MULTIGRID REDUCTION IN TIME FOR NONLINEAR PARABOLIC PROBLEMS

BEN O'NEILL *

Abstract. The need for parallelism in time is being driven by changes in computer architectures, where future speed-ups will be available through greater concurrency, not faster clock speeds. This leads to a bottleneck for sequential time marching schemes because they lack parallelism in the time dimension. Multigrid Reduction in Time (MGRIT) is an iterative procedure that allows for temporal parallelism by utilizing multigrid reduction techniques and a multilevel hierarchy of coarse time grids. The goal of this work is the efficient solution of nonlinear problems with MGRIT, where efficiency is defined as achieving similar performance when compared to an equivalent linear problem. When solving a linear problem, using implicit methods and optimal spatial solvers, e.g. classical multigrid, the spatial multigrid convergence rate is fixed across temporal levels, despite a large variation in time step sizes. This is not the case for nonlinear problems, where the work required increases dramatically on coarser time grids. By using a variety of strategies, most importantly, spatial coarsening and an alternate initial guess for the nonlinear solver, we reduce the work per time step evaluation over all temporal levels to a range similar to those of a corresponding linear problem. This allows for overall speedups comparable with those achieved, in previous work, for linear systems.

1. Introduction. Previously, ever increasing clock speeds allowed for the speedup of sequential time integration simulations of a fixed size, and for stable wall clock times for simulations that were refined in space (and usually time). However, clock speeds are now almost stagnant, leading to the sequential time integration bottleneck.

By allowing for parallelism in time, much greater computational resources can be brought to bear, and overall speedups can be achieved. Because of this, interest in parallel-in-time methods has grown over the last decade. Here, the focus is on the multigrid reduction in time (MGRIT) method [4]. MGRIT is a true multilevel algorithm and, as such, has optimal parallel communication behavior, as opposed to a two-level scheme, where the size of the coarse-level limits concurrency.

Work on parallel-in-time methods actually goes back at least 50 years [12]. For a gentle introduction to this history, please see the review paper [6]. This work focuses on multigrid approaches (and MGRIT [4, 5] in particular) because of multigrid's optimal algorithmic scaling for both parallel communication and number of operations. Note that Parareal [9], perhaps the most well known parallel-in-time method, is equivalent [7] to a two-level multigrid scheme.

Consider a general first order ordinary differential equation (ODE) and the corresponding time discretization:

$$u_t = f(u,t),\ u(0) = u_0,\ t \in [0,T], \tag{1}$$

$$u(t+\delta t) = \Phi(u(t), u(t+\delta t)) + g(t+\delta t), \tag{2}$$

where Φ is a nonlinear operator that encapsulates the chosen time stepping routine and g is a time dependent function that incorporates all the solution independent terms.

Sequential time marching schemes are optimal in that they move from time $t = 0$ to $t = T$ with the fewest possible applications of Φ. By applying Φ iteratively in, comparably expensive, but highly parallel, multigrid cycles, MGRIT sacrifices efficiency for temporal concurrency. Both methods are optimal, i.e. $O(N)$, but the constant for MGRIT is higher.

Application of MGRIT to linear parabolic problems was studied in [4]. Figure 1 shows a strong scaling study of MGRIT for linear diffusion on a $(257)^2 \times 16385$ space time grid. The space-time parallel runs used an 8×8 processor grid in space, with all additional processors

*Department of Applied Mathematics, University of Colorado at Boulder, Boulder, Colorado

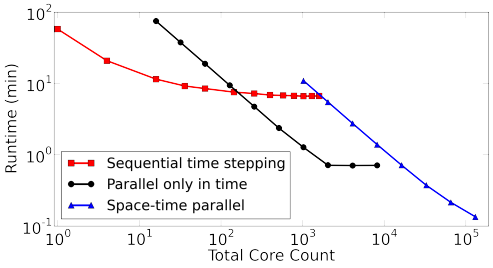

Fig. 1: Time to solve $2D$ linear diffusion on a $(257)^2 \times 16385$ space-time grid using sequential time stepping and two processor decompositions of MGRIT. [4]

being added in time. Both MGRIT curves represent the use of temporal and spatial coarsening, so that the ratio of dt/dx^2 is fixed on coarse time-grids. The goal of this paper is to make the overall performance (i.e., crossover point and speedup) of MGRIT for nonlinear problems similar to that for linear problems.

