Geodynamic Simulator Building

Matthew Knepley and Margarete Jadamec

Computer Science and Engineering & Geology
University at Buffalo

SIAM Parallel Processing,
Tokyo, Kantō JP March 10, 2018

Main Idea

A Simulator is more Useful

when the Researcher

Builds it Themselves

Outline

Main Question

How do I handle

many different mesh types

simply and efficiently?

Current Practice

Most packages handle one kind of mesh,

or have completely separate code paths

for different meshes

Current Practice

Most packages handle one kind of mesh,

or have completely separate code paths

for different meshes

Current Practice

This strategy means there is

a lot more code to maintain,

and results in technical debt.

Current Practice

This strategy means there is

a lot more code to maintain,

and results in technical debt.

PETSc Strategy

The `Plex` abstraction allows us to write code for

parallel distribution and load balancing,
traversal for function/operator assembly,
coarsening and refinement,
generation of missing edges/faces,
and surface extraction,

PETSc Strategy

The `Plex` abstraction allows us to write code for

- parallel distribution and load balancing,
- traversal for function/operator assembly,
- coarsening and refinement,
- generation of missing edges/faces,
- and surface extraction,

PETSc Strategy

The `Plex` abstraction allows us to write code for

parallel distribution and load balancing,
traversal for function/operator assembly,
coarsening and refinement,
generation of missing edges/faces,
and surface extraction,

PETSc Strategy

The `Plex` abstraction allows us to write code for

- parallel distribution and load balancing,
- traversal for function/operator assembly,
- coarsening and refinement,
- generation of missing edges/faces,
- and surface extraction,

PETSc Strategy

The `Plex` abstraction allows us to write code for

parallel distribution and load balancing,
traversal for function/operator assembly,
coarsening and refinement,
generation of missing edges/faces,
and surface extraction,

PETSc Strategy

The `Plex` abstraction allows us to write code for

> parallel distribution and load balancing,
> traversal for function/operator assembly,
> coarsening and refinement,
> generation of missing edges/faces,
> and surface extraction,

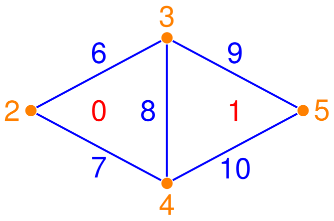
PETSc Strategy

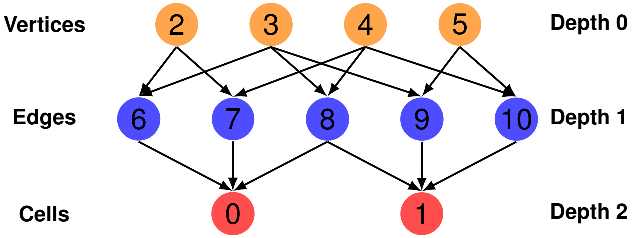
The `Plex` abstraction allows us to write code for

> parallel distribution and load balancing,
> traversal for function/operator assembly,
> coarsening and refinement,
> generation of missing edges/faces,
> and surface extraction,

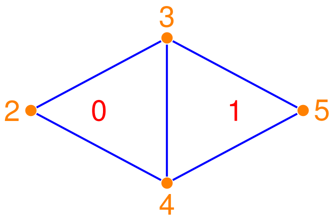
just **once**.

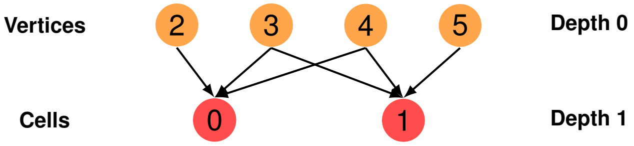
Sample Meshes
Interpolated triangular mesh



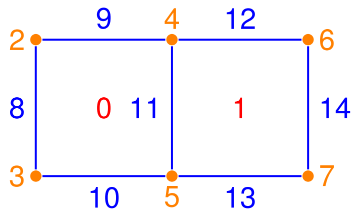


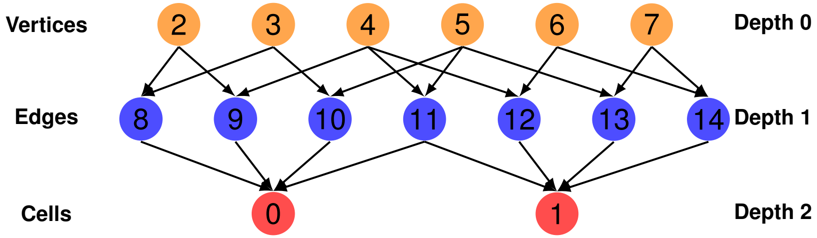
Sample Meshes
Optimized triangular mesh



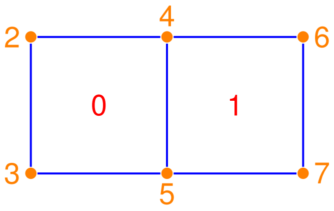


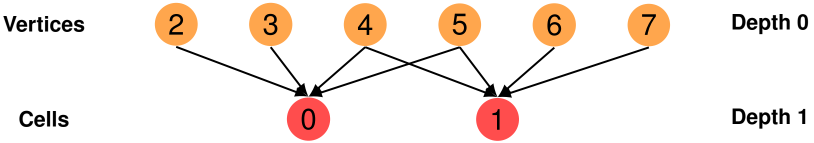
Sample Meshes
Interpolated quadrilateral mesh



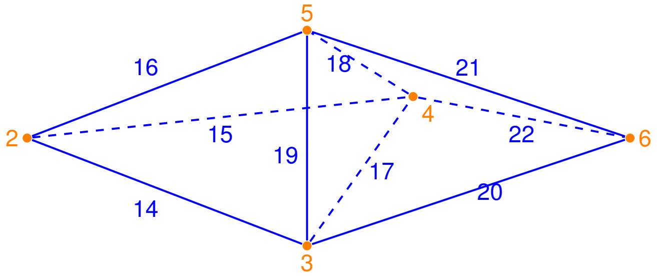


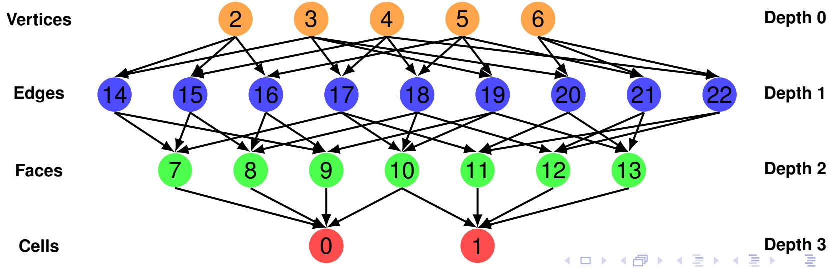
Sample Meshes
Optimized quadrilateral mesh





