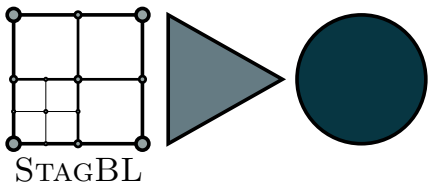
StagBL Tutorial

Patrick Sanan

ETH Zurich

Staggered Grid Geodynamics Workshop, March 4-5, 2020



ETH zürich

Platform for Advanced Scientific Computing

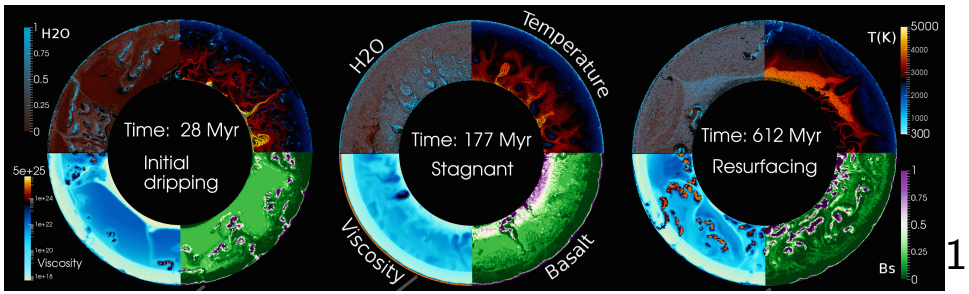
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

Outline

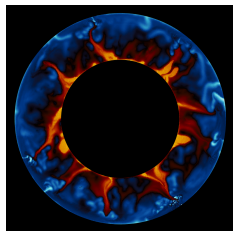
StagBL: Motivating Applications and Challenges

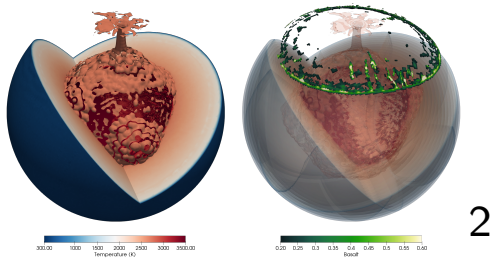
Motivating Challenges from Mantle and Lithospheric Dynamics

- Direct physical modelling of the viscous deformation of large parts of rocky planets, over the longest geological timescales (millions and billions of years)
  - Mantle convection
  - Plate tectonics
  - Planetary formation
- Multiple space/time scales, coupling to other systems, complex local physics/chemistry $\rightarrow$ large demand for computational resources



1

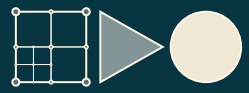



2

[1] A. B. Rozel, G. J. Golabek, C. Jain, P. J. Tackley, and T. Gerya. "Continental Crust Formation on Early Earth Controlled by Intrusive Magmatism". *Nature* 545.7654 (2017), pp. 332–335.

[2] Simulation by Charitra Jain (University of Durham)

Motivating Physics and Discretization



- Conservation of mass and momentum

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

- Conservation of Energy

$$\rho C_p \frac{DT}{Dt} = -\mathrm{Di}_s \alpha \rho T v_r + \nabla \cdot (k \nabla T) + \rho H + \frac{\mathrm{Di}_s}{\mathrm{Ra}} \sigma : \dot{\epsilon}$$

- Advection of arbitrary Lagrangian quantities

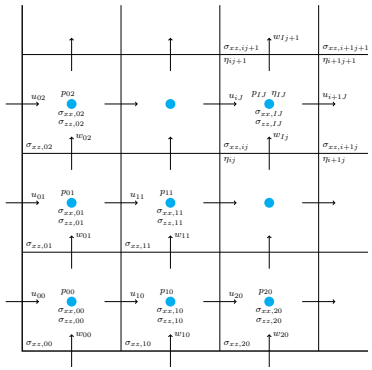
$$\frac{DC}{Dt} = 0$$

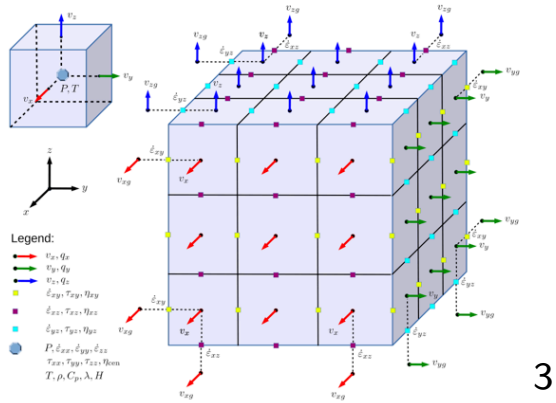
- Constitutive Relationship

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

Motivating Physics and Discretization

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





[3]

- Particle-in-cell/MAC methods[4] ($\rightarrow$ arbitrary discontinuous viscosity field)
- Critical computational step is solving saddle point Stokes linear systems

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

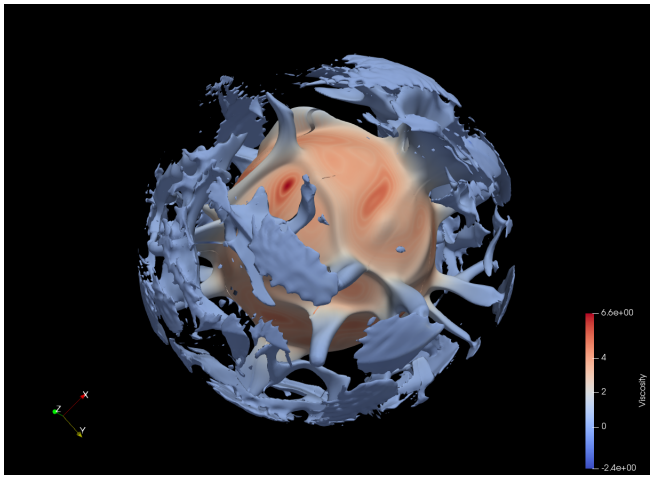
[3] B. J. P. Kaus, A. A. Popov, T. Baumann, A. Pusok, A. Bauville, N. Fernandez, and M. Collignon. "Forward and Inverse Modeling of Lithospheric Deformation on Geological Timescales". *NIC Symposium 2016* (2016).

[4] 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.

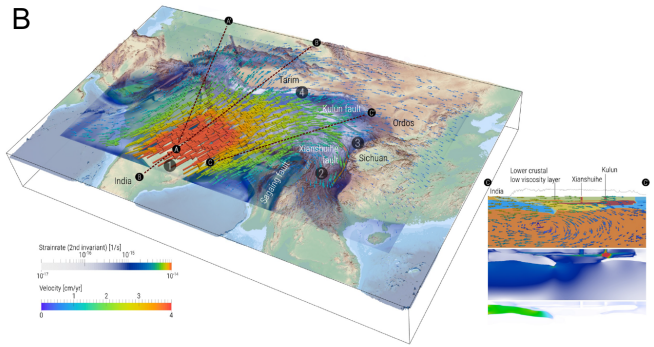
The StagBL Project

Mantle and Lithospheric Dynamics Codes

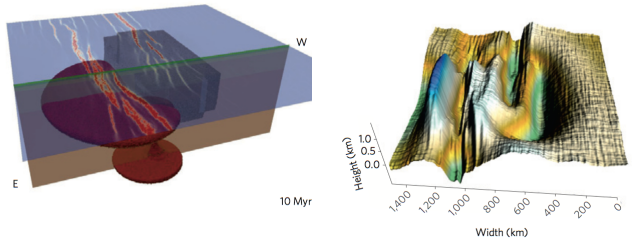
StagYY



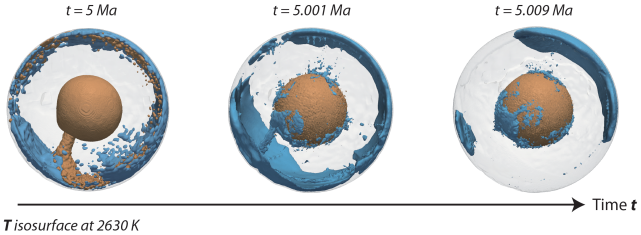
LaMEM[5]



I3ELVIS[6]



I3ELVIS_PLANET [7]



Plus, many others in geodynamics and far beyond.

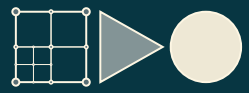
---

[7] https://bitbucket.org/bkaus/lamem, A. E. Pusok and Boris J. P. Kaus. "Development of Topography in 3-D Continental-Collision Models". *Geochemistry, Geophysics, Geosystems* 16.5 (2015), pp. 1378–1400

[7] Taras V. Gerya and David A. Yuen. "Robust Characteristics Method for Modelling Multiphase Visco-Elasto-Plastic Thermo-Mechanical Problems". *Physics of the Earth and Planetary Interiors* 163.1 (2007), pp. 83–105

[7] G.J. Golabek, T.V. Gerya, B.J.P. Kaus, R. Ziethe, and P.J. Tackley. "Rheological Controls on the Terrestrial Core Formation Mechanism". *Geochemistry, Geophysics, Geosystems* 10.11 (2009)

Bottlenecks



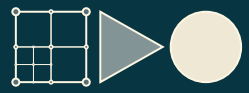
- non-uniform performance
- not all codes can be arbitrarily parallelized
- not all codes can leverage composable solvers
- codes can't share optimized operations

StagBL PASC Project

- Staggered-grid FD plus particle advection is a useful, efficient, and widely-used tool, worthy of a more robust library implementation.
- Low-hanging fruit in accelerating the work of users of 3 application codes as they push the boundaries of their research topics.
- People want solver options, hence need a uniform interface to them
- Address the bottleneck between geodynamics researchers and supercomputers: "Path to performance" from the textbook

- Better analysis tools for multigrid (non-)convergence are needed in the broader community.
- AMR has many potential applications, but needs focused attention to develop and implement for FD grids.
- Inverse modeling: many forward runs incentivize optimization of kernels.

