AN ASYMPTOTIC PARALLEL-IN-TIME METHOD FOR HIGHLY OSCILLATORY PDES

TERRY HAUT & BETH WINGATE

ABSTRACT. We present a new time-stepping algorithm for nonlinear PDEs that exhibit scale separation in time of a highly oscillatory nature. The algorithm combines the parareal method—a parallel-in-time scheme introduced in [24]—with techniques from the Heterogeneous Multiscale Method (HMM) (cf. [13, 2]), which make use of the slow asymptotic structure of the equations [32], [27], [28], [36]. By numerically computing a locally asymptotic solution we are able to "factor out" the fast oscillatory part of the solution, and solve for the remaining slow part of the solution using time steps that are orders of magnitude larger than standard time-stepping methods allow. The scheme has two elements. First, we use HMM to numerically advance the asymptotic form of the equations with a large time step ΔT. Second, we refine the solution in parallel on the subintervals $[n\Delta T, (n+1)\Delta T]$ using small time steps Δt and the iterative scheme scheme in [24]; the intermediate solutions on the sub-intervals $[n\Delta T, (n+1)\Delta T]$ are ephemeral, and are used only to converge the overall solution at the large time steps $n\Delta T$. Using the asymptotic structure allows the computed solutions to be close enough to the actual solution that parallel-in-time methods converge to high accuracy and with significant parallel speedup.

We present error bounds, based on the analysis in [17], that demonstrate convergence of the method. A complexity analysis also demonstrates that the parallel speedup increases arbitrarily with greater scale separation. Finally, we demonstrate the accuracy and efficiency of the method on the (one-dimensional) rotating shallow water equations, which is a standard test problem for new algorithms in geophysical fluid problems. Compared to exponential integrators such as ETDRK4 and Strang splitting—which solve the stiff oscillatory part exactly—we find that we can use coarse time steps ΔT that are orders of magnitude larger (for a comparable accuracy), yielding an estimated parallel speedup of approximately 100 for physically realistic parameter values. For the (one-dimensional) shallow water equations, we also show that the estimated parallel speedup of this "asymptotic parareal method" is more than a factor of 10 greater than the speedup obtained from the standard parareal method.

1. INTRODUCTION

We present a new algorithm to integrate nonlinear PDEs that exhibit scale separation in time. We focus on time scale separation of a highly oscillatory nature, where standard (explicit or implicit) time-stepping methods often require time steps that are on the order of the fastest oscillation to achieve accuracy. This type of equation arises in numerous scientific applications, including the large-scale simulations of the ocean and atmosphere that serve as the primary motivation of this paper [7, 8].

In particular, we consider computing solutions to equations of the form

$$(1.1) \qquad \frac{\partial \mathbf{u}}{\partial t} + \frac{1}{\epsilon} \mathcal{L} \mathbf{u} = \mathcal{N} (\mathbf{u}) + \mathcal{D} \mathbf{u}, \ \mathbf{u} (0) = \mathbf{u}_0,$$

where the linear operator $\mathcal{L}$ has pure imaginary eigenvalues, the nonlinear term $\mathcal{N} (\mathbf{u})$ is of polynomial type, the operator $\mathcal{D}$ encodes some form of dissipation, and ϵ is a small non-dimensional parameter. For notational simplicity, we let $\mathbf{u} (t)$ denote the spatial (vector-valued) function $\mathbf{u} (t, \cdot) = (u_1 (t, \cdot), u_2 (t, \cdot), \ldots)$. The operator $\epsilon^{-1} \mathcal{L}$ results in temporal oscillations on an order $\mathcal{O} (\epsilon)$ time scale, and generally necessitates small time steps if standard numerical integrators are used.

Our approach for integrating (1.1) uses a variant of the parareal algorithm [24], which is a parallel-in-time method that relies on a cheap coarse solver for computing in serial a solution with low accuracy, and a more expensive fine solver for iteratively refining the solutions in parallel. The key novelty in this paper is to replace the numerical coarse solution of the full equations (1.1) with a locally asymptotic approximation of (1.1). This enables us to effectively bypass the Nyquist constraint imposed by the fastest oscillations, and acheive much greater parallel speedup. Examples on the rotating shallow water equations—a standard benchmark against which to test new algorithms in geophysical fluid applications—demonstrate that this approach holds promise for increasing the accuracy and speed of geophysical fluid simulations. In fact, we find (see Section 6.2) that this approach allows us to take step sizes $\Delta T \gg \epsilon$ that are significantly larger than if alternative schemes are used for the coarse solver, including exponential integrators and split-step methods (which solve the stiff linear terms exactly). In the context of large-scale simulations of the ocean and atmosphere, the gains achieved from spatial parallelization alone are beginning to saturate, and the results in this paper are a preliminary effort toward achieving greater efficiency.

We first describe the standard parareal method in more detail, and some of the challenges for acheiving high parallel speedup for problems of the form (1.1). The basic approach of the parareal method is to take large time steps ΔT in serial using a coarse integrator of (1.1), and to iteratively refine the solutions in parallel using small time steps Δt and a more accurate integrator. This can result in significant speedup in real (wall-clock) time if the parareal iterations converge rapidly, and either the ratio $\Delta T / \Delta t$ of coarse and fine step sizes is large, or the cost of the coarse solver is much cheaper than that of the fine solver (see Section 5 for more details). Early applications of the parareal method include simulations of molecular dynamics [4], the Navier Stokes equation [15], and quantum control problems [26]; additional references can be found in [34]. Although the parareal method has been most widely used for parabolic-type PDEs, it has also been analyzed and used for accurate simulation of first and and second order hyperbolic systems (cf. [14], [16], and [10]). A recent variant of the parareal method also allows for the accurate long-time evolution of Hamiltonian systems [23]. Finally, general convergence results for the parareal algorithm can be found in [17, 5] ([17] also numerically demonstrates convergence on the Lorenz equations, which is of particular relevance to geophysical fluid problems).

Despite the many successes of the parareal method, a basic obstacle remains for equations of the form (1.1): namely, the step size ΔT for a coarse integrator that is based on a standard method generally must satisfy $\Delta T = \mathcal{O} (\epsilon)$ in order to achieve any accuracy at all (which is a prerequisite for convergence of the parareal

method). In practice, this can mean that the coarse integrator in the parareal method must use very small time steps for solving (1.1), and the parallel speedup can be minimal. There are, however, some types of highly oscillatory PDEs where numerical integrators have been developed that can take much larger time steps $\Delta T \gg \epsilon$ (cf. [18]).

In this paper, we use a numerically computed 'locally slow' solution that is based on the underlying asymptotic structure of (1.1), and which can allow step sizes $\Delta T \gg \epsilon$ significantly larger than the Nyquist constraint imposed by the $\mathcal{O}(\epsilon)$ temporal oscillations (and thus a potentially significant parallel speedup). A basic observation behind efficiently constructing a slow solution is that the solution $\mathbf{u}(t)$ to (1.1) has the asymptotic approximation $\mathbf{u}(t) = \exp(-t/\epsilon \mathcal{L}) \overline{\mathbf{u}}(t) + \mathcal{O}(\epsilon)$ (cf. [27], [28], [36]), where the slowly varying function $\overline{\mathbf{u}}(t)$ satisfies a reduced equation of the form

$$(1.2) \qquad \frac{\partial \overline{\mathbf{u}}}{\partial t} = \overline{\mathcal{N}}(\overline{\mathbf{u}}) + \overline{\mathcal{D}}\overline{\mathbf{u}}, \ \ \overline{\mathbf{u}}(0) = \mathbf{u}_0.$$

Here the nonlinear term $\overline{N}(\overline{\mathbf{u}})$ is given by the time average

$$(1.3) \qquad \overline{\mathcal{N}}(\overline{\mathbf{u}}(t)) = \lim_{T \to \infty} \frac{1}{T} \int_0^T e^{s\mathcal{L}} \mathcal{N}\left(e^{-s\mathcal{L}}\overline{\mathbf{u}}(t)\right) ds.$$

We emphasize that the above time averaging is performed with $\overline{\mathbf{u}}(t)$ held fixed. Similarly,

$$(1.4) \qquad \overline{\mathcal{D}}\overline{\mathbf{u}}(t) = \lim_{T \to \infty} \frac{1}{T} \int_0^T \left(e^{s\mathcal{L}} \mathcal{D} e^{-s\mathcal{L}}\right) \overline{\mathbf{u}}(t) ds.$$

Note that $\overline{\mathbf{u}}(t)$, and its time derivatives, are formally bounded independently of ϵ, and thus significantly larger time steps $\Delta T \gg \epsilon$ can be taken to evolve (1.2). Section 4 also discusses how a numerical integrator based on a finite version of the time averages (1.3) and (1.4) can be interpreted as a smoothed type of integrating factor method, and allows accuracy even when there is no scale separation in time (i.e. $\epsilon = \mathcal{O}(1)$); this is useful when the scale separation is localized in space, and where it is desirable to have a time step that is constrained only by the slow dynamics.

