Optimal Solvers in PETSc

Matthew Knepley

Mathematics and Computer Science Division
Argonne National Laboratory

Monash AuScope Simulation & Modelling Victoria
Monash University, Victoria
Feb 15, 2008

Outline

Outline

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and
- and Unstructured
- Triangles and
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and
- and Unstructured
- Triangles and
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and Finite Element
- and Unstructured
- Triangles and
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and Finite Element
- Structured and Unstructured
- Triangles and
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and Finite Element
- Structured and Unstructured
- Triangles and Hexes
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and Finite Element
- Structured and Unstructured
- Triangles and Hexes
- Optimal Solvers

Item in red not yet finished

PETSc Capabilities

- Serial (laptop) and Parallel (Cray XT4)
- Linear and Nonlinear
- Finite Difference, Finite Volume, and Finite Element
- Structured and Unstructured
- Triangles and Hexes
- Optimal Solvers

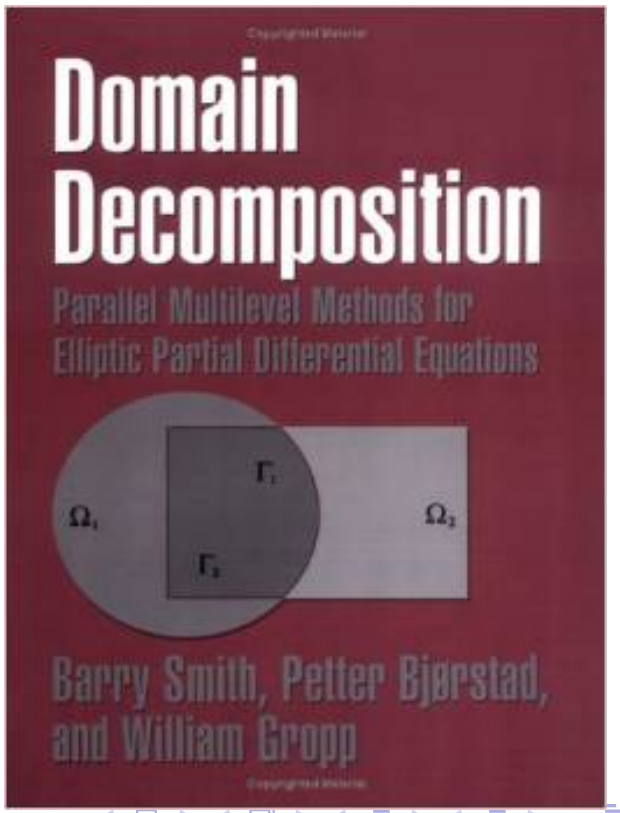
Item in red not yet finished

How did PETSc Originate?

PETSc was developed as a Platform for Experimentation

We want to experiment with different

- Models
- Discretizations
- Solvers
- Algorithms
  - which blur these boundaries



The Role of PETSc

Developing parallel, nontrivial PDE solvers that deliver high performance is still difficult and requires months (or even years) of concentrated effort.

PETSc is a toolkit that can ease these difficulties and reduce the development time, but it is not a black-box PDE solver, nor a silver bullet.

— Barry Smith

Advice from Bill Gropp

You want to think about how you decompose your data structures, how you think about them globally. [...] If you were building a house, you'd start with a set of blueprints that give you a picture of what the whole house looks like. You wouldn't start with a bunch of tiles and say. "Well I'll put this tile down on the ground, and then I'll find a tile to go next to it." But all too many people try to build their parallel programs by creating the smallest possible tiles and then trying to have the structure of their code emerge from the chaos of all these little pieces. You have to have an organizing principle if you're going to survive making your code parallel.

(http://www.rce-cast.com/Podcast/rce-28-mpich2.html)

What is PETSc?

A freely available and supported research code for the parallel solution of nonlinear algebraic equations

Free

- Download from http://www.mcs.anl.gov/petsc
- Free for everyone, including industrial users

Supported

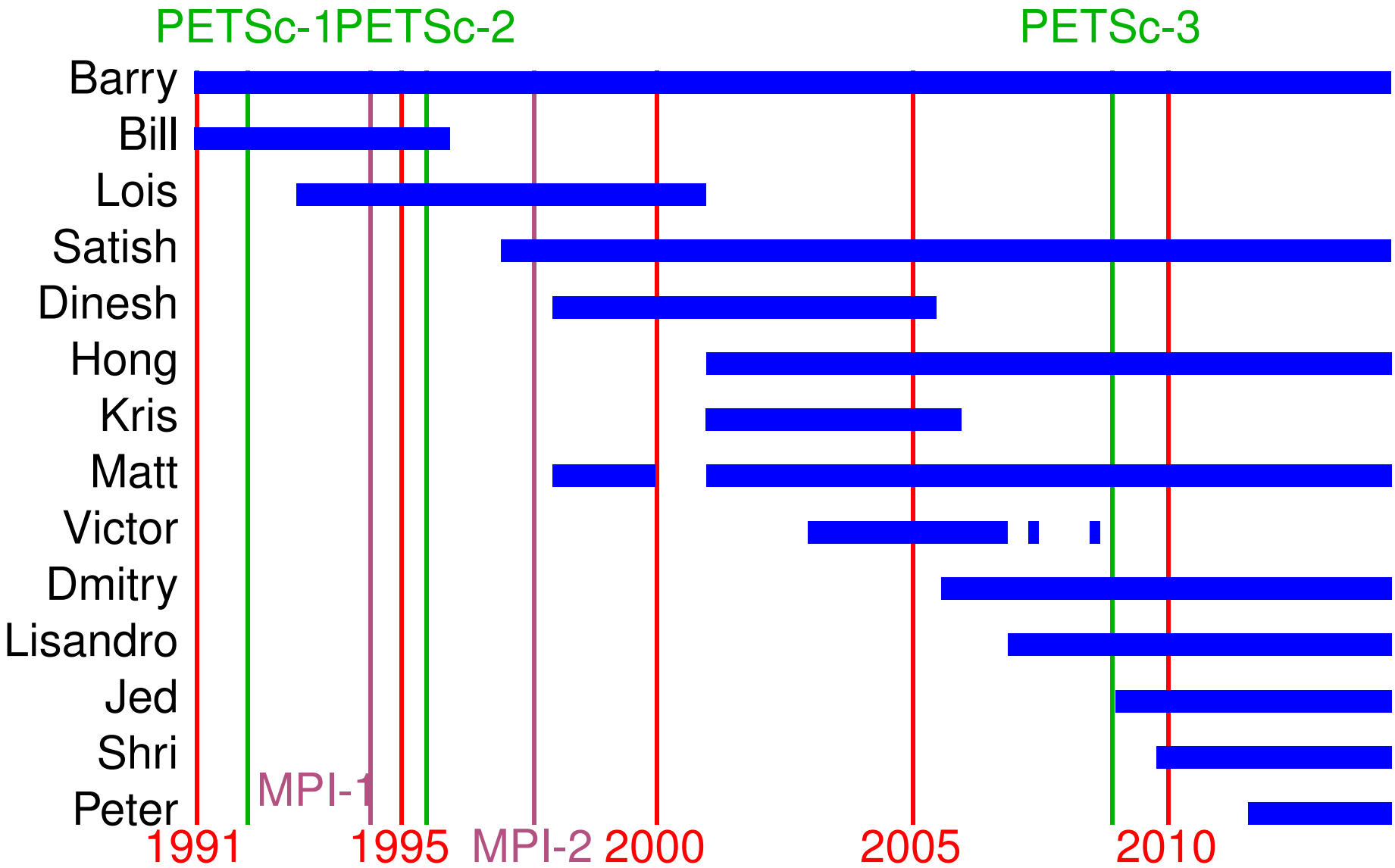
- Hyperlinked manual, examples, and manual pages for all routines
- Hundreds of tutorial-style examples
- Support via email: petsc-maint@mcs.anl.gov

Usable from C, C++, Fortran 77/90, Matlab, Julia, and Python

What is PETSc?

- Portable to any parallel system supporting MPI, including:
  - Tightly coupled systems
    - Cray XT6, BG/Q, NVIDIA Fermi, K Computer
  - Loosely coupled systems, such as networks of workstations
    - IBM, Mac, iPad/iPhone, PCs running Linux or Windows
- PETSc History
  - Begun September 1991
  - Over 60,000 downloads since 1995 (version 2)
  - Currently 400 per month
- PETSc Funding and Support
  - Department of Energy
    - SciDAC, MICS Program, AMR Program, INL Reactor Program
  - National Science Foundation
    - CIG, CISE, Multidisciplinary Challenge Program

Timeline


PETSc-1
PETSc-2
PETSc-3
Barry
Bill
Lois
Satish
Dinesh
Hong
Kris
Matt
Victor
Dmitry
Lisandro
Jed
Shri
Peter
MPI-1
MPI-2
1991
1995
2000
2005
2010

What Can We Handle?

- PETSc has run implicit problems with over 500 billion unknowns
  - UNIC on BG/P and XT5
  - PFLOTRAN for flow in porous media
- PETSc has run on over 290,000 cores efficiently
  - UNIC on the IBM BG/P Jugene at Jülich
  - PFLOTRAN on the Cray XT5 Jaguar at ORNL
- PETSc applications have run at 23% of peak (600 Teraflops)
  - Jed Brown on NERSC Edison
  - HPGMG code

What Can We Handle?

- PETSc has run implicit problems with over 500 billion unknowns
  - UNIC on BG/P and XT5
  - PFLOTRAN for flow in porous media
- PETSc has run on over 290,000 cores efficiently
  - UNIC on the IBM BG/P Jugene at Jülich
  - PFLOTRAN on the Cray XT5 Jaguar at ORNL
- PETSc applications have run at 23% of peak (600 Teraflops)
  - Jed Brown on NERSC Edison
  - HPGMG code

What Can We Handle?

- PETSc has run implicit problems with over 500 billion unknowns
  - UNIC on BG/P and XT5
  - PFLOTRAN for flow in porous media
- PETSc has run on over 290,000 cores efficiently
  - UNIC on the IBM BG/P Jugene at Jülich
  - PFLOTRAN on the Cray XT5 Jaguar at ORNL
- PETSc applications have run at 23% of peak (600 Teraflops)
  - Jed Brown on NERSC Edison
  - HPGMG code

Outline

Who Uses PETSc?

Computational Scientists

- Earth Science
  - PyLith (CIG)
  - Underworld (Monash)
  - Magma Dynamics (LDEO, Columbia, Oxford)
- Subsurface Flow and Porous Media
  - STOMP (DOE)
  - PFLOTRAN (DOE)

Who Uses PETSc?

Computational Scientists

- CFD
  - Firedrake
  - Fluidity
  - OpenFOAM
  - freeCFD
  - OpenFVM
- MicroMagnetics
  - MagPar
- Fusion
  - XGC
  - BOUT++
  - NIMROD

Who Uses PETSc?

Algorithm Developers

- Iterative methods
  - Deflated GMRES
  - LGMRES
  - QCG
  - SpecEst
- Preconditioning researchers
  - Prometheus (Adams)
  - ParPre (Eijkhout)
  - FETI-DP (Klawonn and Rheinbach)

Who Uses PETSc?

