c^{-1}\dot{U}_n]$
    Reconstruct function from gradient:
    $q_1 = \texttt{grad2func}(q, \sum w)$
    $\tilde{q}_1 = \texttt{grad2func}(\tilde{q}, \sum U_n)$
    Update next time step:
    $[u_n^{k+1}, \dot{u}_n^{k+1}] = [v_n, \dot{v}_n] + \mathcal{I}\Big([\texttt{real}(q_1 - \tilde{q}_1), c(p - \tilde{p})]\Big)$
  end
  $k = k + 1$
end

The complexity of standard QR factorization for constructing Ω is

$$(d+1)N_{\Delta x}N^2.$$

Therefore, the total complexity is

$$K(d+1)\Big(\frac{T}{n_{CPU}\delta t}N_{\delta x} + \frac{T}{n_{CPU}\Delta t}N_{\Delta x} + \frac{T}{\Delta t}N_{\Delta x} + N_{\Delta x}N^2\Big), \tag{22}$$

where K is the number of iterations. In this set up, the speed up over a serial fine computation is

$$\Big[K\big(\frac{1}{n_{CPU}} + \big(\frac{1}{n_{CPU}} + 1\big)\frac{\delta t N_{\Delta x}}{\Delta t N_{\delta x}} + \frac{N_{\Delta x}N\delta t}{N_{\delta x}\Delta t_{com}}\big)\Big]^{-1}. \tag{23}$$

Additionally, we have coarse/fine time step ratio $\Delta t/\delta t = m_t$, which implies $\Delta t_{com}/\delta t \ge m_t$, and their corresponding the mesh ratio is $\Delta x/\delta x = m_s$. Hence the theoretical speed up is

$$\frac{1}{K}\min\{\mathcal{O}(n_{CPU}), \mathcal{O}(m_s^d m_t), \mathcal{O}\big(\frac{m_s^d m_t}{N}\big)\}. \tag{24}$$

We note that the third term in above speed up is derived from the classical N^2 complexity for QR factorization (of matrices of fixed number of rows). The speed up analysis (24) presents the worst-case asymptotics as *N approaches infinity*. In practice, we observe that QR factorization has sub-quadratic scaling when multithreading, ubiquitous in modern computers, is enabled. However, to our knowledge, speed up analysis of QR in a multithreading environment is not straightforward. To illustrate the effectiveness of multithreading in computing QR factorization, consider random matrices with fixed $100,000$ number of rows and vary the number of columns in a way relevant to the paper. The computing time is presented in Fig. 3. The computing time roughly grows as the power of 1.5 of the number of columns, rather than quadratically according to the classical QR complexity. We also see that having 68 threads speed up the computation by more than a factor of 10. Finally from numerical experiments, the QR step in our algorithm takes relatively small amount of time compared to other components, see Section 7.2.4.

5. Stability and convergence

In this section, we will derive some estimates that show the stability and the convergence of Algorithm 3 under certain assumptions. We measure the difference, in the discrete energy semi-norm on the coarse grid, between the serially computed fine solution and the iterated solution.

Consider energy components of parareal iterated solution restricted on the coarse grid

$$\begin{bmatrix} \nabla_h U_{n+1}^k \\ \frac{1}{c}\dot{U}_{n+1}^k \end{bmatrix} \equiv \Lambda\mathcal{R}\begin{bmatrix} u_{n+1}^k \\ \dot{u}_{n+1}^k \end{bmatrix}.$$

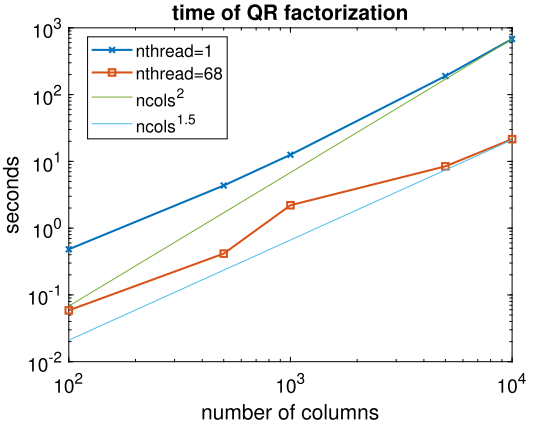

Fig. 3. Computing time of QR factorization as function of number of columns.

Its parareal iterative coupling is expressed as equation (21)

$$\begin{bmatrix} \nabla_h U_{n+1}^k \\ \frac{1}{c}\dot{U}_{n+1}^k \end{bmatrix} = \Lambda\mathcal{R}\left(\theta^{k-1}\mathcal{C}\begin{bmatrix} \mathcal{R}u_n^k \\ \mathcal{R}\dot{u}_n^k \end{bmatrix} + \mathcal{F}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \theta^{k-1}\mathcal{C}\begin{bmatrix} \mathcal{R}u_n^{k-1} \\ \mathcal{R}\dot{u}_n^{k-1} \end{bmatrix}\right). \tag{25}$$

Recall that $\theta[v,\dot{v}] := \mathcal{I}\Lambda^\dagger\Omega\Lambda[v,\dot{v}]$, so

$$\Lambda\mathcal{R}\theta^{k-1} = \Lambda\mathcal{R}\mathcal{I}\Lambda^\dagger\Omega_*^k\Lambda.$$

Since the restriction operator takes point wise values, it cancels action of the interpolation $\mathcal{R}\mathcal{I} = 1$. So equation (25) becomes

$$\begin{bmatrix} \nabla_h U_{n+1}^k \\ \frac{1}{c}\dot{U}_{n+1}^k \end{bmatrix} = \Lambda\Lambda^\dagger\Omega_*^k\Lambda\mathcal{C}\begin{bmatrix} U_n^k \\ \dot{U}_n^k \end{bmatrix} + \Lambda\mathcal{R}\mathcal{F}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger\Omega_k^*\Lambda\mathcal{C}\begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix}. \tag{26}$$

Let us denote the square root of wave energy as $\mathcal{E}([U,\dot{U}]) := \sqrt{E([U,\dot{U}])}$, where E is defined in (9). Thus,

$$\mathcal{E}([U_n^k,\dot{U}_n^k]) = \|\begin{bmatrix} \nabla_h U_n^k \\ \frac{1}{c}\dot{U}_n^k \end{bmatrix}\|_2.$$

Theorem 5.1. *Suppose that*

(1) *the coarse propagator $\mathcal{C}$ satisfies, for some $\epsilon > 0$;*

$$\mathcal{E}(\mathcal{C}[U,\dot{U}]) \le \mathcal{E}([U,\dot{U}]) + \epsilon;$$

(2) *the residual of the energy minimization problem is bounded uniformly for $k = 1, 2, \ldots$:*

$$\|\mathsf{F}^k - \Omega_*^k\mathsf{G}^k\|_F \le \eta$$

where $\mathsf{F}^k, \mathsf{G}^k$ are data matrices in (7), (8) gathered in the first k iterations;

(3) $\|\Lambda^\dagger\Lambda\|_2 \le 1$, *and* $\|1 - \Lambda^\dagger\Lambda\|_2 < \lambda \ll 1/N$.

Then

$$\max_{n\le N}\mathcal{E}([U_n^k,\dot{U}_n^k]) \le \frac{1}{1-\lambda N}\Big(\mathcal{E}([U_0,\dot{U}_0]) + (N+1)\epsilon + C\eta\Big), \tag{27}$$

where C is a norm equivalence constant between $\ell_{2,1}$ norm (sum of ℓ_2 norm of columns) and Frobenius norm.

Proof. Consider the square root of wave energy of (26)

$$\begin{aligned}\mathcal{E}([U_{n+1}^k, \dot{U}_{n+1}^k]) &= \| \begin{bmatrix} \nabla_h U_{n+1}^k \\ \frac{1}{c}\dot{U}_{n+1}^k \end{bmatrix} \|_2 \\ &= \| \Lambda\Lambda^\dagger \Omega_*^k \Lambda\mathcal{C} \begin{bmatrix} U_n^k \\ \dot{U}_n^k \end{bmatrix} + \Lambda\mathcal{R}\mathcal{F} \begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger \Omega_k^* \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2.\end{aligned}$$

We apply triangle inequality to obtain

$$\begin{aligned}\mathcal{E}([U_{n+1}^k, \dot{U}_{n+1}^k]) &\le \| \Lambda\Lambda^\dagger \Omega \Lambda\mathcal{C} \begin{bmatrix} U_n^k \\ \dot{U}_n^k \end{bmatrix} \|_2 + \| \Lambda\mathcal{R}\mathcal{F} \begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger \Omega \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2 \\ &\le \| \Lambda\Lambda^\dagger \|_2 \, \| \Omega \|_2 \, \| \Lambda\mathcal{C} \begin{bmatrix} U_n^k \\ \dot{U}_n^k \end{bmatrix} \|_2 + \| \Lambda\mathcal{R}\mathcal{F} \begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger \Omega \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2.\end{aligned}$$

By construction, $\|\Omega\|_2 = 1$, and by the hypotheses that $\|\Lambda\Lambda^\dagger\|_2 \le 1$ and energy bound of the coarse propagator,

$$\|\Lambda\mathcal{C}[U_n^k, \dot{U}_n^k]\|_2 = \mathcal{E}(\mathcal{C}[U_n^k, \dot{U}_n^k]) \le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon,$$

we have

$$\begin{aligned}\mathcal{E}([U_{n+1}^k, \dot{U}_{n+1}^k]) &\le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon + \| \Lambda\mathcal{R}\mathcal{F} \begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger \Omega_*^k \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2 \\ &\le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon + \| \Lambda\mathcal{R}\mathcal{F} \begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1} \end{bmatrix} - \Omega_*^k \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} - \Lambda\Lambda^\dagger \Omega_*^k \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2.\end{aligned}$$

