Multigrid Solvers in Space and Time for Highly Concurrent Architectures

Future CFD Technologies Workshop, Kissimmee, Florida

Robert D. Falgout
Center for Applied Scientific Computing

January 6-7, 2018

LLNL-PRES-744321
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

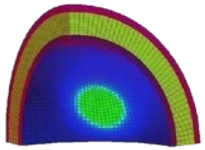
Lawrence Livermore National Laboratory

Outline

- Multigrid (in space)
  - Motivation and background
  - Geometric multigrid, algebraic multigrid (AMG), parallel multigrid
  - Example of current research
- Multigrid in time
  - Motivation and basic approach
  - MGRIT – multigrid reduction (MGR) in time
  - Some progress and current research
- Summary and conclusions

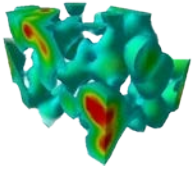
Multigrid will play an important role for addressing exascale challenges

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



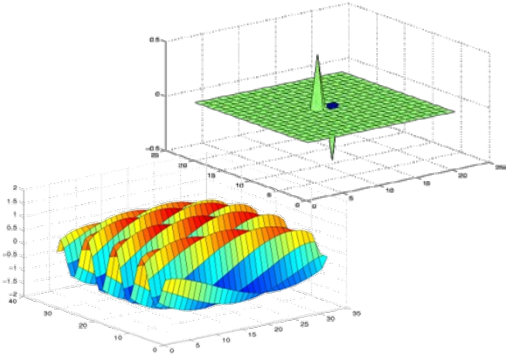
Elasticity / Plasticity

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



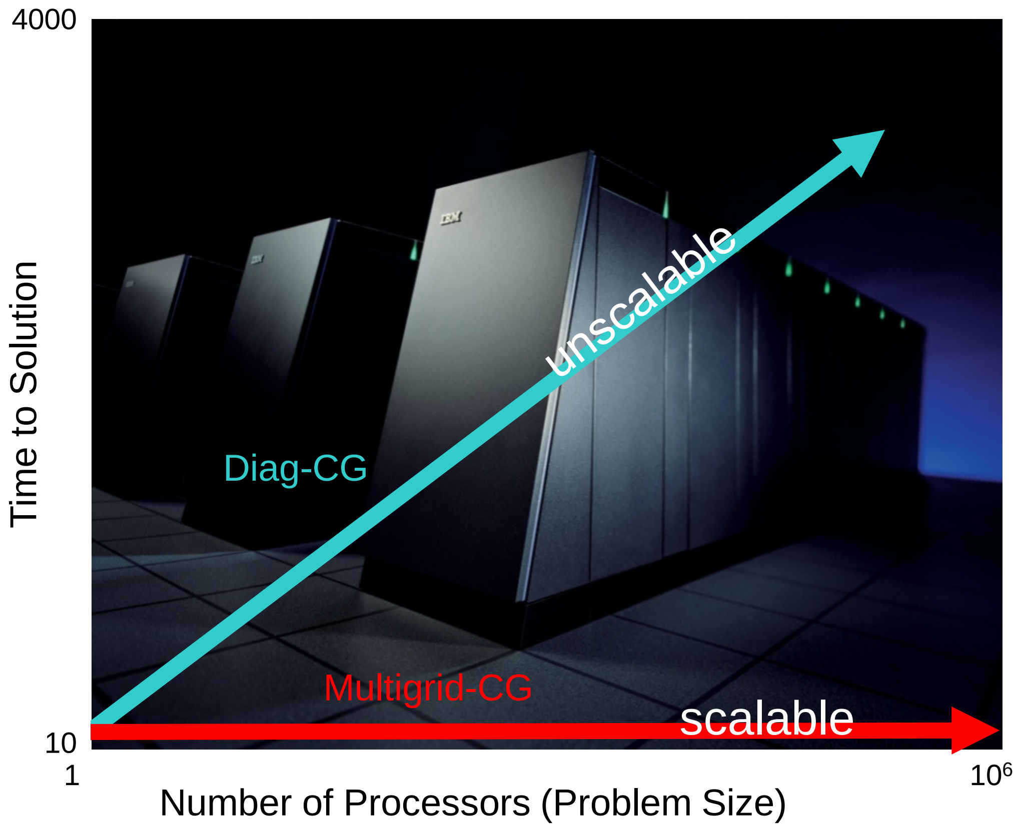
Quantum Chromodynamics

- 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



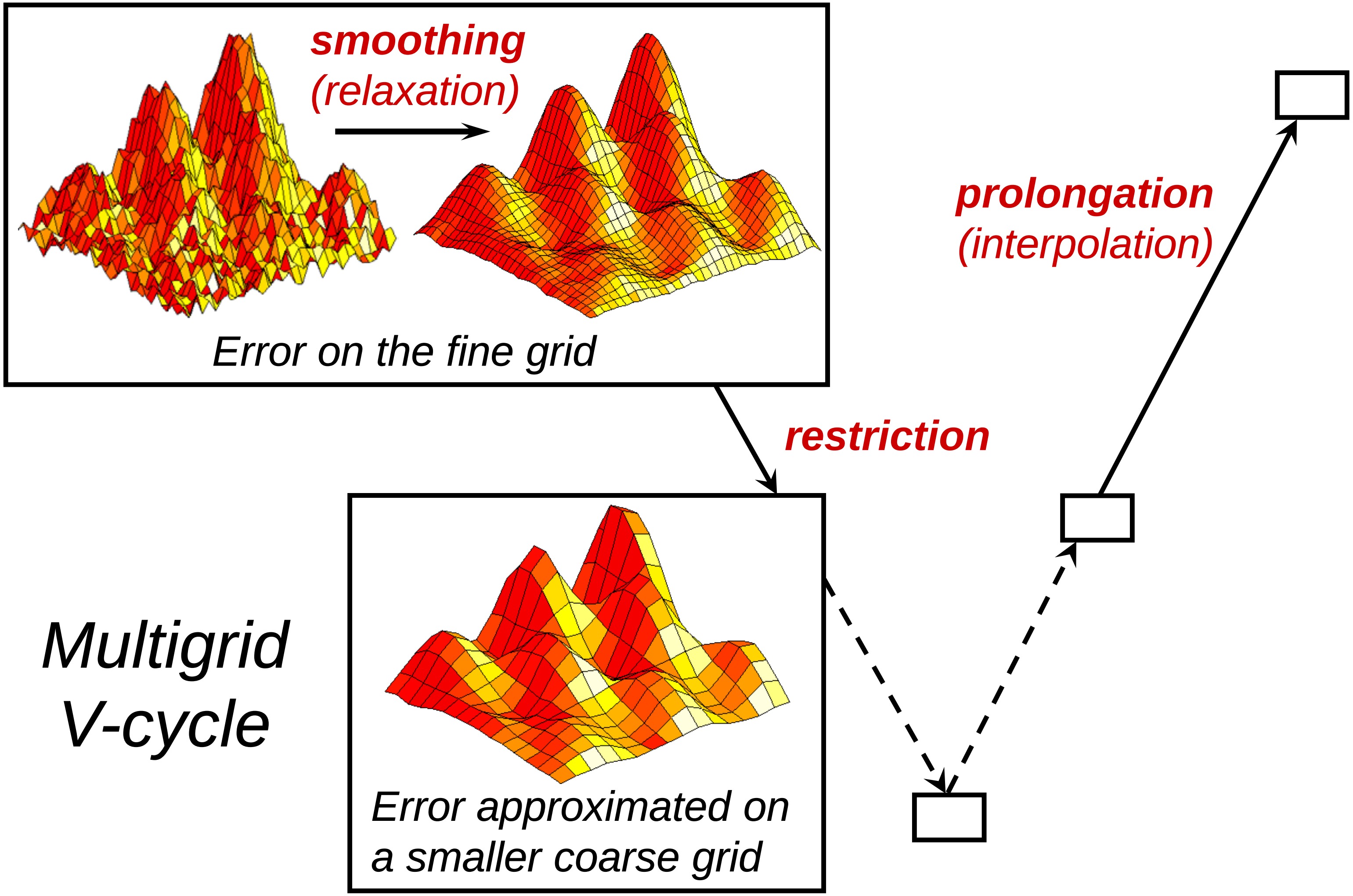
Helmholtz Modes

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



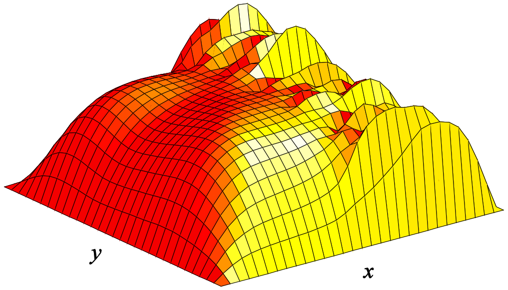
- Weak scaling – want constant solution time as problem size grows in proportion to the number of processors

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



Algebraic Multigrid (AMG) is based on MG principles, but only uses matrix coefficients

- Many algorithms (AMG alphabet soup)
- Automatically coarsens “grids”
- Error left by pointwise relaxation is called algebraically smooth error
  - Not always geometrically smooth