Algorithm Developers

- Finite Elements
  - libMesh
  - MOOSE
  - PETSc-FEM
  - Deal II
  - OOFEM
- Other Solvers
  - Fast Multipole Method (PetFMM)
  - Radial Basis Function Interpolation (PetRBF)
  - Eigensolvers (SLEPc)
  - Optimization (TAO)

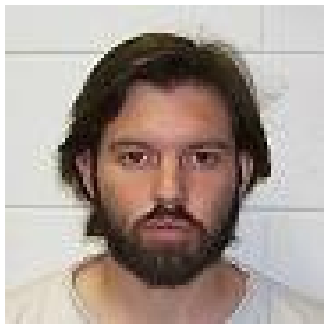
The PETSc Team


Bill Gropp


Barry Smith


Satish Balay


Jed Brown


Matt Knepley


Lisandro Dalcin


Hong Zhang


Mark Adams


Toby Issac

Outline

Downloading PETSc

- The latest tarball is on the PETSc site:
  http://www.mcs.anl.gov/petsc/download
- There is a Debian package (`aptitude install petsc-dev`)
- There is a Git development repository

Cloning PETSc

- The full development repository is open to the public
  - https://bitbucket.org/petsc/petsc/
- Why is this better?
  - You can clone to any release (or any specific ChangeSet)
  - You can easily rollback changes (or releases)
  - You can get fixes from us the same day
- All releases are just tags:
  - Source at tag v3.4.4

Automatic Downloads

- Starting in 2.2.1, some packages are automatically
  - Downloaded
  - Configured and Built (in `$PETSC_DIR/externalpackages`)
  - Installed with PETSc
- Currently works for
  - petsc4py
  - PETSc documentation utilities (Sowing, lgrind, c2html)
  - BLAS, LAPACK, BLACS, ScaLAPACK, PLAPACK
  - MPICH, MPE, OpenMPI
  - ParMetis, Chaco, Jostle, Party, Scotch, Zoltan
  - MUMPS, Spooles, SuperLU, SuperLU_Dist, UMFPack, pARMS
  - BLOPEX, FFTW, SPRNG
  - Prometheus, HYPRE, ML, SPAI
  - Sundials
  - Triangle, TetGen
  - FIAT, FFC, Generator
  - Boost

Outline

Necessity Of Simulation

- Lasers and Energy
  - Combustion, NIF, ICF
  - Experiments are expensive
- Engineering
  - Aerodynamics, crash testing
  - Experiments are difficult to instrument
- Applied Physics
  - Radiation transport, supernovae
  - Experiments are impossible or prohibited
- Environment
  - Global climate, contaminant transport
  - Experiments are impossible or dangerous
- Biology
  - Drug design, ion channels
  - Experiments are controversial

What Is Optimal?

I will define *optimal* as an $\mathcal{O}(N)$ solution algorithm

These are generally hierarchical, so we need

- hierarchy generation
- assembly on subdomains
- restriction and prolongation

Payoff

Why should I care?

1. Current algorithms do not efficiently utilize modern machines
2. Processor flops are increasing much faster than bandwidth
3. Multicore processors are the future
4. Optimal multilevel solvers are necessary

Payoff

Why should I care?

1. Current algorithms do not efficiently utilize modern machines
2. Processor flops are increasing much faster than bandwidth
3. Multicore processors are the future
4. Optimal multilevel solvers are necessary

Payoff

Why should I care?

1. Current algorithms do not efficiently utilize modern machines
2. Processor flops are increasing much faster than bandwidth
3. Multicore processors are the future
4. Optimal multilevel solvers are necessary

Payoff

Why should I care?

1. Current algorithms do not efficiently utilize modern machines
2. Processor flops are increasing much faster than bandwidth
3. Multicore processors are the future
4. Optimal multilevel solvers are necessary

Payoff

Why should I care?

1. Current algorithms do not efficiently utilize modern machines
2. Processor flops are increasing much faster than bandwidth
3. Multicore processors are the future
4. Optimal multilevel solvers are necessary

Claim: Hierarchical operations can be handled by a single interface

Why Optimal Algorithms?

- The more powerful the computer,
  the greater the importance of optimality
- Example:
  - Suppose Alg_1 solves a problem in time CN^2, N is the input size
  - Suppose Alg_2 solves the same problem in time CN
  - Suppose Alg_1 and Alg_2 are able to use 10,000 processors
- In constant time compared to serial,
  - Alg1 can run a problem 100X larger
  - Alg2 can run a problem 10,000X larger
- Alternatively, filling the machine's memory,
  - Alg1 requires 100X time
  - Alg2 runs in constant time

Multigrid

Multigrid is *optimal* in that is does $\mathcal{O}(N)$ work for $||r|| < \epsilon$

- Brandt, Briggs, Wan & Chan & Smith
- Constant work per level
  - Sufficiently strong solver
  - Need a constant factor decrease in the residual
- Constant factor decrease in dof
  - Log number of levels
- Sufficiently good interpolation
  - Preserves low modes
  - Cannot dump too much energy into high modes

Linear Convergence of the Poisson Problem

Convergence to $||r|| < 10^{-9}||b||$ using GMRES(30)/ILU

Elements	Iterations
128	10
256	17
512	24
1024	34
2048	67
4096	116
8192	167
16384	329
32768	558
65536	920
131072	1730

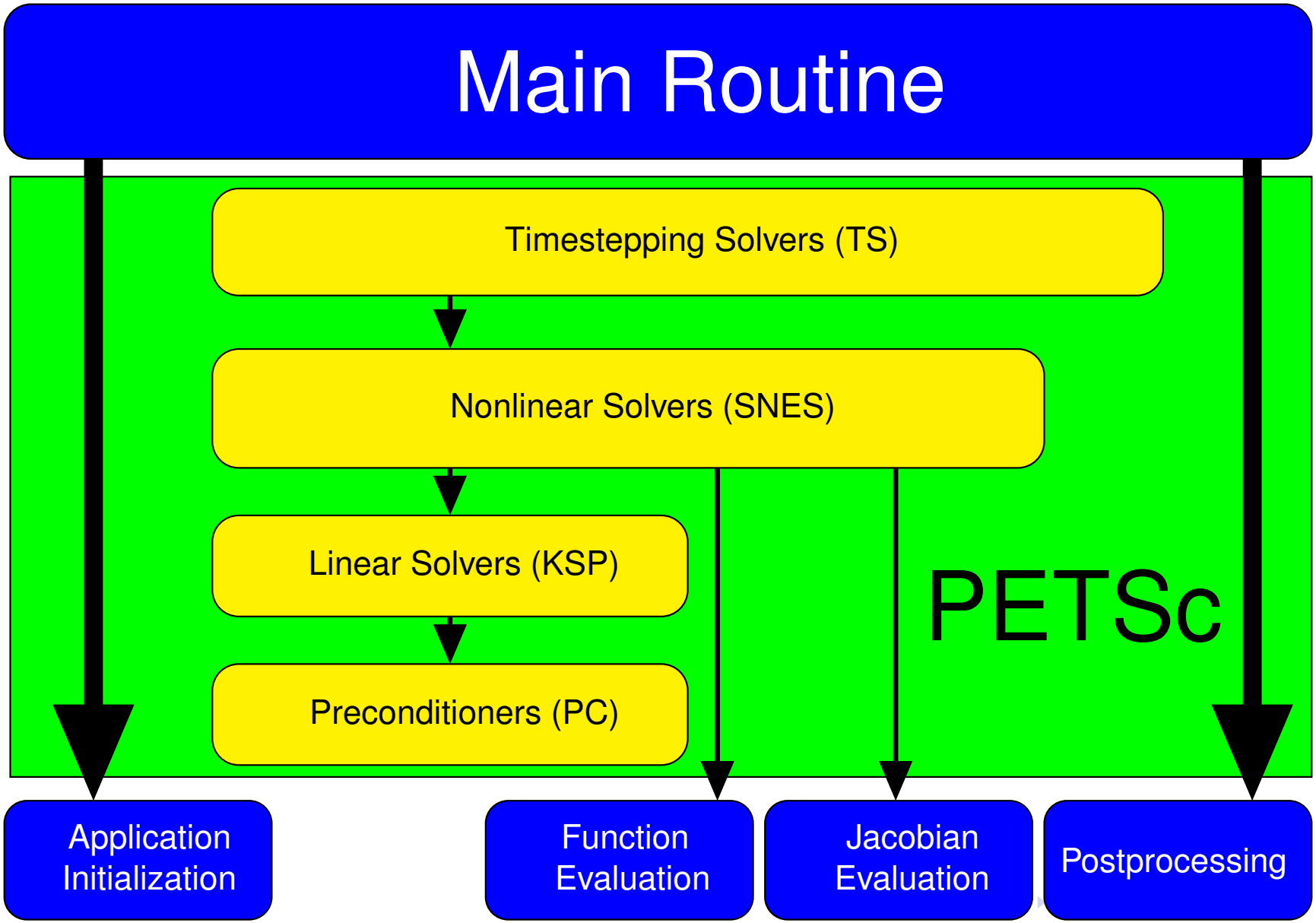
Linear Convergence of the Poisson Problem

Convergence to $||r|| < 10^{-9}||b||$ using GMRES(30)/MG

Elements	Iterations
128	5
256	7
512	6
1024	7
2048	6
4096	7
8192	6
16384	7
32768	6
65536	7
131072	6

Outline

Flow Control for a PETSc Application



SNES Paradigm

The SNES interface is based upon callback functions

- FormFunction(), set by SNESSetFunction()
- FormJacobian(), set by SNESSetJacobian()

When PETSc needs to evaluate the nonlinear residual $F(x)$,

- Solver calls the **user's** function
- User function gets application state through the `ctx` variable
  - PETSc never sees application data

Higher Level Abstractions

The PETSc `DA` class is a topology and discretization interface.

- Structured grid interface
  - Fixed simple topology
- Supports stencils, communication, reordering
  - Limited idea of operators
- Nice for simple finite differences

The PETSc `Mesh` class is a topology interface.

- Unstructured grid interface
  - Arbitrary topology and element shape
- Supports partitioning, distribution, and global orders

Higher Level Abstractions

The PETSc `DM` class is a hierarchy interface.

- Supports multigrid
  - `PCMG` combines it with a multigrid preconditioner
- Abstracts the logic of multilevel methods

The `PetscSection` class is a helper class for data layout.

- Functions over unstructured grids
  - Arbitrary layout of degrees of freedom
- Enables distribution and assembly

A DMDA is more than a Mesh

A DMDA contains topology, geometry, and (sometimes) an implicit Q1 discretization.

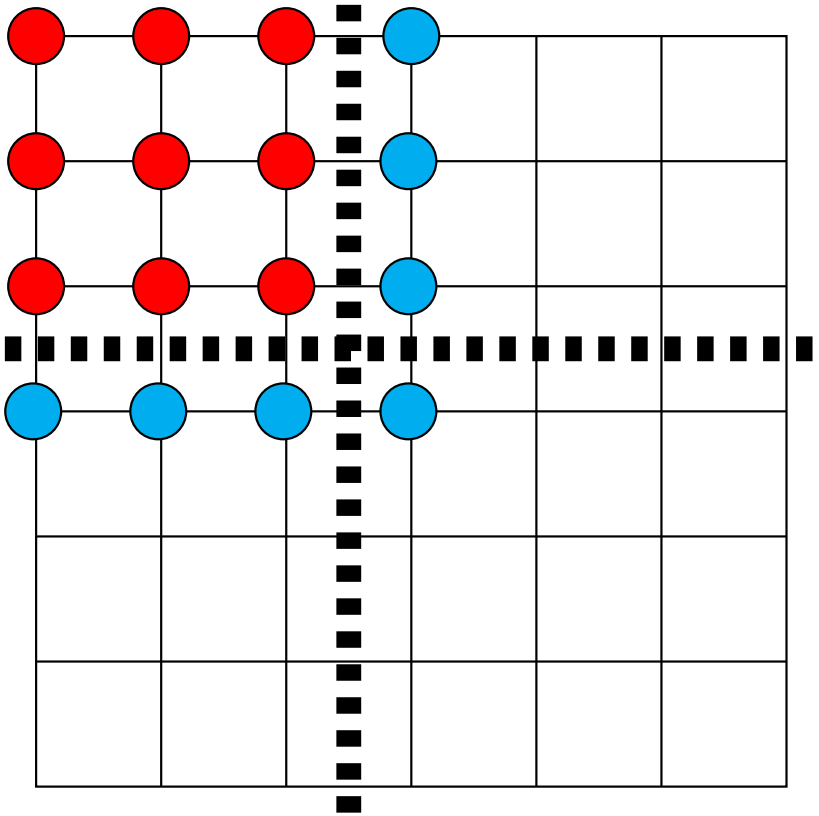
It is used as a template to create

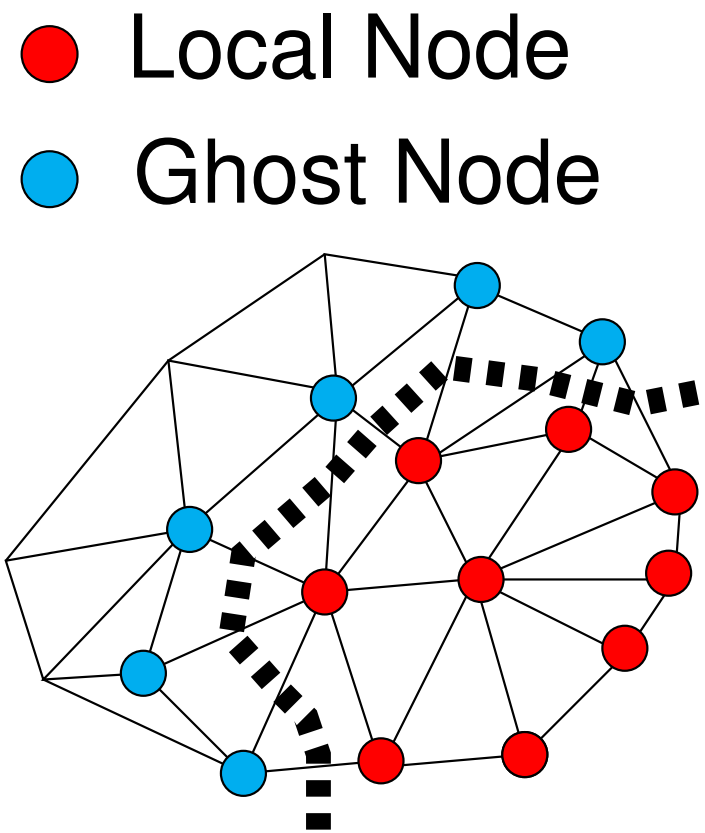
- Vectors (functions)
- Matrices (linear operators)