Seeing the third term as part of the energy minimization problem in (10),

$$\begin{aligned}\mathcal{E}([U_{n+1}^k, \dot{U}_{n+1}^k]) &\le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon + \| f_{n+1} - \Omega_*^k g_{n+1} \|_2 + \| (1 - \Lambda\Lambda^\dagger) \Omega_*^k \Lambda\mathcal{C} \begin{bmatrix} U_n^{k-1} \\ \dot{U}_n^{k-1} \end{bmatrix} \|_2 \\ &\le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon + \| f_{n+1} - \Omega_*^k g_{n+1} \|_2 + \| 1 - \Lambda\Lambda^\dagger \|_2 \Big(\mathcal{E}([U_n^{k-1}, \dot{U}_n^{k-1}]) + \epsilon \Big) \\ &\le \mathcal{E}([U_n^k, \dot{U}_n^k]) + \epsilon + \| f_{n+1} - \Omega_*^k g_{n+1} \|_2 + \lambda \Big(\mathcal{E}([U_n^{k-1}, \dot{U}_n^{k-1}]) + \epsilon \Big) \\ &\le \mathcal{E}([U_0, \dot{U}_0]) + (n+1)\epsilon + \sum_{j=1}^{n} \| f_j - \Omega_*^k g_j \|_2 + \sum_{j=0}^{n} \lambda \Big(\mathcal{E}([U_j^{k-1}, \dot{U}_j^{k-1}]) + \epsilon \Big) \\ &\le \mathcal{E}([U_0, \dot{U}_0]) + (n+1)\epsilon + C\eta + \lambda n \Big(\max_{j \le N} \mathcal{E}([U_j^{k-1}, \dot{U}_j^{k-1}]) + \epsilon \Big).\end{aligned}$$

As the above relation also holds for $\max_{j \le N} \mathcal{E}([U_j^k, \dot{U}_j^k])$, therefore,

$$\begin{aligned}\max_{j \le N} \mathcal{E}([U_j^k, \dot{U}_j^k]) &\le \mathcal{E}([U_0, \dot{U}_0]) + (N+1)\epsilon + C\eta + \lambda N \Big(\max_{j \le N} \mathcal{E}([U_j^{k-1}, \dot{U}_j^{k-1}]) + \epsilon \Big) \\ &= \lambda N \max_{j \le N} \mathcal{E}([U_j^{k-1}, \dot{U}_j^{k-1}]) + \Big(\lambda N\epsilon + \mathcal{E}([U_0, \dot{U}_0]) + (N+1)\epsilon + C\eta \Big).\end{aligned}$$

Applying the discrete Grönwall inequality [21] on index k we get

$$\max_{j \le N} \mathcal{E}([U_j^k, \dot{U}_j^k]) \le (\lambda N)^{k-1} \max_{j \le N} \mathcal{E}([U_j^1, \dot{U}_j^1]) + \Big(\lambda N\epsilon + \mathcal{E}([U_0, \dot{U}_0]) + (N+1)\epsilon + C\eta \Big) \sum_{l=0}^{k-1} (\lambda N)^l.$$

By the assumption $\lambda N \ll 1$,

$$\max_{j \le N} \mathcal{E}([U_j^k, \dot{U}_j^k]) \le \Big(\lambda N\epsilon + \mathcal{E}([U_0, \dot{U}_0]) + (N+1)\epsilon + C\eta \Big) \frac{1}{1 - \lambda N}. \quad \square$$

Next, we will show that, under some hypotheses, the proposed method converges to the solutions computed by applying the fine propagator serially. The hypotheses involve Lipschitz smoothness of the phase corrector, which implies the minimization problem (11) is solved with sufficient accuracy. We shall use the following notation for those reference solutions:

$$[u(t_n), \dot{u}(t_n)] := \mathcal{F}^n [u_0, \dot{u}_0], \quad n = 1, 2, \ldots, N. \tag{28}$$

We measure the overall error on the fine grid as the square root of the difference in the discrete wave energy:

$$\mathcal{E}_n^k := \|\Lambda[u_n^k - u(t_n), \dot{u}_n^k - \dot{u}(t_n)]\|_2. \tag{29}$$

Hypothesis 5.1. *(i) The phase corrected coarse solution is Lipschitz continuous in energy*

$$\begin{aligned}\|\Lambda\theta\mathcal{C}\mathcal{R}\begin{bmatrix} v \\ \dot{v}\end{bmatrix} - \Lambda\theta\mathcal{C}\mathcal{R}\begin{bmatrix} w \\ \dot{w}\end{bmatrix}\|_2 &= \|\Lambda\mathcal{I}\Lambda^\dagger\Omega\Lambda\mathcal{C}\mathcal{R}\begin{bmatrix} v \\ \dot{v}\end{bmatrix} - \Lambda\mathcal{I}\Lambda^\dagger\Omega\Lambda\mathcal{C}\mathcal{R}\begin{bmatrix} w \\ \dot{w}\end{bmatrix}\|_2 \\ &\le (1+\epsilon_{\mathcal{I}\mathcal{R}})(1+\epsilon_{\Lambda^\dagger\Omega\Lambda\mathcal{C}})\|\Lambda\begin{bmatrix} v-w \\ \dot{v}-\dot{w}\end{bmatrix}\|_2.\end{aligned}$$

Let ϵ_θ denote the overall perturbation

$$\|\Lambda\theta\mathcal{C}\mathcal{R}\begin{bmatrix} v \\ \dot{v}\end{bmatrix} - \Lambda\theta\mathcal{C}\mathcal{R}\begin{bmatrix} w \\ \dot{w}\end{bmatrix}\|_2 \le (1+\epsilon_\theta)\|\Lambda\begin{bmatrix} v-w \\ \dot{v}-\dot{w}\end{bmatrix}\|_2. \tag{30}$$

(ii) The energy error between fine and corrected coarse operators is Lipschitz continuous

$$\|(\Lambda\mathcal{F} - \Lambda\theta\mathcal{C}\mathcal{R})\begin{bmatrix} v \\ \dot{v}\end{bmatrix} - (\Lambda\mathcal{F} - \Lambda\theta\mathcal{C}\mathcal{R})\begin{bmatrix} w \\ \dot{w}\end{bmatrix}\|_2 \le \kappa\|\Lambda\begin{bmatrix} v-w \\ \dot{v}-\dot{w}\end{bmatrix}\|_2. \tag{31}$$

Theorem 5.2. *Suppose that the fine and corrected coarse operators satisfy Hypothesis (30) and (31). Then,*

$$\max_{j\le N}\mathcal{E}_j^k \le \kappa\frac{(1+\epsilon_\theta)^N - 1}{\epsilon_\theta}\max_{j\le N}\mathcal{E}_j^{k-1}. \tag{32}$$

Proof. In the following expansion of the parareal iteration, the superscript k in θ^k are dropped for brevity

$$\begin{aligned}\begin{bmatrix} u_{n+1}^k \\ \dot{u}_{n+1}^k\end{bmatrix} &= \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_n^k \\ \dot{u}_n^k\end{bmatrix} + \mathcal{F}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1}\end{bmatrix} \\ &= \theta\mathcal{C}\mathcal{R}\Big(\theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_{n-1}^k \\ \dot{u}_{n-1}^k\end{bmatrix} + \mathcal{F}\begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1}\end{bmatrix}\Big) \\ &+ \mathcal{F}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_{n-1}^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1}\end{bmatrix} \\ &= (\theta\mathcal{C}\mathcal{R})^{n+1}\begin{bmatrix} u_0 \\ \dot{u}_0\end{bmatrix} + (\theta\mathcal{C}\mathcal{R})^n\Big(\mathcal{F}\begin{bmatrix} u_0 \\ \dot{u}_0\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_0 \\ \dot{u}_0\end{bmatrix}\Big) \\ &+ (\theta\mathcal{C}\mathcal{R})^{n-1}\Big(\mathcal{F}\begin{bmatrix} u_1^{k-1} \\ \dot{u}_1^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_1^{k-1} \\ \dot{u}_1^{k-1}\end{bmatrix}\Big)\dots \\ &+ (\theta\mathcal{C}\mathcal{R})\Big(\mathcal{F}\begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1}\end{bmatrix}\Big) + \Big(\mathcal{F}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1}\end{bmatrix} - \theta\mathcal{C}\mathcal{R}\begin{bmatrix} u_n^{k-1} \\ \dot{u}_n^{k-1}\end{bmatrix}\Big).\end{aligned}$$

It can be verified that the serial fine solution $[u(t_{n+1}), \dot{u}(t_{n+1})]$ also satisfies above expression when superscript $k, k-1$ are dropped in solution vector $[u_\cdot^k, \dot{u}_\cdot^k]$. Then we have an expression for the difference of the solutions

$$\begin{aligned}\begin{bmatrix} u_{n+1}^k - u(t_{n+1}) \\ \dot{u}_{n+1}^k - \dot{u}(t_{n+1})\end{bmatrix} =& (\theta\mathcal{C}\mathcal{R})^{n-1}(\mathcal{F} - \theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_1^{k-1} - u(t_1) \\ \dot{u}_1^{k-1} - \dot{u}(t_1)\end{bmatrix} + \dots \\ &(\theta\mathcal{C}\mathcal{R})(\mathcal{F} - \theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_{n-1}^{k-1} - u(t_{n-1}) \\ \dot{u}_{n-1}^{k-1} - \dot{u}(t_{n-1})\end{bmatrix} + \\ &(\mathcal{F} - \theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_n^{k-1} - u(t_n) \\ \dot{u}_n^{k-1} - \dot{u}(t_n)\end{bmatrix}.\end{aligned} \tag{33}$$

Recall the square root of energy error is defined as

$$\mathcal{E}_{n+1}^k = \|\Lambda\begin{bmatrix} u_{n+1}^k - u(t_{n+1}) \\ \dot{u}_{n+1}^k - \dot{u}(t_{n+1})\end{bmatrix}\|_2.$$

