

In lecture, we explored the concept of a random walk, using a set of different models of drunks. Below is the code we used for locations and fields and the base class of drunks – you should not have to study this code in detail, since you have seen it in lecture.

CODE FROM LECTURE

[Click to see the Location, Field, and Drunk classes from lecture](#)

```

import pylab

class Location(object):
    def __init__(self, x, y):
        """x and y are floats"""
        self.x = x
        self.y = y

    def move(self, deltaX, deltaY):
        """deltaX and deltaY are floats"""
        return Location(self.x + deltaX, self.y + deltaY)

    def getX(self):
        return self.x

    def getY(self):
        return self.y

    def distFrom(self, other):
        ox = other.x
        oy = other.y
        xDist = self.x - ox
        yDist = self.y - oy
        return (xDist**2 + yDist**2)**0.5

    def __str__(self):
        return '<' + str(self.x) + ', ' + str(self.y) + '>'

class Field(object):
    def __init__(self):
        self.drunks = {}

    def addDrunk(self, drunk, loc):
        if drunk in self.drunks:
            raise ValueError('Duplicate drunk')
        else:
            self.drunks[drunk] = loc

    def moveDrunk(self, drunk):
        if not drunk in self.drunks:
            raise ValueError('Drunk not in field')
        xDist, yDist = drunk.takeStep()
        currentLocation = self.drunks[drunk]
        #use move method of Location to get new location
        self.drunks[drunk] = currentLocation.move(xDist, yDist)

    def getLoc(self, drunk):
        if not drunk in self.drunks:
            raise ValueError('Drunk not in field')
        return self.drunks[drunk]

import random

class Drunk(object):
    def __init__(self, name):
        self.name = name
    def __str__(self):
        return 'This drunk is named ' + self.name

```

The following function is new, and returns the actual x and y distance from the start point to the end point of a random walk.

```
def walkVector(f, d, numSteps):
    start = f.getLoc(d)
    for s in range(numSteps):
        f.moveDrunk(d)
    return(f.getLoc(d).getX() - start.getX(),
           f.getLoc(d).getY() - start.getY())
```

DRUNK VARIATIONS

Here are several different variations on a drunk.

```
class UsualDrunk(Drunk):
    def takeStep(self):
        stepChoices = \
            [(0.0,1.0), (0.0,-1.0), (1.0, 0.0), (-1.0, 0.0)]
        return random.choice(stepChoices)

class ColdDrunk(Drunk):
    def takeStep(self):
        stepChoices = \
            [(0.0,0.9), (0.0,-1.03), (1.03, 0.0), (-1.03, 0.0)]
        return random.choice(stepChoices)

class EDrunk(Drunk):
    def takeStep(self):
        ang = 2 * math.pi * random.random()
        length = 0.5 + 0.5 * random.random()
        return (length * math.sin(ang), length * math.cos(ang))

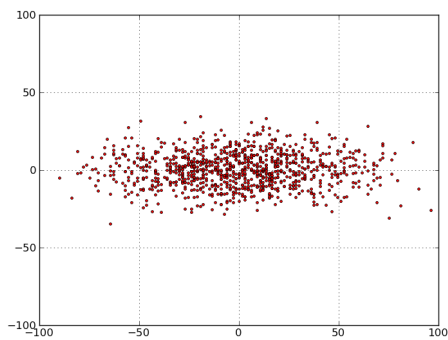
class PhotoDrunk(Drunk):
    def takeStep(self):
        stepChoices = \
            [(0.0, 0.5),(0.0, -0.5),
             (1.5, 0.0),(-1.5, 0.0)]
        return random.choice(stepChoices)

class DDrunk(Drunk):
    def takeStep(self):
        stepChoices = \
            [(0.85, 0.85), (-0.85, -0.85),
             (-0.56, 0.56), (0.56, -0.56)]
        return random.choice(stepChoices)
```

PROBLEM

Suppose we use a Monte Carlo simulation to simulate a random walk of a class of drunk, returning a collection of actual distances from the origin for a set of trials.