Ghost Values

To evaluate a local function $f(x)$, each process requires

- its local portion of the vector x
- its ghost values, bordering portions of x owned by neighboring processes





DMDA Local Function

User provided function calculates the nonlinear residual (in 2D)

```
(* lf )(DMDALocalInfo *info, PetscScalar**x, PetscScalar **r, void *ctx)
```

`info`: All layout and numbering information

`x`: The current solution (a multidimensional array)

`r`: The residual

`ctx`: The user context passed to `DMDASNESSetFunctionLocal()`

The local DMDA function is activated by calling

```
DMDASNESSetFunctionLocal(dm, INSERT_VALUES, lfunc, &ctx)
```

Bratu Residual Evaluation

$$\Delta u + \lambda e^u = 0$$

```
ResLocal(DMDALocalInfo *info, PetscScalar **x, PetscScalar **f, void *ctx)
for(j = info->ys; j < info->ys+info->ym; ++j) {
  for(i = info->xs; i < info->xs+info->xm; ++i) {
    u = x[j][i];
    if (i==0 || j==0 || i == M || j == N) {
      f[j][i] = 2.0*(hydhx+hxdhy)*u; continue;
    }
    u_xx     = (2.0*u - x[j][i-1] - x[j][i+1])*hydhx;
    u_yy     = (2.0*u - x[j-1][i] - x[j+1][i])*hxdhy;
    f[j][i] = u_xx + u_yy - hx*hy*lambda*exp(u);
}}}
```

$PETSC_DIR/src/snes/examples/tutorials/ex5.c

DMDA Local Jacobian

User provided function calculates the Jacobian (in 2D)

```
(* ljac )(DMDALocalInfo *info, PetscScalar**x, Mat J, void *ctx)
```

`info`: All layout and numbering information

`x`: The current solution

`J`: The Jacobian

`ctx`: The user context passed to DASetLocalJacobian()

The local DMDA function is activated by calling

```
DMDASNESSetJacobianLocal(dm, ljac, &ctx)
```

Updating Ghosts

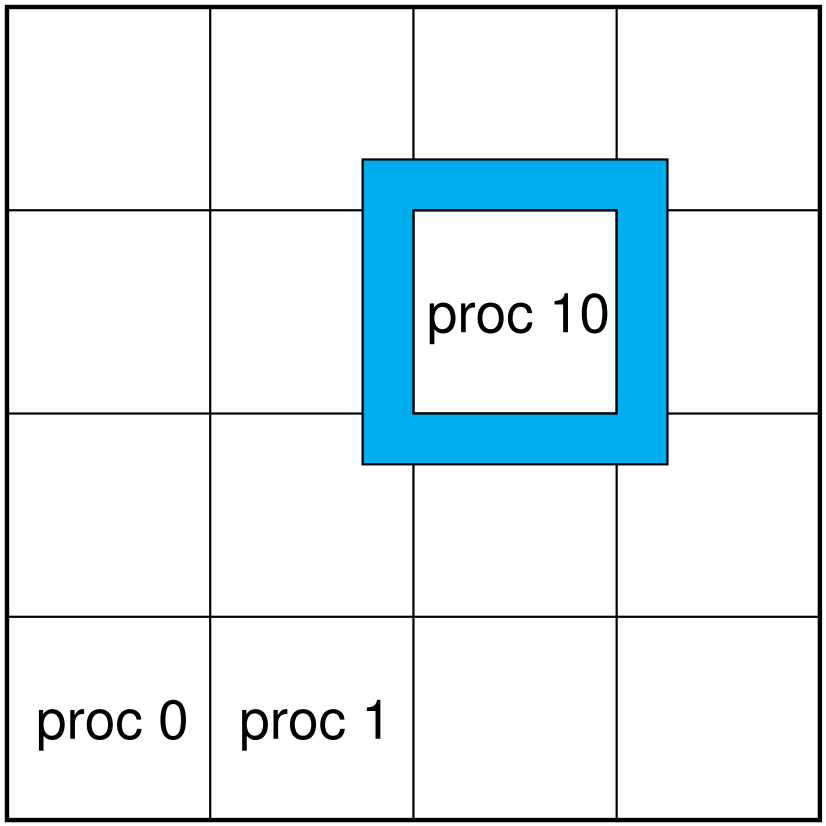
Two-step process enables overlapping computation and communication

- DMGlobalToLocalBegin(da, gvec, mode, lvec)
  - `gvec` provides the data
  - `mode` is either `INSERT_VALUES` or `ADD_VALUES`
  - `lvec` holds the local and ghost values
- DMGlobalToLocalEnd(da, gvec, mode, lvec)
  - Finishes the communication

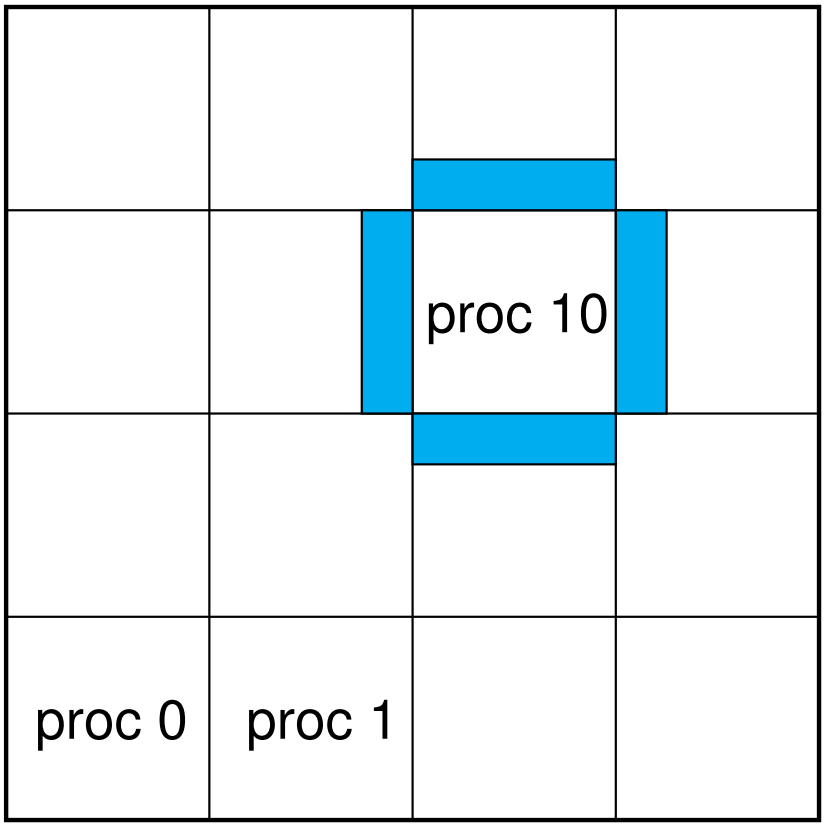
The process can be reversed with DALocalToGlobalBegin/End().

DMDA Stencils

Both the box stencil and star stencil are available.



Box Stencil



Star Stencil

DM Integration with SNES

- DM supplies global residual and Jacobian to SNES
  - User supplies local version to DM
  - The `Rhs_*()` and `Jac_*()` functions in the example
- Allows automatic parallelism
- Allows grid hierarchy
  - Enables multigrid once interpolation/restriction is defined
- Paradigm is developed in unstructured work
  - Solve needs scatter into contiguous global vectors (initial guess)
- Handle Neumann BC using `KSPSetNullSpace()`

Outline

Global and Local

Local (analytical)

- Discretization/Approximation
  - FEM integrals
  - FV fluxes
- Boundary conditions
- Largely dim dependent
  (e.g. quadrature)

Global (topological)

- Data management
  - Sections (local pieces)
  - Completions (assembly)
- Boundary definition
- Multiple meshes
  - Mesh hierarchies
- Largely dim independent
  (e.g. mesh traversal)

Global and Local

Local (analytical)

- Discretization/Approximation
  - FEM integrals
  - FV fluxes
- Boundary conditions
- Largely dim dependent (e.g. quadrature)

Global (topological)

- Data management
  - Sections (local pieces)
  - Completions (assembly)
- Boundary definition
- Multiple meshes
  - Mesh hierarchies
- Largely dim independent (e.g. mesh traversal)

Global and Local

Local (analytical)

- Discretization/Approximation
  - FEM integrals
  - FV fluxes
- Boundary conditions
- Largely dim dependent (e.g. quadrature)

Global (topological)

- Data management
  - Sections (local pieces)
  - Completions (assembly)
- Boundary definition
- Multiple meshes
  - Mesh hierarchies
- Largely dim independent (e.g. mesh traversal)

Global and Local

Local (analytical)

- Discretization/Approximation
  - FEM integrals
  - FV fluxes
- Boundary conditions
- Largely dim dependent (e.g. quadrature)

Global (topological)

- Data management
  - Sections (local pieces)
  - Completions (assembly)
- Boundary definition
- Multiple meshes
  - Mesh hierarchies
- Largely dim independent (e.g. mesh traversal)

Global and Local

Local (analytical)

- Discretization/Approximation
  - FEM integrals
  - FV fluxes
- Boundary conditions
- Largely dim dependent (e.g. quadrature)

Global (topological)

- Data management
  - Sections (local pieces)
  - Completions (assembly)
- Boundary definition
- Multiple meshes
  - Mesh hierarchies
- Largely dim independent (e.g. mesh traversal)

AMG

Why not use AMG?

- Of course we will try AMG
  - GAMG, `-pc_type gamg`
  - ML, `-download-ml`, `-pc_type ml`
  - BoomerAMG, `-download-hypre`, `-pc_type hypre -pc_hypre_type boomeramg`
- Problems with
  - vector character
  - anisotropy
  - scalability of setup time

AMG

Why not use AMG?

