MA4245 Mathematical Principles of Galerkin Methods
Project 2: 1D Wave Equation

Prof. Frank Giraldo
Department of Applied Mathematics
Naval Postgraduate School
Monterey, CA 93943-5216

Due: May 10, 2024 at 12pm

1 Continuous Problem

The governing partial differential equation (PDE) is

$$\frac{\partial q}{\partial t} + \frac{\partial f}{\partial x} = 0 \qquad \forall x \in [-1, +1]$$

where $f = qu$ and $u = 2$ is a constant. Thus, an initial wave $q(x, 0)$ will take exactly $t = 1$ time in order to complete one full revolution (loop) of the domain.

1.1 Initial Condition

Since the governing PDE is a hyperbolic system, then this problem represents an initial value problem (IVP or Cauchy Problem). We, therefore, need an initial condition. Let it be the following Gaussian

$$q(x, 0) = e^{-64x^2}$$

where $x \in [-1, 1]$.

1.2 Boundary Condition

This problem also requires a boundary condition: let us impose periodic boundary conditions, meaning that the domain at $x = +1$ should wrap around and back to $x = -1$. Your solution variable q should have the same solution at $x = -1$ and $x = +1$.

2 Simulations

Here are the steps you should follow to complete this project.

1. Write functions in order to construct the element-wise Mass and (Weak form) Differentiation matrices. For CG, you can also build the strong form Differentiation matrix which will give you similar answers. Confirm that they are correct.

2. Write a DSS function in order to construct the global matrices for Mass and Differentiation matrix. Don't worry about periodicity yet. Once you have that done, I can show you how to include this for CG. For DG, you don't need this since this will be handled by the Flux matrix (which is already provided for you in the code For Students).

3. Using the code For Students, get the problem to run and confirm that it is indeed working.

4. Once this has been achieved, go ahead and work on improving the efficiency and generalization of your code in order to not have to build and store global matrices for D and F. It is OK to keep the global matrix for M though (no way around this). This step here will greatly help you be able to make your Project 3 easier to complete.

5. Perform convergence rates studies as discussed in the next section.

Write a code (or two) that uses both CG and DG. I strongly recommend that you code the CG version first. It is better to use the same code to do both CG and DG with a switch (if statement) to handle the communicator in both CG and DG. You need to show results for exact (let Q=N+1 be exact) AND inexact integration (Q=N) so write your codes in a general way.

2.1 Results You Need to Show

You must show results for linear elements $N = 1$ with increasing number of elements N_e and then show results for $N = 4$, $N = 8$, and $N = 16$ with increasing numbers of elements.

N=1 Simulations For linear elements, use $N_e = 16, 32$ and 64 elements. Plot the normalized L^2 error norm versus N_P (given below) for these 3 simulations on one plot.

N=4 Simulations For $N = 4$ use $N_e = 4, 8$ and 16 elements and plot the norms as above.

N=8 Simulations For $N = 8$ use $N_e = 2, 4$ and 8 elements and plot the norms as above.

N=16 Simulations For $N = 16$ use $N_e = 1, 2$ and 4 elements and plot the norms as above.

3 Helpful Relations

Error Norm The normalized L2 error norm that you should use is:

$$||error||_{L^2} = \sqrt{\frac{\sum_{k=1}^{N_P} \left(q^{numerical} - q^{exact}(x_k)\right)^2}{\sum_{k=1}^{N_P} q^{exact}(x_k)^2}} \tag{1}$$

where $k = 1, ..., N_P$ are $N_p = N_e N + 1$ global gridpoints and $q^{numerical}$ and q^{exact} are the numerical and exact solutions after one full revolution of the wave. Note that the wave should just stop where it began without changing shape (in a perfect world). Your solution will do that for lots of gridpoints (high resolution). At low resolution, you will see much error.

Time-Integrator The code For Students is equipped with two classes of Runge-Kutta time-integrators. The number of stages is controlled by the variable STAGES. More stages means that the time-step you can use can be bigger. Just pick one time-integrator and use it for all of your simulations.

The code For Students is designed to maintain a constant Courant number

$$C = u\frac{\Delta t}{\Delta x}$$

regardless of your Δx. If you are seeing your convergence rates plateau (not getting smaller for more points) then you will need to decrease the *Courant max* variable in the code.