Multigrid-Reduction-in-Time for the Rotating Shallow Water Equations

Nicholas Abel[*] Jehanzeb Chaudhry[†] Robert D. Falgout[‡] Jacob Schroder[§]

Abstract

We consider multilevel time-parallel methods for the numerical solution of the rotating shallow water equations. In particular, the multigrid-reduction-in-time (MGRIT) algorithm is used for the parallel time integration. An asymptotic model is used at the coarse levels while the full model is employed at the finer levels. The asymptotic model is well-suited for highly oscillatory partial differential equations like the rotating shallow water equations because it can accurately and stably take the required large time-steps on coarse levels. Our work exploits the flexibility of the MGRIT algorithm in terms of the number of levels and relaxation schemes to show some computational benefits, especially with respect to FCF-relaxation and data reuse.

1 Introduction

As the clock speeds of individual processors level off, but the amount of parallel resources continue to increase rapidly, further exploitation of parallelism is necessary to speed up the numerical solution of partial differential equations (PDEs) [21]. Parallelism in the spatial component has been long exploited through domain decomposition and other similar techniques [14]. However, as the number of processors increase, the spatial parallelism is reaching its efficiency limits and there has been great interest in exploiting parallelism in the time dimension. A historical overview of parallel-in-time methods is discussed in [14], and a list of many recent application areas is discussed in [34].

In this work, we consider the time-parallel multigrid-reduction-in-time (MGRIT) method [13, 10, 5]. Another widely used method is the two-level parareal method [23, 16], however, MGRIT allows for additional flexibility by appealing to the well-developed field of multigrid. MGRIT decomposes the time domain of a problem into subdomains over which the solution is iteratively improved using error correction terms which come from inexpensive coarse grid approximations to the problem. Each subdomain is updated using a local coarse grid correction term simultaneously, hence the parallelism in time. In contrast to serial time-stepping methods which solve the space-time equations from an initial time to a desired final time in serial, MGRIT improves an initial solution guess across the entire space-time domain at each iteration.

Parallel-in-time techniques have been very successful in speeding up the computation of parabolic problems; however, for hyperbolic problems that are advection-dominated, the performance of parallel-in-time methods deteriorates greatly when using traditional choices for the coarse-grid approximation. This phenomenon is documented in the parareal setting for the linear advection equation and second-order wave equation [15, 32], as well as the Navier-Stokes equation, where convergence improves as the Reynolds number becomes small, making the problem more diffusive [35]. Similarly, MGRIT has been shown to struggle with the linear advection and inviscid Burgers equations, likewise improving with the introduction of diffusivity terms [19, 5, 4, 36]. Ruprecht shows in [32] that under the standard parareal method, instabilities for the linear advection equation are due to phase errors in the coarse grid representation. Much work has been done investigating alternative coarse grid rediscretizations for hyperbolic problems for use with

[*] Department of Mathematics and Statistics, University of New Mexico

[†] Department of Mathematics and Statistics, University of New Mexico

[‡] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, P.O. Box 808, L-561, Livermore,CA 94551. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-TR-813511

[§] Department of Mathematics and Statistics, University of New Mexico parareal and MGRIT. In [36, 20], a technique that constructs a coarse grid propagator by approximating the spectrum of the ideal MGRIT coarse grid operator using a weighted least squares approximation is proposed for the linear advection equation. The use of a semi-Lagrangian scheme as a parareal coarse operator for the viscous Burgers equation is investigated in [33]. Nielsen et al. [27] use Roe's method, an approximate Riemann solver, on the coarse grid combined with a third order WENO scheme on the fine grid for the spatial discretization and third-order Strong-Stability-Preserving Explicit Runge-Kutta time integration to achieve convergence for the two dimensional shallow water equations. However, in all these cases, the convergence of parareal and MGRIT is either slow for strongly hyperbolic problems, or the method is yet to be generalized to variable coefficient nonlinear problems [36, 20].

Thus, the asymptotic parareal method [17, 30, 29] was designed for strongly hyperbolic problems with fast temporal oscillations due to a linear term and slow characteristic advection due to a nonlinear term. A typical example of such PDEs are the rotating shallow water equations. Here, asymptotic parareal achieves superior convergence to standard parareal in highly oscillatory flow regimes and has competitive performance to standard parareal in non-oscillatory flow regimes. In this work, we investigate the asymptotic parareal method when used with the MGRIT algorithm. We design an asymptotic MGRIT algorithm by generalizing the asymptotic parareal method and investigate the use of FCF-relaxation, data reuse, and various multilevel strategies to achieve higher efficiency.

The rest of the article is organized as follows. We review the MGRIT method in §2 and the rotating shallow water equations in §3. The algorithm for asymptotic MGRIT is presented in §4. Numerical experiments are carried out in §5. Conclusions and future directions are discussed in §6.

2 The parallel Multigrid-in-time algorithm

MGRIT is an algorithm for parallel time integration that applies the technique of multigrid reduction (MGR) [31] to iteratively compute the solution of a time-dependent differential equation. To handle nonlinear systems, MGRIT uses the nonlinear full approximation scheme (FAS) [2] to restructure the coarse grid. The widely used parareal algorithm [23] is a parallel-in-time method which is equivalent to a two-level MGRIT solve using F-relaxation [10].

We first present the two-grid MGRIT algorithm. The generalization to multiple levels is intuitive. Consider an ODE system

$$\mathbf{u}'(t) = \mathbf{f}(t, \mathbf{u}(t)), \mathbf{u}(0) = \mathbf{u}_0, t \in [0, T], \tag{1}$$

with a time discretization represented by the points

$$t_i = i\Delta t,\ \ i = 0, 1, \cdots, N, \tag{2}$$

where $\Delta t = \frac{T}{N}$, and $u_i :\approx \mathbf{u}(t_i)$. In the context of PDEs, the system (1) corresponds to a method of lines discretization. Using a one-step method, the discretization takes the form

$$\begin{aligned} u_0 &= g_0, \\ u_i &= \Phi_i(u_{i-1}) + g_i,\ \ i = 1, 2, \cdots, N, \end{aligned} \tag{3}$$

where g_i is a forcing term evaluated at time-step i and Φ_i is the time-stepping operator which evolves the state variable u from time-step $i-1$ to time-step i.
Thus, we formally write (3) as

$$Au = \begin{pmatrix} I & & & \\ -\Phi_1 & I & & \\ & \ddots & \ddots & \\ & & -\Phi_N & I \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \end{pmatrix} = \begin{pmatrix} g_0 \\ g_1 \\ \vdots \\ g_N \end{pmatrix} = g, \tag{4}$$

where the product of Φ with u_i is interpreted as $\Phi(u_i)$. Sequential time integration of (3) is equivalent to directly solving the block lower bidiagonal system (4), whereas using the MGRIT algorithm is equivalent to solving Equation (4) iteratively, using block-Jacobi relaxation with error corrections obtained from coarse approximations to the system.

For simplicity, we assume here that the time-stepping operator Φ is constant, so that $\Phi_i = \Phi$, $i \in [1, 2, \cdots, N]$. To create a coarse approximation to A for the purposes of computing error correction terms, we construct a coarse time grid consisting of $N_\Delta + 1 = \frac{N}{m} + 1$ time points with a correspondingly larger time-step $\Delta T = m\Delta t$, where m is an integer called the **coarsening factor**. Without loss of generality, we assume that N is divisible by m. The coarse time grid is then represented by the points

$$T_i = i\Delta T = i(m\Delta t) = t_{im},\ i = 0, 1, \cdots, N_\Delta. \tag{5}$$

An illustration of the decomposition of the time grid into coarse and fine grids is given in Figure 1.

A typical choice of Φ_Δ for a parabolic problem is the fine grid operator Φ with a time-step size of $\Delta T = m\Delta t$, instead of the fine grid step size Δt. However for advection-dominated problems without strong diffusive terms, this has been shown not to be a good choice for Φ_Δ. Most research for solving these problems with MGRIT and parareal is focused on choosing Φ_Δ that work well, and is the focus of this work.

The coarse grid approximation takes the form $B_\Delta u_\Delta = \tilde{g}_\Delta$ where

$$B_\Delta = \begin{pmatrix} I & & & \\ -\Phi_\Delta & I & & \\ & \ddots & \ddots & \\ & & -\Phi_\Delta & I \end{pmatrix}, \text{ and } \quad \tilde{g}_{km} = \sum_{j=0}^{m-1} \Phi^j g_{km-j}. \tag{6}$$

Equations (2) and (5) give rise to a partitioning of the time grid into **F-points** denoted F_p and **C-points** denoted by C_p, which are defined as

$$\begin{aligned} F_p &= \{t_i\} \setminus \{T_i\}, \\ C_p &= \{T_i\}. \end{aligned} \tag{7}$$

The partition is depicted in Figure 1.

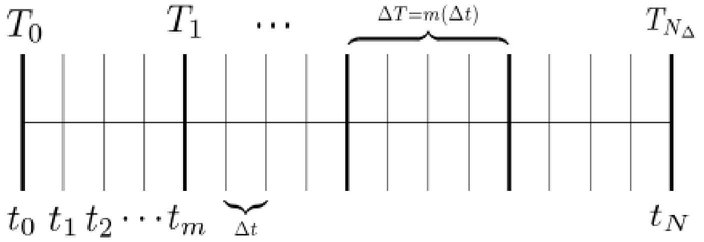

Figure 1: Illustration of a temporal grid decomposed into a fine and coarse grid, uniformly spaced. F-points (thin lines) are only present on the fine grid, and C-points (thick lines) are present on both the fine and coarse grid.

The partitioning in Equation (7) gives rise to the two fundamental types of relaxation in the MGRIT algorithm, **F-relaxation** and **C-relaxation**. F-relaxation consists of applying Φ within each interval of F-points located between two C-points. C-relaxation applies Φ once per C-point, evolving from the F-point immediately preceding the C-point. This is depicted in Figure 2. As shown in Algorithms 1 and 2, the F- and C-relaxation algorithms can be performed in each of the N_Δ coarse intervals in parallel.

Algorithm 1 F-relaxation

```
for $k = 0, 1, \cdots, N_\Delta$ do {Parallel for-loop over $k$}
    for $i = 1, 2, \cdots, m-1$ do {Sequential for-loop over $i$}
        $u_{km+i} \leftarrow \Phi(u_{km+i-1}) + g_{km+i-1}$
    end for
end for
```

An application of F-relaxation, followed by C-relaxation, is equivalent to applying Φ^m to u_{km} for $k = 0, 1, \cdots, N_\Delta$. This FC-relaxation sweep is equivalent to an application of block-Jacobi relaxation on the

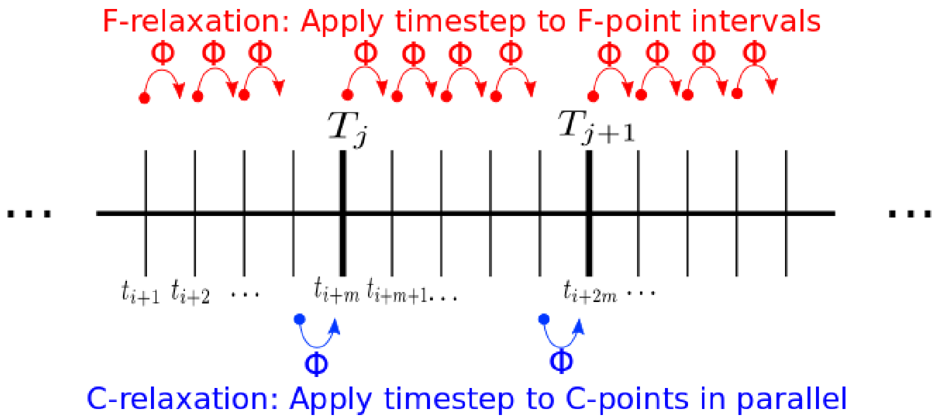

Figure 2: Illustration of F- and C-relaxation. C-points are denoted with T_j, T_{j+1} and F-points are denoted with $t_{i+1}, t_{i+2}, \cdots$.