StagBL Design

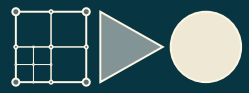


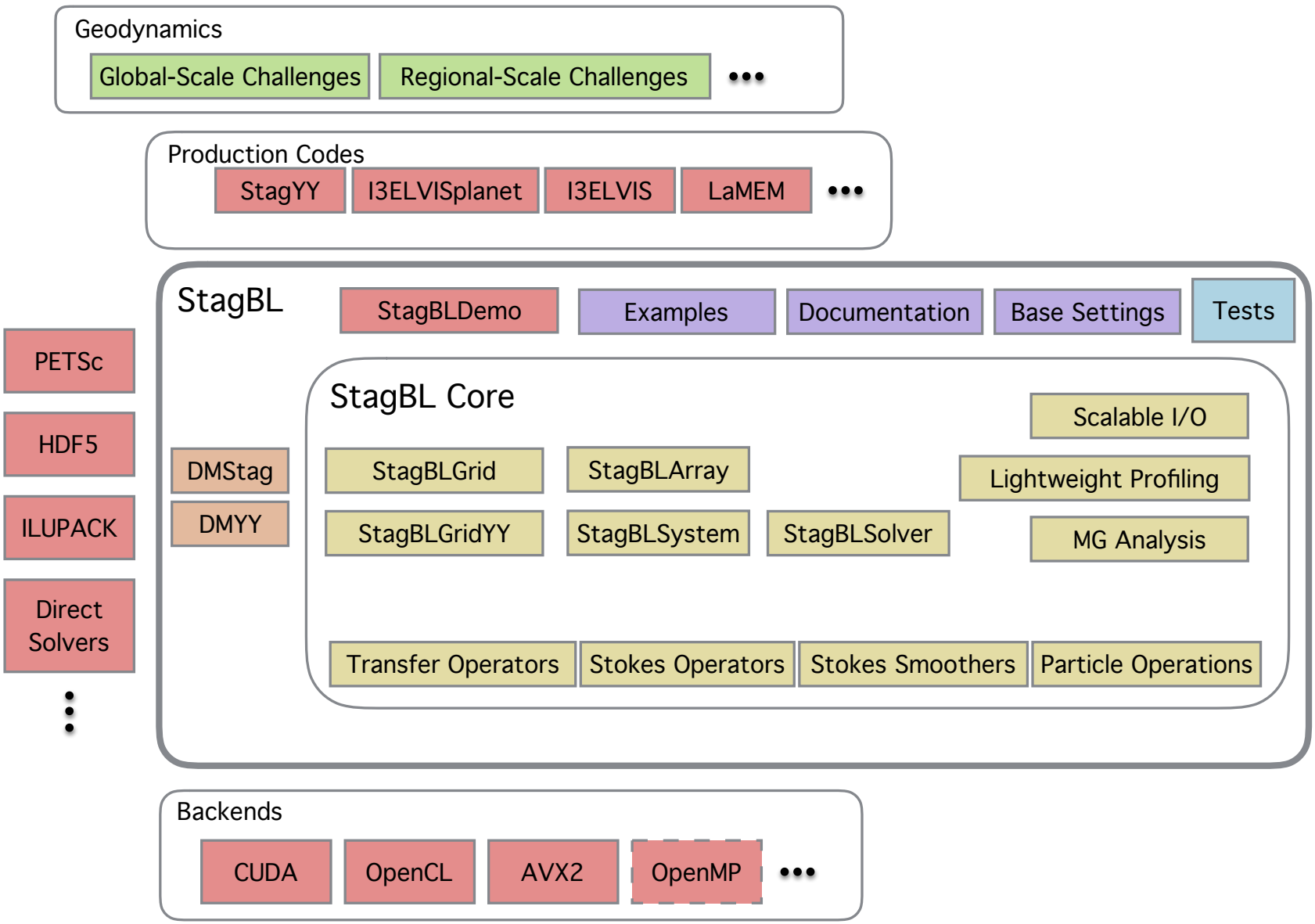
- Central driver: simplicity
- Written in C, with simple object-oriented design
- Focus on components required for efficient, flexible Stokes solvers used by MAC-style geodynamics applications
- Use external libraries whenever possible
- Four core classes
- Modern software engineering practices (version control, portable build, tests)
- Configure/build: simple GNUMake-based approach, inspired by HPGMG[8]
- Testing: simple, custom test harness[9], designed for ease of testing MPI-based scientific codes on local machines and clusters with batch systems
- Document by example, using included demo mini-apps

[8] https://hpgmg.org

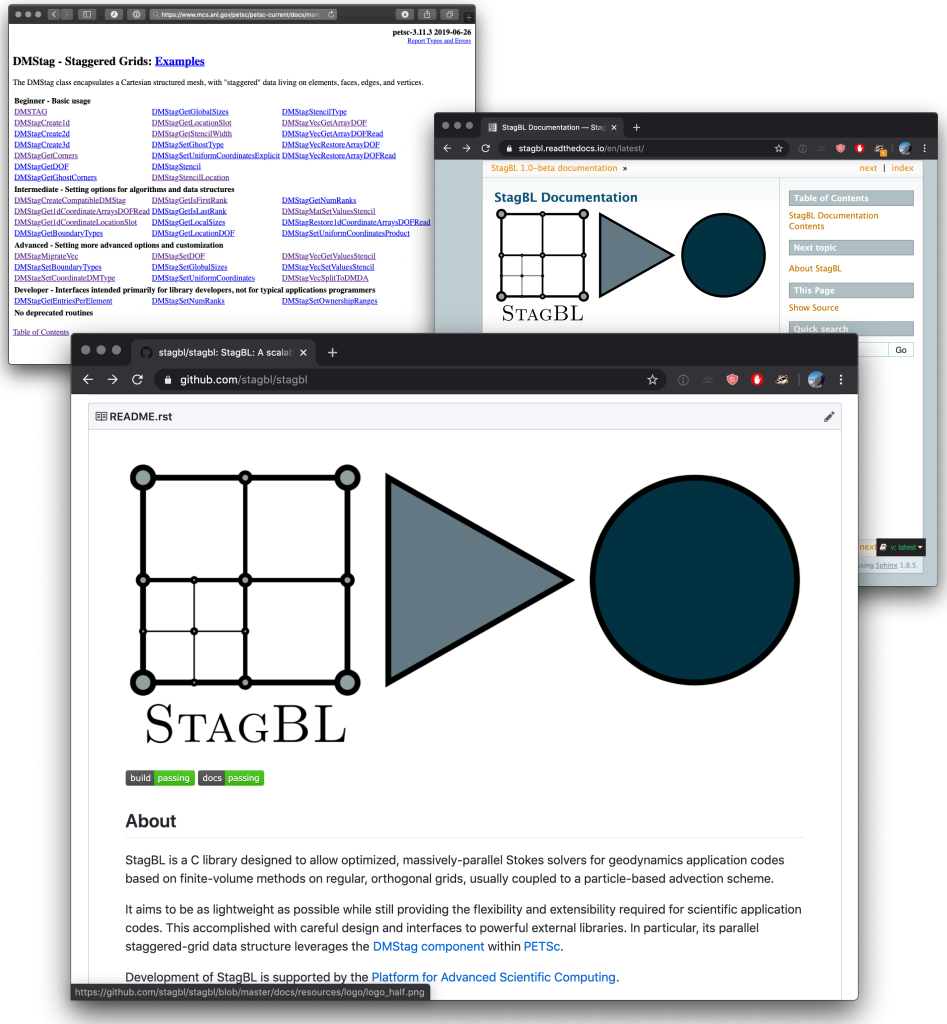
[9] With Dave May, https://bitbucket.org/dmay/pythontestharness (big update underway)

StagBL Project: Components





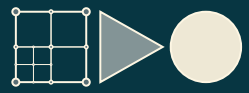
StagBL Website



github.com/stagbl/stagbl

DMStag

DMStag and PETSc Integration

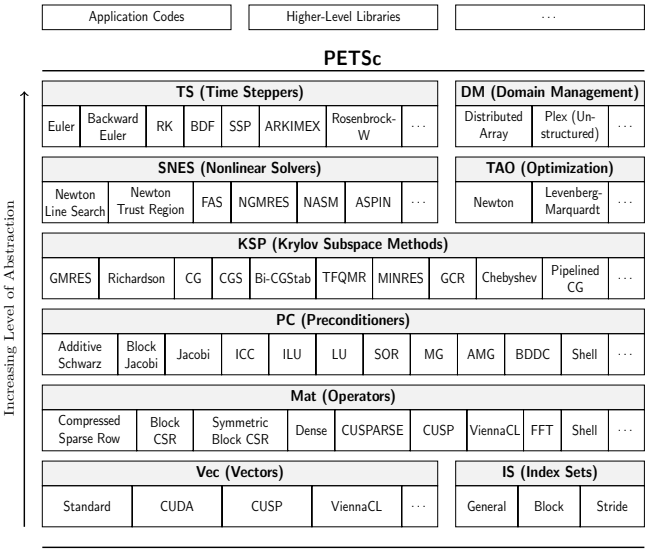


- StagBL heavily leverages capabilities from the PETSc library
- The central object is `DMStag`, a new `DM` implementation
- Available in PETSc 3.11[10] and under active development [11].
- We make heavy use of the notion of "compatibility" to quantify the concept of two objects objects representing "different data on the same domain".

