Parallel Time Integration – An Approaching Paradigm Shift for Scientific Computing

The forty-third Woudschoten Conference
Zeist, The Netherlands

October 3 - 5, 2018

Robert D. Falgout
Center for Applied Scientific Computing

LLNL-PRES-759099
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

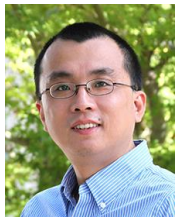
Our Multigrid and Parallel Time Integration Research Team


Veselin Dobrev


Rob Falgout


Tzanio Kolev


Matthieu Lecouvez


Ruipeng Li


Daniel Osei-Kuffuor


Jacob Schroder


Panayot Vassilevski


Lu Wang


Ulrike Yang

- University collaborators and summer interns
  - CU Boulder (Manteuffel, McCormick, Ruge, O'Neill, Mitchell, Southworth), Penn State (Brannick, Xu, Zikatanov), UCSD (Bank), Ball State (Livshits), U Wuppertal (Friedhoff, Kahl), Memorial University (MacLachlan), U Illinois (Gropp, Olson, Bienz), U Stuttgart (Röhrle, Hessenthaler), Monash U (De Sterck), TU Kaiserslautern (Günther)

- Software, publications, and other information

hypre http://llnl.gov/casc/hypre

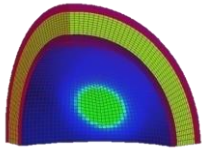
XBRAID http://llnl.gov/casc/xbraid

Outline

- Multigrid (in space)
  - Motivation and background
  - Parallel multigrid
- Multigrid in time
  - Motivation and basic approach
  - MGRIT – multigrid reduction (MGR) in time
  - Progress and current research (a quick preview of Thu)
  - Historical background and connections
  - An Approaching Paradigm Shift for Scientific Computing
- Summary and conclusions

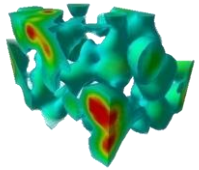
Multigrid will play an important role for addressing exascale challenges

- For many applications, the fastest and most scalable solvers are multigrid methods



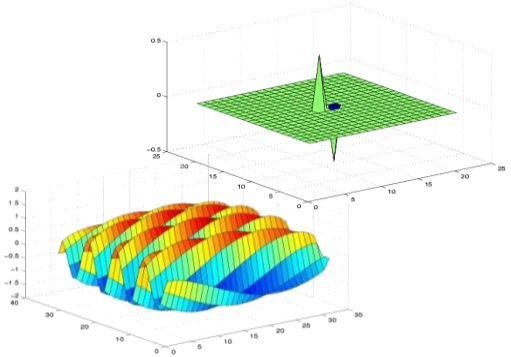
Elasticity / Plasticity

- Exascale solver algorithms will need to:
  - Exhibit extreme levels of parallelism (exascale → 1B cores)
  - Minimize data movement & exploit machine heterogeneity
  - Demonstrate resilience to faults



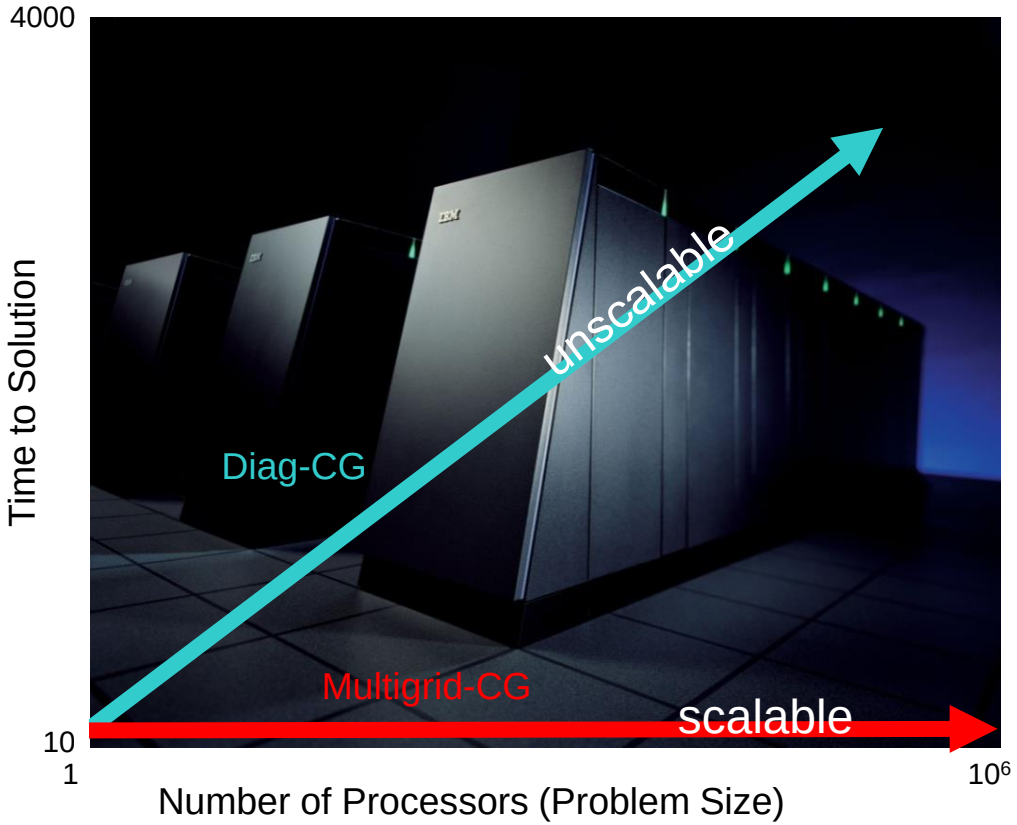
Quantum Chromodynamics

- Multilevel methods are ideal
  - Key feature: Optimal O(N)
- Research challenge:
  - No optimal solvers yet for some applications, even in serial!
  - Parallel computing increases difficulty



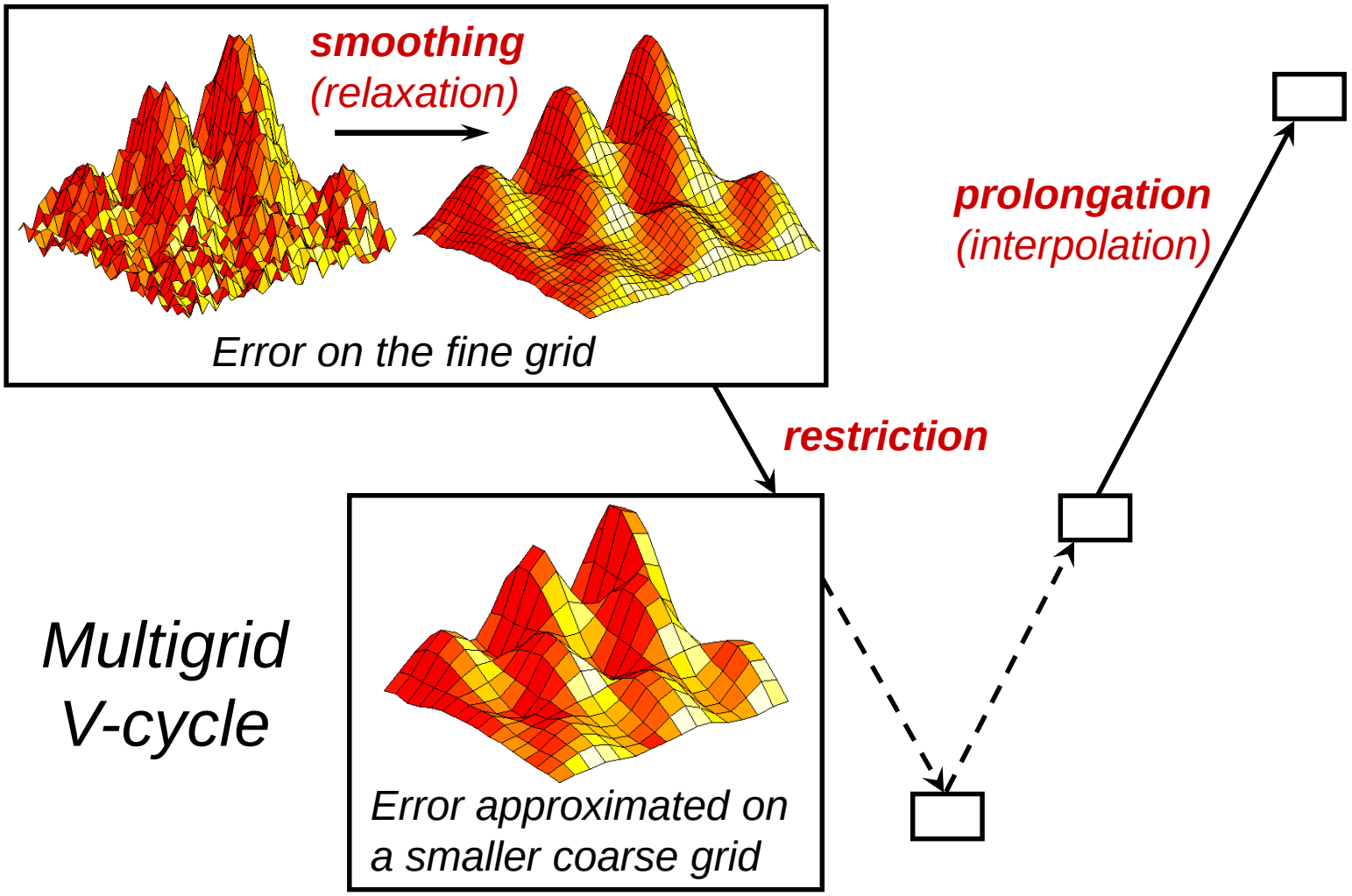
Helmholtz Modes

Multigrid solvers have $O(N)$ complexity, and hence have good scaling potential

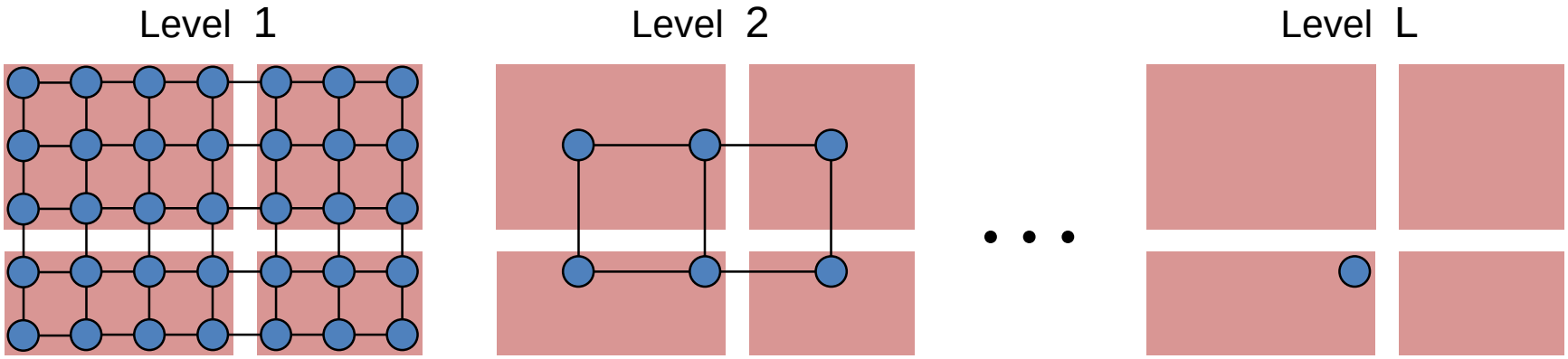


- Weak scaling – want constant solution time as problem size grows in proportion to the number of processors

Multigrid (MG) uses a sequence of coarse grids to accelerate the fine grid solution



