

The Portable Extensible Toolkit for Scientific Computing

Matthew Knepley

Computational and Applied Mathematics
Rice University

Laboratory for Space and Astrophysical Plasmas
Houston, TX Oct 31, 2016



Never believe *anything*,
unless you can run it.

Never believe *anything*,
unless you can run it.

Outline

1 PETSc

- Who uses PETSc?
- Block Solvers

2 Conclusion

The PETSc Team



Matt Knepley



Barry Smith



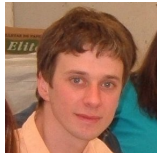
Satish Balay



Jed Brown



Hong Zhang



Lisandro Dalcin



Stefano Zampini



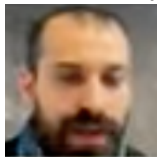
Mark Adams



Toby Issac



Hong Zhang



Pierre Jolivet

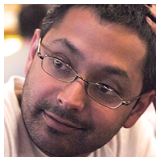


Junchao Zhang

Collaborators



Barry Smith



Jay Bardhan



Boyce Griffith



Jed Brown



Matt Knepley



Michael Lange



Brad Aagaard



Mark Adams



Toby Isaac

Collaborators (Degree)

Math

Barry Smith

EE

Jay Bardhan

Math

Boyce Griffith

Geo

Jed Brown

CS

Matt Knepley

CS

Michael Lange

CivE

Brad Aagaard

CivE

Mark Adams

CSE

Toby Isaac

Collaborators (Occupation)

MCS

Barry Smith

MechE

Jay Bardhan

Math/Med

Boyce Griffith

CS

Jed Brown

Math

Matt Knepley

Geo

Michael Lange

USGS

Brad Aagaard

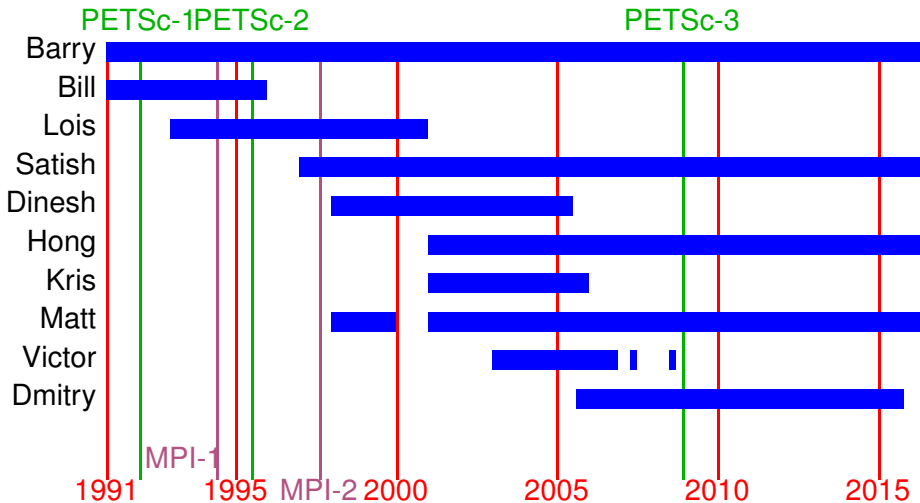
Math

Mark Adams

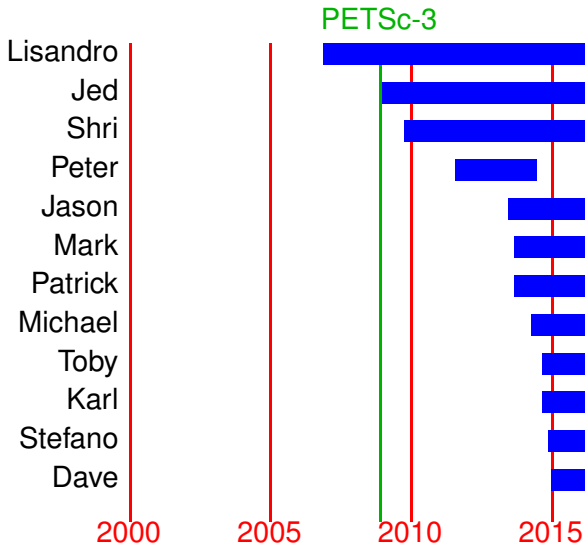
CS

Toby Isaac

Timeline (Old People)



Timeline (Young People)

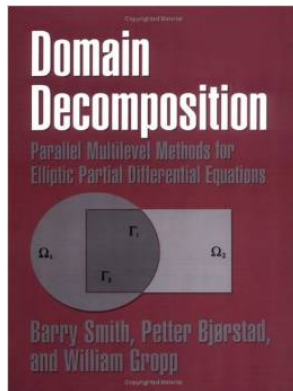


How did PETSc Originate?

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.petsc.org>
- 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
 - ECP, PSAAPIII, AMR, BES, SciDAC, MICS
 - National Science Foundation
 - CSSI, SI2, CIG, CISE
 - Intel Parallel Computing Center

What is PETSc?

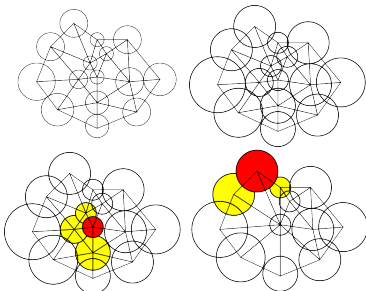
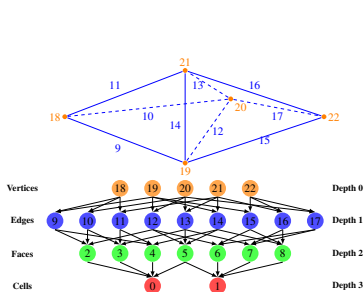
PETSc is one of the most popular software libraries in scientific computing.

