Computational PDEs, Fall 2021
Pseudo-spectral solver for the two-dimensional Navier-Stokes equations

Aleksandar Donev
Courant Institute, NYU, donev@courant.nyu.edu

Oct 26th, 2021
Due on **Sunday November 21st 2021**

The Taylor vortex

$$v_x = v_0 - 2e^{-8\pi^2\mu t/L^2}\cos\left(\frac{2\pi(x-v_0t)}{L}\right)\sin\left(\frac{2\pi(y-v_0t)}{L}\right), \tag{1}$$

$$v_y = v_0 + 2e^{-8\pi^2\mu t/L^2}\sin\left(\frac{2\pi(x-v_0t)}{L}\right)\cos\left(\frac{2\pi(y-v_0t)}{L}\right), \tag{2}$$

$$\pi = -e^{-16\pi^2\mu t/L^2}\left[\cos\left(\frac{4\pi(x-v_0t)}{L}\right)+\cos\left(\frac{4\pi(y-v_0t)}{L}\right)\right]. \tag{3}$$

is a well-known exact solution of the unforced Navier-Stokes equation

$$\partial_t\boldsymbol{v} + \boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v} = -\boldsymbol{\nabla}\pi + \mu\boldsymbol{\nabla}^2\boldsymbol{v}, \tag{4}$$

in two dimensions for a periodic domain of size $L\times L$. You can set $v_0=1$, $L=1$ and $\mu=0.05$ for this homework. Note that to capture the non-zero mean of the velocity in the vorticity-stream function formulation you need to keep track of it separately and add it "by hand",

$$\boldsymbol{v} = \boldsymbol{\nabla}^\perp\psi + \boldsymbol{v}_0.$$

This is because momentum is conserved and thus $\boldsymbol{v}_0 = \hat{\boldsymbol{v}}(\boldsymbol{k}=0)$ is constant in time and not something to be solved for but given in the initial condition.

Note: You can easily make the velocity less smooth by increasing the wavenumber and replacing, for example, $\cos(2\pi(x-v_0t)/L)$ with $\cos(\kappa\cdot 2\pi(x-v_0t)/L)$ where κ is an integer larger than one (you will have to adjust the exponential decay accordingly). Alternatively, you can make your periodic domain be of size $(\kappa L)\times(\kappa L)$ without having to change the Taylor vortex formula. Either way, it is important to write code that is flexible and where things like length of domain are not hard-wired in any way.

For temporal integration of ODEs with a stiff linear term and a non-stiff nonlinear term,

$$\frac{d\hat{\boldsymbol{\phi}}}{dt} = \boldsymbol{A}\hat{\boldsymbol{\phi}} + \boldsymbol{B}\left(\hat{\boldsymbol{\phi}}\right), \tag{5}$$

you can use either IMEX or exponential integrators; just picking one is sufficient for this assignment. Spectral deferred corrections would be the best method but this is more for a final project than a homework.

For example, one integration method you can use is the SBDF2 scheme with time step size Δt,

$$\hat{\boldsymbol{\phi}}^{n+1} = \frac{4}{3}\hat{\boldsymbol{\phi}}^{n} - \frac{1}{3}\hat{\boldsymbol{\phi}}^{n-1} + \frac{2\Delta t}{3}\boldsymbol{A}\hat{\boldsymbol{\phi}}^{n+1} + \frac{2\Delta t}{3}\left(2\boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^{n}\right) - \boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^{n-1}\right)\right),$$

see also Eq. (35) in the paper of Cox and Matthews (CM) where it is called the AB2BDF2 scheme; more IMEX options are given in the lecture notes. Another option for a temporal integrator is the exponential time differencing trapezoidal RK2 method,

$$\hat{\boldsymbol{\phi}}^{n+1,\star} = e^{\boldsymbol{A}\Delta t}\hat{\boldsymbol{\phi}}^{n} + \boldsymbol{A}^{-1}\left(e^{\boldsymbol{A}\Delta t}-\boldsymbol{I}\right)\boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^{n}\right) \quad \text{(predictor)}$$

$$\hat{\boldsymbol{\phi}}^{n+1} = \hat{\boldsymbol{\phi}}^{n+1,\star} + \boldsymbol{A}^{-2}\left(\frac{e^{\boldsymbol{A}\Delta t}-\boldsymbol{I}-\boldsymbol{A}\Delta t}{\Delta t}\right)\left(\boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^{n+1,\star}\right) - \boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^{n}\right)\right) \quad \text{(corrector)},$$

see also Eqs. (20+22) in the paper by CM. Note that direct implementation of these formulas can suffer from roundoff errors for small Δt, more precisely, for small $|\lambda_{\min}|\,\Delta t$ where $\lambda_{\min}$ is the eigenvalue of $\boldsymbol{A}$ with smallest magnitude; make sure this is not the case or find a way to avoid catastrophic cancellation (e.g., by using Taylor series). On the course homepage I have linked a code by A. K. Kassam and L. N. Trefethen that solves KdV with fourth-order ETDRK4 method, along with the paper that explains how to avoid roundoff problems. You can try this method/code and compare to see how much improvement you can get from the higher-order scheme; it should be easy to just extract the temporal integrator from the sample code.

Note: Make sure to write the code in a way that avoids evaluating things more than once. For example, evaluate $\boldsymbol{B}\left(\hat{\boldsymbol{\phi}}^n\right)$ only once per time step and store and reuse between the predictor and the corrector or between successive time steps. Further, observe that if $\boldsymbol{A}$ is constant many things can be computed once and only once at the beginning (and that computation is super cheap if done right).