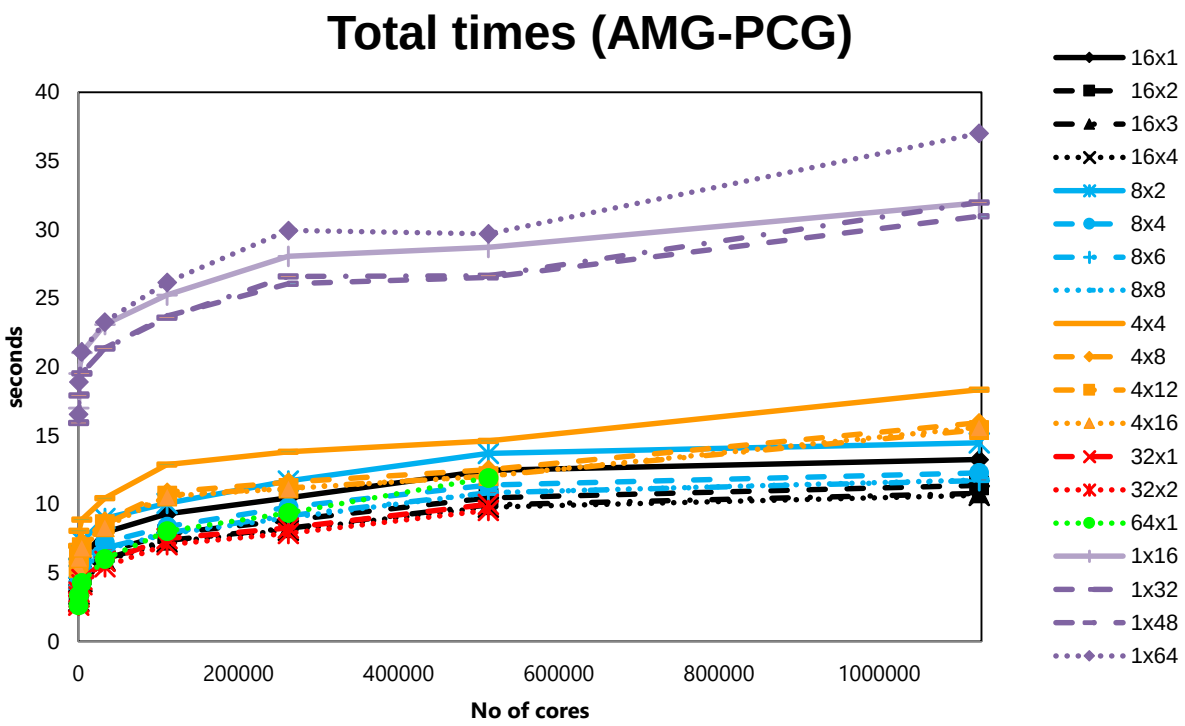
Straightforward MG parallelization yields optimal-order performance for V-cycles



- ~ 1.5 million idle cores on Sequoia!
- Multigrid has a high degree of concurrency
  - Size of the sequential component is only O(log N)!
  - This is often the minimum size achievable
- Parallel performance model has the expected log term

$$T_V = O(\log N)(\text{comm latency}) + O(\Gamma_p)(\text{comm rate}) + O(\Omega_p)(\text{flop rate})$$

Parallel AMG in *hypre* scales to 1.1M cores on Sequoia (IBM BG/Q)

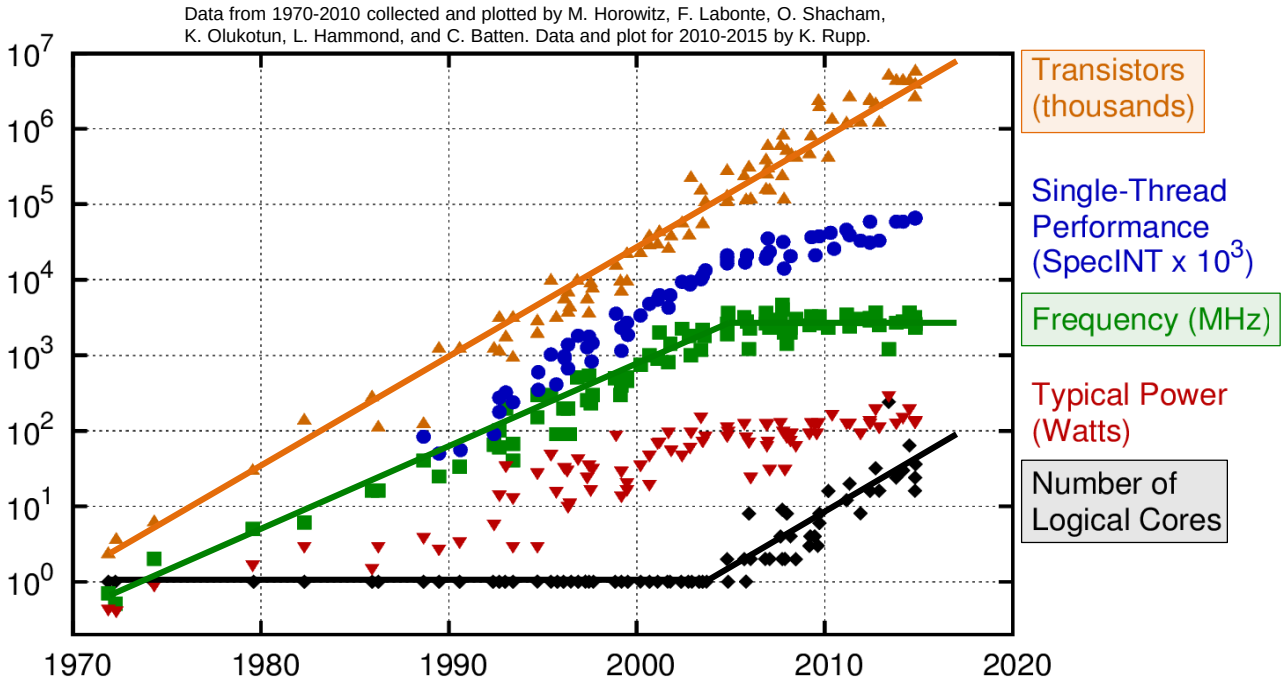


- In 2017: 9,700 downloads, 10K clones, 63 countries
- Adding GPU support

2007 Winner!

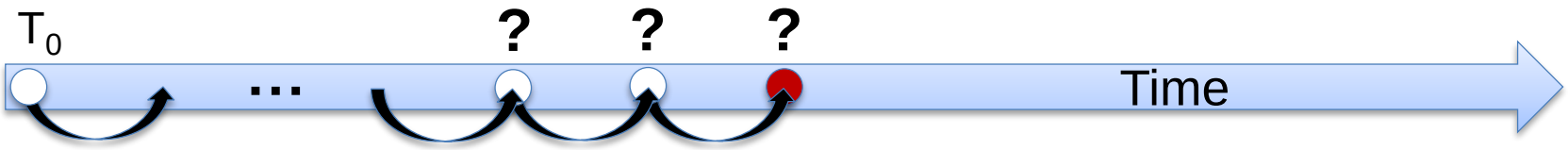
- *m* x *n* denotes *m* MPI tasks and *n* OpenMP threads per node
- Largest problem above: 72B unknowns on 1.1M cores

Parallel time integration is a major paradigm shift driven by hardware design realities



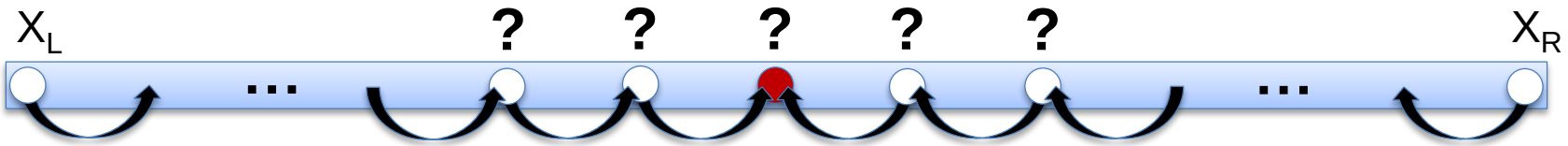
- Architecture trend: flat clock rates, more concurrency
  - Traditional time stepping is becoming a sequential bottleneck
- Continued advancement in scientific simulation will require algorithms that are parallel in time

But how is parallel-in-time even possible?



You can't solve at a given time point until you know the solution at the previous point … and you can't compute the previous point until you know the one before that … etc.

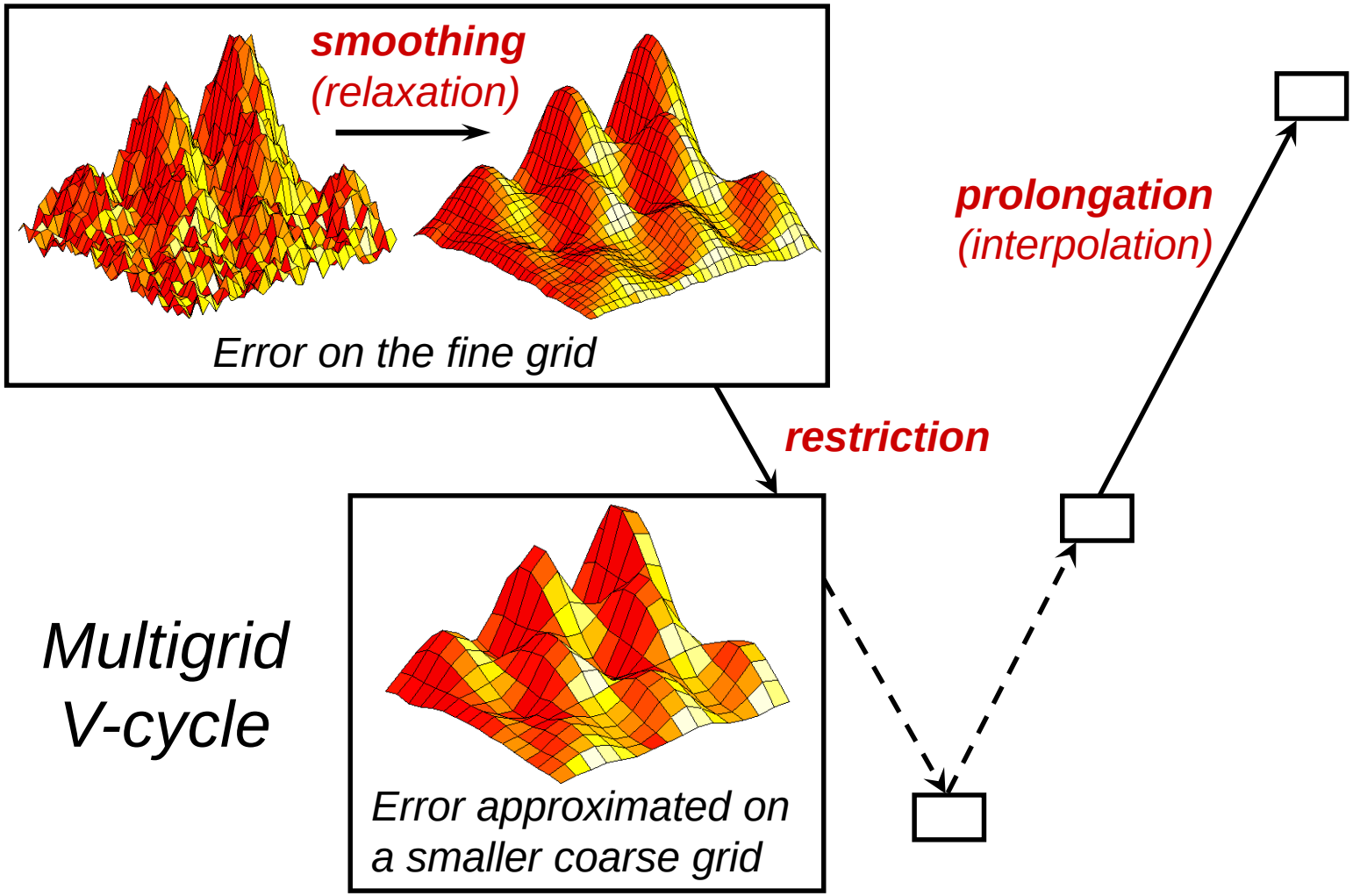
- Is this really any different from space (e.g., diffusion)?