When considering the performance of MGRIT, the application of Φ is the dominant process. When an optimal spatial solver such as classical spatial multigrid [10, 3, 13, 8] is used, the work required for a time step evaluation is independent of the time step size. However, when Φ is nonlinear, each application of Φ is an iterative nonlinear solve, the conditioning of which usually does depend on the time step size. The nonlinear parabolic p-Laplacian equation was selected as the model problem:

$$u_t(\mathbf{x},t) - \nabla \cdot (|\nabla u(\mathbf{x},t)|^{p-2}\nabla u(\mathbf{x},t)) = b(\mathbf{x},t),\ \mathbf{x} \in \Omega,\ t \in [0,T], \tag{3}$$

subject to the following Neumann boundary and initial conditions:

$$|\nabla u(\mathbf{x},t)|^{p-2}\nabla u(\mathbf{x},t) \cdot \mathbf{n} = g(\mathbf{x},t),\ \mathbf{x} \in \partial\Omega,\ t \in (0,T] \tag{4}$$

$$u(\mathbf{x},0) = u_0(\mathbf{x}),\ \mathbf{x} \in \Omega. \tag{5}$$

The p-Laplacian for $p = 4$ is well-known as a means of modeling soil erosion and transport [1] and has also found uses in image processing (denoising, segmentation and inpainting) and machine learning.

A naive application of MGRIT to (3) showed a large increase in nonlinear iteration counts, per Φ evaluation, on the coarser temporal grids. Therefore, our strategy is to minimize the cost of each nonlinear solve. The goal is to ultimately achieve similar efficiencies for the nonlinear and linear versions of (3). Note that the $p = 2$ is the heat equation.

Towards the goal of minimizing the overall cost of each nonlinear solve, a spatial coarsening strategy, where $\delta t/\delta x^2$ is held fixed, was pursued. This kept the conditioning of Φ fixed over all time levels, ultimately requiring far fewer nonlinear iterations per Φ evaluation. In addition, the smaller problem sizes drastically reduced coarse grid compute times. Unfortunately, full spatial coarsening can degrade overall MGRIT convergence and, in some cases, result in non-convergence. Maintaining the full spatial resolution on the first few coarse time levels, but coarsening in space on the coarser levels, minimized the negative effects on MGRIT convergence, while still dramatically reducing the overall cost of the nonlinear solve.

To further reduce the average cost of each Φ evaluation, an improved initial guess was also investigated. A commonly used approach is to use the previous time step as the initial guess for the nonlinear solver. On coarse time grids, this is a poor approximation to the solution. Instead, the approximate solution at the corresponding time point from the previous MGRIT iteration is used. This reduces the average number of nonlinear iterations per Φ evaluation, to below that of an equivalent sequential time integration.

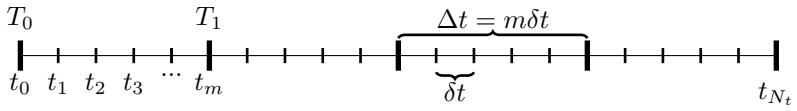

Fig. 2: Fine- and coarse-grid temporal meshes. Fine-grid points are present on only the fine-grid, whereas coarse-grid points are on both the fine- and coarse-grid.

In Section 2, the general MGRIT framework is discussed. In Section 3, our strategy for improving the performance of MGRIT for nonlinear problems is presented and justified. In Section 4, by introducing spatial coarsening and an improved initial guess, this strategy is implemented. In sections 5.1 and 5.2, weak and strong scaling results are presented. Weak scaling results show that the MGRIT algorithm scales effectively, while strong scaling results show overall speedups in line with the comparable linear problem.