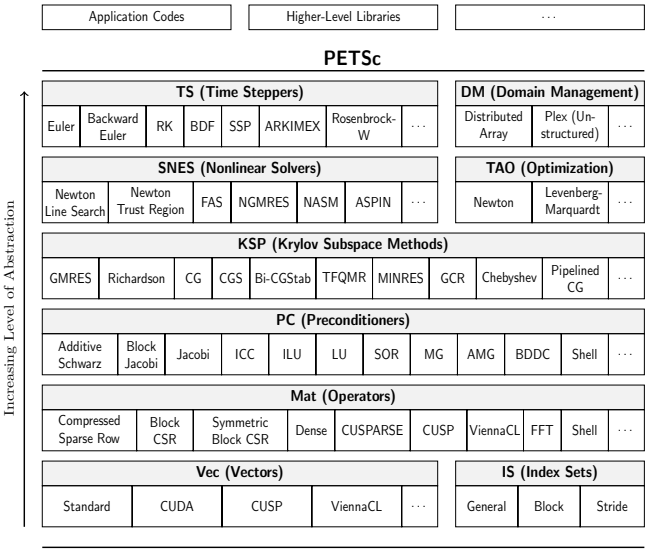
[10] See https://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/DMSTAG/index.html

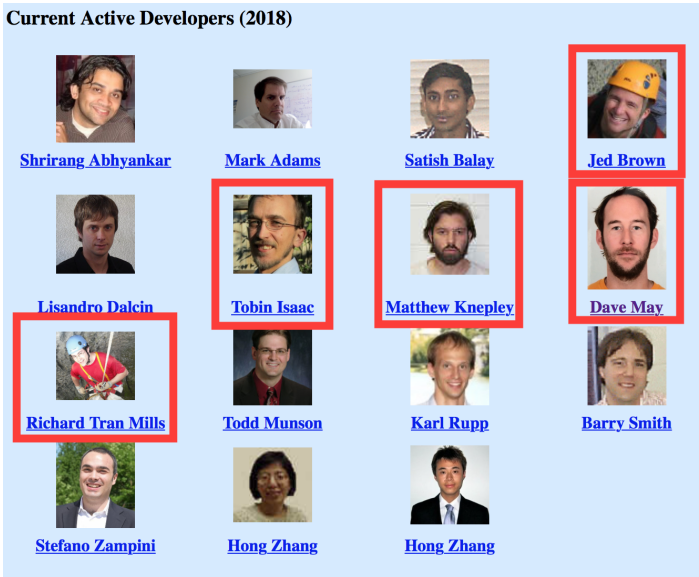
[11] contact patrick.sanan@erdw.ethz.ch with feature requests, etc.

PETSc



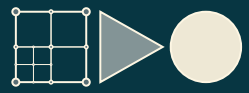
- "**P**ortable, **E**xtensible **T**oolkit for **S**cientific **c**omputation (**P**ortable, **E**xtensible **T**oolkit for **S**olver **c**omposition?)
- Efficient, scalable, parallel linear linear algebra and solvers
- Open-source (2-clause BSD) C library (+ Fortran/Python bindings), built on MPI
- Developed and (well-)supported for over 20 years at ANL, with an active community
- Large, highly configurable, can download and install dependencies





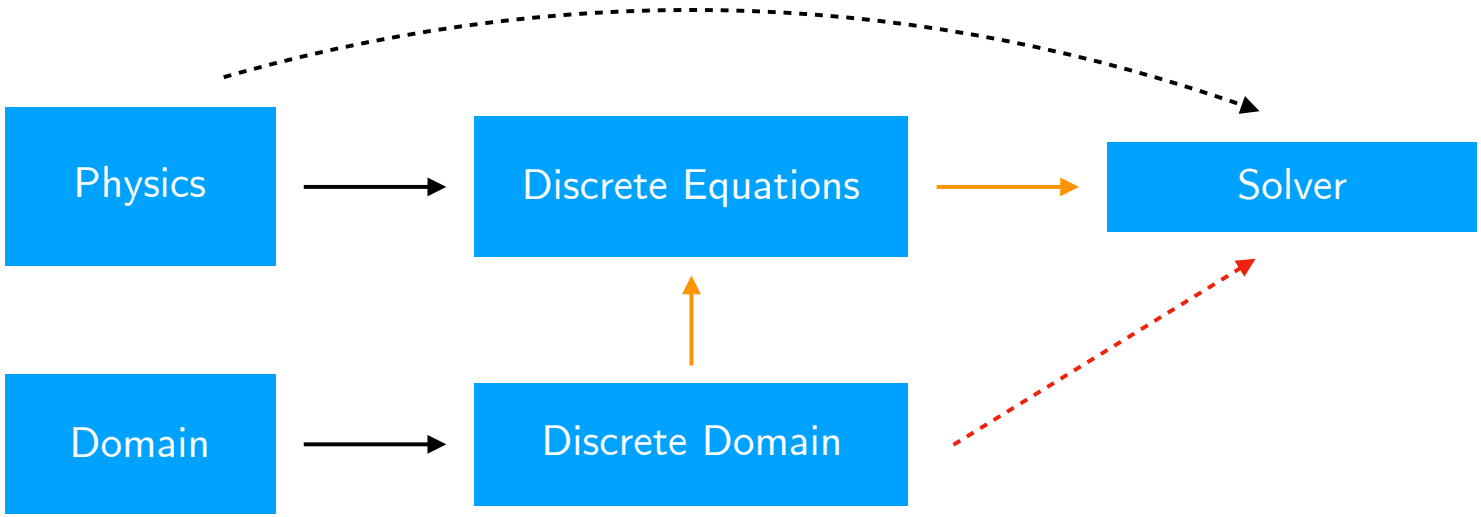
- "**P**ortable, **E**xtensible **T**oolkit for **S**cientific **c**omputation (**P**ortable, **E**xtensible **T**oolkit for **S**olver **c**omposition?)
- Efficient, scalable, parallel linear linear algebra and solvers
- Open-source (2-clause BSD) C library (+ Fortran/Python bindings), built on MPI
- Developed and (well-)supported for over 20 years at ANL, with an active community
- Large, highly configurable, can download and install dependencies

Why does a solver library need grids? DM



- Efficient or optimal solvers for systems of equations arising from PDE almost invariably require information (or assumptions) about the topology of the problem domain
- For PDE representing local physics, this is often provided by describing the topology of the domain
- The motivating example is geometric multigrid
- In computational science, many researchers get stuck at the point where direct solvers (which are very close to "black boxes") do not scale well enough
- PETSc defines a base class, `DM` (**D**omain **M**anager), to encapsulate information required by scalable solvers
- Different *implementations* of this uniform interface represent different types of discrete domain.
- Key examples are `DMDA` (regular, collocated grid), `DMPlex` (unstructured cell complexes), `DMSwarm` (particles)

Why does a solver library need grids? DM



- More information about the problem ("structure") can help solvers perform better
- Adding the dotted red line helps greatly when combined with the discrete equations (and the physics)
- Algebraic multigrid attempts to reconstruct this information from the discrete equations (the matrix)

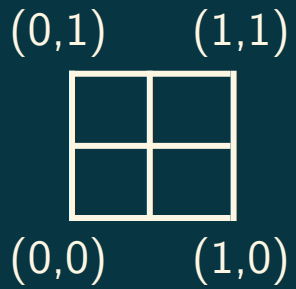
What's in a DM?

A working definition of a DM:

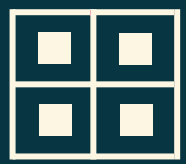
DM

Topology (mesh)

Atlas (parallel decomposition)



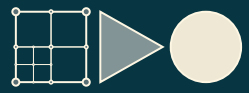
Embedding (coordinates)



Section (fields)

- Note: different sections (fields, data, ..) on the "same domain" require different DMs.

Hands on: DMDA (parallel, regular grid)



- When your problem is set up using a DM, you can test, and hopefully use, advanced parallel solvers
- Let's experiment with the Poisson equation, using PETSc's default linear solver (robust, but not optimal)

```
# Define PETSC_DIR and PETSC_ARCH
cd $PETSC_DIR/src/ksp/ksp/examples/tutorials
make ex50
./ex50 -da_refine 5 -ksp_view -ksp_monitor \
  -ksp_converged_reason
```

- We can do better with algebraic multigrid

```
./ex50 -da_refine 5 -ksp_view -ksp_monitor \
  -ksp_converged_reason -pc_type gamg
```

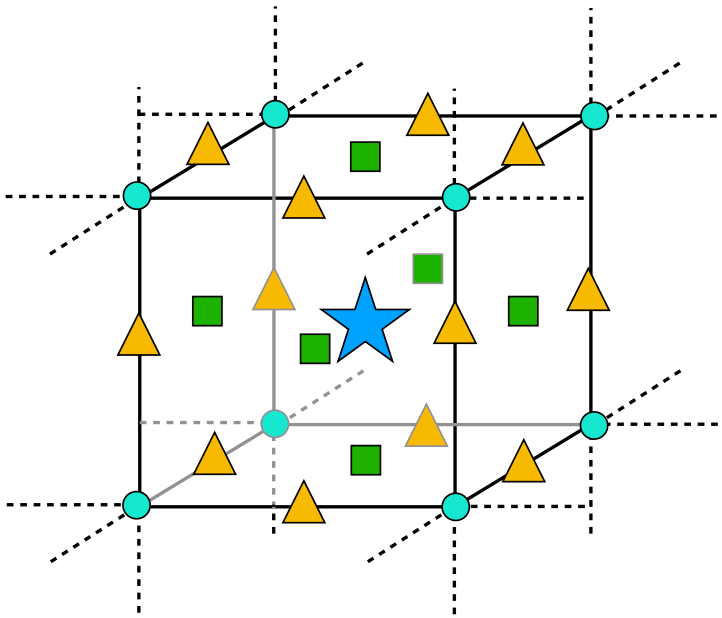
- And even better with geometric multigrid:

```
./ex50 -da_refine 5 -ksp_view -ksp_monitor \
  -ksp_converged_reason -pc_type mg -pc_mg_levels 5 \
  -pc_mg_type full  # add -log_view for timing!
```

- How did it work? The DM helped create the multigrid components!

