Problem Set 8

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.

The focus of todays exercises is the constuction and solution of DG-FEM for elliptic problems.

Let us first consider the problem

$$u_{xx}(x) = f(x), \quad u(x) = \sin(\pi x).$$

- Determine $f(x)$ to have the desired exact solution.
- Formulate and implement a DG-FEM scheme for the 1D Poisson equation. Generalize the flux so you can consider both a stabilized central flux and a "flip-flop" (LDG) flux.
- Check to see if the operator is symmetric for different combinations of boundary conditions – Dirichlet/Neumann at the two boundaries.
- Solve first the problem on $x \in [0, 2]$ with $u(0) = u(2) = 0$. Solve it using both flux types and discuss whether the accuracy is as expected.
- Solve the problem with $x \in [0, 1.5]$ with $u(0) = u_x(2) = 0$. Solve it using both flux types and discuss whether the accuracy is as expected.
- Generalize the solver to also include

$$(au_x)_x = f.$$

- Choose a smooth $a(x)$ and a solution to test the accuracy of the solver.

Let us now also consider the two-dimensional case,

$$\nabla^2 u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in [-1, 1]^2.$$

- Generate several grids with DistMesh or some other grid generator.

- Assume first that $u(x, y) = \sin(\pi x)\sin(\pi y)$ and find $f(x, y)$.
- Formulate and implement a DG-FEM scheme for the 2D Poisson equation. Generalize the flux so you can consider both a stabilized central flux and a "flip-flop" (LDG) flux.
- Solve the problem assuming Dirichlet boundary conditions and look at accuracy of the schemes.
- Use a direct solver and consider the effect of reordering on solution speed and memory usage.
- Consider the generalized problem

  $$\nabla \cdot (a(\mathbf{x})\nabla u) = f(\mathbf{x}), \mathbf{x} \in [-1, 1]^2,$$

  and alter the code to enable the solution of this problem.

If time permits it

- Check that the problem is symmetric and positive definite for different values of N and K.
- Solve the problem using an iterative solver.
- Consider the impact of different preconditioning techniques on solution speed.
- Extend the code to deal with inhomogeneous boundary conditions.

Enjoy!