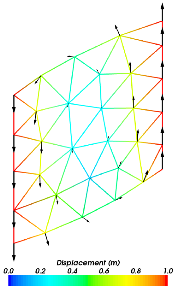
Sample Meshes
Interpolated tetrahedral mesh





Outline

Example: PyLith



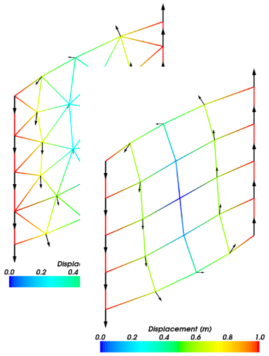
Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

Example: PyLith



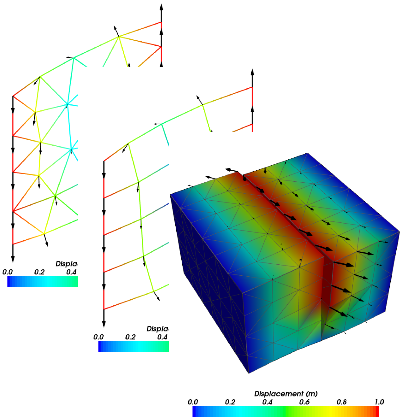
Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

Example: PyLith



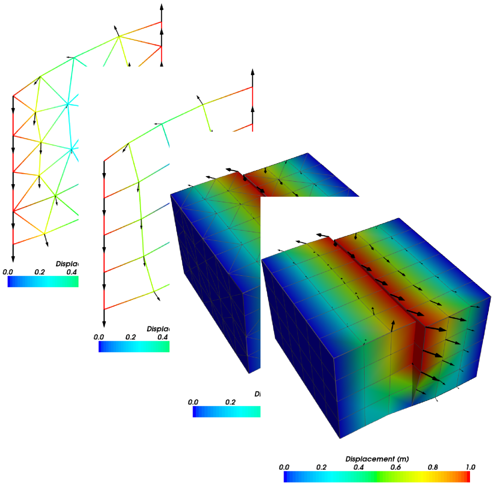
Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

Example: PyLith



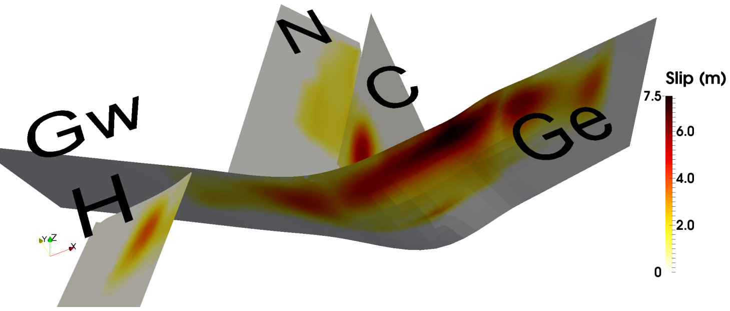
Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

Example: PyLith



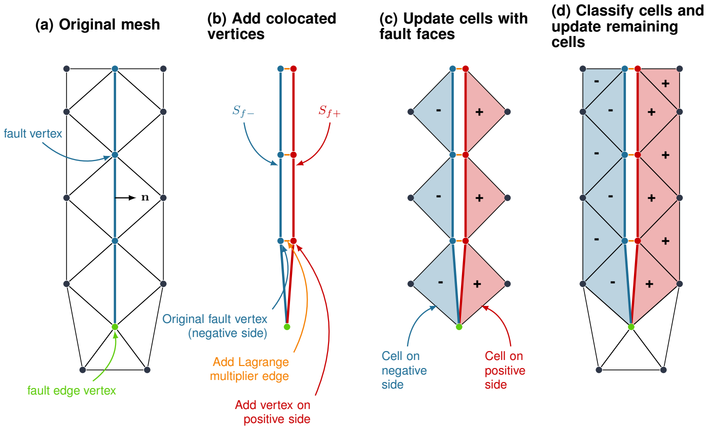
Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

Example: PyLith


(a) Original mesh
(b) Add colocated vertices
(c) Update cells with fault faces
(d) Classify cells and update remaining cells
fault vertex
n
fault edge vertex
S_{f-}
S_{f+}
Original fault vertex (negative side)
Add Lagrange multiplier edge
Add vertex on positive side
Cell on negative side
Cell on positive side
-
+

Many cell types

Surface extraction

Hybrid meshes

Aagaard, Knepley, Williams, J. of Geophysical Research, 2013.

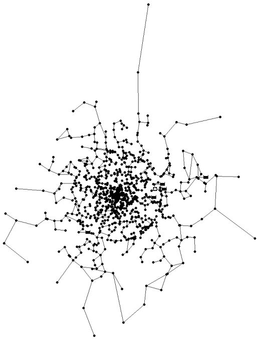
Outline

Example: DMNetwork

`Plex` on 30K cores of Edison for a finite volume hydraulic flow application.

Table IV. Execution Time of Transient State on Edison

No. of Cores	Variables (in millions)	Maximum Variables per Core (in thousands)	Linear Preconditioner: Block Jacobi	Linear Preconditioner: ASM ov. 1	Linear Preconditioner: ASM ov. 2
240	16	106	9.9 (48)	7.3 (25)	6.4 (20)
960	63	106	10.6 (55)	7.0 (24)	6.2 (20)
3,840	253	106	10.4 (53)	7.3 (24)	6.7 (20)
15,360	1,012	104	11.9 (53)	11.4 (26)	9.9 (20)
30,720	2,023	117	20.0 (53)	17.6 (26)	17.2 (20)



Maldonado, Abhyankar, Smith, Zhang, ACM TOMS, 2017

Outline

Main Question

How do I handle

many different discretizations

simply and efficiently?

Current Practice

Most packages handle one discretization,

FEniCS/Firedrake is a notable exception,

and interface poorly with solvers,

especially hierarchical solvers.

Current Practice

Most packages handle one discretization,

FEniCS/Firedrake is a notable exception,

and interface poorly with solvers,

especially hierarchical solvers.

Current Practice

Most packages handle one discretization,

FEniCS/Firedrake is a notable exception,

and interface poorly with solvers,

especially hierarchical solvers.

