MAE 250H, Spring 2019
J. D. Eldredge
Homework 2, Due Tuesday, April 16

This homework is focused on approximation of spatial derivatives with finite difference methods on a staggered grid.

1. **Construction of differencing operator routines for staggered grid.** For later work this term, it will be helpful to have routines that perform the actions of differential operators on a 2-d staggered grid. Namely, you would like to be able to compute basic operations such as div, grad and curl on grid data, as well as other operations like taking inner products and norms. In constructing these routines, it is important to preserve the properties that mimic those of their continuous counterparts. We will discuss a lot of this in class, but I've presented much of it here for reference. First, some notation

   - Let's define the spaces in which the different types of data live. Let $\mathcal{C}$ denote the space of cell-center variables, $\mathcal{E}$ the space of edge-centered variables, and $\mathcal{N}$ the space of node-centered variables. For example, pressure data live in $\mathcal{C}$ (i.e. $\mathsf{p} \in \mathcal{C}$). The velocity components, denoted collectively as q, are in $\mathcal{E}$. And the streamfunction and vorticity (s and w, respectively) are in $\mathcal{N}$.
   - The discrete differential operators map variables from one space to another. They are all defined based on 2nd-order central difference schemes. For example,

     (a) **divergence**. Define

$$\mathsf{D} : \mathcal{E} \mapsto \mathcal{C} \,|\, (\mathsf{D}\mathsf{q})_{i,j} = (q_{x,i,j} - q_{x,i-1,j}) + (q_{y,i,j} - q_{y,i,j-1})$$

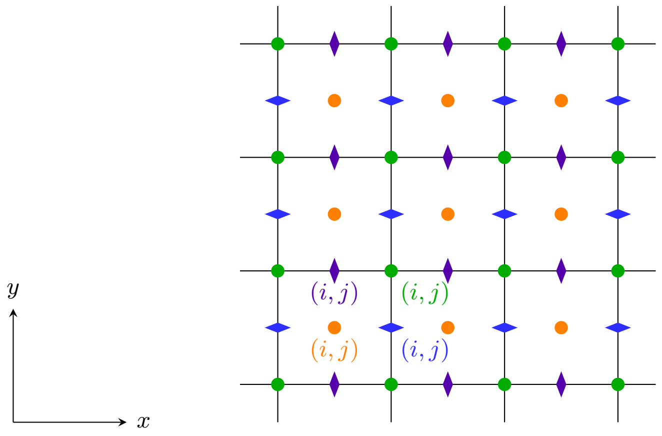

Figure 1: Variable types on two-dimensional domain. Node-centered variable: ● ; x-edge-centered variable: ◆ ; y-edge-centered variable: ⧫ ; cell-centered variable: ● . The relationships between the indexing schemes for each variable type is shown for reference. For example, $(i+1/2, j) = (i, j)$.

(We will discuss the indexing conventions below.) Thus Dq, where $\mathsf{q} \in \mathcal{E}$, is the discrete analog of $\nabla \cdot \boldsymbol{u}$, and the result is stored at the cell centers.

(b) **gradient**. Define

$$\mathsf{G} : \mathcal{C} \mapsto \mathcal{E} \left| \begin{array}{l} (\mathsf{Gp})_{x,i,j} = p_{i+1,j} - p_{i,j} \\ (\mathsf{Gp})_{y,i,j} = p_{i,j+1} - p_{i,j} \end{array} \right.$$

Thus, Gp, where $\mathsf{p} \in \mathcal{C}$, is the discrete analog of ∇p. Note that $\mathsf{G} = -\mathsf{D}^T$.

(c) **node curl** (or just called **curl**). Define

$$\mathsf{C} : \mathcal{N} \mapsto \mathcal{E} \left| \begin{array}{l} (\mathsf{Cs})_{x,i,j} = s_{i,j} - s_{i,j-1} \\ (\mathsf{Cs})_{y,i,j} = -(s_{i,j} - s_{i-1,j}) \end{array} \right.$$

Thus, $\mathsf{q} = \mathsf{Cs}$, where $\mathsf{q} \in \mathcal{E}$ and $\mathsf{s} \in \mathcal{N}$, is the discrete analog of $\boldsymbol{u} = \nabla \times \boldsymbol{\psi}$.

(d) **edge curl** (or sometimes called **rot**). Define

$$\mathsf{C}^T : \mathcal{E} \mapsto \mathcal{N} \,|\, (\mathsf{C}^T\mathsf{q})_{i,j} = (q_{y,i+1,j} - q_{y,i,j}) - (q_{x,i,j+1} - q_{x,i,j})$$

Thus, $\mathsf{w} = \mathsf{C}^T\mathsf{q}$, where $\mathsf{w} \in \mathcal{N}$ and $\mathsf{q} \in \mathcal{E}$, is the discrete analog of $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$. Note that this operator is the transpose of the node curl.