As a principal architect since 2001, I developed

- unstructured meshes (model, algorithms, implementation),
- nonlinear preconditioning (model, algorithms),
- FEM discretizations (data structures, solvers optimization),
- optimizations for multicore and GPU architectures.

What is PETSc?

Knepley, Karpeev, Sci. Prog., 2009. Brune, Knepley, Scott, SISC, 2013.



As a principal architect since 2001, I developed

- unstructured meshes (model, algorithms, implementation),
- nonlinear preconditioning (model, algorithms),
- FEM discretizations (data structures, solvers optimization),
- optimizations for multicore and GPU architectures.

What is PETSc?

Brune, Knepley, Smith, and Tu, SIAM Review, 2015.

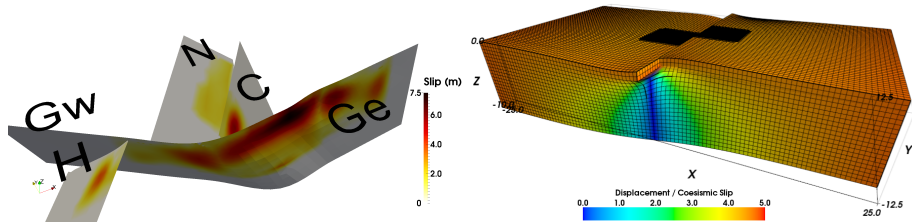
Type	Sym	Statement	Abbreviation
Additive	+	$\vec{x} + \alpha(\mathcal{M}(\mathcal{F}, \vec{x}, \vec{b}) - \vec{x})$ $+ \beta(\mathcal{N}(\mathcal{F}, \vec{x}, \vec{b}) - \vec{x})$	$\mathcal{M} + \mathcal{N}$
Multiplicative	*	$\mathcal{M}(\mathcal{F}, \mathcal{N}(\mathcal{F}, \vec{x}, \vec{b}), \vec{b})$	$\mathcal{M} * \mathcal{N}$
Left Prec.	$-_L$	$\mathcal{M}(\vec{x} - \mathcal{N}(\mathcal{F}, \vec{x}, \vec{b}), \vec{x}, \vec{b})$	$\mathcal{M} -_L \mathcal{N}$
Right Prec.	$-_R$	$\mathcal{M}(\mathcal{F}(\mathcal{N}(\mathcal{F}, \vec{x}, \vec{b})), \vec{x}, \vec{b})$	$\mathcal{M} -_R \mathcal{N}$
Inner Lin. Inv.	\	$\vec{y} = \vec{J}(\vec{x})^{-1} \vec{r}(\vec{x}) = \mathbf{K}(\vec{J}(\vec{x}), \vec{y}_0, \vec{b})$	$\mathcal{N} \backslash \mathbf{K}$

As a principal architect since 2001, I developed

- unstructured meshes (model, algorithms, implementation),
- nonlinear preconditioning (model, algorithms),
- FEM discretizations (data structures, solvers optimization),
- optimizations for multicore and GPU architectures.

What is PETSc?

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

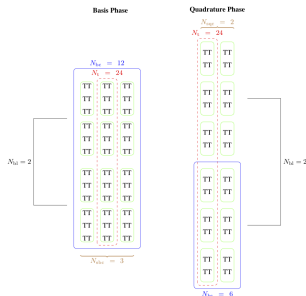
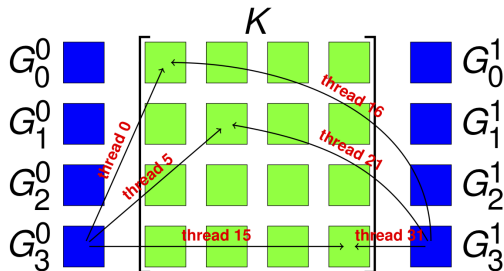


As a principal architect since 2001, I developed

- unstructured meshes (model, algorithms, implementation),
- nonlinear preconditioning (model, algorithms),
- FEM discretizations (data structures, solvers optimization),
- optimizations for multicore and GPU architectures.

What is PETSc?

Knepley and Terrel, Transactions on Mathematical Software, 2012.



As a principal architect since 2001, I developed

- unstructured meshes (model, algorithms, implementation),
- nonlinear preconditioning (model, algorithms),
- FEM discretizations (data structures, solvers optimization),
- optimizations for multicore and GPU architectures.

Outline

- 1 PETSc
 - Who uses PETSc?
 - Block Solvers

Who Uses PETSc?

Computational Scientists

- Earth Science

- PyLith (CIg)
- Underworld (Monash)
- Salvus (ETHZ)
- TerraFERMA (LDEO, Columbia, Oxford)

- Multiphysics

- MOOSE
- GRINS

- Subsurface Flow and Porous Media

- PFLOTRAN (DOE)
- STOMP (DOE)

Who Uses PETSc?

Computational Scientists

- CFD

- IBAMR
- Fluidity
- OpenFVM

- Fusion

- XGC
- BOUT++
- NIMROD
- *M3D - C¹*

Who Uses PETSc?

Algorithm Developers

- Iterative methods
 - Deflated GMRES
 - LGMRES
 - QCG
 - SpecEst
- Preconditioning researchers
 - FETI-DP (Klawonn and Rheinbach)
 - STRUMPACK (Ghysels and Li)
 - HPDDM (Jolivet and Nataf)
 - ParPre (Eijkhout)

Who Uses PETSc?

Algorithm Developers

- Discretization

- Firedrake
- FEniCS
- libMesh
- Deal II
- PETSc-FEM
- OOFEM
- PetRBF

- Outer Loop Solvers

- Eigensolvers (SLEPc)
- Optimization (PERMON)

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 **1,500,000** cores efficiently
 - Gordon Bell Prize Mantle Convection on IBM BG/Q Sequoia
- PETSc applications have run at 23% of peak (**600 Teraflops**)
 - Jed Brown on NERSC Edison
 - HPGMG code

What Can We Handle?