Using triangle inequality on $\mathcal{E}_{n+1}^k$ with equation (33) we obtain

$$\begin{aligned}\mathcal{E}_{n+1}^k \le \|\Lambda(\theta\mathcal{C}\mathcal{R})^{N-1}(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_1^{k-1}-u(t_1)\\ \dot{u}_1^{k-1}-\dot{u}(t_1)\end{bmatrix}\|_2 \dots \\ + \|\Lambda(\theta\mathcal{C}\mathcal{R})(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_{n-1}^{k-1}-u(t_{n-1})\\ \dot{u}_{n-1}^{k-1}-\dot{u}(t_{n-1})\end{bmatrix}\|_2 \\ + \|\Lambda(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_n^{k-1}-u(t_n)\\ \dot{u}_n^{k-1}-\dot{u}(t_n)\end{bmatrix}\|_2.\end{aligned}$$

Apply equation (30) in Hypothesis 5.1 (i) to bound each term

$$\begin{aligned}\mathcal{E}_{n+1}^k \le (1+\epsilon_\theta)^{n-1}\|\Lambda(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_1^{k-1}-u(t_1)\\ \dot{u}_1^{k-1}-\dot{u}(t_1)\end{bmatrix}\|_2 \dots \\ + (1+\epsilon_\theta)\|\Lambda(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_{n-1}^{k-1}-u(t_{n-1})\\ \dot{u}_{n-1}^{k-1}-\dot{u}(t_{n-1})\end{bmatrix}\|_2 \\ + \|\Lambda(\mathcal{F}-\theta\mathcal{C}\mathcal{R})\begin{bmatrix} u_n^{k-1}-u(t_n)\\ \dot{u}_n^{k-1}-\dot{u}(t_n)\end{bmatrix}\|_2.\end{aligned}$$

Finally we use equation (31) in Hypothesis 5.1 (ii) to obtain

$$\begin{aligned}\mathcal{E}_{n+1}^k &\le (1+\epsilon_\theta)^{N-1}\kappa\|\Lambda\begin{bmatrix} u_1^{k-1}-u(t_1)\\ \dot{u}_1^{k-1}-\dot{u}(t_1)\end{bmatrix}\|_2 \dots \\ &+ (1+\epsilon_\theta)\kappa\|\Lambda\begin{bmatrix} u_{n-1}^{k-1}-u(t_{n-1})\\ \dot{u}_{n-1}^{k-1}-\dot{u}(t_{n-1})\end{bmatrix}\|_2 \\ &+ \kappa\|\Lambda\begin{bmatrix} u_n^{k-1}-u(t_n)\\ \dot{u}_n^{k-1}-\dot{u}(t_n)\end{bmatrix}\|_2 \\ &= (1+\epsilon_\theta)^{n-1}\kappa\mathcal{E}_1^{k-1}\cdots+(1+\epsilon_\theta)\kappa\mathcal{E}_{n-1}^{k-1}+\kappa\mathcal{E}_n^{k-1} \\ &\le \kappa\Big((1+\epsilon_\theta)^{n-1}\cdots+(1+\epsilon_\theta)+1\Big)\max_{j\le n}\mathcal{E}_j^{k-1} \\ &= \kappa\frac{(1+\epsilon_\theta)^n-1}{\epsilon_\theta}\max_{j\le n}\mathcal{E}_j^{k-1}.\end{aligned}$$

Thus

$$\max_{j\le N}\mathcal{E}_j^k \le \kappa\frac{(1+\epsilon_\theta)^N-1}{\epsilon_\theta}\max_{j\le N}\mathcal{E}_j^{k-1}.$$

By assumption $\kappa N<1$ and $\epsilon_\theta N \ll 1$, the error goes to zero as k approaches infinity. □

We see that the convergence depends on the Lipschitz constant κ in Hypothesis 5.1 (ii), which reflects the gap between the corrected coarse propagator to the fine propagator. This gap between propagators is quantified by the energy residual of the minimization (13).

6. Numerical study of the new algorithm

In this section, we study the influence of different components of the proposed algorithm to the overall stability and accuracy. From Section 6.1 to Section 6.4, we consider the influence of (i) varying the low-rank approximation of the optimal phase correctors Ω_*, (ii) effect of the phase corrector and the parareal update, (iii) different orders of approximation for the gradient ∇_h and interpolation operator $\mathcal{I}$. Regarding to the last item, we will use the following interpolation methods, written as MATLAB functions, in this section:

- `interpft`: Fourier interpolation
- `akima`: cubic Hermite interpolation
- `pchip`: cubic interpolation
- `linear`: linear interpolation

From Section 6.1 to Section 6.4, we shall consider the simplest one dimensional setting with $c\equiv 1$ for both coarse and fine propagator, and the initial data:

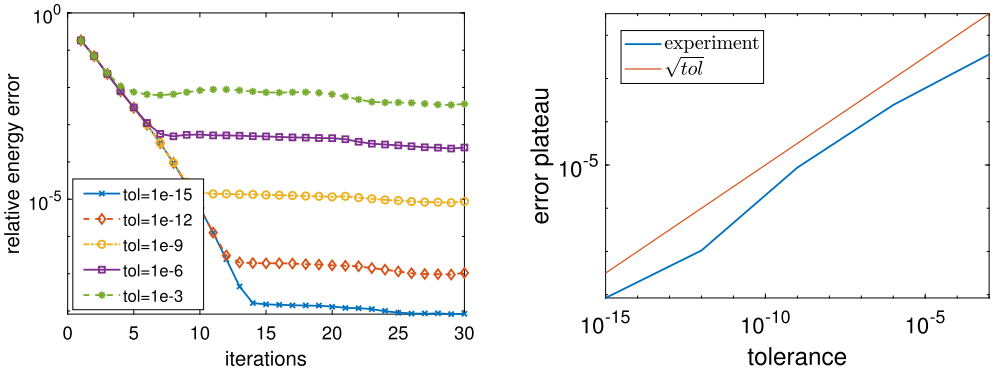

Fig. 4. Dependence of error convergence on rank tolerance which is in Algorithm 1. Left: relative energy error as a function of iterations. Right: the stagnated error value as a function of tolerance.

$$u(x,0) = \cos(10\pi x)\exp(-100x^2), \quad x \in [-0.5, 0.5]$$
$$u_t(x,0) = 0.$$

For Section 6.5, we consider random subsampling of the data matrices to exploit their observed low rank property. In this study, we consider a two dimensional problem with variable wave speed.
We will assume that the coarse grid nodes overlap with the fine grid nodes, and that the restriction operator $\mathcal{R}$ is just a point-wise evaluation on the coarse grid nodes.

The errors at final time $T_N = T$ are defined as square root of energy of difference on the fine grid

$$\sqrt{\frac{E([u_N^k - u(t_N), \dot{u}_N^k - \dot{u}(t_N)])}{E([u(t_N), \dot{u}(t_N)])}}.$$

And similarly the error can also be defined in ℓ^2 of difference in displacement component

$$\frac{\|u_N^k - u(t_N)\|_2}{\|u(t_N)\|_2}.$$

The reference solution $[u(t_N), \dot{u}(t_N)]$ are serially computed using the fine propagators.

6.1. Rank tolerance of the phase corrector

In this example, we study the sensitivity of the algorithm to rank-truncation of the optimal phase corrector Ω_*. We use the same spatial grid for both the coarse and the fine propagators in order to avoid error coming from interpolation/restriction. The fine propagator has an CFL number that is 20 times smaller than the coarse, and the coupling take place every 10 coarse steps. We sample several values for tolerance in Algorithm 1 at $10^{-15}, 10^{-12}, 10^{-9}, 10^{-6}, 10^{-3}$. The parameters are tabulated below:

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
5	0.05	0.01	0.5	1	20	`interpft`	2 order	$10^{\{-15,-12,-9,-6,-3\}}$

Fig. 4 shows the relative energy error along with the iterations as the tolerance in the truncation of Ω_* is varied. The errors decrease in the first few. The rate of decrease seem independent of the chosen tolerance values. As more iterations progress, the errors convergence eventually stagnate at certain values that strongly correlate to the chosen tolerance values. Particularly, the stagnated error values scale as the square root of the tolerance as shown on the right plot of Fig. 4. This scaling can be explained by the fact that the tolerance corresponds to the truncation of Ω_*, which modifies the wave energy components, and we measure the square root of wave energy difference. Hence in general, the convergence rate of our method is expected to slow down after the error has passed 10^{-8} because the tolerance can only be as small as machine epsilon 10^{-16}. Fig. 5 shows the number of retained singular values for different values of tolerance.

6.2. The effect of phase correction ($\Omega \equiv 1$)

Assuming again that the coarse and fine propagators are on the same grid. Without the phase correction, i.e. $\Omega \equiv 1$, the proposed iteration takes the form

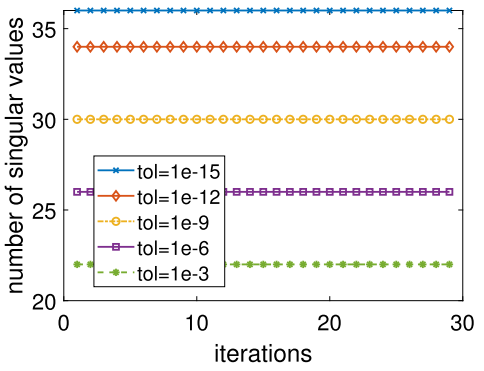

Fig. 5. Number of singular values for different value of tolerance.

$$\begin{bmatrix} u_n^k \\ \dot{u}_n^k \end{bmatrix} = \Lambda^\dagger \Lambda \mathcal{C} \begin{bmatrix} u_{n-1}^k \\ \dot{u}_{n-1}^k \end{bmatrix} + \mathcal{F} \begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1} \end{bmatrix} - \Lambda^\dagger \Lambda \mathcal{C} \begin{bmatrix} u_{n-1}^{k-1} \\ \dot{u}_{n-1}^{k-1} \end{bmatrix}.$$

The above expression becomes the plain parareal method if the term $\Lambda^\dagger \Lambda = 1$. But when the first wave component $\nabla_h u$ is approximated by some finite difference, the term $\Lambda^\dagger \Lambda \neq 1$ in general. In particular when ∇_h is approximated by the standard second order central difference, i.e. $\nabla_h = D^0_{\Delta x}$, $\Lambda^\dagger \Lambda$ corresponds to multiplication of

$$\frac{\sin \xi \Delta x}{\xi \Delta x} = \operatorname{sinc}(\xi \Delta x)$$

to the Fourier mode of the solutions. Since $|\operatorname{sinc}(\xi \Delta x)| \leq 1$, $\Lambda^\dagger \Lambda$ damps high frequency modes, and thus stabilizes parareal-like iterations.