Despite the many successes of the above averaging procedure in elucidating important qualitative features (see e.g. [28] and [36] for geophysical fluid dynamics applications), in practice this approach may not be accurate enough for moderately small values of ϵ (e.g. $\epsilon = 10^{-2}$ is typical in geophysical fluid applications), where the implicit constant hidden in the $\mathcal{O}(\epsilon)$ notation can be significant [33]. Since the parameter ϵ is typically fixed in idealized applications, the resulting asymptotic approximation cannot be refined without some additional approach. Another limitation is that the asymptotic approximation (1.2) is generally only valid on an $\mathcal{O}(1)$ time interval, and this situation is usually not improved by adding more terms in the asymptotic expansion. For many applications, it is therefore necessary to refine this approach in order to approximate (1.1) with a given target accuracy and on longer time intervals.

Our approach for computing the asymptotic approximation (1.2)—for the purpose of constructing a slow solution—is based on evaluating the time averages $\overline{\mathcal{N}}(\overline{\mathbf{u}}(t))$ and $\overline{\mathcal{D}}\overline{\mathbf{u}}(t)$ numerically; this approach has also been used in [6] to solve Hamiltonian systems more efficiently (see also [29] and [19] for related approaches

in geophysical simulations). More generally, our numerical scheme for (1.2) is an instance of the Heterogeneous Multiscale Method (HMM) (cf. [13, 2, 1] for selective applications to highly oscillatory problems), which is a very general framework for efficiently computing approximations to problems that exhibit multiple spatial or temporal scales; a review of HMM can be found in [35]. The basic idea is that, by integrating in time against a carefully chosen smooth kernel, the time average can be performed over a window of length $T_0 = T_0(\epsilon) \ll 1/\epsilon$; therefore, the overall cost of evolving (1.2) is asymptotically smaller than the cost of computing (1.1) directly, and can lead to arbitrarily large efficiency gains. In problems arising in geophysical fluid applications, the value ϵ may only be moderately small (e.g. $\epsilon \approx 10^{-2}$), and in such cases numerically computing the average (1.3) can be as costly as explicitly integrating the full equation (1.1). However, the numerical average can itself be performed in an embarassingly parallel manner (see Section 2), and thus is not expected to impact the overall (wall-clock) speed of the algorithm. Finally, we remark that for solutions that develop sharp gradients, it may be necessary to use the modified version of the parareal method explored in [10]. However, this is beyond the scope of this paper.

The idea of using a coarse solution based on a modified equation is not new, and the possibility has been mentioned early on in the parareal literature (cf. [34]). In [25] and [12], multi-scale versions of the parareal method are developed for, respectively, deterministic and stochastic chemical kinetic simulations; in [12], the coarse solver is based on a deterministic (macroscopic) approximation. A recent paper [22] applies a version of the parareal method to systems of ODEs that exhibit fast and slow components. The multi-scale coarse solver in [22] uses a projection onto the slow, low-dimensional manifold, and examples are provided for singularly perturbed ODEs with dissipative-type scale separation. In contrast to [25], [12], and [22] , here we investigate this procedure for a model nonlinear PDE whose scale separation is of a highly oscillatory nature, and where methods that work well for stiff dissipative problems (e.g. implicit or exponential integrators) generally fail to impart significant speedup. Moreover, the asymptotic approximation (1.2) to (1.1) cannot be computed explicitly in most cases, and this necessitates using additional techniques. The locally asymptotic solver developed here works even when there is no scale separation, which is an important feature when the time scale separation is a function of space and time (as occurs in some geophysical fluid applications); in this case, the time step in the coarse, asymptotic solution is only constrained by the slow dynamics.

In Section 2, we present a version of the Heterogeneous Multiscale Method that is appropriate for efficiently computing the asymptotic approximation (1.2). We then present in Sections 3 a variant of the parareal method that is based on replacing the coarse solver with a locally asymptotic approximation. Section 4 shows that, by averaging over a scale on which the slow dynamics is occuring, this HMM-type coarse solution is able to achieve accuracy even when there is no scale separation. We provide complexity bounds for this algorithm, which demonstrate that the parallel speedup increases arbitrarily as ϵ decreases. We also present error bounds that are based on the analysis in [17] (see also [5]), and that demonstrate convergence of the method under reasonable assumptions. Finally, Section 6.2 discusses some numerical experiments on the (one-dimensional) rotating shallow water equations, which

serve as a standard first test for new numerical algorithms in geophysical applications. Our experiments show that, in contrast to standard versions of the parareal algorithm, the algorithm can converge to high accuracy in few iterations even when large time steps $\Delta T \gg \epsilon$ are taken for the coarse solution. In fact, compared to the exponential time differencing method (ETDRK4), the integrating factor method, and Strang splitting (cf. [9], [20], and [21])—all of which integrate the stiff linear term $\epsilon^{-1}\mathcal{L}$ exactly—our algorithm yields an estimated parallel speedup of ≈ 100 for the physically realistic value of $\epsilon = 10^{-2}$. We also show that, for $\epsilon = 10^{-2}$, this parallel speedup is at least 10 times greater than the speedup that can be achieved by using the standard parareal method with ETDRK4, OIFS, or Strang splitting as the coarse solver. Finally, we demonstrate that the asymptotic parareal method yields high accuracy even when $\epsilon = 1$ (i.e. in the absense of scale separation), and with a parallel speedup that is comparable to using the standard parareal method with ETDRK4, OIFS, or Strang splitting as the coarse solver. Using a coarser spatial discretization in the asymptotic solver may also result in even greater efficiency gains.

2. An asymptotic slow solution

We use the Heterogeneous Multiscale Method (HMM) to solve (1.2), which relies on computing the averages (1.3) and (1.4) numerically (see also [6]). The key idea is that, by averaging in time with respect to an appropriate smooth kernel and over a carefully selected window length $T_0 = T_0(\epsilon)$, the cost is asymptotically smaller than $1/\epsilon$ (which is the cost of solving the full equation (1.1) on an $\mathcal{O}(1)$ time interval). Once such time averages can be computed, then large step sizes $\Delta T \gg \epsilon$, coupled with a standard numerical integrator, can be taken to evolve (1.2). For simplicity, we restrict our discussion to computing the time average (1.3) (in fact, for the equations we consider here, the operators $\mathcal{L}$ and $\mathcal{D}$ commute, and so the average $\overline{\mathcal{D}}$ in (1.4) satisfies $\overline{\mathcal{D}} = \mathcal{D}$).

The basic approach for computing the time average (1.3) involves the following approximations:

$$
\begin{aligned}
\overline{\mathcal{N}}\left(\overline{\mathbf{u}}\left(t\right)\right) &= \lim_{T \to \infty} \frac{1}{T} \int_0^T e^{s\mathcal{L}}\mathcal{N}\left(e^{-s\mathcal{L}}\overline{\mathbf{u}}\left(t\right)\right) ds \\
&\approx \frac{1}{T_0} \int_0^{T_0} \rho\left(\frac{s}{T_0}\right) e^{s\mathcal{L}}\mathcal{N}\left(e^{-s\mathcal{L}}\overline{\mathbf{u}}\left(t\right)\right) ds \\
&\approx \frac{1}{M} \sum_{m=0}^{M-1} \rho\left(\frac{s_m}{T_0}\right) e^{s_m\mathcal{L}}\mathcal{N}\left(e^{-s_m\mathcal{L}}\overline{\mathbf{u}}\left(t\right)\right).
\end{aligned}
$$

(2.1)

The smooth kernel $\rho(s)$, $0 \leq s \leq 1$, is chosen so that the length $T_0 = T_0(\epsilon)$ of the time window over which the averaging is done is as small as possible, and that the error introduced by using the trapezoidal rule is negligible (see e.g. [13] and [11] for an error analysis).