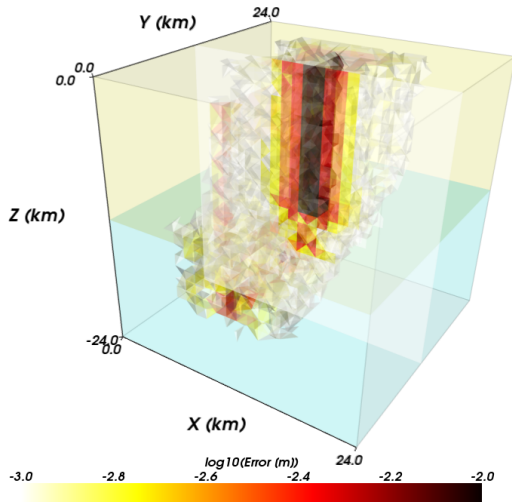
- PETSc has run implicit problems with over **500 billion** unknowns
 - UNIC on BG/P and XT5
 - PFLOTRAN for flow in porous media
- PETSc has run on over **1,500,000** cores efficiently
 - Gordon Bell Prize Mantle Convection on IBM BG/Q Sequoia
- PETSc applications have run at 23% of peak (**600 Teraflops**)
 - 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 **1,500,000** cores efficiently
 - Gordon Bell Prize Mantle Convection on IBM BG/Q Sequoia
- PETSc applications have run at 23% of peak (**600 Teraflops**)
 - Jed Brown on NERSC Edison
 - HPGMG code

PyLith

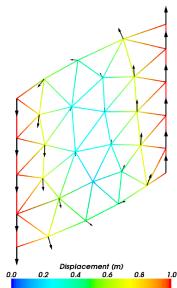
- Multiple problems
 - Dynamic rupture
 - Quasi-static relaxation
 - Numerical Green functions
 - Earthquake cycle
- Multiple models
 - Nonlinear visco-plastic
 - Finite deformation
 - Fault constitutive models
- Multiple meshes
 - 1D, 2D, 3D
 - Hex and tet meshes
 - Refinement and fault insertion
- Scalable, parallel solvers



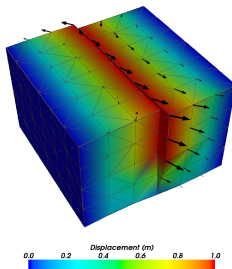
^aAagaard, Knepley, Williams

Multiple Mesh Types

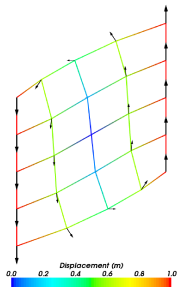
Triangular



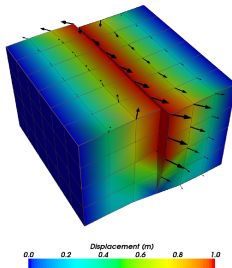
Tetrahedral



Rectangular

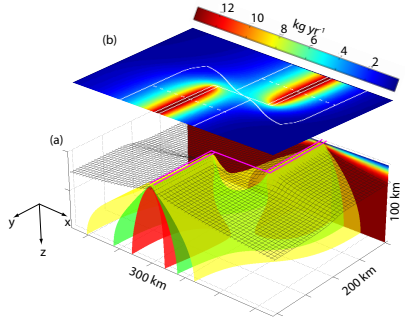
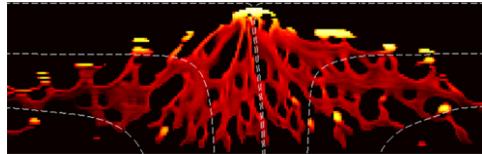


Hexahedral



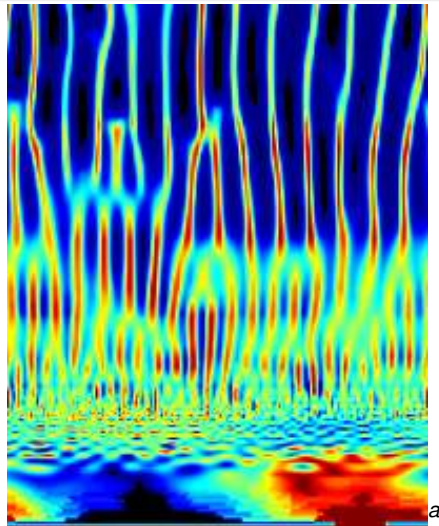
Magma Dynamics

- Couples scales
 - Subduction
 - Magma Migration
- Physics
 - Incompressible fluid
 - Porous solid
 - Variable porosity
- Deforming matrix
 - Compaction pressure
- Code generation
 - FEniCS
- Multiphysics Preconditioning
 - PETSc FieldSplit

^aKatz

Magma Dynamics

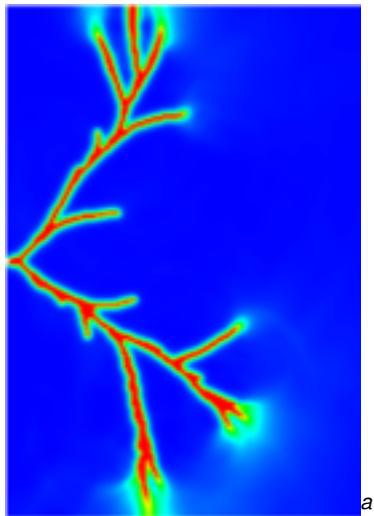
- Couples scales
 - Subduction
 - Magma Migration
- Physics
 - Incompressible fluid
 - Porous solid
 - Variable porosity
- Deforming matrix
 - Compaction pressure
- Code generation
 - FEniCS
- Multiphysics Preconditioning
 - PETSc FieldSplit



