Chorin's Projection Method with Spectral Collocation for 2D Incompressible Navier Stokes

August 23, 2019

The setup and time discretization with Crank Nicholson remains the same as the finite difference version of this (see other directory in repo). Here, we focus on deriving a Chebyshev pseudo-spectral estimator.

Recall the NSE (with no force function)

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p = \Delta \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0$$
$$\mathbf{u} = 0 \quad \text{on} \quad \partial\Omega$$

Chorin: Step 1 Also recall that the Chorin's projection method first ignores pressure to comute an intermediate velocity field

$$\frac{\partial \mathbf{u}^*}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = \Delta \mathbf{u}^*$$
$$\mathbf{u}^* = 0 \quad \text{on} \quad \partial\Omega$$

We discretize time first with Adams-Bashford and implicit Crank-Nicholson

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\triangle t} + (\mathbf{u}^{\frac{n+1}{2}} \cdot \nabla)\mathbf{u}^{\frac{n+1}{2}} = \Delta \mathbf{u}^{\frac{n+1}{2}}$$

So far things are identical to the finite difference version of Chorin's. But to solve this, we do not discretize spatial dimensions anymore: we use (pseudo)spectral methods:

We want to approximate the solution $\mathbf{u}^*$ as truncated series of Chebyshev polynomials: $\{T_k(x)\}_{k=1}^{\infty}$ where $T_k(x) = \cos(k \cos^{-1} x)$; each polynomial is restricted to $[-1, 1]$.

$$u_N^*(x) = \sum_{k=0}^{\infty} \hat{u}_k^* T_k(x) \approx \sum_{k=0}^{N} \hat{u}_k^* T_k(x)$$

where $\mathbf{u}^* = (u^*, v^*)$. We just write the derivation for the first equation for simplicity. To get the Chebyshev coefficients, we have to compute the (normalized) inner product:

$$\hat{u}_k = \frac{2}{\pi c_k}\int_{-1}^{1} u T_k w dx$$

where $c_k = \begin{Bmatrix} 2 & \text{if} & k=0 \\ 1 & \text{if} & k \geq 1 \end{Bmatrix}$. This is hard to compute, so we need to estimate the integral. Normally, this would require a lot of points but we can get away with carefully selected ones and a Gaussian quadrature. The best points turn out to be the roots of another Chebyshev polynomial with degree one higher; these are called the Gauss-Lobatto points – they end up with denser spread near the boundaries (-1 and 1).

$$x_i = \cos\frac{\pi i}{N}, i = 0, ..., N \tag{1}$$

resulting in,

$$\hat{u}_k = \frac{2}{\bar{c}_k N}\sum_{i=0}^{N}\frac{1}{\bar{c}_i}u_i T_k(x_i), k = 0, ..., N \tag{2}$$

where $\bar{c}_k = \begin{Bmatrix} 2 & \text{if} & k=0 \\ 1 & \text{if} & 1 \leq k \leq N-1 \\ 2 & \text{if} & k=N \end{Bmatrix}$. Note that to convert back and forth between spectral coefficients ($\hat{u}_k$) and the values at the collocation points ($u_N(x_i)$) is a matrix multiplication. To be explicit, let $\mathcal{T} = [cosk\pi i/N], k, i = 0, ..., N$ and $\mathcal{T}^{-1} = [2(\cos\pi i/N)/(\bar{c}_k\bar{c}_i N)]$. Then,

$$\mathcal{U}^* = \mathcal{T}\hat{\mathcal{U}}^*$$
$$\hat{\mathcal{U}}^* = \mathcal{T}\mathcal{U}^*$$

where $\mathcal{U}^* = [u^*(x_0), ..., u^*(x_N)]$ and $\hat{\mathcal{U}}^* = [\hat{u}_0, ..., \hat{u}_N]$.

An interesting fact that is useful is that the approximation:

$$u_N^*(x) = \sum_{k=0}^{N}\hat{u}_k^* T_k(x) \tag{3}$$

can be viewed as a Lagrange interpolating polynomial with a set $\{x_i\}$. One can explicitly write this as

$$u_N^*(x) = \sum_{j=0}^{N} h_j(x) u^*(x_j) \tag{4}$$

where $h_j(x) = \frac{(-1)^{j+1}(1-x^2)T'_N(x)}{\bar{c}_j N^2(x-x_j)}$ is some crazy polynomial. This is really useful because it lets us do differentiation in closed form in physical space (no need for fourier transforms). In particular, we can write the p-th derivative,

$$u_N^{*,(p)}(x_i) = \sum_{k=0}^{N}\hat{u}_k T_k^{(p)}(x_i) = \sum_{j=0}^{N} h_j^{(p)}(x_i) u_N(x_j) \tag{5}$$

Notice then that computing the derivative is just a matrix multiplication on the existing coordinate values! If we let $d_{i,j}^{(p)} = h_j^{(p)}(x_i)$, then we can get a series of formulas populating a "derivative matrix", $\mathcal{D} = [d_{i,j}^{(1)}], i, j = 0, ..., N$.

$$
\begin{aligned}
d_{i,j}^{(1)} &= \frac{\bar{c}_i}{\bar{c}_j}\frac{(-1)^{i+j}}{(x_i - x_j)}, 0 \leq i, j \leq N, i \neq j \\
d_{i,i}^{(1)} &= -\frac{x_i}{2(1 - x_i^2)}, 1 \leq i \leq N - 1 \\
d_{0,0}^{(1)} &= -d_{N,N}^{(1)} = \frac{2N^2 + 1}{6}
\end{aligned}
$$

So, in short, $\mathcal{U}^{*,(1)} = \mathcal{D}\mathcal{U}^*$, $\mathcal{U}^{*,(2)} = \mathcal{D}^2\mathcal{U}^*$. As nice as this is,there are a few hacks we need to be careful of for numerical stability.

First, calculating $(1 - x_i^2)$ and $(x_i - x_j)$ may be hard if points are very close, so we use the following:

$$
\begin{aligned}
x_i - x_j &= 2\sin\frac{(j + i)\pi}{2N}\sin\frac{(j - i)\pi}{2N} \\
1 - x_i^2 &= \sin^2\frac{i\pi}{N}
\end{aligned}
$$

Second, since this differentiation matrix is approximated, it doesn't always represent the derivative of a constant (which it should). In other words, it should be that

$$
\sum_{j=0}^{N} d_{i,j}^{(1)} = 0, i = 0, ..., N \tag{6}
$$

. To fix this, we should calculate the off diagonal entries later to satisfy this constraint. So...

$$
d_{i,j}^{(1)} = -\sum_{j=0, j\neq i}^{N} d_{i,j}^{(1)}, i = 0, ..., N \tag{7}
$$

When we want to compute $\mathcal{D}$ and $\mathcal{D}^2$, we use the following procedure:

- compute $\mathcal{D}$ with the diagonal term correction
- square it to get a provisional $\tilde{\mathcal{D}}^2$.
- correct $\tilde{\mathcal{D}}^2$ to nsum to 1. As in first set $d_{i,j}^{(2)} = \tilde{d}_{i,j}^{(2)}$ for $j \neq i$. Then set $d_{i,i}^{(2)} = -\sum_{j=0,j\neq i}^{N} \tilde{d}_{i,j}^{(2)}, i = 0, ..., N$.

Ok, now that we know how to differentiate, we revisit the original equation. The intuition is that we can replace all unknown values in terms of the coordinate values. Then there's a system of equations that we need to matrix diagonalize.