… and the dependence is in both directions (is this harder or easier?)

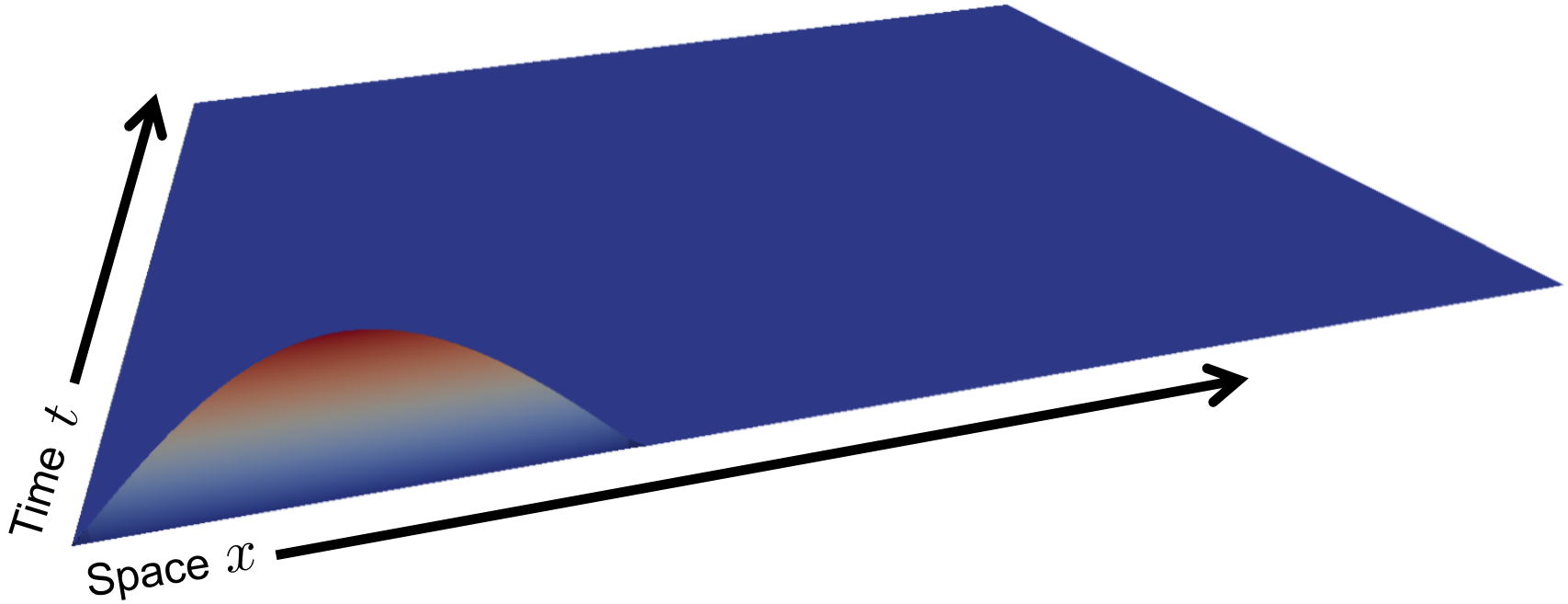
- MG is used routinely to solve the spatial problem
- Why not use it to solve the time problem too?

Our approach for parallel-in-time: leverage spatial multigrid research and experience



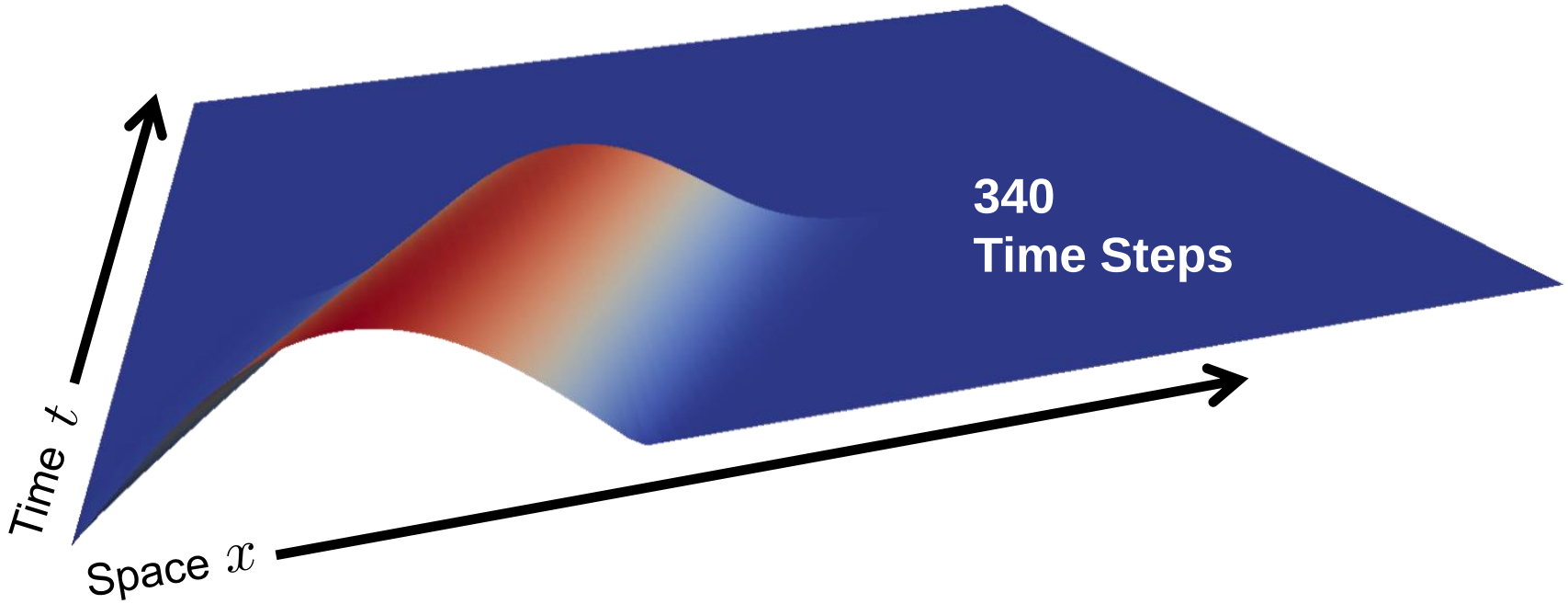
Time stepping is sequential

- Simple advection equation, $u_t = -cu_x$
- Initial condition is a wave



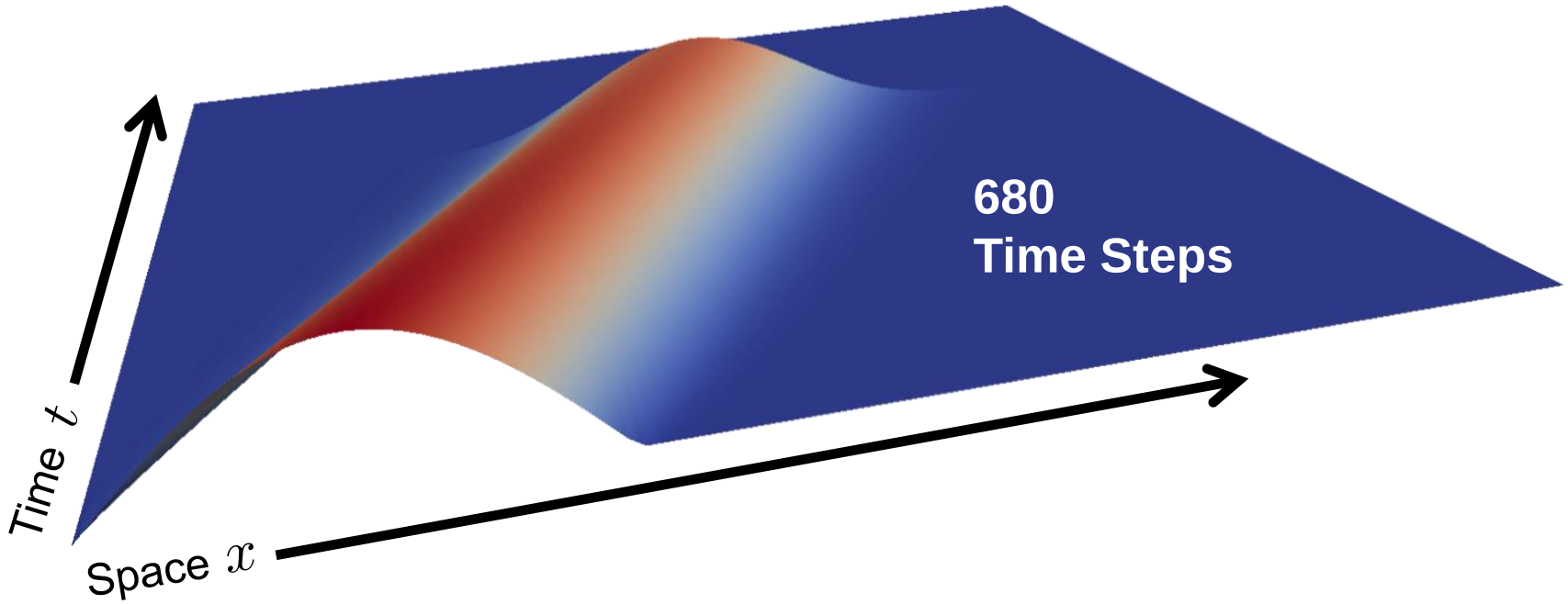
Time stepping is sequential

- Simple advection equation, $u_t = -cu_x$
- Wave propagates serially through space