Algorithm 2 C-relaxation

```
for $k = 0, 1, \cdots, N_\triangle$ do {Parallel for-loop over $k$}
    $u_{km} \leftarrow \Phi(u_{km-1}) + g_{km-1}$
end for
```

coarse grid system. MGRIT allows for the use of F-relaxation or FCF-relaxation smoothing schemes, where FCF-relaxation is an application of an FC-sweep followed by another application of F-relaxation.

For restriction to a coarse grid, MGRIT uses injection at the C-points, $u_k^c \leftarrow u_{km}$, where u^c is the coarse grid vector. Interpolation is injection from the coarse to fine grid, $u_k^c \rightarrow u_{km}$, followed by an F-relaxation. The F-relaxation guarantees that if the solution is exact (i.e., equal to the solution one would obtain from a forward solve of Equation 4) at the C-points u_{km}, then the solution will also be exact at F-points. Thus it is called ideal interpolation.

Putting these pieces together, MGRIT first carries out an F- or FCF-relaxation, followed by restriction of the problem and residual to the coarse-grid. There, the error correction is computed according to the FAS coarse-grid formulation using $B_\triangle$. Lastly, this error correction is interpolated back to the fine grid and added to the current solution guess u and the algorithm repeats iteratively until the 2-norm of the coarse grid residual $r_\triangle$ is smaller than a given halting tolerance *tol*. Pseudocode for the two-grid FAS MGRIT algorithm is given in Algorithm 3.

Algorithm 3 Two-Grid FAS MGRIT($\Phi, \Phi_\triangle, u, g, tol$)

```
1: Apply F- or FCF-relaxation to $Au = g$
2: Compute and restrict fine grid approximation and its residual to the coarse grid via injection: $u_{\triangle,i} \leftarrow u_{mi}$, $r_{\triangle,i} \leftarrow g_{mi} - (Au)_{mi}$
3: Solve $B_\triangle v_\triangle = B_\triangle u_\triangle + r_\triangle$
4: Compute coarse grid error correction: $e_\triangle \leftarrow v_\triangle - u_\triangle$
5: Correct $u$ at C-points: $u_{mi} \leftarrow u_{mi} + e_{\triangle,i}$
6: If $||r_\triangle||_2 < tol$, apply F-relaxation to $Au = g$ and terminate
7: Else, go to step 1
```

3 The Rotating Shallow Water Equations

The Rotating Shallow Water Equations (RSWE) are an important set of equations in geophysical fluid dynamics that is of particular interest in the weather and climate modeling community [17, 7, 25, 24]. The

RSWE in one space dimension are,

$$\begin{aligned}\frac{\partial v_1}{\partial t} + \frac{1}{\epsilon}\left(-v_2 + F^{-1/2}\frac{\partial h}{\partial x}\right) + v_1\frac{\partial v_1}{\partial x} &= \mu\partial_x^4 v_1, \\ \frac{\partial v_2}{\partial t} + \frac{1}{\epsilon}v_1 + v_1\frac{\partial v_2}{\partial x} &= \mu\partial_x^4 v_2, \\ \frac{\partial h}{\partial t} + \frac{F^{-1/2}}{\epsilon}\frac{\partial v_1}{\partial x} + \frac{\partial}{\partial x}(hv_1) &= \mu\partial_x^4 h,\end{aligned} \tag{8}$$

where $h(x,t)$ denotes the surface height of the fluid, and $v_1(x,t)$ and $v_2(x,t)$ denote the horizontal fluid velocities. The quantity v_1 denotes horizontal velocity in the x-direction (along the direction of the spatial domain,) and v_2 denotes horizontal velocity in the y-direction.

The non-dimensional parameter ϵ is the **Rossby number**, which represents the ratio of the rotation timescale to the timescale of horizontal advection. The temporal oscillations in the solution to the RWSE occur on a $\mathcal{O}(\epsilon)$ timescale. Hence, as $\epsilon \to 0$, the frequency of oscillations increases accordingly. In realistic geophysical flows, ϵ is typically small, and a reasonable choice for testing the handling of fast gravity waves is $\epsilon = \mathcal{O}(10^{-2})$. The value μ is the coefficient to a fourth-order hyperviscosity term which is introduced for stability, and is taken here to be 10^{-4} unless otherwise noted. As μ becomes larger, the magnitudes of temporal oscillations decrease as the problem evolves in time and the problem becomes more diffusion-dominated. Figures 3 and 4 depict the effect that different choices of ϵ and μ have on the exact space-time solution.

The quantity $Fr := F^{1/2}\epsilon$ is the **Froude number**, representing the ratio of fluid velocity to gravity wave speed. F is the **Rossby deformation radius**, which is the distance a wave travels before being significantly effected by rotation. Except where otherwise specified, we examine the case where $F = 1$ and hence $Fr = \epsilon$.

These equations are given in a two-dimensional form by Embid and Majda in [7]. The simplified one-dimensional form in (8) is given by Haut and Wingate in [17], and is obtained from the two-dimensional form by assuming that all derivatives in the y-direction are zero. This is done in order to preserve terms which take rotation into account, while still having an essentially 1-dimensional model problem.

4 Multi-grid in time for the Rotating Shallow Water Equations

We form a MGRIT algorithm for the RWSE in this section.

4.1 RWSE as highly oscillatory PDEs

The RWSE are a form of highly oscillatory PDEs with a special form,

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{1}{\epsilon}\mathcal{L}\mathbf{u} = \mathcal{N}(\mathbf{u}) + \mathcal{D}\mathbf{u}, \quad \mathbf{u}(0) = \mathbf{u}_0, \tag{9}$$

where $\mathcal{L}$ is a linear operator with purely imaginary eigenvalues, $\mathcal{N}(\mathbf{u})$ is a nonlinear quadratic polynomial, $\mathcal{D}$ is a dissipation operator, and ϵ is a nondimensional parameter which determines the time scale separation between $\mathcal{L}$ and $\mathcal{N}$. In particular, when ϵ is small enough, the eigenvalues of the linear operator $\mathcal{L}$ inhabit an area of the complex plane distinctly separate from where the eigenvalues of $\mathcal{N}$ and $\mathcal{D}$ are located. As ϵ becomes larger, the eigenvalues of $\mathcal{L}$, $\mathcal{N}$ and $\mathcal{D}$ become collocated on the complex plane, which corresponds to a lack of scale separation. Here, the oscillations from $\mathcal{L}$ mix with the diffusion and nonlinear advection of $\mathcal{D}$ and $\mathcal{N}$.

We write the RWSE in (8) in the general form (9) by letting the state variable

$$\mathbf{u}(t,\mathbf{x}) = \begin{pmatrix} v_1(t,x) \\ v_2(t,x) \\ h(t,x) \end{pmatrix}. \tag{10}$$

Then, the operators from Equation ((9)) are

$$\mathcal{L} = \begin{pmatrix} 0 & -1 & F^{-1/2}\partial_x \\ 1 & 0 & 0 \\ F^{-1/2}\partial_x & 0 & 0 \end{pmatrix}, \quad \mathcal{D} = \mu\partial_x^4\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathcal{N}(\mathbf{u}) = \begin{pmatrix} v_1(v_1)_x \\ v_1(v_2)_x \\ (hv_1)_x \end{pmatrix}. \tag{11}$$