More formally, we define the finite time average

$$
\overline{\mathcal{N}}_{\rho,T_0}\left(\overline{\mathbf{u}}\left(t\right)\right) = \int_0^{T_0} \rho\left(\frac{s}{T_0}\right) e^{s\mathcal{L}}\mathcal{N}\left(e^{-s\mathcal{L}}\overline{\mathbf{u}}\left(t\right)\right) ds.
$$

(2.2)

Then we need to choose the kernel $\rho(s)$ and the parameters $T_0 = T_0(\epsilon)$ and M so that the truncation error,

$$\left\| \overline{\mathcal{N}}(\overline{\mathbf{u}}(t)) - \overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t)) \right\|,$$

and the discretization error,

$$\left\| \overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t)) - \frac{1}{M} \sum_{m=0}^{M-1} \rho\left(\frac{s_m}{T_0}\right) e^{(s_m/\epsilon)\mathcal{L}} \mathcal{N}\left(e^{-(s_m/\epsilon)\mathcal{L}} \overline{\mathbf{u}}(t)\right) \right\|,$$

are smaller than our desired approximation tolerance. This can be accomplished by requiring that the kernel $\rho(s)$ satisfies $\rho^{(m)}(0) = \rho^{(m)}(1) = 0$, $m = 0, 1, \dots$ It will also be convenient for comparison with Section 4 to change variables $s \to s/\epsilon$ in the integrand in (2.2) to obtain the equivalent form

$$\overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t)) = \frac{1}{\epsilon T_0} \int_0^{\epsilon T_0} \rho\left(\frac{s}{\epsilon T_0}\right) e^{s/\epsilon \mathcal{L}} \mathcal{N}\left(e^{-s/\epsilon \mathcal{L}} \overline{\mathbf{u}}(t)\right) ds.$$

To better understand the role of $\rho(s)$ and the parameters T_0 and M, we informally analyze the above averaging procedure in more detail (6.1). First, since we are assuming that the nonlinear operator $\mathcal{N}$ in (1.3) is of polynomial type and $\mathcal{L}$ has pure imaginary eigenvalues, we can (in principle) expand $\overline{\mathbf{u}}(t)$ in terms of eigenfunctions of $\mathcal{L}$ and express the nonlinear term in the form

$$
\begin{aligned}
e^{s\mathcal{L}} \mathcal{N}\left(e^{-s\mathcal{L}} \overline{\mathbf{u}}(t)\right) &= \sum_{\lambda_n} e^{i\lambda_n s} \mathcal{N}_n(\overline{\mathbf{u}}(t)) \\
(2.3) \qquad &= \sum_{\lambda_n = 0} \mathcal{N}_n(\overline{\mathbf{u}}(t)) + \sum_{\lambda_n \neq 0} e^{i\lambda_n s} \mathcal{N}_n(\overline{\mathbf{u}}(t)),
\end{aligned}
$$

Here the pure imaginary numbers $i\lambda_n$ are linear combinations of the eigenvalues of $\mathcal{L}$, and the set $\lambda_n = 0$ corresponds to resonant interactions (see Section 6.1 for a concrete example in the context of the rotating shallow water equations). In fact, using the definition of the averaging $\overline{\mathcal{N}}(\overline{\mathbf{u}}(t))$ operator and the above decomposition, we see that

$$\overline{\mathcal{N}}(\overline{\mathbf{u}}(t)) = \sum_{\lambda_n = 0} \mathcal{N}_n(\overline{\mathbf{u}}(t)).$$

To compare this to the finite time average $\overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t))$, use (2.3) to express this as

$$
\begin{aligned}
\overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t)) &= \frac{1}{T_0} \int_0^{T_0} \rho\left(\frac{s}{T_0}\right) e^{s\mathcal{L}} \mathcal{N}\left(e^{-s\mathcal{L}} \overline{\mathbf{u}}(t)\right) ds \\
&= \sum_{\lambda_n} \left(\int_0^1 \rho(s) e^{i\lambda_n T_0 s} ds\right) \mathcal{N}_n(\overline{\mathbf{u}}(t)).
\end{aligned}
$$

Comparing $\overline{\mathcal{N}}_{\rho,T_0}(\overline{\mathbf{u}}(t))$ and $\overline{\mathcal{N}}(\overline{\mathbf{u}}(t))$, we therefore require that

$$(2.4) \qquad \int_0^1 \rho(s) e^{i\lambda_n T_0 s} ds \approx \begin{cases} 1, & \text{for } \lambda_n = 0, \\ 0, & \text{for } \lambda_n \neq 0. \end{cases}$$

In order to satisfy (2.4) with a time window length T_0 as small as possible, we choose a smooth kernel $\rho(s)$ that satisfies $\rho^{(m)}(0) = \rho^{(m)}(1) = 0$. Repeated

integration by parts shows that

$$\left| \int_0^1 \rho(s) e^{i\lambda_n T_0 s} ds \right| \le C_m \left| \lambda_n T_0 \right|^{-m}, \quad m = 1, 2, \dots$$

In particular, the above calculation indicates that choosing a time window $1 \ll T_0(\epsilon) \ll 1/\epsilon$ can formally yield an error

$$\left\| \overline{\mathcal{N}}(\overline{\mathbf{u}}(t)) - \overline{\mathcal{N}}_{\rho, T_0}(\overline{\mathbf{u}}(t)) \right\| = \mathcal{O}(\epsilon^m), \quad m > 1.$$

Moreover, repeated integration by parts, coupled with the Euler–Maclaurin formula, also shows that the error induced from the trapezoidal rule is similarly small,

$$\left\| \overline{\mathcal{N}}_{\rho, T_0}(\overline{\mathbf{u}}(t)) - \frac{1}{M} \sum_{m=0}^{M-1} \rho\left(\frac{s_m}{T_0}\right) e^{(s_m/\epsilon)\mathcal{L}} \mathcal{N}\left(e^{-(s_m/\epsilon)\mathcal{L}} \overline{\mathbf{u}}(t) \right) \right\| = \mathcal{O}(\epsilon^m), \quad m > 1.$$

One commonly used choice of kernel function is given by

$$\rho(s) = \begin{cases} C \exp\left(-1/\left(s(1-s)\right)\right), & 0 < s < 1, \\ 0, & |s| \ge 1, \end{cases}$$

where the constant C is such so that $\|\rho\|_1 = 1$.

3. The asymptotic Parareal Method

We briefly review the parareal algorithm, in the context of replacing the coarse solver with a numerically computed locally asymptotic solution based on the asymptotic structure of the equations described by [27], [28], [36][32, 36].

We suppose that we are interested in solving (1.1) on the time interval $t \in [0, 1]$. We let $\varphi_t(\mathbf{u}_0)$ denote the evolution operator associated with (1.1), so that $\mathbf{u}(t) = \varphi_t(\mathbf{u}_0)$ solves (1.1). In a similar way, we let $\overline{\varphi}_t(\mathbf{u}_0)$ denote the evolution operator associated with the approximation obtained from (1.2), so that $\overline{\mathbf{u}}(t) = e^{t/\epsilon\mathcal{L}} \overline{\varphi}_t(\mathbf{u}_0)$ is the solution to (1.2) and $\varphi_t(\mathbf{u}_0) - \overline{\varphi}_t(\mathbf{u}_0) = \mathcal{O}(\epsilon)$.

To describe the parareal method, we first divide the time interval $[0, T]$ into N subintervals $[n\Delta T, (n+1)\Delta T]$, $n = 0, \dots, N-1$. Starting with the identity

$$\mathbf{U}_n = \overline{\varphi}_{\Delta T}(\mathbf{U}_{n-1}) + (\varphi_{\Delta T}(\mathbf{U}_{n-1}) - \overline{\varphi}_{\Delta T}(\mathbf{U}_{n-1})), \quad \mathbf{U}_n = \mathbf{u}(n\Delta T),$$

the parareal method computes approximations $\mathbf{U}_n^k \approx \mathbf{U}_n$ by the iterative procedure:

$$\mathbf{U}_n^k = \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^k\right) + \left(\varphi_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right) - \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right)\right), \quad k = 1, 2, \dots$$

At iteration level $k = 0$, the slow approximation $\mathbf{U}_n^0 = \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^0\right)$ is used. Notice that, at iteration level k, the quantities $\mathbf{U}_{n-1}^{k-1}$ in the difference $\varphi_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right) - \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right)$ are already computed; consequently, the difference $\varphi_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right) - \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^{k-1}\right)$ can be computed in parallel for each n. Since the computation of $\overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^k\right)$ is inexpensive, the overall algorithm is also inexpensive in a parallel environment if the iterates converge rapidly. This parareal method is illustrated in figure 3.1. We note that the parareal method can be interpreted as an inexact Newton-type iteration (cf. [17]).

Full pseudocode for the asymptotic parareal method is presented below. In the pseudocode, we use the mid-point rule in the HMM-type scheme for the slow integrator, and Strang splitting for the fine integrator. We assume that $\varphi_{\Delta T}(\mathbf{u}_0)$ is computed using M small time steps Δt (so that $\Delta T = M\Delta t$), and that $\overline{\varphi}_{\Delta T}(\mathbf{u}_0)$ is computed using one big time step ΔT.