- Of course we will try AMG
  - GAMG, `-pc_type gamg`
  - ML, `-download-ml`, `-pc_type ml`
  - BoomerAMG, `-download-hypre`, `-pc_type hypre -pc_hypre_type boomeramg`
- Problems with
  - vector character
  - anisotropy
  - scalability of setup time

AMG

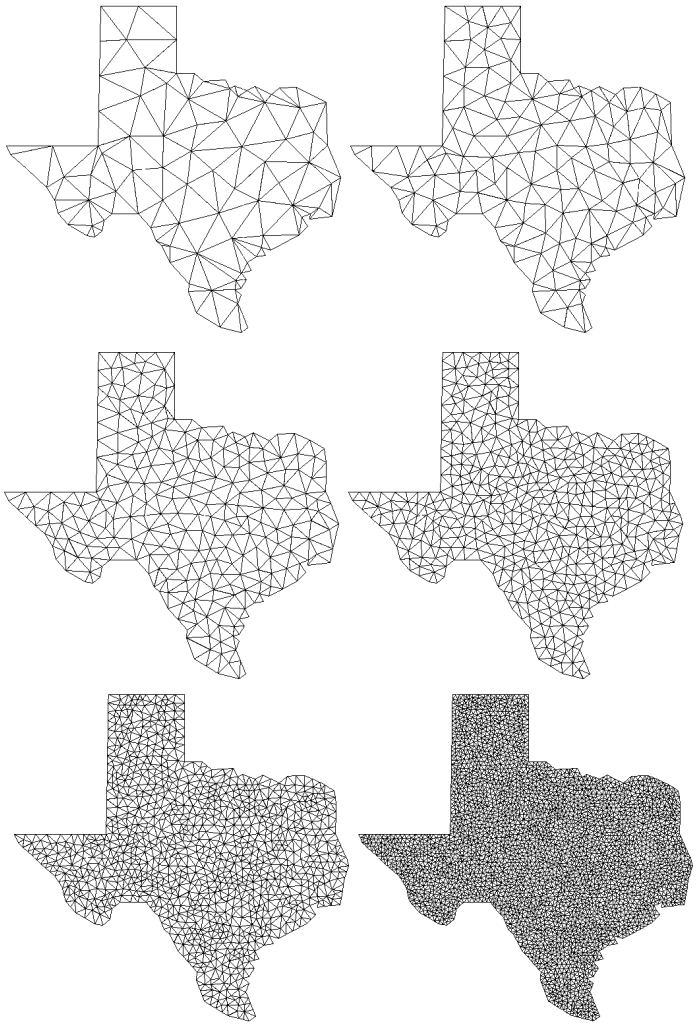
Why not use AMG?

- Of course we will try AMG
  - GAMG, `-pc_type gamg`
  - ML, `-download-ml`, `-pc_type ml`
  - BoomerAMG, `-download-hypre`, `-pc_type hypre -pc_hypre_type boomeramg`
- Problems with
  - vector character
  - anisotropy
  - scalability of setup time

Unstructured Meshes

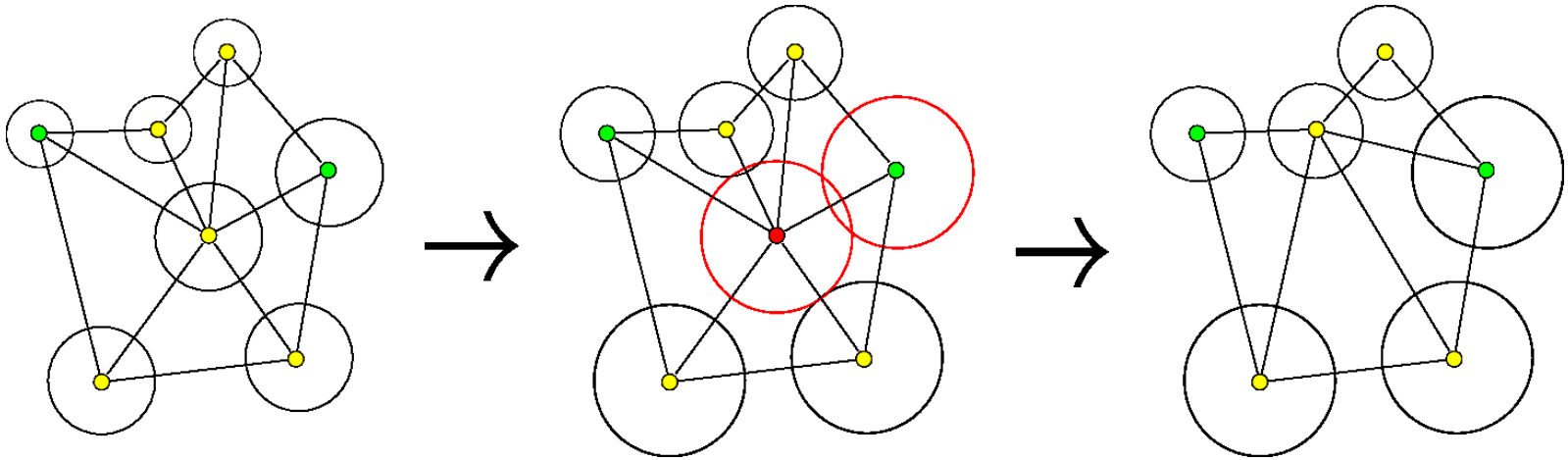
- Same `DMMG` options as the structured case
- Mesh refinement
  - Ruppert algorithm in Triangle and TetGen
- Mesh coarsening
  - Talmor-Miller algorithm in PETSc
- More advanced options
  - `-dmmg_refine`
  - `-dmmg_hierarchy`
- Current version only works for linear elements

Coarsening



- Users want to control the mesh
- Developed efficient, topological coarsening
  - Miller, Talmor, Teng algorithm
- Provably well-shaped hierarchy

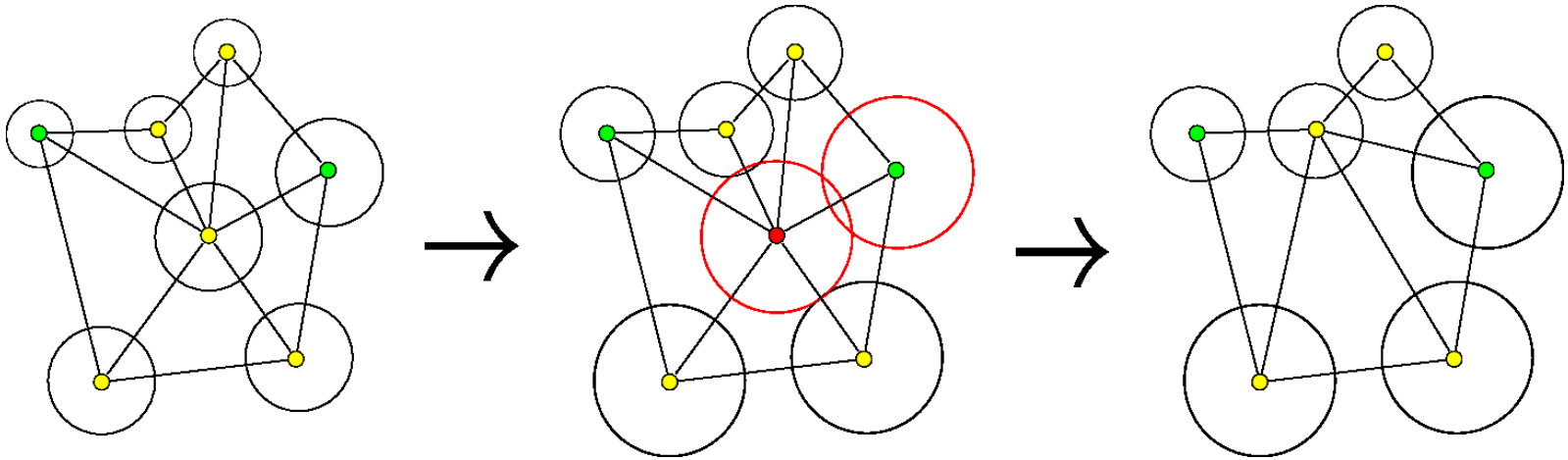
Miller-Talmor-Teng Algorithm



Simple Coarsening

1. Compute a spacing function f for the mesh (Koebe)
2. Scale f by a factor $C > 1$
3. Choose a maximal independent set of vertices for new f
4. Retriangulate

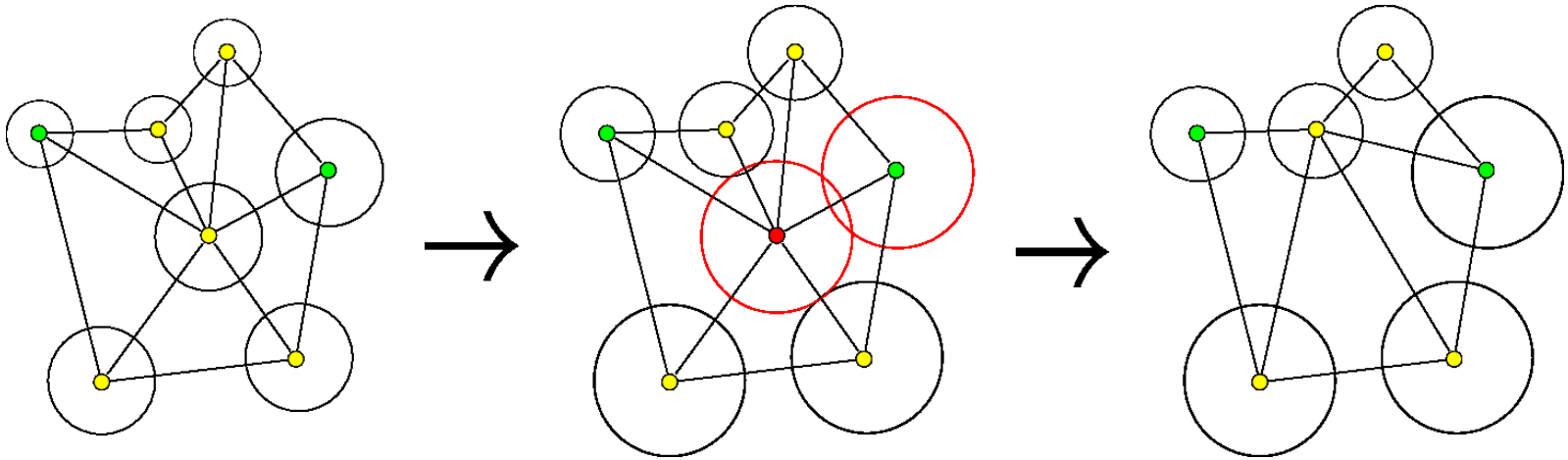
Miller-Talmor-Teng Algorithm



Simple Coarsening

1. Compute a spacing function f for the mesh (Koebe)
2. Scale f by a factor $C > 1$
3. Choose a maximal independent set of vertices for new f
4. Retriangulate

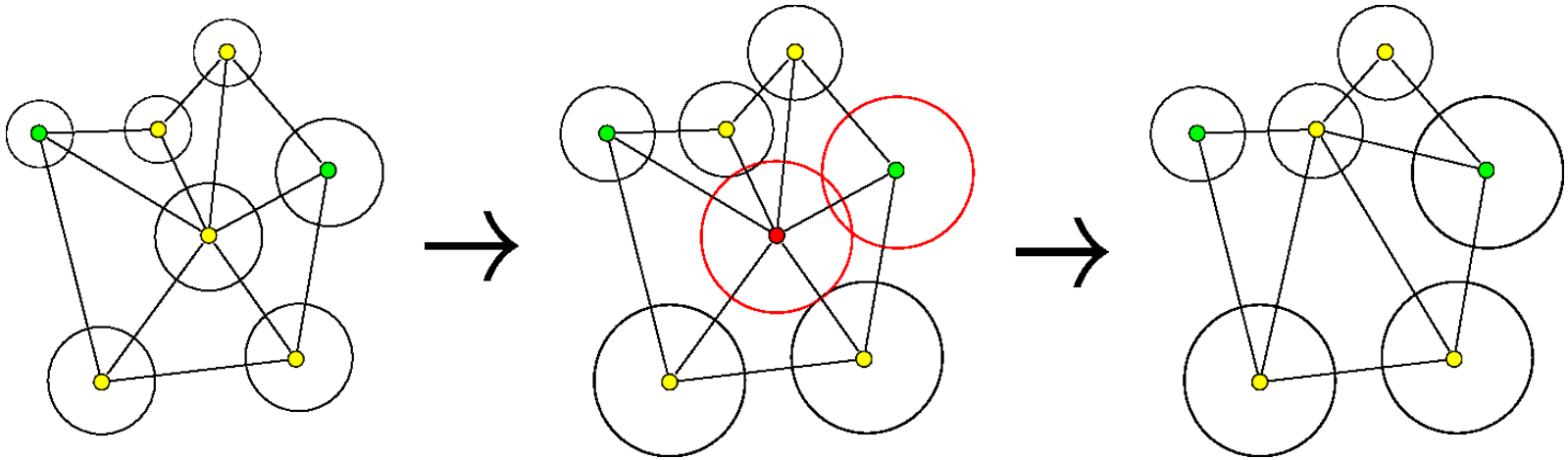
Miller-Talmor-Teng Algorithm



Simple Coarsening

1. Compute a spacing function f for the mesh (Koebe)
2. Scale f by a factor $C > 1$
3. Choose a maximal independent set of vertices for new f
4. Retriangulate