Nevertheless, for long time simulations, such high frequency damping may be insufficient to stabilize the parareal-like iterations. To illustrate this, we take the same discretization as above but now consider four terminal times $T = 2.5, 5, 10, 50$:

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
$*$	0.05	0.01	0.5	1	20	`interpft`	2 order FD	10^{-14}

$* : \{2.5, 5, 10, 50\}$. Fig. 6 presents a comparison of the errors computed with $\Omega \equiv 1$ and with $\Omega = \Omega_*^k$, for different terminal times. For shorter time intervals, such as $T = 2.5$, the two choices of Ω yield similar convergence rates until after some iterations when the errors computed with Ω_* plateau around a much larger value. For larger terminal times, $T = 5, 10, 50$, the instability that comes with using $\Omega \equiv 1$ becomes more and more apparent, while the computations with $\Omega = \Omega_*^k$ remain stable.

6.3. The effect of parareal-like corrections

If the parareal-style additive correction is omitted, solution is propagated with just the phase corrected coarse propagator:

$$\begin{bmatrix} u_n^k \\ \dot{u}_n^k \end{bmatrix} = \theta^{k-1} \mathcal{C} \begin{bmatrix} \mathcal{R} u_{n-1}^k \\ \mathcal{R} \dot{u}_{n-1}^k \end{bmatrix}.$$

The simulation parameters are given as follow

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
10	0.05	0.01	0.5	1	20	`interpft`	4 order	10^{-14}

We first point out that if $\mathcal{C}$ preserves the discrete wave energy, then the above scheme will also preserve it by construction of θ^k. Fig. 7 shows the errors comparing to the serial fine solution. At iteration $k = 1$, the solution is serially computed with the coarse propagator $\mathcal{C}$. At iteration $k = 1$, a phase corrector θ^2 is constructed based on the data computed in $k = 1$. The solution at $k = 2$ is serially computed with $\theta\mathcal{C}$. On the right subplot of Fig. 7, we see that the coarse solution now has the same phase as the fine solution, but has a slightly different amplitude. For iteration after $k = 3$, however, the error does not decrease further since the parareal-style additive correction has been omitted. Comparing to the examples with similar simulation parameters presented in the previous subsection, we see that the parareal-style correction

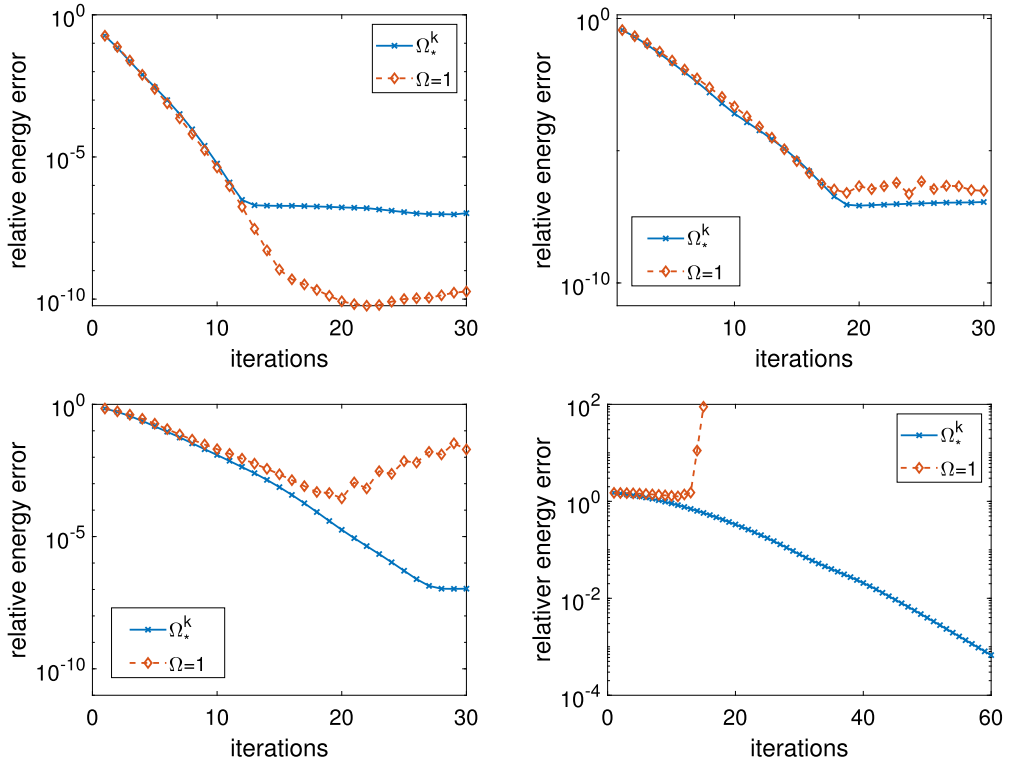

Fig. 6. Comparison of error convergence at different terminal time. Top row, left: $T = 2.5$, right: $T = 5$. Bottom row, left: $T = 10$, right: $T = 50$. Applying optimized Ω_* to solution is shown in cross solid curve. No optimization $\Omega = 1$, but fine and coarse solutions are coupled in wave energy components, is shown in diamond dashes.

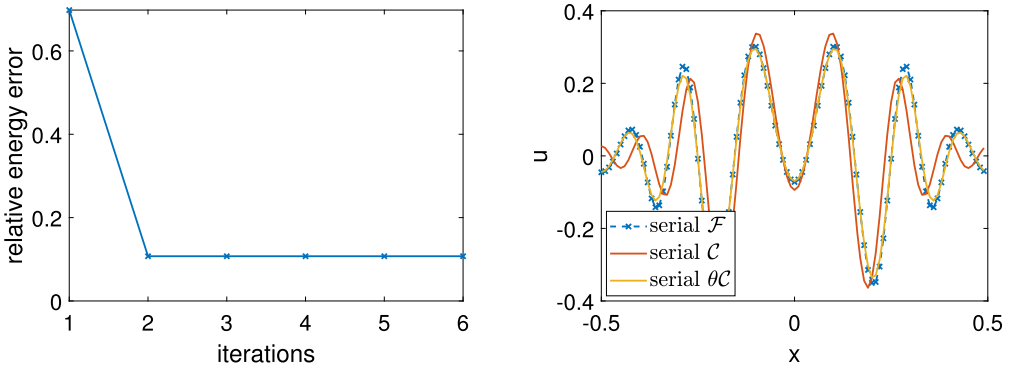

Fig. 7. The solution computed serially by the phase-corrected coarse propagator. Left: relative energy error of the phase-corrected coarse solution. Right: comparison with the serial fine and serial coarse solutions at $T = 10$.

$$\mathcal{F}[u_{n-1}^{k-1}, \dot{u}_{n-1}^{k-1}] - \theta^{k-1}\mathcal{C}[\mathcal{R}u_{n-1}^{k-1}, \mathcal{R}\dot{u}_{n-1}^{k-1}]$$

is important, as it adds the missing amplitudes back to improve accuracy (when the solutions are properly aligned).

6.4. Influence of interpolation and gradient approximation

So far in this section, we have only considered examples in which the coarse and fine propagators operate on the same spatial grid. When these propagators are on two different grids, interpolation is needed to couple the solutions. In this subsection, we study the effect of interpolation. To illustrate this point, take coarse/fine grid ratio to be 2 and keep the discretization as before

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
10	0.05	0.01	0.5	10	200	$*$	4 order	10^{-14}

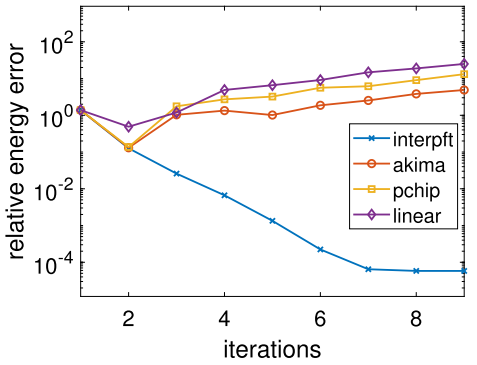

Fig. 8. Error convergence of the proposed method using different interpolation schemes.

$*:\{$`interpfft`, `akima`, `pchip`, `linear`$\}$. Input wave speed for coarse propagator is $c = 1$ as well. Fig. 8 shows the error convergence with different methods for grid interpolation. We observe particularly for this example that the spectral interpolation `interpft` performs better than the lower order methods because it resolves the initial wave form much better.

We also study the influence of the accuracy in approximating the gradient of the wavefield in forming the data matrices. We observe from the following examples that higher order approximations of gradient estimation accelerates convergence rate of the proposed method. The parameters used in the simulations are tabulated below:

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
10	0.05	0.01	0.5	1	20	`interpft`	$*$ order	10^{-14}

$*:\{2, 4, 6, 8, spectral\}$. To isolate other factors that can also influence the convergence rate, the table below shows the relative residual in Sec 3.3 averaged over all iterations, denoted as $\left\langle \|\mathsf{F} - \Omega_*^k \mathsf{G}\|_F \Big/ \|\mathsf{F}\|_F \right\rangle_k$. We see that the residual does not change while we increase the order of finite difference. In the last column, the errors in reconstruction of U from the its approximated gradient is provided. To be specific, we denote operation in equation (18) as $Y : \nabla_h v \mapsto v$.

approx. order of ∇_h	$\left\langle \frac{\|\mathsf{F} - \Omega_*^k \mathsf{G}\|_F}{\|\mathsf{F}\|_F} \right\rangle_k$	$\max_{i,k} \frac{\|Y \nabla_h U_i^k - U_i^k\|_2}{\|U_i^k\|_2}$
2	$3.8634 \cdot 10^{-3}$	$2.9435 \cdot 10^{-2}$
4	$3.8101 \cdot 10^{-3}$	$1.4023 \cdot 10^{-3}$
6	$3.8079 \cdot 10^{-3}$	$8.2100 \cdot 10^{-5}$
8	$3.8077 \cdot 10^{-3}$	$5.8569 \cdot 10^{-6}$
spectral	$3.8077 \cdot 10^{-3}$	$1.7706 \cdot 10^{-12}$

Fig. 9 shows the convergence of errors for different central differencing and Fourier approximations for ∇_h. The ones with second order approximation has the slowest convergence rate, while those using sixth order or higher converge faster.