PETSc Strategy

The `Section` abstraction allows us to write code for

parallel data layout,

block/field decompositions,

(variable) point-block decompositions,

removing Dirichlet conditions,

(nonlinear) hierarchical rediscretization,

and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

parallel data layout,
block/field decompositions,
(variable) point-block decompositions,
removing Dirichlet conditions,
(nonlinear) hierarchical rediscretization,
and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

- parallel data layout,
- block/field decompositions,
- (variable) point-block decompositions,
- removing Dirichlet conditions,
- (nonlinear) hierarchical rediscretization,
- and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

parallel data layout,
block/field decompositions,
(variable) point-block decompositions,
removing Dirichlet conditions,
(nonlinear) hierarchical rediscretization,
and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

- parallel data layout,
- block/field decompositions,
- (variable) point-block decompositions,
- removing Dirichlet conditions,
- (nonlinear) hierarchical rediscretization,
- and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

parallel data layout,
block/field decompositions,
(variable) point-block decompositions,
removing Dirichlet conditions,
(nonlinear) hierarchical rediscretization,
and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

parallel data layout,
block/field decompositions,
(variable) point-block decompositions,
removing Dirichlet conditions,
(nonlinear) hierarchical rediscretization,
and partial assembly (BDDC/FETI),

PETSc Strategy

The `Section` abstraction allows us to write code for

> parallel data layout,
> block/field decompositions,
> (variable) point-block decompositions,
> removing Dirichlet conditions,
> (nonlinear) hierarchical rediscretization,
> and partial assembly (BDDC/FETI),

just **once**.

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size, offset})$$

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size, offset})$$

Data Layout	mesh point $\Longrightarrow$ # dofs
Boundary conditions	mesh point $\Longrightarrow$ # constrained dofs
Fields	mesh point $\Longrightarrow$ # field dofs

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size}, \text{offset})$$

Decouples Mesh, Discretization, and Solver

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size}, \text{offset})$$

Decouples Mesh, Discretization, and Solver

Assembly gets dofs on each point and mesh traversal, no need for discretization spaces

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size}, \text{offset})$$

Decouples Mesh, Discretization, and Solver

Solver gets data layout and ordering,
no need for mesh traversal

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow (\text{size}, \text{offset})$$

Decouples Mesh, Discretization, and Solver

Solver gets field and point blocking,
no need for discretization spaces

Section

A `Section` is a map

$$\text{mesh point} \Longrightarrow \text{(size, offset)}$$

Decouples Mesh, Discretization, and Solver

Provides interface layer between PETSc and discretization packages Firedrake and LibMesh

Outline

Example: PCTelescope

`PCTelescope` abstracts the parallel distribution of a linear system, so that

May, Sanan, Rupp, Knepley, Smith, PASC, 2016. slides

Example: PCTelescope

`PCTelescope` abstracts the parallel distribution of a linear system, so that

a user can bring their coarse level onto a single process for a direct solve,

```
-pc_type mg
  -pc_mg_levels N
  -mg_coarse_pc_type telescope
    -mg_coarse_pc_telescope_reduction_factor nc
    -mg_coarse_telescope_pc_type lu
```

May, Sanan, Rupp, Knepley, Smith, PASC, 2016. slides

Example: PCTelescope

`PCTelescope` abstracts the parallel distribution of a linear system, so that

or recreate the solver from the Gordon Bell Prize Winner 2015.

```
-pc_type mg
  -pc_mg_levels NR
  -mg_coarse_pc_type telescope
    -mg_coarse_pc_telescope_reduction_factor r
    -mg_coarse_telescope_pc_type mg
    -mg_coarse_telescope_pc_mg_levels NG
    -mg_coarse_telescope_pc_mg_galerkin
      -mg_coarse_telescope_mg_coarse_pc_type gamg
```

May, Sanan, Rupp, Knepley, Smith, PASC, 2016. slides

Example: PCTelescope

`PCTelescope` abstracts the parallel distribution of a linear system, so that

The paper shows scaling up to 32^3 processors on Piz Daint.

May, Sanan, Rupp, Knepley, Smith, PASC, 2016. slides

Example: PCTelescope

`PCTelescope` abstracts the parallel distribution
of a linear system, so that

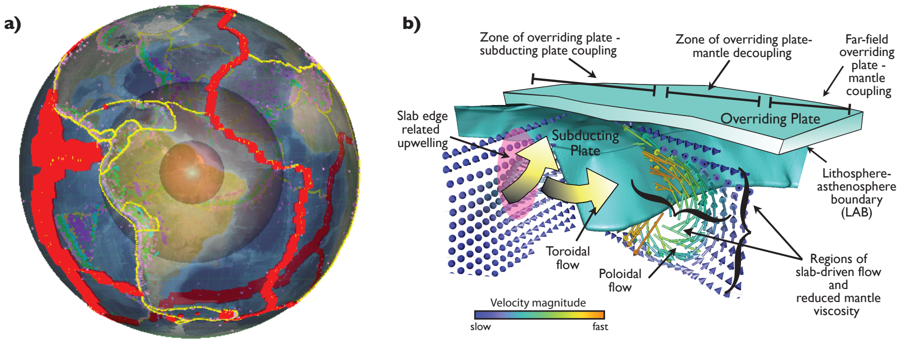
The paper shows scaling up to
32^3 processors on Piz Daint,
and also hybrid CPU-GPU solvers.

May, Sanan, Rupp, Knepley, Smith, PASC, 2016. slides

Outline

Geometric Multigrid with a Coefficient

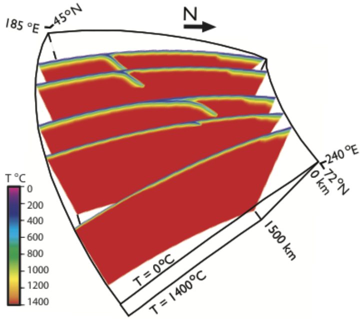
Regional mantle convection
has highly variable viscosity,
due to temperature and strain rate.


a)
b)
Zone of overriding plate - subducting plate coupling
Zone of overriding plate- mantle decoupling
Far-field overriding plate - mantle coupling
Slab edge related upwelling
Subducting Plate
Overriding Plate
Lithosphere-asthenosphere boundary (LAB)
Toroidal flow
Poloidal flow
Regions of slab-driven flow and reduced mantle viscosity
Velocity magnitude
slow
fast

Jadamec, Billen, Nature, 2009.

Geometric Multigrid with a Coefficient

We will specify an initial temperature,
on some initial mesh, and
let strain develop self-consistently.



