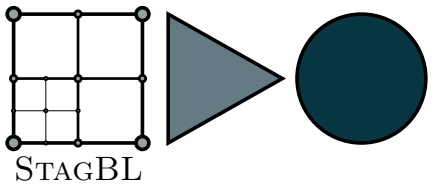
Composable, Scalable Solvers on Staggered Grids

Patrick Sanan[1], Dave A. May[2]
Boris J. P. Kaus[3], The PETSc community[4]

[1]ETH Zurich, [2]University of Oxford, [3]Johannes Gutenberg University Mainz,
[4]gitlab.com/petsc/petsc

EGU General Assembly, April 4-8, 2020, Vienna, Austria



ETH zürich

Platform for Advanced Scientific Computing

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

Motivating Physics and Discretization

- Conservation of mass and momentum

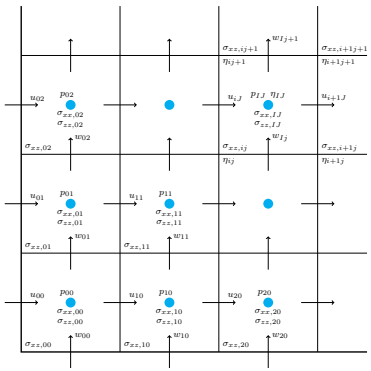
$$\nabla \cdot (\rho v) = 0, \qquad \nabla \cdot \sigma - \nabla p = \frac{\mathrm{Ra}\hat{r}\rho}{\Delta\rho_{\mathrm{thermal}}}$$

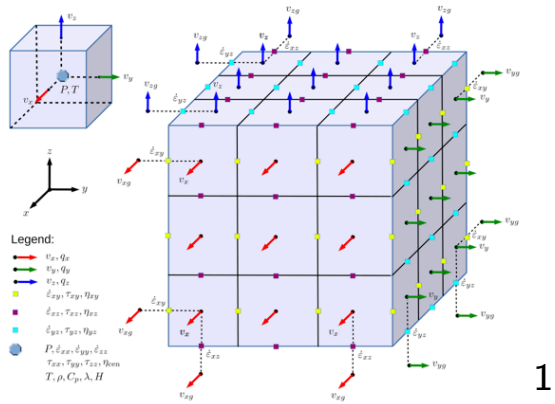
- Constitutive Relationship

$$\dot{\epsilon} \doteq \frac{1}{2}\left(\nabla u + (\nabla u)^T\right), \quad \sigma = \eta\dot{\epsilon}$$

Motivating Physics and Discretization

- Staggered grid finite difference / finite volume methods: Eulerian grid and low order, narrow stencil





1

- MAC discretization[2], with an arbitrary discontinuous viscosity field
- Use conservative finite differences - evaluate η at elements and edges (in 3D)
- Critical computational step: solving saddle point Stokes linear systems

$$\begin{bmatrix} A & G \\ D & \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}, \quad \text{or } \mathcal{A}v = \mathcal{F}$$

[1] B. J. P. Kaus, A. A. Popov, T. Baumann, A. Pusok, A. Bauville, N. Fernandez, and M. Collignon. "Forward and Inverse Modeling of Lithospheric Deformation on Geological Timescales". *NIC Symposium 2016* (2016).

[2] Francis H Harlow and J Eddie Welch. "Numerical Calculation of Time-dependent Viscous Incompressible Flow of Fluid with Free Surface". *The Physics of Fluids* 8.12 (1965), pp. 2182–2189.

Which solver should I implement?

- Good solvers
  - Are scalable
  - take advantage of *block structure* and *ellipticity*
  - have some robustness to coefficient layout (viscosity jumps)
- A large family of approaches have been proposed, all taking advantage of
  - Multigrid
  - Approximate Block Factorization
  - Approximate commutators

Which solver should I implement?