2. MGRIT overview. First, a brief overview of the MGRIT algorithm is presented. Define a uniform temporal grid with time step δt and nodes $t_j,\ j = 0, \ldots, N_t$ (non-uniform grids can easily be accommodated). Further, define a coarse temporal grid with time step $\Delta T = m\delta t$ and nodes $T_j = j\Delta T,\ j = 0, 1, \ldots, N_t/m$, for some coarsening factor m. This is depicted in Figure 2. In block triangular form, the time-stepping problem (2) is

$$A(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi_0 & I & & \\ & \ddots & \ddots & \\ & & -\Phi_{N_t-1} & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{N_t} \end{bmatrix} = \mathbf{g}. \tag{6}$$

Sequential time marching is a forward block solve of this system. MGRIT solves this system iteratively, in parallel, using a coarse-grid correction scheme based on multigrid reduction. Both are $O(N)$ methods but MGRIT is highly concurrent. Multigrid reduction strategies are essentially approximate cyclic reduction methods and, as such, successively eliminate unknowns in the system. If the fine points are eliminated, the system becomes

$$A_\Delta(\mathbf{u}_\Delta) = \begin{bmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & & \\ & & -\Phi^m & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\Delta,0} \\ \mathbf{u}_{\Delta,1} \\ \vdots \\ \mathbf{u}_{\Delta,N_t} \end{bmatrix} = \mathbf{g}_\Delta, \tag{7}$$

By defining "ideal" restriction, R, and interpolation, P, this system can be represented in a multigrid fashion. Let R be injection, and

$$P = \begin{bmatrix} I & \Phi^T & \ldots & \Phi^{m-1,T} & & & & \\ & & & \ddots & & & & \\ & & & & I & \Phi^T & \ldots & \Phi^{m-1,T} \end{bmatrix}^T. \tag{8}$$

With this, the "ideal" coarse-grid operator is $A_\Delta = RAP$. The limitation of this exact reduction method is that the coarse-grid problem is, in general, as expensive to solve as the original fine-grid problem. Multigrid reduction methods address this by approximating A_Δ

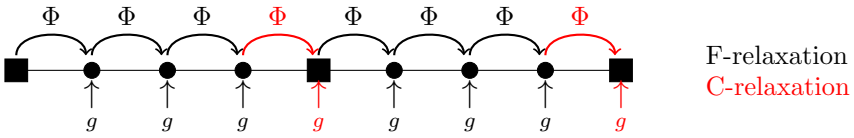

Fig. 3: F- and C-relaxation for coarsening by factor of 4 . Note that each F-interval and C-update is independent.

with B_Δ, where

$$B_\Delta = \begin{bmatrix} I & & & \\ -\Phi_\Delta & I & & \\ & \ddots & & \\ & & -\Phi_\Delta & I \end{bmatrix}, \tag{9}$$

and Φ_Δ is an approximate coarse-grid time stepper. One obvious choice for defining Φ_Δ is to re-discretize the problem on the coarse grid. Convergence of MGRIT is governed by the approximation $A_\Delta \approx B_\Delta$. Using a re-discretization of Φ with $\Delta T = m\delta t$ has proven effective [4, 5].

The coarse-grid is used to compute an error correction based on the residual equation (see Algorithm 1). Relaxation is a local fine-grid process used to resolve fine-scale behavior. Figure 3 shows the actions of F- and C-relaxation on a temporal grid with $m = 4$. F-relaxation propagates the solution, forward in time, from each coarse point to the neighboring F-points. Each interval of F-points is updated independently (i.e, in parallel) during F-relaxation. Each C-point update is similarly independent.

2.1. MGRIT algorithm for nonlinear problems. Putting the above components together results in the MGRIT algorithm for nonlinear problems. This is a straight-forward extension of the linear algorithm [4] using the FAS (nonlinear multigrid) scheme [2]. The F-relaxation, two-grid variant is equivalent to the popular Parareal algorithm [7], which has been shown to be effective for a variety of nonlinear problems. The FAS description of MGRIT first appeared in [5].

Injection is used to map to the coarse level (like Parareal). The exception is the *spatial coarsening* option where the spatial restriction and interpolation functions, R_x and P_x, are used to coarsen in space. With this, the two-level the MGRIT algorithm is presented in Algorithm 1.