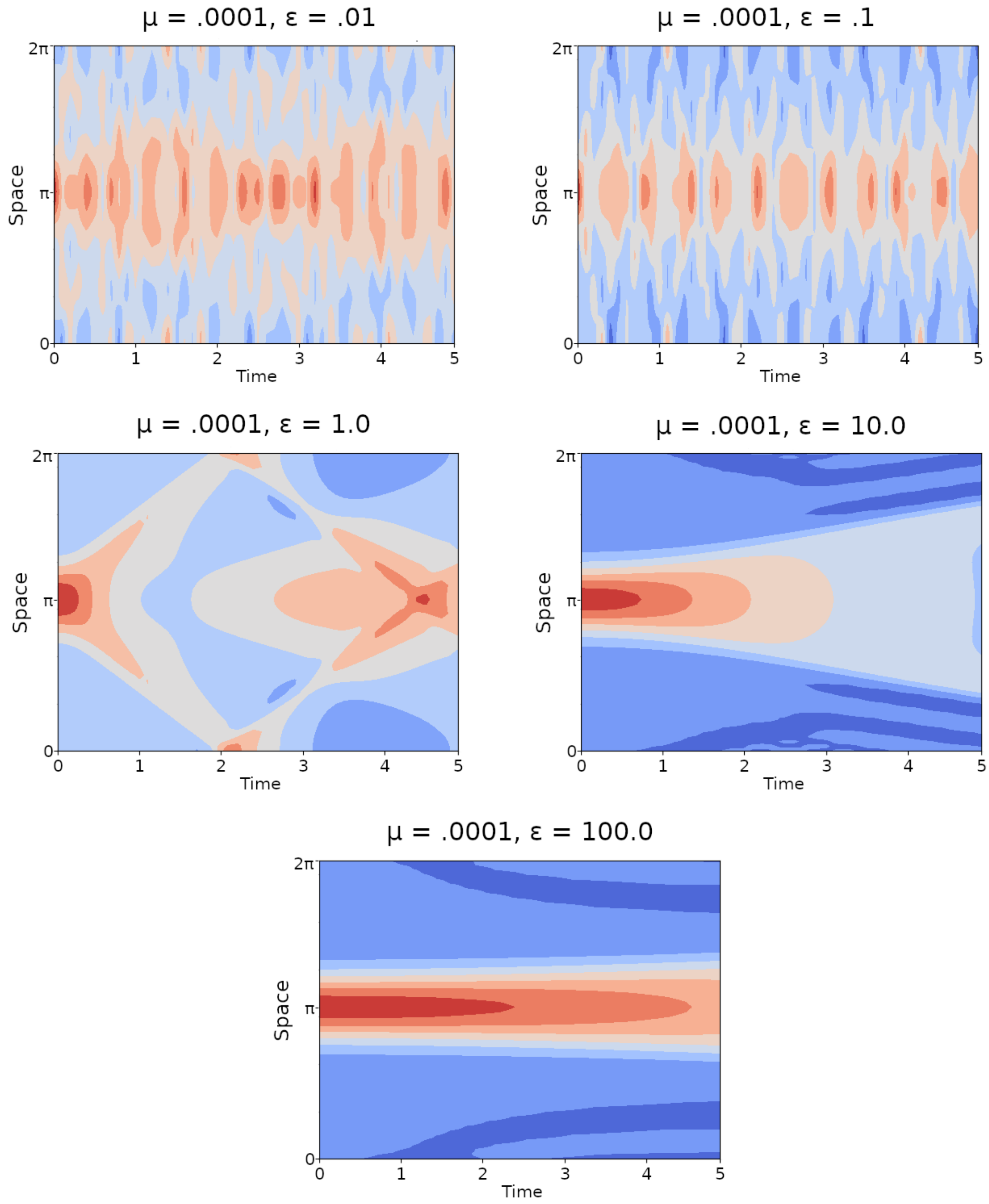

Figure 3: Wave-height solution profile of h in a space-time domain $X = [0, 2\pi], T = [0, 5]$ with $N_x = 64$ Fourier modes in space, $N_\Delta = 50$ time points, various ϵ values, and a fixed $\mu = .0001$. Here, time evolves in the x-axis and the y-axis is the spatial dimension. For small values of ϵ, the magnitude of $\mathcal{L}$ is large and oscillations in the direction of the time domain are rapid. As ϵ becomes large, the magnitude of the $1/\epsilon\mathcal{L}$ term shrinks and oscillations occur over much larger timescale. In particular, when $\epsilon = 10$ and $\epsilon = 100$, a complete temporal oscillation is no longer present because oscillations occur on an $\mathcal{O}(\epsilon)$ timescale. See subfigure titles for specific ϵ, μ values.

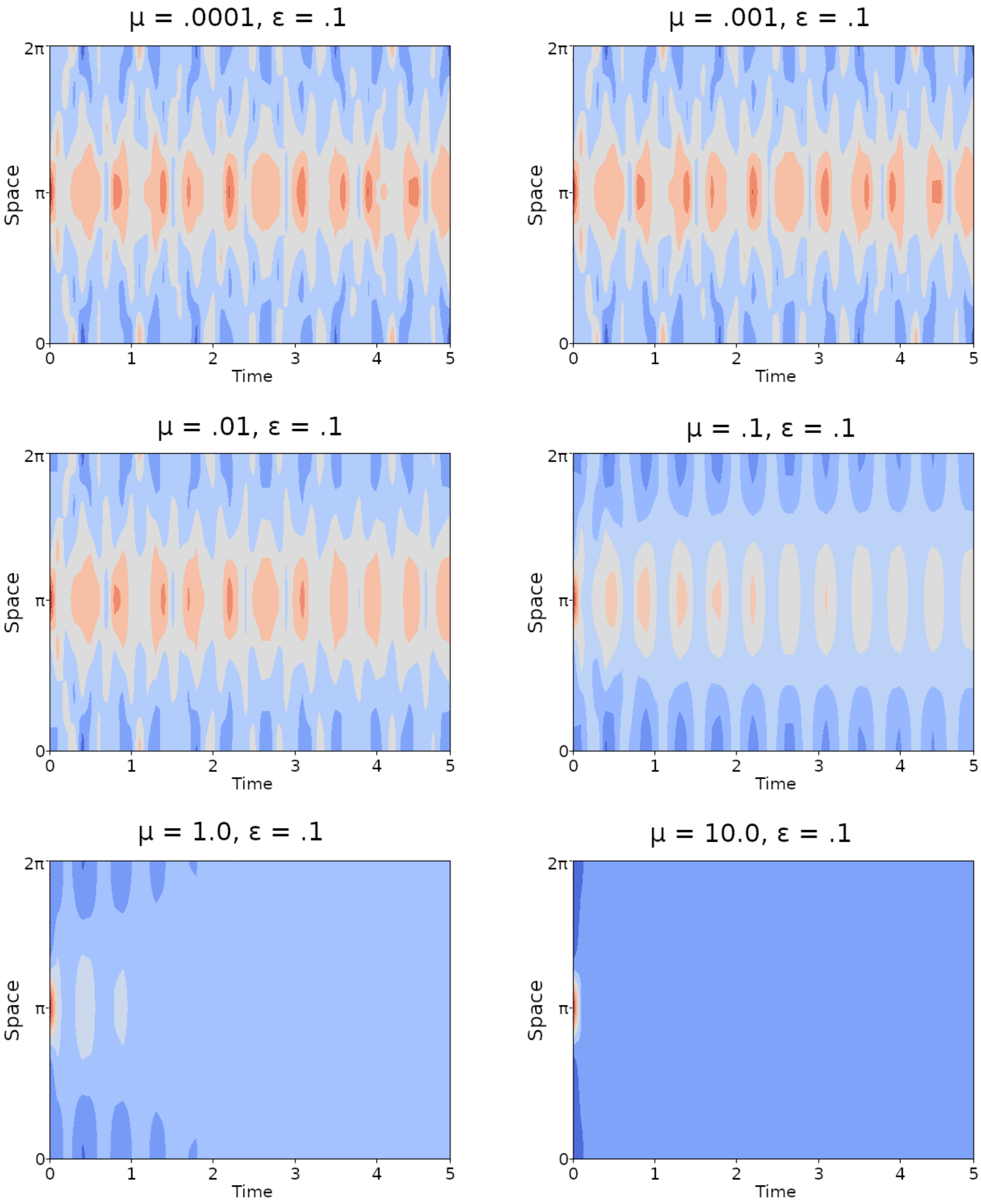

Figure 4: Wave-height solution profile of h in a space-time domain $X = [0, 2\pi], T = [0, 5]$ with $N_x = 64$ Fourier modes in space, $N_\Delta = 50$ time points, various μ values, and a fixed $\epsilon = .1$. Here, time evolves in the x-axis and the y-axis is the spatial dimension. As μ increases, so does the magnitude of the dissipation term. The result is that we see the effects of dissipation become prominent at ever earlier times. See subfigure titles for specific ϵ, μ values.

Indeed, $\mathcal{L}$ is skew-hermitian, i.e., $-\mathcal{L} = (\mathcal{L}^T)^*$ and has purely imaginary eigenvalues with an orthonormal basis of eigenvectors. Thus, $\mathcal{L}$ gives rise to oscillations. Furthermore, we see that $\mathcal{D}$ is a standard hyperviscosity operator and $\mathcal{N}()$ is a nonlinear quadratic term.

4.2 Review of the asymptotic two grid Parareal method

We review the asymptotic parareal method presented in [17]. This time parallel method is designed for hyperbolic problems which take the form (9). Rapid oscillations, due to the linear term $\frac{1}{\epsilon}\mathcal{L}\mathbf{u}$, occur on a timescale of $\mathcal{O}(\epsilon)$. In order to resolve these oscillations, it is necessary to take time-steps of size $\mathcal{O}(\epsilon)$ when using traditional time-stepping methods. Thus, the coarse grid approximation in parareal or MGRIT would also require a time-step size of $\mathcal{O}(\epsilon)$, which would require too fine a coarse grid representation for parareal or MGRIT to be efficient [17]. In summary, using a coarse time-step size in parareal or MGRIT that is $>> \mathcal{O}(\epsilon)$ with standard choices of coarse grid operators will violate the Nyquist sampling rate of the temporal oscillations. This will alias the oscillations on the coarse grid and lead to inaccurate time derivatives. For this reason, problems of the form given by Equation (9) are difficult for parareal and MGRIT to solve efficiently when using traditional coarse grid approximations.

To overcome this, a coarse grid propagator is proposed in [17], that is based on the observation in [25] that the solution $\mathbf{u}(t)$ to Equation (9) has the asymptotic approximation

$$\mathbf{u}(t) = e^{(-t/\epsilon)\mathcal{L}}\bar{\mathbf{u}}(t) + \mathcal{O}(\epsilon), \tag{12}$$

where the asymptotic solution $\bar{\mathbf{u}}(t)$ varies slowly in time and satisfies the equation

$$\frac{\partial \bar{\mathbf{u}}}{\partial t} = \overline{\mathcal{N}}(\bar{\mathbf{u}}) + \overline{\mathcal{D}}\bar{\mathbf{u}}, \quad \bar{\mathbf{u}}(0) = \mathbf{u}_0, \tag{13}$$

and the nonlinear term $\overline{\mathcal{N}}$ is given by the time-averaged quantity

$$\overline{\mathcal{N}}(\bar{\mathbf{u}}(t)) = \lim_{T\to\infty} \frac{1}{T}\int_0^T e^{s\mathcal{L}}\mathcal{N}(e^{-s\mathcal{L}}\bar{\mathbf{u}}(t))ds. \tag{14}$$

The operator $\overline{\mathcal{D}}$ is time-averaged in the same fashion, but note that if D is a linear operator, then $\overline{\mathcal{D}} = \mathcal{D}$. Also note that as opposed to being evolved over the entire time domain $[0, T]$, the coarse grid operator only time-steps $\bar{\mathbf{u}}(t)$ over a coarse time interval $[(j-1)\Delta T, j\Delta T]$. An approximation to $\mathbf{u}(t)$ is then obtained by applying $e^{(-t/\epsilon)\mathcal{L}}$ to $\bar{\mathbf{u}}(t)$. Consequently, time integration for the time-averaged solution $\bar{\mathbf{u}}(t)$ can be done without evolving the oscillatory $e^{(1/\epsilon)\mathcal{L}}$ term. This allows coarse grid time-steps $\Delta T >> \epsilon$ to be taken without violating the Nyquist sampling rate of the rapid temporal oscillations.

A finite approximation to Equation (14) is necessary for numerical evaluation. The finite time average is formally introduced as

$$\overline{\mathcal{N}}(\bar{\mathbf{u}}(t)) \approx \frac{1}{\eta}\int_0^\eta \rho(s/\eta)e^{s\mathcal{L}}\mathcal{N}(e^{-s\mathcal{L}}\bar{\mathbf{u}}(t))ds, \tag{15}$$

where η is referred to as the **averaging window**, and $\rho(s)$ is referred to as the **integrating kernel**. The choice of $\rho(s)$ is investigated thoroughly in [17, 30, 29]. In particular, $\rho(s)$ is chosen to be a Gaussian bump function with compact support in $(0, 1)$ and an integral of unity; i.e:

$$\int_0^1 \rho(s)ds = 1.$$

We compute $\rho(s)$ using the approximation (from [28])

$$\rho(s) \approx \exp(-50 * (s - 0.5)^2).$$

The purpose of introducing the integrating kernel is to increase the accuracy of the approximation to $\overline{\mathcal{N}}(\bar{\mathbf{u}}(t))$, as well as to prevent discontinuities in the solution profile $\bar{\mathbf{u}}(t)$. The use of integrating kernels for these purposes is a standard technique in averaging computations for multiscale methods [9, 8]. This time-averaging process over $\overline{M}$ quadrature points for computing (13) is described in Algorithm 4.

