Physics 120 – Problem Set # 2

(due Friday, January 25)

Throughout this course, homework problems from Griffiths' textbook will be from the Third Edition (with the brown and black cover). If you have the Second Edition (with the blue and black cover), check with a friend to be sure that you are doing the correct problems.

1. Griffiths, problems 2.7, 2.8, 2.12, 2.18. Problem 2.7 is a brute-force calculation. The other problems are straightforward, and 2.18 is actually interesting. You might want to do 2.12 before you do 2.7 (or do 2.7 if you can't figure out 2.12).

2. Griffiths, problem 2.21.

3. Griffiths, problem 2.26.

4. Griffiths, problem 2.32(a) and 2.33.

5. In class, we saw that the electrostatic field is an incompressible fluid, flowing from positive to negative charges. One way to visualize this field is to draw the *field lines*, the lines of flow which follow the electric field vectors.

   (a) A field line $\vec{\ell}(s)$ can be found by solving one of the differential equations

   $$\frac{d}{ds}\vec{\ell}_1 = \vec{E}(\ell_1) \ , \qquad \frac{d}{ds}\vec{\ell}_2 = \frac{\vec{E}(\ell_2)}{|\vec{E}(\ell_2)|} \ . \tag{1}$$

   Why is this true? How do the curves $\vec{\ell}_1(s)$ and $\vec{\ell}_2(s)$ differ?

   (b) Justify the following statement: If we have an array of charges in the $(\hat{x}, \hat{y})$ plane, a field line that emerges from a charge in a direction in the plane will always stay in the plane.

   (c) Justify the following statement: If the total charge Q of an array of charges satisfies $Q \geq 0$, we can find all of the field lines by following the field lines which emerge from the positive charges.

   (d) Sketch the field lines in the $(\hat{x}, \hat{y})$ plane for the following collections of charges (all with $z = 0$).

      i. A positive charge $Q = +1$ at $(x, y) = (0, 0)$.

      ii. Charges $Q_1 = +1$ at $(x_1, y_1) = (0, 1)$, $Q_2 = -1$ at $(x_2, y_2) = (0, -1)$.

      iii. Charges $Q_1 = +1$ at $(x_1, y_1) = (0, 1)$, $Q_2 = +1$ at $(x_2, y_2) = (0, -1)$, $Q_3 = -1$ at $(x_3, y_3) = (1, 0)$, $Q_4 = -1$ at $(x_4, y_4) = (-1, 0)$.

      iv. Charges $Q_1 = +3$ at $(x_1, y_1) = (0, 1)$, $Q_2 = -1$ at $(x_2, y_2) = (0, -1)$.

(e) Use the java plotter to draw the field lines in case (i) of part (d). You can find an applet FieldLines.java that does this at the course URL: `http://www.slac.stanford.edu/~mpeskin/Physics120/java/`. Download:

`FieldLines.java`, `FieldLines.html`

and the `Plotter` materials from the previous problem set. The source code for `FieldLines.java` is given at the end of the problem set. Here are some features that you might like to note:

i. The function Efield returns a type called `double[]`. This is a vector with real (double-precision) arguments. In java, vectors begin with the index 0. To avoid confusion, I have asked for a 3-component vector so that the entries 1 and 2 are available. In principle, I could have written separate functions to compute the $\hat{x}$ and $\hat{y}$ components of $\vec{E}$, but this approach is more coherent.

ii. In the computation of $\vec{E}$, I have omitted $4\pi\epsilon_0$. (Why is this permissible?)

iii. I have used the second of the equations in (1) to compute the field lines. (Why is this the better choice?)

iv. `plotOneLine` is the routine that plots a single field line by integrating the differential equation. I hope you can recognize that the loop is the discretization of the differential equation. The line `if (Eval > 50.0) break;` tests whether the field lines have come near a negative charge. The command `break;` means, exit the loop.

(f) Modify the applet to draw the field lines for the other three cases in (d). Make the plotting region as large as you need to to see the global features. Hand in the three plots and the corresponding java code.

```
import java.awt.*;
import java.applet.Applet;
import java.util.*;

public class FieldLines extends Plotter {

 int Nlines = 16;
 double smallcircle = 0.3;
 double tstep = 0.1;

    //  location of charge
 double x1 = 0.0;  double y1 = 0.0;

 public void setup(){
   setPlotSize(400,400);
   setLimits(-10.0, 10.0, -10.0, 10.0);
   setTicks(1.0, 1.0);
 }

 public void plot(Graphics g){
   g.setColor(Color.red);
   plotCharge(g,x1,y1);
   g.setColor(Color.black);
   plotFieldLines(g,x1,y1);
 }

 public void plotCharge(Graphics g, double x0,double y0){
    plotStream P = new plotStream();
    P.add(x0,y0);
    plotPoints(g,P);
 }

 void plotFieldLines(Graphics g, double x0, double y0){
    int i;
    for (i = 0; i < Nlines; i++){
       plotOneLine(g, x0,y0, (2.0 * Math.PI * i)/Nlines);
    }
 }
```

Figure 1: The source file `FieldLines.java`.

```
 double[] Efield(double x, double y){
    double [] efield = new double[3];
    double r1 = Math.sqrt((x-x1)*(x-x1) + (y-y1)*(y-y1));
    efield[1] = (x-x1)/(r1*r1*r1);
    efield[2] = (y-y1)/(r1*r1*r1);
    return efield;
 }

 void plotOneLine(Graphics g, double x0, double y0, double theta){
    double [] E = new double[3];
    plotStream P = new plotStream();
    double x = x0 + smallcircle * Math.cos(theta);
    double y = y0 + smallcircle * Math.sin(theta);
    double t;
    for (t = 0.0; t <= 30.0; t+= tstep){
      P.add(x,y);
      E = Efield(x,y);
      double Eval = Math.sqrt(E[1]*E[1] + E[2]*E[2]);
      if (Eval > 50.0) break;
      x += tstep*E[1]/Eval;
      y += tstep*E[2]/Eval;
    }
    plotCurve(g,P);
  }

}
```

Figure 2: The source file `FieldLines.java` (cont.).