Boundary Value Problems and Relaxation

HW 6: Monday, Oct. 21, 2019
DUE: Wednesday, Oct. 30, 2019
READ: *Numerical Recipes*, Section 1.4 page 24-30, Section 20.0, page 1024-1031
and section 20.5 page 1059-1066.
Handout on array class template.
OPTIONAL: Landau, Paez, and Bordeianu,
finite differences, section 17.1-17.9
finite elements, section 17.10-17.18

This problem will use Gauss-Seidel iteration with successive over-relaxation.

PROBLEM (10 points):

Consider the cylindrically symmetric set of electrodes shown in Figure 1. There are two thin disks and one thick disk, all with an outer radius of R_3. The outside two disks have a potential of V_1 and the inner disk has a potential of V_2. The left disk has an inner radius of R_1 and is at $z = Z_1$. The right two disks have an inner radius of $r = R_2$. The center disk is at $z = 0$ and the right most thick disk starts at $z = Z_2$ and goes to $z = Z_3$.

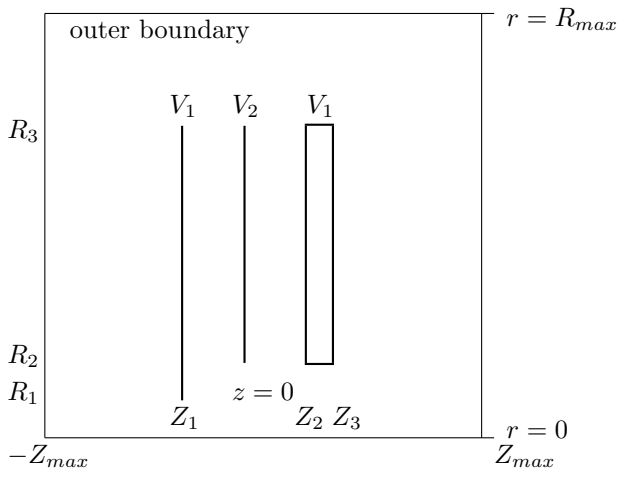

Figure 1: Electrode geometry.

A device similar to this could be used to focus charged particles and is sometimes called an electron lens. For example if an electron beam in an electron microscope or cathode ray tube (i.e. an old fashioned TV picture tube) were traveling down the r=0 axis (from left to right) then changing the voltage V_2 could vary its focus (like an optical lens) to a point on the screen.

Use the following dimensions:

R_1 = 2 mm	Z_1 = -2 mm
R_2 = 3 mm	Z_2 = +1.5 mm
R_3 = 10 mm	Z_3 = 3.0 mm
R_{max} = 15 mm	Z_{max} = 15 mm

You should calculate the potential V and the electrostatic field (i.e. E_r and E_z) in this volume using the finite difference and relaxation methods. The outer boundary should be held at 0 volts and the electrodes at V_1 =0 volts and V_2 = +2500 volts. The outer boundary also appears as if it were an electrode when held at a constant value. To get an accurate answer the boundary should be far away so it does not affect the field in the region of interest (near r=z=0). Therefore R_{max} and Z_{max} are set large. The sampling grid must line up with the positions of the electrodes, which may restrict your choice of grid spacing.

First write a program to find V everywhere inside this boundary using relaxation of the finite difference equations in cylindrical coordinates(see Hint 1 below). You should use a 2D array. You may use the class template (with optional array bounds checking for debugging) as described in a separate handout to allocate the large 2D array(s). Another approach using the STL vector<> template is in Note 3 below. (You may use arrayt.hpp or write your own.) It is easiest to use one array for the potential (type float) and another 2D char array for flags indicating whether each point is on the boundary. Large amounts of memory must be dynamically allocated from the heap as in the arrayt.hpp template or with new/delete. For h=0.10 mm., plot the number of iterations required to reduce the maximum voltage difference per iteration to 1 volt for various relaxation parameters ω between 1 and 2 (maybe ω=1,1.2,1.4,1.6,1.8,...,almost 2.0).

Next, using the optimum value of ω found above, vary h and the final tolerance $|\Delta V|_{MAX}$ until you get about 3 or 4 significant digits (or less than a percent) accuracy (i.e. consistent for small changes in the grid spacing and final tolerance) in the potential at $r = z = 0$ (give a table of values as a function of h and $|\Delta V|_{MAX}$). Remember that the final tolerance (used to test for convergence) is NOT the same as the accuracy. To save time you may assume that this single value is representative of the overall accuracy. Plot V and E_z as a function of z along $r = 0$. Also plot V, E_r and E_z with r = 1.5mm. for $-10mm. < z < +10mm.$ using the final value of h and produce a contour plot of $V(z, r)$ in the region near the electrodes and the axis. You may use a graphics program such as python/matplotlib, or Matlab, or GnuPlot, to make a contour plot.

If this set of electrodes were driven electronically you might need to know the capacitance of these electrodes. Therefore once you have the final potential, calculate the capacitance between the electrodes held at V_1 and those held at V_2 including fringe fields. The capacitance may be found by calculating the total energy in the E field and the energy in the capacitor. In mks units the total energy in the field is:

$$W = \frac{1}{2}CV_C^2 = \frac{1}{2}\epsilon_0 \int |\vec{E}|^2 dv$$

where C is the capacitance in Farads, V_C is the voltage across the capacitor, ϵ_0 = 8.8542 pico-

Farads/meter, $\vec{E}$ is the electrostatic field and $dv = 2\pi r dr dz$ is a volume element. This is a more difficult calculation so you only need to get about 2-3 significant figures in the final value. Quote your answer in picoFarads.