FIGURE 3.1. This figure illustrates the asymptotic parareal algorithm. The verticl axes represents a typical prognostic variable such as h, the thickness of the layer of fluid in the shallow water system. The pink line depicts the asymptotic solution at the large time steps $n\Delta T$. The blue line depicts the parallel-in-time, fine scale corrections, using small time steps $\Delta t \ll \Delta T$. Finally, the black line depicts the updated solution at the large time steps $n\Delta T$.
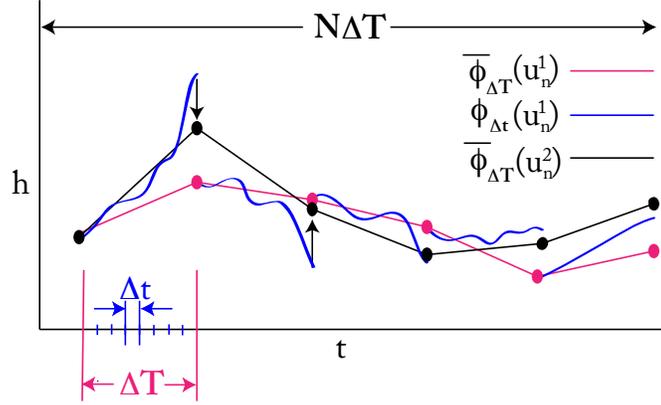
Algorithm 1 Evaluate time average (in parallel)

$$\overline{\mathcal{N}}\left(\overline{\mathbf{u}}_0\right):$$

```
parfor  j = 1, ..., M − 1:
```
$$s_m = T_0 m/M$$
$$\mathbf{u}_m \leftarrow \rho\left(s_m/T_0\right) e^{s_m \mathcal{L}} \mathcal{N}\left(e^{-s_m \mathcal{L}}\overline{\mathbf{u}}_0\right)$$
```
end parfor
```
$$\mathbf{u}_1 \leftarrow \mathtt{Sum}\left(\mathbf{u}_1, \ldots, \mathbf{u}_M\right)$$

Algorithm 2 asymptotic slow solver

$$\mathrm{Coarse\_Solver}(\mathbf{u}_0, \Delta T):$$

Take a $\Delta T/2$ timestep for the linear dissipative term:

$$\mathbf{v} \leftarrow e^{(\Delta T/2)\mathcal{D}}\mathbf{u}_0.$$

Take a ΔT timestep for the averaged nonlinear term:

$$\mathbf{v} \quad \leftarrow \quad \overline{\mathcal{N}}\left(\mathbf{v}\right),$$
$$\mathbf{v} \quad \leftarrow \quad \overline{\mathcal{N}}\left(\mathbf{u}_0 + \frac{\Delta T}{2}\mathbf{v}\right).$$

Take a $\Delta T/2$ timestep for the linear dissipative term:

$$\mathbf{v} \leftarrow e^{(\Delta T/2)\mathcal{D}}\mathbf{v}.$$

Transform back to the fast time coordinate:

$$\mathbf{u}_1 \leftarrow e^{(\Delta T/\epsilon)\mathcal{L}}\mathbf{v}.$$

Return $\mathbf{u}_1$

Algorithm 3 Fine solver

$$\text{Fine\_Solver}(\mathbf{u}_0, \Delta t, \Delta T):$$

$M = \Delta T / \Delta t$

for $m = 1, \ldots, M$:

    take $\Delta t/2$ timestep for the linear term:

$$\mathbf{v} \leftarrow e^{(\Delta t/2)(\epsilon^{-1}\mathcal{L}+\mathcal{D})}\mathbf{u}_m.$$

    take a Δt timestep for the nonlinear:

$$\mathbf{v} \quad \leftarrow \quad \mathcal{N}(\mathbf{v}),$$

$$\mathbf{v} \quad \leftarrow \quad \mathcal{N}\left(\mathbf{u}_m + \frac{\Delta t}{2}\mathbf{v}\right).$$

    take $\Delta t/2$ timestep for the linear term:

$$\mathbf{u}_{m+1} \leftarrow e^{(\Delta t/2)(\epsilon^{-1}\mathcal{L}+\mathcal{D})}\mathbf{v}.$$

end for

Return $\mathbf{u}_M$.

Algorithm 4 Parallel-in-time integrator

```
Compute the initial guess using the slow solver:
U_0^old ← u_0
for  n = 1, …, N − 1
     U_n^old ← Coarse_Solver(U_{n-1}^old, ΔT)
endfor
Now refine the solution until convergence:
U_0^new ← u_0
while  (max_n ‖U_n^new − U_n^old‖ / ‖U_n^new‖ > tol)
     parfor  n = 1, …, N − 1:
          U_n^old ← U_n^new
          V_n ← Fine_Solver(U_n^old, Δt, ΔT)
              V_n ← V_n − Coarse_Solver(U_n^old, ΔT)
     end parfor
     for  n = 1, …, N − 1
          U_n^new ← Coarse_Solver(U_{n-1}^new, ΔT) + V_{n-1}
     endfor
end while
return  U_1^new, …, U_N^new
```

---

4. The parallel-in-time algorithm without scale separation

In geophysical fluid problems, it is often the case that the time scale separation can change in space and time, and it is important that this algorithm works even when there is no scale separation. We give a heuristic derivation of the time average (2.2) from a different point of view, which indicates that the coarse solution yields accuracy even when $\epsilon = \mathcal{O}(1)$, as long as the time average in (2.2) is performed over a time scale on which the dynamics of the slow nonlinear terms are occurring. Figure 4.1 schematically depicts how the large time step ΔT varies as a function of

FIGURE 4.1. Schematic of ΔT as a function of ϵ for (a) the asymptotic parareal method (solid blue line), (b) the standard parareal method with a linearly exact coarse solver (dashed red line), and (c) a typical time-stepping method used in serial
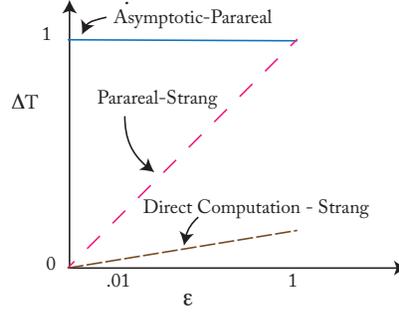
FIGURE 4.2. Schematic depiction of the moving time average
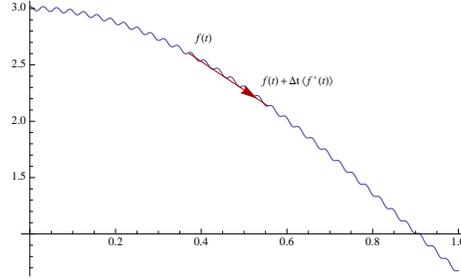
the scale separation parameter ϵ, for the asymptotic parareal method, the standard parareal method, and a typical time-stepping method that is used in serial.

As in the integrating factor method, we first factor out the fast oscillatory part,

$$\mathbf{u}(t) = e^{-t/\epsilon\mathcal{L}}\mathbf{v}(t),$$

so that $\mathbf{v}(t)$ satisfies

(4.1)
$$\frac{\partial \mathbf{v}}{\partial t} = e^{t/\epsilon\mathcal{L}}\mathcal{N}\left(e^{-t/\epsilon\mathcal{L}}\mathbf{v}(t)\right).$$

Since

$$\frac{\partial \mathbf{v}}{\partial t} = \mathcal{O}(1),$$

$\mathbf{v}(t)$ varies more slowly than $\mathbf{u}(t)$ and thus time steps $\Delta T \gg \epsilon$ can potentially be used to solve for $\mathbf{v}(t)$. However, simply using a standard time-stepping scheme for $\mathbf{v}(t)$ will still require small step sizes. In fact, differentiating the equation (4.1) shows that $\mathbf{v}(t)$ has small but rapid fluctuations,

$$\frac{\partial^2 \mathbf{v}}{\partial t^2} = \mathcal{O}\left(\frac{1}{\epsilon}\right),$$

and standard time-stepping schemes will not be accurate unless ΔT is small.