6.5. Random subsample of the data matrices

We point out here that the cost of the stabilization can be further reduced by certain randomized algorithms [41, 27]. These randomized algorithms exploit the observed low-rank nature of our data matrices. Another approach is directly subsampling the data matrices. To illustrate the low-rank property, consider a plane wave in a 2D wave guide

$$c(x, y) = 1 - 0.3\cos(2\pi x), \quad -0.5 \le x \le 0.5, -0.5 \le y \le 0.5,$$

with the following discretization

T	Δt_{com}	Δt	δt	$N_{\Delta x}$	$N_{\delta x}$
4	0.02	$8 \cdot 10^{-4}$	$8 \cdot 10^{-5}$	100×100	200×200

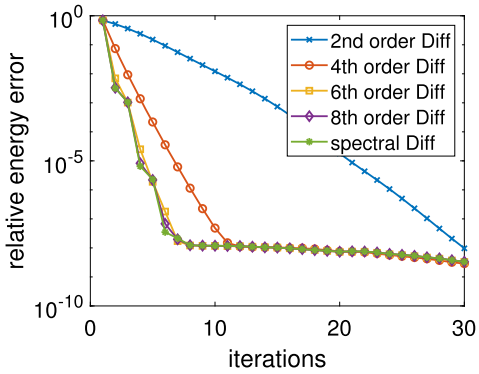

Fig. 9. Relative energy errors at $T = 10$, computed with different order of approximations to $\nabla_h U$.

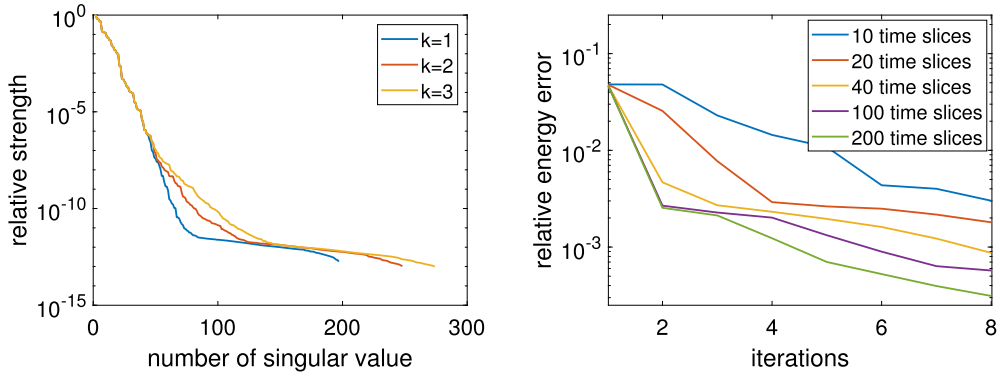

Fig. 10. Low rank property of the data matrices in Section 6.5. Left: normalized singular values of the correlation matrix M. Right: relative energy errors at final time using random subsample of the data matrices.

After each parallel computation of coarse and fine data G, F, we plot the normalized strength of singular values of the correlation matrix $\mathsf{M} = \mathsf{F}\mathsf{G}^T$ for a few iterations in Fig. 10.

The normalized strength of the singular values drops exponentially in this particular example. A quick and simple strategy to exploit this low-rank property is to randomly sample time slices in matrices F and G. By reducing the sample size, the data matrices becomes thinner so that QR factorization is faster. We compared the convergence of different sample sizes in Fig. 10.

7. Numerical examples

In this section, we shall consider one and two dimension examples, including an example that involve a large scale wave speed model commonly used in the seismic migration community. When the spatial grid of coarse and fine are different, wave speed on the coarse grid is point wise evaluation of the given wave speed.

7.1. One dimensional examples

Consider a medium with the wave speed

$$c(x) = 1 + 0.25\cos(4\pi x),$$

and the initial wavefield in $[-0.5, 0.5]$

$$u(x, 0) = \cos(10\pi x)\exp(-100x^2),$$
$$u_t(x, 0) = 0.$$

We present a numerical simulation using the parameters listed below:

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
10	0.05	0.01	0.5	10	100	`interpft`	4 order	10^{-14}

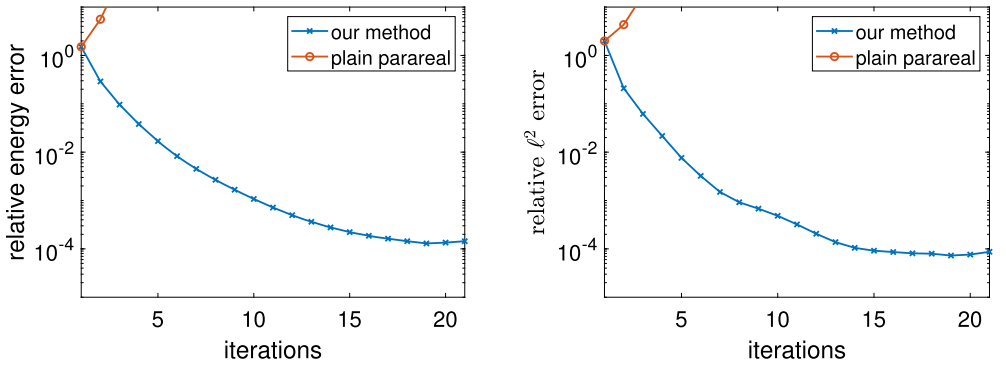

Fig. 11. Relative error of the solutions computed in numerical example Sec. 7.1. Left: the energy error, right: the ℓ^2 error. Our method, shown in cross solid line, generalizes beyond constant wave speed while the plain parareal method, shown in circle dash, diverges right away.

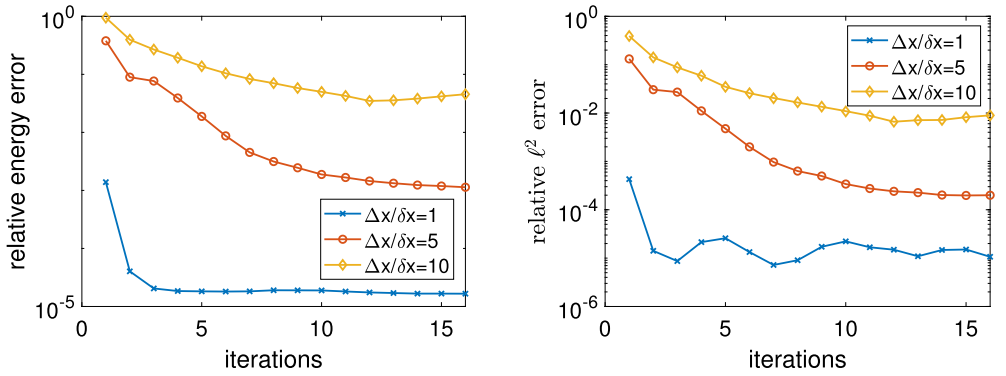

Fig. 12. Relative error of parareal iterated solutions for the waveguide example in Sec 7.2.1. Left: the energy error, right: the ℓ^2 error.

The fine propagator operate on a spatial grid which is 10 times finer than the coarse grid, and uses a CFL which is 10 times smaller. Fig. 11 shows convergence of the proposed method comparing to the plain parareal. Because fine and coarse solution in a variable medium may differ a lot, the plain parareal method becomes even more unstable.

7.2. Two dimensional cases

We apply the proposed method to three types of media: one with a smoothly varying wave speed (wave guide), one containing a piece-wise constant wave speed (inclusion), and a more complicated wave speed profile which is often used in exploration seismology as a standard case study (Marmousi).

7.2.1. Waveguide

We consider a wave guide in xy-plane $[-1, 1] \times [-0.5, 0.5]$ with the wave speed

$$c(x, y) = 1 - 0.3\cos(2\pi y).$$

The initial data is a plane wave traveling left to right along the x-axis:

$$u(x, y; 0) = \exp(-50(x + 0.5)^2),$$
$$u_t(x, y; 0) = 100\exp(-50(x + 0.5)^2).$$

The parameters used in the simulation are set as follow

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\lambda_\Delta/\lambda_\delta$	$\mathcal{I}$	∇_h	tol
5	0.05	0.005	1/4	$\{1, 5, 10\}$	5	`interpft`	4 order	10^{-13}

Fig. 12 shows error of the solution with different coarse fine grid ratio.

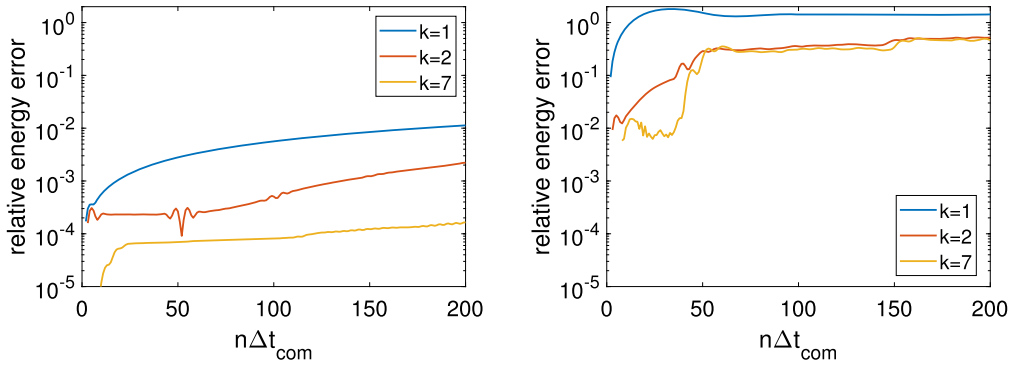

Fig. 13. Relative error over time for the inclusion example described in Sec 7.2.2. The initial plane wave "hits" the small inclusion at around $T = n\Delta t_{com}$. Left: the energy error for $\Delta x/\delta x = 1$, right: the energy error for $\Delta x/\delta x = 5$.