Algorithm 1 MGRIT(A, $\mathbf{u}$, $\mathbf{g}$)

1: Apply F- or FCF-relaxation to $A(\mathbf{u}) = \mathbf{g}$.
2: Inject the fine grid approximation and its residual to the coarse grid
$u_{\Delta,i} \leftarrow u_{mi}, \quad r_{\Delta,i} \leftarrow g_{mi} - (A(\mathbf{u}))_{mi}$
3: **If** Spatial coarsening **then**
$u_{\Delta,i} \leftarrow R_x(u_{\Delta,i}), \quad r_{\Delta,i} \leftarrow R_x(r_{\Delta,i})$
4: Solve $B_\Delta(\mathbf{v}_\Delta) = B_\Delta(\mathbf{u}_\Delta) + \mathbf{r}_\Delta$.
5: Compute the coarse grid error approximation: $\mathbf{e}_\Delta \simeq \mathbf{v}_\Delta - \mathbf{u}_\Delta$
6: **If** Spatial coarsening **then**
$e_{\Delta,i} \leftarrow P_x(e_{\Delta,i})$,
7: Correct using ideal interpolation: $\mathbf{u} = \mathbf{u} + P\mathbf{e}_\Delta$

A variety of cycling strategies are available in multigrid (e.g., V, W, F). All the results

δt	δx	Av.	Max	Min
1/1024	1/32	3.4	4	2
1/256	1/16	3.8	6	2
1/64	1/8	4.9	12	2
1/16	1/4	6.7	12	3
1/4	1/2	9	13	5
1	1	7.5	12	5

(a) Full spatial Coarsening

δt	δx	Av.	Max	Min
1/1024	1/32	3.4	4	2
1/256	1/32	4.01	5	3
1/64	1/32	7.0	11	3
1/16	1/32	10.01	17	4
1/4	1/32	12.8	16	8
1	1/32	11.7	15	8

(b) No spatial Coarsening

Table 1: Baseline iteration counts for Newton solver using the sequential method.

presented here use the standard V-cycle, corresponding to Algorithm 1 with the "Solve" step turned into a single recursive call. The recursion ends when a trivially sized grid is reached, at which point a sequential solver is used.

The chosen implementation of MGRIT is XBraid [14], an open source package developed at Lawrence Livermore National Laboratory (*LLNL*). The key computational kernel is the time-stepping routine, but all the specifics are opaque to XBraid and done in user code. This allows the user to add temporal parallelism to existing time stepping routines with minimal modifications. For more details, see [4] and [14].

3. Strategy for efficient MGRIT.

3.1. Numerical parameters. The forcing function, Neumann boundary conditions, and initial conditions of the model problem (3) were chosen to prescribe an exact solution of $u(x, y) = \sin(\kappa x)\sin(\kappa y)\sin(\tau t)$, where $\kappa = \pi$ and $\tau = (2 + 1/6)\pi$. Unless otherwise stated, the p-Laplacian with $p = 4$ was used. The spatial discretization was computed with standard linear quadrilateral elements using MFEM [11]. Time stepping was completed using Backward Euler, with Newton as the nonlinear solver.

The numerical testing parameters (unless otherwise mentioned) were as follows. An absolute, residual based, Newton tolerance of 10^{-7} was used. The spatial solver for each Newton iteration was BoomerAMG [8].

The test problem size was a $(64)^2 \times 4096$ space-time grid on the domain $[0, 2]^2 \times [0, 4]$ with 1 processor in space and 64 processors in time. MGRIT used V-cycles, FCF-relaxation, and a fixed stopping criteria of $10^{-9}/(\sqrt{\delta t}\delta x)$. The coarsening factor was $m = 4$.

3.2. Sequential time-stepping baseline for efficiency. To establish a baseline for efficiency, experiments using standard sequential time-stepping were completed. The previous time-step value was used as the initial guess for the Newton solve.

Table 1 gives the average, maximum and minimum Newton solver iteration counts over several space-time grids. On the left, the grids used by MGRIT when applying full spatial coarsening are mimicked. On the right the iteration counts for grids associated with no spatial coarsening are given.

These baseline results make it clear that the average cost of a nonlinear solve is highly dependent on the grid, with there being a distinct advantage to coarsening in both space and time.