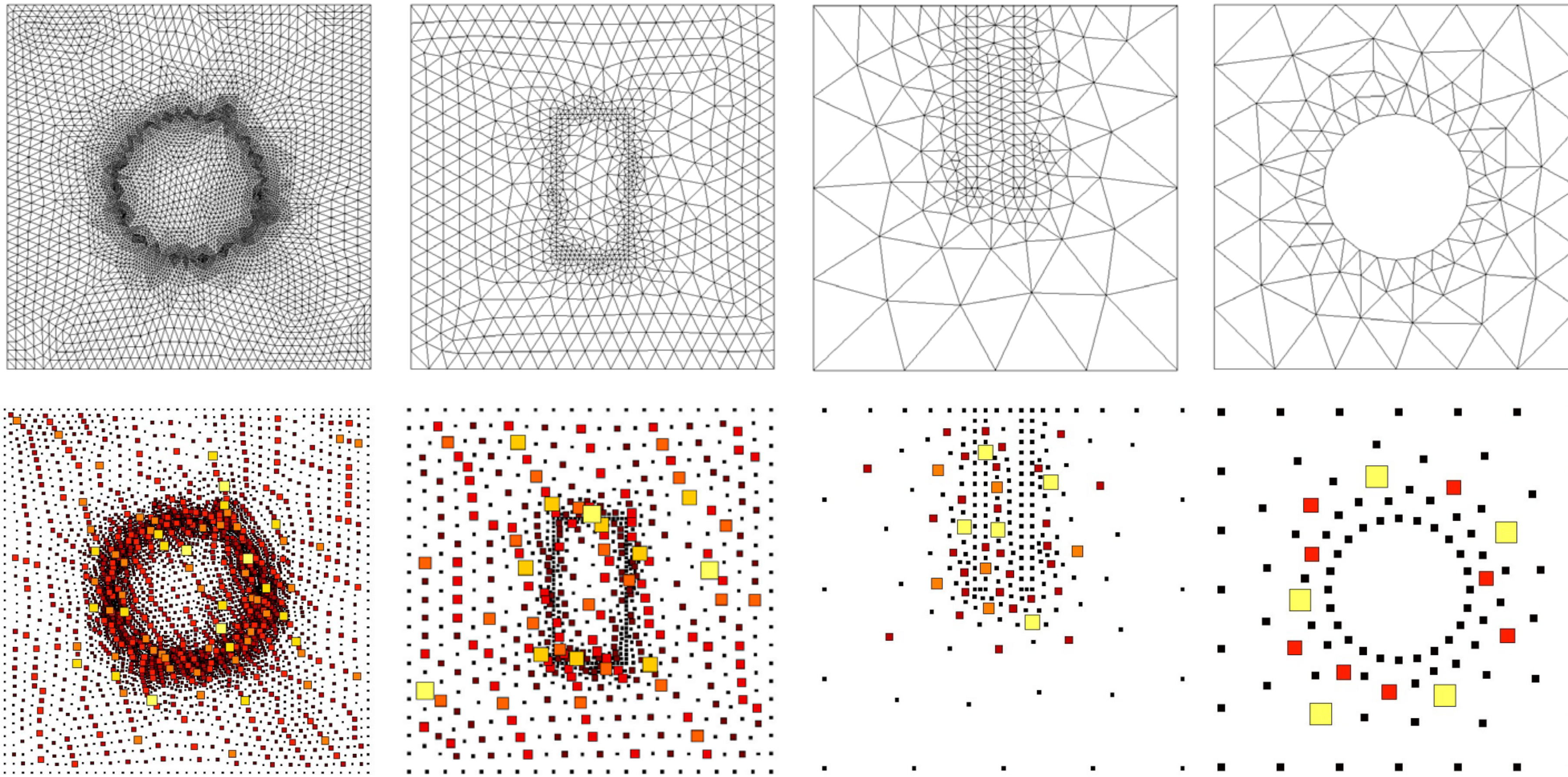
- Weak approximation property: interpolation must interpolate small eigenmodes well

$$\|E_{TG}\|_A^2 \leq 1 - \frac{1}{K}; \quad K = \sup_e \|A\| \frac{\|(I - P[0 \;\; I])e\|_2}{\|e\|_A^2}$$

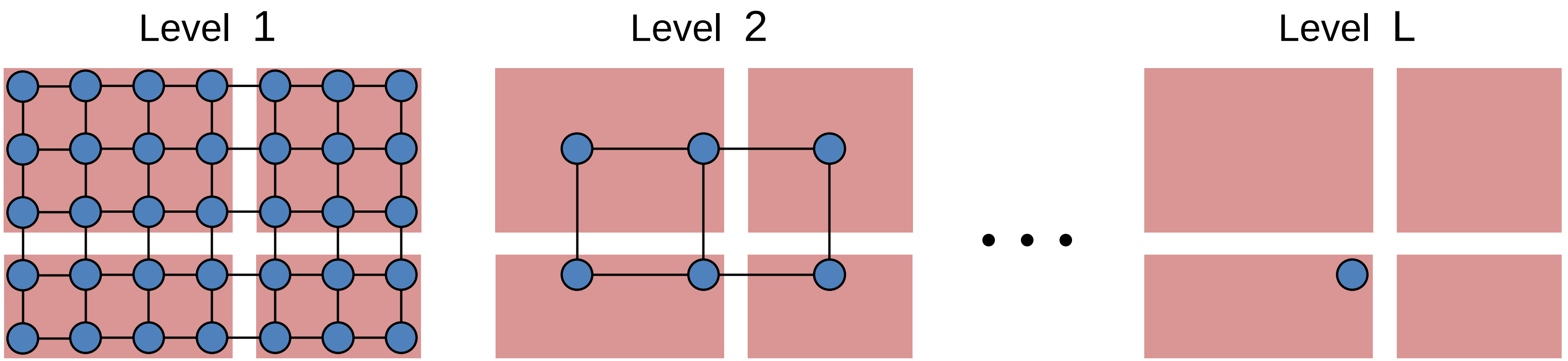
- Near null-space is important!

AMG grid hierarchies for several 2D problems

domain1 - 30°　　domain2 - 30°　　pile　　square-hole



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

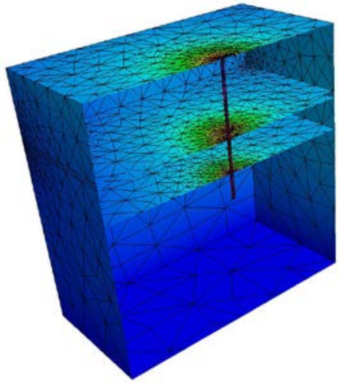


- ~ 1.5 million idle cores on Sequoia!
- Multigrid has a high degree of concurrency
  - Size of the sequential component is only O(log N)!
  - This is often the minimum size achievable
- Parallel performance model has the expected log term

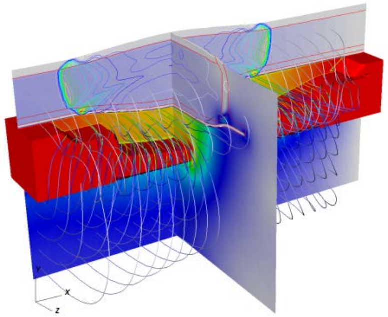
$$T_V = O(\log N)(\text{comm latency}) + O(\Gamma_p)(\text{comm rate}) + O(\Omega_p)(\text{flop rate})$$

Parallel computing imposes restrictions on multigrid algorithm development

- Avoid sequential techniques
  - Classical AMG coarsening
  - Gauss-Seidel smoother
  - Cycles with large sequential component
    - F-cycle: $O(\log^2 N)$
    - W-cycle: $O(2^{\log N}) = O(N)$
- Control communication
  - Galerkin coarse-grid operators (P^TAP) can lead to high communication costs in AMG
- Need both CS and Math advances!
  - New methods have new convergence and robustness characteristics
  - Successful addressing issues so far

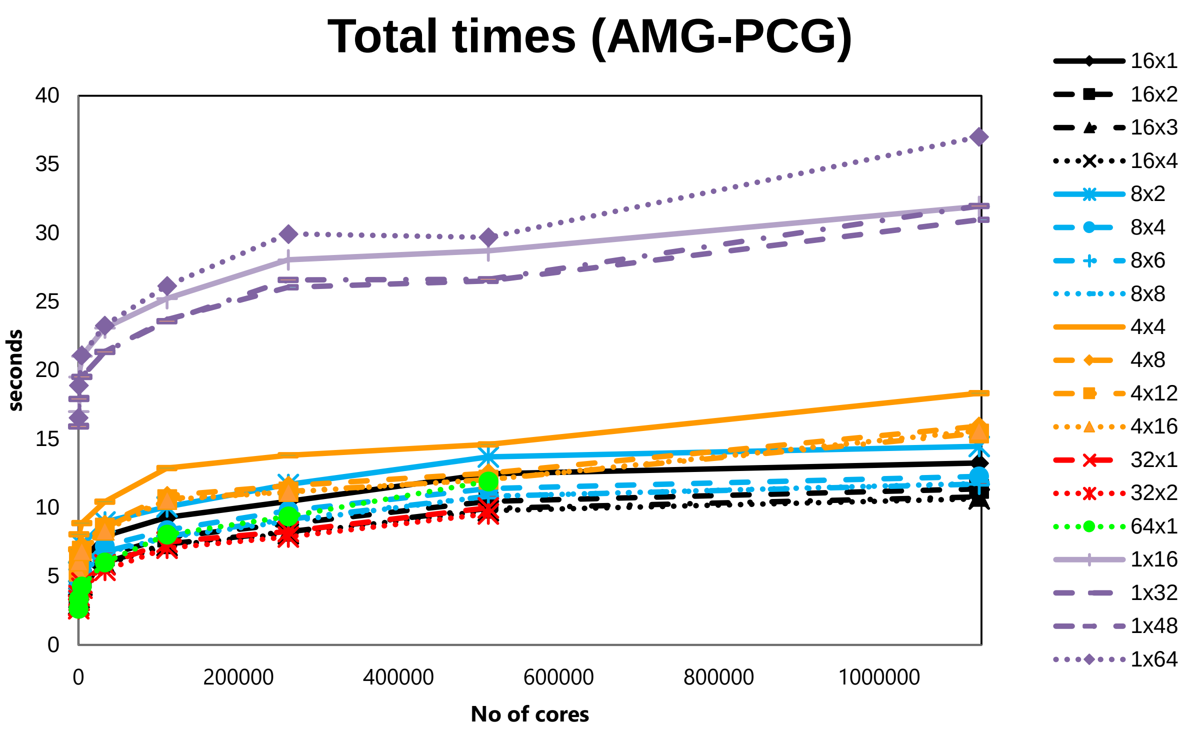


10x speedup for subsurface problems with new coarsening and interpolation approach



Magnetic flux compression generator simulation enabled by MG smoother research

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!