- These operations have several neat properties
  - $\mathsf{DCf} = 0$ for all $\mathsf{f} \in \mathcal{N}$. This is the discrete analog of $\nabla \cdot (\nabla \times \boldsymbol{f}) = 0$. In linear algebraic terms, this means that the columns of C are in the nullspace of D.
  - $\mathsf{C}^T\mathsf{Gf} = 0$ for all $\mathsf{f} \in \mathcal{C}$. This is the discrete analog of $\nabla \times \nabla f = 0$. In linear algebraic terms, this means that the columns of G are in the nullspace of C^T. However, it follows directly from the previous property because $(\mathsf{C}^T\mathsf{G})^T = -\mathsf{DC}$.
  - We can generate three different types of Laplacian operator from these. First is a node-based Laplacian,

$$\mathsf{L}^\dagger : \mathcal{N} \mapsto \mathcal{N} \,|\, \mathsf{L}^\dagger = -\mathsf{C}^T\mathsf{C}$$

This is the analog of $\nabla^2 \boldsymbol{f} = -\nabla \times \nabla \times \boldsymbol{f} + \nabla(\nabla \cdot \boldsymbol{f})$. In 2-d, node-based variables $\boldsymbol{f}$ are vorticity or streamfunction, which are vectors that are directed out of the plane, so $\nabla \cdot \boldsymbol{f} = 0$ automatically. Thus, this is simply $\nabla^2 \boldsymbol{f} = -\nabla \times \nabla \times \boldsymbol{f}$.

The second is a cell center-based Laplacian,

$$\mathsf{L} : \mathcal{C} \mapsto \mathcal{C} \,|\, \mathsf{L} = \mathsf{DG}$$

This is the analog of $\nabla^2 f = \nabla \cdot \nabla f$. Finally, we can develop a vector Laplacian, $\tilde{\mathsf{L}}$, that acts upon edge-centered variables in $\mathcal{E}$ and maps them into the same space,

$$\tilde{\mathsf{L}} : \mathcal{E} \mapsto \mathcal{E} \,|\, \tilde{\mathsf{L}} = -\mathsf{CC}^T + \mathsf{GD}$$

which is the discrete analog of $\nabla^2 \boldsymbol{u} = -\nabla \times \nabla \times \boldsymbol{u} + \nabla(\nabla \cdot \boldsymbol{u})$, when $\boldsymbol{u}$ is a vector field in the plane.

Note that one can verify that all of these result in the standard five-point stencil for centered differences of second derivatives. Based on these definitions, we have a node-based Poisson equation $\mathsf{w} = \mathsf{C}^T\mathsf{q} = \mathsf{C}^T\mathsf{C}\mathsf{s} = -\mathsf{L}^\dagger\mathsf{s}$ and a similar one based at cell centers.

- It is important to remember that these are defined based on an *infinite* grid. However, on a finite grid, there is a boundary, and so some adjustment is needed. The domain is divided into an integer number of cells in each direction, $N_x \times N_y$, with $(N_x + 1) \times (N_y + 1)$ nodes. We use *ghost values* for some data types in order to accommodate these differencing formulas where they are needed:
  - Cell data will include a layer of ghost cells on all sides, so the actual number of cell data is $(N_x + 2) \times (N_y + 2)$.
  - Node data will not include any ghosts.
  - Edge component data only includes ghosts along a direction in which it lies strictly in the interior. For example, the y velocity components are aligned with cell centers in the x direction, so they would have ghost values in that direction, but they are aligned with nodes in the y direction, so they would not have ghosts in that case. So, for example, there will be at total of $(N_x + 2) \times (N_y + 1)$ values of y velocity component data.

Your task is to develop routines that compute each of the four basic operations defined above for a 2-d staggered grid. I have provided two routines, called `divergence` and `rot`, in the Julia code in the HW2 repository. I have also defined some data types for cell centers, edges and nodes, and some other useful operations, such as inner product (`dot`) and L_2 norm (`norm`) on cell centered data. So you must create `curl`, `gradient`, `laplacian` (separate Laplacians on interior cells, nodes, and edges), and inner product and norm for other data types. Implement all in your preferred language, as usual. Please note the structure of the routines that I have provided. In particular, we are only working in index space for now, so there is no discussion of grid spacing yet.

As usual, you should develop unit tests for all operations. I have provided a few in the `runtests.jl` file.

2. **Application and error of finite difference schemes.** Here, you are to use your code to evaluate the derivatives of a given field. Suppose you are considering a domain $0 \leq x \leq 1$, $0 \leq y \leq 1$ and the function $u(x, y) = \sin(P\pi x)\sin(Q\pi y)$, where P and Q are integers. You will suppose that this field is known at the cell centers (including the ghost cells).

   - Establish a mapping from index space to physical space with uniform cell size Δx and N cells in each direction.
   - Evaluate the Laplacian of the cell-centered data and compare with the exact Laplacian. Plot L_2 error versus N and verify again. (Note that you will have to scale your differ-

encing result by $1/\Delta x^2$ before comparing.) Do this for $P = 1$, $Q = 1$ and $P = 2$, $Q = 3$ and plot error on the same log-log plot.

- Verify that the divergence of the gradient also gives the same result as the Laplacian.
- Verify that the rot of the gradient gives zero (or machine precision value) for the L_2 norm.
- If you evaluate the original field at nodes rather than cell centers, verify that the divergence of the curl of this field also has zero L_2 norm.