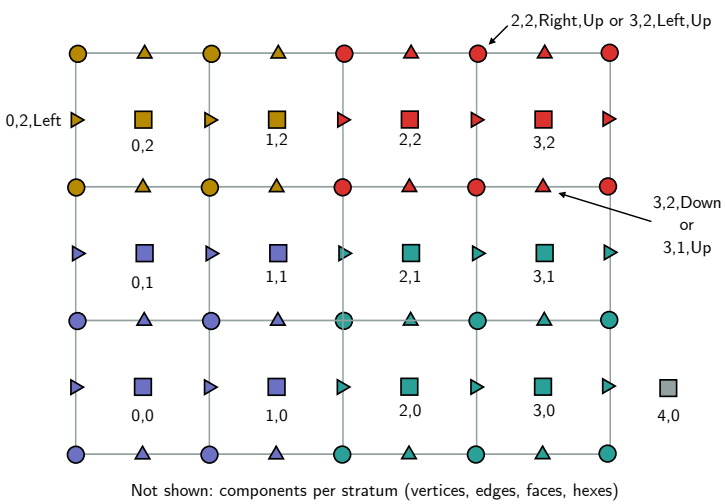
There is a huge literature on solvers which might be of use in any particular application.

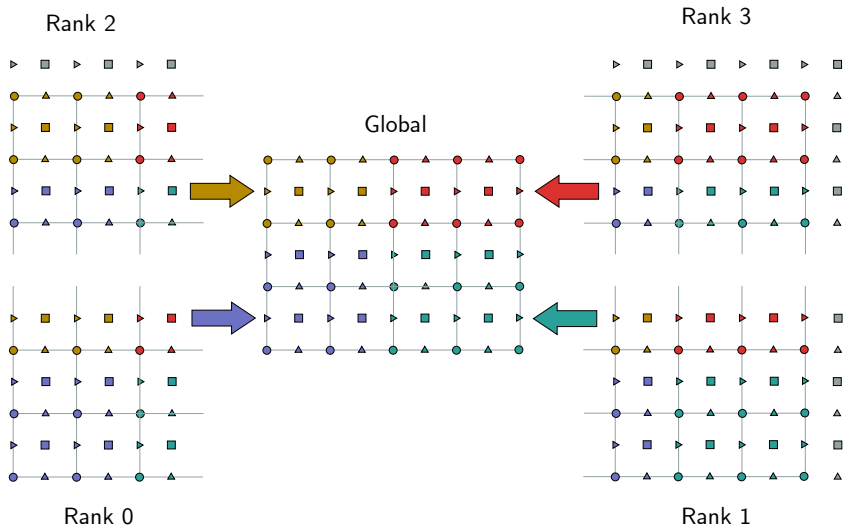
- Segregated methods
- ABF preconditioners
- Monolithic multigrid smoothers
  - Distributed Gauss-Seidel smoothers
  - Vanka Smoothers
  - Braess-Sarazin smoothers

Our solution

- These solvers are all built of the similar components: grid hierarchies, operators, and nested solvers.
- Practically speaking, experimentation is required
- The PETSc library provides a platform to provide the required components.
  - Provide a native parallel staggered-grid object, `DMStag`
  - Provide enough default components to quickly test some perhaps-reasonable solvers, to be further optimized
- Here we will quickly demonstrate the use of some new components with 3 solvers for similar simple 3D benchmark problems with 64^3 elements.
- Experiments are all performed in serial on a workstation, but solvers all work with 3-d parallel domain decompositions (solves do not depend on per-rank subdomain sizes)