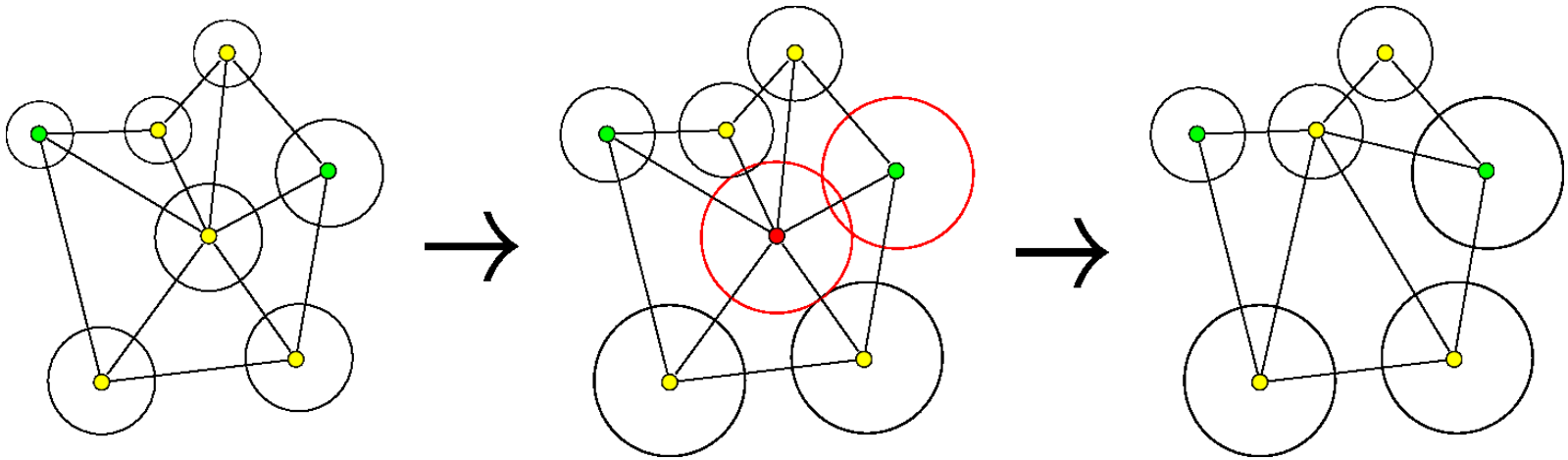
Miller-Talmor-Teng Algorithm



Simple Coarsening

1. Compute a spacing function f for the mesh (Koebe)
2. Scale f by a factor $C > 1$
3. Choose a maximal independent set of vertices for new f
4. Retriangulate

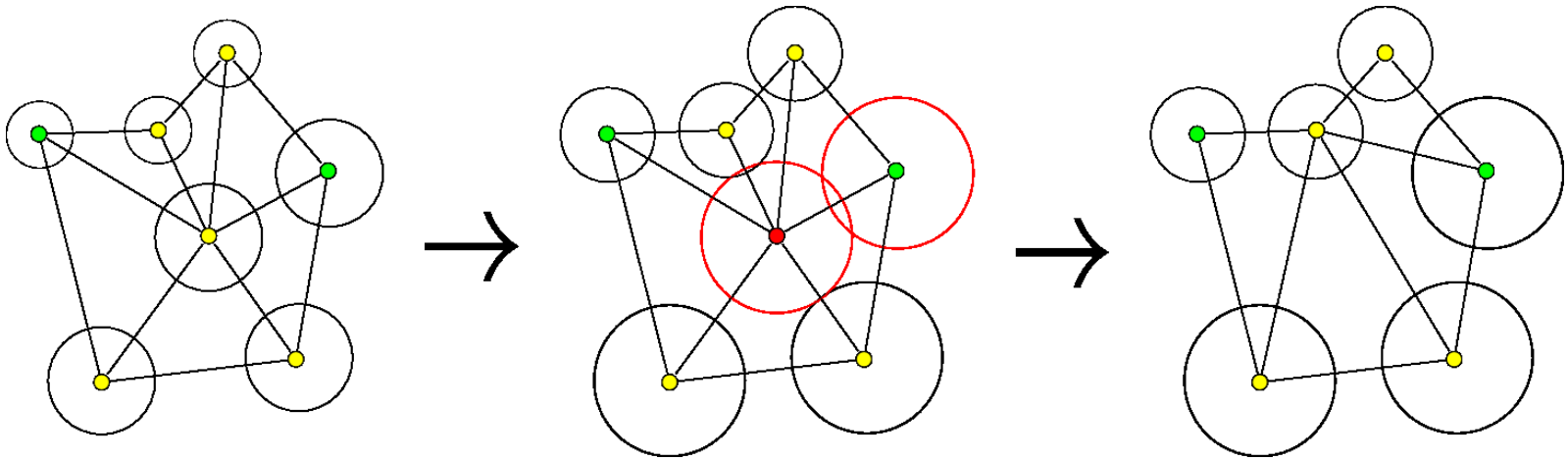
Miller-Talmor-Teng Algorithm



Caveats

1. Must generate coarsest grid in hierarchy first
2. Must choose boundary vertices first (and protect boundary)
3. Must account for boundary geometry

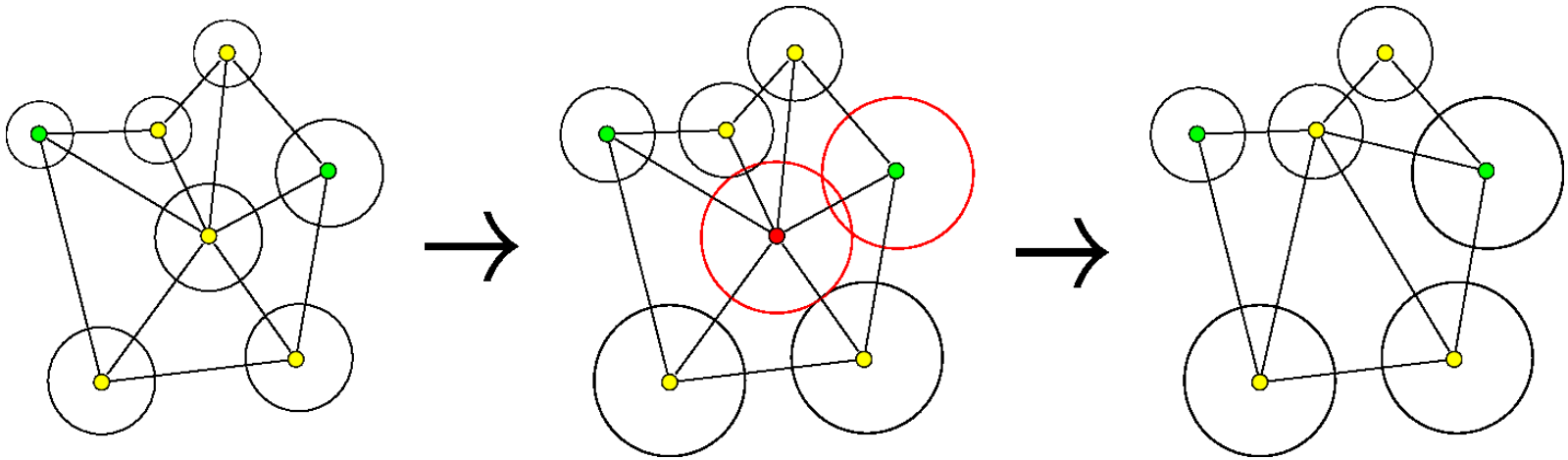
Miller-Talmor-Teng Algorithm



Caveats

1. Must generate coarsest grid in hierarchy first
2. Must choose boundary vertices first (and protect boundary)
3. Must account for boundary geometry

Miller-Talmor-Teng Algorithm

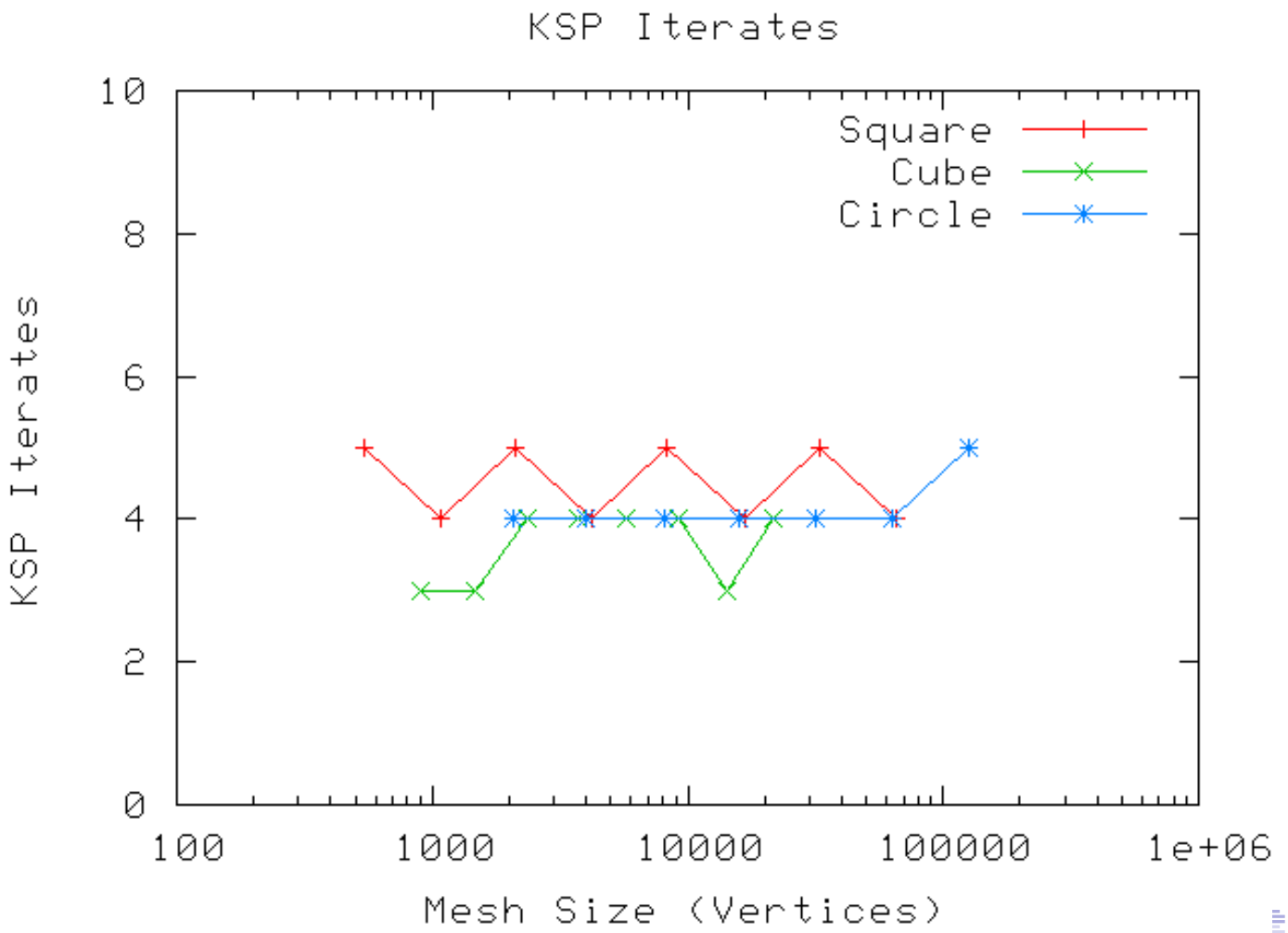


Caveats

1. Must generate coarsest grid in hierarchy first
2. Must choose boundary vertices first (and protect boundary)
3. Must account for boundary geometry