The dependence on δx (i.e, spatial coarsening) is explained by considering a standard backward Euler time step,

$$\left(I - \frac{\delta t}{\delta x^2} A(u_{k+1})\right) = f(u_k),$$

where A is some nonlinear diffusion integrator. As $\delta t/\delta x^2$ increases, the nonlinear operator

Iteration	$\delta t = 9.77e$-4	3.91e-3	1.56e-3	6.25e-2	2.50e-1	1.0
0	5.45	5.60	6.62	9.71	14.15	14.75
1	3.59	4.10	6.93	10.94	14.78	15.75
2	3.44	4.04	7.01	10.35	14.42	16.50
3	3.44	4.01	7.18	10.32	14.52	16.25
4	3.44	4.01	7.18	10.35	14.40	16.50

Table 2: The average number of Newton iterations per time step (a_ℓ) across each temporal level and MGRIT iteration

moves away from the identity, becoming more expensive to solve. Coarsening in space and time bounds this ratio, making the nonlinear inversion of Φ cheaper on the coarse grid.

Despite this, the iteration counts continue to grow with the time step size. This is likely attributed to our choice of initial guess. On coarse time levels, where δt is large, the previous time step is clearly a poor approximation to the current solution, and as such, more iterations are required for convergence.

3.3. Strategy for naive MGRIT. Our strategy is to minimize the average cost of a Newton solve, c_n. The cost of a Newton solve depends on both the number of iterations required for convergence, and the cost of a linear solve. Let a be the average number of Newton iterations per time step over all levels, and let c_l be the average computational cost of a single time step for the linear solve. Then $c_n \propto a\ c_l$. Here, c_l is an absolute measure that decreases with the spatial grid size.

A single MGRIT V cycle, using FCF relaxation, and no spatial coarsening, completes $\Phi_T = (2m/(m-1)+1)N_t$ calls to the linear solve routine, where m is the coarsening factor and N_t is the total number of time steps. In [4] it was shown that, for a linear problem, the computational model for MGRIT's cost, c_{lin}, is essentially defined by the number of calls to the linear solve routine. Thus, $c_{lin} \propto c_l\ \Phi_T$. In a nonlinear setting, the cost of a MGRIT V cycle is

$$c_{nl} \propto a\ c_l\ \Phi_T \propto c_n\ \Phi_T.$$

By minimizing c_n we can greatly reduce the cost of a MGRIT V cycle. Our heuristic is that this minimization of c_n (through reductions in both a and c_l) will yield a nearly optimal solver.

As a baseline, MGRIT was applied in a naive fashion, as one might apply MGRIT initially to an existing sequential implementation of the model problem. Table 2 gives the average number of Newton iterations per time step (a_ℓ), on each temporal level, with no spatial coarsening. Each δt (column) value corresponds to a temporal level, while the rows represent different XBraid iterations.

As mentioned above, when using a multigrid solver, the cost of a linear solve, c_l, is independent of the time step size. This is clearly not the case for a nonlinear solve. Newton iteration counts grow steadily across temporal grids, with 3-5 times as many Newton iterations taking place on the coarsest grids. One might initially guess that increased Newton iteration counts on the coarse grid are irrelevant. After all, for this example we took around 900 Newton iterations on the coarsest grid, compared to 200 000 on the fine grid. However, in a highly parallel setting, where concurrency is key, increased iteration counts on coarse grids can substantially reduce performance.

Consider the case where each MPI process owns m points in time on the finest level, i.e., one CF-interval. On the coarsest level, each process owns, at most, one point in time. Due to synchronization, the slowest processor determines the cost of each solve. In the example above, on iteration 3, the cost of FCF-relaxation on the coarsest grid is approximately $33c_l$ (each point is relaxed twice), compared to $8c_l$ on the second time grid ($\delta t = 3.91e$-3). In this case, the coarse grid is definitely not free, it is the most expensive part of a V cycle.

Initial guess	Spatial Level	MGRIT iter.	Total Wall-clock time	Wall-clock time/iter.	a
PTS	1	12.00	11269.82	939.15	4.19
PTS	3	12.00	9602.59	800.22	4.18
PTS	4	12.00	8504.12	708.68	4.15
PTS	6	40.00+	17470.33+	436.76	4.14
PMI	1	12.00	7121.18	593.43	2.82
PMI	4	12.00	5692.31	474.36	2.82

Table 3: Overall run-times, total iteration counts and average time per iteration for our various solver options, with a $(64)^2 \times 4096$ space-time grid.

4. Efficient MGRIT for the model nonlinear problem. In this section a variety of approaches designed to make MGRIT efficient, for our chosen model nonlinear parabolic problem (3), are investigated. The goal is to achieve a similar efficiency as that seen for a corresponding linear problem.