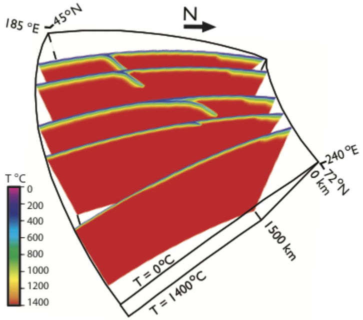
Jadamec, Billen, Nature, 2009.

Geometric Multigrid with a Coefficient

This temperature must be distributed,

matching the mesh partition,

and interpolate/restrict to meshes.



Jadamec, Billen, Nature, 2009.

Geometric Multigrid with a Coefficient: Part I
Distribution

Create `Section` mapping temperature to coarse cells,
using `PetscFECreateDefault()` for a DG function space

Distribute the coarse mesh,
using `DMPlexDistribute()`

Distribute the cell temperatures,
using `DMPlexDistributeField()`

Transfer cell temperatures to finer cells (purely local)

Transfer cell temperatures to vertices on fine grid (purely local)

Geometric Multigrid with a Coefficient: Part I
Distribution

Create `Section` mapping temperature to coarse cells,
using `PetscFECreateDefault()` for a DG function space

Distribute the coarse mesh,
using `DMPlexDistribute()`

Distribute the cell temperatures,
using `DMPlexDistributeField()`

Transfer cell temperatures to finer cells (purely local)

Transfer cell temperatures to vertices on fine grid (purely local)

Geometric Multigrid with a Coefficient: Part I
Distribution

Create `Section` mapping temperature to coarse cells,
using `PetscFECreateDefault()` for a DG function space

Distribute the coarse mesh,
using `DMPlexDistribute()`

Distribute the cell temperatures,
using `DMPlexDistributeField()`

Transfer cell temperatures to finer cells (purely local)

Transfer cell temperatures to vertices on fine grid (purely local)

Geometric Multigrid with a Coefficient: Part I
Distribution

Create `Section` mapping temperature to coarse cells,
using `PetscFECreateDefault()` for a DG function space

Distribute the coarse mesh,
using `DMPlexDistribute()`

Distribute the cell temperatures,
using `DMPlexDistributeField()`

Transfer cell temperatures to finer cells (purely local)

Transfer cell temperatures to vertices on fine grid (purely local)

Geometric Multigrid with a Coefficient: Part I

Distribution

Create `Section` mapping temperature to coarse cells,
using `PetscFECreateDefault()` for a DG function space

Distribute the coarse mesh,
using `DMPlexDistribute()`

Distribute the cell temperatures,
using `DMPlexDistributeField()`

Transfer cell temperatures to finer cells (purely local)

Transfer cell temperatures to vertices on fine grid (purely local)

Geometric Multigrid with a Coefficient: Part II

Interpolation

Interpolation is straightforward

```
DMRefine(coarseMesh, comm, &fineMesh);
DMCreateInterpolation(coarseMesh, fineMesh, &I, &Rscale);
MatMult(I, coarseTemp, fineTemp);
```

Geometric Multigrid with a Coefficient: Part II

Interpolation

Interpolation is straightforward

```
DMRefine(coarseMesh, comm, &fineMesh);
DMCreateInterpolation(coarseMesh, fineMesh, &I, &Rscale);
MatMult(I, coarseTemp, fineTemp);
```

Geometric Multigrid with a Coefficient: Part II

Interpolation

Now we restrict the input temperature to coarser meshes.

```
DMCreateInterpolation(coarseMesh, fineMesh, &I, &Rscale);
MatMultTranspose(I, fineTemp, coarseTemp);
VecPointwiseMult(coarseTemp, coarseTemp, Rscale);
```

Geometric Multigrid with a Coefficient: Part II

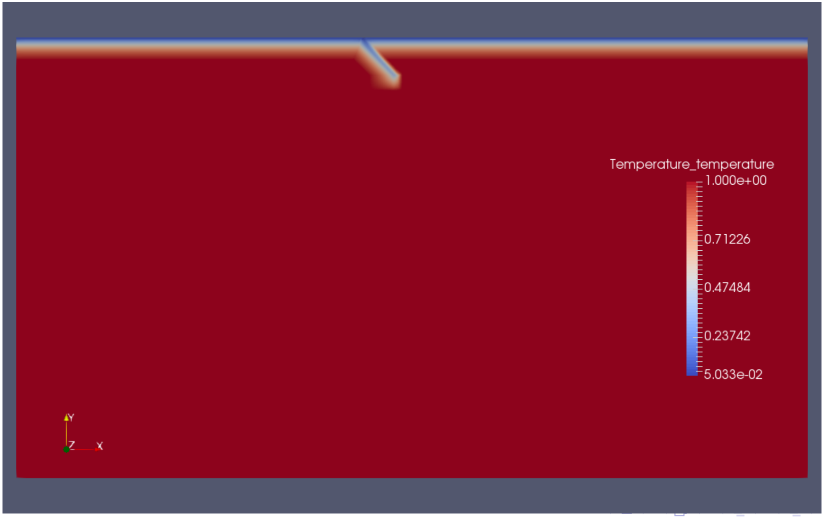
Interpolation

Now we restrict the input temperature to coarser meshes.

```
DMCreateInterpolation(coarseMesh, fineMesh, &I, &Rscale);
MatMultTranspose(I, fineTemp, coarseTemp);
VecPointwiseMult(coarseTemp, coarseTemp, Rscale);
```

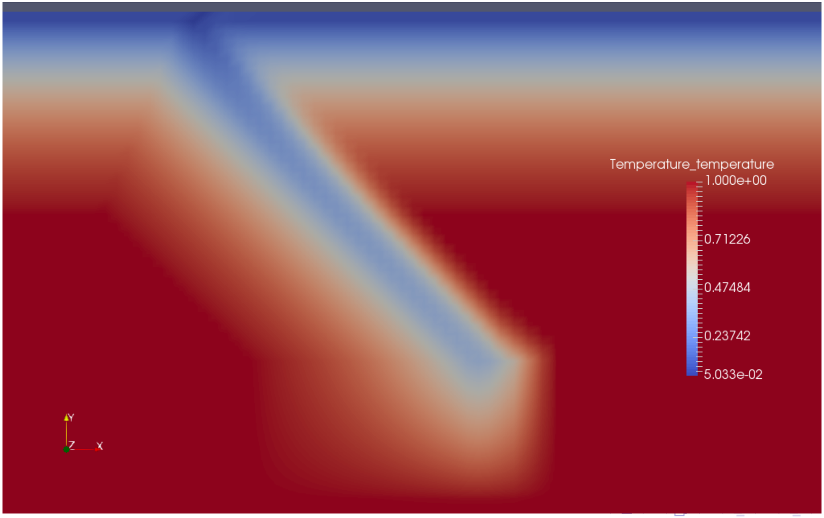
Geometric Multigrid with a Coefficient: Part II
Interpolation

