Integration in Electrostatics with a Computational Perspective

David Roundy

Eric Krebs Jeff Schulte

National Science Foundation
DUE-1141330

Oregon State University

The Context: A computational lab for the Paradigms

The class

- Covers same physics content as the junior-year Paradigms.
- 1 credit, meets 3 hours per week in-class, no homework.
- Uses pair programming, python with matplotlib and numpy.
- No example code provided to students: they Google for help.
- Begins with six weeks of electrostatics.

The students

- This was an elective course.
- We had 8 students this Fall.
- Most had previous computational course with visual python.

Introduction

- Setting up integrals in electrostatics is challenging for students.
- These integrals are very different from what is taught in calculus.

$$V(\vec{r}) = \int \frac{k\,dq'}{|\vec{r} - \vec{r}'|} \qquad \vec{E}(\vec{r}) = \int k \frac{\vec{r} - \vec{r}'}{|\vec{r} - \vec{r}'|^3} dq'$$

Conclusions

- "Chopping and adding"[1] is made explicit in computation.
- Once you have written down an integral, the problem becomes easy.

[1] For more information, attend Corinne's talk in this session one hour ago.

Six weeks of electrostatics

Schedule (remember: 3 hours per week)

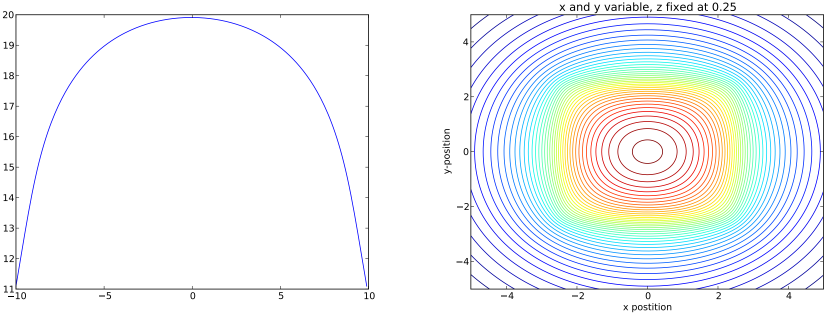
- Week 1: Potential of four point charges
  - Writing functions in python.
  - Visualizing a scalar field.
- Week 2-3: Potential of a square of surface charge
  - Programming loops.
  - Viewing integration as “chopping and adding.”
- Week 4-6: Electric field of a solid cylinder of charge
  - Integrating a vector quantity.
  - Using cylindrical coordinates.
  - Visualizing a vector field.

For each project pairs of students...

- ... write down function on paper in math notation.
- ... write python function to evaluate the field.
- ... visualize the field.
- ... present their code to the class.

Week 2 and 3: Potential of a square of surface charge

- 1/2 hour writing down the integral on paper.
- Students struggled with getting dimensions correct (omitting $\Delta x \Delta y$)
- Students struggled with creating loops.
- Ended with students presenting their code to the class.



Students reported learning to name their variables with “physics” names.

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2}\int_{-L/2}^{L/2} \frac{k\sigma dx' dy'}{\sqrt{(x-x')^2 + (y-y')^2 + z^2}}$$

Student code (distance $\rightarrow$ dist)

```
def V(x,y,z):
    dx = 0.01
    dy = dx
    v_total = 0
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v
    return v_total
```

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2}\int_{-L/2}^{L/2} \frac{k\sigma\, dx'\, dy'}{\sqrt{(x - x')^2 + (y - y')^2 + z^2}}$$

Student code (distance → dist) (comments mine)

```
def V(x,y,z):                # distinction between r and r'
    dx = 0.01
    dy = dx
    v_total = 0
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v
    return v_total
```

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2} \int_{-L/2}^{L/2} \frac{k\sigma\, dx'\, dy'}{\sqrt{(x-x')^2 + (y-y')^2 + z^2}}$$

Student code (distance → dist) (comments mine)

```
def V(x,y,z):                 # distinction between r and r’
    dx = 0.01                 # limits of integration
    dy = dx
    v_total = 0
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v
    return v_total
```

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2}\int_{-L/2}^{L/2} \frac{k\sigma\, dx'\, dy'}{\sqrt{(x-x')^2+(y-y')^2+z^2}}$$

Student code (distance → dist) (comments mine)

```
def V(x,y,z):                 # distinction between r and r'
    dx = 0.01                 # limits of integration
    dy = dx                   # |r-r'|
    v_total = 0
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v
    return v_total
```

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2}\int_{-L/2}^{L/2} \frac{k\sigma\, dx'\, dy'}{\sqrt{(x-x')^2+(y-y')^2+z^2}}$$

Student code (distance → dist) (comments mine)

```
def V(x,y,z):                # distinction between r and r'
    dx = 0.01                # limits of integration
    dy = dx                  # |r-r'|
    v_total = 0              # a little bit of charge
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v
    return v_total
```

Week 2 and 3: Potential of a square of surface charge

Math solution

$$V(\vec{r}) = \int_{-L/2}^{L/2}\int_{-L/2}^{L/2} \frac{k\sigma\, dx'\, dy'}{\sqrt{(x-x')^2+(y-y')^2+z^2}}$$

Student code (distance → dist) (comments mine)

```
def V(x,y,z):                  # distinction between r and r'
    dx = 0.01                  # limits of integration
    dy = dx                    # |r-r'|
    v_total = 0                # a little bit of charge
    for xp in numpy.arange(-length/2, length/2, dx):
        x_dist = (x - xp)**2
        for yp in numpy.arange(-length/2,length/2,dy):
            y_dist = (y - yp)**2
            z_dist = (z)**2
            dist = (x_dist**2 + y_dist**2 + z_dist**2)**.5
            v = k * sigma * dx**2 / dist
            v_total += v # add up the little bits of potential
    return v_total
```