DMStag

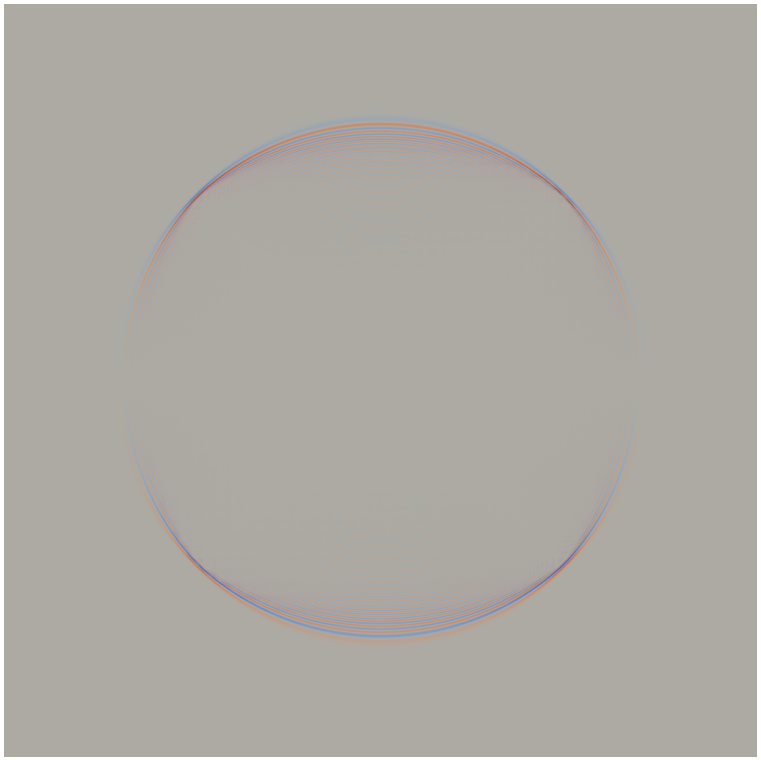
- A generalization of DMDA for staggered grids
- Why not just use DMDA(s) for staggered grid problems?
  - Must either use dummy/ghost points with a multi-dof collocated grid, plus extra indexing (I3ELVIS), or..
  - Multiple DMDAs, plus extra indexing (LaMEM)
  - Difficult to present a uniform interface to solvers



Tutorial Example

DMStag tutorial ex6: Seismic wave propagation

- We'll work with an existing example, and use it to demonstrate the features of DMStag
- With each topic, we'll modify the example to do something new
- Disclaimer: I am not a seismologist!



Example Code

- The code is found at `$PETSC_DIR/src/dm/impls/stag/examples/tutorials/ex6.c`
- It shows an example of taking a published staggered-grid algorithm and implementing it
- The application is seismic wave propagation, from Virieux's 1986 paper[12] (Thanks to B. Kaus for suggesting this as an example)

$$\frac{\partial v_x}{\partial t} = b\left(\frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xz}}{\partial z}\right),$$

$$\frac{\partial v_z}{\partial t} = b\left(\frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{zz}}{\partial z}\right),$$

$$\frac{\partial \tau_{xx}}{\partial t} = (\lambda + 2\mu)\frac{\partial v_x}{\partial x} + \lambda\frac{\partial v_z}{\partial z}, \qquad (2)$$

$$\frac{\partial \tau_{zz}}{\partial t} = (\lambda + 2\mu)\frac{\partial v_z}{\partial z} + \lambda\frac{\partial v_x}{\partial x},$$

and

$$\frac{\partial \tau_{xz}}{\partial t} = \mu\left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right)$$

[12] Jean Virieux. "P-SV Wave Propagation in Heterogeneous Media: Velocity-Stress Finite-Difference Method". *Geophysics* 51.4 (1986), pp. 889–901.

Example Code

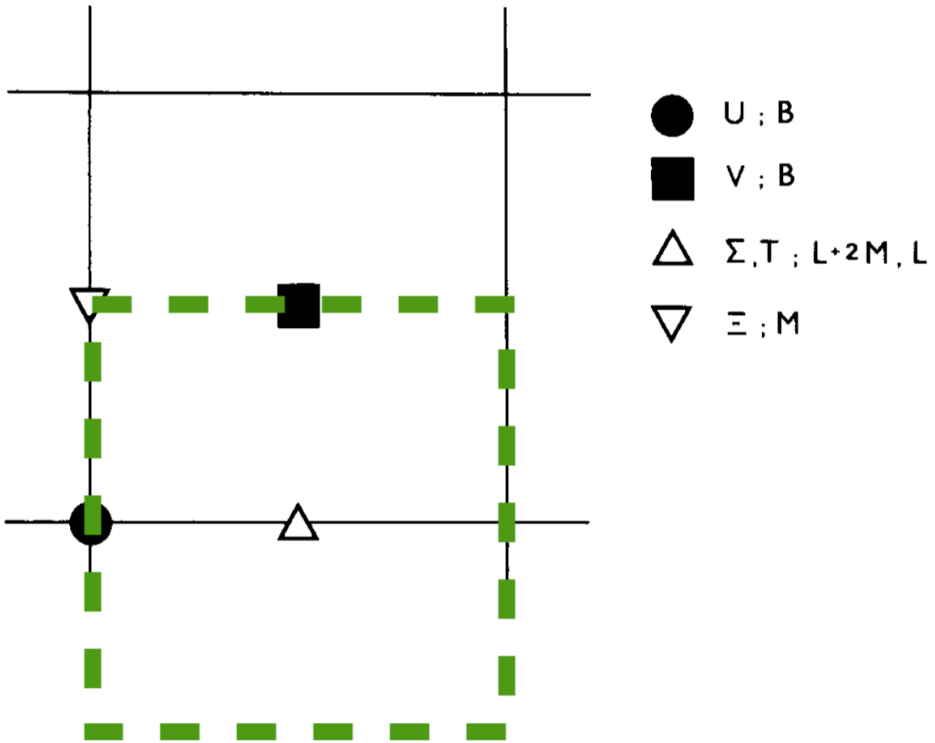


FIG. 1. Discretization of the medium on a staggered grid. Black symbols are for velocities and buoyancy at time $k\Delta t$. White symbols are for stresses and Lamé coefficients at time $(k + 1/2)\Delta t$.

(modified to add a more natural control volume)

$$U_{i,j}^{k+1/2} = U_{i,j}^{k-1/2} + B_{i,j}\frac{\Delta t}{\Delta x}(\Sigma_{i+1/2,j}^{k} - \Sigma_{i-1/2,j}^{k}) + B_{i,j}\frac{\Delta t}{\Delta z}(\Xi_{i,j+1/2}^{k} - \Xi_{i,j-1/2}^{k}),$$

$$V_{i+1/2,j+1/2}^{k+1/2} = V_{i+1/2,j+1/2}^{k-1/2} + B_{i+1/2,j+1/2}\frac{\Delta t}{\Delta x}(\Xi_{i+1,j+1/2}^{k} - \Xi_{i,j+1/2}^{k}) + B_{i+1/2,j+1/2}\frac{\Delta t}{\Delta z}(T_{i+1/2,j+1}^{k} - T_{i+1/2,j}^{k}),$$

$$\Sigma_{i+1/2,j}^{k+1} = \Sigma_{i+1/2,j}^{k} + (L+2M)_{i+1/2,j}\frac{\Delta t}{\Delta x}(U_{i+1,j}^{k+1/2} - U_{i,j}^{k+1/2}) + L_{i+1/2,j}\frac{\Delta t}{\Delta z}(V_{i,j+1}^{k+1/2} - V_{i,j}^{k+1/2}),$$

$$T_{i+1/2,j}^{k+1} = T_{i+1/2,j}^{k} + (L+2M)_{i+1/2,j}\frac{\Delta t}{\Delta z}(V_{i,j+1}^{k+1/2} - V_{i,j}^{k+1/2}) + L_{i+1/2,j}\frac{\Delta t}{\Delta x}(U_{i+1,j}^{k+1/2} - U_{i,j}^{k+1/2}),$$

and

$$\Xi_{i,j+1/2}^{k+1} = \Xi_{i,j+1/2}^{k} + M_{i,j+1/2}\frac{\Delta t}{\Delta z}(U_{i,j+1}^{k+1/2} - U_{i,j}^{k+1/2})$$

Example Code

To run the code, you must first have a working build of PETSc (which you hopefully already do!)

```
export PETSC_DIR=xxx
export PETSC_ARCH=yyy
cd $PETSC_DIR/src/dm/impls/stag/examples/tutorials/ex6.c
make ex6
rm -f *.vtr && $PETSC_DIR/$PETSC_ARCH/bin/mpiexec -np 4 ./ex6
```

Examine the resulting `.vtr` files in Paraview

Creation and Setup

Creation

High-level functions create new DMStag objects.

```
ierr = DMStagCreate2d(
  PETSC_COMM_WORLD,
  DM_BOUNDARY_NONE,DM_BOUNDARY_NONE, /* Boundary types           */
  7,9,                               /* Global sizes             */
  PETSC_DECIDE,PETSC_DECIDE,         /* Local sizes              */
  dof0,dof1,dof2,                    /* dof per stratum:
                                          0: vertices
                                          1: edges
                                          2: faces
                                          3: hexes                */
  DMSTAG_GHOST_STENCIL_STAR,         /* Elementwise ghosting     */
  stencilWidth,                      /* Elementwise ghost width  */
  NULL,NULL,                         /* explicit decomposition   */
  &my_dmstag
);CHKERRQ(ierr);
```

Manual pages for all DMStag functions can be found on the web

- Release: https://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/DMSTAG/index.html
- Development (latest): https://www.mcs.anl.gov/petsc/petsc-dev/docs/manualpages/DMSTAG/index.html

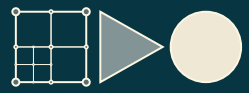
Creation: Exercise

The call to `DMSetFromOptions()` allows many options to be specified with command line flags, and we also add our own.