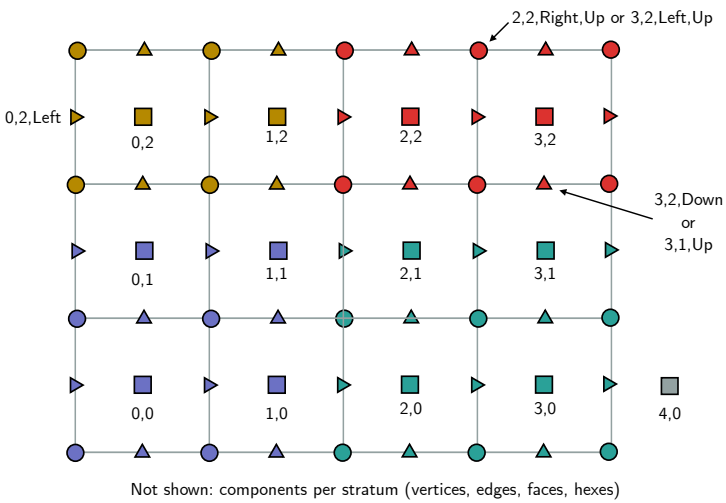
DMStag

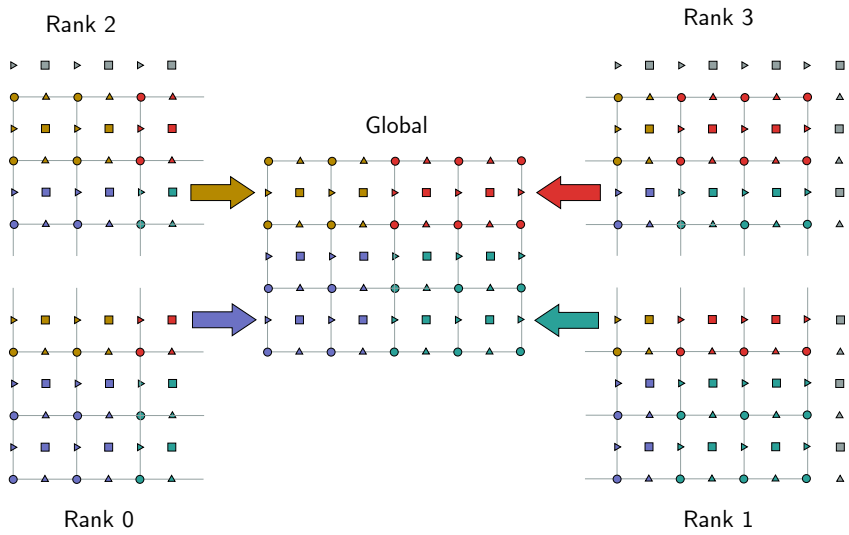




- Periodic and "ghosted" boundary conditions
- Arbitrary grid sizes
- Degrees of freedom on elements, faces, edges, and vertices
- MPI decomposition as a Cartesian product of arbitrary 1-D decompositions
- Coordinates as a Cartesian product of 1-D arrays, with new DMProduct
- Simple API allows user to reason only about global element indices and intuitive "locations" such as `DMSTAG_UP_LEFT`, not e.g. $(i + \frac{1}{2}, j)$

DMStag





Direct access to 1-d coordinate arrays (also available for primary grid-based arrays)

