CS598: High-Order Methods for PDEs
Assignment 5 — Due Tuesday, Mar. 1

These exercises are intended to familiarize you with some tools used to derive, analyze, and understand the Galerkin formulation in 1D. Estimated write-up is about two to three pages, including figures. (Note, semhat.m and dependencies are provided. Please let me know if you have difficulty finding python equivalents.)

1. Consider the time-dependent advection equation with periodic boundary conditions,

$$\frac{\partial u}{\partial t} = -c\frac{\partial u}{\partial x} \qquad u(0) = u(1) = 0,$$

with $c = 1$ and Gaussian initial condition

$$u^0(x) = e^{-a},\ a(x) := 1000\left(x - \frac{1}{2}\right)^2.$$

Solve this problem using AB3 in time and spectral elements in space. We will use n=90 points for each case. Take $E := n/N$ and N = 1, 2, 3, 5, 6, 9, 10, 15. (Be sure to make your matrices `sparse`.)

Plot the solution at time t_f=10 for a couple of representive cases.

Plot the (maximum pointwise) error at time t_f=10 as a function of N for the cases considered. Note, you will also plot your Problem 2 results on this same graph — so just submit one graph for both Problem 1 and 2. As this is discrete data, use lines and points (e.g. "o-" in matlab).

How did you choose your timestep size? Choose it to be small enough to not influence the error in your second graph (i.e., error vs. N). What value of Δt did you use in the case of N=15 ? What is the corresponding value of the CFL,

$$\mathrm{CFL} := \max_j \frac{c\Delta t}{\Delta x_j}\ ?$$

Note: When checking your error at t_f, be certain that $n_{\text{steps}} \cdot \Delta t = t_f$. I usually use the following to ensure this condition.

```
nstep = ceil (tfinal / dt);
dt    = tfinal / nstep;
```

(In fact, I usually set $t_f = 1$, and then loop 10 times. This allows me to plot the solution at unit times, t=1, 2, etc.)

2. Consider the case of a full mass matrix. This can be constructed local to each spectral element as

$$\hat{B}_f = J^T \hat{B}_M J,$$

where $\hat{B}_M$ is the diagonal mass matrix for polynomial degree $M > N$ and J is the interpolation matrix mapping from $N+1$ GLL points to $M+1$ GLL points. (J is the output of `interp_mat.m` provided in the last homework.)

Comment on the results. (What about accuracy? What about time?)

3. For the case $N=1$, what does your convection matrix look like? (It should look familiar.)