The convergence rate of asymptotic parareal depends on the choice of η. The optimal choice of η, in the asymptotic parareal setting, is shown in [29] to depend most heavily on the coarse grid step size ΔT and scale separation parameter ϵ. The optimal η for a given ΔT and ϵ is the solution of an optimization problem where the sum of the averaging error and time-stepping error are minimized.

Algorithm 4 $\mathcal{N}(\bar{\mathbf{u}}_0)$, Evaluate time average (in parallel):

```
for $j = 1, \ldots, \overline{M} - 1$ do {*Parallel for-loop over j*}
    $s_m = \eta m/\overline{M}$
    $\bar{\mathbf{u}}_m \leftarrow \rho(s_m/\eta) e^{s_m\mathcal{L}}\mathcal{N}(e^{-s_m\mathcal{L}}\bar{\mathbf{u}}_0)$
end for
return $(1/\overline{M}) * \texttt{Sum}(\bar{\mathbf{u}}_1, \ldots, \bar{\mathbf{u}}_{\overline{M}})$
```

The fine grid time-stepping operator Φ for asymptotic parareal evolves $\mathbf{u}$ over the fine time-step Δt using a three-step Strang splitting method [3] which first advances the linear terms $\epsilon^{-1}\mathcal{L} + \mathcal{D}$ by a half time-step $\Delta t/2$ using an exponential integrator, then advances the nonlinear term $\mathcal{N}$ by a full time-step Δt using midpoint quadrature, and finally takes another $\Delta t/2$ half step for $\epsilon^{-1}\mathcal{L} + \mathcal{D}$ with the exponential integrator. The pseudocode for the fine propagator is given in Algorithm 5.

The coarse grid operator Φ_Δ used in asymptotic parareal also uses the Strang splitting scheme, integrating over the coarse interval ΔT instead. The exponential integrator only acts on the dissipative operator $\mathcal{D}$, and the averaged nonlinear term $\overline{\mathcal{N}}$ is evaluated using the midpoint rule. Moreover, the slowly-varying $\bar{\mathbf{u}}$ is transformed back to an approximation of $\mathbf{u}$ by applying the $e^{(-t/\epsilon)\mathcal{L}}$ operator, as shown in Equation (12). Pseudocode for the coarse solver is shown in Algorithm 6.

Algorithm 5 fine solver Φ: Fine_Solver$(\mathbf{u}_0, \Delta t, \Delta T)$

```
$M = \Delta T/\Delta t$
for $m = 1, \cdots, M$ do
    Take a $\Delta t/2$ time-step for the linear term:
    $\hat{\mathbf{v}} \leftarrow e^{(\Delta t/2)(\epsilon^{-1}\mathcal{L}+\mathcal{D})}\mathbf{u}_m$ {$\Delta t/2$ *step on* $\mathbf{u}' = (\epsilon^{-1}\mathcal{L} + \mathcal{D})\mathbf{u}$}
    Take a $\Delta t$ time-step for the nonlinear term:
    $\mathbf{v} \leftarrow \mathcal{N}(\hat{\mathbf{v}})$,
    $\mathbf{v} \leftarrow \hat{\mathbf{v}} + \Delta t \mathcal{N}\left(\hat{\mathbf{v}} + \frac{\Delta t}{2}\mathbf{v}\right)$ {$\Delta t$ *step on* $\mathbf{u}' = \mathcal{N}(\mathbf{u})$ *with mid-point*}
    Take a $\Delta t/2$ time-step for the linear term:
    $\mathbf{u}_{m+1} \leftarrow e^{(\Delta t/2)(\epsilon^{-1}\mathcal{L}+\mathcal{D})}\mathbf{v}$ {$\Delta t/2$ *step on* $\mathbf{u}' = (\epsilon^{-1}\mathcal{L} + \mathcal{D})\mathbf{u}$}
end for
return $\mathbf{u}_M$
```

Algorithm 6 asymptotic slow solver Φ_Δ: Coarse_Solver$(\mathbf{u}_0, \Delta T)$

```
Take a $\Delta T/2$ time-step for the linear dissipative term:
$\hat{\mathbf{v}} \leftarrow e^{(\Delta T/2)\mathcal{D}}\mathbf{u}_0$ {$\Delta T/2$ *step on* $\mathbf{u}' = \mathcal{D}\mathbf{u}$}
Take a $\Delta T$ time-step for the averaged nonlinear term:
$\mathbf{v} \leftarrow \overline{\mathcal{N}}(\hat{\mathbf{v}})$,
$\mathbf{v} \leftarrow \hat{\mathbf{v}} + \Delta T \overline{\mathcal{N}}\left(\hat{\mathbf{v}} + \frac{\Delta T}{2}\mathbf{v}\right)$ {$\Delta T$ *step on* $\mathbf{u}' = \overline{\mathcal{N}}(\mathbf{u})$ *with midpoint formula*}
Take a $\Delta T/2$ time-step for the linear dissipative term:
$\mathbf{v} \leftarrow e^{(\Delta T/2)\mathcal{D}}\mathbf{v}$ {$\Delta T/2$ *step on* $\mathbf{u}' = \mathcal{D}\mathbf{u}$}
Transform back to the fast time coordinate: $\mathbf{u}_1 = e^{(\Delta T/\epsilon)\mathcal{L}}\mathbf{v}$
return $\mathbf{u}_1$
```

4.3 Asymptotic Multigrid in time algorithm for RWSE

The asymptotic MGRIT algorithm for the RWSE is formed by employing the asymptotic solver in Algorithm 6 at the coarser levels of the MGRIT algorithm discussed in §2. The finest level is always solved by Algorithm 5, while the coarsest level always employs Algorithm 6. The intermediate levels may adopt either of these and we numerically investigate the different choices in §5. In addition, MGRIT allows the use of F- and FCF-cycles; a flexibility missing in the parareal algorithm.

5 Numerical Results

We examine the performance of asymptotic MGRIT to solve RWSE for a variety of setups.

5.1 Problem parameters

Our choices of problem parameters follow general conventions for asymptotic parareal studies. The experiments are run on a machine with two Intel Xeon Silver 4110 2.10GHz processors, based on the Skylake x86 architecture. Each processor has 8 cores and 16 threads per processor, giving a total of 16 cores and 32 threads.

In all experiments, the RSWE are semi-discretized to a system of ODEs as in (3). The size of the time domain $T = [0, t_f]$ and the number of points in the time grids vary between experiments, but are always given. Note that the initial time of the time domain is always 0. In space, we use a spectral discretization of 64 Fourier modes on a spatial domain with size $[0, 2\pi]$ [17, 30]. The initial conditions are set up in real space such that velocity profiles v_1, v_2 are identically zero and the wave height profile h is a Gaussian bump with a maximum height of 1 at the center of the spatial domain. The hyperviscosity coefficient μ is set to 10^{-4} in all experiments, which provides stability for more oscillatory Fourier modes while still allowing the problem to be advection-dominated. We choose the Froude number $Fr = F^{1/2}\epsilon$ with $F = 1$ and hence $Fr = \epsilon$. We focus on two choices of the scale separation parameter ϵ: $\epsilon = 1.0$ (where no separation between timescales occurs and temporal oscillations are not rapid,) and $\epsilon = 10^{-2}$ (where the timescale of gravity waves is 100 times smaller than the timescale of Rossby waves and fast temporal oscillations occur.) These parameter choices are based on [17, 30].

The time averaging window η is computed using the formula (from [28])

$$\eta = \alpha \frac{dT}{\epsilon^{0.2}}, \tag{16}$$

where α is a coefficient that is equal to 1 unless otherwise noted and dT is the time-step size on the level where time averaging is being done. The number of quadrature points $\overline{M}$ over which the time average is done is computed by the formula (from [28])

$$\overline{M} = \max\{25, \frac{80}{\beta}\eta\}, \tag{17}$$

where β is a nonzero coefficient that is equal to 1 unless otherwise noted. Equation (16) is derived experimentally in [29].

5.2 Implementation

A Python implementation of an asymptotic parareal solver for the 1-dimensional RSWE, titled Cyclops-Lite, is publicly available from Wingate's research group [28]. In Cyclops, the RSWE are semi-discretized yielding a system of ODEs like in Equation (3), and a Fourier spectral method is used as a spatial discretization.

Cyclops is written in Python and our chosen MGRIT implementation, XBraid [1], is written in C/C++. We have developed an interface written in the Cython language between Cyclops and XBraid to extend the asymptotic parareal method to asymptotic MGRIT, titled CycloBraid. CycloBraid allows XBraid to call Cyclops' routines for handling Fourier spectral discretizations as well as functions for time-averaging and time-integration (refer to Algorithms 4, 5, 6). XBraid then uses these wrapped Cyclops functions to define Φ on each level and carry out the MGRIT cycling. Some other wrapper functions were required, e.g., taking a norm, performing vector addition, and MPI buffer packing and unpacking.

5.3 Asymptotic MGRIT: Two-Level With FCF-Relaxation

Here, we investigate whether FCF-relaxation can outperform F-relaxation in a two-grid asymptotic MGRIT solve. FCF-relaxation has previously shown a benefit (sometimes significant) over F-relaxation, e.g., in [10, 18, 19], with the theoretical justification given in [5]. Essentially, FCF-relaxation effectively damps all error modes that correspond to an eigenvalue of the error propagator that is significantly less than 1.

We consider $\epsilon = .01$ and time domains of the size $[0, t_f] \in \{[0.0, 10.0], [0.0, 15.0], [0.0, 20.0], [0.0, 25.0]\}$. We consider the two-level case with $N_\Delta = 16$ time points on the coarsest grid, and coarsening factors of $m = \{16, 128, 256\}$, which yields $N = \{256, 2048, 4096\}$ time points on the fine grid. We choose these coarse grid sizes and coarsening factors to follow the choices of coarse grid sizes used in previous studies of asymptotic parareal [17, 30, 29].

The residual halting tolerance is chosen to be $(1.0e\text{-}8)/\sqrt{\Delta t}$, and we note that standard MGRIT (i.e., without the use of the asymptotic solver on the coarse grid) fails to achieve convergence for this setup, underscoring the importance of time averaging at coarse levels of MGRIT for highly oscillatory PDEs. The iteration counts for the various runs are shown in Table 1. Corresponding wall times are given in Table 2. The wall times are much faster for FCF-relaxation compared to simply using F-relaxation.

t_f	F, $m = 16$	FCF, $m = 16$	F, $m = 128$	FCF, $m = 128$	F, $m = 256$	FCF, $m = 256$
10	14	10	10	7	10	7
15	15	9	16	8	15	8
20	18	10	17	8	17	8
25	**	**	14	9	13	9

Table 1: Iteration counts for 2-level asymptotic MGRIT runs, $\epsilon = .01$, using the coarsening factors m and relaxation schemes denoted in the top row of each column. Runs that fail to converge are marked with "**".

t_f	F, $m = 16$	FCF, $m = 16$	F, $m = 128$	FCF, $m = 128$	F, $m = 256$	FCF, $m = 256$
10	22	16	18	13	19	15
15	36	22	42	21	43	23
20	61	34	60	29	62	31
25	**	**	63	42	64	44

Table 2: Wall times (using 16 cores) for asymptotic MGRIT runs, $\epsilon = .01$, using the coarsening factors m and relaxation schemes denoted in the top row of each column. These results are "strange" when compared with the iteration counts in Table 1, in that we do not generally expect improvements in wall time from FCF-relaxation, unless we are cutting down on iteration counts far more than 50% when compared to F-relaxation iteration counts. Runs that fail to converge are marked with "**".