^aKatz, Speigelman

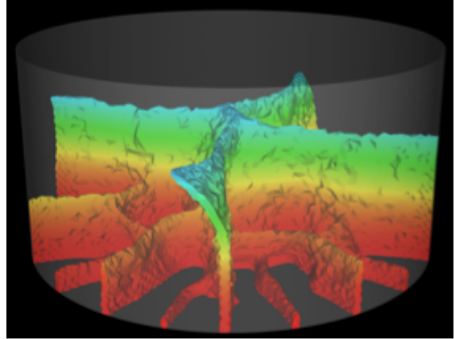
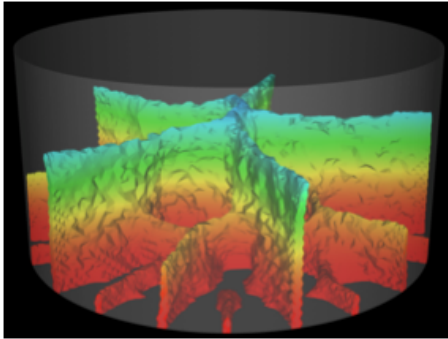
Fracture Mechanics

- Full variational formulation
 - Phase field
 - Linear or Quadratic penalty
- Uses TAO optimization
 - Necessary for linear penalty
 - Backtacking
- No prescribed cracks (**movie**)
 - Arbitrary crack geometry
 - Arbitrary intersections
- Multiple materials
 - Composite toughness



^aBourdin

Fracture Mechanics



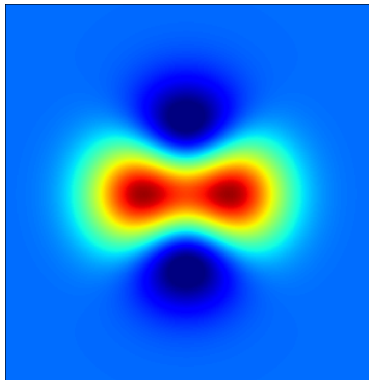
1

¹Bourdin

Vortex Method

$t = 000$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

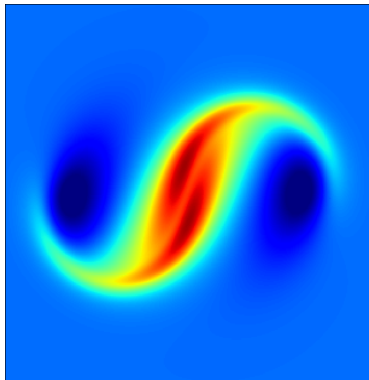


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 100$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

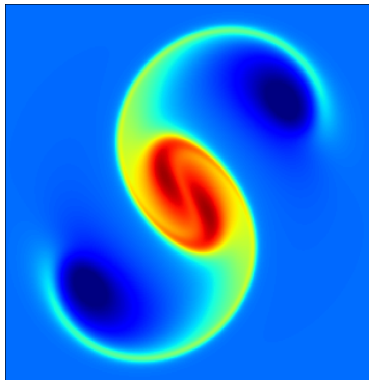


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 200$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

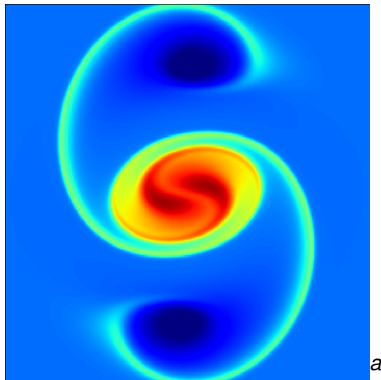


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 300$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

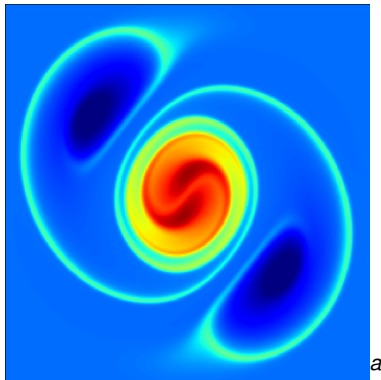


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 400$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

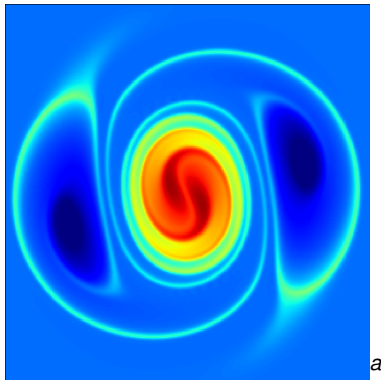


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 500$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

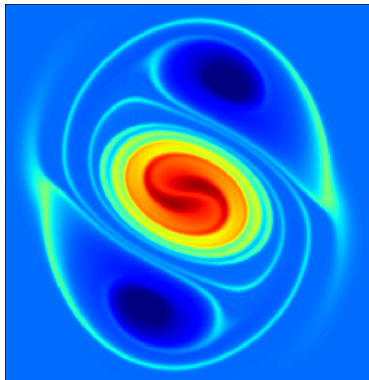


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 600$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