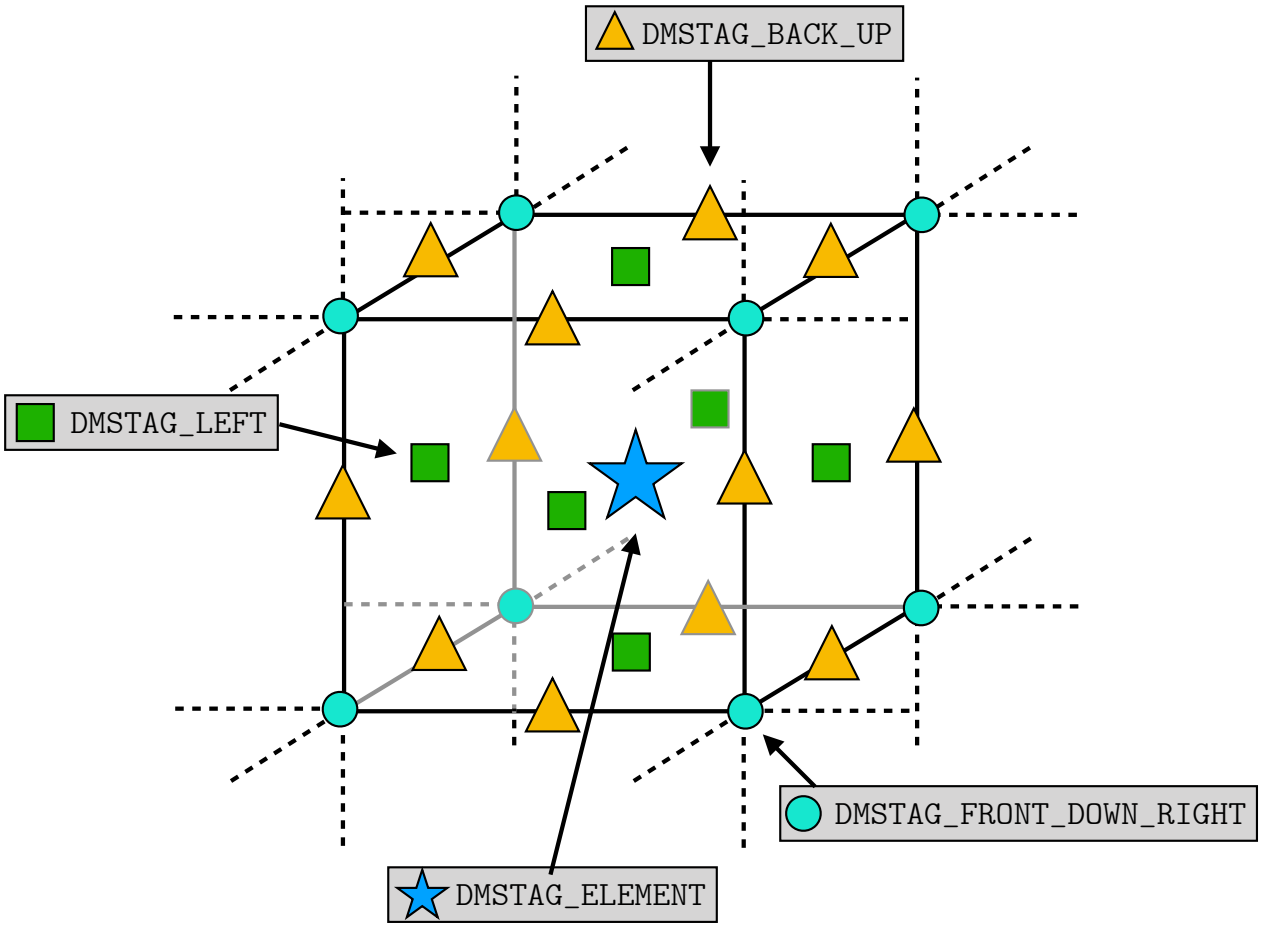
Low-overhead "stencil" objects allow easy construction of operators

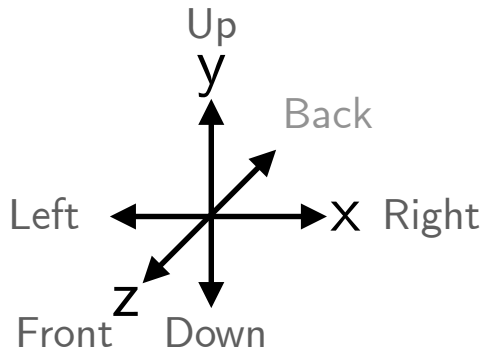
```
// ...
DMStagGetCorners(dm_sol,&startx,&starty,NULL,&nx,&ny,NULL,NULL,NULL,NULL);
DMStagGetGlobalSizes(dm_sol,&N[0],&N[1],NULL);
DMStagGetProductCoordinateArraysRead(dm_sol,&c_arr_x,&c_arr_y,NULL);
DMStagGetProductCoordinateLocationSlot(dm_sol,DMSTAG_ELEMENT,&icenter);
DMStagGetProductCoordinateLocationSlot(dm_sol,DMSTAG_LEFT,&iprev);
DMStagGetProductCoordinateLocationSlot(dm_sol,DMSTAG_RIGHT,&inext);

for (ey = starty; ey<starty+ny; ++ey) {
  for (ex = startx; ex<startx+nx; ++ex) {
    if (ex == N[0]-1) {
      /* Right Boundary velocity Dirichlet */
      DMStagStencil     row;
      PetscScalar       valRhs;
      const PetscScalar valA = 1.0;

      row.i = ex; row.j = ey; row.loc = DMSTAG_RIGHT; row.c = 0;
      DMStagMatSetValuesStencil(dm_sol,A,1,&row,1,&row,&valA,INSERT_VALUES);
      valRhs = uxRef(c_arr_x[ex][inext],c_arr_y[ey][icenter]);
      DMStagVecSetValuesStencil(dm_sol,rhs,1,&row,&valRhs,INSERT_VALUES);
    }
    // ...
  }
}
// ...
```

User need not even know local element ranges, as the API provides them.