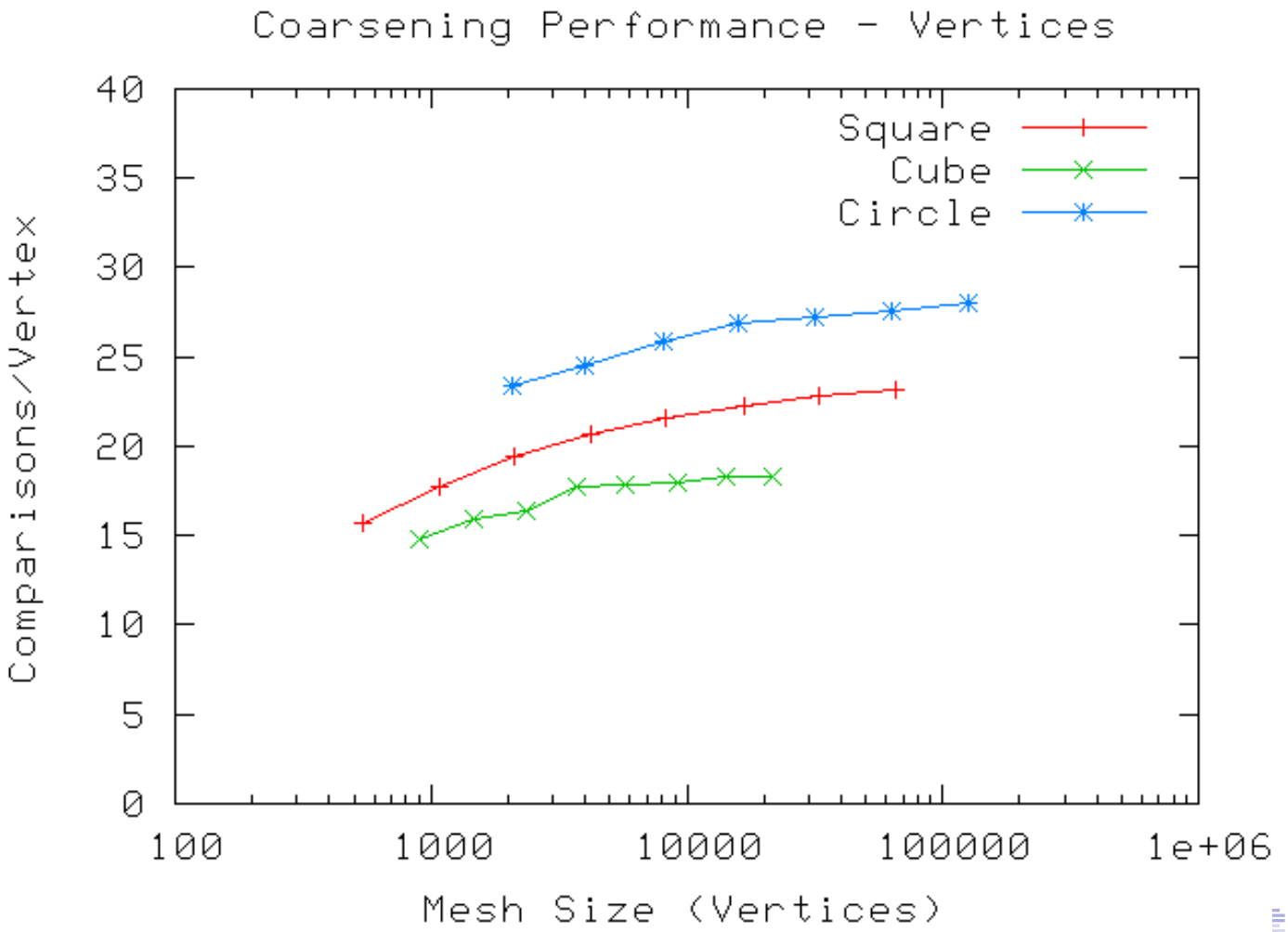
GMG Performance

For simple domains, everything works as expected:
Linear solver iterates are constant as system size increases:



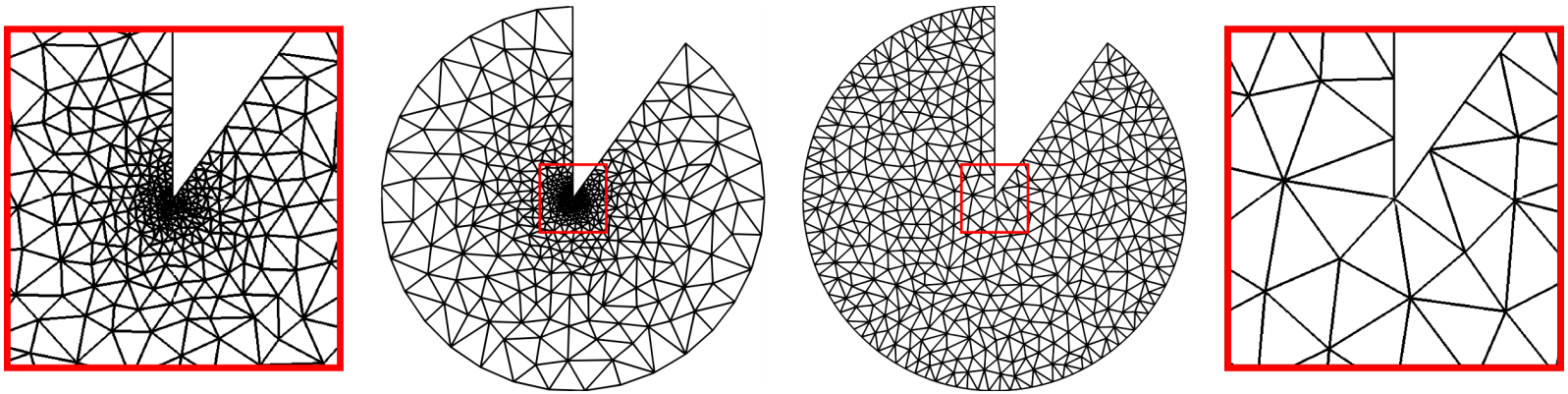
GMG Performance

For simple domains, everything works as expected:
Work to build the preconditioner is constant as system size increases:



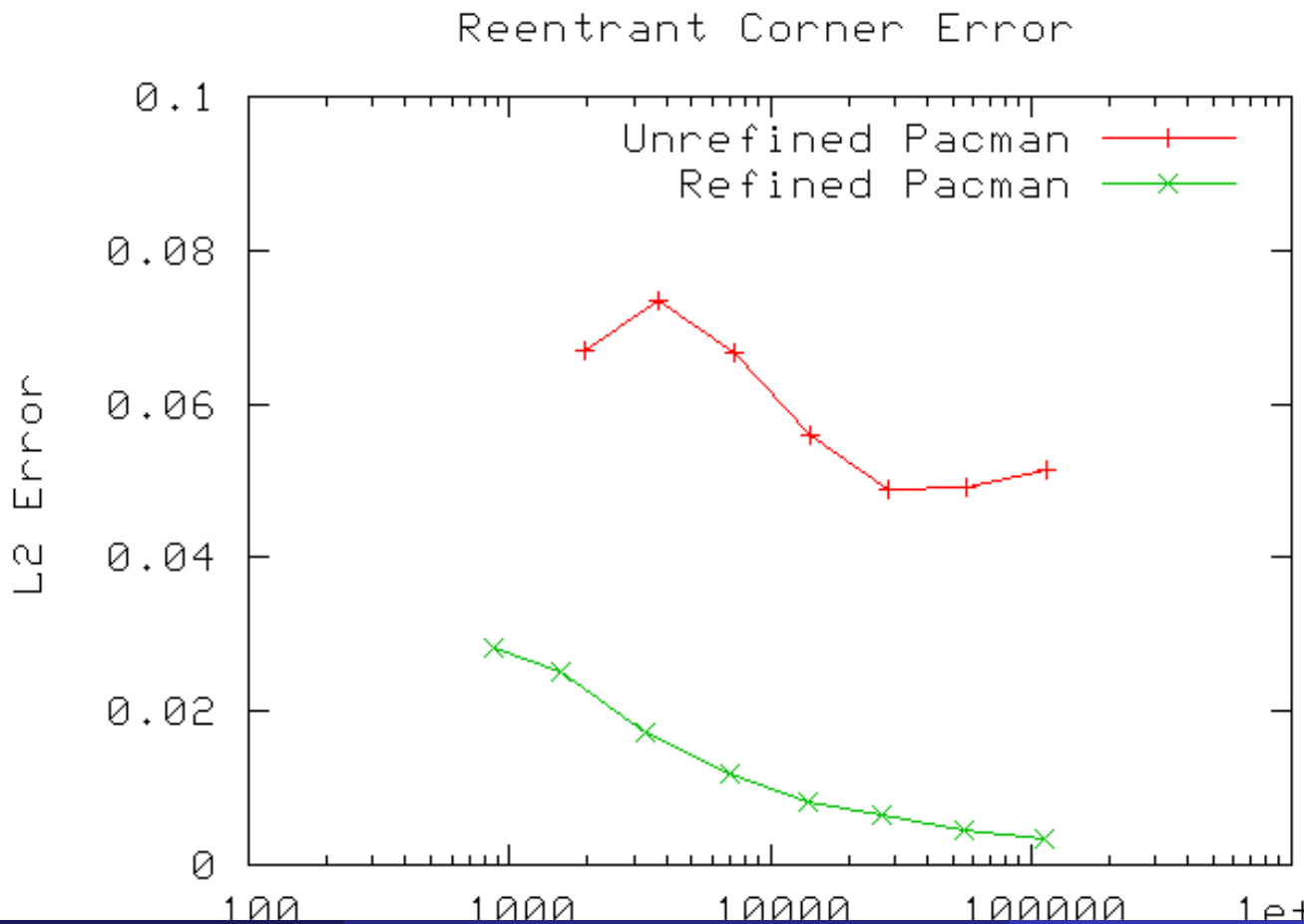
Reentrant Problems

- Reentrant corners need nonnuiform refinement to maintain accuracy
- Coarsening preserves accuracy in MG without user intervention



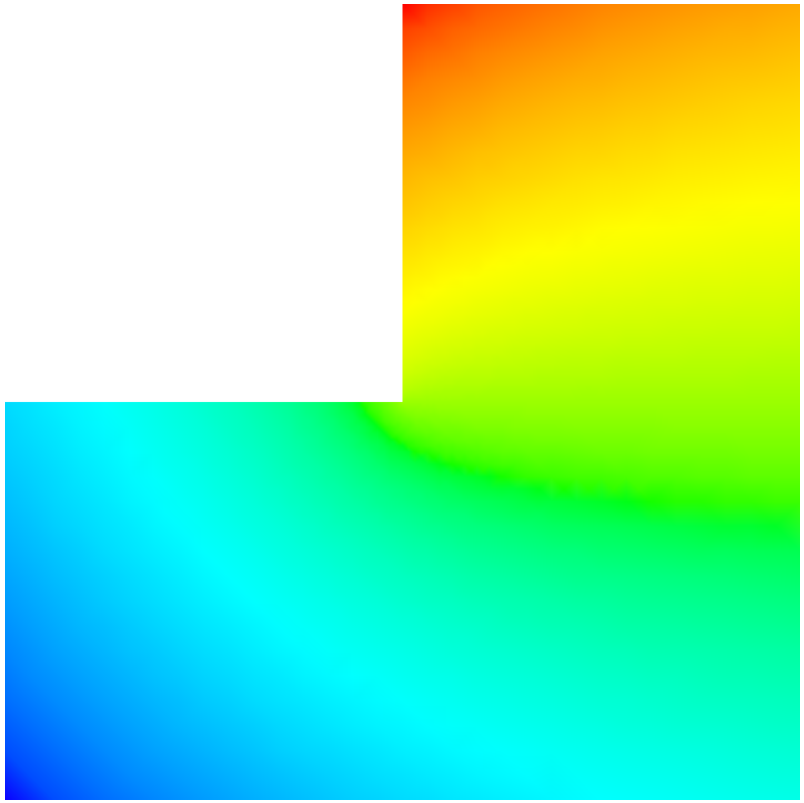
Reentrant Problems

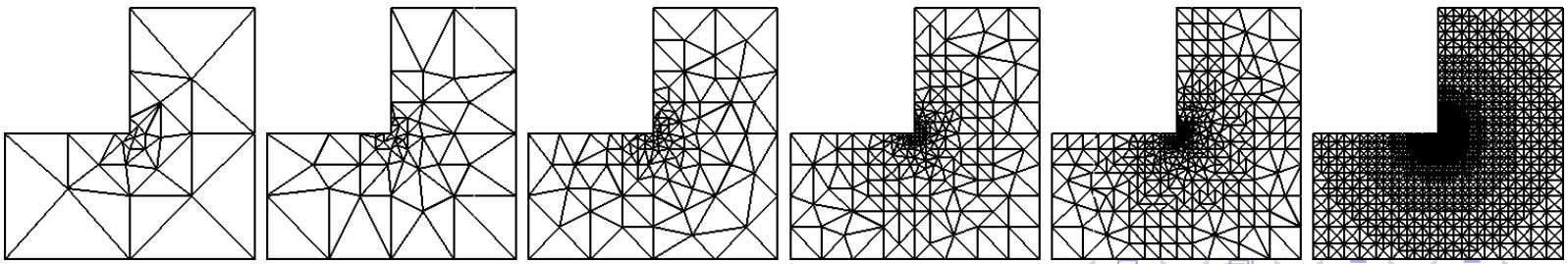
- Reentrant corners need nonnuiform refinement to maintain accuracy
- Coarsening preserves accuracy in MG without user intervention