4.1. MGRIT with spatial coarsening. Motivated by the results seen in Section 3.2, the naive solver from Section 3 was updated with spatial coarsening. In general, the user's code defines the separate spatial interpolation and restriction functions P_x and R_x. The natural finite element restriction operator (and its transpose) was used to move between regularly refined grids. This operator was provided by MFEM. The previous time step was used as the initial guess for the Newton solver.

Using coarse spatial grids, on the coarse time levels, drastically reduces the cost per iteration (c_l). The trade-off is that the coarse grid solves are less accurate, which, in turn, can reduce the MGRIT convergence rate. One benefit of MGRIT is that, given the exact solution on the coarse grid, interpolation yields the exact solution on fine grid. Error introduced by restriction and interpolation between spatial meshes removes this property. Without this exactness, any error modes introduced, specifically error modes in the null space of the restriction operator, must be damped solely by FCF-relaxation. In many cases this also causes a degradation of the MGRIT convergence rate.

Table 3 shows the effect of spatial coarsening on MGRIT run times (called wall-clock time). Here, "initial guess" corresponds to the initial guess for the Newton solver with "PTS" referring to the previous time step, and "PMI" referring to the previous MGRIT iteration (see section 4.2). "Spatial levels" represents the number of different spatial grids used. The value of 1 corresponds to no spatial coarsening, i.e. one fixed spatial grid for all time levels, while 6 represents a different spatial grid on every temporal level. In the case where 4 spatial grids were used, full spatial resolution was kept on the 2 largest temporal grids, with spatial coarsening used on time grids $3-6$. A similar approach was used for the case where 3 spatial grids were used.

The reader will first note that the case of 6 yielded an MGRIT algorithm that did not converge in the allowed amount of time. This severe degradation of MGRIT convergence is still a subject of active research, but likely relates to the discussion above regarding the information lost when moving to coarse levels. For practical purposes, this solver is unusable. Delaying spatial coarsening until the third or fourth temporal level limited any degradation in the MGRIT convergence rate, while still allowing for dramatic reductions in c_l.

Moreover, replicating results seen in Section 3.2, spatial coarsening slightly reduced a, the average number of Newton iterations required per time step. Table 4 explores this idea by comparing a_ℓ on each temporal level and iteration. The values for a_ℓ increase with temporal level in Table 4, but considerably less than in Table 2.

In conclusion, if spatial coarsening is not possible on all temporal levels, as is the case here,

Iteration	$\delta t = 9.77e$-4	3.91e-3	1.56e-3	6.25e-2	2.50e-1	1.0
0	5.45	5.63	6.74	8.93	11.42	10.25
1	3.60	4.09	6.92	8.93	11.43	10.75
2	3.44	4.04	7.02	8.82	11.43	10.50
3	3.44	4.01	7.18	8.84	11.43	10.50
4	3.44	4.01	7.18	8.84	11.43	10.50

Table 4: The average number of Newton iterations per time step (a_ℓ) with spatial coarsening across each temporal level and MGRIT iteration. Four levels of spatial coarsening are used. c.f. table 2.

keeping full spatial resolution on the finest grids, but coarsening after that, is a good way to balance both the accuracy and the cost of a MGRIT V-cycle. By reducing the size of the coarse grids, spatial coarsening reduces c_l. Furthermore, by minimizing $\delta t/\delta x^2$, a_ℓ was drastically reduced on the coarse grids. Even so, these a_ℓ values are still higher than those for the same δt and δx in Table 1, leaving room for improvement.

4.2. MGRIT with an improved initial guess for Newton's method. In Section 3.2, it was suggested that the dependence of a_ℓ on time step size was due to our choice of initial guess. Recall that using the previous time step as the initial guess for the nonlinear solver is a common approach, and that on coarse time grids, this is a poor approximation to the solution. After MGRIT has completed one iteration, the user has two choices, the previous time step, and the solution from the previous MGRIT iteration *(PMI)*. As MGRIT converges, the solution from the previous iteration becomes an ever improving initial guess.