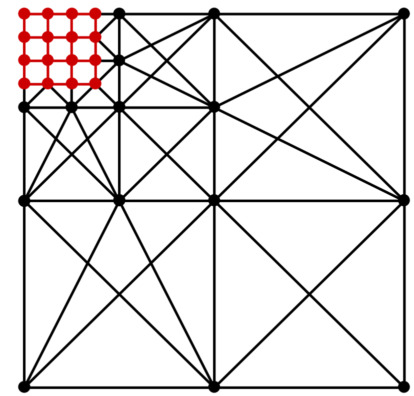
- *m* x *n* denotes *m* MPI tasks and *n* OpenMP threads per node
- Largest problem above: 72B unknowns on 1.1M cores

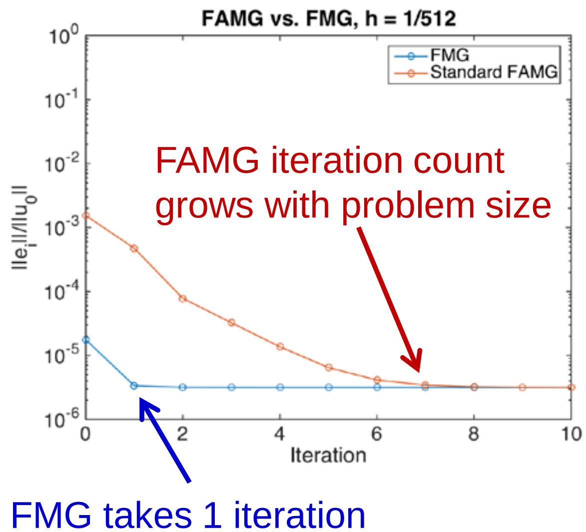
Reducing communication is the key to improving performance in parallel algebraic multigrid (AMG)

- AMG domain decomposition (AMG-DD) employs cheap global problems to speed up convergence
  - Constructs problems algebraically from an existing parallel AMG method
  - Developed a setup phase algorithm with the same $O(\log N)$ communication overhead as multigrid
  - Asymptotic convergence is unfortunately not better than the underlying AMG method

- Potential: FMG-like $O(N)$ convergence to discretization accuracy with only log N latency (vs $\log^2 N$)!

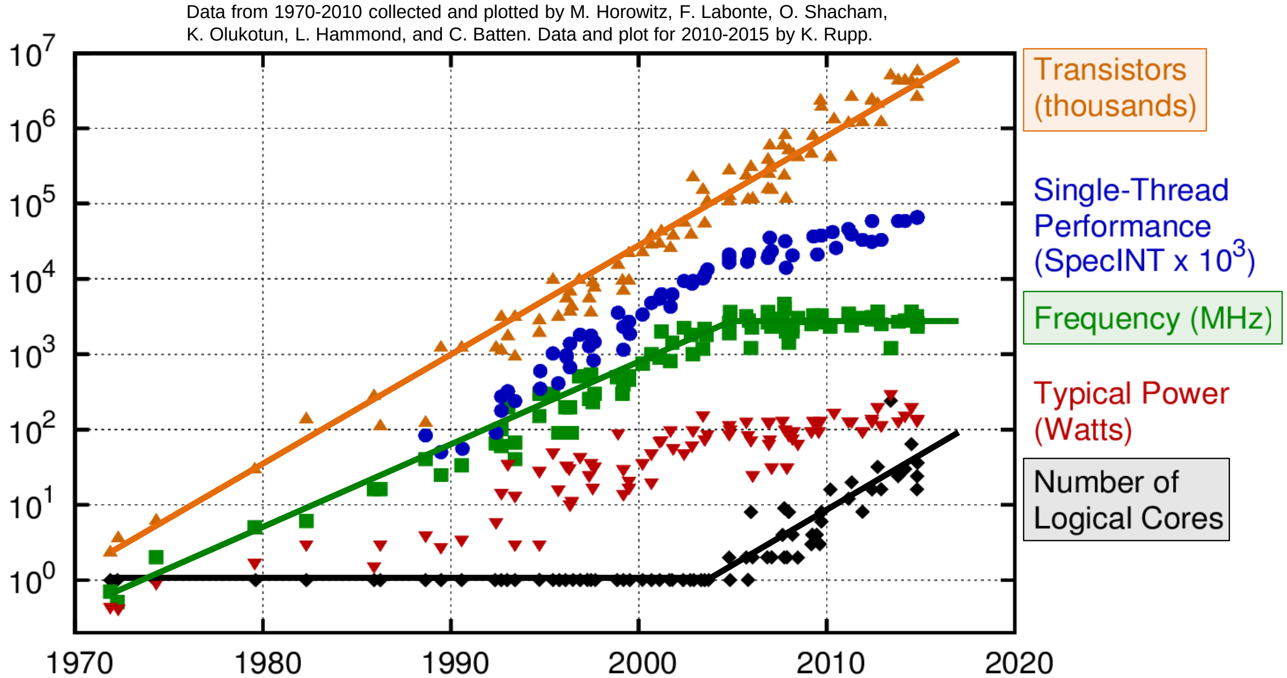
- Issue: Standard FAMG does not produce FMG-like convergence!
  - FAMG is AMG with an F-cycle





Parallel Time Integration

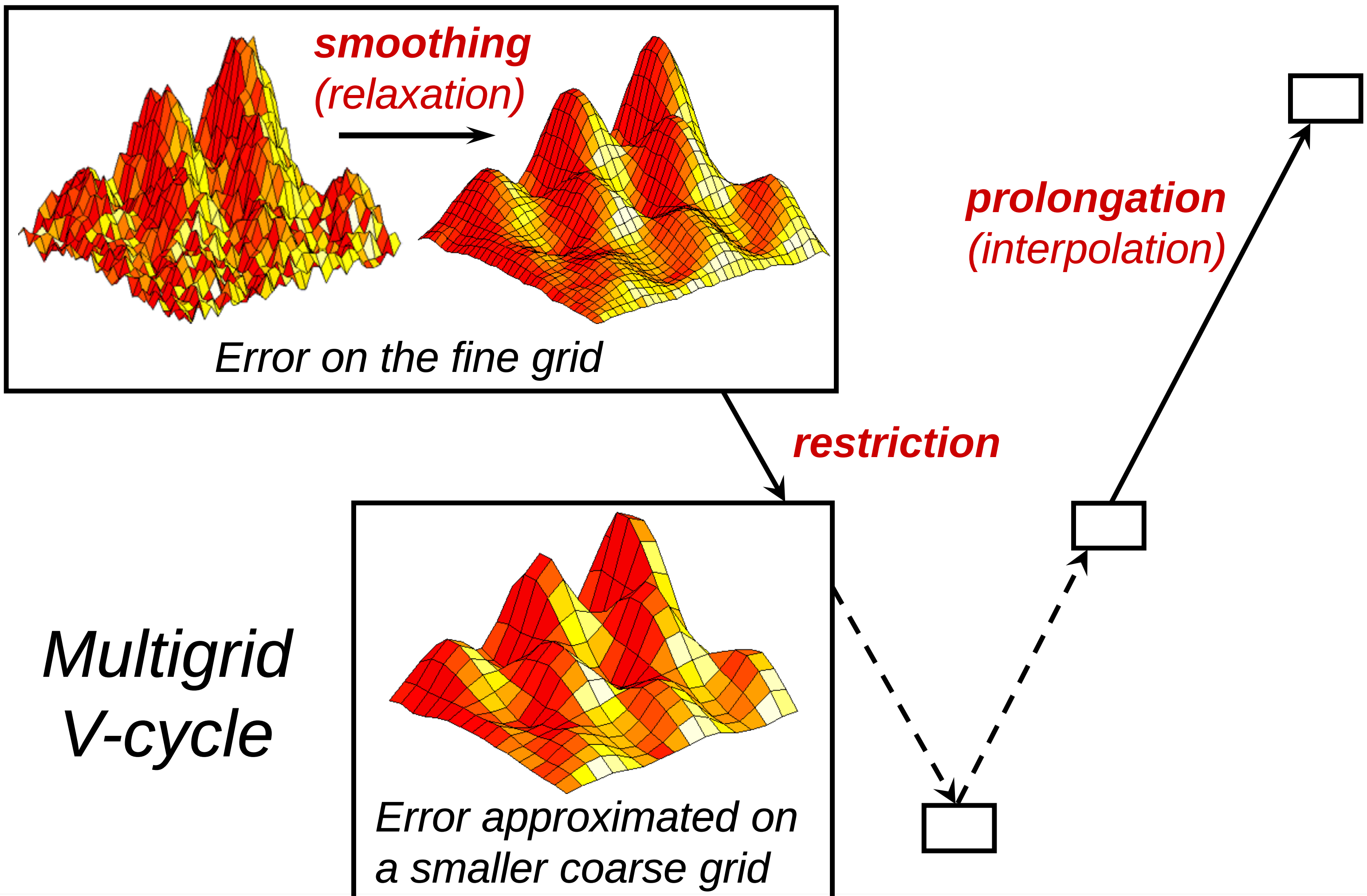
Parallel time integration is a major paradigm shift driven by hardware design realities



- Architecture trend: flat clock rates, more concurrency
  - Traditional time stepping is becoming a sequential bottleneck