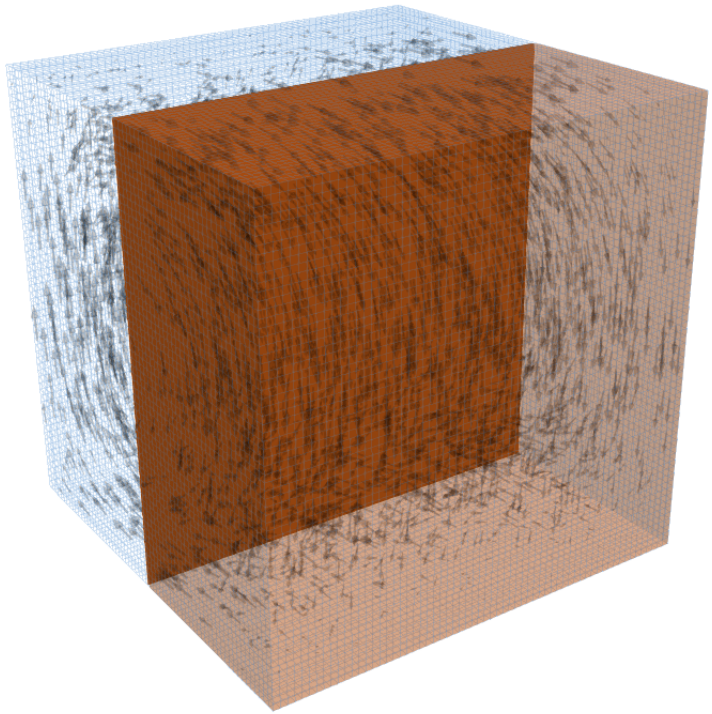
Loop over local elements works for any grid size or parallel decomposition. Only global element indices are used

DMStag in 3D


DMSTAG_BACK_UP
DMSTAG_LEFT
DMSTAG_FRONT_DOWN_RIGHT
DMSTAG_ELEMENT


Up
y
Back
Left
x
Right
z
Front
Down

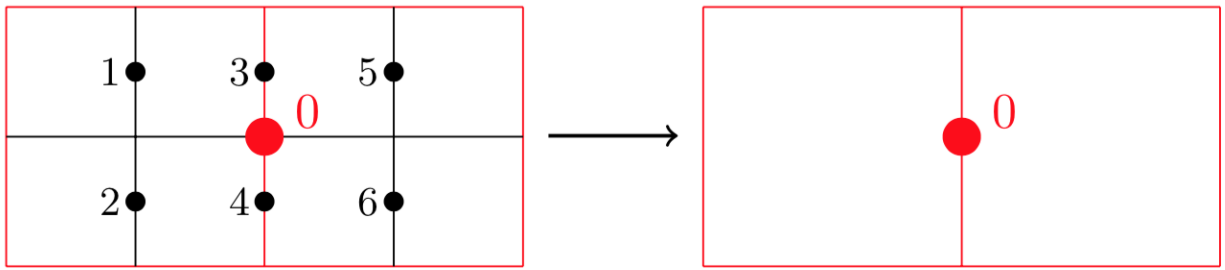
Two-layer test case



- $\rho = 0$, $\eta = 1$ outside the inclusion
- $\rho = 1$, $\eta = 10^2$ inside the inclusion
- Free slip boundary conditions

Monolithic Multigrid

- Multigrid requires a grid hierarchy, transfer operators, and smoothers.
- DMStag can generate refined grids and transfer operators
- The default DMStag transfer operators are a common, low-order choice (e.g.[3][4]) appropriate for MAC Stokes flow.



