

Physics 420/580: Lab 5

Thursday, October 13, 2011

1. Download `Lab5.tar.gz` from the class website. Unpack the archive and `cd` into the `Lab5` directory. The `make` command will create an executable `relax`, which implements the Jacobi and Gauss-Seidel relaxation algorithms for the two-dimensional Laplace equation ($\nabla^2\phi = 0$). The program reads the initial conditions from a file (several `inputn.dat` files are provided) and outputs a sequence of files `outputabcd.dat` containing the $\phi(x, y)$ values after `abcd` sweeps. It also outputs a file `movie.gp` that can be invoked to view the results as a gnuplot animation.

```
$ ./relax
Usage: relax jacobi|gauss filename
$ ./relax jacobi input1.dat
Error : 0.3675
Error : 0.155625
Error : 0.107188
...
Error : 1.29241e-05
Error : 1.01436e-05
Error : 9.23734e-06
Converged in 58 passes
$ gnuplot movie.gp
```

2. The `relax` program accepts input that is formatted in a particular way. Observe that the input files consist of both numerical data and an array of symbols (`#X0.`). Try to reverse-engineer the file format and generate input files of your own.
3. In this implementation, the x - y plane is discretized into a uniform grid of spacing $\Delta x = \Delta y$. The grid cells are updated until they all locally satisfy the self-consistency condition

$$\frac{1}{(\Delta x)^2} [\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{i,j}] = 0.$$

The Jacobi and Gauss-Seidel methods differ only in whether each $\phi_{i,j}$ is updated using new or old values of its neighbours. Implement the checkerboard update scheme mentioned in class. Which of these three methods converges in the fewest steps?

4. Modify the program so that it can also solve the Poisson equation, $\nabla^2\phi = \rho$. I suggest that you allow for a source term ρ that consists of individual monopoles of unit charge: $\rho(x, y) = \sum_k q_k \delta(x - x_k) \delta(y - y_k)$ with $q_k = \pm 1$. At this level of approximation, $\delta(x) \approx \frac{1}{\Delta x} \theta(\frac{1}{2}\Delta x - x) \theta(x + \frac{1}{2}\Delta x)$. Thus, we have

$$\frac{1}{(\Delta x)^2} [\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{i,j}] = \begin{cases} +\frac{1}{(\Delta x)^2} & \text{positive charge in box } i, j, \\ -\frac{1}{(\Delta x)^2} & \text{negative charge in box } i, j, \\ 0 & \text{otherwise.} \end{cases}$$

Extend the file format so that the symbols `+` and `-` denote a single positive or negative charge. If you are successful, your program should now be able to interpret the `output3.dat` file. Using `output4.dat` as a template, set up lone-charge, electric-dipole, and parallel-plate-capacitor examples.

5. After each sweep, the program writes $\phi_{i,j}$ to a file in a tabular form appropriate for gnuplot's `splot "..."` `matrix` call. Modify the program so that you can choose instead to output the electric field (potential gradient) $\mathbf{E} = \nabla\phi$ in a four-column style: `x y $\partial\phi/\partial x$ $\partial\phi/\partial y$` . (Note that your definition of the gradient will have to change if the cell is at the grid edge or next to a dead cell.) The `movie.gp` file should be changed to use a vector field plotting style:

plot "output0000.dat" using 1:2:3:4 with vectors

[Since the scale of the gradients is set by Δx , you might want to put $dx=0.1$ (or some other appropriate value) in the preamble and write using `1:2:($3/dx):($4/dx)` instead.] Go back and view the field lines for your examples from question 4. Make sure that they flow smoothly, except at the charge locations where they should diverge.

6. Update each cell as a weighted average of its “new” and “old” values.

$$\phi_{i,j} = \frac{\alpha}{4} \underbrace{[\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - (\Delta x)^2 \rho_{i,j}]}_{\text{new}} + (1 - \alpha) \underbrace{\phi_{i,j}}_{\text{old}}.$$

Experiment with the over- ($1 < \alpha < 2$) and under-relaxation ($0 < \alpha < 1$) regimes. What is the optimum value of α ? Is it the same for all initial conditions?