Note that the iteration counts for FCF-relaxtion are roughly half as those of F-relaxation. Typically, one FCF-relaxation sweep is twice as expensive as F-relaxation sweep. Hence, we would expect that the wall-times for both schemes should be roughly equivalent. We next discuss the reason for the surprising FCF-relaxation timings.

When the coarse grid operator Φ_Δ is computationally cheap, the dominant cost of one Braid iteration is the application of F-relaxation. In such a case, it is reasonable to assume that one Braid iteration using FCF-relaxation will be roughly twice as expensive as one Braid iteration using F-relaxation, as FCF-relaxation performs two sweeps of F-relaxation. However, the wall times in Table 2 indicate that asymptotic MGRIT wall times are primarily a function of the iteration counts seen in Table 1, regardless of the relaxation strategy. In particular, we see that in the problems where FCF-relaxation halves the iteration count, the wall times are roughly halved. Here, we show the results of a performance study that explain the observations in Table 2.

We measure the wall time taken for a single execution of the time-integrator, given by Algorithms 5 and 6, either at the fine level or the coarse level. Based on previous notation, this is the time to execute Φ at the fine level and to execute Φ_Δ at the coarse level. Since F- and FCF-relaxation experiments for the same problem execute this operator a different number of times, we examine the average of the wall clock times for a single execution of the time-integrator.

To do this, we let $\omega_{0,f}, \omega_{1,f}$ be the sum of the wall clock times for a single execution of the solver at levels 0 and 1 (fine and coarse), $s_{0,f}, s_{1,f}$ be the total number of solver executions on levels 0 and 1, and $a_{0,f}, a_{1,f}$ be the average wall clock time for a solver execution on levels 0 and 1, respectively, when using F-relaxation. The notation FCF-relaxation is similar, except that f in the subscript is replaced by fcf. We observe,

$$\omega_{0,f} = .022, \quad s_{0,f} = 44, \quad a_{0,f} = .0005$$
$$\omega_{1,f} = .363, \quad s_{1,f} = 28, \quad a_{1,f} = .013$$
$$\omega_{0,fcf} = .032, \quad s_{0,fcf} = 68, \quad a_{0,fcf} = .0005$$
$$\omega_{1,fcf} = .355, \quad s_{1,fcf} = 28, \quad a_{1,fcf} = .013$$

Thus, the execution of $\Phi_\triangle$ on the coarse level is roughly two orders of magnitude more expensive than the execution of Φ on the fine level.

Additionally, the dominant computational cost when using Algorithm 6 is the evaluation of the integral for $\overline{\mathcal{N}}$ (see Equation (14)). Letting $a_{\overline{\mathcal{N}}}$ be the average wall time to for computing $\overline{\mathcal{N}}$, our timings reported that $a_{\overline{\mathcal{N}}} = .006$. Since $\Phi_\triangle$ requires two computations of $\overline{\mathcal{N}}$, we have that $2a_{\overline{\mathcal{N}}} = .012$, and $2\dfrac{a_{\overline{\mathcal{N}}}}{a_{1,f/fcf}} = \frac{.012}{.013} \approx .92$. That is, roughly 92% of the run-time for $\Phi_\triangle$ on the coarse level is attributable to this computation, making this time-averaging computation the dominant cost overall for an asymptotic MGRIT iteration.

One can see why this computation is so expensive. The dominant cost of evaluating $\overline{\mathcal{N}}$ involves a nonlinear multiplication routine. In order to compute the nonlinear terms for the time-averaged quantity in Line 3 of Algorithm 4, it is necessary to perform the multiplications to find the terms of $\mathbf{N}(\mathbf{u})$ in real space. So, one inverse Fourier transform and one forward Fourier transform is necessary per quadrature point when computing $\overline{\mathcal{N}}$. With M total quadrature points, $2M$ Fourier transforms are necessary for computation of $\overline{\mathcal{N}}$.

This cost can be mitigated, however. The `for` loop in Algorithm 4 can be parallelized, so if at least M idle processors are available, then a speedup of roughly M would be available when executing Algorithm 6 (i.e., $\Phi_\triangle$). In this case, the cost of $\Phi_\triangle$, F-, and FCF-relaxation would be more balanced, and further performance analysis would be required. However in our case, compute time is primarily a function of the iteration count, and the use of FCF-relaxation is advised given the large savings in wall time. We note that this preference for FCF is novel and should be reflected often in practice, as idle processors are not always available.

5.4 A three level Asymptotic MGRIT

In this section, we research the construction of a third MGRIT level and examine performance effects. Given a two-level asymptotic MGRIT solve, we introduce a coarser grid as a third level, which has not been studied before for asymptotic parareal. In doing so, we introduce parallelism to the solver on the second level (i.e., first coarse level) as the second level will no longer be solved exactly but instead with parallel FCF-relaxation. Thus, if the third grid does not incur a significant penalty in convergence and the solution on the third grid itself can be computed cheaply, then the introduction of a third grid could lead to performance gains in parallel.

Once the oscillations are removed on level two through the asymptotic time-averaging, it is possible that using the non-asymptotic time-stepper Φ described in Algorithm 5 would make for a good time propagator on the third grid. To do this, the transform back to the fast time coordinate operation $e^{-(t/\epsilon)L}$ (see Equation (12)) must not be used to add the fast temporal oscillations back into the space-time solution guess on the second level before restriction is performed to transfer information to the third level. In this case where there are now no rapid oscillations to alias on the third grid, the relatively cheap Φ operator could be a good option on the third level, if using it does not incur a penalty in convergence.

Following this, we consider two techniques for time-stepping on the coarsest grid. One we will refer to as "Strang-asymptotic-asymptotic MGRIT", or SAA-MGRIT, which uses the time-averaged coarse grid operator $\Phi_\triangle$ on the second and third levels. This approach applies the $e^{-(t/\epsilon)L}$ transform to reintroduce the rapid temporal oscillations before performing restriction and interpolation between the second and third levels, as well as when interpolating from the second level to the fine level. The other we refer to as "Strang-asymptotic-Strang MGRIT", or SAS-MGRIT, which uses the time-averaged operator $\Phi_\triangle$ on the second level and the standard Strang splitting operator Φ described in Algorithm 5 on the coarsest level. In SAS-MGRIT, the transformation $e^{-(t/\epsilon)L}$ is only applied when interpolating from the second level to the fine level.

For convenience, we introduce the following notation to denote a general multilevel MGRIT time grid hierarchy. Let $N_{t,0} \geq N_{t,1} \geq \cdots \geq N_{t,k-1}$ be the numbers of time points on a k-level MGRIT time grid hierarchy, where level 0 corresponds to the finest grid and level $k-1$ corresponds to the coarsest grid. Then, the notation $\{N_{t,0}, N_{t,1}, \cdots, N_{t,k-1}\}$ denotes the number of time points on each level in an MGRIT hierarchy. We will use subscript "$_A$" to denote the use of the asymptotic coarse grid operator Φ_Δ and subscript "$_S$" to denote the use of the Strang splitting operator Φ on the given level. For instance, $\{4096_S, 64_A\}$ denotes a two-level asymptotic MGRIT solve with 4096 time points and Strang splitting Φ on the fine grid and 64 time points with the asymptotic time integrator Φ_Δ on the coarse grid. We also say $\{4096_S, 64_S\}$ to denote a non-asymptotic MGRIT solve with the same two-level time grid hierarchy, and $\{4096_S, 64_A, 8_S\}$ denotes a three-level SAS-MGRIT solve with 4096 time points and Strang splitting Φ on the fine level, 64 time points and the asymptotic time integrator Φ_Δ on the second level, and 8 time points with Strang splitting Φ on the third level. Likewise, $\{4096_S, 64_A, 8_A\}$ denotes a three-level SAA-MGRIT solve with 4096 time points and Strang splitting Φ on the fine level, 64 time points and the asymptotic time integrator Φ_Δ on the second level, and 8 time points with the asymptotic time integrator Φ_Δ on the third level.

In Tables 3 and 4, we compare the iteration counts and wall times of a two-grid asymptotic MGRIT solve $\{4096_S, 64_A\}$ with three-level SAA- and SAS-MGRIT solves $\{4096_S, 64_A, x_A\}$ and $\{4096_S, 64_A, x_S\}$ where $x = \{2, 4, 8, 16, 32\}$. In other words, given a two-grid asymptotic MGRIT solver, we consider the performance effects of adding a third coarser grid under the SAA- and SAS-MGRIT techniques. The two-level solver is given 64 time points on the coarse level because this is comparable to the coarse grid sizes used in studies of asymptotic parareal for the time domain sizes considered [17, 30, 29] and because this coarse grid size is required for MGRIT convergence in the case of $\epsilon = 1.0$. The time domains considered are $T = [0, t_f]$ where $t_f = \{1, 5, 10\}$. F-relaxation is used, and both $\epsilon = .01$ and $\epsilon = 1.0$ are considered. The halting tolerance is set to $tol = 1.0e\text{-}5$. Averaging windows η and quadrature points $\overline{M}$ are computed at each level using Equations (16) and (17) and the ΔT of that level.

For $t_f = 1.0$, SAA- and SAS-MGRIT can perform as well or better than two-level asymptotic MGRIT for both $\epsilon = .01$ and $\epsilon = 1.0$. In particular, we see in Table 3 for $t_f = 1.0$, the described multilevel MGRIT techniques are able to offer a 36% improvement in compute time over the two-grid method for $\{4096_S, 64_A, 32_S\}$ and a 43% improvement in compute time for $\{4096_S, 64_A, 16_A\}$ when $\epsilon = 1$. We see similar improvements in compute time when $\epsilon = .01$ in Table 4, where $\{4096_S, 64_A, 32_S\}$ reduces compute time by 47% and $\{4096_S, 64_A, 16_A\}$ reduces compute time by 20% when $t_f = 1.0$. Furthermore, when $t_f = 5.0$ and $\epsilon = .01$, we also observe speedup; $\{4096_S, 64_A, 16_A\}$ yields a 16% reduction in wall time and $\{4096_S, 64_A, 16_A\}$ gives a 21% reduction in wall time.

For larger time domains where $\epsilon = 1.0$, SAA- and SAS-MGRIT both fail to converge. This is not attributable to the SAA- or SAS-MGRIT methods, but it is instead attributable to the "coarse-grid time domain limit" phenomenon for $\epsilon = 1.0$ which also causes two-grid MGRIT time grid hierarchies (i.e., $\{4096_S, x_A\}$ where $x = \{2, 4, 8, 16\}$ for $t_f = 5.0$ and $x = \{2, 4, 8, 16, 32\}$ for $t_f = 10.0$) to diverge. This phenomenon is addressed in detail in Section 5.5.