Table 5 shows a_ℓ over 5 iterations, using the PMI as the initial guess, for the case of no spatial coarsening. Large reductions in a_ℓ were seen, but primarily at the finer levels. Compared with Table 2, a large reduction in a_ℓ across all temporal grids was achieved. Furthermore, reductions were seen during the later MGRIT iterations. By iteration three a_ℓ was comparable, across all grids, with those from the sequential solver in Table 1. Similar trends were seen when using spatial coarsening (4 levels) in conjunction with PMI.

Iteration	$\delta t = 9.77e$-4	3.91e-3	1.56e-3	6.25e-2	2.50e-1	1.0
0	6.99	6.85	6.89	6.97	8.97	12.00
1	4.34	3.77	4.14	5.42	8.47	12.50
2	3.18	3.01	3.32	5.53	7.02	10.75
3	2.41	2.19	2.90	5.39	7.07	10.25
4	2.00	2.00	2.91	5.44	7.00	10.50

Table 5: Comparison of the average number of Newton iterations per time step (a_ℓ), across each temporal level and MGRIT iteration, when using the PMI as the initial guess to the nonlinear solver.

Table 3 further validates our heuristic. By using the PMI as the initial guess we almost halved a_ℓ, and, in doing so, drastically reduced overall run times.

In conclusion, these results indicate that using the PMI as the initial guess, after the first MGRIT iteration, is very beneficial. During the first MGRIT iteration, no value but the previous time step exists, and thus, it must be used as the initial guess.

5. Numerical results. Previous sections focused on producing the most efficient Newton solver as a proxy for MGRIT efficiency. Parallel scaling studies were completed to validate those findings. Let *SC* stand for spatial coarsening.

5.1. Weak scaling. In this subsection, a domain refinement study for MGRIT, using the above strategies, is presented. The space-time domain was held fixed ($[0,2]^2 \times [0,4s]$), while the spatial and temporal resolution were scaled up, holding $\delta t/\delta x^2$ fixed. In all cases 16 time

$N_x^2 \times N_t$	$64^2 \times 16$	$128^2 \times 64$	$256^2 \times 256$	$512^2 \times 1024$	$1024^2 \times 4096$
PTS	4	9	11	–	–
PMI	4	9	11	11	11
PMI + SC	4	9	11	11	11

Table 6: Weak scaling study for nonlinear MGRIT algorithm. MGRIT iteration counts are bounded independently of the problem size. The PMI was used at all points and 4 levels of spatial coarsening (SC) were used.

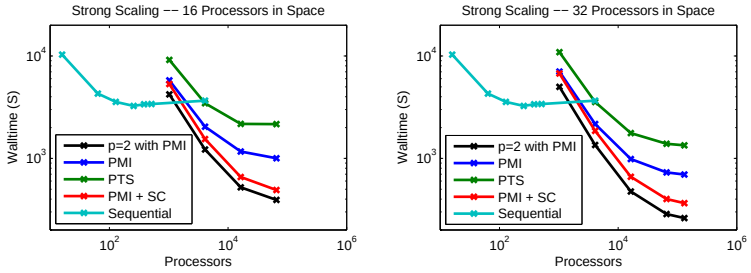

Fig. 4: Strong scaling study for a $(128)^2 \times 16385$ space-time grid, Left: 16 processors in space, Right: 32 processors in space.

steps, and 4096 spatial unknowns per processor were used. For runs using spatial coarsening, the number of levels of spatial coarsening was increased on each subsequent test, resulting in 4 levels of spatial coarsening on a $(1024)^2 \times 4096$ space-time grid.

Results are shown in Table 6. For the PTS case, the final two grid sizes are omitted because the run-times became excessive given our machine privileges. The PMI and PMI + SC cases both showed optimal iteration counts, bounded independently of problem size.

5.2. Strong scaling. Both MGRIT and classical time stepping are $O(N)$ optimal methods, but the constant for MGRIT is larger. On the other hand, MGRIT allows for temporal parallelism. This leads to a crossover point, after which MGRIT is beneficial. To illustrate this, a strong scaling study of MGRIT, for the space-time grid of $(128)^2 \times 16384$, was completed. Figure 4 shows the results. The plot for "p=2" corresponds to the linear heat equation with $p = 2$ in equation (3). This allows us to compare MGRIT for the nonlinear problem to a corresponding linear problem. For this experiment no optimizations, other than setting $p = 2$, were made, e.g., the spatial matrix and solver were rebuilt during each application of Φ, as was required in the nonlinear case, so that the comparison is fair.