(A) Fine grid (B) Coarse grid

[3] Mingchao Cai, Andy Nonaka, John B. Bell, Boyce E. Griffith, and Aleksandar Donev. "Efficient Variable-Coefficient Finite-Volume Stokes Solvers". *Communications in Computational Physics* 16.5 (2014), 1263–1297.

[4] Long Chen. *Programming of MAC Scheme for Stokes Equations*. 2018. URL: https://www.math.uci.edu/~chenlong/226/MACcode.pdf.

Monolithic Multigrid

- We can produce a solver similar to one used in LaMEM[5]
- The test code assembles an auxiliary matrix to use to build smoothers.

$$\begin{bmatrix} A & G \\ D & C_{1/\eta} \end{bmatrix}$$

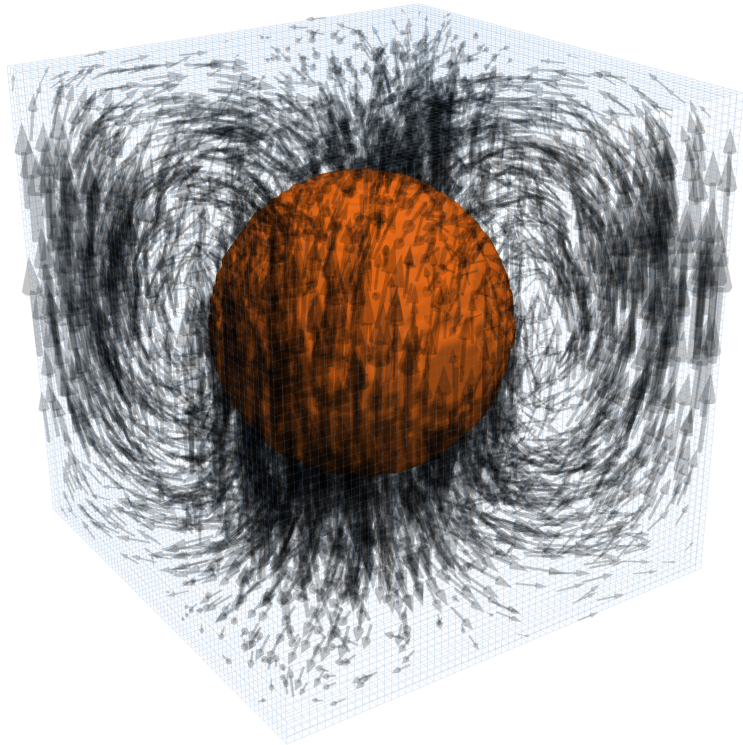
- Otherwise, the solver is defined from the command line

```
-pc_type mg -pc_mg_galerkin -pc_mg_levels 6
-mg_levels_ksp_type richardson
-mg_levels_ksp_richardson_scale 0.5
-mg_levels_pc_type jacobi
-mg_levels_ksp_max_it 20
-mg_coarse_pc_type lu -mg_coarse_pc_factor_mat_solver_type
    umfpack
```

- **Solve**: 28 iterations (42s)

[5] B. J. P. Kaus, A. A. Popov, T. Baumann, A. Pusok, A. Bauville, N. Fernandez, and M. Collignon. "Forward and Inverse Modeling of Lithospheric Deformation on Geological Timescales". *NIC Symposium 2016* (2016).

Spherical Sinker



- $\rho = 0$, $\eta = 1$ outside the inclusion
- $\rho = 1$, $\eta = 10^4$ inside the inclusion (sharp transition)
- Free slip boundary conditions

Spherical Sinker test case

- Remarkably, we can solve this challenging 3-D benchmark problem simply by using built-in capabilities of PETSc and DMStag. That is, the only auxiliary information is that defined by the DMStag object itself and used to assemble the operator, M.
- Fields are extracted automatically, the grid is refined automatically, and default restriction and interpolation operators are used to build Galerkin coarse grid operators.