NOTE-1: Once your code is working correctly, you may find it useful to recompile in release mode instead of debug mode so that it runs faster.

NOTE-2: int iZ1 = (int) (Z1/h + 0.5); rounds but int iZ1 = Z1/h; truncates!

NOTE-3: You can also use the STL vector<> class to generate a 2D array (no bounds checking however):

```
#include <vector>
:
vector<float> tmp(n2);
vector< vector<float> > p( n1, tmp)
:
p[i][j] = p[i+1][j] + p[i-1][j] ....
:
```

HINT 1: In cylindrical coordinates the finite difference equations require a special case on axis. The potential $V = \phi(z, r)$ satisfies the Laplace equation:

$$\nabla^2\phi \;=\; \left(\frac{\partial^2}{\partial z^2} + \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right)\phi(z,r) = 0 \tag{1}$$

This may be solved using a finite difference grid with $z = ih$ and $r = jh$ where h=grid spacing and i, j are integers. The Laplace equation becomes:

$$\phi_{i,j} \;=\; \frac{1}{4}\left[\phi_{i,j+1} + \phi_{i,j-1} + \phi_{i+1,j} + \phi_{i-1,j}\right] + \frac{1}{8j}\left[\phi_{i,j+1} - \phi_{i,j-1}\right] \;\;;\;\; j > 0 \tag{2}$$

everywhere except on the axis (j=0) where the above must be replaced by;

$$\phi_{i,j} \;=\; \frac{1}{6}\left[4\phi_{i,j=1} + \phi_{i+1,j=0} + \phi_{i-1,j=0}\right] \;\;;\;\; j = 0 \tag{3}$$

HINT 2: To produce a contour plot in Matlab print all the values in a file separated by spaces (one line of the file per line in the array). Print the corresponding r and z values to label the axes into separate files. Then:

```
load hw6b.dat;
load hw6br.dat;
y = hw6br;
load hw6bz.dat;
x = hw6bz;
[Cs,h] = contour( x, y, hw6b );
```

```
clabel( Cs, h );
xlabel( 'position z (in mm.)' );
ylabel( 'position r (in mm.)' );
print -deps fig6d.eps;
```

This will print a postscript version of the plot to a file. This file can be printed from ghostview or inserted into a word or LaTeX document etc.

HINT 3: To produce a contour plot in python/matplotlib print all the values in a file separated by spaces (one line of the file per line in the array). Print the corresponding r and z values to label the axes into separate files. Then in python:

```
from pylab import *

clf()
data = loadtxt( "hw6b.dat", 'float' )
r = loadtxt( "hw6br.dat" )
z = loadtxt( "hw6bz.dat" )
cs = contour( r, z, data, 10 )  # only lines
clabel( cs )  # label contour levels
xlabel( "r (in mm.)" )
ylabel( "z (in mm.)" )
savefig('pyContour.eps')
show()
```

This will print a postscript version of the plot to a file (use .png for a bitmap file). This file can be printed from ghostview or inserted into a word or LaTeX document etc.

HINT 4: To produce a contour plot in Scilab print all the potential values in a file separated by spaces (one line of the file per line in the array, for this example each line has the same value of z). Print the corresponding r and z values to label the axes into separate files (hw6br.dat and hw6bz.dat in this example). (Just like Matlab). To contour an array from file hw6b.dat with 10 levels:

```
//
//  Scilab script to make contour plot
//
r = read( 'hw6br.dat', -1, 1 );
z = read( 'hw6bz.dat', -1, 1 );
v = read( 'hw6b.dat', size( z ), size( r ) );
contour2d( z, r, v, 10);
xtitle( 'potential', 'z (in mm.)', 'r (in mm.)' );
xs2eps(0,'fig6a.eps');  // output graph as postscript file
```

You can also print a postscript version of the plot to a file using the file menu item 'export'. This file can be printed from ghostview or inserted into a word or LaTeX document etc.

HINT 5: If you use Gnuplot (probably under Linux) the following code segment writes the array v(i,j) (defined using the array<> template) in a form that GnuPlot can read is:

```
/*  Print out v on grid for contour or 3D surface  plot with GnuPlot
    Remember that GnuPlot wants a blank line between rows */

    fp = fopen( "hw6b.dat", "w+");
    for (ir=0; ir<nr1; ir++ ) {
        r = ir*h ;
        for( iz=iz1; iz<=iz2; iz++) {
            z = iz*h;
            fprintf(fp, "%f %f %f\n", z, r, v(iz,ir) );
        }
        fprintf( fp, "\n");
    }
    fclose( fp );
```

GnuPlot is a command-line based interface. The commands to produce a contour plot in GnuPlot are:

```
set term x11
set hidden3d
set nosurface
set view 0,0
set clabel
set contour
set xlabel "Z"
set ylabel "R"
set cntrparam levels 7 auto
set title "Contour Plot"
set parametric
splot "hw6b.dat" with lines
```

These commands may be typed in directly or entered into a file. If you want to see a 3D surface plot then replace two command lines as shown below:

```
set nosurface  change to->set surface
set view 0,0 change to->set view 30,60
```

Once it looks OK on the screen you can create a postscript file that has a higher resolution (i.e. looks better) and print it (see HW-1 for details).