Reentrant Problems

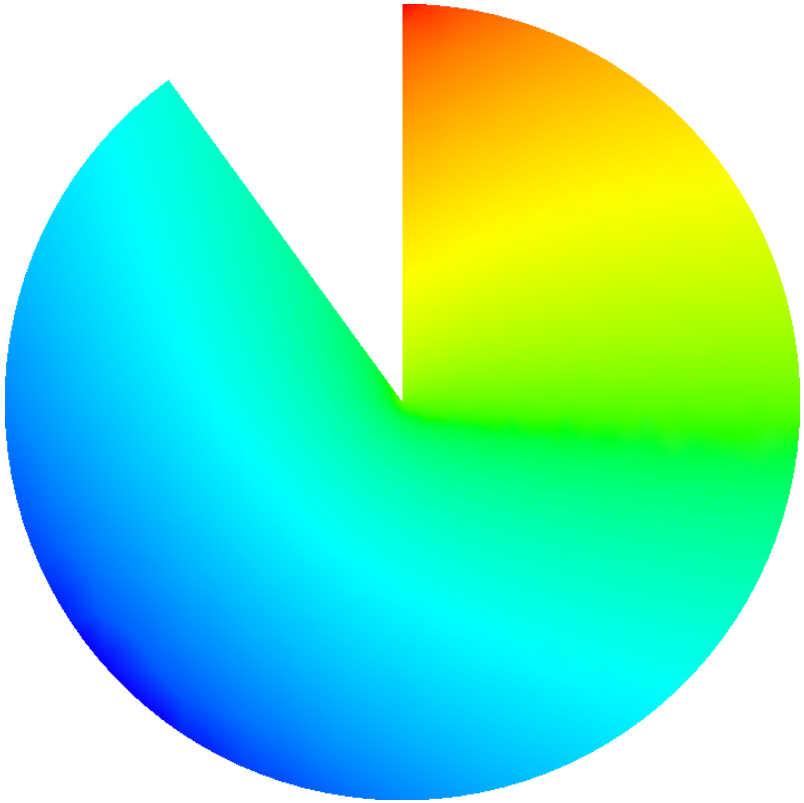
Exact Solution for reentrant problem: $u(x, y) = r^{\frac{2}{3}} sin(\frac{2}{3}\theta)$

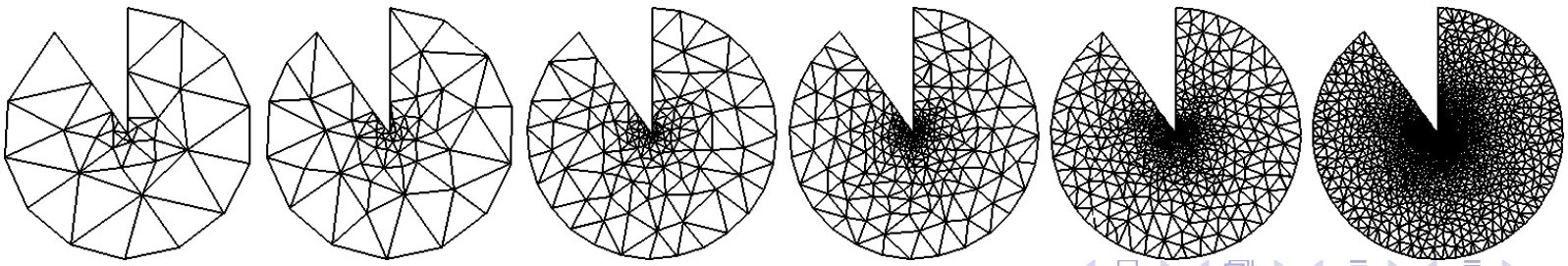




Reentrant Problems

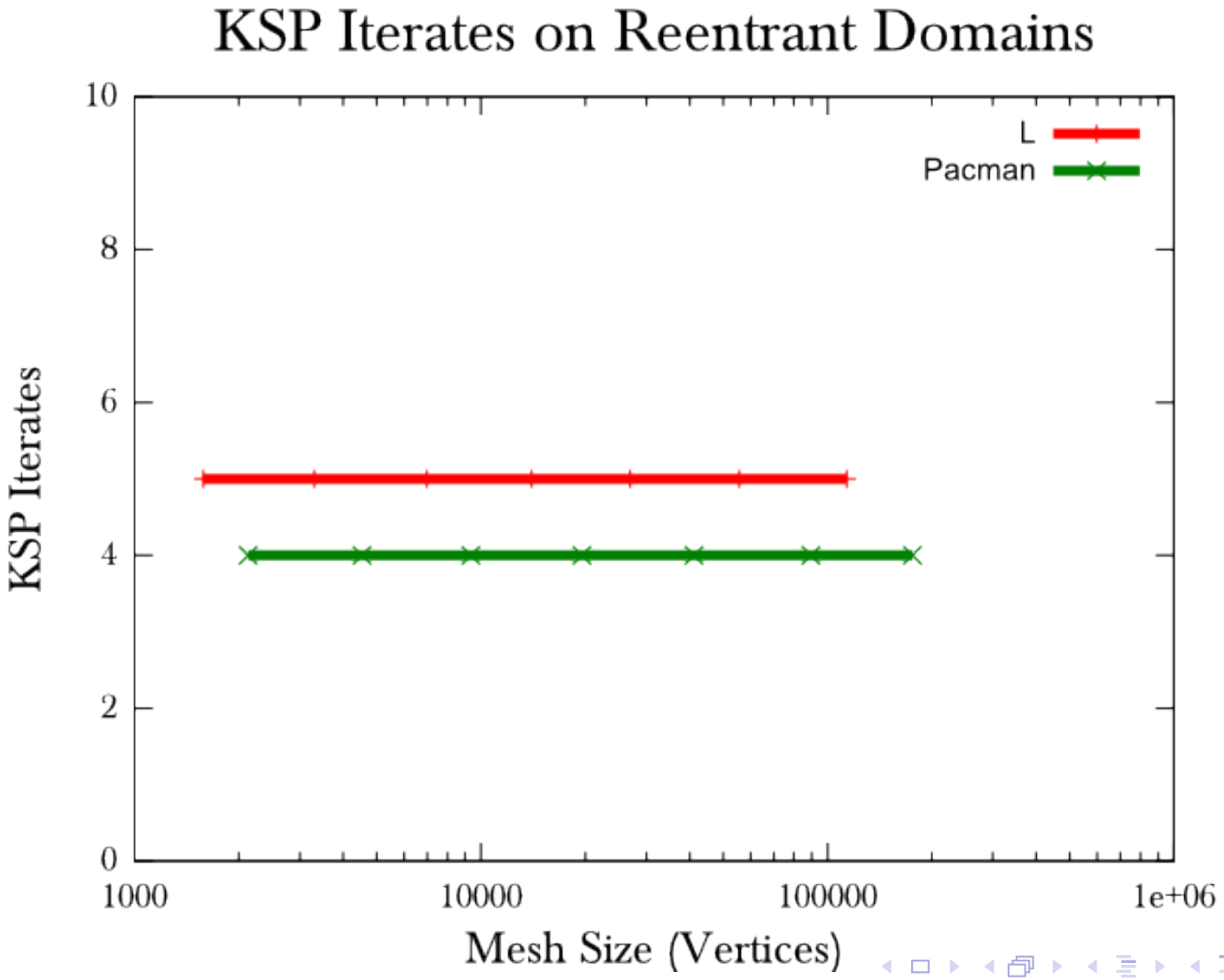
Exact Solution for reentrant problem: $u(x, y) = r^{\frac{2}{3}} sin(\frac{2}{3}\theta)$





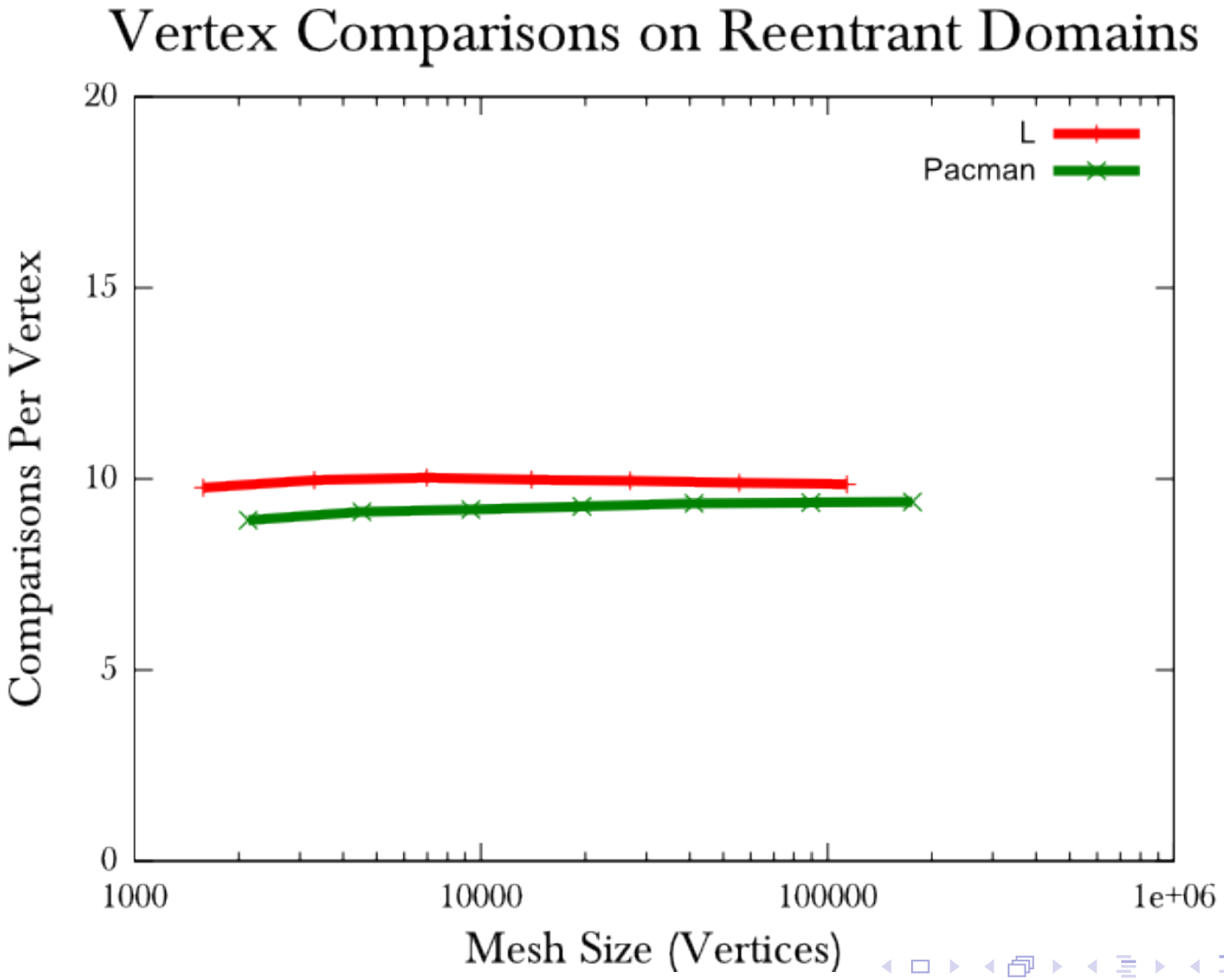
GMG Performance

Linear solver iterates are constant as system size increases:



GMG Performance

Work to build the preconditioner is constant as system size increases:



References

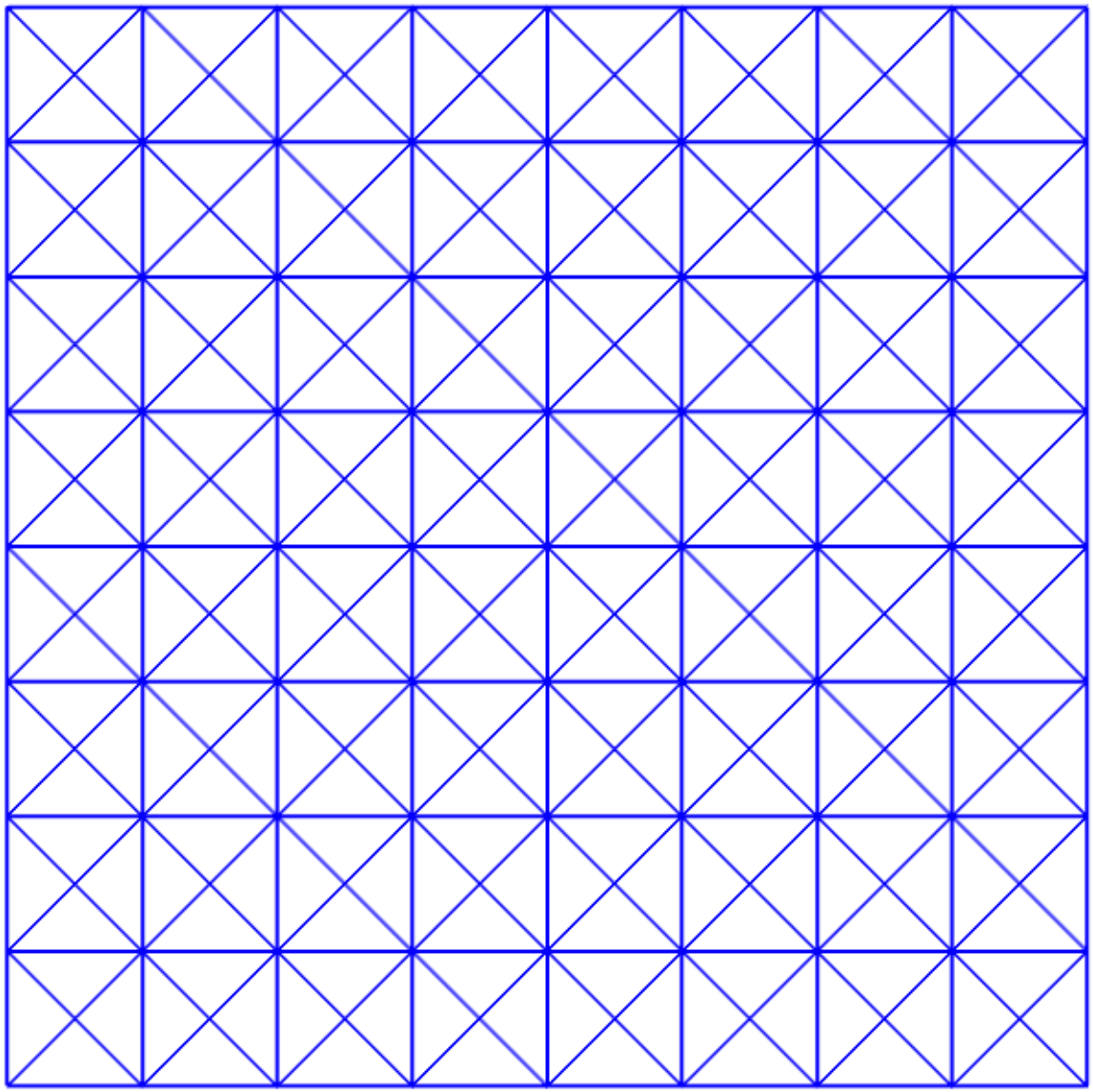
- Documentation: http://www.mcs.anl.gov/petsc/docs
  - PETSc Users manual
  - Manual pages
  - Many hyperlinked examples
  - FAQ, Troubleshooting info, installation info, etc.
- Publications: http://www.mcs.anl.gov/petsc/publications
  - Research and publications that make use PETSc
- MPI Information: http://www.mpi-forum.org
- **Using MPI** (2nd Edition), by Gropp, Lusk, and Skjellum
- **Domain Decomposition**, by Smith, Bjorstad, and Gropp

Experimentation is Essential!

Proof is not currently enough to examine solvers

- N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., **13**, pp.778–795, 1992.
- Anne Greenbaum, Vlastimil Ptak, and Zdenek Strakos, *Any Nonincreasing Convergence Curve is Possible for GMRES*, SIAM J. Matrix Anal. Appl., **17** (3), pp.465–469, 1996.

Problem Domain



The Stokes Problem – Strong Form

$$
\begin{aligned}
-\Delta u + \nabla p &= f \\
\nabla \cdot u &= 0 \\
u|_{\partial\Omega} &= g \\
\int_{\Omega} p &= 0
\end{aligned}
$$

The Stokes Problem – Weak Form

For $u, v \in V$ and $p, q \in \Pi$

$$< \nabla v, \nabla u > - < \nabla \cdot v, p > = < v, f >$$
$$< q, \nabla \cdot u > = 0$$
$$u|_{\partial\Omega} = g$$
$$\int_{\Omega} p = 0$$

Continuity

For all $u, v \in V$ and $p \in \Pi$ we have

$$< \nabla v, \nabla u > \leq C_a ||u||_V ||v||_V \quad (1)$$

$$< \nabla \cdot v, p > \leq C_b ||v||_V ||p||_\Pi \quad (2)$$

Coercivity

For all $v \in Z \cup Z_h$ and $p \in \Pi_h$ we have

$$< \nabla v, \nabla v > \geq \alpha ||v||_V^2 \tag{3}$$

$$\sup_{u \in V_h} \frac{< \nabla \cdot u, p >}{||u||_V} \geq \beta ||p||_\Pi \tag{4}$$

Coercivity

For all $v \in Z \cup Z_h$ and $p \in \Pi_h$ we have

$$\sup_{u \in V_h} \frac{< \nabla v, \nabla u >}{||u||_V} \geq \alpha ||v||_V \tag{3}$$

$$\sup_{u \in V_h} \frac{< \nabla \cdot u, p >}{||u||_V} \geq \beta ||p||_\Pi \tag{4}$$

Iterated Penalty Formulation

Introduce a penalty term and solve iteratively for u^n and p^n,

$$< \nabla v, \nabla u^n > + r < \nabla \cdot v, \nabla \cdot u^n > = < v, f > - < \nabla \cdot v, p^n > \quad (5)$$

$$p^{n+1} = p^n + \rho \nabla \cdot u^n \quad (6)$$

Notice that eqn. 5 will be symmetric and coercive if $r > 0$.

Iterated Penalty Formulation

Introduce a penalty term and solve iteratively for u^n and w^n,

$$< \nabla v, \nabla u^n > + r < \nabla \cdot v, \nabla \cdot u^n > = < v, f > - < \nabla \cdot v, \nabla \cdot w^n > \quad (5)$$

$$w^{n+1} = w^n + \rho u^n \quad (6)$$

Notice that eqn. 5 will be symmetric and coercive if $r > 0$.

Iterated Penalty Formulation

Introduce a penalty term and solve iteratively for u^n and w^n,

$$< \nabla v, \nabla u^n > + r < \nabla \cdot v, \nabla \cdot u^n > = < v, f > - < \nabla \cdot v, \nabla \cdot w^n > \quad (5)$$

$$w^{n+1} = w^n + \rho u^n \quad (6)$$

Notice that eqn. 5 will be symmetric and coercive if $r > 0$.

Pressure FE Space

We assume that

$$\Pi_h = \mathcal{D} V_h \tag{7}$$

meaning $\mathcal{D}$ has a right-inverse L

$$\mathcal{D}(Lq) = q \qquad \forall q \in \Pi_h$$

such that

$$||Lq||_V \leq \frac{1}{\beta}||q||_\Pi$$

Error Estimates

The Iterated Penalty Method (IP) converges for sufficiently large r and $0 < -\rho < 2r$. For the case $r = -\rho$, the convergence rate ρ_{IP} is

$$\rho_{IP} = \frac{C_a \left(\frac{1}{\beta} + \frac{C_a}{\alpha\beta}\right)^2}{r} \tag{8}$$

and we have error estimates

$$||u^n - u_h||_V \leq \left(\frac{1}{\beta} + \frac{C_a}{\alpha\beta}\right) ||\mathcal{D}u^n||_\Pi \tag{9}$$

$$||p^n - p_h||_\Pi \leq \left(\frac{C_a}{\beta} + \frac{C_a^2}{\alpha\beta} + rC_b\right) ||\mathcal{D}u^n||_\Pi \tag{10}$$

which provide a stopping criteria.

Advantages

- Single FE Space
  - Easy layout
  - No compatiblity condition for spaces
  - No saddle point
  - Seems ideal for Multigrid
- Control of the divergence residual
  - On a fixed mesh, can drive $||\nabla \cdot u|| \to 0$
- Simple characterization of pressure space

$$\Pi_h = \mathcal{D} V_h$$

Problems

- What is the condition of the IP system?

$$< \nabla, \nabla > + < \nabla\cdot, \nabla\cdot >$$

- Can we use simple, local interpolation?
  - We only have P_1 interpolation at present
- Is it stable?
  - We can prove stability for P_k, $k > 3$
  - Tests with quadratic elements work
- Can I use exotic elements?
- Can I estimate the convergence parameters?

Condition of the Laplacian

2D P_1 Lagrange Elements

Num. Elements	Longest edge (h)	κ
64	1/4	12.6
128	$\sqrt{2}/8$	25.2
256	1/8	51.5
512	$\sqrt{2}/16$	103.1
256	1/16	207.2
1024	$\sqrt{2}/32$	414.3
2048	1/32	829.7
4096	$\sqrt{2}/64$	1659.4
8192	1/64	3319.8

Table: 2D P_1 Laplacian Condition Number

so we have

$$\kappa \approx 0.8h^{-2} \tag{11}$$

Condition of the Laplacian

2D P_2 Lagrange Elements

Num. Elements	Longest edge (h)	κ
64	1/4	68.1
128	$\sqrt{2}/8$	137.2
256	1/8	275.6
512	$\sqrt{2}/16$	552.2
256	1/16	1105.6
1024	$\sqrt{2}/32$	2212.3
2048	1/32	4425.7
4096	$\sqrt{2}/64$	8852.6
8192	1/64	17708.1

Table: 2D P_2 Laplacian Condition Number

so we have

$$\kappa \approx 4.3h^{-2} \tag{12}$$

Condition of the IP Operator

Num. Elements	Longest edge (h)	min κ	max κ
8	$\sqrt{2}/2$	18.5	18.5
16	1/2	47.3	2497.8
32	$\sqrt{2}/4$	120.5	3624.6
64	1/4	2589.7	2593.0
128	$\sqrt{2}/8$	3461.5	3573.1
256	1/8	2610.4	2619.6

Table: 2D P_2 IP Operator Condition Number, $r = 10^3$

Condition of the IP Operator

Num. Elements	Longest edge (h)	min κ	max κ
8	$\sqrt{2}/2$	2.5	2.6
16	1/2	195.5	203.5
32	$\sqrt{2}/4$	428.5	435.7
64	1/4	400.4	404.8
128	$\sqrt{2}/8$	839.4	841.9
256	1/8	566.5	578.8

Table: 2D P_2 IP Operator with SOR(2) Condition Number, $r = 10^3$

Problem Solution

We use Dirichlet conditions from an exact solution

$$\begin{aligned} u &= x^2 - 2xy \\ v &= y^2 - 2xy \end{aligned}$$