1 Velocity Solver

Write a pseudo-spectral solver for the two-dimensional Navier-Stokes equations in the vorticity-stream formulation, as we discussed in class. Choose a temporal integrator and discuss your choice. If you don't want to deal with unmatched modes, it is OK to use grids that are of odd size, for example, powers of three instead of powers of two.

For evolution PDEs, we are always really concerned with combined spatio-temporal error. For this assignment, temporal error will dominate so the order is controlled by the temporal integrator. I suggest approaching this in the following way: First, figure out how many spatial points you need so that the error is dominated by the temporal error. Then, keep the number of spatial points fixed and only change the time step size to estimate a temporal order of accuracy and report that.

Important note: For pseudospectral methods, the solution you get from the code is actually a **function not a vector** (sum of basis functions where the vector is the set of coefficients in front of the basis functions), in this specific case, a trigonometric polynomial. So when talking about solution/error we are talking about a solution/error function, *not* a solution/error vector. Numerically, the best way to estimate/plot the solution or the error is to evaluate the trigonometric polynomial on a refined grid (say 1024^2 points) using the FFT, and then plot or compute error norms using that fine grid. For this, you will need the equivalent of the Matlab function *interpft* but for two dimensions; best to write your own but there are also various codes online, just google *interpftn* (note that some codes may only work for even/odd grids).

1.1 Taylor Vortex

Using the Taylor vortex as initial condition, solve the NS equations from time $t = 0$ to time $t = T = 0.25$ and compare the numerical solution to the exact solution. Discuss (both numerically and analytically) the order of accuracy of the scheme you developed, as well as the computational cost. What would be different (better, worse) if you used a finite-volume spatial discretization?

1.2 Vortices

To create a more interesting test for pseudo-spectral codes, set $L = 2\pi$ and initialize the vorticity field with a superposition of three Gaussian peaks (i.e., three "vortices"):

$$\begin{aligned}\omega(x,y;t=0) = &\exp\left(-5\left((x-\pi)^2+(y-3\pi/4)^2\right)\right)+\exp\left(-5\left((x-\pi)^2+(y-5\pi/4)^2\right)\right). \\ &-\frac{1}{2}\exp\left(-\frac{5}{2}\left((x-5\pi/4)^2+(y-5\pi/4)^2\right)\right)\end{aligned} \tag{6}$$

Note that this is not strictly speaking a periodic function but it is sufficiently close to one due to the rapid decay of the Gaussian peaks. To get an actual periodic vorticity each peak would have to periodically replicated on an infinite square grid, for example, for the first peak, use

$$\omega(x,y;t=0) = \sum_{i,j=-M}^{M} \exp\left(-5\left((x-(\pi+iL))^2+(y-(3\pi/4+jL))^2\right)\right),$$

where in practice setting $M = 1$ (i.e., only including 9 copies of the peak) should be more than sufficient to get a function that is periodic to roundoff tolerance.

1. Confirm that your code computes the right-hand side of the PDE for vorticity ($\partial_t \omega = \text{rhs}$) to spectral accuracy. This means that the rhs of the system of ODEs is evaluated to spectral accuracy in space. How many grid points do you need to get the rhs to about ~~9 digits~~6 digits of accuracy? Try with and without anti-aliasing and comment on your observations.
2. Apply your temporal integrator and estimate empirically the temporal order of accuracy of your method at time $T = 0.25$. How accurate is your answer for the velocity at the final time, in some norm of choice?
3. [Optional but encouraged] Make a movie of your solution, for example, a vector plot of the velocity. *Please do not email large movies with your solutions, put them on a shared drive.*

2 Advection-Diffusion Solver

Solve also a forced scalar advection-diffusion equation for the concentration $c(\boldsymbol{r}, t)$ a passive tracer,

$$\partial_t c + \boldsymbol{v} \cdot \boldsymbol{\nabla} c = D \boldsymbol{\nabla}^2 c + f(r, t). \tag{7}$$

Here the velocity $\boldsymbol{v}\,(x, y, t)$ may come from the numerical solution in the first part of this homework or be specified analytically. In many practical applications the concentration can couple back into the velocity equation but here we do not consider this case.

1. If you set $D = \mu$ and $f = -\partial_x \pi$ then $c = v_x$ from the Taylor vortex is a solution of (7). Use this to validate your advection-diffusion solver as you would use a manufactured solution.
2. Now set $f = 0$ and $D = 0$ (pure advection) and start with the initial condition

$$c(x, y; t = 0) = \left[\sin\left(\frac{\pi x}{L}\right) \sin\left(\frac{\pi y}{L}\right)\right]^{100}. \tag{8}$$

   First, set $\boldsymbol{v}\,(x, y, t) = \boldsymbol{v}_0 = \text{const}$ and advect the peak to time $t = 1$, when it should come back to its starting position unchanged. Try a few grid sizes (explain how you chose them) and comment on how well the method performs. Examine numerically what the stability limit is for the time step size.
3. [Optional but encouraged] Try the initial conditions (6) and (8) and observe what happens to the peak as it is advected around by the divergence-free flow for different grid resolutions (best done using a movie) for $D = 0$; observe that the velocity decays exponentially so after some time the peak should reach a steady final state. Comment on your observations.