When $\epsilon = .01$, SAS-MGRIT is stable for smaller time domains but becomes unstable for larger time domains, whereas SAA-MGRIT is convergent for every examined time grid hierarchy.

	$\{4096, 64, 2\}$	$\{4096, 64, 4\}$	$\{4096, 64, 16\}$	$\{4096, 64, 32\}$	(ref) $\{4096_S, 64_A\}$
$t_f = 1.0$	SAA: 4 (5.2s) SAS: 4 (5.0s)	SAA: 6 (5.4s) SAS: 6 (5.1s)	SAA: 3 (2.3s) SAS: 5 (2.7s)	SAA: 3 (3.0s) SAS: 5 (2.6s)	3 (4.1s)
$t_f = 5.0$	**	**	SAA: 14 (10.3s) SAS: 15 (8.2s)	SAA: 12 (12.3s) SAS: 16 (8.7s)	5 (7.1s)
$t_f = 10.0$	**	**	**	**	12 (16.9s)

Table 3: $\epsilon = 1$ comparison of three-level MGRIT solvers using either a "Strang-asymptotic-asymptotic" (SAA) or "Strang-asymptotic-Strang" (SAS) method of adding a third grid to a two-level asymptotic MGRIT solve with the structure $\{4096_S, 64_A\}$. F-relaxation is used. Runs that fail to converge are marked with "**". Columns are ordered by the number of coarse points N_3 on the third grid, except for the rightmost column which provides the reference iteration count and wall time for the two-level $\{4096_S, 64_A\}$ solve.

	$\{4096, 64, 2\}$	$\{4096, 64, 4\}$	$\{4096, 64, 16\}$	$\{4096, 64, 32\}$	(ref) $\{4096_S, 64_A\}$
$t_f = 1.0$	SAA: 4 (5.5s) SAS: 5 (6.6s)	SAA: 5 (4.7s) SAS: 6 (5.2s)	SAA: 6 (4.6s) SAS: 7 (3.7s)	SAA: 5 (5.0s) SAS: 6 (3.0s)	4 (5.7s)
$t_f = 5.0$	SAA: 11 (24.5s) SAS: 16 (22.6s)	SAA: 11 (17.5s) SAS: **	SAA: 11 (13.5s) SAS: **	SAA: 16 (18.5s) SAS: 23 (12.6s)	11 (16.1s)
$t_f = 10.0$	SAA: 21 (90.5s) SAS: **	SAA: 21 (60.8s) SAS: **	SAA: 11 (22.1s) SAS: **	SAA: 10 (19.1s) SAS: **	21 (38.2s)

Table 4: $\epsilon = .01$ comparison of three-level MGRIT solvers using either a "Strang-asymptotic-asymptotic" (SAA) or "Strang-asymptotic-Strang" (SAS) method of adding a third grid to a two-level asymptotic MGRIT solve with the structure $\{4096_S, 64_A\}$. F-relaxation is used. Runs that fail to converge are marked with "**". Columns are ordered by the number of coarse points N_3 on the third grid, except for the rightmost column which provides the reference iteration count and wall time for the two-level $\{4096_S, 64_A\}$ solve.

5.5 The $\epsilon = 1$ Case: On the Use of CF-relaxation Sweeps and Larger Averaging Values η to Allow For More Efficient Coarse Grids

For the rotating shallow water equations, the case where $\epsilon = \mathcal{O}(1)$ corresponds to regimes in geophysical flows where gravity waves are not prominent and the behavior of the flow is influenced instead by slow Rossby waves. Thus, we want to improve the solver for this physically relevant case. In [17, 29, 30], attention is paid to the case where $\epsilon = 1$. In particular, it is shown in numerical experiments that the performance of asymptotic parareal is roughly equivalent, but not superior, to standard parareal for the rotating shallow water equations in terms of convergence. Thus, this is a difficult problem for asymptotic parareal. We demonstrate in this section that two-level asymptotic MGRIT can achieve convergence where standard and asymptotic parareal cannot when using an averaging window $\eta > \Delta T$ and/or an increased number of CF-sweeps in the relaxation scheme.

We choose the following parameters similar to [17, 30, 29], in particular the coarse and fine grid sizes are the same as found in much of [29]. The smallest time domain sizes of $t_f = \{1, 4\}$ are among the time domain sizes used in the previous literature to show convergence, and we introduce larger time domains thereafter to demonstrate where divergence occurs. Using $\epsilon = 1$ and a fixed coarse grid of $N_\Delta = 50$ time points and fine grid of $N = 50000$ time points, we vary the time domain $T = [0, t_f]$ where $t_f = \{1, 4, 8, 12, 16, 20\}$. We also vary the size of the averaging window, using Equation (16) and choosing the coefficient $\alpha = \{1, 4, 12, 16, 20\}$. Thus as α increases, the size of the averaging window also increases commensurately. Equation (17) is used with $\beta = 1$ to determine the number of quadrature points $\overline{M}$ used in the computation of the slowly-varying term $\overline{\mathcal{N}}$ in Algorithm 4. The result is that the number of quadrature points scales proportionally with the size of the averaging window. In Tables 5 and 6, iteration counts to reach a halting tolerance of 1.0e-6 and associated wall times are shown for time domains of $T = [0, t_f]$, for $t_f = \{1, 4, 8, 12, 16, 20\}$ and various F(CF)^n-relaxation strategies of $n = 0, 1, 2, 3, 4$. Since the relative cost of FC-relaxation is small, it is reasonable to consider these larger n.

The iteration counts shown in Table 5 are for a fine grid size of $N = 50000$; similar results were observed for $N = \{10000, 25000, 50000, 75000, 100000\}$. We also see in Table 5 that in the $\epsilon = 1$ case, an increased number of CF-relaxation sweeps and an increased averaging window length are both individually capable of allowing two-level asymptotic MGRIT to converge over significantly longer time domains when using a 50 point coarse time grid, yielding convergence for $t_f = 12.0$. When increasing both the number of CF-sweeps and averaging window size, it is possible to achieve convergence for $t_f = 16.0$ with a 50 point coarse time grid. The number of CF-relaxation sweeps to achieve convergence for a time domain of $t_f = 16.0$ reaches a minimum of 2 at $\alpha = 12.0$, but for other values of α, more CF-sweeps are necessary to achieve convergence. This suggests that there is an optimal averaging window length $\eta > \Delta T$ for large time domains that would ordinarily diverge in asymptotic parareal where additional CF-relaxations are not an option.

Our experiments suggest the existence of a "coarse grid time domain size limit" in asymptotic MGRIT and parareal that prevents taking large coarse time-steps when $\epsilon = 1$, even when a problem with the same coarse time-step size converges for $\epsilon = 10^{-2}$. In Section 5.4 and Table 3, we saw that when $\epsilon = 1$, the introduction of a coarse grid with 4 or 2 points when $t_f = 5.0$ results in divergence with SAA-MGRIT and

SAS-MGRIT. The same divergence occurs when a coarse grid with 32 points or fewer is used with $t_f = 10.0$. In contrast, we see in Table 4 the introduction of the same-sized coarse grids when $\epsilon = .01$ converges with SAA-MGRIT. We also see this in Table 5 where increasing the time domain over a fixed coarse grid size $N_\Delta = 50$ eventually results in divergence when $\epsilon = 1$, although we see in Table 1 how a coarse grid of $N_\Delta = 16$ points and $\epsilon = 0.1$ works well with asymptotic MGRIT and much larger time domain sizes. In summary, each of these cases has a limit to how small N_Δ can be before MGRIT diverges.

The wall timings given in Table 6 suggest the relationship between CF-sweeps and overall wall time depend primarily on the computational cost of the coarse grid operator Φ_Δ relative to the cost of relaxation. When $\alpha = 1$ and $\alpha = 4$, increasing the number of CF-sweeps increases the overall wall time, even when the iteration count is decreased by the extra relaxation. Since the number of points $\overline{M}$ in the midpoint quadrature computation is small, the coarse grid operator Φ_Δ is cheap and the dominant cost of a Braid iteration in this case is the application of relaxation. However, for $\alpha = 12$ and above, $\overline{M}$ is sufficiently increased to the point where the dominant cost of a Braid iteration is the application of the coarse grid operator Φ_Δ. Here, reductions in iteration count as a result of the increased CF-sweeps on larger time domains translate to a reduction in wall time.

For instance, when $\alpha = 1$, although increasing the number of CF-sweeps from $n = 2$ to $n = 4$ reduces the iteration count required for convergence when $t_f = 12.0$ from 12 to 9, it also increases the wall time from 74.3 seconds to 90.0 seconds. The average wall time per iteration increases from 6.2 seconds when $n = 2$ to 10.0 seconds when $n = 4$; hence when $n = 4$, a Braid iteration is on average 38% more expensive than when $n = 2$. However, when $\alpha = 12$ and $t_f = 16$, 16 iterations are required to reach convergence with $n = 2$ and 11 iterations are required with $n = 4$. The corresponding wall times are 295.5 seconds (18.5 seconds per iteration) when $n = 2$ and 229.8 seconds (20.9 seconds per iteration) when $n = 4$, so a Braid iteration is merely 11% more expensive when $n = 4$.

We speculate that this MGRIT convergence benefit from larger α and n is due to the eigenstructure of Φ. In [5], FCF-relaxation is shown to be effective in smoothing out error modes associated with eigenvalues of Φ that are $<< 1$. Thus, we speculate that the troublesome error modes for MGRIT and longer time domains have an eigenvalue $<< 1$. Enlarging α to 12 and 16 may also be having a positive effect by making the coarse grid Φ_Δ more accurate.

5.6 Accelerating $\bar{\mathbf{u}}$ Computations by Reusing Stale Nonlinear Quantities $\overline{\mathcal{N}}()$

In instances where Φ or Φ_Δ is computationally expensive, one previously used technique is to reuse information computed from previous MGRIT iterations or from previous MGRIT levels [11, 19]. As shown in Section 5.3, the dominant cost of an asymptotic MGRIT iteration is the application of the coarse grid operator Φ_Δ, particularly the forward and inverse Fourier transforms used to compute nonlinear quantities in Algorithm 4. We now examine strategies for reusing stale values of the computationally expensive nonlinear quantities $\overline{\mathcal{N}}$, which are used to compute $\bar{\mathbf{u}}$ in Algorithm 6. We fix a coarse time grid of $N_\Delta = 50$ time points, a fine grid of $N = 50,000$ time points, time domain $T = [0, 5]$, and consider $\epsilon = 0.01$ as well as $\epsilon = 1$. F-relaxation is used.

As discussed in Section 5.3, the majority of the cost of computing $\bar{\mathbf{u}}$ comes from the two $\overline{\mathcal{N}}()$ evaluations in Step 2 of Algorithm 6; in contrast, the exponential integration calls in Step 1 and Step 3 of Algorithm 6 are relatively cheap. Here, we consider reuse schemes which use stale values of the expensive $\overline{\mathcal{N}}(\hat{\mathbf{v}})$ and $\overline{\mathcal{N}}(\hat{\mathbf{v}} + \frac{\Delta T}{2}\mathbf{v})$ computations in Step 2 of Algorithm 6 at certain MGRIT iterations, but otherwise perform the time integration on the coarse grid as laid out in Algorithm 6. We consider the following four reuse patterns and compare them to a baseline solve without reuse.