Mantle Temperature (Fine Grid, Level 3)



Geometric Multigrid with a Coefficient: Part II
Interpolation

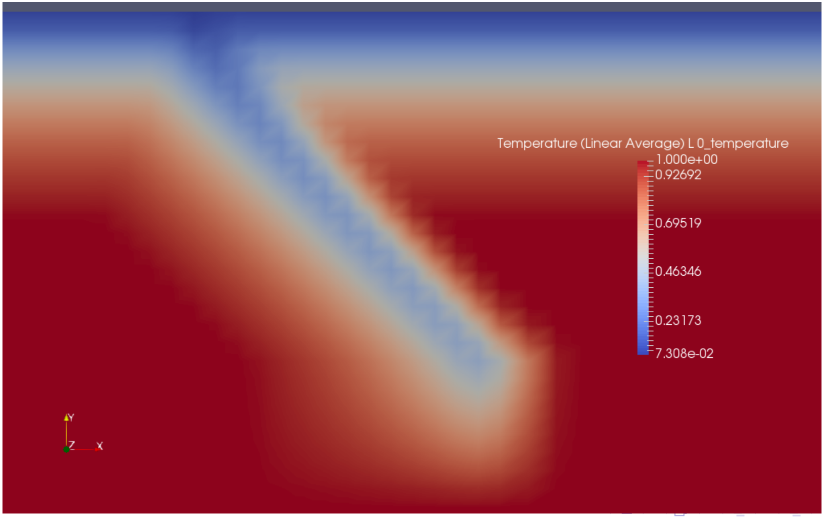
Mantle Temperature (Fine Grid, Level 3)



Geometric Multigrid with a Coefficient: Part II

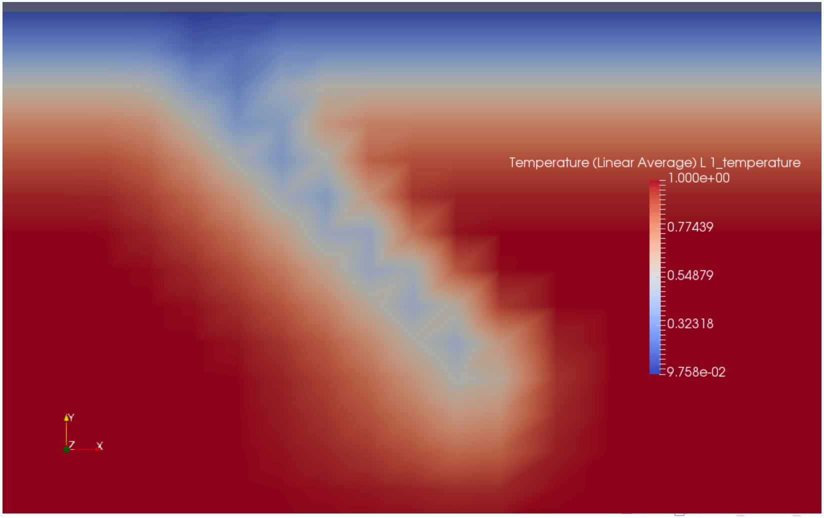
Interpolation

Mantle Temperature (Level 2, Q1 Restriction)



Geometric Multigrid with a Coefficient: Part II
Interpolation

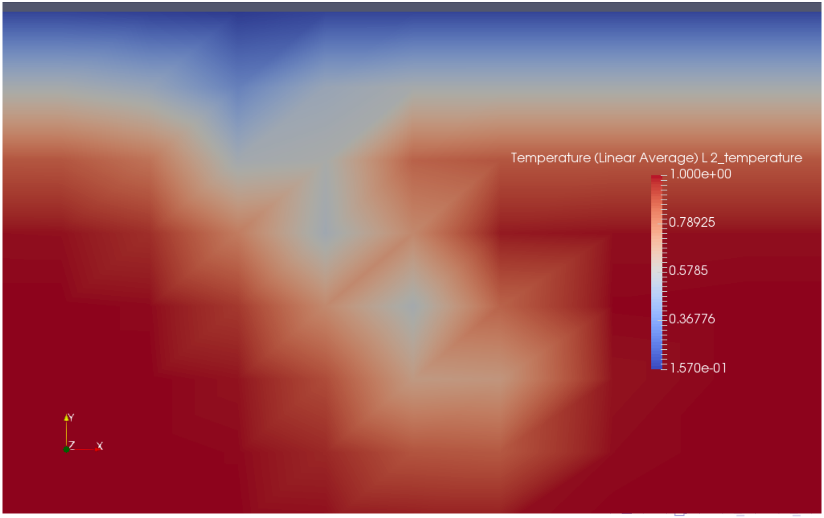
Mantle Temperature (Level 1, Q1 Restriction)



Geometric Multigrid with a Coefficient: Part II

Interpolation

Mantle Temperature (Level 0, Q1 Restriction)



Geometric Multigrid with a Coefficient: Part II

Interpolation

The power mean could better preserve low temperatures

$$\bar{x} = \left(\sum_i x_i^p\right)^{\frac{1}{p}}$$

Geometric Multigrid with a Coefficient: Part II

Interpolation

The power mean could better preserve low temperatures

$$\bar{x} = \left(\sum_i x_i^p\right)^{\frac{1}{p}}$$

```
DMCreateInterpolation(coarseMesh, fineMesh, &I, &Rscale);
MatShellSetOperation(I, MATOP_MULT_TRANSPOSE,
  MatMultTransposePowerMean_SeqAIJ);
MatMultTranspose(I, fineTemp, coarseTemp);
VecPointwiseMult(coraseTemp, coarseTemp, Rscale);
VecPow(coarseTemp, p);
```