ABF Preconditioners

- Most of the details are defined with options

```
-pc_type fieldsplit -pc_fieldsplit_type schur
-fieldsplit_element_ksp_type preonly
-fieldsplit_face_pc_type mg -fieldsplit_face_pc_mg_levels 5
-fieldsplit_face_pc_mg_galerkin
-fieldsplit_element_pc_type none
-fieldsplit_face_mg_levels_ksp_max_it 6
-pc_fieldsplit_schur_fact_type upper
-pc_fieldsplit_schur_precondition selfp
```

- Right preconditioning with upper-triangular block

$$\begin{bmatrix} A & G \\ D & \end{bmatrix} \begin{bmatrix} \tilde{A}^{-1} & G \\ & \tilde{S}^{-1} \end{bmatrix} \begin{bmatrix} u' \\ p' \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}$$

- Schur preconditioner

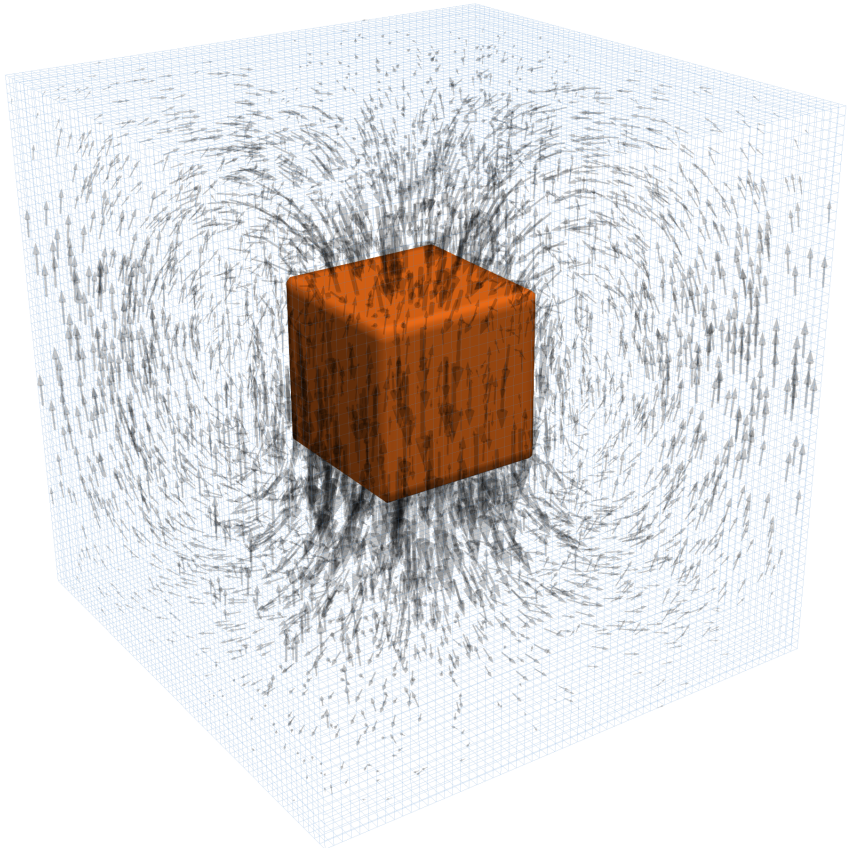
$$\tilde{S} = G \ \mathrm{diag}(A)^{-1} D$$

- **Solve**: 14 FGMRES iterations (195 s)

ABF Preconditioners

- We can directly provide rediscretized operators to PETSc's geometric multigrid preconditioner (not difficult, since our function to create the system uses DMStag)
- This is particularly important if one would like to use matrix-free smoothers
- Similarly, we can assemble a custom Schur complement preconditioning matrix, here the same inverse viscosity-weighted block as used in the monolithic multigrid example earlier.