The idea (see also [3] and [30]) is to take a time step using a smoothed out version of the derivative; this is accomplished by using a moving time average, so that the small $\mathcal{O}\left(\epsilon\right)$ fluctuations in the derivative are removed; see Figure 4.2 for a schematic depiction. In particular, we average the derivative out over a time scale ϵT_0 on which the slow dynamics occur (that is, we average over T_0 fast oscillations). Then using e.g. forward Euler we get the approximation

$$\mathbf{v}\left(\Delta T\right) \approx \mathbf{v}\left(0\right) + \Delta T \left\langle \frac{\partial \mathbf{v}}{\partial t}\left(0\right) \right\rangle_{T_0},$$

where

$$
\begin{aligned}
\left\langle \frac{\partial \mathbf{v}}{\partial t}\left(0\right) \right\rangle_{T_0} &= \frac{1}{\epsilon T_0} \int_0^{\epsilon T_0} \rho\left(\frac{s}{\epsilon T_0}\right) \frac{\partial \mathbf{v}}{\partial s}\left(s\right) ds \\
&\approx \frac{1}{\epsilon T_0} \int_0^{\epsilon T_0} \rho\left(\frac{s}{\epsilon T_0}\right) e^{s/\epsilon \mathcal{L}} \mathcal{N}\left(e^{-s/\epsilon \mathcal{L}} \mathbf{v}\left(0\right)\right) ds.
\end{aligned}
$$

This approximation allows error control via two different mechanisms. First, when $\epsilon \ll 1$ the above approximation is also an asymptotic approximation, and the time average serves to eliminate secular terms that arise from nonlinear resonances as long as $\Delta T \gg \epsilon$. However, when $\epsilon = \mathcal{O}\left(1\right)$, then we can take $\epsilon T_0 = \Delta T$, so that the time average is performed over a scale ΔT on which the dynamics is slow. In this case, we see that the above approximation is essentially the forward Euler method. In fact, in this case the derivatives of $\mathbf{v}\left(s\right)$ are slow on an $\mathcal{O}\left(\Delta T\right)$ time scale, and we can Taylor expand to get

$$
\begin{aligned}
\frac{1}{\Delta T} \int_0^{\Delta T} \rho\left(\frac{s}{\Delta T}\right) \frac{\partial \mathbf{v}}{\partial s}\left(s\right) ds &\approx \frac{1}{\Delta T} \int_0^{\Delta T} \rho\left(\frac{s}{\Delta T}\right) \left(\frac{\partial \mathbf{v}}{\partial s}\left(0\right) + s \frac{\partial^2 \mathbf{v}}{\partial s^2}\left(0\right) + \dots\right) ds \\
&= \frac{\partial \mathbf{v}}{\partial s}\left(0\right) + \mathcal{O}\left(\Delta T\right).
\end{aligned}
$$

For systems of ODEs, rigorous error bounds for this "partial time averaging" are derived in Section 3.2 of [31].

5. Error and complexity bounds

We first discuss the complexity of our algorithm. Although this analysis is standard for the parareal method, the complexity bounds demonstrate (in theory) arbitrarily large parallel speedup as the parameter ϵ gets smaller. We assume that the time interval $[0, 1]$ is divided into N sub-intervals $[T_{n-1}, T_n]$ of length $\Delta T = T_n - T_{n-1} = 1/N$. We also assume that, within each subinterval, M time steps of size Δt are needed for the fine integrator, so that $M = \Delta T / \Delta t$.

To obtain an initial guess for the parareal method, we first compute the slow approximations $\mathbf{U}_n^0 = \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^0\right)$, $n = 1, \dots, N$, which takes a wall-clock time of $\tau_c N$. Next, suppose that we are at a given iteration level k, and we need to compute the next iterations $\mathbf{U}_n^{k+1}$ from $\mathbf{U}_n^k$. To do so, we first need to compute, in parallel, the difference $\mathbf{V}_n^k = \varphi_{\Delta T}\left(\mathbf{U}_n^k\right) - \overline{\varphi}_{\Delta T}\left(\mathbf{U}_n^k\right)$ between the coarse and fine solutions. This takes a wall-clock time of $\tau_c + \tau_f M$. We then need to compute, in serial, the updated approximations $\mathbf{U}_n^{k+1} = \overline{\varphi}_{\Delta T_{n-1}}\left(\mathbf{U}_{n-1}^{k+1}\right) + \mathbf{V}_{n-1}^k$, $n = 1, \dots, N - 1$. This takes a wall-clock time of $\tau_c N$, for an overall cost of $\tau_f M + \tau_c N + \tau_c$ per iteration. Thus, after ν iterations, the overall cost of the parareal method is

$\nu \left(\tau_{\mathrm{f}} M + \tau_{\mathrm{c}} N + \tau_{\mathrm{c}}\right) + \tau_{\mathrm{c}} N$. In contrast, directly solving (1.1) in serial requires time $\tau_{\mathrm{f}} N M$. Thus, the estimated parallel speedup is

$$\frac{\tau_{\mathrm{f}} N M}{\nu \left(\tau_{\mathrm{f}} M + \tau_{\mathrm{c}} N + \tau_{\mathrm{c}}\right) + \tau_{\mathrm{c}} N} \leq \min \left\{ \frac{\tau_{\mathrm{f}}}{\tau_{\mathrm{c}}} \frac{M}{(\nu+1)}, \frac{N}{\nu} \right\}.$$

Notice that an upper bound on the speedup is proportional to $M = \Delta T / \Delta t$, the ratio of slow to fine time step sizes.

In order for the asymptotic parareal method to converge, the fine solver needs time steps Δt that are some fraction of ϵ; therefore, $\Delta t \sim \epsilon$, $\Delta T \sim M \epsilon$, and $N M \sim 1/\Delta t \sim 1/\epsilon$. By choosing $N = M = \sqrt{1/\epsilon}$, we see that the parallel speedup is given by

$$\frac{\tau_{\mathrm{f}} N M}{\nu \left(\tau_{\mathrm{f}} M + \tau_{\mathrm{c}} N\right) + \tau_{\mathrm{c}} N} = \frac{\tau_{\mathrm{f}}}{\nu \left(\tau_{\mathrm{f}} + \tau_{\mathrm{c}}\right) + \tau_{\mathrm{c}}} \sqrt{\frac{1}{\epsilon}}.$$

Thus, the estimated parallel speedup increases as ϵ decreases, which results in arbitrarily greater efficiency gains relative to standard numerical integrators.

We now use the theory developed in [17] for our error bounds (the analysis in [5] would provide a more refined convergence analysis). We assume that the number N of large time steps ΔT taken satisfies $N = \epsilon^{-1/2}$ (based on the above complexity analysis, this yields, in principle, optimal parallel speedup). In particular, we assume that the slow evolution operator $\overline{\varphi}_{\Delta T}$ satisfies

(5.1) $$\left\| \overline{\varphi}_{\Delta T} \left(\mathbf{u}_1\right) - \overline{\varphi}_{\Delta T} \left(\mathbf{u}_2\right) \right\| \leq \left(1 + C_0 \Delta T\right) \left\| \mathbf{u}_1 - \mathbf{u}_2 \right\|.$$

We also assume that the difference between the slow and fine approximations satisfies

(5.2) $$\overline{\varphi}_{\Delta T} \left(\mathbf{u}_0\right) - \varphi_{\Delta T} \left(\mathbf{u}_0\right) = \mathcal{E} \left(\mathbf{u}_0\right) \epsilon,$$

where the operator $\mathcal{E} \left(\cdot\right)$ satisfies the bounds

(5.3) $$\left\| \mathcal{E} \left(\mathbf{u}_1\right) - \mathcal{E} \left(\mathbf{u}_2\right) \right\| \leq C_1 \left\| \mathbf{u}_1 - \mathbf{u}_2 \right\|, \quad \left\| \mathcal{E} \left(\mathbf{u}_0\right) \right\| \leq C_2.$$

Letting $T_n = n \Delta T$, a slight modification of the proof of Theorem 1 in [17] immediately yields the following result; we only provide the details that differ from [17].

Theorem 1. *The error, $\mathbf{u} \left(T_n\right) - \mathbf{U}_n^k$, after the kth parareal iteration is bounded by*

$$\left\| \mathbf{u} \left(T_n\right) - \mathbf{U}_n^k \right\| \leq \epsilon^{k/2+1} C_2 \frac{\left(C_1\right)^k}{(k+1)!} e^{C_0 \left(T_n - T_{k+1}\right)}.$$

Therefore, as long as the constants in the above error bound remain $\mathcal{O} \left(1\right)$ (which is observed in practice), the error decreases by a factor of $\epsilon^{1/2}$ after each iteration.

Proof. Following the proof of Theorem 1 in [17], we obtain the bound

$$
\begin{aligned}
\left\| \mathbf{u} \left(T_n\right) - \mathbf{U}_n^k \right\| & \leq & \epsilon C_2 \frac{\left(C_1 \epsilon\right)^k}{(k+1)!} \left(1 + C_0 \Delta T\right)^{n-k-1} n^k \\
& \leq & \epsilon C_2 \frac{\left(C_1 n \epsilon\right)^k}{(k+1)!} e^{C_0 \left(T_n - T_{k+1}\right)} \\
& \leq & \epsilon^{k/2+1} C_2 \frac{\left(C_1\right)^k}{(k+1)!} e^{C_0 \left(T_n - T_{k+1}\right)}.
\end{aligned}
$$

where the second inequality used that $1 + x \leq e^x$, and the third inequality used that $n\epsilon \leq N\epsilon \leq \epsilon^{1/2}$ (recall the number N of big time steps is given by $N = \epsilon^{-1/2}$).