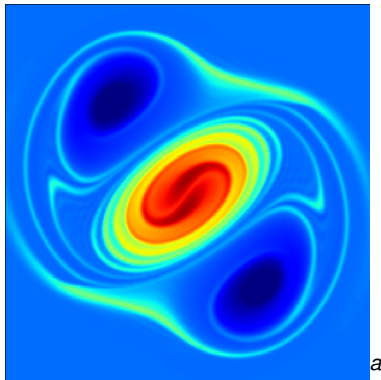


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 700$

- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU

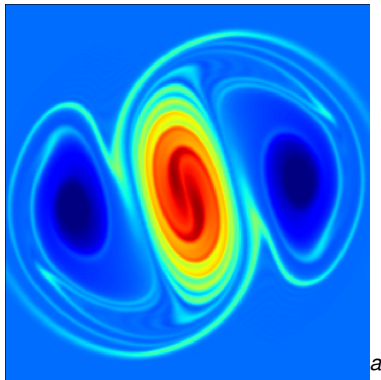


^aCruz, Yokota, Barba, Knepley

Vortex Method

$t = 800$

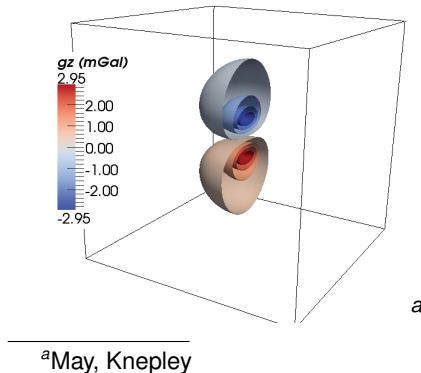
- Incompressible Flow
 - Gaussian vortex blobs
 - High Re
- PetFMM
 - 2D/3D domains
 - Automatic load balancing
 - Variety of kernels
 - Optimized with templates
- PetRBF
 - Variety of RBFs
 - Uses PETSc solvers
 - Scalable preconditioner
- Parallelism
 - MPI
 - GPU



^aCruz, Yokota, Barba, Knepley

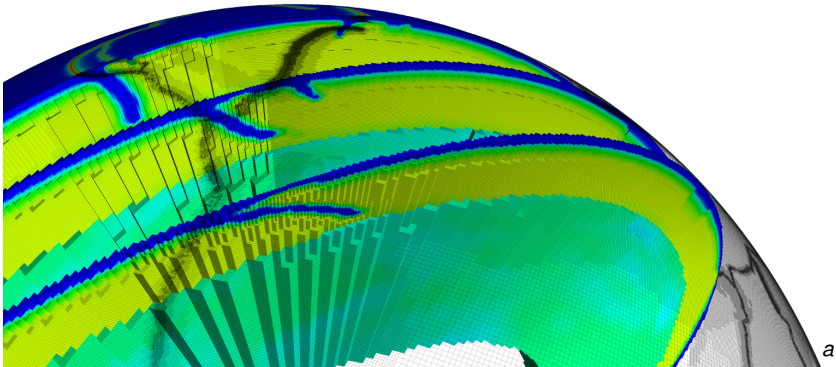
Gravity Anomaly Modeling

- Potential Solution
 - Kernel of inverse problem
 - Needs optimal algorithm
- Implementations
 - Direct Summation
 - FEM
 - FMM
- Parallelism
 - MPI
 - 4000+ cores
 - All methods scalable



Adaptive Mesh Refinement

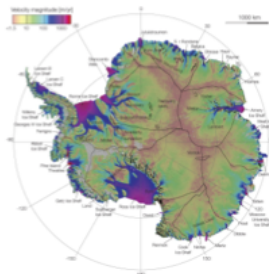
- DM interface with [p4est](#) package from Burstedde and Isaac
- PETSc solvers can be used seamlessly
- 2015 Gordon Bell Winner for Mantle Convection Simulation
- Inversion for basal traction on the full Antarctic ice sheet



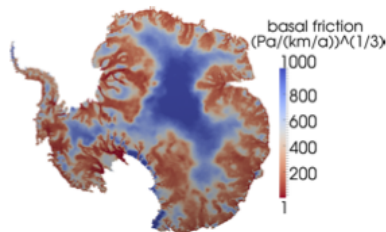
^aIsaac

Adaptive Mesh Refinement

- DM interface with [p4est](#) package from Burstedde and Isaac
- PETSc solvers can be used seamlessly
- 2015 Gordon Bell Winner for Mantle Convection Simulation
- Inversion for basal traction on the full Antarctic ice sheet



Observed surface flow velocity (Rignot et al., 2011)



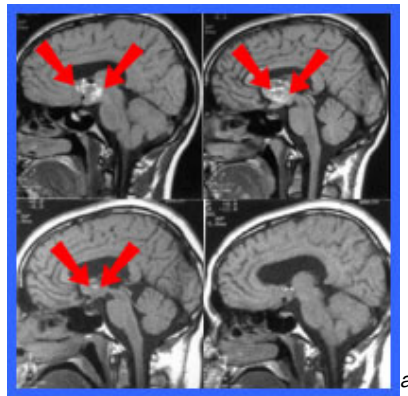
Antarctic ice sheet inversion for the basal friction parameter field using InSAR surface velocity measurements

a

^aIsaac

Real-time Surgery

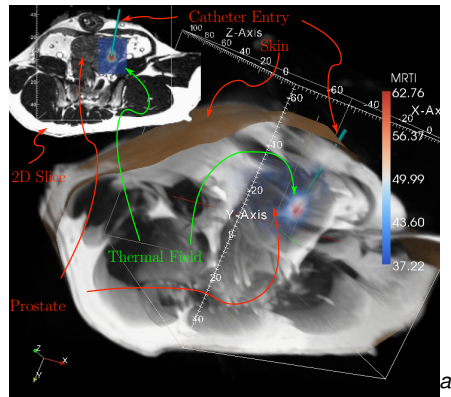
- Brain Surgery
 - Elastic deformation
 - Overlaid on MRI
 - Guides surgeon
- Laser Thermal Therapy
 - PDE constrained optimization
 - Per-patient calibration
 - Thermal inverse problem



^aWarfield, Ferrant, et.al.

Real-time Surgery

- Brain Surgery
 - Elastic deformation
 - Overlaid on MRI
 - Guides surgeon
- Laser Thermal Therapy
 - PDE constrained optimization
 - Per-patient calibration
 - Thermal inverse problem



^aFuentes, Oden, et.al.