- From the command line,
  - Get short (`-help intro`) or long (`-help`) sets of options
  - more timesteps (`-nsteps`)
  - A different grid size (`stag_grid_x 50 -stag_grid_y 50`
  - Try different parallel decompositions (`-stag_ranks_x 1`)
  - Investigate how performance changes (`-log_view`)

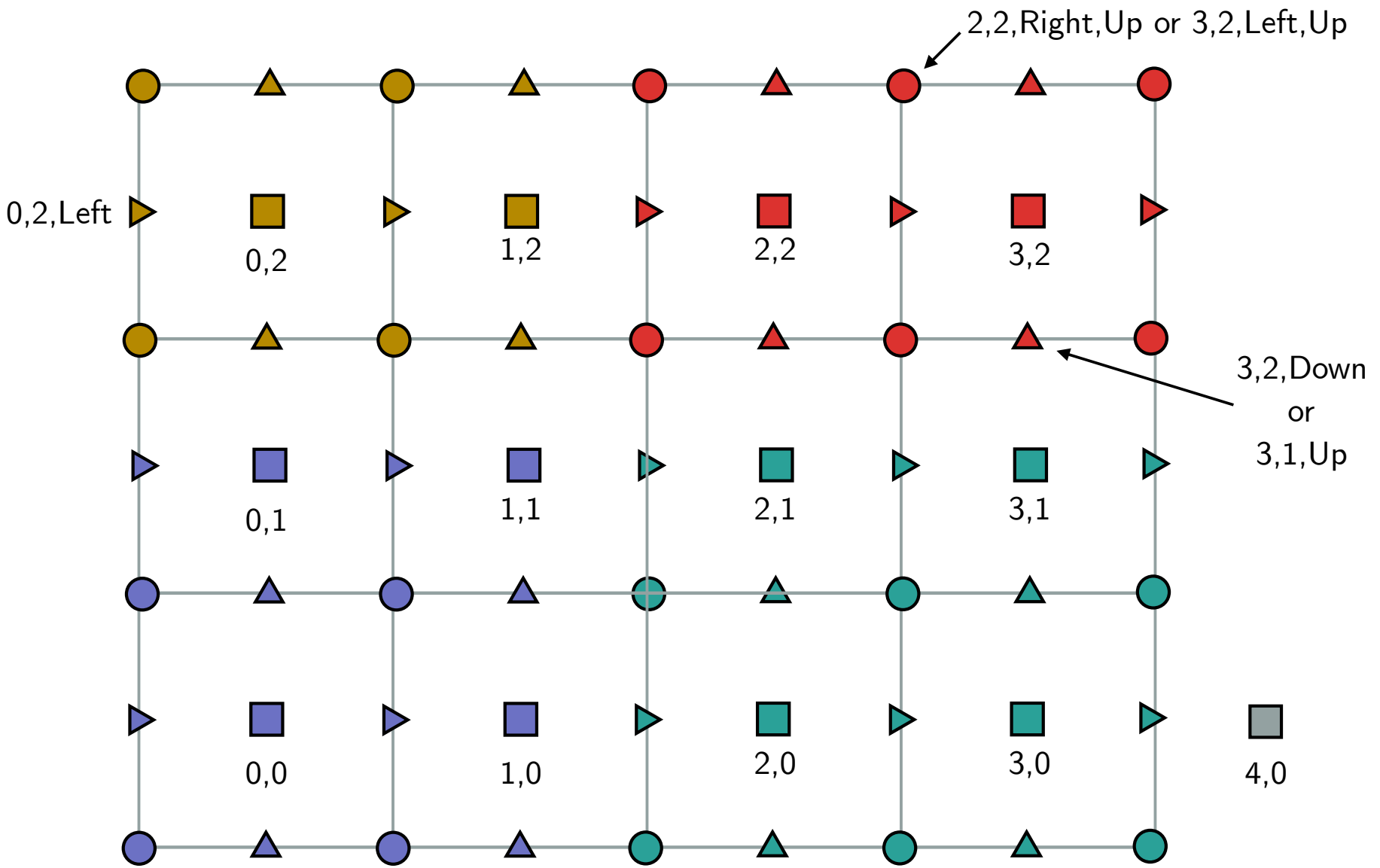
Local and Global Representations

DMStag Local-Global Wrinkles



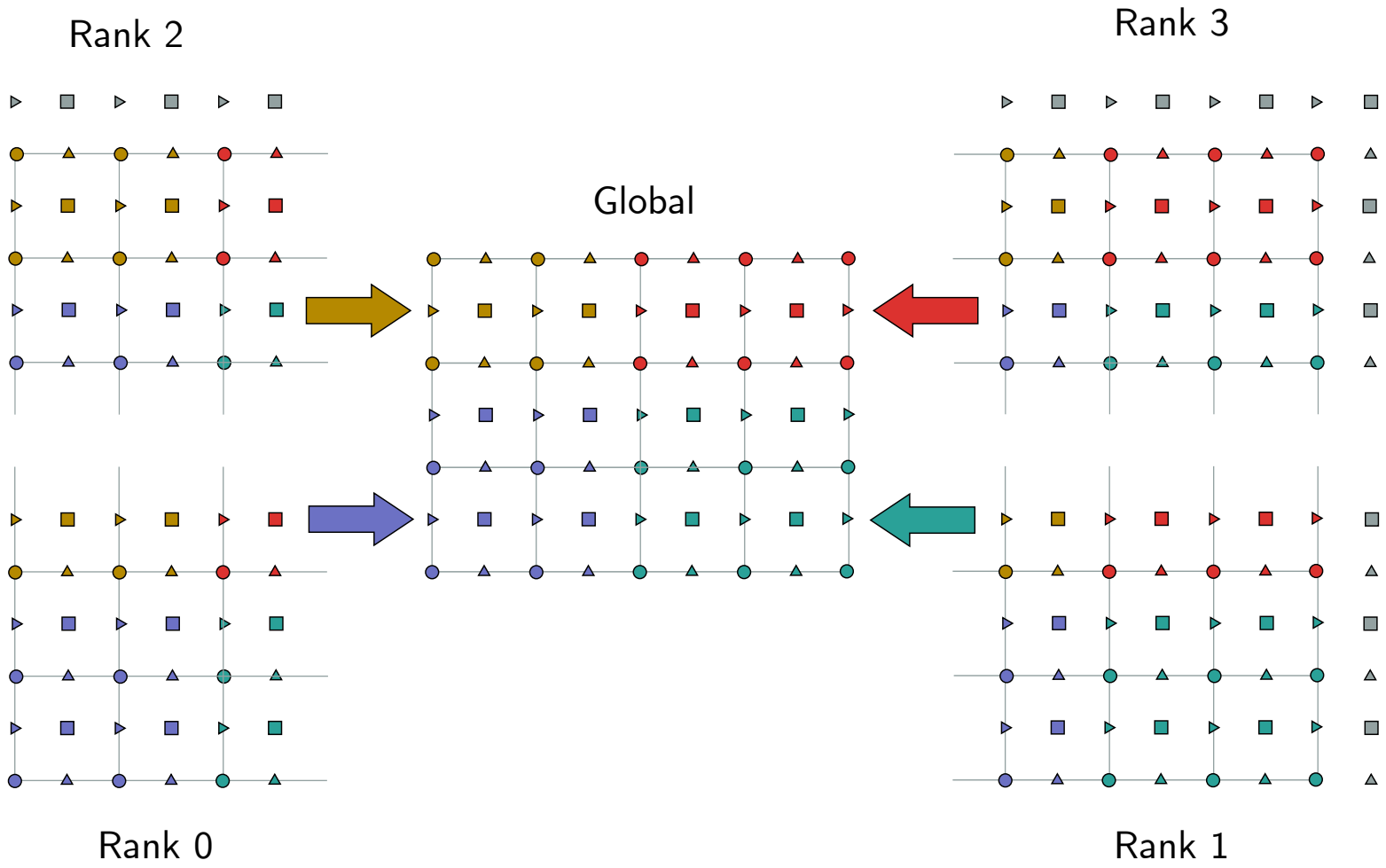
- The local representation is uniformly-blocked, but has "dummy" points
- The global representation is not uniformly-blocked, but only contains dof corresponding to actual cells in the cell complex
- "Think globally, act locally"
- Based on the experience of users with PETSc's existing local-global paradigm, performance bottlenecks related to redundant local-global information aren't as common as one might expect, but an available future optimization is implementing `DMLocalToLocal()` for `DMStag`.

DMStag Natural (user-facing) indexing



Not shown: components per stratum (vertices, edges, faces, hexes)

Internal Local and Global Indexing (1 dof/stratum)



* 3D: similar but more difficult to draw

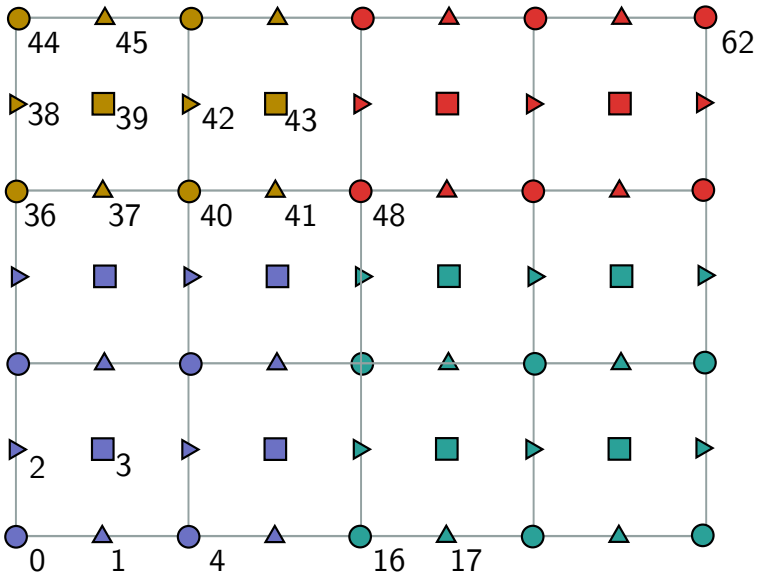
Internal Local and Global Indexing (1 dof/stratum)