7.2.2. Inclusion

In this example, we consider the two dimensional domain in the xy-plane $[-1, 1] \times [-0.5, 0.5]$ where a plane wave encounters an inclusion of radius $\sqrt{0.002}$ centered at $[0.5, 0.1]$, modeled by the wave speed

$$c(x, y) = 1 - 0.9\, \chi_{((x-0.5)^2+(y+0.1)^2)<0.002}.$$

We used the initial data traveling from left to right

$$u(x, y; 0) = \cos(4\pi\,(x + 0.5)) \exp(-50(x + 0.5)^2),$$
$$u_t(x, y; 0) = \Big(-4\pi \sin(4\pi\,(x + 0.5)) + 100(x + 0.5) \cos(4\pi\,(x + 0.5)) \Big) \exp(-50(x + 0.5)^2),$$

and discretization parameters

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\lambda_\Delta/\lambda_\delta$	$\mathcal{I}$	∇_h	tol
4	0.02	0.005	1/2	$\{1, 5\}$	5	`interpft`	4 order	10^{-13}

When the coarse grid is the same as fine grid, the iterations converge to the serial fine solution for the whole time interval (shown in left subplot of Fig. 13). On the other hand, when coarse/fine grid ratio is 5, the right subplot of Fig. 13 shows that the error escalates quickly at $n = 50$ (or $t = 1$), when the initial plane wave hits the inclusion for the first time, and again at $n = 150$ (or $t = 2$), as some parts of the initial plane wave wraps around the domain the interact with the inclusion again. The error does not decreasing for later iterations.

Fig. 14 shows the relative density error in the Fourier modes of the computed solution at different times. For the short time range $n = 15$ (before the wave energy is scattered by the inclusion), most of the error concentrates at low frequencies which the coarse grid is able to resolve. Once the wave touches the inclusion at $n = 40$ and thereafter $n = 140, n = 200$, the errors in the higher frequencies becomes significant. These scattered higher frequency wave is not resolved by the coarse grid and cannot be corrected by the proposed method.

7.2.3. Marmousi experiment

We test our method with the Marmousi wave speed model [8], as shown in Fig. 15. The fine scale domain has 2422×7367 grid points while the coarse scale has 49×147 grid points or 50 times smaller in each dimension. The initial data is a pulse waveform centered at $x_0 = (400\text{ m}, 3880\text{ m})$, where m denotes the length unit in meter,

$$u(x, y; 0) = \cos(0.01(x - x_0)) \exp(-1.6 \cdot 10^{-5}((x - x_0)^2)),$$
$$u_t(x, y; 0) = 0.$$

The discretization parameters are in the following table where coarse and fine computation communicate every 500 coarse time steps

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
2 (s)	0.05 (s)	62.45 (m)	$1.6 \cdot 10^{-4}$	50	500	`imresize`	4 order	10^{-10}

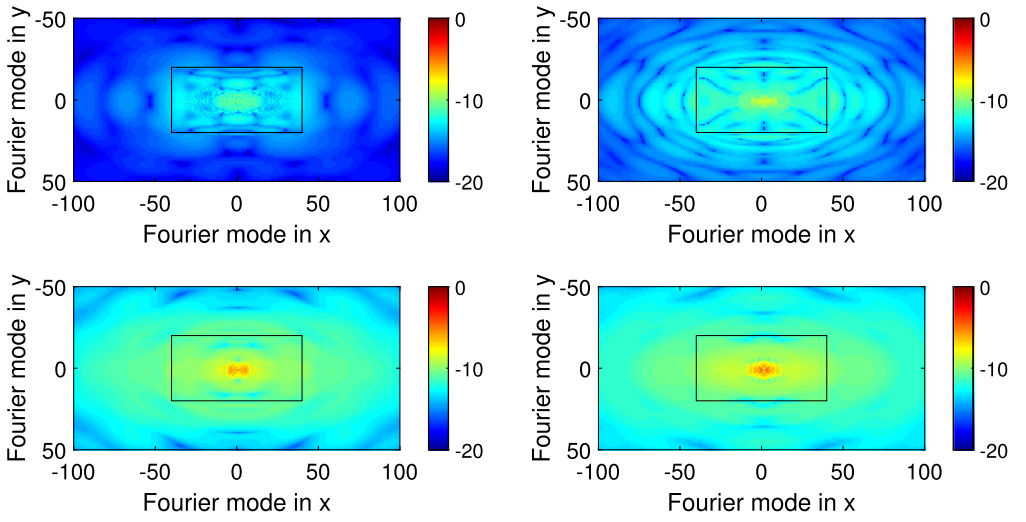

Fig. 14. Relative density error of solution of inclusion example for $\Delta x/\delta x = 5$ in Fourier modes $\frac{|\texttt{fft2}\{u_N^k(\xi_j) - u(\xi_j, t_N)\}|}{\sum_j |\texttt{fft2}\{u(\xi_j, t_N)\}|}$. The rectangular box indicates the Fourier domain of the coarse spatial grid. Top row, left: error at $n = 15$, right: error at $n = 40$. Bottom row, left: error at $n = 140$, right: error at $n = 200$.

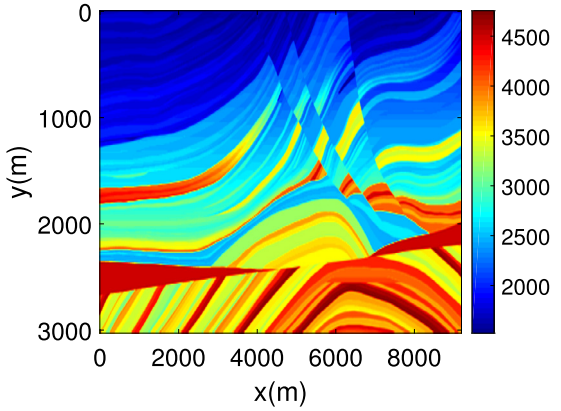

Fig. 15. Marmousi wave speed model. Domain size is 3022 (m) $\times$ 9192 (m) (m denotes 'meter') and unit of wave speed is in meter per second.

The computation is executed on one node consisting of 20 cores on the Stampede2 system at Texas Advanced Computing Center (TACC).[1] With our non-optimized MATLAB code, it took 26 hours to run 6 parareal iterations and 12 hours to compute the serial fine solution. Hence each iteration takes about 4 hours, almost 3 times faster on the wall clock than the serial fine computation. For more detailed experiment, see Section 7.2.4.

Fig. 16 shows the solutions computed by the proposed method. One observes that some finer details are added back to the computed solution along the iterations. However, Fig. 17 reveals that the errors decreases rather slowly after the first few iterations. Indeed, the setup in this experiment is a challenging example of strong scattering due to discontinuities in the wave speed (compared to the previous Example).

It is natural to wonder if the proposed method computes solutions that would converge to the serially computed fine solutions, *when the coarse and fine propagators run on the same spatial grid.* For this purpose, it suffices to consider a smaller version of the Marmousi velocity model, which is defined on 485 $\times$ 1474 grid points. A different set of discretization parameters are described in the following table

T	Δt_{com}	Δx	$\Delta t/\Delta x$	$\Delta x/\delta x$	$\Delta t/\delta t$	$\mathcal{I}$	∇_h	tol
2 (s)	0.05 (s)	6.245 (m)	$3.2026 \cdot 10^{-6}$	1	10	`imresize`	4 order	10^{-10}

[1] www.tacc.utexas.edu.

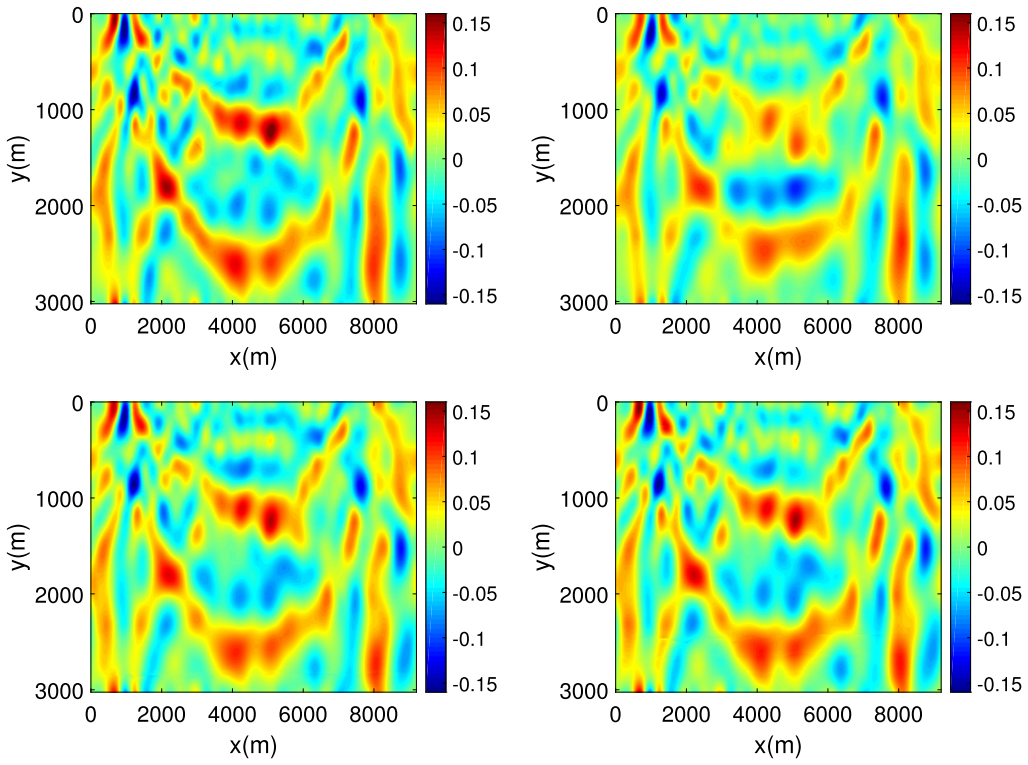

Fig. 16. Solution at $T = 2$ (s) of Marmousi example in Sec 7.2.3. Top row, left: serial fine solution, right: serial coarse solution. Bottom row, left: $k = 2$ iterated solution, right: $k = 4$ iterated solution.