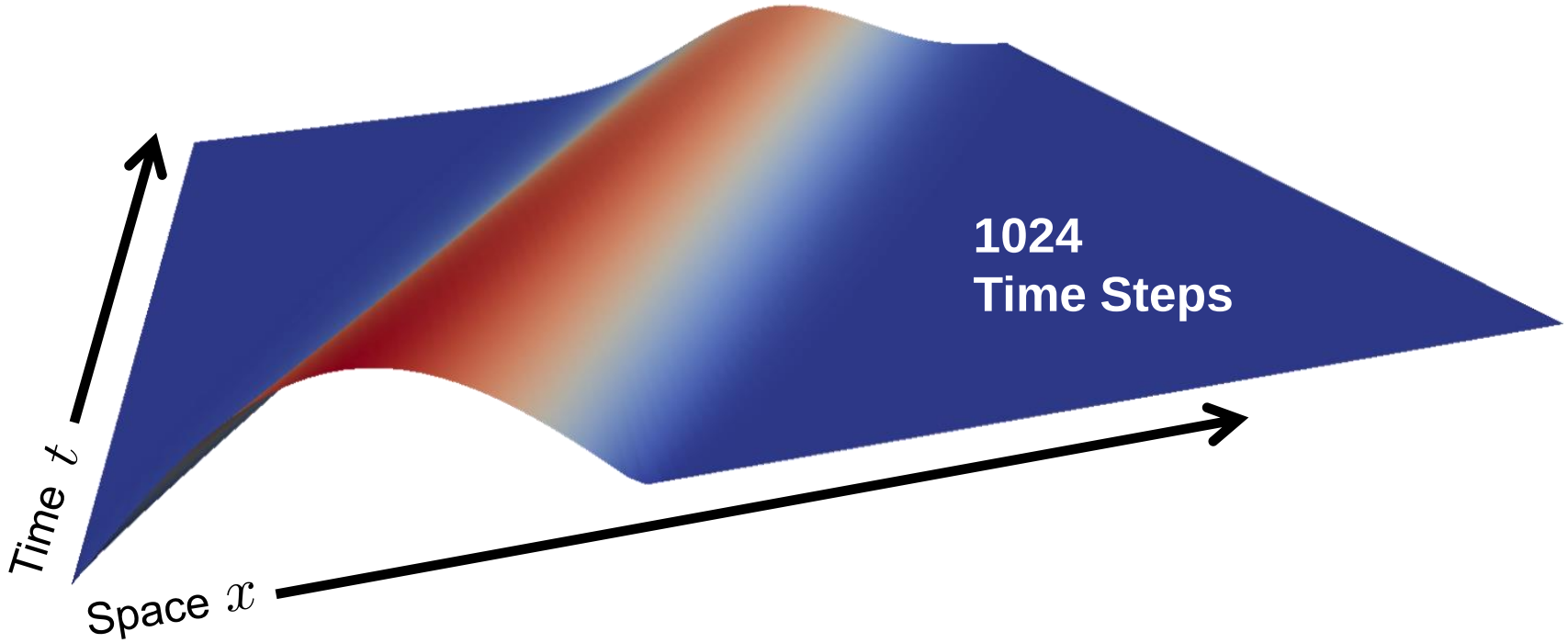
Time stepping is sequential

- Simple advection equation, $u_t = -cu_x$
- Wave propagates serially through space



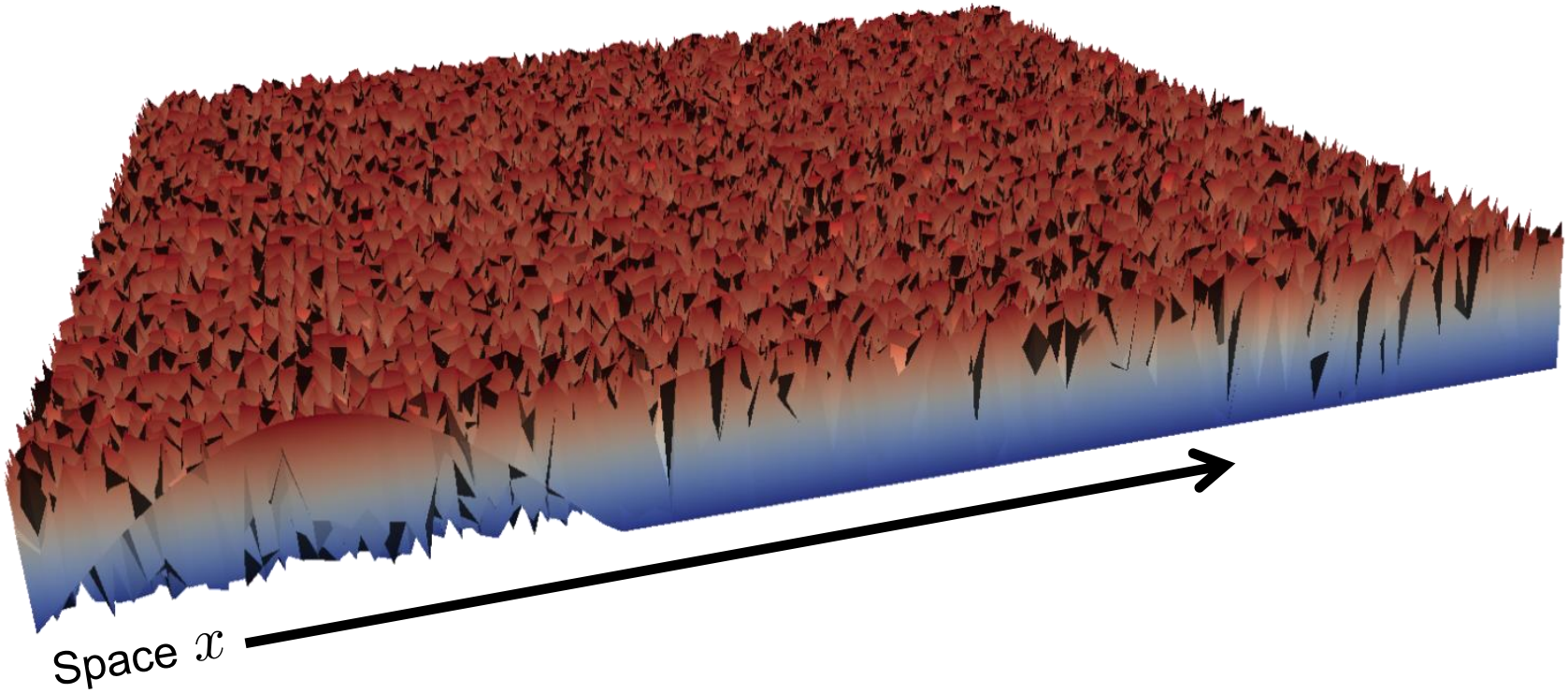
Time stepping is sequential

- Simple advection equation, $u_t = -cu_x$
- Wave propagates serially through space



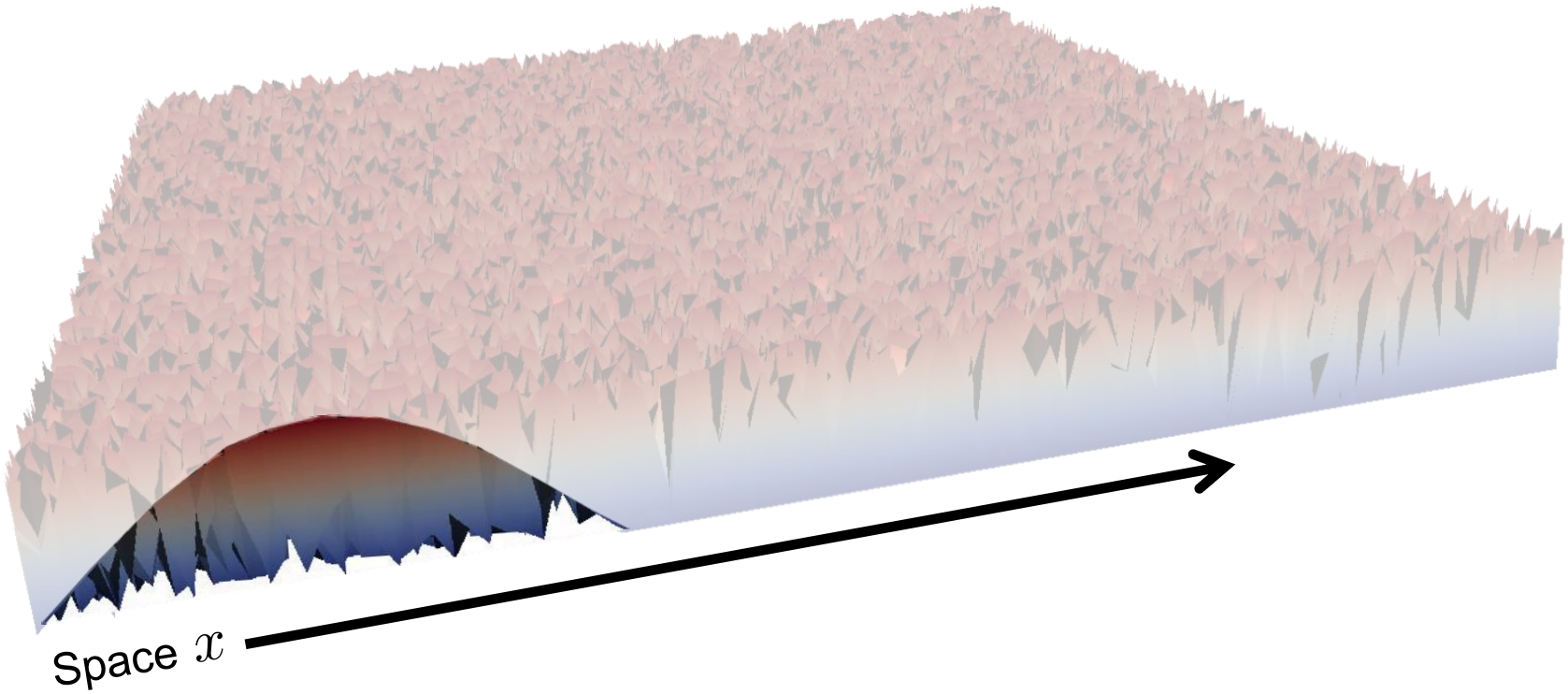
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Random initial space-time guess (only for illustration)



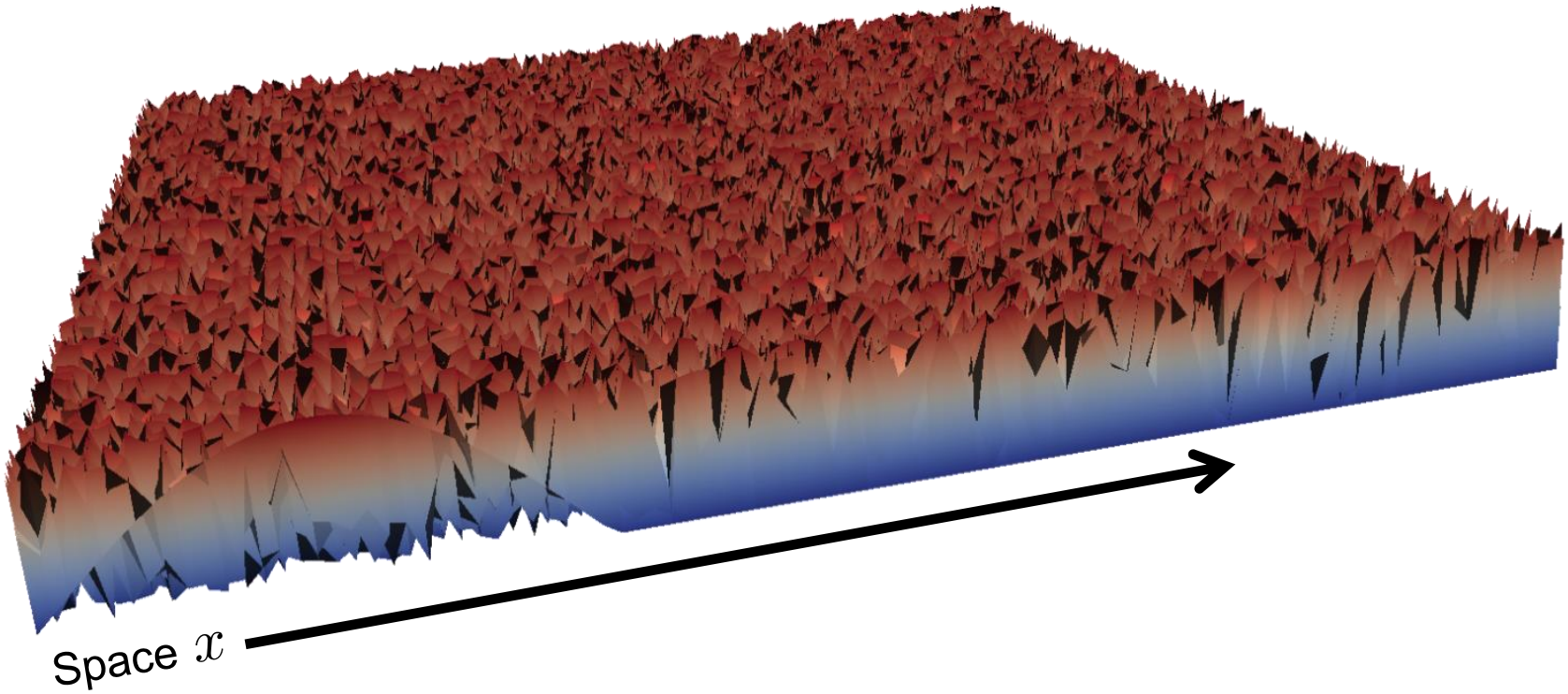
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Initial condition is a wave