Geometric Multigrid with a Coefficient: Part II

Interpolation

The power mean could better preserve low temperatures

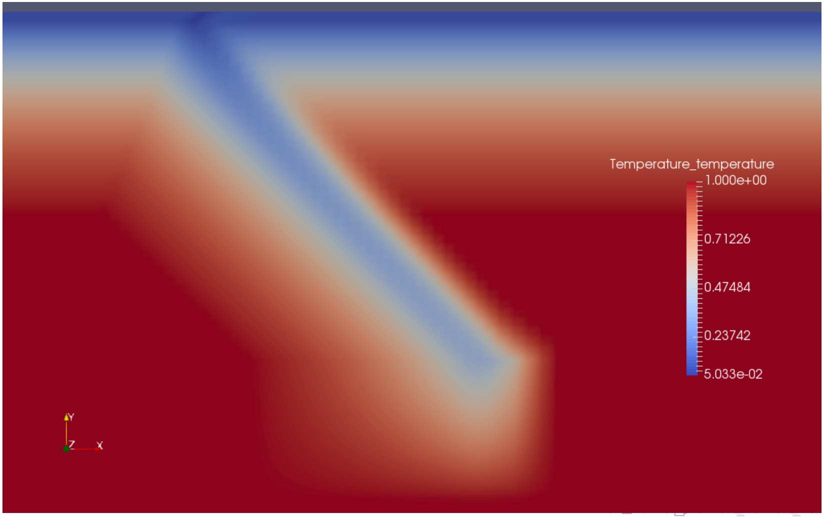
$$\bar{x} = \left(\sum_i x_i^p \right)^{\frac{1}{p}}$$

```
for (i = 0; i < m; ++i) {
  idx = a->j + a->i[i];
  v   = a->a + a->i[i];
  n   = a->i[i+1] - a->i[i];
  xi  = x[i];
  for (j = 0; j < n; ++j) {
    y[idx[j]] += v[j]*PetscPowScalarReal(xi, p);
  }
}
```

It reuses the parallel `MatMultTranspose()` implementation.

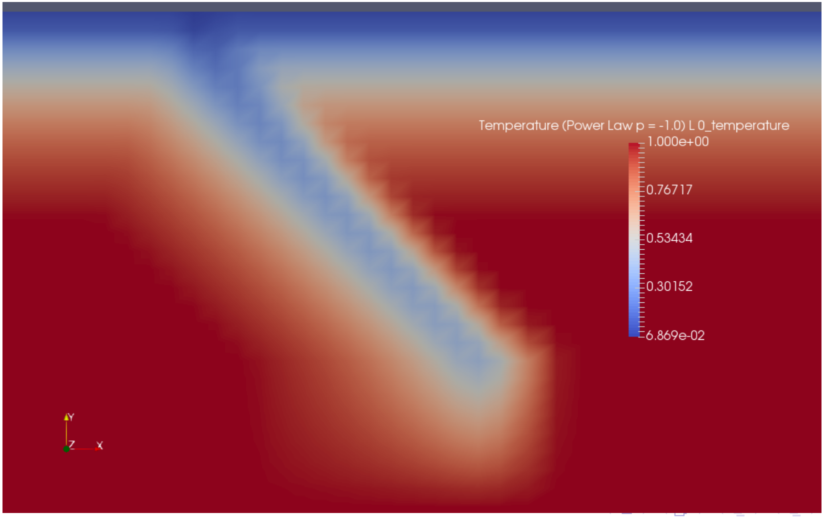
Geometric Multigrid with a Coefficient: Part II
Interpolation

Mantle Temperature (Fine Grid, Level 3)



Geometric Multigrid with a Coefficient: Part II
Interpolation

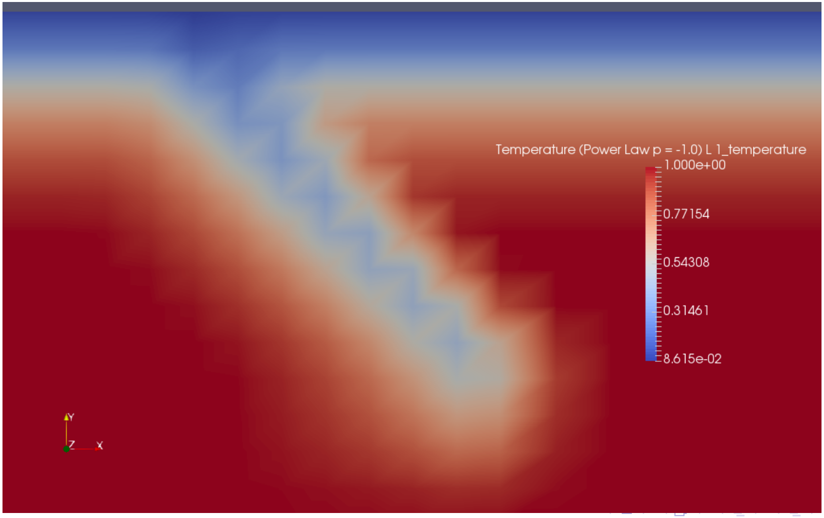
Mantle Temperature (Level 2, Harmonic Restriction)



Geometric Multigrid with a Coefficient: Part II

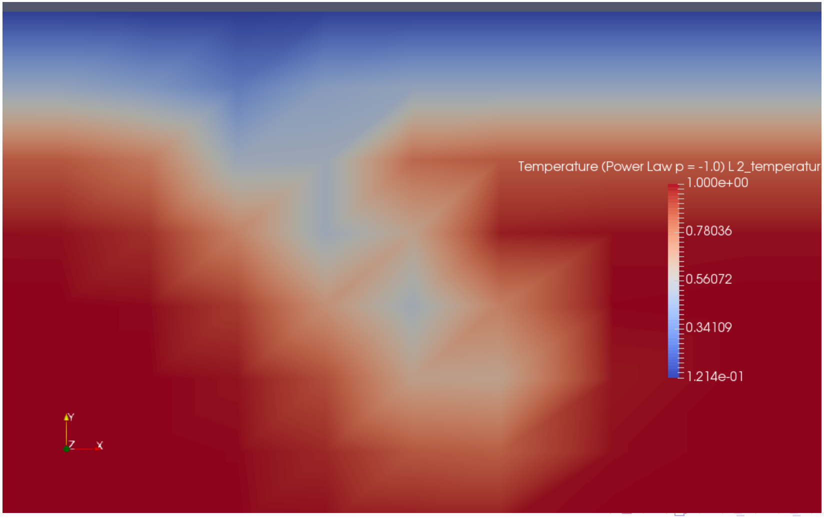
Interpolation

Mantle Temperature (Level 1, Harmonic Restriction)