SINKER



- $\rho = 0$, $\eta = 1$ outside the inclusion
- $\rho = 1$, $\eta = 10^2$ inside the inclusion (sharp transition, not grid-aligned)
- Free slip boundary conditions

ABF Preconditioners with rediscretized operators

- We attempt to create a solver similar to a published one[6]
- Many Richardson-Jacobi smoothing iterations on the viscous block multigrid levels
- Not an exact reproduction
  - the original study uses double-double precision inside the MG solver
  - the original study uses different transfer operators
- Here, we assemble an inverse viscosity-scaled mass matrix, as in the previous example, to use as a Schur complement preconditioner, but could also assemble a BFBT[7] or w-BFBT preconditioner[8].
- **Solve** : 35 FGMRES iterations (1500 s) (Bad, but we can incrementally optimize!)
- **Solve (isoviscous)**: 14 FGMRES iterations (195 s)

[6] Mikito Furuichi, Dave A. May, and Paul J. Tackley. "Development of a Stokes Flow Solver Robust to Large Viscosity Jumps using a Schur Complement Approach with Mixed Precision Arithmetic". *Journal of Computational Physics* 230.24 (2011), pp. 8835–8851.

[7] Dave A. May and Louis Moresi. "Preconditioned Iterative Methods for Stokes Flow Problems Arising in Computational Geodynamics". *Physics of the Earth and Planetary Interiors* 171.1-4 (2008), pp. 33–47.

[8] Johann Rudi, Georg Stadler, and Omar Ghattas. "Weighted BFBT Preconditioner for Stokes Flow Problems with Highly Heterogeneous Viscosity". *SIAM Journal on Scientific Computing* 39.5 (2017), S272–S297.

Other Interesting options

Additional relevant smoothers can be enabled with some additional implementation:

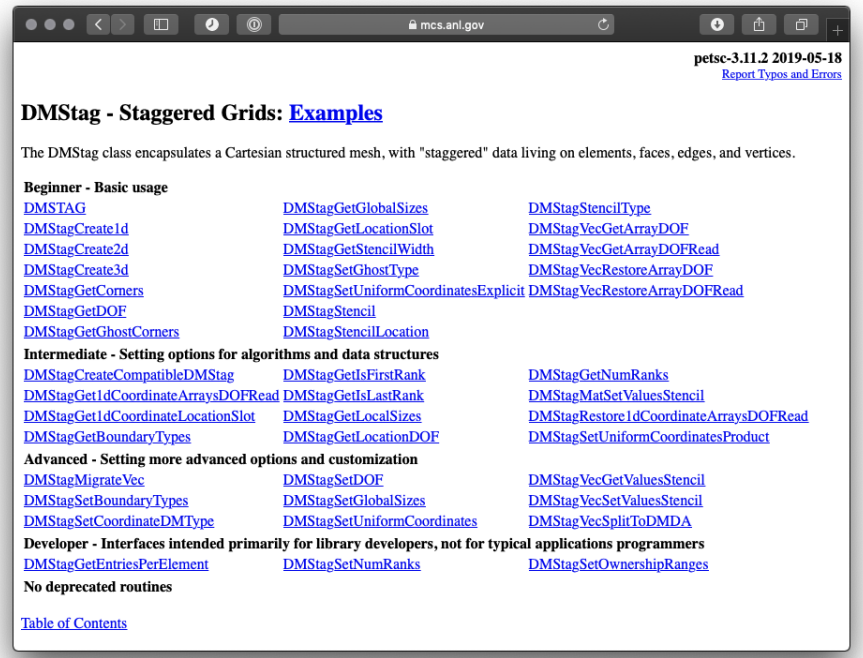
- Vanka Smoothers : provide required field information from DMStag to PCPatch[9]
- Distributed Gauss-Seidel smoothers - new options for PCFieldSplit in development.
- Other restriction and interpolation operators. In addition to $R = P^T$ and/or per-startum bilinear interpolation/restriction, some researchers use a wider stencil[10]

[9] Patrick E. Farrell, Matthew G. Knepley, Florian Wechsung, and Lawrence Mitchell. *PCPATCH: Software for the Topological Construction of Multigrid Relaxation Methods*. 2019. eprint: arXiv:1912.08516.

[10] Anton Niestegge and Kristian Witsch. "Analysis of a Multigrid Stokes solver". *Applied Mathematics and Computation* 35.3 (1990), pp. 291–303.

Code Availability

- DMStag basic components in the release version of PETSc



https://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/DMSTAG/index.html

- Code used here and other new features on a public development branch, making its way to the master branch.
- patrick.sanan@erdw.ethz.ch