Outline

- 1 PETSc
 - Who uses PETSc?
 - Block Solvers

User Solve

```
MPI_Comm comm;
```

```
SNES snes;
```

```
DM dm;
```

```
Vec u;
```

```
SNESCreate(comm, &snes);
```

```
SNESSetDM(snes, dm);
```

```
SNESSetFromOptions(snes);
```

```
DMCreateGlobalVector(dm, &u);
```

```
SNESolve(snes, NULL, u);
```

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Block-Jacobi (Exact), Cohouet & Chabard, IJNMF, 1988.

```
-ksp_type gmres -pc_type fieldsplit -pc_fieldsplit_type additive  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type lu  
-fieldsplit_pressure_ksp_type preonly -fieldsplit_pressure_pc_type jacobi
```

$$\begin{pmatrix} A & 0 \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Block-Jacobi (Inexact), Cohouet & Chabard, *IJNMF*, 1988.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type additive  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type preonly -fieldsplit_pressure_pc_type jacobi
```

$$\begin{pmatrix} \hat{A} & 0 \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Gauss-Seidel (Inexact), Elman, **DTIC**, 1994.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type multiplicative  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type preonly -fieldsplit_pressure_pc_type jacobi
```

$$\begin{pmatrix} \hat{A} & B \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Gauss-Seidel (Inexact), Elman, DTIC, 1994.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type multiplicative  
-pc_fieldsplit_0_fields 1 -pc_fieldsplit_1_fields 0  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type preonly -fieldsplit_pressure_pc_type jacobi
```

$$\begin{pmatrix} I & B^T \\ 0 & \hat{A} \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Diagonal Schur Complement, Olshanskii, et.al., **Numer. Math.**, 2006.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type diag  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type minres -fieldsplit_pressure_pc_type none
```

$$\begin{pmatrix} \hat{A} & 0 \\ 0 & -\hat{S} \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Lower Schur Complement, May and Moresi, **PEPI**, 2008.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type lower  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type minres -fieldsplit_pressure_pc_type none
```

$$\begin{pmatrix} \hat{A} & 0 \\ B^T & \hat{S} \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Upper Schur Complement, May and Moresi, PEPI, 2008.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type upper  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type gamg  
-fieldsplit_pressure_ksp_type minres -fieldsplit_pressure_pc_type none
```

$$\begin{pmatrix} \hat{A} & B \\ & \hat{S} \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Uzawa Iteration, Uzawa, 1958

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type upper  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type lu  
-fieldsplit_pressure_ksp_type richardson -fieldsplit_pressure_pc_type jac  
-fieldsplit_pressure_ksp_max_it 1
```

$$\begin{pmatrix} A & B \\ & \hat{S} \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Full Schur Complement, Schur, 1905.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type full  
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type lu  
-fieldsplit_pressure_ksp_rtol 1e-10 -fieldsplit_pressure_pc_type jacobi
```

$$\begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

SIMPLE, Patankar and Spalding, **IJHMT**, 1972.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur
-pc_fieldsplit_schur_factorization_type full
-fieldsplit_velocity_ksp_type preonly -fieldsplit_velocity_pc_type lu
-fieldsplit_pressure_ksp_rtol 1e-10 -fieldsplit_pressure_pc_type jacobi
-fieldsplit_pressure_inner_ksp_type preonly
-fieldsplit_pressure_inner_pc_type jacobi
-fieldsplit_pressure_upper_ksp_type preonly
-fieldsplit_pressure_upper_pc_type jacobi
```

$$\begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & B^T D_A^{-1} B \end{pmatrix} \begin{pmatrix} I & D_A^{-1} B \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex62: P_2/P_1 Stokes Problem on Unstructured Mesh

Least-Squares Commutator, Kay, Loghin and Wathen, SISC, 2002.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur  
-pc_fieldsplit_schur_factorization_type full  
-pc_fieldsplit_schur_precondition self  
-fieldsplit_velocity_ksp_type gmres -fieldsplit_velocity_pc_type lu  
-fieldsplit_pressure_ksp_rtol 1e-5 -fieldsplit_pressure_pc_type lsc
```

$$\begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & \hat{S}_{\text{LSC}} \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

Solver Configuration: No New Code

ex31: P_2/P_1 Stokes Problem with Temperature on Unstructured Mesh

Additive Schwarz + Full Schur Complement, Elman, Howle, Shadid, Shuttleworth, and Tuminaro, **SISC**, 2006.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type additive
-pc_fieldsplit_0_fields 0,1 -pc_fieldsplit_1_fields 2
-fieldsplit_0_ksp_type fgmres -fieldsplit_0_pc_type fieldsplit
-fieldsplit_0_pc_fieldsplit_type schur
-fieldsplit_0_pc_fieldsplit_schur_factorization_type full
-fieldsplit_0_fieldsplit_velocity_ksp_type preonly
-fieldsplit_0_fieldsplit_velocity_pc_type lu
-fieldsplit_0_fieldsplit_pressure_ksp_rtol 1e-10
-fieldsplit_0_fieldsplit_pressure_pc_type jacobi
-fieldsplit_temperature_ksp_type preonly
-fieldsplit_temperature_pc_type lu
```

$$\begin{pmatrix} \begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} & \begin{pmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{pmatrix} & \begin{pmatrix} I & A^{-1} B \\ 0 & I \end{pmatrix} & 0 \\ 0 & & & L_T \end{pmatrix}$$

Solver Configuration: No New Code

ex31: P_2/P_1 Stokes Problem with Temperature on Unstructured Mesh

Upper Schur Comp. + Full Schur Comp. + Least-Squares Comm.