1. Reuse at iterations $j = \{7, 8, 9, 10\}$, recompute $\bar{\mathbf{u}}$ by reusing nonlinear terms computed at iteration $j = 6$. All other iterations are computed without reuse.

2. Reuse at iterations $j = \{5, 6, 7, 8, 9, 10\}$ recompute $\bar{\mathbf{u}}$ by reusing nonlinear terms computed at iteration $j = 4$. All other iterations are computed without reuse.

3. Reuse at all odd iterations $j = \{1, 3, 5, 7, 9\}$, recompute $\bar{\mathbf{u}}$ by reusing nonlinear terms computed at the respective prior iterations $j = \{0, 2, 4, 6, 8\}$. All other iterations are computed without reuse.

$\alpha = 1$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	3	6	14	**	**	**
1	3	5	9	**	**	**
2	3	5	7	12	**	**
3	3	5	7	10	**	**
4	3	5	6	9	**	**

$\alpha = 4$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	6	14	18	20	**	**
1	6	13	15	15	**	**
2	6	10	12	13	**	**
3	6	9	11	11	**	**
4	6	8	10	10	10	**

$\alpha = 12$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	10	13	17	28	**	**
1	10	11	13	19	**	**
2	9	10	12	15	16	**
3	9	10	11	13	13	**
4	8	9	10	11	11	**

$\alpha = 16$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	11	13	19	31	**	**
1	10	11	16	21	**	**
2	10	10	13	16	**	**
3	9	9	11	13	13	**
4	9	9	10	11	11	**

$\alpha = 20$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	11	13	21	**	**	**
1	10	11	17	21	**	**
2	10	10	14	15	**	**
3	9	10	11	13	**	**
4	9	9	10	11	11	**

Table 5: Iteration counts for two-level asymptotic MGRIT solves of the RSWE. Here, the averaging window η and quadrature points $\overline{M}$ are chosen using Equations (16) and (17), where α is given in the upper-left cell of each table and $\beta = 1$. The value $\epsilon = 1$ is used corresponding to no scale separation and the time grid size is fixed at $N_\Delta = 50$, $N = 50000$. Columns are ordered as the final time t_f varies from 1.0 to 20.0 as given by the top row. Rows are ordered by number of CF-relaxations applied in smoothing i.e., $F(CF)^n$-relaxation where the left-most column gives n for each row. Runs that fail to converge are marked with "**".

4. Reuse at odd iterations $j = \{5, 7, 9\}$ recompute $\bar{\mathbf{u}}$ by reusing nonlinear terms computed at respective prior iterations $j = \{4, 6, 8\}$. All other iterations are computed without reuse.

In Figure 5 for $\epsilon = .01$, all of the considered $\overline{\mathcal{N}}()$ reuse patterns do not incur a convergence penalty of more than one order of magnitude when compared to the baseline solve. More strikingly, we see that for the schemes that perform $\overline{\mathcal{N}}()$ reuse at all odd iterations, the residual norm at iteration 10 is marginally smaller than the reference solution.

However for $\epsilon = 1$ in Figure 6, the convergence penalty for reusing prior $\overline{\mathcal{N}}()$ terms is generally steeper. In this case, the slowing of convergence results in all cases of reuse leading to an order of magnitude or greater penalty in residual norm convergence at iteration 10, when compared with the baseline solve. However, the timings discussed below still indicate significant possible savings.

In Table 7, we see that the average wall time of a call to the time-stepping routine is roughly halved when $\overline{\mathcal{N}}()$ terms are reused in accordance with the even-odd alternating $\overline{\mathcal{N}}()$ reuse pattern (numbered 3 above).

$\alpha = 1$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	8.4	16.2	37.7	**	**	**
1	11.8	22.6	40.6	**	**	**
2	15.3	31.2	43.2	74.3	**	**
3	20.1	36.2	54.2	83.0	**	**
4	24.2	46.3	56.8	90.0	**	**

$\alpha = 4$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	16.0	38.5	69.6	91.9	**	**
1	27.9	63.5	85.9	100.9	**	**
2	36.2	64.5	85.4	102.7	**	**
3	45.9	72.8	99.0	105.0	**	**
4	57.0	76.7	104.4	111.2	126.2	**

$\alpha = 12$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	27.5	60.0	134.7	308.6	**	**
1	48.4	68.4	120.5	242.5	**	**
2	57.4	88.1	139.0	214.1	295.5	**
3	75.9	101.1	137.7	207.5	252.4	**
4	77.5	106.5	142.2	203.3	229.8	**

$\alpha = 16$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	30.1	75.4	191.0	449.6	**	**
1	44.8	84.9	193.8	341.1	**	**
2	44.9	87.4	170.1	286.2	**	**
3	71.7	93.0	160.6	255.5	320.4	**
4	91.8	108.0	164.5	231.6	288.1	**

$\alpha = 20$	$t_f = 1.0$	4.0	8.0	12.0	16.0	20.0
$n = 0$	34.3	89.5	258.6	**	**	**
1	46.7	89.7	232.7	421.5	**	**
2	64.4	104.4	217.3	323.5	**	**
3	72.6	118.0	186.2	303.2	**	**
4	92.0	117.9	186.2	274.4	351.9	**

Table 6: MPI 16-core wall clock times for two-level asymptotic MGRIT solves of the RSWE. Here, the averaging window η and quadrature points are chosen using Equations (16) and (17), where α is given in the upper-left cell of each table and $\beta = 1$. The value $\epsilon = 1$ is used corresponding to no scale separation and the time grid size is fixed at $N_\Delta = 50$, $N = 50000$. Columns are ordered as the final time t_f varies from 1.0 to 20.0 as given by the top row. Rows are ordered by number of CF-relaxations applied in smoothing i.e., $F(CF)^n$-relaxation where the left-most column gives n for each row. Runs that fail to converge are marked with "**".

This holds true for both $\epsilon = .01$ and $\epsilon = 1$. Given that this strategy results in data reuse for 5 out of 11 MGRIT iterations, it is expected to yield significant cost savings.

	$\epsilon = .01$	$\epsilon = 1$
Reuse	$a_{f,0} = .00045$ $a_{f,1} = .00682$	$a_{f,0} = .00045$ $a_{f,1} = .00676$
No Reuse	$a_{f,0} = .00047$ $a_{f,1} = .01335$	$a_{f,0} = .00047$ $a_{f,1} = .01273$

Table 7: Average wall times on levels 0 and 1 for $\overline{\mathcal{N}}()$ reuse. The reuse pattern used is (3). F-relaxation is used as indicated by the f subscripts, $\mathcal{N}_\Delta = 50$, and $t_f = 5.0$.

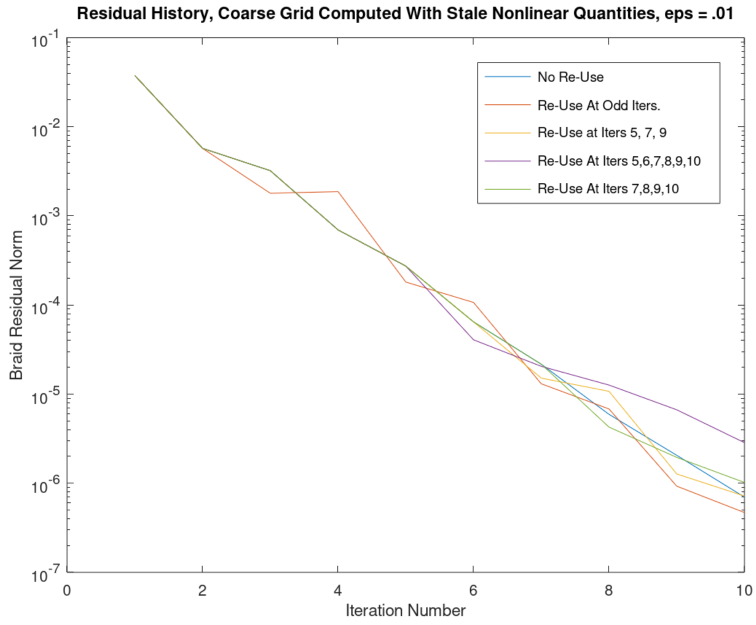

Figure 5: Comparison of various data reuse patterns for $\epsilon = 0.01$, showing the residual norm histories for the various two-level asymptotic MGRIT solves. Data reuse is based on using stale nonlinear quantities $\overline{\mathcal{N}}()$ for computing $\bar{\mathbf{u}}$. A coarse grid of $N_\Delta = 50$, a fine grid of $N = 50,000$, a final time $t_f = 5.0$ and F-relaxation are used.

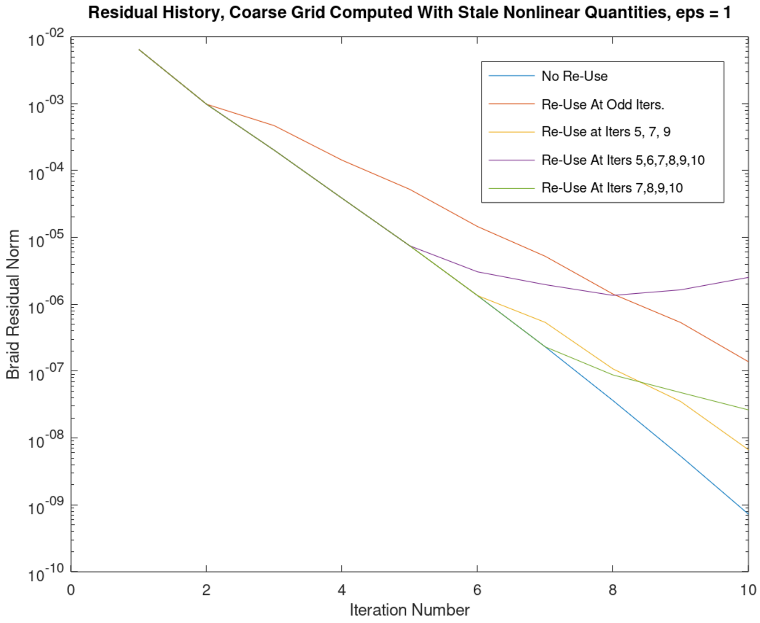

Figure 6: Comparison of various data reuse patterns for $\epsilon = 1.0$, showing the residual norm histories for the various two-level asymptotic MGRIT solves. Data reuse is based on using stale nonlinear quantities $\overline{\mathcal{N}}()$ for computing $\bar{\mathbf{u}}$. A coarse grid of $N_\Delta = 50$, a fine grid of $N = 50,000$, a final time $t_f = 5.0$ and F-relaxation are used.

6 Conclusions

In this article, we developed and investigated variations of an asymptotic MGRIT approach for solving the RSWE parallel-in-time. These approaches are based on the the asymptotic parareal method, which uses the key component of an asymptotically time-averaged coarse grid time-stepping scheme.