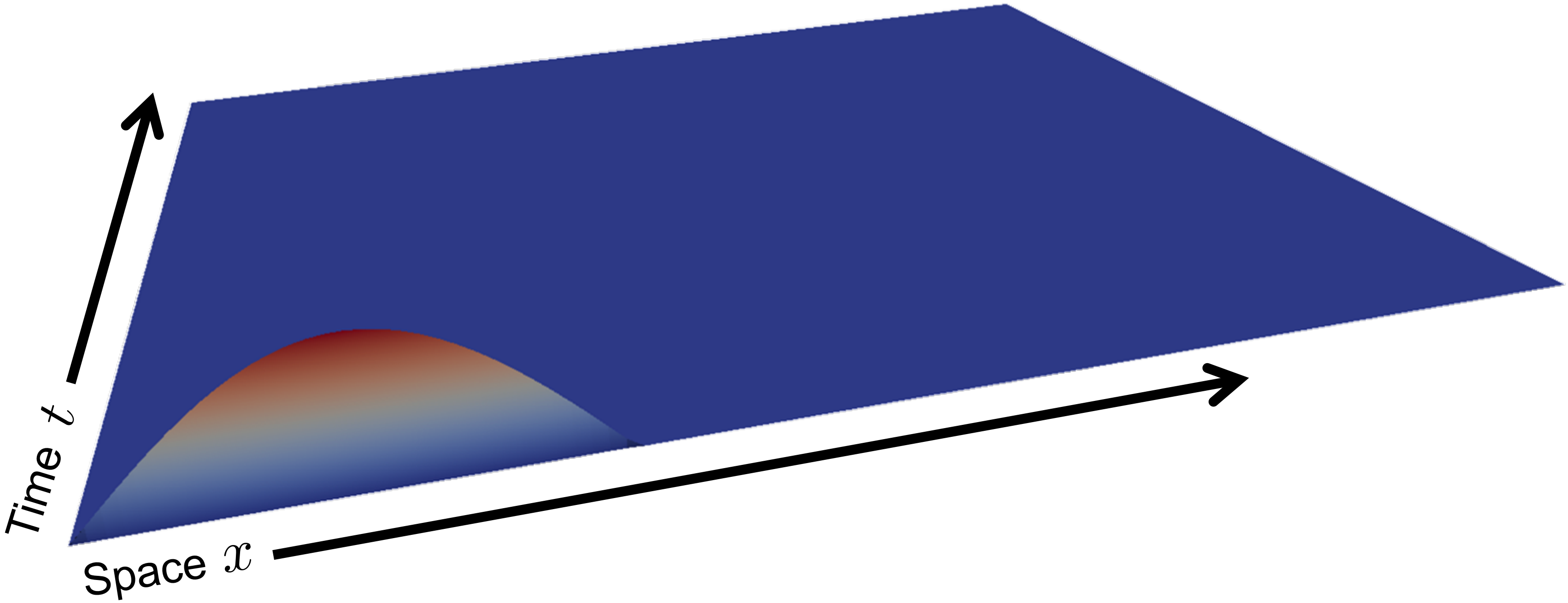
- Continued advancement in scientific simulation will require algorithms that are parallel in time