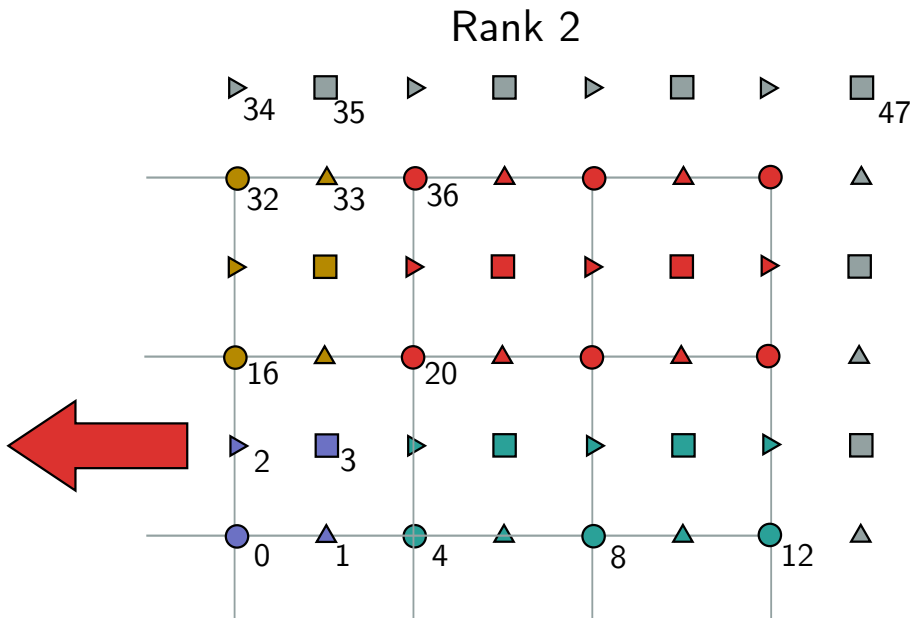
Global

- Global numbering of points (and thus dof)
- No "non-physical" points
- Each point "lives" on an MPI rank (color)

Local

- Local (0-based) numbering of points
- Includes "ghost" points
  - for values "living" on other ranks
  - to create regular blocking ("dummies")





Hands on: Local and Global Representations

Examine the raw data in the local and global representations. How are the sizes and entries different?

- Run a single timestep, for a small problem

```
./ex6 -nsteps 1 -stag_grid_x 4 -stag_grid_z 4
```

- View a local coefficient vector by adding e.g.

```
ierr = VecView(lame_local,PETSC_VIEWER_STDOUT_WORLD);CHKERRQ(
    ierr);
```

- View the global coefficient vector by adding e.g.

```
ierr = VecView(lame_local,PETSC_VIEWER_STDOUT_WORLD);CHKERRQ(
    ierr);
```

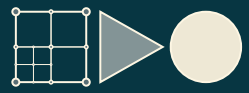
- Repeat in parallel. Use the write `mpiexec`. If you used `--download-mpich`, you can use

```
$PETSC_DIR/$PETSC_ARCH/bin/mpiexec -np 4 ./ex6
```

Bonus: try to do the same without changing the code, using a debugger (`gdb` or `lldb`) and calling `VecView(vec,0)` directly.

Working with arrays

Easier indexing with stencils!



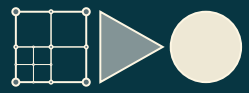
- Key feature in `DMStag` : `DMStagStencil` (akin to `MatStencil`)
- Language like "The edge above element (i, j)" better than raw indices.
- It simply holds 5 integers: which element, which point, which dof.

```
typedef struct {
  DMStagStencilLocation loc; /* enum */
  PetscInt              i,j,k,c;
} DMStagStencil;
```

- API uses these to translate between global element numbers/locations/components and local indices, for both vectors and matrices.

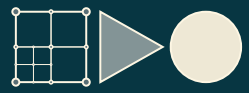
```
PetscErrorCode DMStagVecGetValuesStencil(
                 DM dm,Vec vec, PetscInt n,
                 const DMStagStencil* pos,
                 PetscScalar*);
```

Example: compute element-averaged velocities



```
PetscInt ex,ey,startx,starty,nx,ny;
Vec      stokesLocal;

ierr = DMStagCreateCompatibleDMStag(dmStokes,0,0,2,0,&dmVelAvg);CHKERRQ(ierr);
ierr = DMSetUp(dmVelAvg);CHKERRQ(ierr);
ierr = DMCreateGlobalVector(dmVelAvg,&velAvg);CHKERRQ(ierr);
ierr = DMGetLocalVector(dmStokes,&stokesLocal);CHKERRQ(ierr);
ierr = DMGlobalToLocalBegin(dmStokes,x,INSERT_VALUES,stokesLocal);CHKERRQ(ierr);
ierr = DMGlobalToLocalEnd(dmStokes,x,INSERT_VALUES,stokesLocal);CHKERRQ(ierr);
ierr = DMStagGetCorners(dmVelAvg,&startx,&starty,NULL,&nx,&ny,NULL,NULL,NULL,NULL);
     CHKERRQ(ierr);
for (ey = starty; ey<starty+ny; ++ey) {
  for (ex = startx; ex<startx+nx; ++ex) {
    DMStagStencil from[4],to[2];
    PetscScalar   valFrom[4],valTo[2];
    from[0].i = ex; from[0].j = ey; from[0].loc = DMSTAG_UP;    from[0].c = 0;
    from[1].i = ex; from[1].j = ey; from[1].loc = DMSTAG_DOWN;  from[1].c = 0;
    from[2].i = ex; from[2].j = ey; from[2].loc = DMSTAG_LEFT;  from[2].c = 0;
    from[3].i = ex; from[3].j = ey; from[3].loc = DMSTAG_RIGHT; from[3].c = 0;
    ierr = DMStagVecGetValuesStencil(dmStokes,stokesLocal,4,from,valFrom);CHKERRQ(ierr);
    to[0].i = ex; to[0].j = ey; to[0].loc = DMSTAG_ELEMENT; to[0].c = 0;
    valTo[0] = 0.5 * (valFrom[2] + valFrom[3]);
    to[1].i = ex; to[1].j = ey; to[1].loc = DMSTAG_ELEMENT; to[1].c = 1;
    valTo[1] = 0.5 * (valFrom[0] + valFrom[1]);
    ierr = DMStagVecSetValuesStencil(dmVelAvg,velAvg,2,to,valTo,INSERT_VALUES);CHKERRQ(
         ierr);
  }
}
ierr = VecAssemblyBegin(velAvg);CHKERRQ(ierr);
ierr = VecAssemblyEnd(velAvg);CHKERRQ(ierr);
ierr = DMRestoreLocalVector(dmStokes,&stokesLocal);CHKERRQ(ierr);
```

Direct Array Access



- Direct array access often very efficient (no `VecSetValues()` overhead, can apply complex kernels, etc.) and even readable.
- Local vectors have an integral numbers of entries/element
- Access with `DMStagVecGetArray()` and friends
- Use helper functions to determine which indices to use

```
PetscScalar ****arr;
PetscInt    ivx;
/* ... */
DMStagGetLocationSlot(dm,DMSTAG_RIGHT,0,&ivx);
DMStagVecGetArray(dm,vec_local,&arr);
/* ... */
arr[k][j][i][ivx] = computeXVelocity();
/* ... */
DMStagVecRestoreArray(dm,vec_local,&arr);
```

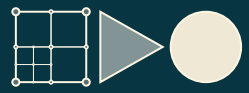
- Note that in the above snippet, `ivx` has a gridsize-dependent value!

Hands-on: Array Access

Modify the routine `CreateLame()` to use direct array access, instead of `DMStagVecSetValuesStencil()`.

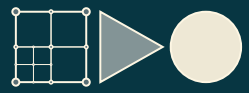
Coordinates

Product Coordinates and DMProduct



- Many applications rely on grids which are orthogonal, hence allowing for compressed representation of coordinates
- Example: for a 3D problem with N^3 grid points, directly storing coordinates requires $3N^3$ total entries, but if decomposed on P^3 ranks and represented with local products, only requires $3N/P * P^3 = 3NP^2$ entries, a reduction of $(N/P)^2$.
- Our 3 geodynamics applications all use orthogonal grids
- Examples exist e.g. in climate or ice sheet modelling which use an unstructured grid over a surface with identical 1-d columns extruding each 2d grid cell.
- Hence introduce a lightweight DM, DMProduct, to keep track of a product of DMs on each rank
- Allow DMStag to generate its coordinates as products, e.g. with DMStagSetUniformCoordinatesProduct().