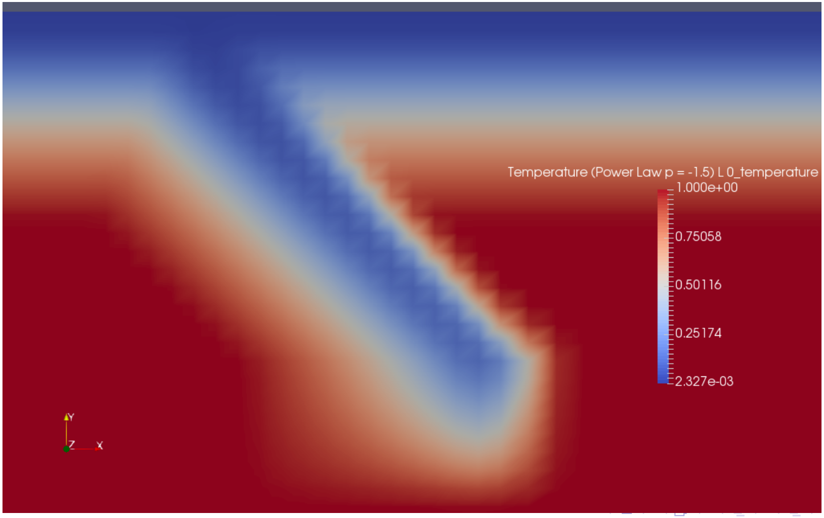
Geometric Multigrid with a Coefficient: Part II
Interpolation

Mantle Temperature (Level 0, Harmonic Restriction)



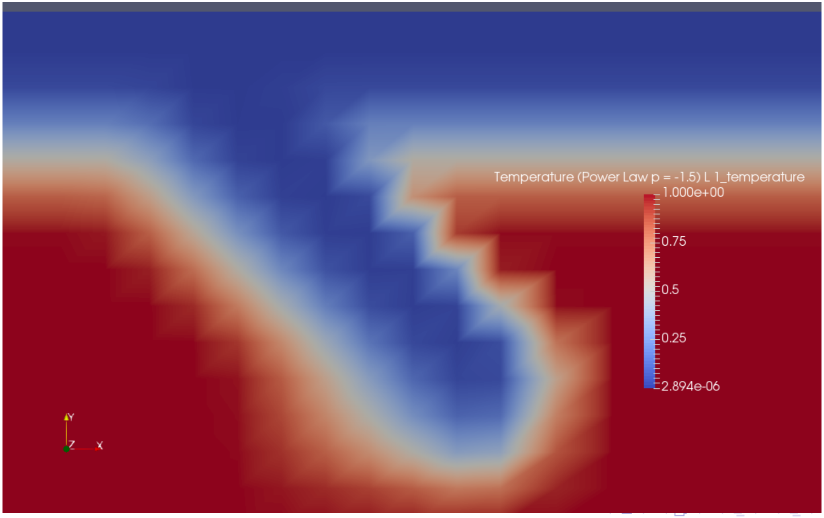
Geometric Multigrid with a Coefficient: Part II
Interpolation

Mantle Temperature (Level 2, $p = -1.5$ Restriction)



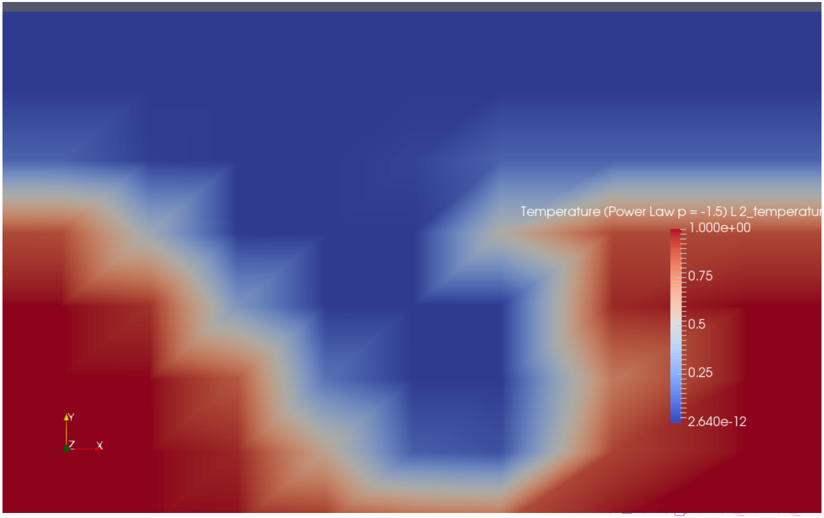
Geometric Multigrid with a Coefficient: Part II
Interpolation

Mantle Temperature (Level 1, $p = -1.5$ Restriction)



Geometric Multigrid with a Coefficient: Part II
Interpolation

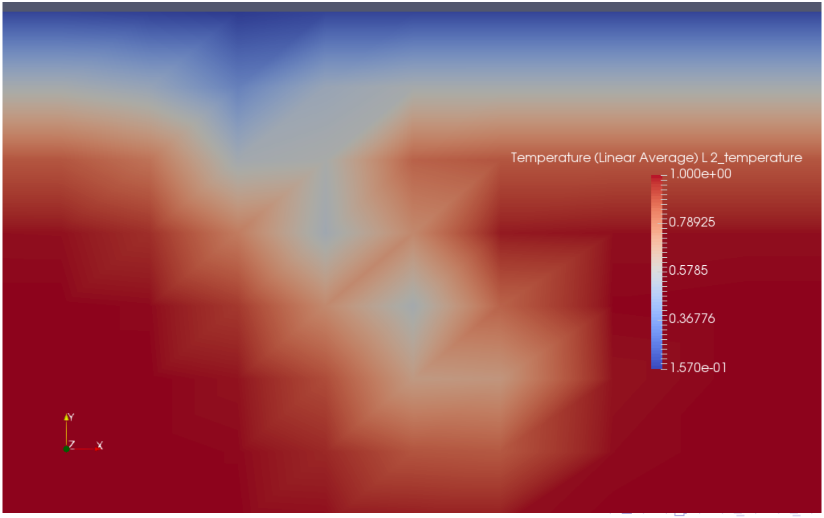
Mantle Temperature (Level 0, $p = -1.5$ Restriction)



