Problem Set 5

Ph.D. Course 2009:
An Introduction to DG-FEM for solving partial differential equations

---

If you have not already done so, please download all the Matlab codes from the book from

http://www.nudg.org/

and store and unpack them in a directory you can use with Matlab.

We shall consider the prototype model for hyperbolic PDEs, namely, the linear advection equation stated in differential form as

$$u_t + c_x u_x + c_y u_y = 0, \quad \mathbf{x} \in \Omega([-1,1]^2) \tag{1}$$

which describes translation of some quantity u with advection speed vector given as $\mathbf{c} = (c_x, c_y)^T$. The exact solution to this equation can be shown to be of the form

$$u(x,y,t) = f(x - c_x t)g(y - c_y t) \tag{2}$$

where $g(\cdot), f(\cdot)$ are arbitrary functions.

- Verify that the system is hyperbolic and conserves energy if u is assumed periodic.
- Using an energy technique, discuss how many boundary conditions are needed in a finite domain at different boundaries in a square domain.

Having established well-posedness for the problem, we shall seek to develop a numerical solver for this problem using DG-FEM.

For this exercise, you can use the mesh file stored in squaremesh.dat. Load into matlab and visualize with the commands

```
>> load
>> triplot(EToV,VX,VY)
```

- Implement a DG-FEM solver by modifying existing Matlab codes and thereby set up as usual your own three Advec2dxx Matlab scripts to solve the problem.
- Implement both a central flux and an upwind flux. Boundary conditions should be imposed according to the incoming characteristics.

Scientific Computing Section, DTU Informatics, Kgs.-Lyngby, Denmark.

- Carry out *hp*-convergence tests to establish the error behavior for the scheme.

Now, we will modify the problem slightly by assuming that the velocities are dependent on the spatial coordinates such that we instead solve the linear advection equation with spatially varying coefficients $c_x = c_x(x, y)$ and $c_y = c_y(x, t)$.

- Consider what changes will be required for your DG-FEM solver to account for varying advection coefficients?
- Implement those changes in your solver and prepare it for use on a finite domain where the exact solution is imposed at all of the outer domain boundary $\mathbf{x} \in \partial\Omega_h$ using mapB.
- Consider the case of a rotating Gaussian hill with $\mathbf{c} = (2\pi y, -2\pi x)^T$ in $(x, y) \in \Omega$ where $\Omega = \{-1 \leq x, y, \leq 1\}$ with Dirichlet boundary conditions. Use the following Gaussian-shaped initial condition

$$u(x, y, 0) = \exp\left(-\frac{(x - x_0)^2 + (y - y_0)^2}{2\lambda_0^2}\right)$$

  with shape-parameter $\lambda_0 = 1/8$ and initial center location $(x_0, y_0) = (-\frac{1}{2}, 0)$.
- Investigate the error behaviour at $T_{final} = 1$ after a full rotation of the hill using your modified solver. Discuss pros and cons for low-order vs. high-order.

Enjoy!