$\square$

6. Numerical examples

6.1. **Rotating Shallow Water Equations.** We consider as a test problem the non-dimensional rotating shallow water equations (RSW) equations,

$$
\begin{aligned}
\frac{\partial v_1}{\partial t} + \frac{1}{\epsilon}\left(-v_2 + F^{-1/2}\frac{\partial h}{\partial x}\right) + v_1\frac{\partial v_1}{\partial x} &= \mu\partial_x^4 v_1, \\
\frac{\partial v_2}{\partial t} + \frac{1}{\epsilon}v_1 + v_1\frac{\partial v_2}{\partial x} &= \mu\partial_x^4 v_2, \\
\frac{\partial h}{\partial t} + \frac{F^{-1/2}}{\epsilon}\frac{\partial v_1}{\partial x} + \frac{\partial}{\partial x}(hv_1) &= \mu\partial_x^4 h,
\end{aligned}
$$

(6.1)

with spatially periodic boundary conditions on the interval $[0, 2\pi]$. Here $h(x,t)$ denotes the surface height of the fluid, and $v_1(x,t)$ and $v_2(x,t)$ denote the horizontal fluid velocities. The non-dimensional parameter ϵ denotes the Rossby number (a ratio of the characteristic advection time to the rotation time), and is often small (e.g. 10^{-2}) in realistic oceanic flows. The non-dimensional parameter $F^{1/2}\epsilon$ gives the Froude number (a ratio of the characteristic fluid velocity to the gravity wave speed), where $F = \mathcal{O}(1)$ is a free parameter which we set to unity in our subsequent calculations. The scaling we have taken is for quasi-geostrophic dynamics (cf. [28]), which governs the fluid flow dominated by strong stratification and strong constant rotation. As is standard, we also introduce a hyperviscosity operator $\mu\partial_x^4$ to prevent singularities from forming; using hyperviscosity also ensures that the low frequencies are less effected by dissipation than the high frequencies. The RSW equations represent a standard framework in which to develop and test new numerical algorithms for geophysical fluid applications.

To relate equations (6.1) to the abstract formulation (1.1), we define

$$
\mathbf{u}(t,\mathbf{x}) = \begin{pmatrix} v_1(t,x) \\ v_2(t,x) \\ h(t,x) \end{pmatrix}.
$$

Then the system (6.1) can be written in the form (1.1) by setting

$$
\mathcal{L} = \begin{pmatrix} 0 & -1 & F^{-1/2}\partial_x \\ 1 & 0 & 0 \\ F^{-1/2}\partial_x & 0 & 0 \end{pmatrix}, \quad \mathcal{D} = \mu\partial_x^4 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathcal{N}(\mathbf{u}) = \begin{pmatrix} v_1(v_1)_x \\ v_1(v_2)_x \\ (hv_1)_x \end{pmatrix}.
$$

Because we work in a periodic domain, it is also convenient to consider (1.1) and (1.2) in the Fourier domain. This will also make explicit the time-averaging in (1.2) (periodicity is not required for our approach, and is only used to simplify the numerical scheme). A straightforward calculation (see [28]) shows that

$$
\mathcal{L}\left(e^{ikx}\mathbf{r}_k^\alpha\right) = i\omega_k e^{ikx}\mathbf{r}_k^\alpha, \quad \omega_k^\alpha = \alpha\sqrt{1 + F^{-1}k^2},
$$

where $\mathbf{r}_k^\alpha$ is a vector that depends on the wavenumber k and $\alpha = -1, 0, 1$. Therefore, by expanding the function $\overline{\mathbf{u}}(t)$ in the basis of eigenfunctions for $\mathcal{L}$, we have that

$$e^{\tau\mathcal{L}}\overline{\mathbf{u}}(t) = \sum_{k\in\mathbb{Z}} e^{ikx} \sum_{\alpha=-1}^{1} e^{i\omega_k^\alpha} u_k^\alpha \mathbf{r}_k^\alpha.$$

As shown in e.g. [27], the nonlinear term $e^{s\mathcal{L}}\mathcal{N}\left(e^{-s\mathcal{L}}\overline{\mathbf{u}}(t)\right)$ in (1.1) can be written in the form

$$(6.2) \qquad \sum_k e^{ikx} \left(\sum_{k_1+k_2=k} \sum_{\alpha_1,\alpha_2} e^{i\left(\omega_k^\alpha - \omega_{k_1}^{\alpha_1} - \omega_{k_2}^{\alpha_2}\right)s} C_{k,k_1,k_2}^{\alpha,\alpha_1,\alpha_2} u_{k_1}^{\alpha_1}(t) u_{k_2}^{\alpha_2}(t) \right) \mathbf{r}_k^\alpha.$$

The interaction coefficients $C_{k,k_1,k_2}^{\alpha,\alpha_1,\alpha_2}$ are explicitly given in e.g. [27]. From (6.2) it is clear that the time average defining $\overline{\mathcal{N}}(\overline{\mathbf{u}}(t))$ in (1.3) retains only three-wave resonances. Indeed, since

$$\lim_{\mathrm{T}\to\infty} \frac{1}{T} \int_0^T e^{i\left(\omega_k^\alpha - \omega_{k_1}^{\alpha_1} - \omega_{k_2}^{\alpha_2}\right)s} ds = \begin{cases} 0, & \omega_k^\alpha - \omega_{k_1}^{\alpha_1} - \omega_{k_2}^{\alpha_2} \neq 0 \\ 1, & \omega_k^\alpha - \omega_{k_1}^{\alpha_1} - \omega_{k_2}^{\alpha_2} = 0, \end{cases}$$

we see that the time average is given by

$$(6.3) \qquad \overline{\mathcal{N}}(\overline{\mathbf{u}}(t)) = \sum_k e^{ikx} \left(\sum_{(k,k_1,k_2,\alpha,\alpha_1,\alpha_2)\in S_\mathrm{r}} C_{k,k_1,k_2}^{\alpha,\alpha_1,\alpha_2} u_{k_1}^{\alpha_1}(t) u_{k_2}^{\alpha_2}(t) \right) \mathbf{r}_k^\alpha.$$

Here the resonant set S_r is defined by

$$S_\mathrm{r} = \left\{ (k, k_1, k_2, \alpha, \alpha_1, \alpha_2) \mid k_1 + k_2 = k \ \text{ and } \ \omega_k^\alpha - \omega_{k_1}^{\alpha_1} - \omega_{k_2}^{\alpha_2} = 0 \right\}.$$

The HMM (outlined in Section 2) allows the resonant terms in (6.3) to be efficiently computed.

6.2. **Numerical experiments on the RSW equations.** We solve equations (6.1) with the initial conditions consisting of $v_1(0, x) = v_2(0, x) = 0$ and

$$(6.4) \quad h(x, 0) = c_1 \left(e^{-4(x-\pi/2)^2} \sin(3(x-\pi/2)) + e^{-2(x-\pi)^2} \sin(8(x-\pi)) \right) + c_0,$$

where c_1 and c_2 are chosen so that

$$\int_0^{2\pi} h(0, x)\, dx = 0, \quad \max_x |h(x, 0)| = 1.$$

We also choose the viscosity parameter $\mu = 10^{-4}$, and the values $\epsilon = 10^{-2}$, $\epsilon = 10^{-1}$, and $\epsilon = 1$ (corresponding to strong, weak, and no scale separation), which are physically realistic values in geophysical ocean flows; smaller values of ϵ would yield even greater parallel speedup, but are less physically relevant. Although the choice of the initial height $h(x, 0)$ in (6.4) is somewhat arbitrary, the conclusions in this section appear to be insensitive to the initial conditions (as long as they are sufficiently smooth). For all three choices of ϵ, we perform the following numerical experiments. First, we compute the solution using the asymptotic parareal method outlined in Section 3. We then compare the estimated parallel speedup with the results of using three numerical integrators in serial: exponential time differencing 4th order Runge-Kutta (ETDRK4) [9], the operator integrating factor method (cf. [20]), and Strang splitting (see Algorithm 3 of Section 3). Finally, for comparison we compute the solution using the standard parareal method by solving the full

equation (1.1) using big and small step sizes ΔT and Δt, where we use Strang splitting for the coarse solver (we find that using ETDRK4 or OIFS as a coarse solver yields similar parallel speedup). For the sake of simplicity, we also assume that the cost of computing a single time step using ETDRK4, OIFS, and Strang splitting is the same; a more careful analysis that takes into account the number of operations required for each integrator would yield greater parallel speedup.

As we show below, with the coarse time step $\Delta T \geq 3/10$, we find similar convergence and accuracy in the asymptotic parareal method for all values of ϵ. The parallel speedup for $\epsilon = 10^{-2}$ is about a factor of 100 relative to using ETDRK4, the integrating factor method, or Strang splitting in serial. In contrast, the standard parareal method (using ETDKRK4, the integrating factor method, or Strang splitting as a coarse solver) requires much smaller time steps, resulting in a parallel speedup that is about 10 times smaller for $\epsilon = 10^{-2}$.