Geometric Multigrid with a Coefficient: Part II

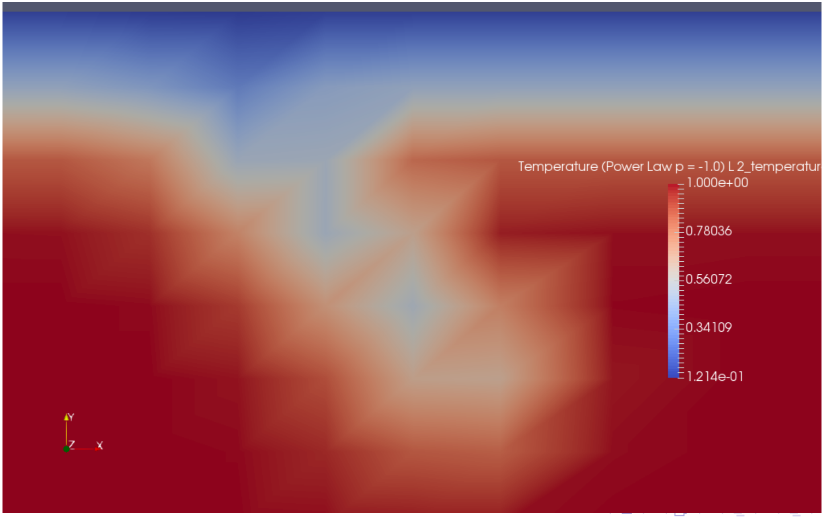
Interpolation

Mantle Temperature (Level 0, Q1 Restriction)



Geometric Multigrid with a Coefficient: Part II
Interpolation

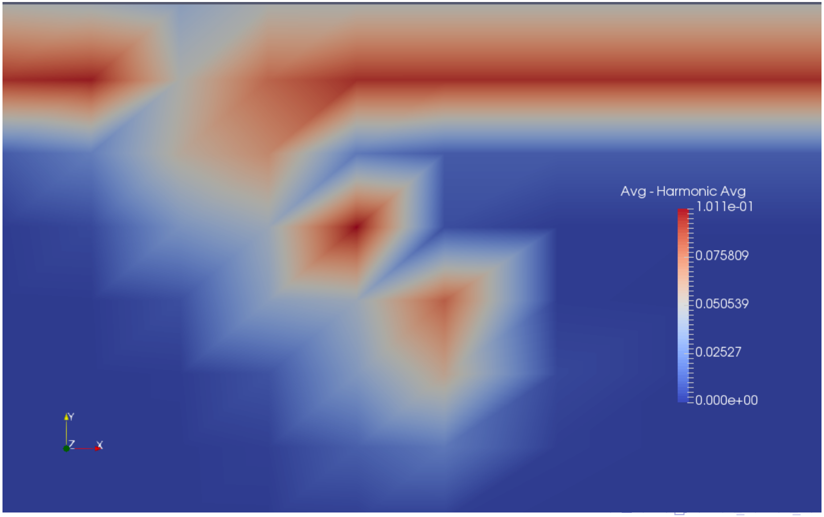
Mantle Temperature (Level 0, Harmonic Restriction)



Geometric Multigrid with a Coefficient: Part II

Interpolation

Mantle Temperature (Level 0, Q1 Avg - Harmonic Avg)



Geometric Multigrid with a Coefficient: Part III
Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Geometric Multigrid with a Coefficient: Part III

Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Geometric Multigrid with a Coefficient: Part III

Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Geometric Multigrid with a Coefficient: Part III
Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Geometric Multigrid with a Coefficient: Part III
Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Geometric Multigrid with a Coefficient: Part III

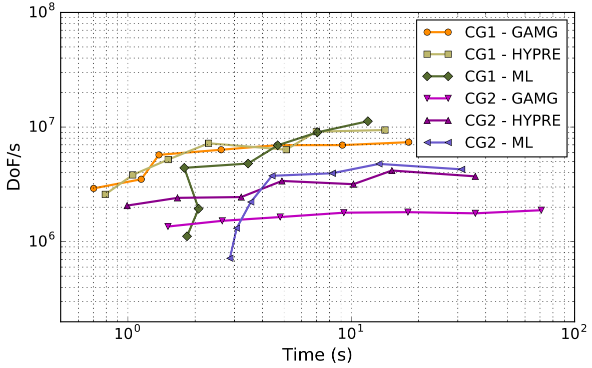
Solving

```
-snes_rtol 1e-7 -snes_atol 1e-12 -snes_linesearch_maxstep 1e20
  -ksp_rtol 1e-5
  -pc_type fieldsplit
    -pc_fieldsplit_diag_use_amat
    -pc_fieldsplit_type schur
    -pc_fieldsplit_schur_factorization_type full
    -pc_fieldsplit_schur_precondition a11
      -fieldsplit_velocity_ksp_type gmres
        -fieldsplit_velocity_ksp_rtol 1e-8
      -fieldsplit_velocity_pc_type mg
        -fieldsplit_velocity_pc_mg_levels n
          -fieldsplit_velocity_mg_levels_ksp_type gmres
            -fieldsplit_velocity_mg_levels_ksp_max_it 4
            -fieldsplit_velocity_mg_levels_pc_type pbjacobi
            -fieldsplit_velocity_mg_levels_pc_pbjacobi_variable
            -fieldsplit_velocity_mg_levels_pc_use_amat
      -fieldsplit_pressure_pc_type asm
        -fieldsplit_pressure_sub_pc_type ilu
        -fieldsplit_pressure_ksp_rtol 1e-4
        -fieldsplit_pressure_ksp_max_it 20
```

Outline

Example: TAS Performance Analysis

Static Scaling (1K procs)



Measure dof/s,

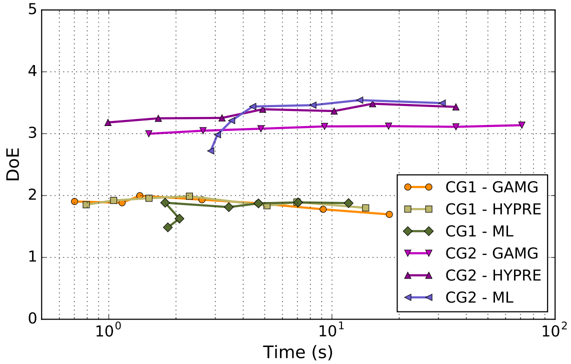
fixed parallelism,

different sizes.

Chang, Fabien, Knepley, Mills, submitted, 2018.

Example: TAS Performance Analysis

Accuracy Scaling (1K procs)



Measure error $\times$ time

fixed parallelism

different sizes.

Chang, Fabien, Knepley, Mills, submitted, 2018.

Conclusions

Abstractions for
Topology and Geometry,
Data Layout, and
Operator Composition
let the
User construct the Simulator.

Conclusions

http://bitbucket.org/petsc/petsc

http://github.com/petsc/petsc

Example: Magma Dynamics

Show magma performance using FAS

Knepley, Melt in the Mantle Program, Newton Institute, 2016.