```
-ksp_type fgmres -pc_type fieldsplit -pc_fieldsplit_type schur
-pc_fieldsplit_0_fields 0,1 -pc_fieldsplit_1_fields 2
-pc_fieldsplit_schur_factorization_type upper
-fieldsplit_0_ksp_type fgmres -fieldsplit_0_pc_type fieldsplit
-fieldsplit_0_pc_fieldsplit_type schur
-fieldsplit_0_pc_fieldsplit_schur_factorization_type full
-fieldsplit_0_fieldsplit_velocity_ksp_type preonly
-fieldsplit_0_fieldsplit_velocity_pc_type lu
-fieldsplit_0_fieldsplit_pressure_ksp_rtol 1e-10
-fieldsplit_0_fieldsplit_pressure_pc_type jacobi
-fieldsplit_temperature_ksp_type gmres
-fieldsplit_temperature_pc_type lsc
```

$$\begin{pmatrix} \begin{pmatrix} I & 0 \\ B^T A^{-1} & I \end{pmatrix} \begin{pmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{pmatrix} \begin{pmatrix} I & A^{-1} B \\ 0 & I \end{pmatrix} & G \\ 0 & \hat{S}_{\text{LSC}} \end{pmatrix}$$

Programming with Options

ex55: Allen-Cahn problem in 2D

- constant mobility
- triangular elements

Geometric multigrid method for saddle point variational inequalities:

```
./ex55 -ksp_type fgmres -pc_type mg -mg_levels_ksp_type fgmres  
-mg_levels_pc_type fieldsplit -mg_levels_pc_fieldsplit_detect_saddle_point  
-mg_levels_pc_fieldsplit_type schur -da_grid_x 65 -da_grid_y 65  
-mg_levels_pc_fieldsplit_factorization_type full  
-mg_levels_pc_fieldsplit_schur_precondition user  
-mg_levels_fieldsplit_1_ksp_type gmres -mg_coarse_ksp_type preonly  
-mg_levels_fieldsplit_1_pc_type none -mg_coarse_pc_type svd  
-mg_levels_fieldsplit_0_ksp_type preonly  
-mg_levels_fieldsplit_0_pc_type sor -pc_mg_levels 5  
-mg_levels_fieldsplit_0_pc_sor_forward -pc_mg_galerkin  
-snes_vi_monitor -ksp_monitor_true_residual -snes_atol 1.e-11  
-mg_levels_ksp_monitor -mg_levels_fieldsplit_ksp_monitor  
-mg_levels_ksp_max_it 2 -mg_levels_fieldsplit_ksp_max_it 5
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Run flexible GMRES with 5 levels of multigrid as the preconditioner

```
./ex55 -ksp_type fgmres -pc_type mg -pc_mg_levels 5  
      -da_grid_x 65 -da_grid_y 65
```

Use the Galerkin process to compute the coarse grid operators

```
-pc_mg_galerkin
```

Use SVD as the coarse grid saddle point solver

```
-mg_coarse_ksp_type preonly -mg_coarse_pc_type svd
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Run flexible GMRES with 5 levels of multigrid as the preconditioner

```
./ex55 -ksp_type fgmres -pc_type mg -pc_mg_levels 5  
      -da_grid_x 65 -da_grid_y 65
```

Use the Galerkin process to compute the coarse grid operators

```
-pc_mg_galerkin
```

Use SVD as the coarse grid saddle point solver

```
-mg_coarse_ksp_type preonly -mg_coarse_pc_type svd
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Run flexible GMRES with 5 levels of multigrid as the preconditioner

```
./ex55 -ksp_type fgmres -pc_type mg -pc_mg_levels 5  
      -da_grid_x 65 -da_grid_y 65
```

Use the Galerkin process to compute the coarse grid operators

```
-pc_mg_galerkin
```

Use SVD as the coarse grid saddle point solver

```
-mg_coarse_ksp_type preonly -mg_coarse_pc_type svd
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Run flexible GMRES with 5 levels of multigrid as the preconditioner

```
./ex55 -ksp_type fgmres -pc_type mg -pc_mg_levels 5  
      -da_grid_x 65 -da_grid_y 65
```

Use the Galerkin process to compute the coarse grid operators

```
-pc_mg_galerkin
```

Use SVD as the coarse grid saddle point solver

```
-mg_coarse_ksp_type preonly -mg_coarse_pc_type svd
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Smoother: Flexible GMRES (2 iterates) with a Schur complement PC

```
-mg_levels_ksp_type fgmres -mg_levels_pc_fieldsplit_detect_saddle_point  
-mg_levels_ksp_max_it 2 -mg_levels_pc_type fieldsplit  
-mg_levels_pc_fieldsplit_type schur  
-mg_levels_pc_fieldsplit_factorization_type full  
-mg_levels_pc_fieldsplit_schur_precondition diag
```

Schur complement solver: GMRES (5 iterates) with no preconditioner

```
-mg_levels_fieldsplit_1_ksp_type gmres  
-mg_levels_fieldsplit_1_pc_type none -mg_levels_fieldsplit_ksp_max_it 5
```

Schur complement action: Use only the lower diagonal part of A00

```
-mg_levels_fieldsplit_0_ksp_type preonly  
-mg_levels_fieldsplit_0_pc_type sor  
-mg_levels_fieldsplit_0_pc_sor_forward
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Smoother: Flexible GMRES (2 iterates) with a Schur complement PC

```
-mg_levels_ksp_type fgmres -mg_levels_pc_fieldsplit_detect_saddle_point  
-mg_levels_ksp_max_it 2 -mg_levels_pc_type fieldsplit  
-mg_levels_pc_fieldsplit_type schur  
-mg_levels_pc_fieldsplit_factorization_type full  
-mg_levels_pc_fieldsplit_schur_precondition diag
```

Schur complement solver: GMRES (5 iterates) with no preconditioner

```
-mg_levels_fieldsplit_1_ksp_type gmres  
-mg_levels_fieldsplit_1_pc_type none -mg_levels_fieldsplit_ksp_max_it 5
```

Schur complement action: Use only the lower diagonal part of A00

```
-mg_levels_fieldsplit_0_ksp_type preonly  
-mg_levels_fieldsplit_0_pc_type sor  
-mg_levels_fieldsplit_0_pc_sor_forward
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Smoother: Flexible GMRES (2 iterates) with a Schur complement PC

```
-mg_levels_ksp_type fgmres -mg_levels_pc_fieldsplit_detect_saddle_point  
-mg_levels_ksp_max_it 2 -mg_levels_pc_type fieldsplit  
-mg_levels_pc_fieldsplit_type schur  
-mg_levels_pc_fieldsplit_factorization_type full  
-mg_levels_pc_fieldsplit_schur_precondition diag
```