In our first numerical experiment, we take $\epsilon = 10^{-2}$. For the asymptotic parareal method, we use a coarse time step $\Delta T = 50 \times \epsilon = 1/2$, a fine time step $\Delta t = \epsilon/25 = 1/2500$, and $N = \Delta T/\Delta t = 1250$ time intervals $[(n-1)\Delta T, n\Delta T]$. Even though the solution executes many temporal oscillations within the time intervals $[n\Delta T, (n+1)\Delta T]$—see Figure 6.1 for a plot of the 4th Fourier coefficient $\hat{h}(k,t)$ for $0 \leq t \leq \Delta T$—the method converges in a small number of iterations. In fact, Figure 6.2 shows the maximum relative L^∞ error,

$$\max_{0 \leq n \leq N} \left\| \mathbf{U}_n^k - \mathbf{u}(n\Delta T, \cdot) \right\|_\infty / \left\| \mathbf{u}(n\Delta T, \cdot) \right\|_\infty,$$

as a function of the iteration levels $k = 0, 1, \ldots, 5$. Comparing the parallel speedup relative to directly integrating the full equation (1.1) using ETDRK4, the integrating factor method, and Strang splitting, we find that we need step sizes $\Delta t = \epsilon/25$, $\Delta t = \epsilon/20$, and $\Delta t = \epsilon/20$, respectively, in order to obtain a comparable accuracy. Therefore, using the complexity analysis in Section 5, we expect a parallel speedup of

$$\frac{N(\Delta T/\Delta t)}{5((\Delta T/\Delta t) + N) + N} \approx 110.$$

To constrast this speedup with that obtained using a standard version of the parareal method, we show in Figure 6.2 the results obtained via a coarse solver based on solving the full equation (1.1) using Strang splitting (the results are similar with using ETDRK4 or the integrating factor method as coarse solvers). In this experiment, we take $\Delta t = \epsilon/25 = 1/2500$, $\Delta T = 4\epsilon = 1/25$, and $N = \Delta T/\Delta t = 100$, which yields an expected parallel speedup that is 10 times smaller.

In our second experiment, we take $\epsilon = 10^{-1}$, corresponding to weak scale separation. In Figure 6.3, we show the maximum relative L^∞ error as a function of the iteration levels $k = 0, 1, \ldots, 5$.. Here we used a coarse time step $\Delta T = 3\epsilon = 3/10$, a fine time step $\Delta t = \epsilon/25 = 1/250$, and $N = 150$ time intervals $[(n-1)\Delta T, n\Delta T]$. Comparing the parallel speedup relative to directly integrating the full equation (1.1) using ETDRK4, integrating factor method, and Strang splitting, we find that we need time steps $\Delta t = \epsilon/20 = 1/200$, $\Delta t = \epsilon/10 = 1/100$, and $\Delta t = \epsilon/20 = 1/200$, respectively, in order to obtain a comparable accuracy. Thus, we obtain an estimated parallel speed of about $1/6(\Delta T/\Delta t) = 13$ relative to using Strang splitting in serial; taking into account the number of operations required for each time step in ETDRK4 and the integrating factor method, the estimated parallel speedup relative to these integrators will be comparable. In contrast, plot (b) of Figure 6.3

FIGURE 6.1. Plot of the 4th Fourier coefficient $\hat{h}(k, t)$ as a function of time $0 \leq t \leq \Delta T$, for $\epsilon = 10^{-2}$
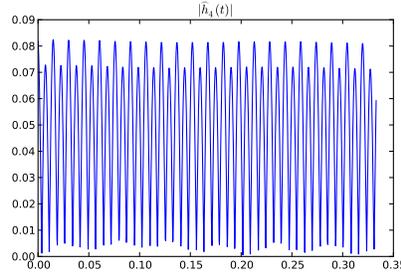
FIGURE 6.2. Maximum relative L^∞ error (on a $\log_{10}$ scale), $\max_{0 \leq n \leq N} \left\| \mathbf{U}_n^k - \mathbf{u}(n\Delta T, \cdot) \right\|_\infty / \left\| \mathbf{u}(n\Delta T, \cdot) \right\|_\infty$, as a function of the iteration level k; the initial condition (6.4) is used, and $\epsilon = 10^{-2}$. The solid green line depicts the errors from the asymptotic parareal method (Parareal-HMM) with a coarse time step $\Delta T = 50\epsilon$, and the dashed red and dashed-dotted blue lines depict the errors from the standard parareal method using Strang splitting (Parareal-Strang) with $\Delta T = 4\epsilon$ and $\Delta T = 5\epsilon$, respectively. A fine time step $\Delta t = \epsilon/25$ is used for all three cases.
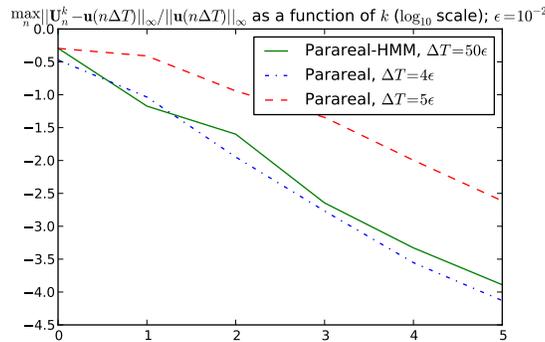
shows the relative L^∞ errors, where this time the standard parareal algorithm is used with Strang splitting for the coarse and fine solvers, and using the same step sizes $\Delta T = \epsilon = 1/20$ and $\Delta t = \epsilon/20 = 1/200$ (again, similar speedup is obtained with ETDRK4 and the integrating factor method). Here the estimated parallel speedup is 3 times smaller.

In our final experiment, we take $\epsilon = 1$, corresponding to no scale separation. In Figure 6.4, we show the relative L^∞ error for the asymptotic parareal method, where we use a coarse time step $\Delta T = 3/10$, a fine time step $\Delta t = 1/200$, and $N = 60$ time intervals $[(n-1)\Delta T, n\Delta T]$. Comparing the parallel speedup relative to directly integrating the full equation (1.1), we obtain an estimated parallel speedup of about

FIGURE 6.3. Maximum relative L^∞ error (on a $\log_{10}$ scale), $\max_{0 \le n \le N} \left\| \mathbf{U}_n^k - \mathbf{u}\left(n\Delta T, \cdot\right) \right\|_\infty / \left\| \mathbf{u}\left(n\Delta T, \cdot\right) \right\|_\infty$, as a function of the iteration level k; the initial condition (6.4) is used, and $\epsilon = 10^{-1}$. The solid green line depicts the errors from the asymptotic parareal method (Parareal-HMM) with a coarse time step $\Delta T = 3\epsilon$, and the dashed red and dashed-dotted blue lines depict the errors from the standard parareal method using Strang splitting (Parareal-Strang) with $\Delta T = \epsilon$ and $\Delta T = 2\epsilon$, respectively. A fine time step $\Delta t = \epsilon/50$ is used for all three cases.
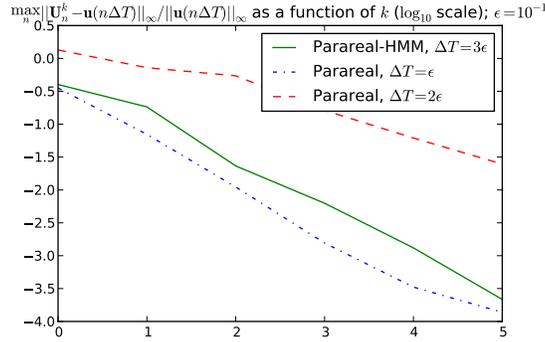
FIGURE 6.4. Maximum relative L^∞ error (on a $\log_{10}$ scale), $\max_{0 \le n \le N} \left\| \mathbf{U}_n^k - \mathbf{u}\left(n\Delta T, \cdot\right) \right\|_\infty / \left\| \mathbf{u}\left(n\Delta T, \cdot\right) \right\|_\infty$, as a function of the iteration level k; the initial condition (6.4) is used, and $\epsilon = 1$. The solid green line depicts the errors from the asymptotic parareal method (Parareal-HMM) with a coarse time step $\Delta T = 3/10\epsilon$, and the dashed red and dashed-dotted blue lines depict the errors from the standard parareal method using Strang splitting (Parareal-Strang) with $\Delta T = 3/10\epsilon$ and $\Delta T = 4/10\epsilon$, respectively. A fine time step $\Delta t = \epsilon/200$ is used for all three cases.
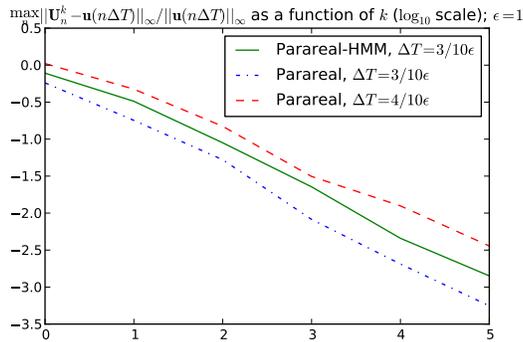
a factor of 5; the speedup with the standard parareal method (and using Strang splitting for a coarse solver) is about the same.