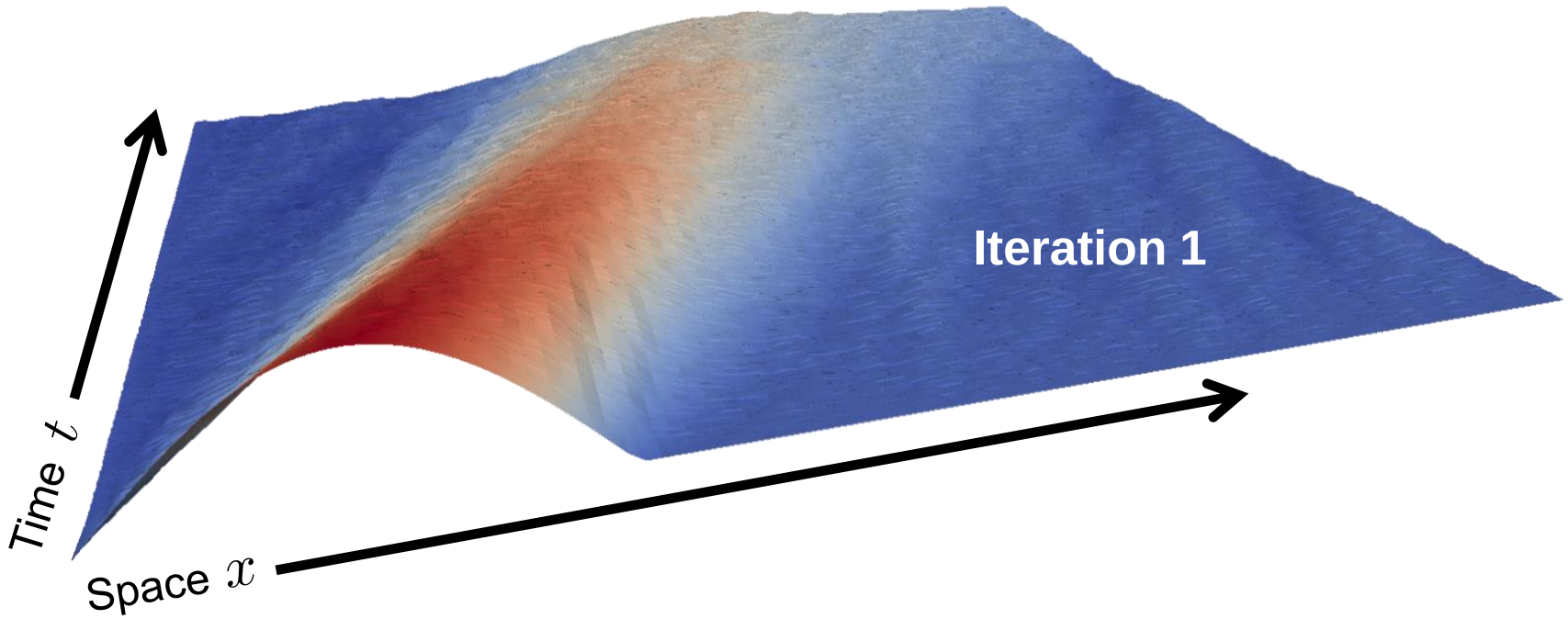
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Initial condition is a wave



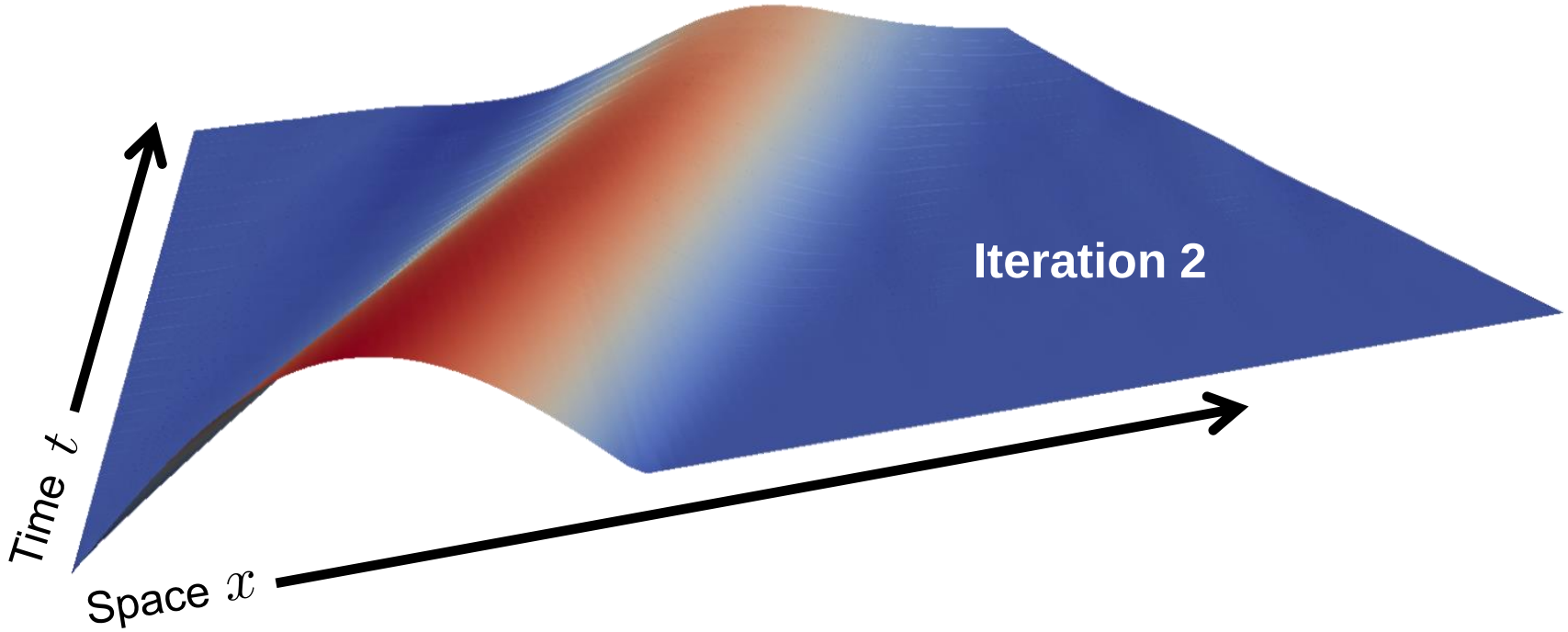
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Multilevel structure allows for fast data propagation



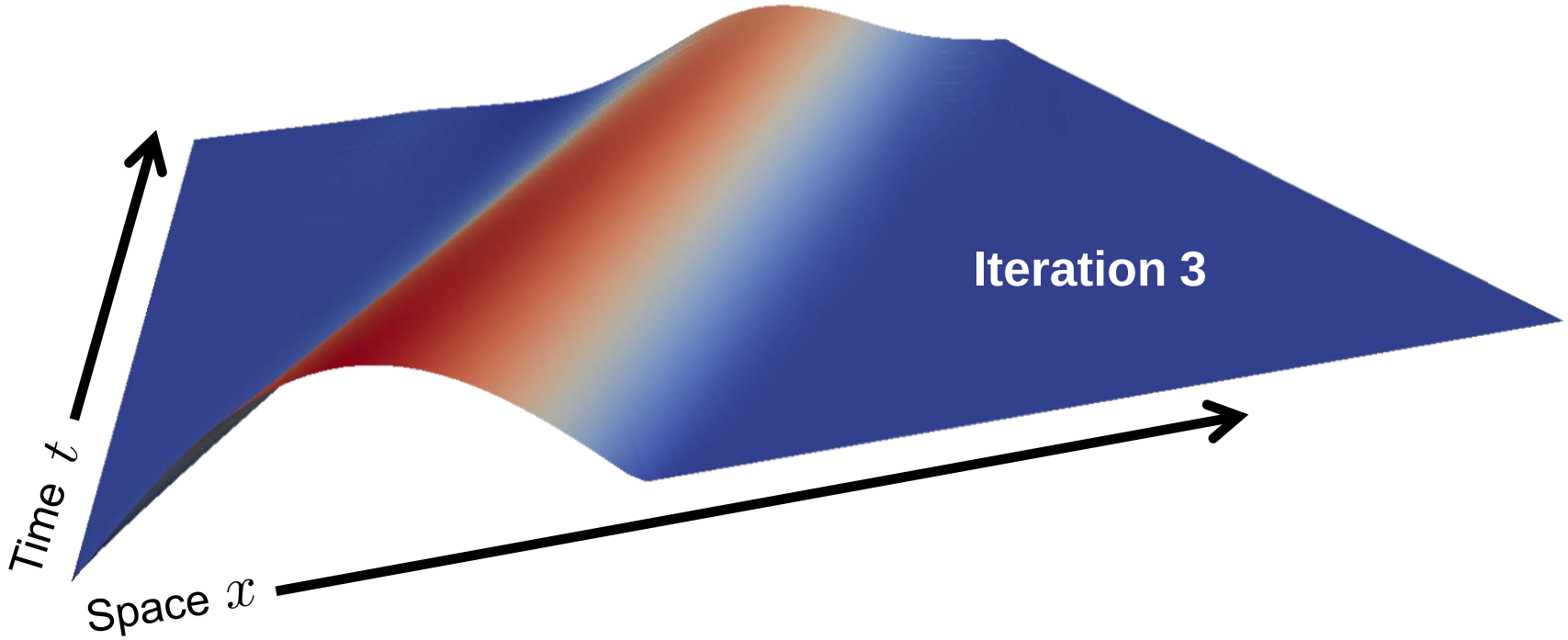
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Multilevel structure allows for fast data propagation



