HOMEWORK 2: FINITE ELEMENT METHOD

Due electronically February 20 at 5:00pm

In Homework 1, you created a finite difference solver for the diffusion equation. In this homework, you will explore the finite element method for solving equations of this type. For this assignment, you will need:

- **fem.py**: python file where you will implement an FEM solver
- **fem_err.py**: python file where you will compute the error in the FEM solutions
- **uniform4.npy, uniform8.npy, uniform16.npy**: numpy files with data to create element meshes with uniform element sizes
- **nonuniform.npy**: numpy file with data to create a nonuniform element mesh

We will consider a simplified 1D diffusion equation with a solute source term, i.e. Poisson's equation:

$$\frac{\partial^2 c}{\partial x^2} = -m$$

This equation has an implicit, constant diffusion coefficient $D = 1$. The solute source term m contributes a constant amount of solute per unit length, and we will apply constant-composition (Dirichlet) boundary conditions such that $c(x{=}0) = c(x{=}1) = 0$. Since there is no time dependence, this is a steady state diffusion equation, and you can easily check that its exact solution for $0 \le x \le 1$ is given by the parabolic composition profile

$$c = \frac{m}{2}(x - x^2)$$

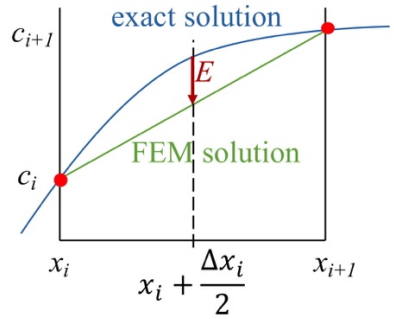
We can assemble a linear finite element model for this system. It looks like this:

$$\begin{bmatrix} \frac{1}{\Delta x_0}+\frac{1}{\Delta x_1} & \frac{-1}{\Delta x_1} & 0 & 0 & \cdots & 0 \\ \frac{-1}{\Delta x_1} & \frac{1}{\Delta x_1}+\frac{1}{\Delta x_2} & \frac{-1}{\Delta x_2} & 0 & \cdots & 0 \\ 0 & \frac{-1}{\Delta x_2} & \frac{1}{\Delta x_2}+\frac{1}{\Delta x_3} & \frac{-1}{\Delta x_3} & \cdots & 0 \\ 0 & 0 & \frac{-1}{\Delta x_3} & \frac{1}{\Delta x_3}+\frac{1}{\Delta x_4} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{1}{\Delta x_{N-2}}+\frac{1}{\Delta x_{N-1}} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ \vdots \\ c_{N-1} \end{bmatrix} = \frac{m}{2} \begin{bmatrix} \Delta x_0 + \Delta x_1 \\ \Delta x_1 + \Delta x_2 \\ \Delta x_2 + \Delta x_3 \\ \Delta x_3 + \Delta x_4 \\ \vdots \\ \Delta x_{N-2} + \Delta x_{N-1} \end{bmatrix}$$

Here, N is the number of elements, and Δx_i where $i \in [0,N{-}1]$ gives the element sizes, which can vary from element to element.

1. (4 pts) Follow the prompt in the **fem.py** docstring to read in a 1D element mesh given in a NPY file, construct the FEM model, and program a FEM solver to compute a composition profile. Turn in your completed **fem.py** file with your solutions filled in; there is no need to turn in the input or output files you used.

2. (4 pts) Recall that the FEM model computes an exact solution to the constitutive equation at the nodes; the FEM solution inside the elements is a linear interpolation of the solution at the nodes:



There are many ways to calculate the difference between the exact and numerical solutions. One simple way that works well for this particular problem is to compute the difference between the curves at the center of each element *i*:

$$E_i = c(x) - \left[\frac{c(x_{i+1}) - c(x_i)}{\Delta x_i} x + c(x_i)\right] \text{ where } x = x_i + \frac{\Delta x_i}{2} \text{ and } c(x) = x - x^2$$

The average of E_i over all *N* elements, $< E >= \frac{1}{N}\sum_i E_i$, is related to the accuracy of the numerical solution. Follow the prompt in the **fem_err.py** docstring to create a function that reads in a numerical solution for our system and returns <*E*>. Turn in your completed **fem_err.py** file with your solutions filled in; there is no need to turn in the input or output files you used.

3. In this problem, you will use your FEM code to explore some effects of mesh selection in FEM models.

   (a) (4 pts) Baseline: Using your code in **fem.py** with $m = 2$ and the mesh given in **uniform4.npy**, calculate the exact composition profile and the FEM solution. Turn in a plot showing both solutions, and report the average absolute error between the exact and numerical solutions.

   (b) (4 pts) Mesh refinement: The mesh in **uniform4.npy** has 4 elements and constant mesh size $\Delta x = 0.25$. The mesh in **uniform8.npy** is more refined, with 8 elements $\Delta x = 0.125$, and that in **uniform16.npy** is more refined yet, with 16 elements and $\Delta x = 0.0625$. Plot the average absolute error as a function of the number of elements, *N*. How does the error scale with *N*? Does the FEM solution seem to be converging on the exact solution?

(c) (2 pts) Mesh adjustment: The mesh in **nonuniform.npy** has only 6 elements, but the elements are of different sizes. What is the average absolute error for this mesh? How does it compare to the error for the uniform meshes?

(d) (4 pts) Your turn: Design a mesh with as few elements as possible to achieve an average absolute error less than or equal to that of the 16 element mesh in **uniform16.npy**. Report your list of Δx_i values, the average absolute error achieved, and a plot showing the exact and numerical solutions.