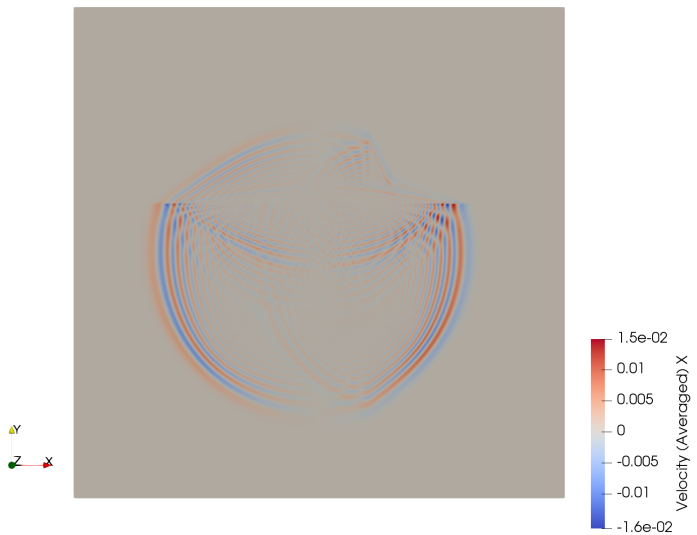
Coordinates direct array access



```
PetscScalar ***arr; /* 3-d array for 2-D problem */
/* ... */
ierr = DMStagGetCorners(dm,&startx,&starty,NULL,&nx,&ny,NULL,&extrax,&extray,NULL);
     CHKERRQ(ierr);

/* Get slot numbers for efficient local array access */
ierr = DMStagGetProductCoordinateArraysRead(dm,&arr_coordinates_x,&arr_coordinates_y,&
     arr_coordinates_z);CHKERRQ(ierr);
ierr = DMStagGetProductCoordinateLocationSlot(dm,DMSTAG_ELEMENT,&slot_center);CHKERRQ(
     ierr);
ierr = DMStagGetLocationSlot(dm,DMSTAG_ELEMENT,0,&slot_element_coefficient);CHKERRQ(ierr)
     ;

/* Get temporary access to local vector and raw array */
ierr = DMGetLocalVector(dm,vec_local);CHKERRQ(ierr);
ierr = DMStagVecGetArraydm,vec,&arr);CHKERRQ(ierr);

/* Element values */
for (ey = starty; ey < starty + ny; ++ey) {
  for (ex = startx; ex < startx + nx; ++ex) {
    const PetscReal x = arr_coordinates_x[ex][slot_center];
    const PetscReal y = arr_coordinates_y[ey][slot_center];

    arr[ey][ex][slot_element_coefficient] = some_function(x,y);
  }
}
ierr = DMStagVecRestoreArray(dm,vec_local,&arr);CHKERRQ(ierr);

/* Map to global vector and release local vector */
ierr = DMLocalToGlobal(dm,vec_local,INSERT_VALUES,vec_global);CHKERRQ(ierr);
ierr = DMRestoreLocalVector(dm,&vec_local);CHKERRQ(ierr);
```

Hands on: Variable coefficients

- Modify `ex6.c` to use different elastic coefficients on part of the domain, in 2d.
- You can do this first by examining the element number relative to the total number of elements, or more realistically by gaining direct access to coordinates.
- I will demonstrate doing the latter, using direct array access.

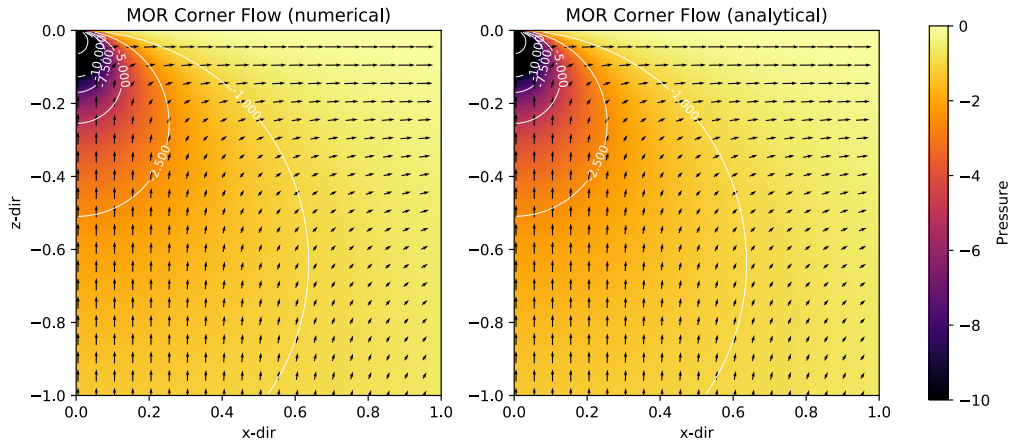



```
./ex61 -nsteps 500 -variable -stag_grid_x 300 -stag_grid_y 300 -dt
    4e-4
```

Output

Output

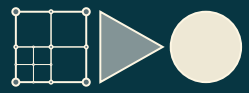
- Output is done in a roundabout way, which works but isn't ideal
- Functions are provided to convert data associated with a DMStag to data associated with a DMDA
- Existing output functionality is used to generate files
- PETSc itself isn't heavily concerned with output, so applications (including StagBL) should explore other options
- Promising approach in 2D, thanks to Adina Püsök and Dave May: using PETSc's Python-loading scripts, and some custom code to work with data living on DMStag's.



A. Püsök, University of Oxford

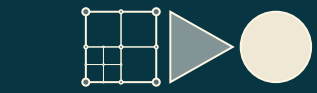
StagBL and StagBLDemo

Building an Application on to of DMStag

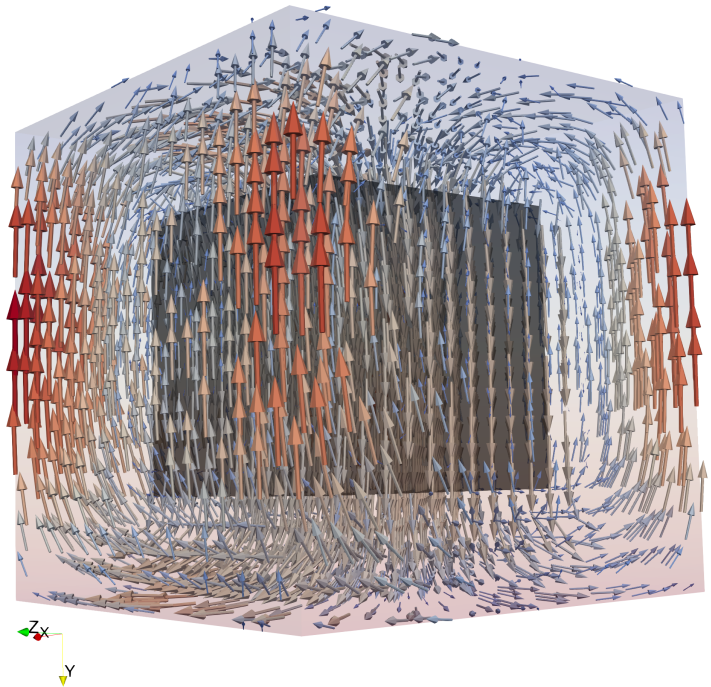


- StagBL is a library, built on top of PETSc, including our new DMStag component, to make it easier to leverage advanced solvers for staggered-grid geodynamics codes.
- Currently, DMStag does most of the heavy lifting, but the advantage of the StagBL library is that it can provide less-general, and thus easier-to-use, functions (though it currently does not do a great job of this).
- It includes a demo application, to show how its components may be used. We will use this demonstration to introduce topics that we didn't get to in the previous lecture, on DMStag:
  - Stokes systems
  - Stokes Solvers
  - Interaction with Particle systems (DMSwarm)

Demonstration Code

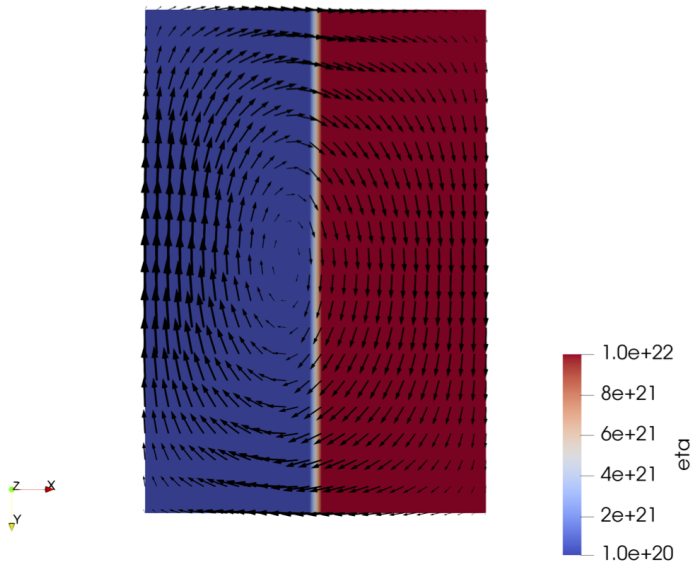


- StagBL contains a demonstration code, which we'll be using.
- While application codes using StagBL do not have to be based on PETSc (even though StagBL depends on it), we use PETSc to help us quickly write this demo.



Running the Code

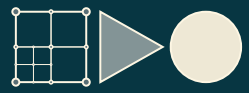
- Obtain and build a custom version of PETSc, and StagBL, as described at https://github.com/stagbl/stagbl
- Run it, as described there, and confirm that you can get the expected output.



- Run it, trying to reproduce the Blankenbach benchmark, using the instructions at https://stagbl.readthedocs.io/en/latest/benchmarks.html

Constructing a Stokes System

Using DMStag to Create an Operator



StagBL currently uses DMStag directly to construct Stokes operators in a simple case.

```
% From src/stokes/stokes.c
PetscErrorCode StagBLCreateStokesSystem(StagBLStokesParameters parameters, StagBLSystem *
    system)
{
  PetscErrorCode ierr;
  DM             dm_stokes;
  PetscInt       dim;

  PetscFunctionBegin;
  ierr = StagBLGridPETScGetDM(parameters->stokes_grid,&dm_stokes);CHKERRQ(ierr);
  ierr = DMGetDimension(dm_stokes,&dim);CHKERRQ(ierr);
  ierr = StagBLGridCreateStagBLSystem(parameters->stokes_grid,system);CHKERRQ(ierr);
  switch (dim) {
    case 2:
    ierr = CreateSystem_2D_FreeSlip(parameters,*system);CHKERRQ(ierr);
  break;
    case 3:
    ierr = CreateSystem_3D_FreeSlip(parameters,*system);CHKERRQ(ierr);
    break;
    default: StagBLError1(PetscObjectComm((PetscObject)dm_stokes),"Unsupported dimension
        %D",dim);
  }
  PetscFunctionReturn(0);
}
```

Let's take a look at this function and how it's used in the demo.