From this, a number of conclusions can be made. For small processor counts, sequential time stepping is both faster and uses less memory. However, on larger numbers of processors MGRIT is faster. The crossover point at which using MGRIT was beneficial, for 16 processors in space, was at about 1024 processors, or 64 processors in time. At it's maximum the speedup was about a factor of 10 at 130K cores and 32 processors in space.

The results in Figure 4 are not as good as those presented in Figure 1. The behavior of the linear heat equation ($p = 2$) is similar to that for the nonlinear p-Laplacian ($p = 4$), which indicates the difference may be due to: (1) using linear finite elements, on a regular grid in space, as opposed to finite differencing, (2) using the BoomerAMG solver in hypre , as opposed to the more efficient geometric-algebraic solver PFMG in hypre, and (3) the spatial discretization and spatial multigrid solver being built every time step. Improving the scaling of MGRIT will require addressing these differences, with (2) and (3) the more likely culprits.

This is a topic for future research.

6. Conclusions. The MGRIT algorithm effectively adds temporal parallelism to existing sequential solvers and has been shown to be effective for linear problems. However, when moving to the nonlinear setting, the relatively large time-step sizes on coarse grids make the application of MGRIT nontrivial.

In summary, it was shown that, after the first iteration, the user should always use the solution from the previous MGRIT iteration as the initial guess to the nonlinear time-stepping routine (here, a Newton solver). Moreover, spatial coarsening should be used whenever possible. For the linear example, presented in Figure 1, spatial coarsening was implemented on all levels effectively. For the p-Laplacian this was not the best strategy. However, keeping full spatial resolution on the second, and possibly the third, time grid, but coarsening after that, was very effective. This approach dramatically reduced the cost of a Newton solve on the coarse grid, whilst also limiting degradation of the MGRIT convergence rate.

Weak scaling results showed that MGRIT is a scalable algorithm for nonlinear problems, with iteration counts bounded independently of problem size. Strong scaling showed benefits of MGRIT, with 10x speedups seen over sequential time stepping in some parameter regimes. The performance is not as ideal as in [4], but with the modifications given here, we were able to match performance for the comparable linear problem.

Future work will involve addressing some of the efficiency issues in the MFEM finite-element matrix construction routines and BoomerAMG setup-phase, as well as varying the nonlinear solver tolerance, in a level and iteration based way, in an attempt to further reduce a_ℓ.

REFERENCES

[1] B. Bjorn and J. Rowlett, *Mathematical models for erosion and the optimal transportation of sediment*, International Journal of Nonlinear Sciences and Numerical Simulation, 14 (2013), pp. 323–337.

[2] A. Brandt, *Multi-level adaptive computations in fluid dynamics*, 1979. Technical Report AIAA-79-1455, AIAA, Williamsburg, VA.

[3] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge Univ. Press, Cambridge, 1984, pp. 257–284.

[4] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder, *Parallel time integration with multigrid*, SIAM Journal on Scientific Computing, 36 (2014), pp. C635–C661, doi:10.1137/130944230.

[5] R. D. Falgout, A. Katz, T. Kolev, J. B. Schroder, A. Wissink, and U. M. Yang, *Parallel time integration with multigrid reduction for a compressible fluid dynamics application.*, Journal of Computational Physics, (submitted) (2015).

[6] M. J. Gander, *50 years of Time Parallel Time Integration*, Multiple Shooting and Time Domain Decomposition, Springer, 2015. In press.

[7] M. J. Gander and S. Vandewalle, *Analysis of the parareal time-parallel time-integration method.*, SIAM Journal on Scientific Computing, 29 (2007), pp. 556–578.

[8] *HYPRE: High performance preconditioners.* http://www.llnl.gov/CASC/hypre/.

[9] 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 (2001), pp. 661–668.

[10] S. F. McCormick and J. W. Ruge, *Multigrid methods for variational problems*, SIAM J. Numer. Anal., 19 (1982), pp. 924–929.

[11] *MFEM: Modular finite element methods.* mfem.googlecode.com.

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

[13] J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., Frontiers Appl. Math., SIAM, Philadelphia, 1987, pp. 73–130.

[14] *XBraid: Parallel multigrid in time.* http://llnl.gov/casc/xbraid.