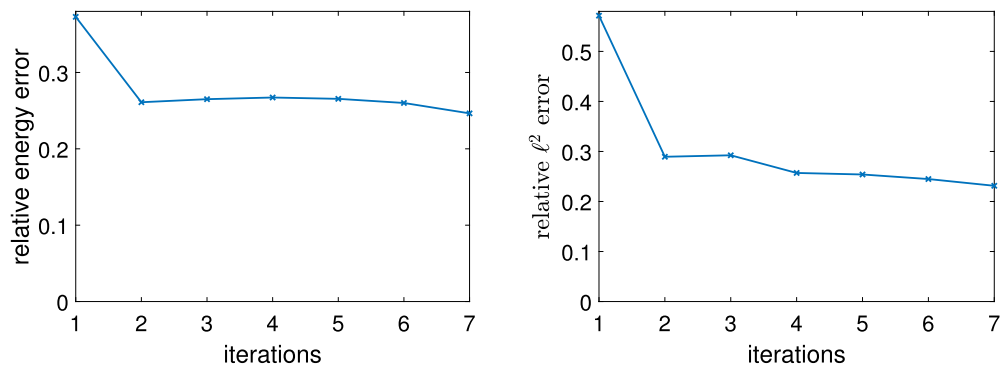

Fig. 17. Relative errors by the proposed method applied to simulate wave propagation in the Marmousi model. Left: the energy error, right: the ℓ^2 error.

Fig. 18 shows the absolute error $|u_n^k - u(t_n)|$ and energy error fields at iterations $k = 1$ and $k = 7$. On the left column, we see that the solution at $k = 1$ has larger point-wise absolute error and energy error in regions of high wave speed contrast (e.g. the lower left region in the image domain) than the regions of low wave speed contrast (e.g. upper left region in the image domain). On the right column, however, the solution at $k = 7$ has large patches of point-wise absolute error at regions of low wave speed contrast. These errors contribute to the increase of overall ℓ^2 error in the initial few iterations shown in the right subplot in Fig. 19.

We observe a discrepancy between the two errors curves. The energy error decreases while the ℓ^2 error increases, particularly in the regions of low wave speed contrast. This discrepancy in regions of low wave speed contrast is likely due to the construction of the phase corrector. At regions of high contrast, when locally scattered wave emerged, the phase corrector is constructed to decrease the error there but because it is a global operator, it also perturbs solution everywhere else that in effect increases the overall error.

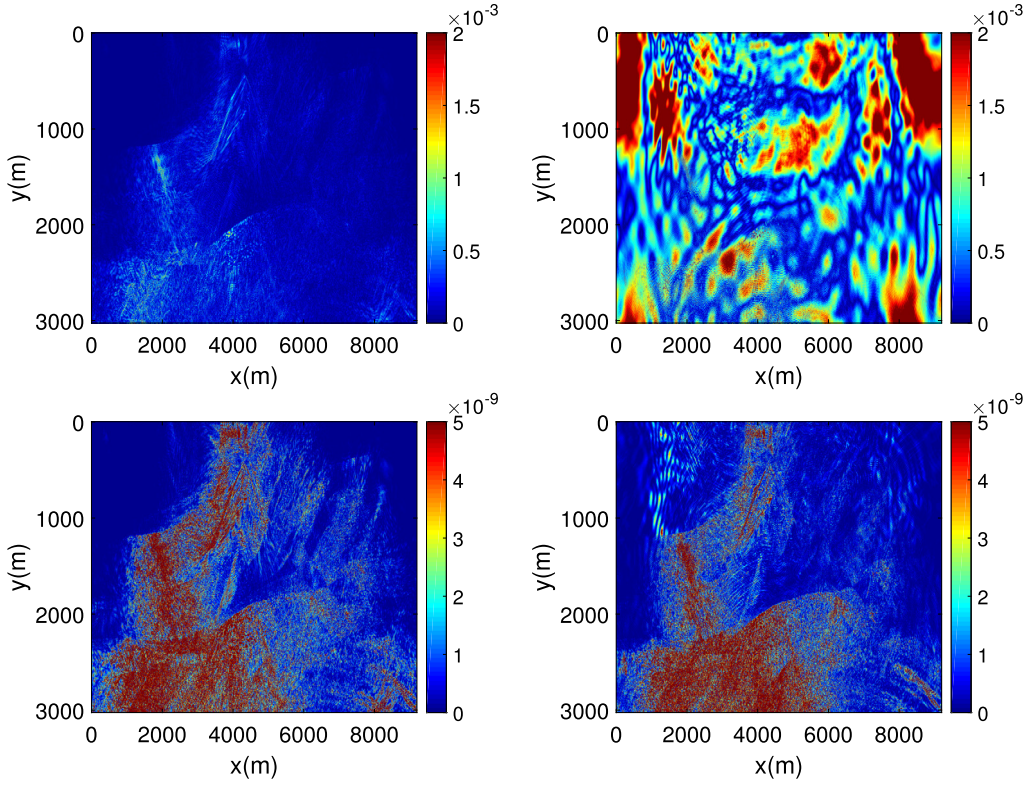

Fig. 18. Absolute and energy of absolute error field at $T = 2$ (s) when fine and coarse propagators run on the same grid for Marmousi example in Sec 7.2.3. Top row, left: absolute error of wavefield for $k = 1$ solution, right: absolute error of wavefield for $k = 7$ solution. Bottom row, left: energy error for $k = 1$ solution, right: energy error for $k = 7$ solution.

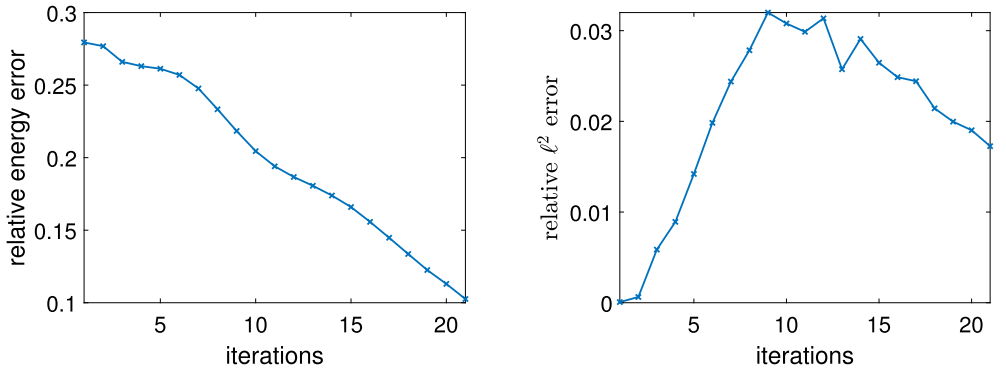

Fig. 19. Relative errors when coarse and fine propagators run on the same grid for the Marmousi model. Left: the energy error, right: the ℓ^2 error.

7.2.4. Timing

To see how wall clock computing time changes as the number of cores changes, we use the Marmousi model again and the discretization parameters are as follow

T	Δt_{com}	Δt	δt	$N_{\Delta x}$	$N_{\delta x}$
1, 5, 10	0.05	$8 \cdot 10^{-4}$	$4 \cdot 10^{-4}$	485×1474	2422×7367

We used an Intel Skylake node and varied the number of cores to perform the computation. In Table 1, computing time in seconds is recorded for different parts in the algorithm: parallel computation, creation of the phase corrector (requiring QR), serial coarse update. We see the computing time of the stabilization process is small, relative to other parts of the algorithm.

Table 1
Computing time (in seconds) of each part in our algorithm. Number of parareal iteration is 4. Projected speed up is calculated as if the number of CPUs is equal to the number of time slices. In the projected speed up calculation, we assume the time to create phase corrector does not change when the number of CPUs increases.

Cores	$N = T/\Delta t_{com}$	Parallel computation	Creating corrector	Serial update	Serial fine computation	Projected speed up
4	20	186.81	0.26	1.99	334.49	8.44
		180.84	0.11	1.52		4.32
		180.31	0.13	1.49		2.91
		180.35	0.15	1.52		2.18
	100	908.74	0.48	8.64	1724.30	37.92
		922.19	0.49	8.46		18.88
		911.85	0.64	8.72		12.57
		921.26	0.74	8.71		9.40
	200	1807.23	0.81	17.13	3479.63	64.34
		1837.55	1.06	17.90		31.69
		1819.07	3.14	17.82		20.82
		1854.84	1.77	18.98		15.47
8	20	119.61	0.95	1.78	333.78	7.83
		119.06	0.1	1.41		3.98
		129.25	0.11	1.43		2.60
		119.51	0.14	1.47		1.96
	100	525.09	0.46	8.27	1725.16	35.12
		547.06	0.49	7.78		17.34
		543.50	0.57	8.31		11.49
		527.34	0.73	8.33		8.63
	200	1023.37	0.99	16.26	3491.95	60.02
		1050.60	1.08	16.53		29.64
		1032.15	1.39	17.80		19.58
		1029.58	1.94	20.47		14.43
20	20	73.40	0.26	2.41	332.83	4.37
		65.93	0.11	1.51		2.32
		63.73	0.11	1.53		1.59
		66.45	0.13	1.58		1.46
	100	337.38	0.46	8.00	1734.02	22.84
		331.03	0.46	8.18		11.50
		335.75	0.58	8.34		7.64
		345.86	0.72	8.76		5.67
	200	656.80	1.06	16.66	3492.86	41.88
		644.41	1.03	17.73		20.96
		655.77	1.35	17.39		13.92
		653.80	1.73	19.35		10.35

As a benchmark, we also timed the serial fine computation. The projected speed up is calculated as if the number of cores is equal to the number of time slices $n_{CPU} = N = T/\Delta t_{com}$.