7. Summary

In this paper we have introduced an asymptotic-parallel-in-time method for solving highly oscillatory PDEs. The method is a modification of the parareal algorithm introduced by [24]. The modification replaces the coarse solver used in [24] by a numerically computed locally asymtotic solution based on the asymptotic mathematical structure of the equations ([32], [27], [28], [36]) and concepts used in HMM.

In addition to presenting the method we also include psuedo code. We discuss the performance of the method when $\epsilon = 1$, which is important for using the method in realistic simulations where the time scale separation may vary in space and time. We also present a complexity analysis that shows that the parallel speed-up increases as ϵ decreases which results in an arbitrarily greater efficiency gain relative to standard numerical integrators and a Theorem following [17] that shows that as long as the constants remain bounded the error decreases by a factor of $\epsilon^{1/2}$ after each iteration.

We also present numerical experiments for the shallow water equations. These results demonstrate that the parallel speedup is more than 100 relative to exponential integrators such as ETDRK4 (for realistic parameter values in the shallow water equations); the speedup is also more than 10 relative to using the standard parareal method with a linearly exact coarse solver. Finally, the results demonstrate that the method works in the absence of scale separation, and with as much speedup as the standard parareal method.

References

[1] G. Ariel, B. Engquist, S. Kim, Y. Lee, and R. Tsai. A multiscale method for highly oscillatory dynamical systems using a poincaré map type technique. *Journal of Scientific Computing*, 54:247–268, 2013.

[2] Gil Ariel, Bjorn Engquist, and Richard Tsai. A multiscale method for highly oscillatory ordinary differential equations with resonance. *Math. Comp.*, 78(266):929–956, 2009.

[3] Gil Ariel, Björn Engquist, Heinz-Otto Kreiss, and Richard Tsai. Multiscale computations for highly oscillatory problems. In Björn Engquist, Per Lötstedt, and Olof Runborg, editors, *Multiscale Modeling and Simulation in Science*, volume 66 of *Lecture Notes in Computational Science and Engineering*, pages 237–287. Springer Berlin Heidelberg, 2009.

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

[5] Guillaume Bal. On the convergence and the stability of the parareal algorithm to solve partial differential equations. In *Domain Decomposition Methods in Science and Engineering*, volume 40 of *Lecture Notes in Computational Science and Engineering*, pages 425–432. Springer Berlin Heidelberg, 2005.

[6] F. Castella, P. Chartier, and E. Faou. An averaging technique for highly oscillatory Hamiltonian problems. *SIAM J. Numer. Anal.*, 47(4):2808–2837, 2009.

[7] J.G. Charney. On the scale of atmospheric motions. *Geofysiske Publikasjoner*, 17(2):3–17, 1948.

[8] J.G. Charney. On a physical basis for numerical prediction of large-scale motions in the atmosphere. *Journal of Meterology*, 6:371–385, 1949.

[9] S.M. Cox and P.C. Matthews. Exponential time differencing for stiff systems. *Journal of Computational Physics*, 176(2):430 – 455, 2002.

[10] Xiaoying Dai and Yvon Maday. STABLE PARAREAL IN TIME METHOD FOR FIRST AND SECOND.

[11] Weinan E. Analysis of the heterogeneous multiscale method for ordinary differential equations. *Commun. Math. Sci.*, 1(3):423–436, 2003.

[12] S. Engblom. Parallel in time simulation of multiscale stochastic chemical kinetics. *Multiscale Modeling and Simulation*, 8(1):46–68, 2009.

[13] Bjorn Engquist and Yen-Hsi Tsai. Heterogeneous multiscale methods for stiff ordinary differential equations. *Mathematics of Computation*, 74(252):pp. 1707–1742, 2005.

[14] Charbel Farhat and Marion Chandesris. Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid?structure applications. *International Journal for Numerical Methods in Engineering*, 58(9):1397–1434, 2003.

[15] Paul F. Fischer, Frédéric Hecht, and Yvon Maday. A parareal in time semi-implicit approximation of the navier-stokes equations. In *Proceedings of Fifteen International Conference on Domain Decomposition Methods*, pages 433–440. Springer Verlag, 2004.

[16] Martin J. Gander. Analysis of the parareal algorithm applied to hyperbolic problems using characteristics. *Bol. Soc. Esp. Mat. Apl. SẽMA*, (42):21–35, 2008.

[17] Martin J Gander and Ernst Hairer. Nonlinear Convergence Analysis for the Parareal Algorithm.

[18] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration*, volume 31 of *Springer Series in Computational Mathematics*. Springer, Heidelberg, 2010. Structure-preserving algorithms for ordinary differential equations, Reprint of the second (2006) edition.

[19] D.a. Jones, a. Mahalov, and B. Nicolaenko. A Numerical Study of an Operator Splitting Method for Rotating Flows with Large Ageostrophic Initial Data. *Theoretical and Computational Fluid Dynamics*, 13(2):143, 1999.

[20] Aly khan Kassam, Lloyd, and N. Trefethen. Fourth-order time stepping for stiff pdes. *SIAM J. Sci. Comput*, 26:1214–1233, 2005.

[21] J. Douglas Lawson. Generalized runge-kutta processes for stable systems with large lipschitz constants. *SIAM Journal on Numerical Analysis*, 4(3):pp. 372–380, 1967.

[22] F. Legoll, T. Lelievre, and G. Samaey. A micro-macro parareal algorithm: application to singularly perturbed ordinary differential equations. *ArXiv e-prints*, April 2012.

[23] Frédéric Legoll, Xiaoying Dai, Claude Le Bris, and Yvon Maday. Symmetric parareal algorithms for Hamiltonian systems. *ESAIM: Mathematical Modelling and Numerical Analysis*, pages 1–56, September 2012.

[24] Jacques-Louis Lions, Yvon Maday, and Gabriel Turinici. Résolution d'{EDP} par un schéma en temps "pararéel". *C. R. Acad. Sci. Paris Sér. I Math.*, 332(7):661–668, 2001.

[25] Yvon Maday. Parareal in time algorithm for kinetic systems based on model reduction. In *High-dimensional partial differential equations in science and engineering*, volume 41 of *CRM Proc. Lecture Notes*, pages 183–194. Amer. Math. Soc., Providence, RI, 2007.

[26] Yvon Maday and Gabriel Turinici. Parallel in time algorithms for quantum control: Parareal time discretization scheme. *International Journal of Quantum Chemistry*, 93(3):223–228, 2003.

[27] Andrew Majda. *Introduction to {PDE}s and waves for the atmosphere and ocean*, volume 9 of *Courant Lecture Notes in Mathematics*. New York University Courant Institute of Mathematical Sciences, New York, 2003.

[28] Andrew J. Majda and Pedro Embid. Averaging over Fast Gravity Waves for Geophysical Flows with Unbalanced Initial Data. *Theoretical and Computational Fluid Dynamics*, 11(3-4):155–169, June 1998.

[29] B Nadiga, Matthew Hecht, L Margolin, and Piotr Smolarkiewicz. On simulating flows with multiple time scales using a method of averages. *Theoretical and Computational Fluid Dynamics*, 9(3-4):281–292, 1997.

[30] Sebastian Reich. Smoothed dynamics of highly oscillatory Hamiltonian systems, December 1995.

[31] J. A. Sanders and F. Verhulst. *Averaging methods in nonlinear dynamical systems / J.A. Sanders, F. Verhulst*. Springer-Verlag, New York:, 1985.

[32] S. Schochet. Fast singular limits of hyperbolic pde's. *Journal of Differential Equations*, 114:476–512, 1994.

[33] Leslie M. Smith and Youngsuk Lee. On near resonances and symmetry breaking in forced rotating flows at moderate Rossby number. *Journal of Fluid Mechanics*, 535(2005):111–142, July 2005.

[34] G. A. Staff. The parareal algorithm. a survey of present work. Technical report, Norwegian University of Science and Technology, Dept. of Math. Sciences.

[35] E. Weinan, Bjorn Engquist, Xiantao Li, Weiqing Ren, and Eric Vanden-Eijnden. Heterogeneous multiscale methods: A review. *Communications in Computational Physics*, 2(3):367–450, June 2007.

[36] Beth a. Wingate, Pedro Embid, Miranda Holmes-Cerfon, and Mark a. Taylor. Low Rossby limiting dynamics for stably stratified flow with finite Froude number. *Journal of Fluid Mechanics*, 676(2011):546–571, April 2011.