Schur complement solver: GMRES (5 iterates) with no preconditioner

```
-mg_levels_fieldsplit_1_ksp_type gmres  
-mg_levels_fieldsplit_1_pc_type none -mg_levels_fieldsplit_ksp_max_it 5
```

Schur complement action: Use only the lower diagonal part of A00

```
-mg_levels_fieldsplit_0_ksp_type preonly  
-mg_levels_fieldsplit_0_pc_type sor  
-mg_levels_fieldsplit_0_pc_sor_forward
```

Programming with Options

ex55: Allen-Cahn problem in 2D

Smoother: Flexible GMRES (2 iterates) with a Schur complement PC

```
-mg_levels_ksp_type fgmres -mg_levels_pc_fieldsplit_detect_saddle_point  
-mg_levels_ksp_max_it 2 -mg_levels_pc_type fieldsplit  
-mg_levels_pc_fieldsplit_type schur  
-mg_levels_pc_fieldsplit_factorization_type full  
-mg_levels_pc_fieldsplit_schur_precondition diag
```

Schur complement solver: GMRES (5 iterates) with no preconditioner

```
-mg_levels_fieldsplit_1_ksp_type gmres  
-mg_levels_fieldsplit_1_pc_type none -mg_levels_fieldsplit_ksp_max_it 5
```

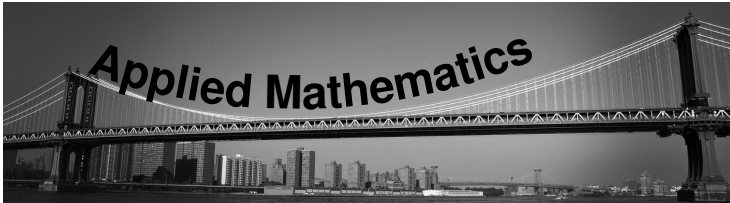
Schur complement action: Use only the lower diagonal part of A00

```
-mg_levels_fieldsplit_0_ksp_type preonly  
-mg_levels_fieldsplit_0_pc_type sor  
-mg_levels_fieldsplit_0_pc_sor_forward
```

Outline

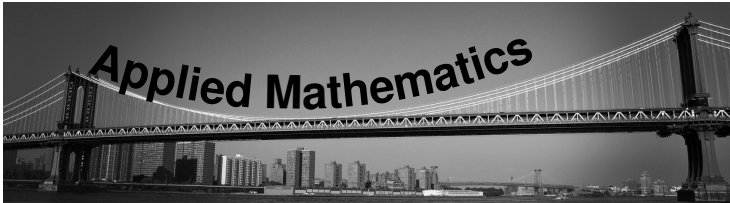
- 1 PETSc
- 2 Conclusion

Impact of Mathematics on Science

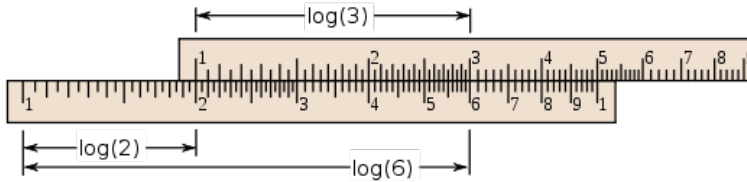


Computational Leaders have always
embraced the latest technology
and been inspired by physical problems,

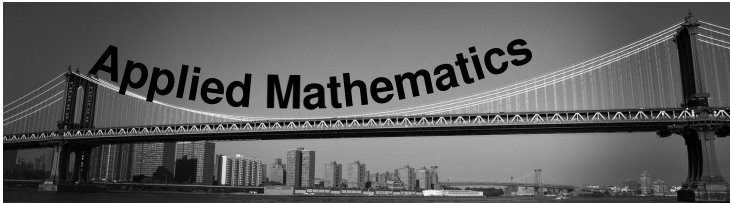
Impact of Mathematics on Science



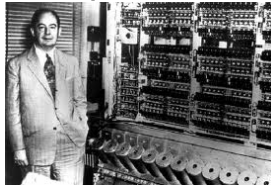
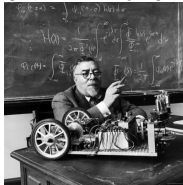
Computational Leaders have always
embraced the latest technology
and been inspired by physical problems,



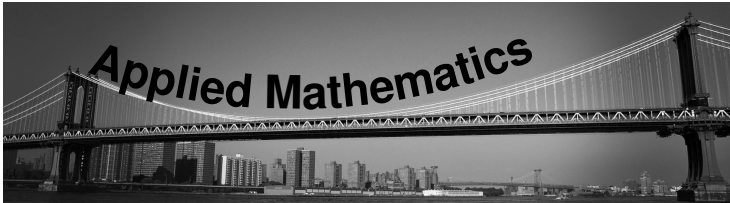
Impact of Mathematics on Science



Computational Leaders have always
embraced the latest technology
and been inspired by physical problems,



Impact of Mathematics on Science

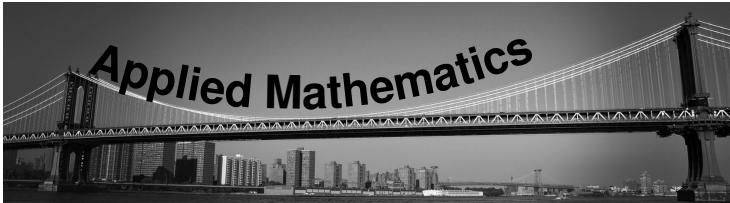


Computational Leaders have always
embraced the latest technology
and been inspired by physical problems,



PETSc

Impact of Mathematics on Science



Computational Leaders have always
embraced the latest technology
and been inspired by physical problems,

Enabling Scientific Discovery