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



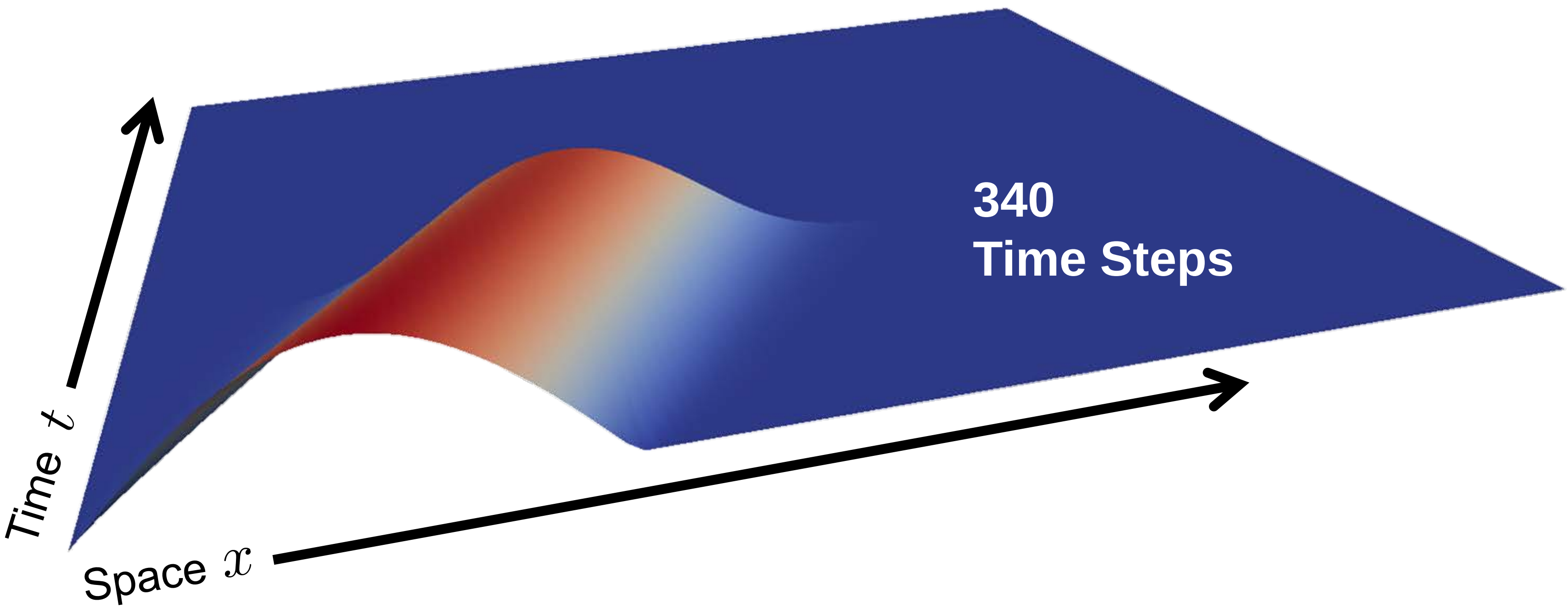
Time stepping is sequential

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



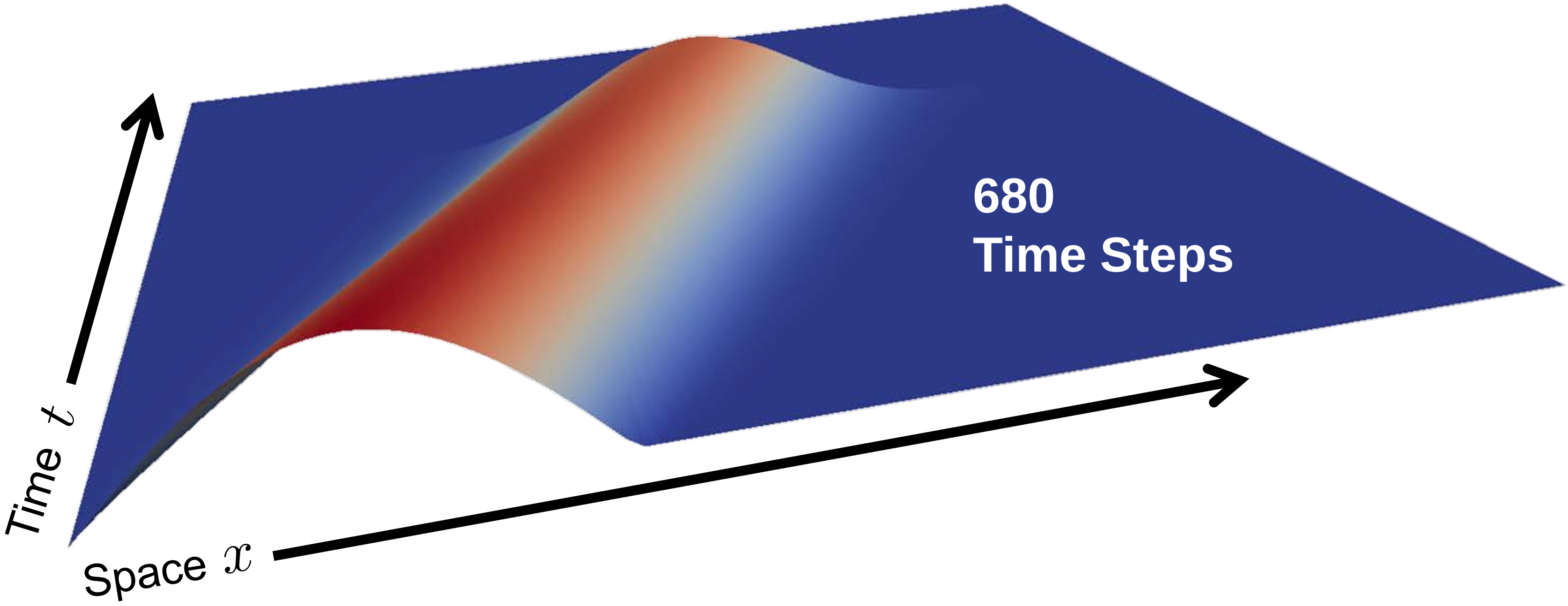
Time stepping is sequential

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



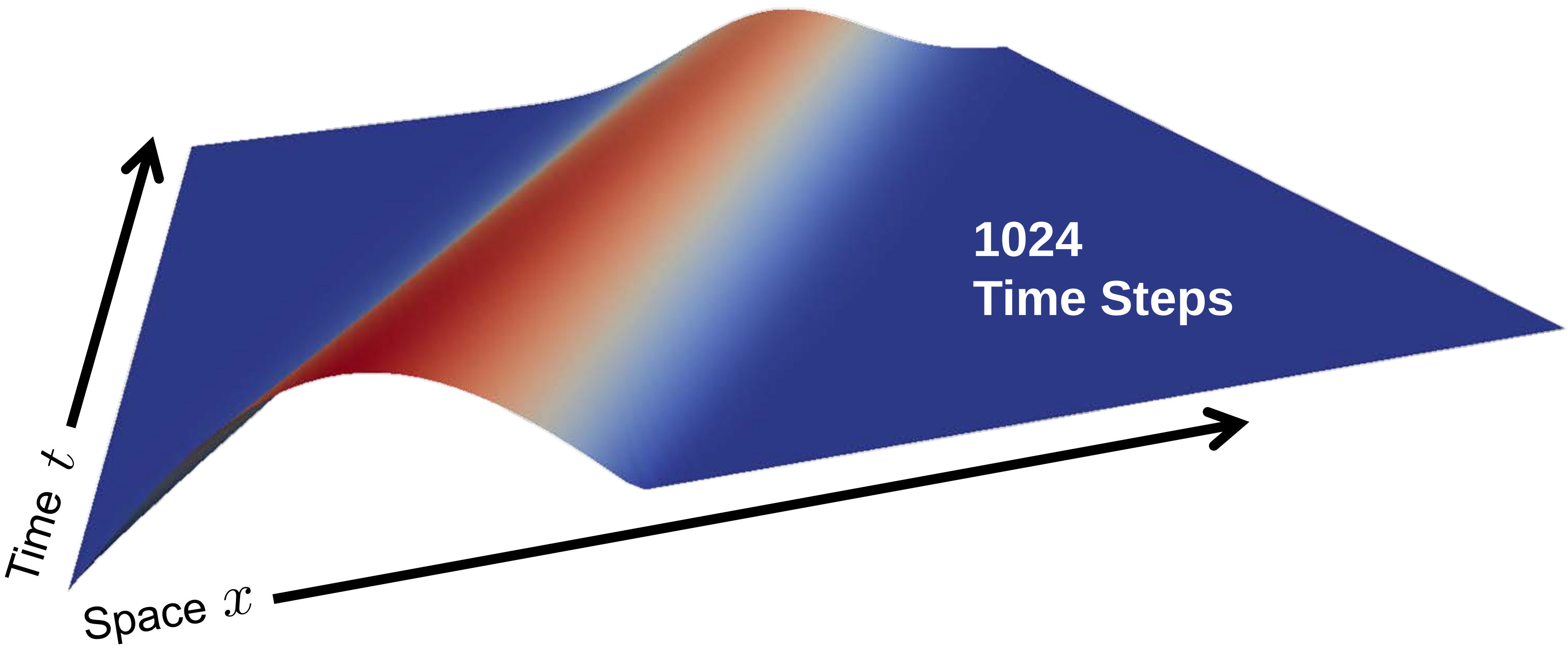
Time stepping is sequential

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



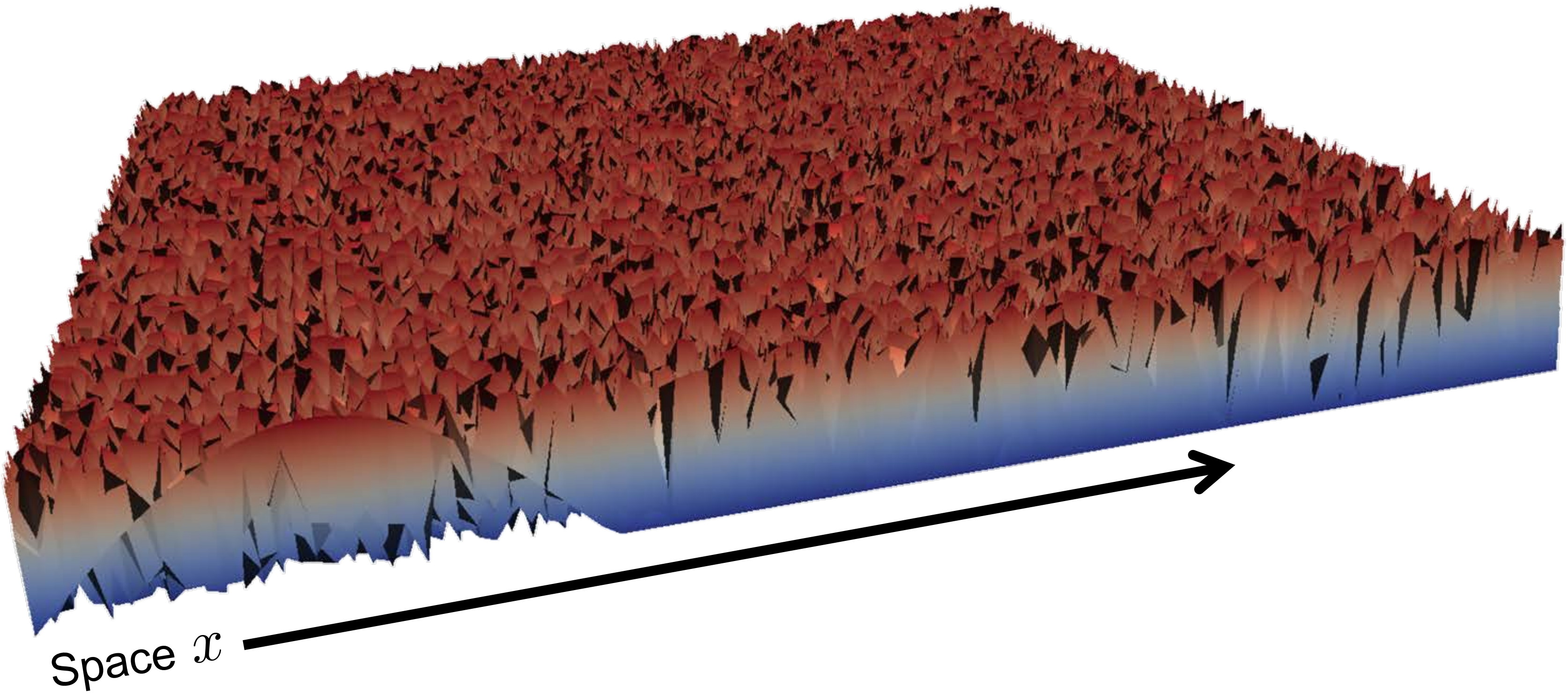
Time stepping is sequential

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



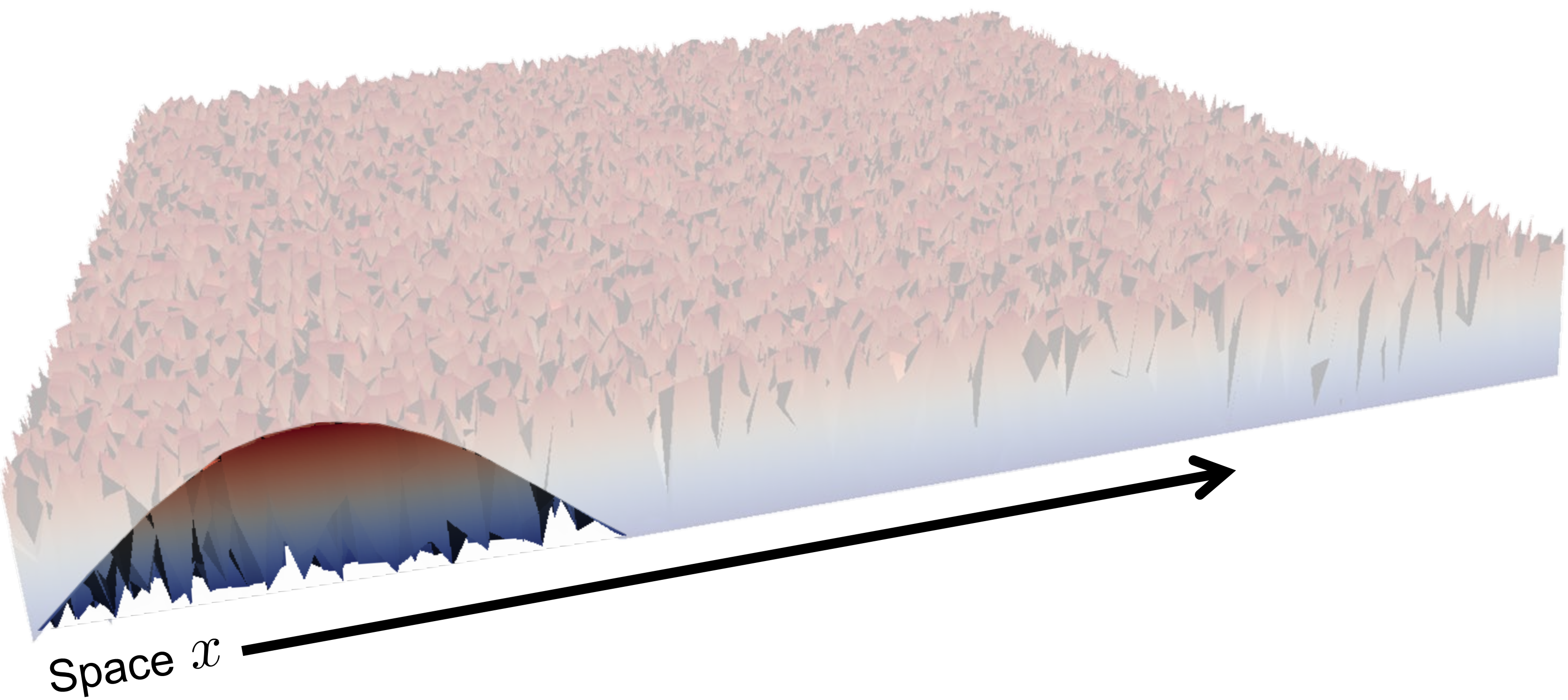
Multigrid-in-time converges to the serial space-time solution in parallel