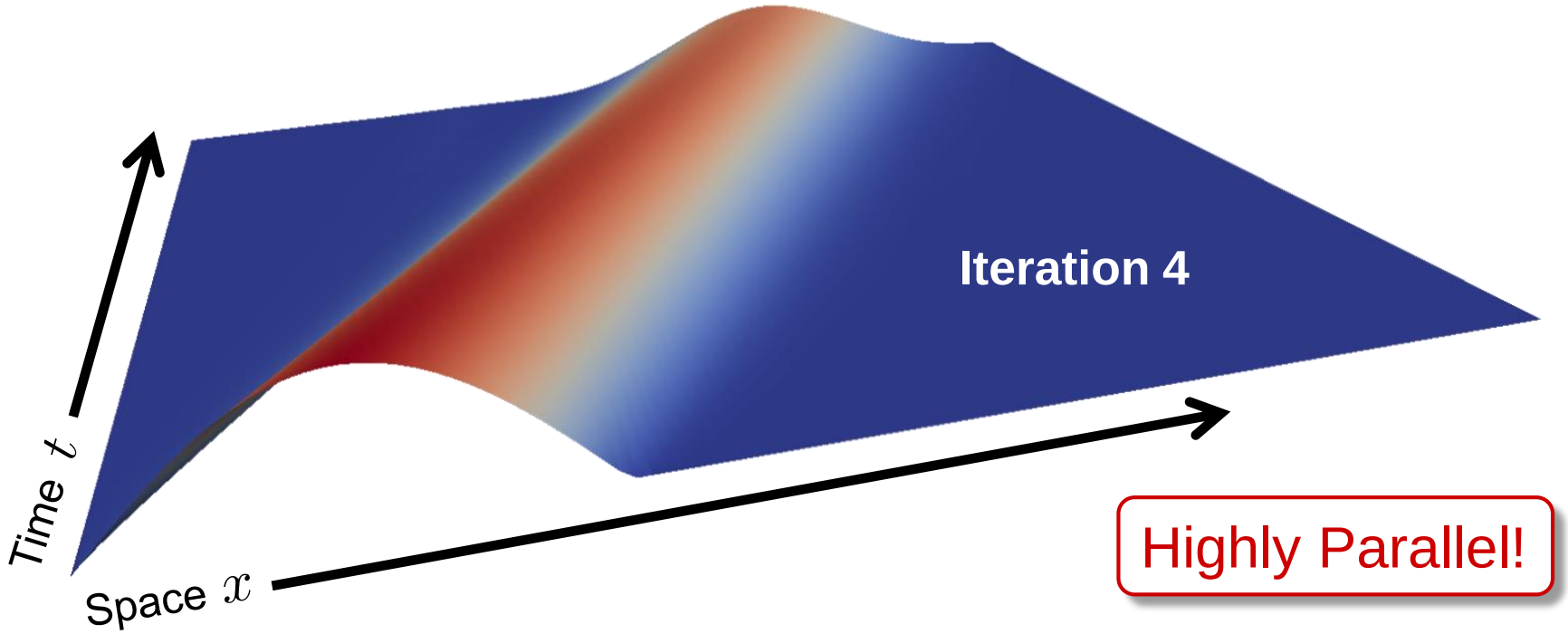
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Multilevel structure allows for fast data propagation



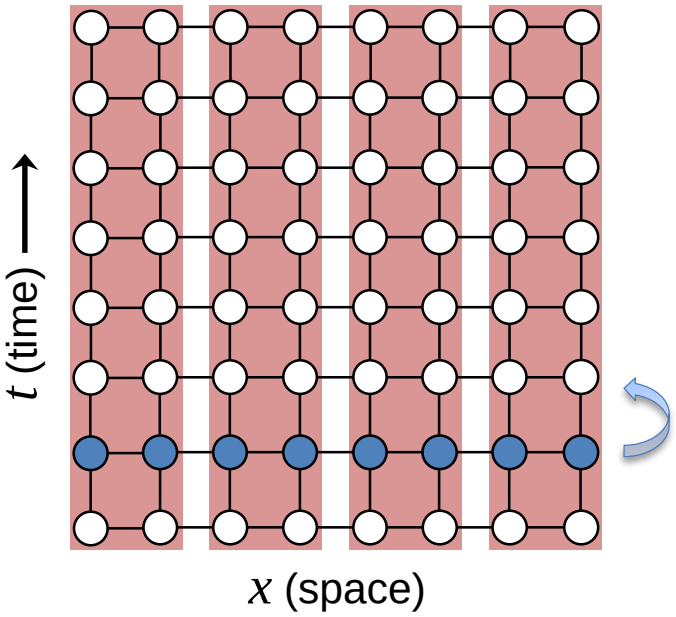
Multigrid-in-time converges to the serial space-time solution in parallel

- Simple advection equation, $u_t = -cu_x$
- Already very close to the solution

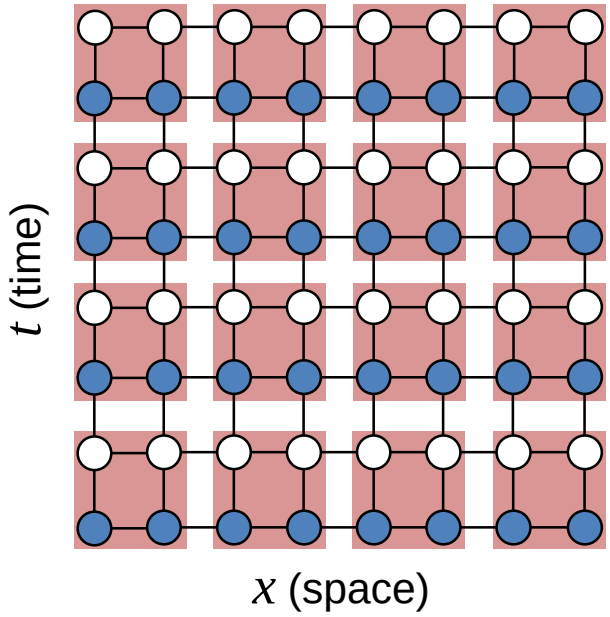


Significantly more parallel resources can be exploited with multigrid in time

Serial time stepping



Multigrid in time



- Parallelize in space only
- Store only one time step
- Parallelize in space and time
- Store several time steps

It's useful to view the time integration problem as a large block matrix system

- General one-step method

$$\boldsymbol{u}_i = \Phi_i(\boldsymbol{u}_{i-1}) + \boldsymbol{g}_i, \quad i = 1, 2, ..., N$$

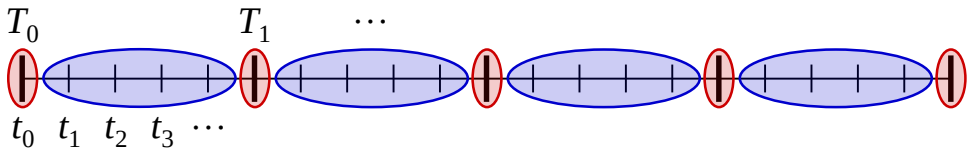
- Linear setting: time marching = block forward solve
  - $O(N)$ direct method, but sequential

$$A\mathbf{u} \equiv \begin{pmatrix} I & & & \\ -\Phi & I & & \\ & \ddots & \ddots & \\ & & -\Phi & I \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_N \end{pmatrix} = \begin{pmatrix} \boldsymbol{g}_0 \\ \boldsymbol{g}_1 \\ \vdots \\ \boldsymbol{g}_N \end{pmatrix} \equiv \mathbf{g}$$

- Our approach is based on multigrid reduction (MGR) methods (approximate cyclic reduction)
  - $O(N)$ iterative method, but highly parallel

MGR dates to 1979 (Ries & Trottenberg) and we have extended it “in time” (MGRIT)

- Partition the grid into *C*-points and *F*-points and define so-called ideal restriction and interpolation operators



$$A = \begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix}$$

$$R_F = \begin{pmatrix} -A_{cf}A_{ff}^{-1} & I_c \end{pmatrix}, \quad P_F = \begin{pmatrix} -A_{ff}^{-1}A_{fc} \\ I_c \end{pmatrix}, \quad S = \begin{pmatrix} I_f \\ 0 \end{pmatrix}$$

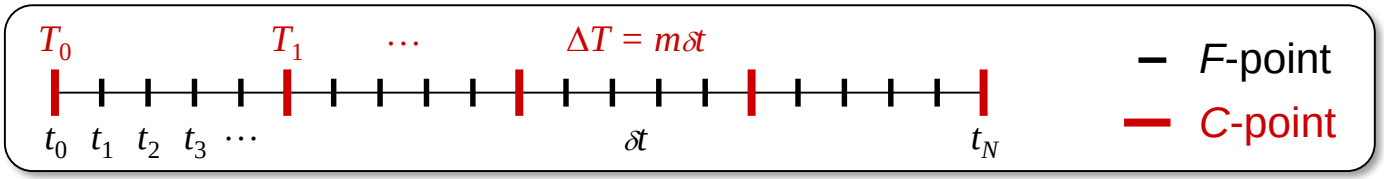
- Then, the following is a (two-level) error propagator for an exact method

$$(I - P_F(R_F A P_F)^{-1} R_F A)\ (I - S(S^T A S)^{-1} S^T A) = 0$$

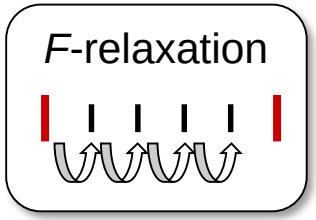
Coarse-grid correction F-relaxation

- MGR replaces the coarse operator $R_F A P_F$ and extends relaxation to recursively define an optimal multilevel method
  - Ries, Trottenberg (1979); Foerster, Stüben, Trottenberg (1981), Schröder (1954)

Applying the ideal operators in MGRIT involves applying the time integrator Φ



- Relaxation alternates between *F* / *C*-points
  - F-relaxation = integration over coarse intervals
- Coarse system is a time re-discretization
  - Replaces the exact Petrov-Galerkin system
- Non-intrusive approach
  - Time discretization is unchanged
  - User only provides time integrator Φ



Coarse Petrov-Galerkin system is not practical → approximate it

$$A_\Delta \mathbf{u}_\Delta = \mathbf{g}_\Delta \equiv R_\Phi \mathbf{g}$$

$$A_\Delta = \begin{pmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m & I \end{pmatrix}$$

Our MGRIT approach builds as much as possible on existing codes and technologies

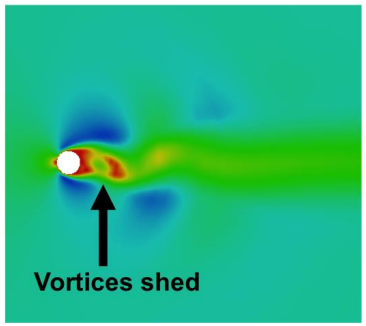
- Combines algorithm development, theory, and software proof-of-principle

- Goal: Create concurrency in the time dimension
- Non-intrusive, with unchanged time discretization
  - Implicit, explicit, multistep, multistage, …
- Converges to same solution as sequential time stepping
- Extends to nonlinear problems with FAS formulation

$$\begin{pmatrix} I & & & \\ -\Phi & I & & \\ & \ddots & \ddots & \\ & & -\Phi & I \end{pmatrix}$$

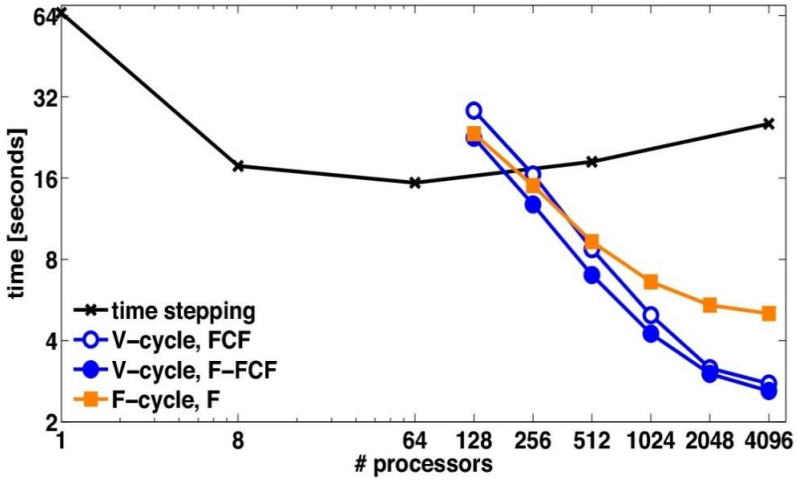
- XBraid is our open source implementation of MGRIT
  - User defines two objects and writes several wrapper routines (Step)
  - Only stores *C*-points to minimize storage

- Many active research topics, applications, and codes
  - Adaptivity in space and time, moving meshes, BDF methods, …
  - Linear/nonlinear diffusion, advection, fluids, power grid, elasticity, …
  - MFEM, hypre, Strand2D, Cart3D, LifeV, CHeart, GridDyn



Parallel speedups can be significant, but in an unconventional way

- Parallel time integration is driven entirely by hardware
  - Time stepping is already $O(N)$
- Useful only beyond some scale
  - There is a crossover point
  - Sometimes need significantly more parallelism just to break even
  - Achievable efficiency is determined by the space-time discretization and degree of intrusiveness