Solving the Stokes System

Solving the Stokes System

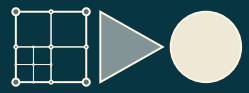
- Once the system is assembled, we have the power of PETSc
- We wrap the solver object in our core `StagBLSolver` class (see https://stagbl.readthedocs.io/en/latest/benchmarks.html)
- Let's look at all the solvers from the command line[13]

```
./stagbldemo2d -ksp_view
```

- In the simplest case (the current demo), we can use a direct solver
- Let's see how this is done
- However, the real point of StagBL is to enable advanced solvers, as we saw early with our DMDA multigrid example, so let's look at some work-in-progress..

[13] currently not using options prefixes correctly!

Solving the Stokes System

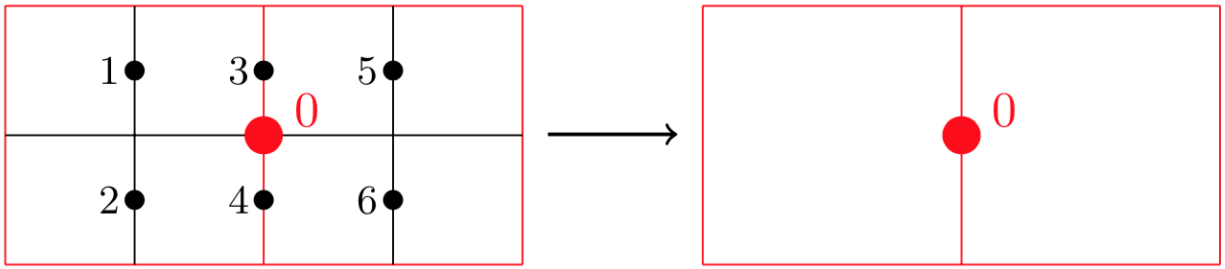


- Once the system is assembled, we have the power of PETSc
- In the simplest case (the current demo), we can use a direct solver
- Let's see how this is done
- However, the real point of StagBL is to enable advanced solvers, as we saw early with our DMDA multigrid example..

Multigrid Components

Optimal transfer operators will vary with the particular problem, but we provide widely-used choices (e.g.[14]) appropriate for MAC Stokes flow.

1. Injection/averaging for element-based values (pressures)
2. Interpolation on faces (velocities) is linear for faces overlaying coarse faces, and bilinear otherwise
3. Restriction for faces is also bilinear



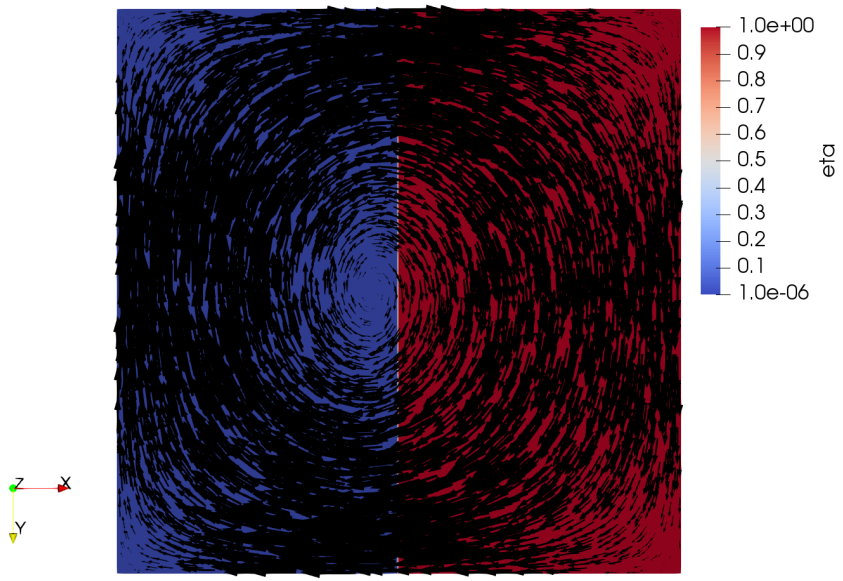
(A) Fine grid (B) Coarse grid

15

[14] 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, App. C.

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

ABF + Multigrid Test Case

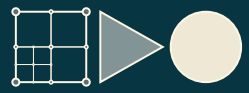


$$\eta_1 \longleftrightarrow \eta_2$$

- DMStag tutorial ex4, based on textbook exercise[16]
- Variable-viscosity bouyancy-driven Stokes flow, free-slip boundary conditions
- modified to
  - Incorporate proper pressure nullspace
  - Use non-dimensional units
  - Construct diagonal auxiliary matrix, with entries as inverse viscosity

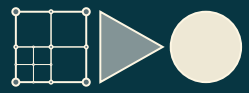
[16] Taras Gerya. *Introduction to Numerical Geodynamic Modelling*. 1st. Cambridge University Press, 2009, ex. 7.2.

ABF + Multigrid Test Case: Isoviscous



```
-nondimensional
-isoviscous
-stag_grid_x 256
-stag_grid_y 256
-ksp_type fgmres
-pc_type fieldsplit
-pc_fieldsplit_type schur
-pc_fieldsplit_schur_fact_type upper
-pc_fieldsplit_schur_precondition user #  Diagonal inverse-
    viscosity approximation
-fieldsplit_element_ksp_type preonly
-fieldsplit_element_pc_type jacobi
-fieldsplit_face_pc_type mg
-fieldsplit_face_pc_mg_levels 5
-fieldsplit_face_pc_mg_galerkin
-fieldsplit_face_mg_levels_pc_type jacobi
-fieldsplit_face_mg_levels_ksp_type chebyshev
```

ABF + Multigrid Test Case: Isoviscous



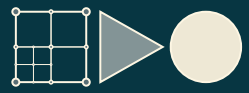
Set $\eta_1 = \eta_2 = 1$ and run on Piz Daint

Grid	Ranks	MG Levels	FGMRES Its. (10^{-6} rel. tol.)	`KSPSolve` tim
256^2	4	5	5	0.23
512^2	16	6	5	0.32
1024^2	64	7	5	0.42
2048^2	256	8	5	1.06
4096^2	1024	8*	7	1.98

* At this point, one should use `PCTelescope`[17].

[17] Dave A. May, Patrick Sanan, Karl Rupp, Matthew G. Knepley, and Barry F. Smith. "Extreme-Scale Multigrid Components Within PETSc". *PASC '16: Proceedings of the Platform for Advanced Scientific Computing Conference*. 2016.

ABF + Multigrid Test Case: Variable Viscosity



```
-nondimensional
-eta1 1e-6 # Set viscosity contrast
-stag_grid_x 256
-stag_grid_y 256
-ksp_type fgmres
-pc_type fieldsplit
-pc_fieldsplit_type schur
-pc_fieldsplit_schur_fact_type upper
-pc_fieldsplit_schur_precondition selfp
-fieldsplit_element_ksp_type preonly
-fieldsplit_element_pc_type jacobi
-fieldsplit_face_pc_type mg
-fieldsplit_face_pc_mg_galerkin
-fieldsplit_face_pc_mg_levels 5
-fieldsplit_face_mg_levels_ksp_max_it 6
-fieldsplit_face_mg_levels_ksp_type chebyshev
-fieldsplit_face_mg_levels_pc_type jacobi
-fieldsplit_face_mg_coarse_pc_type redundant
-fieldsplit_face_mg_coarse_redundant_pc_type lu
-fieldsplit_face_mg_coarse_redundant_pc_factor_mat_solver_type
    umfpack
```

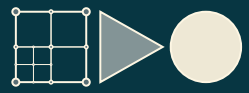
ABF + Multigrid Test Case: Viscosity Jump

With a viscosity jump of 6 orders of magnitude, $\eta_1 = 10^{-6}, \eta_2 = 1$, we see good algorithmic scalability:

Grid	MG Levels	FGMRES Iterations (10^-6 rel. tol.)	`KSPSolve` time
32^2	3	12	0.025
64^2	4	12	0.11
128^2	5	12	0.48
256^2	6	11	2.05
512^2	7	11	9.56
1024^2	8	11	41.2

Run on a single rank on a desktop, but identical iteration counts in parallel.

Current Work: Monolithic Smoothers



- `DMStag`'s default interpolation operators are suitable for geometric multigrid on the entire saddle point matrix
- An important smoother in this case is the Distributed Gauss-Seidel (DGS) smoother[18] and related algorithms[19].
- These are not quite available from the command line, though they are extremely similar to ABF preconditioners
- Thus, we will introduce a new implementation of `PCFieldSplit` to support preconditioners of the following form , for an operator L:

$$P = M(L\hat{M})^{-1}$$

$$LM = \begin{bmatrix} A & G \\ D & \end{bmatrix} \underbrace{\begin{bmatrix} I & G \\ & -DG \end{bmatrix}}_{M} = \begin{bmatrix} A & AG - GDG \\ D & DG \end{bmatrix}$$

[18] Achi Brandt and Nathan Dinar. "Multigrid Solutions to Elliptic Flow Problems". *Numerical methods for partial differential equations*. Elsevier, 1979, pp. 53–147.

[19] Ming Wang and Long Chen. "Multigrid Methods for the Stokes Equations Using Distributive Gauss–Seidel Relaxations Based on the Least Squares Commutator". *Journal of Scientific Computing* 56.2 (2013), pp. 409–431.

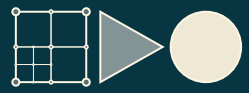
Interacting with Particles

Particles with `DMSwarm`

Particle Integration

- By integrating with PETSc's `DMSwarm`, one can buld a full MAC code with `DMStag`
- Particles store a cell ID, coordinates, and other data
- Coordinates represented using `DMProduct` allow fast location of particles
- `DMStagMigrate()` takes care of parallel details
- Default interpolation routines are provided
- For an example, see `$PETSC_DIR/src/dm/impls/stag/examples/tutorials/ex5.c` (in the `psanan/stagbl-working-base` branch, currently)

Particles with StagBLDemo2d



- Our 2D demo includes particles, but not in a ‘full-featured‘ way
- Temperature is interpolated to the particles from the grid
- Information isn't propagated the other way, yet, though the tools are in place (hackathon topic for me?)
- Let's look at `src/demos/particles.c`