PHYS 511: Computational Modeling and Simulation - Fall 2017
Assignment #2, due Monday November 6, by 5:00 pm

Solving the Poisson equation in 2D

Electrostatic potentials appear in many applications pertinent to physics, chemistry, or engineering. The electrostatic potential ϕ satisfies the Poisson equation, which in some "natural" units in 2D has the following form

$$\nabla^2\phi(x,y) = -4\pi\rho(x,y), \tag{1}$$

and is subject to some boundary conditions. Here $\rho(x,y)$ is some given charge distribution, and

$$\nabla^2 \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}.$$

In this assignment you will need to solve this equation in a 2D rectangular box $-1 \le x \le 1$, $-1 \le y \le 1$ using the finite difference approach, i.e. by discretizing the Laplacian operator. Finite difference approach in real space is not the most efficient or elegant approach to solve the Poisson equation. Yet, it is a very instructive problem to learn about finite differences.

For the boundary condition use $\phi = 0$ on the sides of the box, i.e.

$$\phi(-1,y) = 0, \quad \phi(1,y) = 0, \quad \phi(x,-1) = 0, \quad \phi(x,1) = 0.$$

For the charge distribution ρ, use a uniformly charged sphere of radius 0.5 placed in the middle of the box:

$$\rho(x,y) = \begin{cases} 1, & r = \sqrt{x^2+y^2} < 0.5 \\ 0, & r = \sqrt{x^2+y^2} > 0.5 \end{cases} \tag{2}$$

1. Write a Fortran program that can use either a three-point or five-point approximation for the second derivatives (there could be a simple user "switch" in your program, e.g. a parameter `np` that one can easily change from 3 to 5). The number of grid points along each dimension should also be adjustable by the user. That will allow you to play with this number and estimate how converged your calculations are. In the production run use $n_x = n_y = 60$ points along each dimension. The program should essentially generate and solve a system of simulataneous linear equations. Since the system is not that huge in size (e.g. 3600), you can use any method to solve it (e.g. find something relevant in LAPACK), even though it may be inefficient. However, you should rememeber that direct methods are generally avoided in actual applications with sparse matrices due to their unfavourable computational scaling with system size.

2. Run the program and output the solution, $\phi(x,y)$, in a data file (e.g. `phi.dat`).

3. Use GNUPLOT or any other relevant software to generate a nice 2D density plot or a 3D plot of your numerical solution in png format (`solution.png`) and see if it looks as you would expect. Make sure the plot has all relevant labels and uses proper units. Also, try to make high resolution (1024×768 pixels or higher). Compare the results obtained with three-point or five-point approximation and/or different mesh size (e.g. 20, 40, 60). Do you see any improvement?

4. Put the source file of your program and the plot with your numerical solution in subdirectory `as2` in your google-drive directory that is shared with the instructor. Any comments/descriptions you want to add should be written in file `report.txt`. Do not submit the data file (`phi.dat`) as it may be too large.