8. Summary and conclusion

We present here a new stable parareal-like method for the second order wave equation. The method uses the solutions computed along the iterations to construct linear operators which bridge the energy difference between the coarse and fine propagators. Such operators are referred to as the phase correctors in this paper. We presented an extensive set of numerical studies which aim at revealing the properties of the proposed method. From the experiments, we see that the proposed method works well for constant and smooth wave speeds.

For piece-wise smooth wave speeds, the algorithm is stable, but does not seem to produce numerical solutions that converge to the solutions computed by the fine propagators (as the number of iterations increase), when the fine and coarse propagators run on different spatial resolution. This is expected because the higher Fourier modes of the solutions computed by the fine propagator on a finer spatial grid cannot be resolved by coarser grids. This is true even when the initial wavefield is resolved by the coarse grid. As our simulations reveal, the stagnation of the errors may be caused additionally by a couple of different approximations used in the algorithm. This paper outline these factors for future improvement. In the last two examples involving piece-wise smooth wave speeds with high contrast, we observe that the relative errors are in general much larger than the previous cases. Most likely, this is due to strong local scattering of waves cause by the discontinuities in the wave speeds. Such scatterings cannot be corrected efficiently by the proposed Procrustean approach.

Finally, if domain decomposition in space is applied, due to the finite speed of propagation nature of wave, we expect that different phase correctors in the subdomains can be constructed in the same way and the resulting algorithm would be stable. This important topic should be investigated more carefully in a separate paper.

Declaration of competing interest

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

Acknowledgement

The authors are supported partially by NSF grants DMS-1620396 and DMS-1720171. Nguyen is supported by an ICES NIMS fellowship. Part of this research was performed while the second author was visiting the Institute for Pure and Applied Mathematics (IPAM), which is supported by the National Science Foundation (Grant No. DMS-1440415). This work was partially supported by a grant from the Simons Foundation. The authors thanks TACC for providing computing resources.

References

[1] G. Ariel, S.J. Kim, R. Tsai, Parareal multiscale methods for highly oscillatory dynamical systems, SIAM J. Sci. Comput. 38 (6) (2016) A3540–A3564.
[2] G. Ariel, H. Nguyen, R. Tsai, Theta-parareal scheme, arXiv:1704.06882 [math.NA], 2017.
[3] A. Arteaga, D. Ruprecht, R. Krause, A stencil-based implementation of parareal in the C++ domain specific embedded language STELLA, Appl. Math. Comput. 267 (2015) 727–741.
[4] G. Bal, On the Convergence and the Stability of the Parareal Algorithm to Solve Partial Differential Equations, Springer, Berlin, Heidelberg, 2005, pp. 425–432.
[5] A.-M. Baudron, J.-J. Lautard, Y. Maday, M.K. Riahi, J. Salomon, Parareal in time 3d numerical solver for the lwr benchmark neutron diffusion transient model, J. Comput. Phys. 279 (2014) 67–79.
[6] A. Blouza, L. Boudin, S.M. Kaber, Parallel in time algorithms with reduction methods for solving chemical kinetics, Commun. Appl. Math. Comput. Sci. 5 (2) (2011) 241–263.
[7] M. Brand, Fast low-rank modifications of the thin singular value decomposition, Linear Algebra Appl. 415 (1) (2006) 20–30.
[8] A. Brougois, M. Bourget, P. Lailly, M. Poulet, P. Ricarte, R. Versteeg, Marmousi, model and data, in: EAEG Workshop-Practical Aspects of Seismic Data Inversion, 1990.
[9] E.J. Bylaska, J.Q. Weare, J.H. Weare, Extending molecular simulation time scales: parallel in time integrations for high-level quantum chemistry and complex force representations, J. Chem. Phys. 139 (7) (2013) 074114.
[10] F. Chen, J.S. Hesthaven, 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, 2014, pp. 187–214.
[11] R. Croce, D. Ruprecht, R. Krause, Parallel-in-space-and-time simulation of the three-dimensional, unsteady Navier-Stokes equations for incompressible flow, in: Modeling, Simulation and Optimization of Complex Processes-HPSC 2012, Springer, 2014, pp. 13–23.
[12] X. Dai, Y. Maday, Stable parareal in time method for first-and second-order hyperbolic systems, SIAM J. Sci. Comput. 35 (1) (2013) A52–A78.
[13] M. Duarte, M. Massot, S. Descombes, Parareal operator splitting techniques for multi-scale reaction waves: numerical analysis and strategies, ESAIM: Math. Model. Numer. Anal. 45 (5) (2011) 825–852.
[14] P.F. Fischer, F. Hecht, Y. Maday, A parareal in time semi-implicit approximation of the Navier-Stokes equations, in: Domain Decomposition Methods in Science and Engineering, Springer, 2005, pp. 433–440.
[15] M.J. Gander, M. Petcu, Analysis of a Krylov subspace enhanced parareal algorithm for linear problem, ESAIM Proc. 25 (2008) 114–129.
[16] M.J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, SIAM J. Sci. Comput. 29 (2) (2007) 556–578.
[17] G.H. Golub, C.F. Van Loan, Matrix Computations, vol. 3, JHU Press, 2012.
[18] J.C. Gower, G.B. Dijksterhuis, et al., Procrustes Problems, vol. 30, Oxford University Press on Demand, 2004.
[19] E. Grave, A. Joulin, Q. Berthet, Unsupervised alignment of embeddings with Wasserstein procrustes, arXiv preprint, arXiv:1805.11222, 2018.
[20] T. Haut, B. Wingate, An asymptotic parallel-in-time method for highly oscillatory pdes, SIAM J. Sci. Comput. 36 (2) (2014) A693–A713.
[21] J.M. Holte, Discrete Gronwall Lemma and Applications, 2009.
[22] M. Iizuka, K. Ono, Influence of the phase accuracy of the coarse solver calculation on the convergence of the parareal method iteration for hyperbolic PDEs, Comput. Vis. Sci. (2018).
[23] A. Kreienbuehl, A. Naegel, D. Ruprecht, R. Speck, G. Wittum, R. Krause, Numerical simulation of skin transport using parareal, Comput. Vis. Sci. 17 (2) (2015) 99–108.
[24] J.-L. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of PDE's, C. R. Acad. Sci. 332 (2001) 661–668.
[25] T. Lunet, J. Bodart, S. Gratton, X. Vasseur, Time-parallel simulation of the decay of homogeneous turbulence using parareal with spatial coarsening, Comput. Vis. Sci. 19 (1–2) (2018) 31–44.
[26] Y. Maday, O. Mula, M.-K. Riahi, Towards a fully scalable balanced parareal method: Application to neutronics, 2015.
[27] P.-G. Martinsson, G. Quintana-Ortí, N. Heavner, R. van de Geijn, Householder QR factorization with randomization for column pivoting (HQRRP), SIAM J. Sci. Comput. 39 (2) (2017) C96–C115.
[28] D. Mercerat, L. Guillot, J.-P. Vilotte, Application of the parareal algorithm for acoustic wave propagation, in: AIP Conference Proceedings, vol. 1168, AIP, 2009, pp. 1521–1524.
[29] A. Randles, E. Kaxiras, Parallel in time approximation of the lattice Boltzmann method for laminar flows, J. Comput. Phys. 270 (2014) 577–586.
[30] A. Randles, E. Kaxiras, A spatio-temporal coupling method to reduce the time-to-solution of cardiovascular simulations, in: Parallel and Distributed Processing Symposium, 2014 IEEE 28th International, IEEE, 2014, pp. 593–602.
[31] D. Rocha, N. Tanushev, P. Sava, Acoustic wavefield imaging using the energy norm, Geophysics 81 (4) (2016) S151–S163.
[32] D.A. Ross, D. Tarlow, R.S. Zemel, Unsupervised learning of skeletons from motion, in: European Conference on Computer Vision, Springer, 2008, pp. 560–573.
[33] D. Ruprecht, Convergence of parareal with spatial coarsening, PAMM 14 (1) (2014) 1031–1034.
[34] D. Ruprecht, Wave propagation characteristics of parareal, Comput. Vis. Sci. 19 (1–2) (June 2018) 1–17.
[35] D. Ruprecht, R. Krause, Explicit parallel-in-time integration of a linear acoustic-advection system, Comput. Fluids 59 (2012) 72–83.
[36] D. Samaddar, D. Coster, X. Bonnin, C. Bergmeister, E. Havlícková, L.A. Berry, W.R. Elwasif, D.B. Batchelor, Temporal parallelization of edge plasma simulations using the parareal algorithm and the SOLPS code, Comput. Phys. Commun. 221 (2017) 19–27.
[37] D. Samaddar, D. Coster, X. Bonnin, L. Berry, W. Elwasif, D. Batchelor, Application of the parareal algorithm to simulations of elms in iter plasma, Comput. Phys. Commun. 235 (2019) 246–257.

[38] D. Samaddar, D.E. Newman, R. Sánchez, Parallelization in time of numerical simulations of fully-developed plasma turbulence using the parareal algorithm, J. Comput. Phys. 229 (18) (2010) 6558–6573.
[39] N.M. Tanushev, B. Engquist, R. Tsai, Gaussian beam decomposition of high frequency wave fields, J. Comput. Phys. 228 (23) (2009) 8856–8871.
[40] J. Trindade, J. Pereira, Parallel-in-time simulation of the unsteady Navier–Stokes equations for incompressible flow, Int. J. Numer. Methods Fluids 45 (10) (2004) 1123–1136.
[41] J.A. Tropp, A. Yurtsever, M. Udell, V. Cevher, Streaming low-rank matrix approximation with an application to scientific simulation, arXiv preprint, arXiv:1902.08651, 2019.
[42] C. Wang, S. Mahadevan, Manifold alignment using procrustes analysis, in: Proceedings of the 25th International Conference on Machine Learning, ACM, 2008, pp. 1120–1127.