3D Heat Equation: 33^3 x 4097,
8 procs in space, 6x speedup

- The more time steps, the more speedup potential
  - Applications that require lots of time steps benefit first
  - Speedups (so far) up to 49x on 100K cores

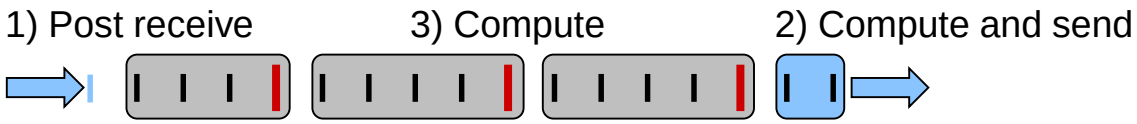
XBraid is open source and designed to be both non-intrusive and flexible

- User defines two objects:
  - *App* and *Vector*

- User also writes several wrapper routines:
  - *Step*, *Init*, *Clone*, *Sum*, *SpatialNorm*, *Access*, *BufPack*, *BufUnpack*
  - *Coarsen*, *Refine* (optional, for spatial coarsening)

- Example: *Step(app, u, status)*
  - Advances vector *u* from time *tstart* to *tstop* and returns a target refinement factor

- Code stores only C-points to minimize storage
  - Ability to coarsen by large factors means fewer parallel resources
  - Memory multiplier per processor: ~O(log N) with time coarsening, O(1) with space-time coarsening
  - Each proc starts with the right-most interval to overlap comm/comp



Experiments coupling XBraid with various application research codes

- Navier-Stokes (compressible and incompressible)
  - Strand2D, CarT3D, LifeV (Trilinos-based)
- Heat equation (including moving mesh example)
  - MFEM, hypre
- Nonlinear diffusion, the p-Laplacian
  - MFEM
- Power-grid simulations
  - GridDyn
- Explicit time-stepping coupled with space-time coarsening
  - Heat equation
  - Advection plus artificial dissipation
  - MFEM, hypre

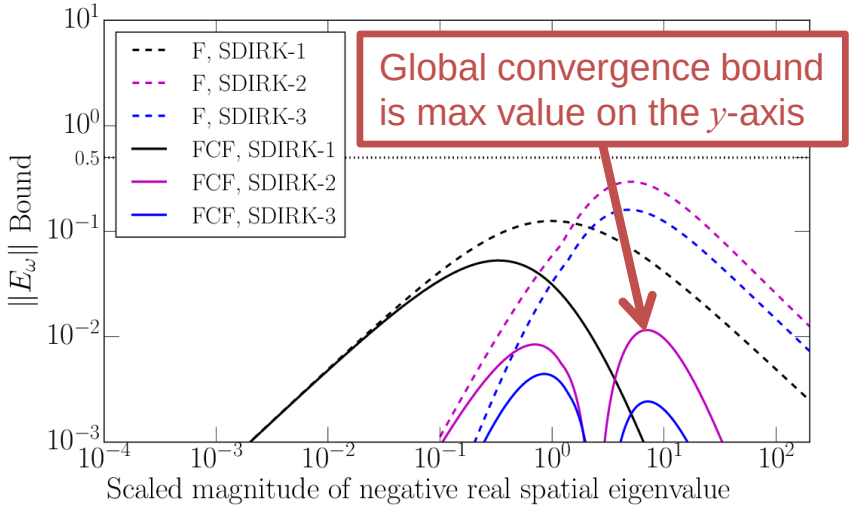
Some Progress and Current Research Directions (a quick preview of Thursday)

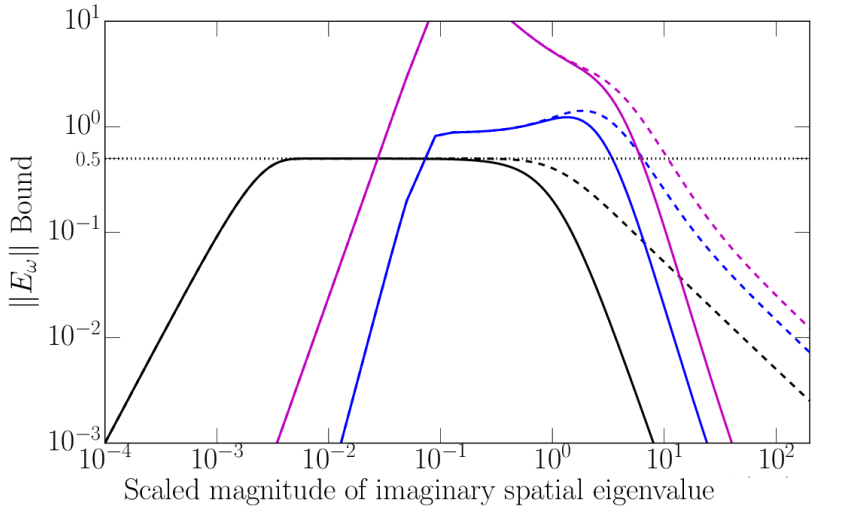
We developed a linear two-grid convergence theory to guide MGRIT algorithm development

- Assume Φ and Φ_Δ are simultaneously diagonalizable with eigenvalues $\lambda_\omega, \mu_\omega$
- Sharp bound for error propagator

$$\|E\| \leq \max_\omega |\lambda_\omega^m - \mu_\omega| \frac{1 - |\mu_\omega|^{N_T - 1}}{1 - |\mu_\omega|} |\lambda_\omega|^m$$

- Agnostic to space-time discretization
  - But discretization affects convergence
- Eigenvalues (representative equation):
  - Real (parabolic)
  - Imaginary (hyperbolic without dissipation)
  - Complex (hyperbolic with dissipation)
- Insights:
  - FCF significantly faster
  - High order can be faster or slower
  - Artificial dissipation helps a lot
  - Small coarsening factors sometimes needed

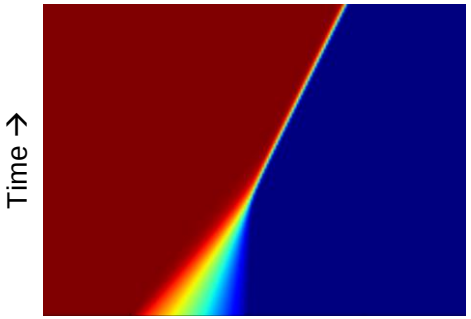




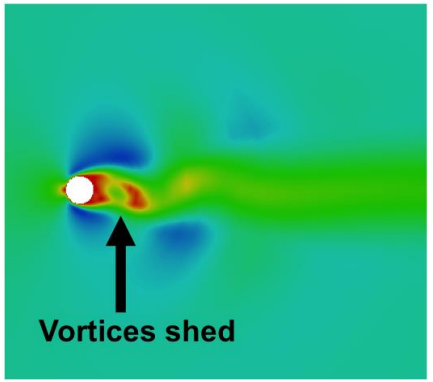
Hyperbolic problems are a major new emphasis for our MGRIT algorithm research

- We have already had some initial success…
- 1D/2D advection and Burgers' equation
  - F-cycles needed (multilevel), slow growth in iterations
  - Requires adaptive spatial coarsening
  - Dissipation improves convergence
  - Mainly SDIRK-k schemes to date (implicit & explicit)
- Combination of FCF relaxation, F-cycles, and small coarsening factors improves robustness
  - Confirmed by theory
- Navier-Stokes in 2D and 3D
  - Multiple codes: Strand2D, Cart3D, LifeV, CHeart
  - Compressible and incompressible
  - Modest Reynolds numbers (100 – 1500)

1D Inviscid Burgers

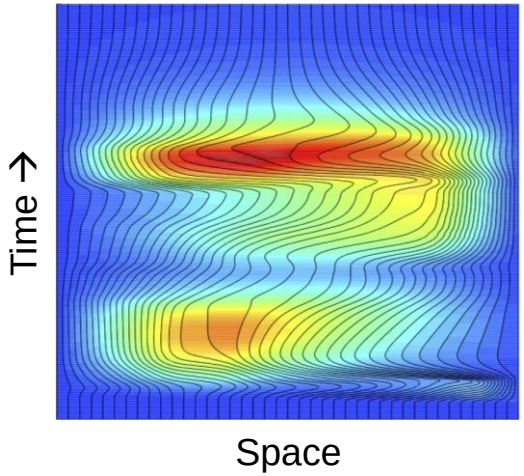


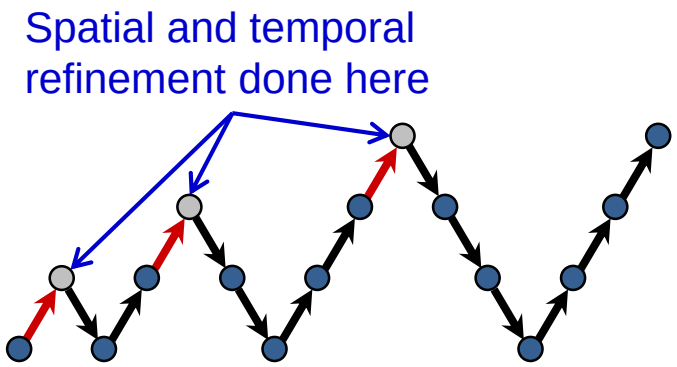
Navier-Stokes
7.5x speedup in Strand2D



Adaptivity is an important feature of many codes and we have begun to develop support for it in XBraid

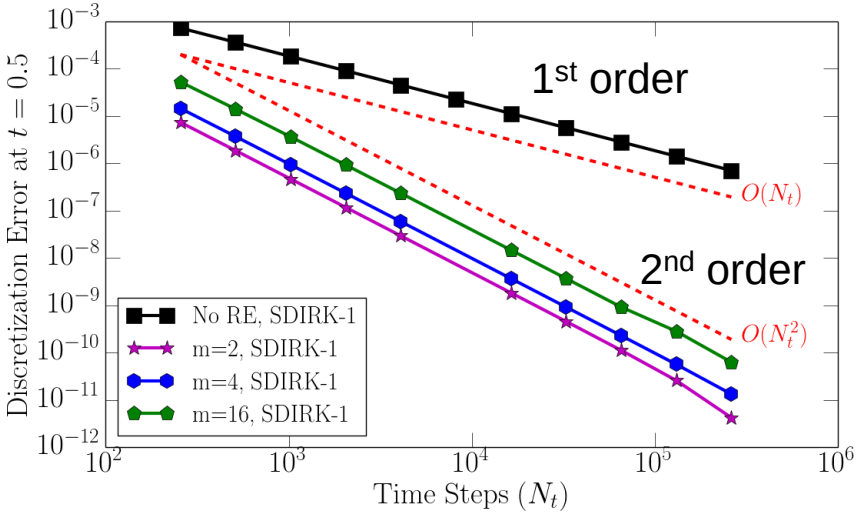
- Moving spatial mesh
  - 1D diffusion with time dependent source
  - Unsteady flow around moving cylinder
- Temporal refinement via Full Multigrid (FMG)
  - ODE simulation of satellite orbit
  - DAE power grid simulations in GridDyn (25x speedup)
- Temporal and spatial refinement
  - 2D heat equation with FOSLS (6x speedup)
- Initial emphasis is algorithm development
  - Demonstrating parallel speedup is the eventual goal