Introduction

- Setting up integrals in electrostatics is challenging for students.
- These integrals are very different from what is taught in calculus.

$$V(\vec{r}) = \int \frac{k\, dq'}{|\vec{r} - \vec{r}'|} \qquad \vec{E}(\vec{r}) = \int k \frac{\vec{r} - \vec{r}'}{|\vec{r} - \vec{r}'|^3} dq'$$

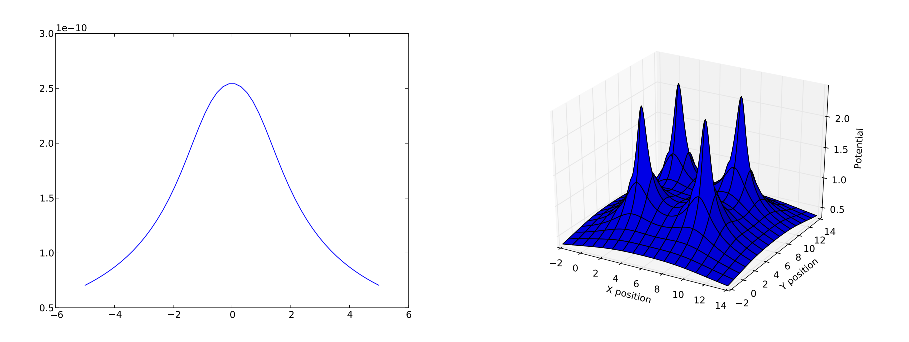
Conclusions

- "Chopping and adding"[2] is made explicit in computation.
- Once you have written down an integral, the problem becomes easy.

[2] For more information, attend Corinne's talk in this session one hour ago.

Week 1: Potential of four point charges

- Ended with student presentations of their code.

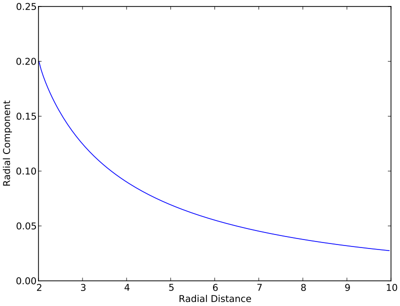


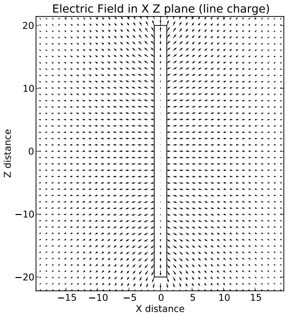
Students reported learning to "get the little parts done [and tested] first," and reported learning a variety of programming concepts

(if/else, arrays, meshgrid, etc.).

Week 4-6: Electric field of solid cylinder of charge

- About 1 1/2 hours writing down the integral on paper.
- Students struggled with cylindrical coordinates.
- Students struggled with integrating a vector quantity.
- Ended with students studying and presenting the code of *another* pair.





Week 4-6: Electric field of solid cylinder of charge

$$E_x(\vec{r}) = \int_{-L/2}^{L/2} \int_0^{2\pi} \int_0^R k\rho \frac{x - r'\cos\phi'}{\left(\sqrt{r^2 + r'^2 - 2rr'\cos\phi' + (z-z')^2}\right)^3} r'\,dr'\,d\phi'\,dz'$$

Student code (I cut a print statement)

```
def E_x(x,y,z):                    # Cartesian coordinates for position, computes Ex
    dr_p = 0.01
    E_x = 0
    r_p = 0
    r = (x**2 + y**2)**(1/2)       # Computes cylindrical r coordinate (mixed coordinates)
    dphi_p = np.pi/50
    phi_p = 0
    dz_p = 0.01
    z_p = -length/2                # Uses while loops rather than for loops...
    while z_p < length/2:          # ...this scatters the limits of integration
        while phi_p < 2*np.pi:
            while r_p < radius:
                r_minus_r_p = (r**2 + r_p**2 - 2 * r * r_p * np.cos(phi_p) + (z - z_p)**2)**(1/2)
                dE_x = (((x - r_p * np.cos(phi_p))*r_p)* dr_p * dphi_p * dz_p) / r_minus_r_p**3
                E_x = E_x + dE_x
                r_p = r_p + dr_p   # This pair oddly broke up the tiny chunk of volume
            r_p = 0                # They entirely omit the charge density and k
            phi_p = phi_p + dphi_p
        phi_p = 0
        z_p = z_p + dz_p
    return E_x
```

Week 4-6: Electric field of solid cylinder of charge

Math solution

$$E_x(\vec{r}) = \int_{-L}^{L}\int_0^{2\pi}\int_0^{R} k\rho \frac{r\cos\phi - r'\cos\phi'}{\left(r^2 + r'^2 - 2rr'\cos\phi' + (z-z')^2\right)^{3/2}} r'dr'd\phi'dz'$$

Student code (I broke a very long line of code)

```
def Ex(r,phi,z):                  # Cylindrical coordinates for position, find Ex
  exfield = 0                     # They entirely omit the charge density and k
  for rp in np.arange(0,R,drp):   # Very oddly broke up the tiny chunk of volume
    for phip in np.arange(0,2*np.pi,dphip):
      for zp in np.arange(-L,L,dzp):
        exfield = exfield + ((rp*r*np.cos(phi)-rp**2*np.cos(phip))/
            (r**2+rp**1-2*r*rp*np.cos(phi-phip)+(z-zp)**2)**(3/2))*drp*dphip*dzp
  return exfield
```