In Section 5.4, we proposed the so-called SAS-MGRIT and SAA-MGRIT three-level techniques with the goal of introducing a nonintrusive and cheap third grid to the two-grid asymptotic MGRIT/parareal method. This strategy allows the expensive applications of the asymptotic coarse grid operator Φ_Δ on the second level (i.e., first coarse grid) to be parallelized. Thus, instead of solving a larger second level sequentially, asymptotic MGRIT instead solves a smaller third level (i.e., coarsest grid) sequentially. While SAS-MGRIT and SAA-MGRIT were successful in improving wall times for small time domains, we saw that SAS-MGRIT became divergent and SAA-MGRIT became inefficient (compared to two-level asymptotic MGRIT) for longer time domains. In response to this, we suggest that future research target an alternative time integration technique for the third level, which increases robustness for long time domains.

In Section 5.3, we research the use of $F(CF)^n$ relaxation inside of asymptotic MGRIT. We find that the use of additional CF-sweeps ($n > 0$), which is novel for asymptotic parareal-type algorithms, both increases robustness with respect to problem parameters and decreases runtime in some settings. For instance, we demonstrate that when $\epsilon = .01$, FCF-relaxation ($n = 1$) can halve the number of Braid iterations required to reach convergence. In addition, FCF-relaxation is able to reduce the wall time when compared to F-relaxation, with the runtime roughly proportional to the total iteration count. We showed that this is the result of the coarse grid operator Φ_Δ dominating the cost of an individual Braid iteration. Thus, FCF-relaxation is able to reduce wall time by reducing the total number of time integrations with Φ_Δ on the coarse grid.

In Section 5.5, further benefits of additional CF-sweeps ($n > 1$) in conjunction with larger time averaging window sizes η are demonstrated by showing greater robustness of the solver when $\epsilon = 1.0$. When η and the number of CF-sweeps are increased, we see that asymptotic MGRIT is able to quickly converge over relatively long time domains, compared to F-relaxation, i.e., asymptotic parareal. The benefit from increasing η for our test cases appears to peak when η is computed using the formulas (16) and (17) with $\alpha = 12.0, \beta = 1.0$ and $N_\Delta = 50$. The optimal n for a problem depends on the time domain size and the cost of an individual Braid iteration. For example, when $t_f = \{1.0, 4.0\}$ and $N_\Delta = 50$, it is clear that $n = 0$ is the best choice. For larger time domains, however, $n = 2$ can drastically reduce iteration counts over $n = 0$. In these cases $n = 3$ or larger does not offer as dramatic a reduction in iteration count.

We believe that the results in Section 5.4, where three-level asymptotic MGRIT is explored (SAA- and SAS-MGRIT), indicate some promise for three-level (or more) solves with asymptotic MGRIT. In particular for small time-domains, the three-level solves showed an improvement over two-level, converging faster and in less wall clock time. However as discussed in Section in 5.4, more research is required to further develop the coarse level time-stepping operator for longer time domains. There is some previous work that has been done with regards to taking large time steps in the shallow water setting which may be appropriate for consideration. In [37], Wingate shows that a semi-implicit method due to Dukowicz and Smith [6] is capable of recovering the Rossby wave amplitudes for large time steps when solving the linearized quasigeostrophic equations. LeVeque describes a generalization of Godunov's method allowing arbitrarily large time steps in [22]. Further large time step schemes have been studied in a shallow water setting [12, 26, 38].

Lastly, data reuse for the expensive time-average nonlinear quantities on coarse levels offers an additional path for improved efficiency.

Another fruitful direction would be to parallelize the computation of $\overline{\mathcal{N}}$ by parallelizing over the quadrature points $\overline{M}$ (see for-loop in Algorithm 4). This would be particularly useful in cases where asymptotic MGRIT benefits from longer averaging windows η, which require more points $\overline{M}$ to resolve. For instance in [29], it is shown that when $\epsilon = .01$, convergence improves as the size of the averaging window η increases. Additionally in Section 5.5, we demonstrate improved convergence and more efficient coarse grids (i.e., fewer coarse time points N_Δ) when $\epsilon = 1.0$ and larger averaging windows $\eta > \Delta T$ are chosen. Thus if given enough parallel resources, an increase in the averaging window size would not incur a sizeable penalty in compute time, because the for-loop in Algorithm 4 would be parallelized..

References

[1] XBraid: Parallel multigrid in time. https://github.com/XBraid/xbraid.

[2] A. Brandt. Multi–level adaptive solutions to boundary–value problems. *Math. Comp.*, 31(138):333–390, 1977.

[3] Alina Chertock, Alexander Kurganov, and Guergana Petrova. Fast explicit operator splitting method for convection–diffusion equations. *International Journal for Numerical Methods in Fluids*, 59(3):309–332, 2009.

[4] Hans De Sterck, Stephanie Friedhoff, Alexander JM Howse, and Scott P MacLachlan. Convergence analysis for parallel-in-time solution of hyperbolic systems. *arXiv preprint arXiv:1903.08928*, 2019.

[5] V. A. Dobrev, Tz. Kolev, N. A. Petersson, and J. B. Schroder. Two-level convergence theory for multigrid reduction in time (mgrit). *SIAM Journal on Scientific Computing*, 39(5):S501–S527, 2017.

[6] John K Dukowicz and Richard D Smith. Implicit free-surface method for the bryan-cox-semtner ocean model. *Journal of Geophysical Research: Oceans*, 99(C4):7991–8014, 1994.

[7] Pedro F. Embid and Andrew J. Majda. Averaging over fast gravity waves for geophysical flows with arbitrary potential vorticity. *Communications in Partial Differential Equations*, 21(3-4):619–658, 1996.

[8] Bjorn Engquist, Henrik Holst, and Olof Runborg. Multiscale methods for the wave equation. *PAMM*, 7(1):1140903–1140904, December 2007.

[9] Bjorn Engquist and Yen-Hsi Tsai. Heterogeneous multiscale methods for stiff ordinary differential equations. *Mathematics of Computation*, 74(252):1707–1743, May 2005.

[10] R. D. Falgout, S. Friedhoff, Tz. V. Kolev, S. P. MacLachlan, and J. B. Schroder. Parallel time integration with multigrid. *SIAM J. Sci. Comput.*, 36(6):C635–C661, 2014.

[11] R. D. Falgout, T. A. Manteuffel, B. O'Neill, and J. B. Schroder. Multigrid reduction in time for nonlinear parabolic problems: A case study. *SIAM Journal on Scientific Computing*, 39(5):S298–S322, 2017.

[12] Luigi Fraccarollo and Eleuterio F. Toro. Experimental and numerical assessment of the shallow water model for two-dimensional dam-break type problems. *Journal of Hydraulic Research*, 33(6):843–864, 1995.

[13] Stephanie Friedhoff, Robert D Falgout, TV Kolev, S MacLachlan, and Jacob B Schroder. A multigrid-in-time algorithm for solving evolution equations in parallel. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2012.

[14] M. J. Gander. 50 years of time parallel time integration. In T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, editors, *Multiple Shooting and Time Domain Decomposition*, pages 69–114. Springer, 2015.

[15] Martin J. Gander. Analysis of the parareal algorithm applied to hyperbolic problems using characteristics. 2008.

[16] Martin J Gander and Stefan Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing*, 29(2):556–578, 2007.

[17] Terry Haut and Beth Wingate. An asymptotic parallel-in-time method for highly oscillatory pdes, 2013.

[18] Andreas Hessenthaler, David Nordsletten, Oliver Röhrle, Jacob B Schroder, and Robert D Falgout. Convergence of the multigrid reduction in time algorithm for the linear elasticity equations. *Numerical Linear Algebra with Applications*, 25(3):e2155, 2018.

[19] Alexander J Howse, Hans De Sterck, Robert D Falgout, Scott MacLachlan, and Jacob Schroder. Parallel-in-time multigrid with adaptive spatial coarsening for the linear advection and inviscid burgers equations. *SIAM Journal on Scientific Computing*, 41(1):A538–A565, 2019.

[20] Oliver A. Krzysik, Hans De Sterck, Scott P. MacLachlan, and Stephanie Friedhoff. On selecting coarse-grid operators for parareal and mgrit applied to linear advection, 2019.

[21] Suhas Kumar. Fundamental limits to moore's law, 2015.

[22] Randall J. LeVeque. A large time step generalization of godunov's method for systems of conservation laws. *SIAM Journal on Numerical Analysis*, 22(6):1051–1073, 1985.

[23] J. L. Lions, Y. Maday, and G. Turinici. Résolution d'EDP par un schéma en temps pararéel. *C.R.Acad Sci. Paris Sér. I Math*, 332:661–668, 2001.

[24] Andrew Majda. *Introduction to P.D.E.'s and waves for the atmosphere and ocean.* Courant Lecture Notes. New York University, Courant Institute of Mathematical Sciences and American Mathematical Society, 2002.

[25] 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.

[26] J. Murillo, P. García-Navarro, P. Brufau, and J. Burguete. Extension of an explicit finite volume method to large time steps (cfl¿1): application to shallow water flows. *International Journal for Numerical Methods in Fluids*, 50(1):63–102, 2006.

[27] Allan S. Nielsen, Gilles Brunner, and Jan S. Hesthaven. Communication-aware adaptive parareal with application to a nonlinear hyperbolic system of partial differential equations. *Journal of Computational Physics*, 371:483 – 505, 2018.

[28] Adam Peddle. Cyclops-lite. https://github.com/agpeddle/cyclops-lite.

[29] Adam Peddle. Components of nonlinear oscillation and optimal averaging for stiff pdes, Jan 2018.

[30] Adam Peddle, Terry Haut, and Beth Wingate. Parareal convergence for oscillatory pdes with finite time-scale separation, 2017.

[31] Manfred Ries, Ulrich Trottenberg, and Gerd Winter. A note on mgr methods. *Linear Algebra and its Applications*, 49:1 – 26, 1983.

[32] Daniel Ruprecht. Wave propagation characteristics of parareal, 2017.

[33] Martin Schreiber, Andreas Schmitt, Pedro Peixoto, and Michael Schäfer. A numerical study of a semi-lagrangian parareal method applied to the viscous burgers equation. *Computing and Visualization in Science*, 19, 12 2017.

[34] Jacob Bayer Schroder and Benjamin W. Ong. Applications of time parallelization. *Computing and Visualization in Science, Springer*, (accepted, June, 2020).

[35] Johannes Steiner, Daniel Ruprecht, Robert Speck, and Rolf Krause. Convergence of parareal for the navier-stokes equations depending on the reynolds number, 2013.

[36] Hans De Sterck, Robert D. Falgout, Stephanie Friedhoff, Oliver A. Krzysik, and Scott P. MacLachlan. Optimizing mgrit and parareal coarse-grid operators for linear advection, 2019.

[37] B. A. Wingate. The maximum allowable time step for the shallow water α-model and its relation to time-implicit differencing. *Monthly Weather Review*, 132(12):2719–2731, 2004.

[38] Renyi Xu, D. Zhong, Baosheng Wu, Xudong Fu, and Runze Miao. A large time step godunov scheme for free-surface shallow water equations. *Chinese Science Bulletin*, 59:2534–2540, 07 2014.

Disclaimer: This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.