Lab Group Members: Nick Taylor, Matthew Withey, Tatsushi Matsumoto

EE460J Lab 1 Report

Written Questions:

1.
   a. P(X = 1) = **7/12**
   b. P(X = 1 | Y = 1) = **2/3**
   c. Var(X) = (5/12)(0 - 7/12)^2 + (7/12)(1 - 7/12)^2 = (5/12)(49/144) + (7/12)(25/144) = **35/144**
   d. P(X = 0 | Y = 1) = **1/3**
      Var(X | Y = 1) = (1/3)(0 - 2/3)^2 + (2/3)(1 - 2/3)^2 = **2/9**
   e. E[X^3 + X^2 + 3Y^7 | Y = 1] = (1/3)(3) + (2/3)(5) = **13/3**

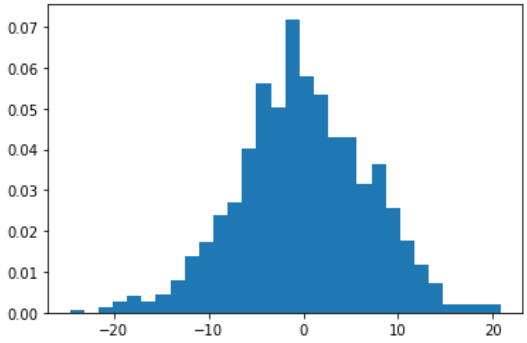
2. V3[x,y,z] x + y + z = 0
   V3 . v1 = 0, v2.v3 = 0
   Y + z = 0, x = 0
   V3 = [0, 1, -1]
   P1 = 3v1
   P1 = [3, 3, 3]
   For p2
   1 = v1 + v2
   2 = v1 + v3
   3 = v1 - v3
   V1 = 5/2
   V2 = -3/2
   V3 = -½
   P2 = (5/2)v1 +(-3/2)v2 + (-½)v3
   P2 = [5/2 - 3/2, 5/2, 5/2]
   P2 = [1, 5/2, 5/2]
   For p3
   P3 = (½)v1 + (-½)v2 + (-½)v3
   P3 = [0, 1/2, ½]

3. X is a random variable for the number of heads in 100 throws, prob of heads = ⅔,
   Using binomial pmf:
   P(X <= 50) = 0.000419

Coding Questions:

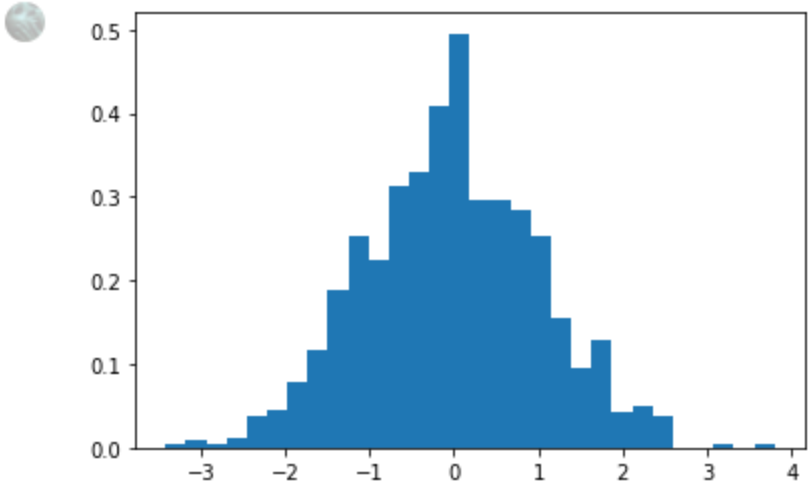
1.

```python
#Question 1

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats
import math

mu, sigma = -10, 5
s1 = np.random.normal(mu, sigma, 1000)
mu, sigma = 10, 5
s2 = np.random.normal(mu, sigma, 1000)
s = s1 + s2

count, bins, ignored = plt.hist(s, 30, density=True)

plt.show()

print("Mean = " + str(np.mean(s)))
print("Variance = " + str(np.var(s, 0)))

print("Mean is close to the sum of the two gaussians' means. The standard deviation also became larger.")
```




```
Mean = -0.013341485782995441
Variance = 47.73481945060966
Mean is close to the sum of the two gaussians' means. The standard deviation also became larger.
```

```
#Question 2

n = 1000
total = np.empty([n, 1], float)

for i in range(n):
  s = np.random.binomial(1, 0.5, n)
  s[s==0] = -1
  z = (1/math.sqrt(n))*np.sum(s)
  total[i] = z
plt.hist(total, 30, density=True)
plt.show()
```



2.

```
#Question 3

mu, sigma = 0, 5
s = np.random.normal(mu, sigma, 25000)
sum = np.sum(s)
mean = sum/25000
sd = np.sqrt(np.sum((s-mean)**2)/24999)

print("Mean = " + str(mean))
print("Standard Deviation = " + str(sd))
```

```
Mean = -0.016289211468201347
Standard Deviation = 5.0408782911687275
```

3.

4.

```
#Question 4

mean = [-5,5]
cov = [[20, .8],[.8, 30]]

s = np.random.multivariate_normal(mean, cov, 10000)

sum = 0
for i in s:
    sum += i
s_mean = sum/10000

print("Mean = ", s_mean)

ex = s_mean[0]
ey = s_mean[1]

xvar = np.sum((s.T[0] - ex)**2)/9999
yvar = np.sum((s.T[1] - ey)**2)/9999

xycov = 0
for i in s:
    xycov += (i[0] - ex)*(i[1] - ey)
xycov = xycov/9999

s_cov = [[xvar, xycov],[xycov, yvar]]

print("Covariance Matrix = ", s_cov)
```

```
Mean =  [-5.00833185  5.07925922]
Covariance Matrix =  [[20.067504846213755, 0.8319279722570775], [0.8319279722570775, 29.392940563879502]]
```

5.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats
import math
import os

df = pd.read_csv(os.getcwd() + "\HW1\PatientData.csv", header=None)
print("# Patients = " + str(len(df)))
print("# Features = " + str(len(df.columns)))
print("The first four features seem to be age, sex(0m, 1f),
height(cm), and weight(kg).")
for column in df:
    temp = df[column]
    temp.replace('?', np.nan, inplace=True)
    temp = pd.to_numeric(temp)
    mean = temp.mean()
```

```
    temp.replace(np.nan, mean, inplace=True)
    df[column] = temp
print("By plotting different features alongside the patient
condition(i.e. x = feature, y = condition), we could find which
features correlate the strongest positively/negatively with patient
condition by seeing how clustered the points are. From there, we
could hypothesize which features impact patient condition the
strongest by measuring the correlation, and hopefully be able to
predict patient condition by using a combination of those features
in the future.")
print("We think that the 3 most important features would be the ones
that cluster most strongly in our graphs. We could measure this by
finding the average or 'center' of all of our points on each graph,
and then finding the average distance from center on each graph,
perhaps accounting for outliers.")
```