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



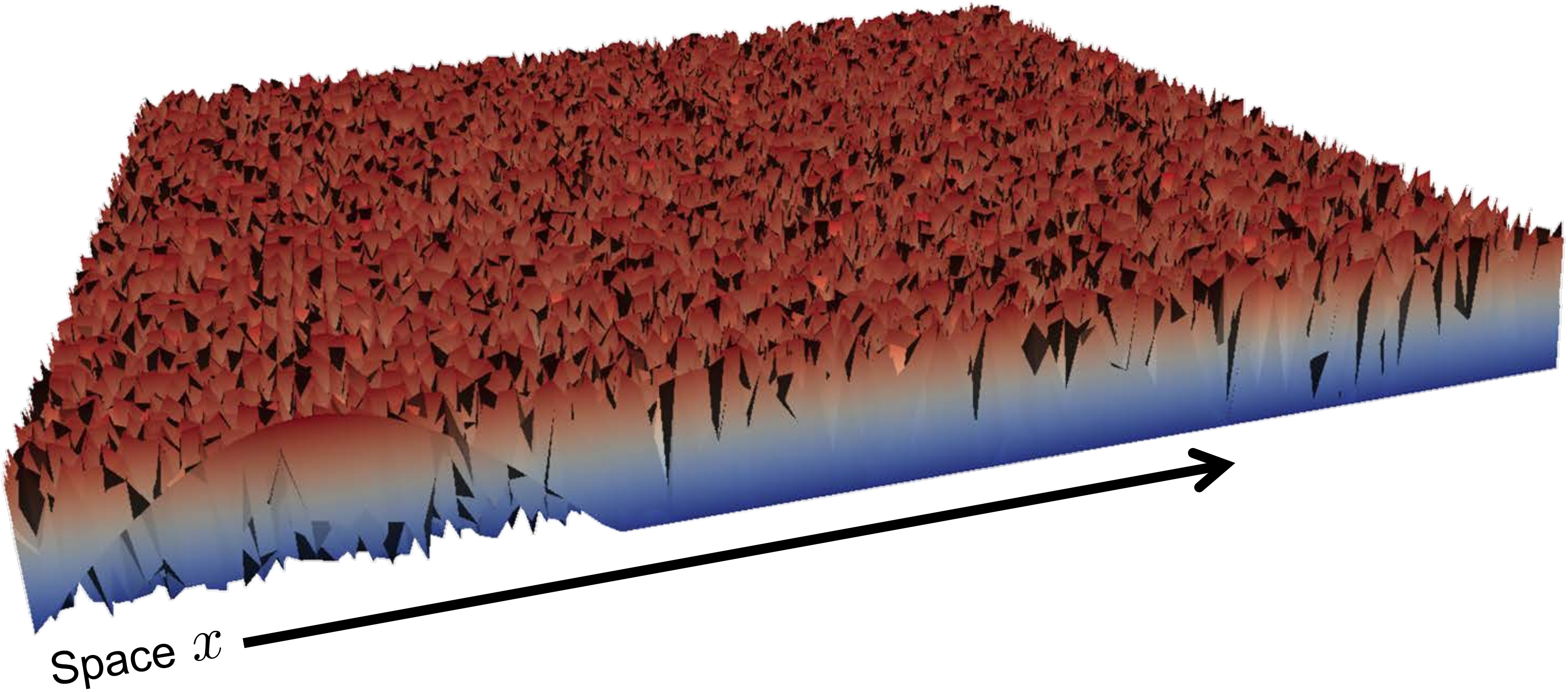
Multigrid-in-time converges to the serial space-time solution in parallel

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



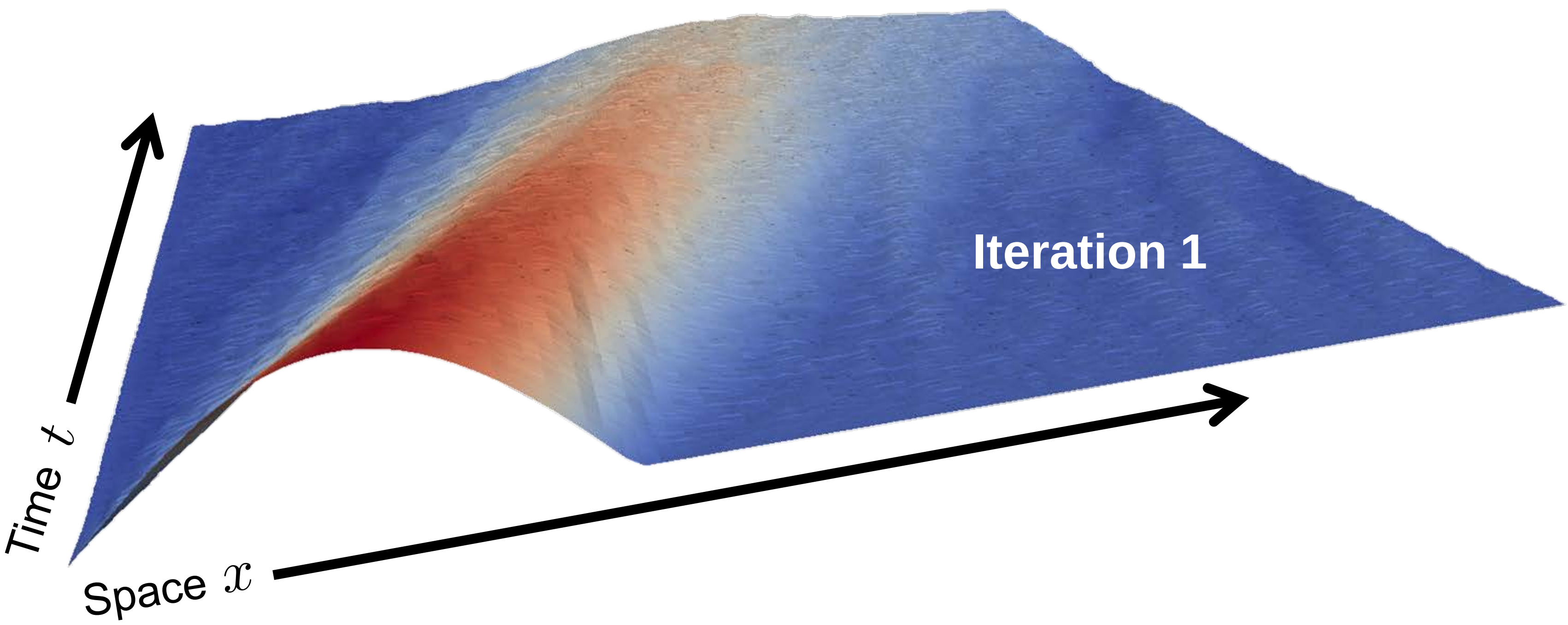
Multigrid-in-time converges to the serial space-time solution in parallel

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



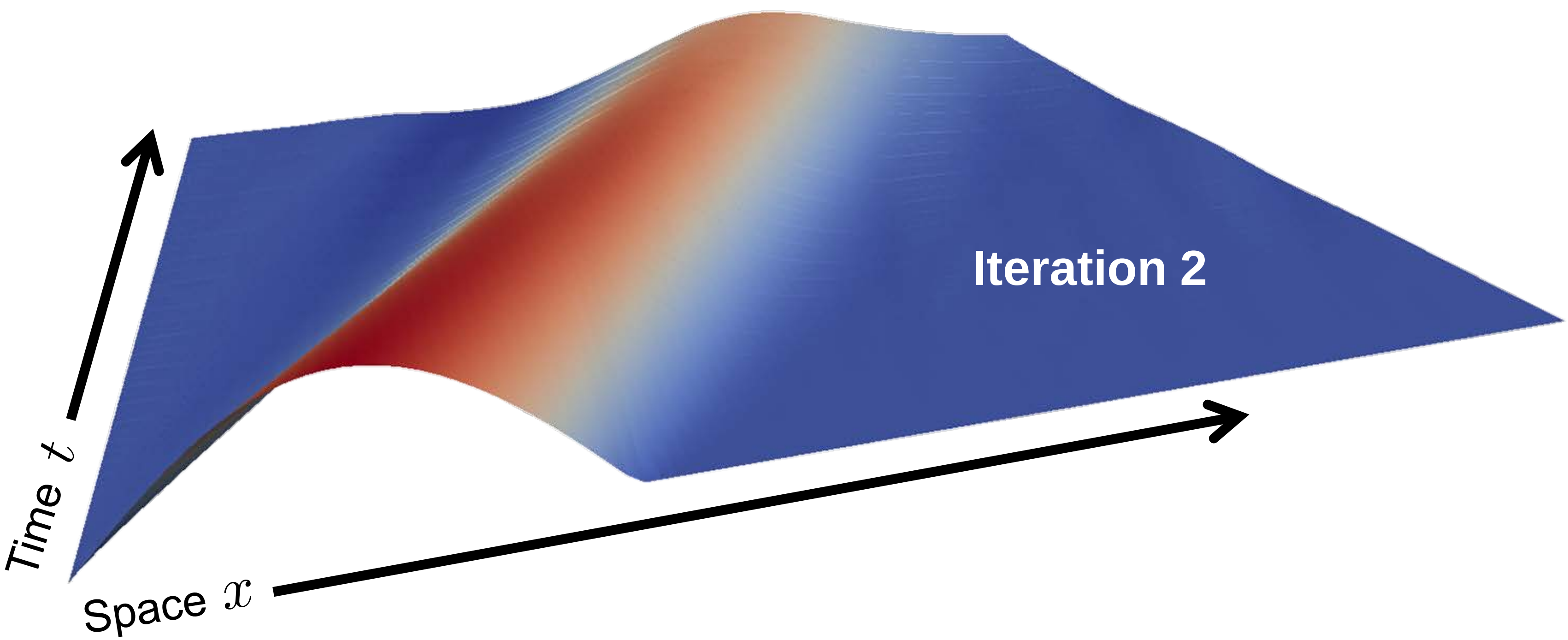
Multigrid-in-time converges to the serial space-time solution in parallel