Each graph below was generated by using one of the above five classes of a drunk (UsualDrunk, ColdDrunk, EDrunk, PhotoDrunk, or DDrunk). For each graph, indicate which Drunk class is mostly likely to have resulted in that distribution of distances. Click on each image to see a larger view.



Graph 1

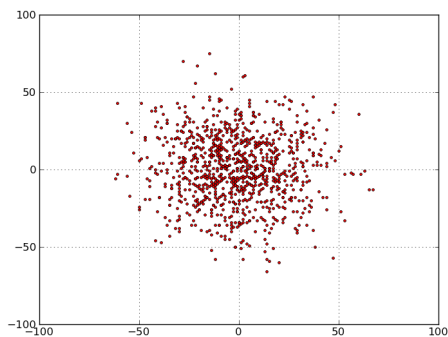
PhotoDrunk ▼

Final Check

Save

You have used 1 of 2 submissions

PROBLEM 4-2 (2/2 points)



Graph 2

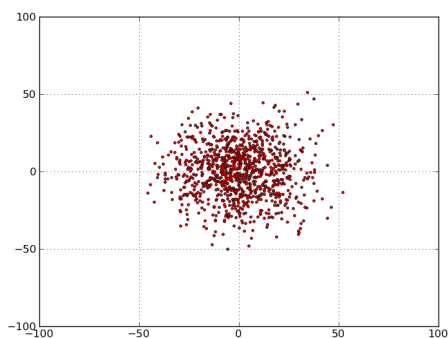
UsualDrunk ▼

Final Check

Save

You have used 1 of 2 submissions

PROBLEM 4-3 (2/2 points)



Graph 3

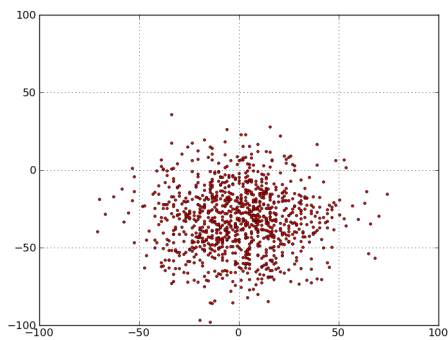
EDrunk ▼

Final Check

Save

You have used 1 of 2 submissions

PROBLEM 4-4 (2/2 points)



Graph 4

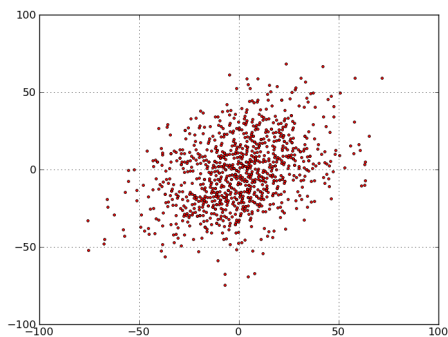
ColdDrunk ▼

Final Check

Save

You have used 1 of 2 submissions

PROBLEM 4-5 (2/2 points)



Graph 5

DDrunk ▼

Final Check

Save

You have used 1 of 2 submissions



EdX offers interactive online classes and MOOCs from the world's best universities. Online courses from MITx, HarvardX, BerkeleyX, UTX and many other universities. Topics include biology, business, chemistry, computer science, economics, finance, electronics, engineering, food and nutrition, history, humanities, law, literature, math, medicine, music, philosophy, physics, science, statistics and more. EdX is a non-profit online initiative created by founding partners Harvard and MIT.

© 2014 edX, some rights reserved.

[Terms of Service and Honor Code](#)

About & Company Info

[About](#)

[News](#)

[Contact](#)

[FAQ](#)


[edX Blog](#)

[Donate to edX](#)

Follow Us

 [Twitter](#)

 [Facebook](#)

 [Meetup](#)

 [LinkedIn](#)

 [Google+](#)