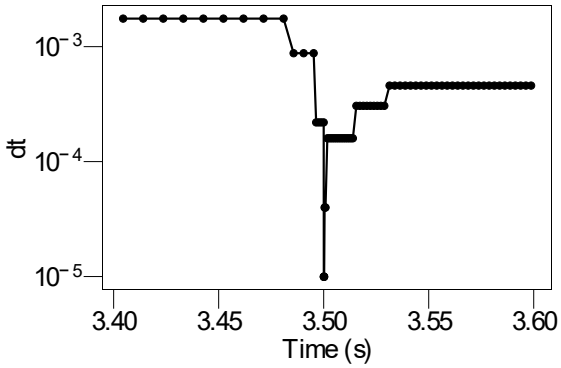


Other developments and research directions

- Higher order with Richardson Extrapolation MGRIT at no cost
- Adjoint-based MGRIT solver for design optimization
- Showed potential for speeding up neural network training
- Power grid simulation with discontinuities and adaptivity
  - WECC 179 bus system
  - 12x to 53x speedup
  - Investigating approaches for unscheduled discontinuities



Adaptive time grid around load discontinuity



Some Historical Background and Connections

Nearly 50 years of research exists but has only scratched the surface

- Earliest work goes back to 1964 by Nievergelt
  - Led to multiple shooting methods, Keller (1968)
- Space-time multigrid methods for parabolic problems
  - Hackbusch (1984); Horton (1992); Horton and Vandewalle (1995)
  - The latter is one of the first optimal & fully parallelizable methods to date
- Parareal was introduced by Lions, Maday, and Turinici in 2001
  - Probably the most widely studied method
  - Gander and Vandewalle (2007) show that parareal is a two-level FAS multigrid method
- Discretization specific work includes
  - Minion, Williams (2008, 2010) – PFASST, spectral deferred correction, FAS
  - DeSterck, Manteuffel, McCormick, Olson (2004, 2006) – FOSLS
- Research on these methods continues to ramp up!
  - Ruprecht, Krause, Speck, Emmett, Langer, … this is not an exhaustive list
- Recent review: Gander (2015), “50 years of time parallel time integration”

Parareal is an important PinT algorithm with a connection to MGRIT

- Basic parallelization algorithm
  - F is called the fine propagator
  - F_Δ is the coarse propagator

Initialize $\mathbf{u}_\Delta^0$	(1)	
For $k = 0, 1, ...$		
Distribute $\mathbf{u}_\Delta^k$ to processors $\{j\}$	(2)	
Compute $F^m \boldsymbol{u}_{\Delta,j}^k$ in parallel	(3)	← parallel (fine)
Aggregate these results to a single processor	(4)	
Compute $\mathbf{u}_\Delta^{k+1}$ in serial using values from (3)	(5)	← serial (coarse)

- Update step (5) is usually written in the literature as

$$\boldsymbol{u}_{\Delta,j+1}^{k+1} = F_\Delta \boldsymbol{u}_{\Delta,j}^{k+1} + F^m \boldsymbol{u}_{\Delta,j}^k - F_\Delta \boldsymbol{u}_{\Delta,j}^k, \quad j = 1, 2, ...$$

Although they are related, MGRIT is not Parareal

- Parareal is an important PinT algorithm and Parareal researchers continue to make significant contributions to PinT research and development
- Two-level MGRIT with F-relaxation is a Parareal method (Gander/Vandewalle, SISC, 2007)
- However, conclusions about Parareal do not automatically apply to MGRIT
- Parareal is a two-level method, MGRIT is multilevel
  - Two-level limits scalability
- Parareal convergence results often assume an exact fine-scale propagator
  - From an MGRIT perspective, this is an infinite coarsening factor
  - Hyperbolic problems in particular are much harder to handle in this setting
- Parareal is often viewed as an approach for discretizing in time
  - Parallel predictor/corrector scheme, stability analysis, …
  - MGRIT is primarily viewed as a parallel-in-time solver

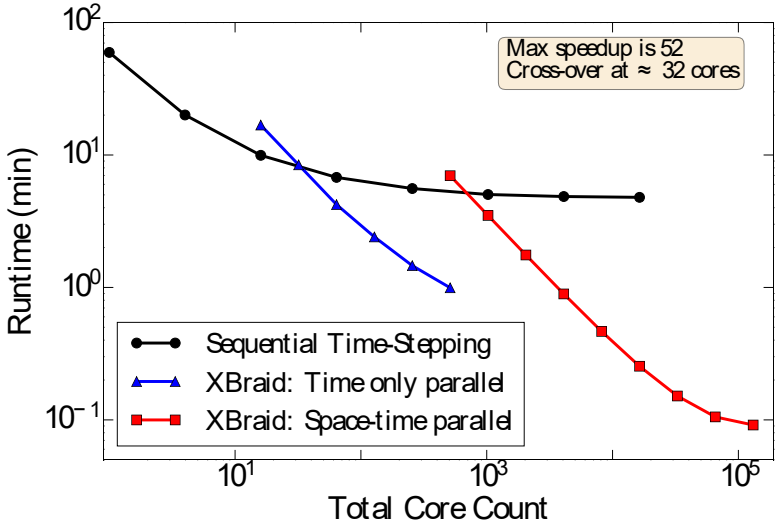
An Approaching Paradigm Shift for Scientific Computing

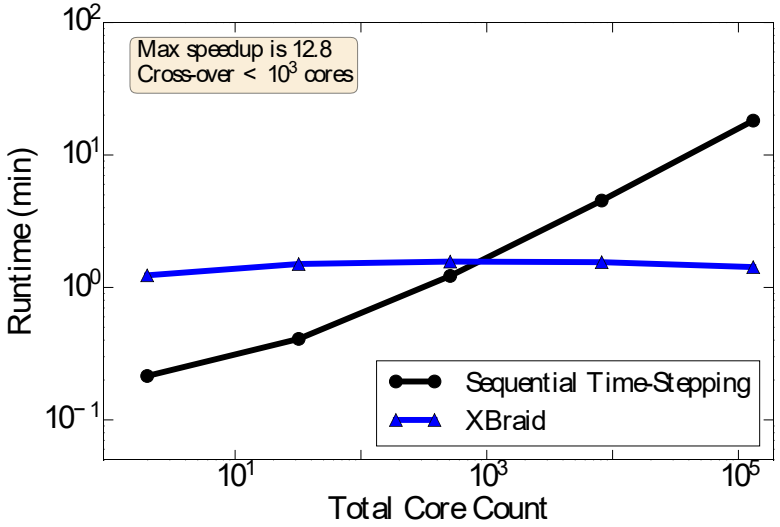
There continues to be skepticism about the idea of parallel in time

- What about causality?
  - PinT algorithms solve the same space-time system as time stepping
- What about shocks and discontinuities?
  - Coarse temporal grids do not (by definition) and need not (by demonstration) capture fine-scale features
- PinT requires too much memory
  - Eventually there will be ample resources (this is reminiscent of the transition from 2D-space to 3D-space in the 1990s)
- What about hyperbolic problems?
  - This is indeed hard and requires more research (have made progress)
- PinT algorithms have terrible parallel efficiencies
  - Parallel efficiencies for sequential time stepping are much worse

In both strong and weak scaling settings, parallel in time eventually outperforms time stepping

- Results for 2D heat equation





Strong scaling: $257^2 \times 16{,}384$ grid, max speedup is 52x

Weak scaling: grid sizes range from $129^2 \times 512$ to $2049^2 \times 131{,}072$, max speedup is 12.8x

A "proof" that science simulation codes will eventually need to use parallel in time (PinT)

Assumptions (accept these for the moment):

1. Scientists will continue to want higher fidelity simulations, requiring an increasing temporal dimension (unboundedly)

   $$n \xrightarrow[time \to \infty]{} \infty$$

2. Simulation time for time stepping is linear in n (T_0 is a fixed atomic time):

   $$T_{seq} \gtrsim nT_0$$

3. Simulation time for PinT is polylogarithmic in n:

   $$T_{pint} \lesssim \log^p(n)\, T_0$$

4. Scientists have a threshold beyond which they will switch to a faster method:

   If $(T_{seq} > T_{thresh})$ and $(T_{seq} > T_{pint})$, switch to PinT

Result:

$$n > n_0 \implies T_{seq} \gtrsim nT_0 > \log^p(n)\, T_0 \gtrsim T_{pint} \implies 4$$

Comments on assumptions 1 and 2

1. Scientists will continue to want higher fidelity simulations, requiring an increasing temporal dimension (unboundedly)

$$n \xrightarrow[time \to \infty]{} \infty$$

- Think of n as being a function of available memory
  - This one is not hard to argue with
  - Can increase fidelity by adding new features such as UQ – delays growth of n

2. Simulation time for time stepping is linear in n (T_0 is a fixed atomic time):

$$T_{seq} \gtrsim nT_0$$

- Note that if $T_0 \sim 1/n$ (e.g., due to clock speed increases), the simulation time threshold could be avoided
  - This is what happened throughout the 1990's and into the 2000's
- Note also that the result ($T_{seq} > T_{pint}$) is true even if T_0 is not fixed

Comments on assumption 3

3. Simulation time for PinT is polylogarithmic in n:

$$T_{pint} \lesssim \log^p(n)\, T_0$$

- For multigrid (solving to discretization accuracy), $p = 2$
  - FMG cycles: fixed number of cycles with $\log^2 n$ communication
  - V cycles: $\log n$ cycles with $\log n$ communication
- Example: d spatial dimensions with $\Delta t \sim \Delta x$
  - $T_{pint} \lesssim \log^2\left(n^{d/(d+1)}\right) T_0 = \left(\frac{d}{d+1}\right) \log^2(n)\, T_0$
  - Temporal and spatial dimensions are smaller due to available memory
  - Similar result with smaller constant when $\Delta t \sim \Delta x^2$
- Can show that speedup $\left(T_{seq}/T_{pint}\right)$ increases with n
  - This result does not require a fixed T_0
- Major research issue – developing multigrid methods that have the right convergence behavior

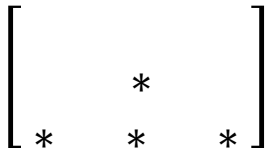
Comments on assumption 4

4. Scientists have a threshold beyond which they will switch methods:
   If $(T_{seq} > T_{thresh})$ and $(T_{seq} > T_{pint})$, switch to PinT

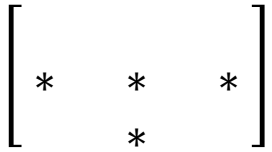
- Scientists have different thresholds
  - Importance of simulation time vs simulation accuracy varies
- Some applications will need PinT sooner than others
  - Different constants in the models yield different crossover values of n
- Assumes that memory and parallelism continue to grow
  - This is the expected trend for the foreseeable future
  - Even quantum computing is all about providing extreme concurrency
- Result does not require fixed T_0
  - Even if T_0 decreases, eventually we have $T_{seq} \gg T_{pint}$
  - Increased parallelism is the real driver (not fixed clock speeds)

A Paradigm Shift

- Need to worry about temporal solver convergence
  - Time stepping is a direct solve of a temporal discretization method (traditionally no distinction between the two)
- Choice of explicit vs implicit method may change
  - Cost of parallel time integration is the same for both
- Computational steering changes
  - Intervention at coarse temporal scales across large timelines
- Full space-time adaptivity becomes commonplace
  - Use coarse time "steps" in coarse spatial regions
- Unstructured space-time grids
  - Method of lines not needed



Explicit Stencil



Implicit Stencil

Summary and Conclusions

- Parallel time integration is needed on future architectures
  - Major paradigm shift for computational science!
- MGRIT algorithm extends multigrid reduction "in time"
  - Non-intrusive yet flexible approach (open-source code XBraid)
- MGRIT approach is showing promise in a variety of settings
  - Adaptivity in space and time, moving meshes, BDF methods, …
  - Linear/nonlinear diffusion, advection, fluids, power grid, elasticity, …
  - Coupling to codes: MFEM, hypre, Strand2D, Cart3D, LifeV, CHeart, GridDyn
- There is much future work to be done!
  - More problem types, more complicated discretizations, performance improvements, adaptive meshing, ...

CASC

Center for Applied
Scientific Computing

Thank You!

Lawrence Livermore
National Laboratory