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



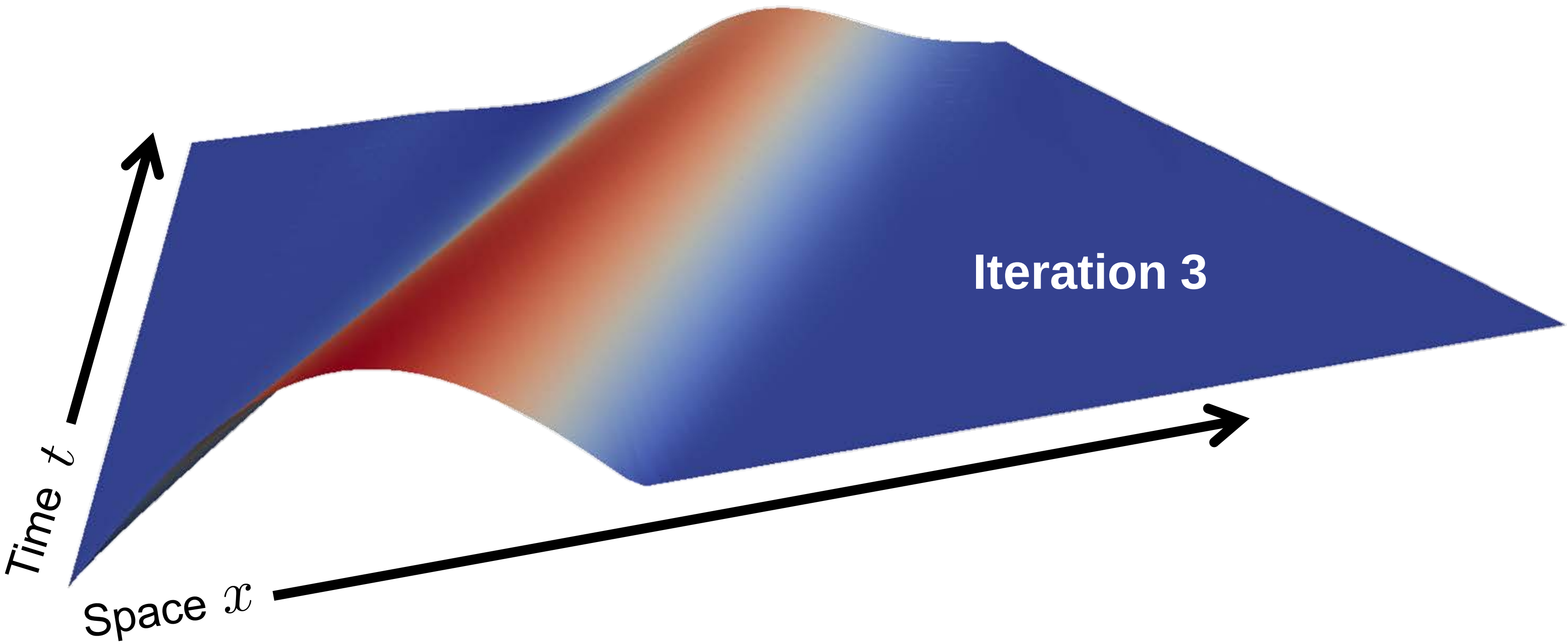
Multigrid-in-time converges to the serial space-time solution in parallel

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



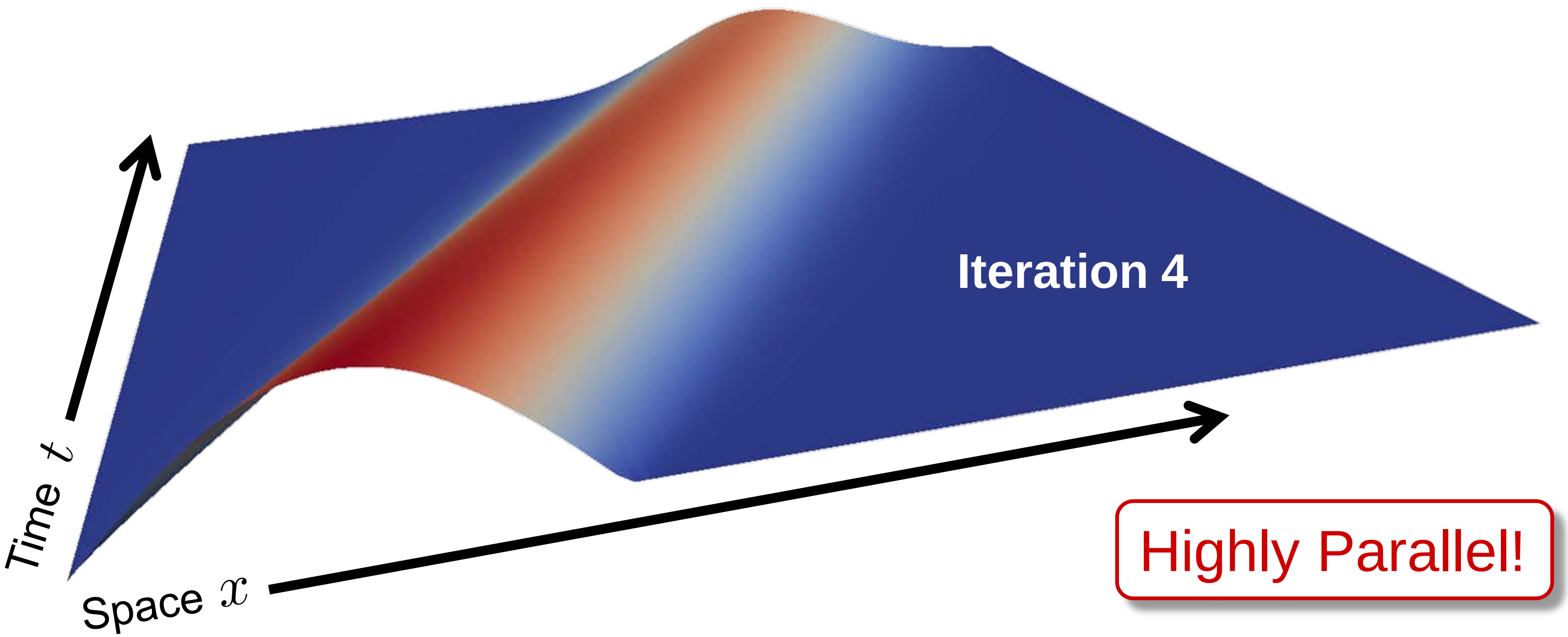
Multigrid-in-time converges to the serial space-time solution in parallel

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



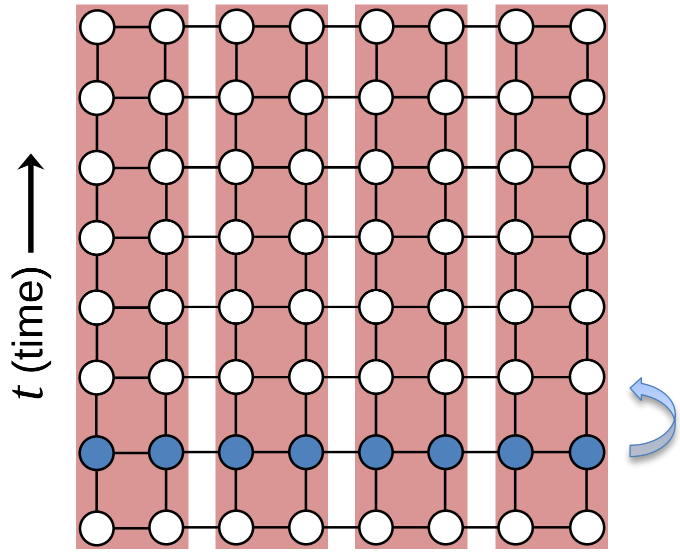
Multigrid-in-time converges to the serial space-time solution in parallel

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

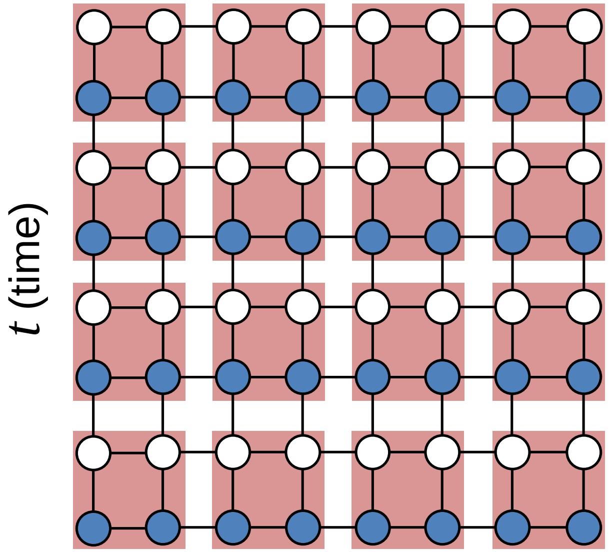


Significantly more parallel resources can be exploited with multigrid in time

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

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

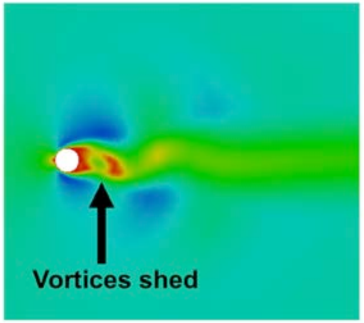
- Combines algorithm development, theory, and software proof-of-principle

- 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

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

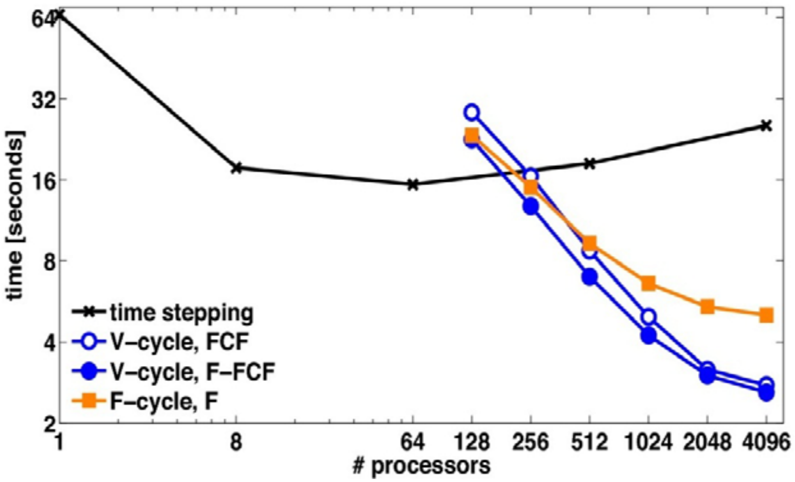
- XBraid is our open source implementation of MGRIT
  - User defines two objects and writes several wrapper routines (Step)
  - Only stores *C*-points to minimize storage

- 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 dictated 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

Some Progress and Current Research Directions

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

- Assume Φ and Φ_Δ are simultaneously diagonalizable with eigenvalues λ_ω, μ_ω

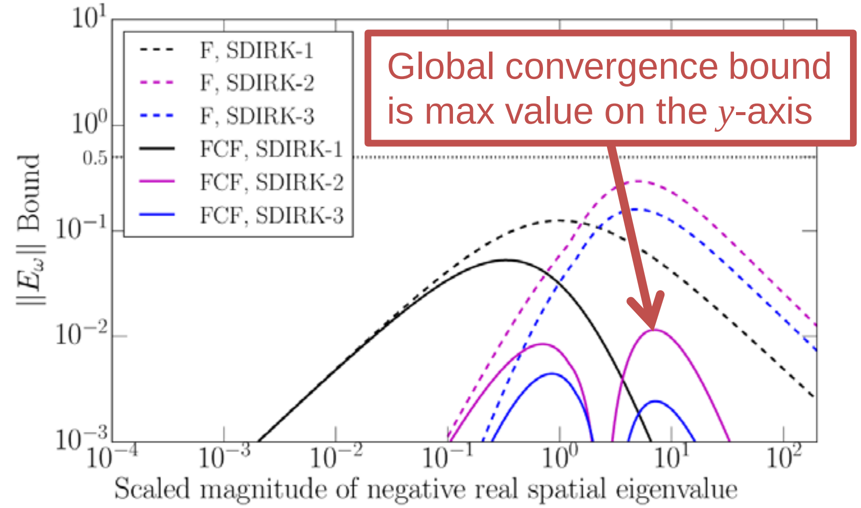
- Sharp bound for error propagator

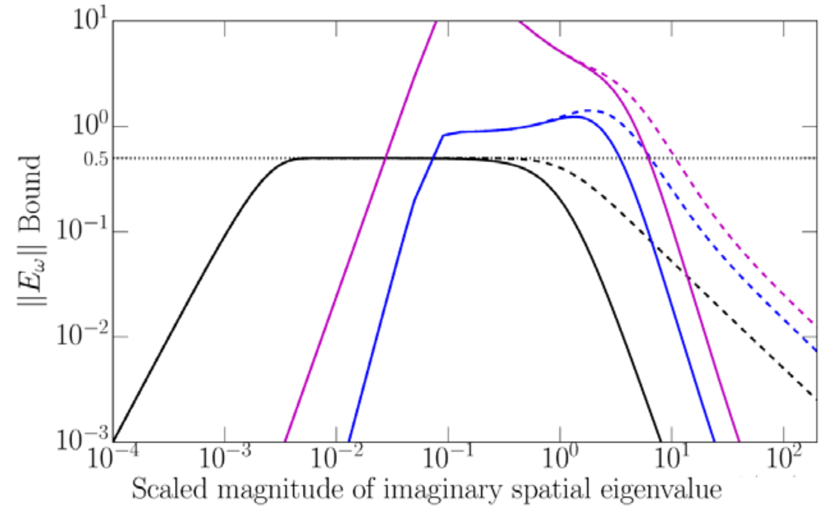
$$\|E\| \leq \max_{\omega} |\lambda_\omega^m - \mu_\omega| \frac{1 - |\mu_\omega|^{N_T - 1}}{1 - |\mu_\omega|} |\lambda_\omega|^m$$

- 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

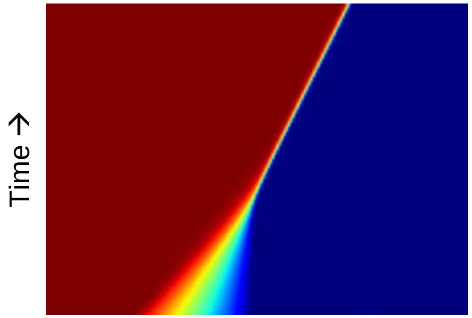




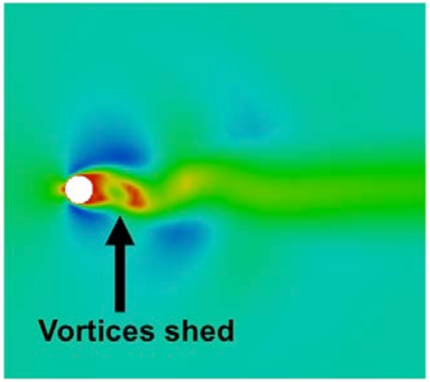
Hyperbolic problems are a major new emphasis for our MGRIT algorithm research

- 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 (implicit) schemes to date
- 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

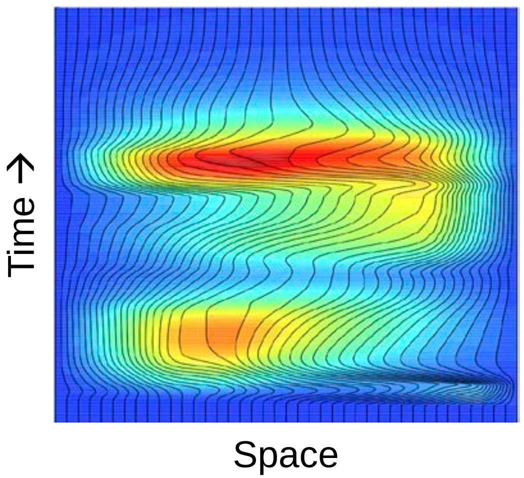


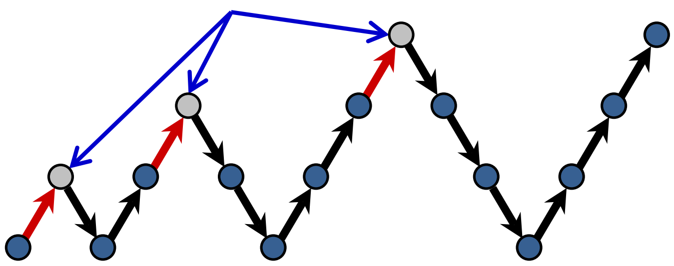
Navier-Stokes
7.5x speedup in Strand2D



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

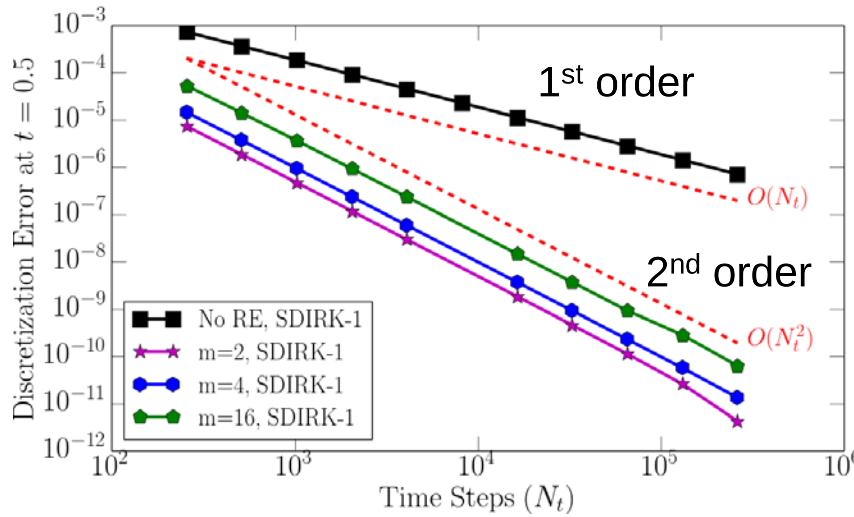
- Moving spatial mesh
  - 1D diffusion with time dependent source
  - Unsteady flow around moving cylinder
- Temporal refinement via Full Multigrid (FMG)
  - ODE simulation of satellite orbit
  - DAE power grid simulations in GridDyn (25x speedup)
- Temporal and spatial refinement
  - 2D heat equation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



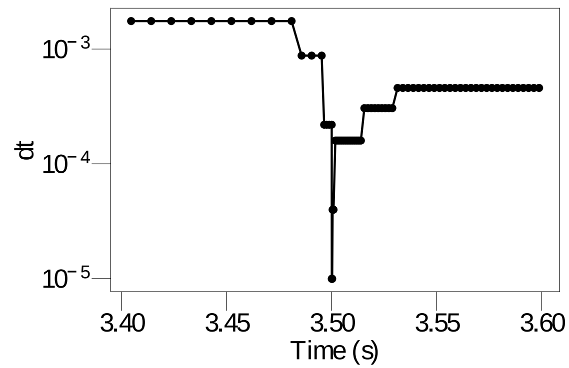


Other developments and research directions

- 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
  - 10x to 50x speedup
  - Investigating approaches for unscheduled discontinuities



Adaptive time grid around load discontinuity



Summary and Conclusions

- Multigrid methods are ideal for exascale
  - Optimal, naturally resilient to faults, minimize data movement
- Parallel computing imposes additional restrictions on multigrid algorithmic development, especially for AMG
  - Success scaling to BG/Q-class machines
- 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)
  - Demonstrated speedups for a variety of problems
- There is much future work to be done!
  - More problem types, more complicated discretizations, performance improvements, adaptive meshing, ...

Our Multigrid and Parallel Time Integration Research Team


Veselin Dobrev

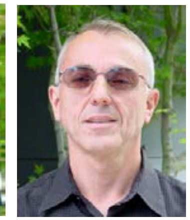

Rob Falgout


Tzanio Kolev


Ruipeng Li


Daniel Osei-Kuffuor


Jacob Schroder


Panayot Vassilevski


Lu Wang


Ulrike Yang

- 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), CEA (Lecouvez)
- Software, publications, and other information

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

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

Thank You!

Lawrence Livermore
National Laboratory