CS573 DM HW0 SOLUTION

Q1 (4 pts)

(a) By the linearity of Normal distribution, the distribution of (Y_1, Y_2) is Normal distribution.

$$E[Y_1] = E[X_1] + 2E[X_2] = 0$$
$$Var[Y_1] = Var[X_1] + 4Var[X_2] = 5$$

$$E[Y_2] = 2E[X_1] + E[X_2] = 0$$
$$Var[Y_2] = 4Var[X_1] + Var[X_2] = 5$$

$$Cov[Y_1, Y_2] = Cov[Y_2, Y_1] = 2Var[X_1] + 2Var[X_2] = 4$$

Therefore, $(Y_1, Y_2) \sim Normal(\mu, \mathbf{\Sigma})$, where

$$\mu = [0, 0]$$
$$\mathbf{\Sigma} = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$$

(b)
```
import numpy as np
mu, sigma = 0, 1 # mean and standard deviation
N = 100
x_1 = np.random.normal(mu, sigma, N)
x_2 = np.random.normal(mu,sigma,N)

# Q1 (b)
import matplotlib.pyplot as plt
plt.scatter(x_1, x_2)
plt.xlim(-2, 2)
plt.ylim(-2, 2)
plt.xlabel('x1')
plt.ylabel('x2')
plt.show()
```

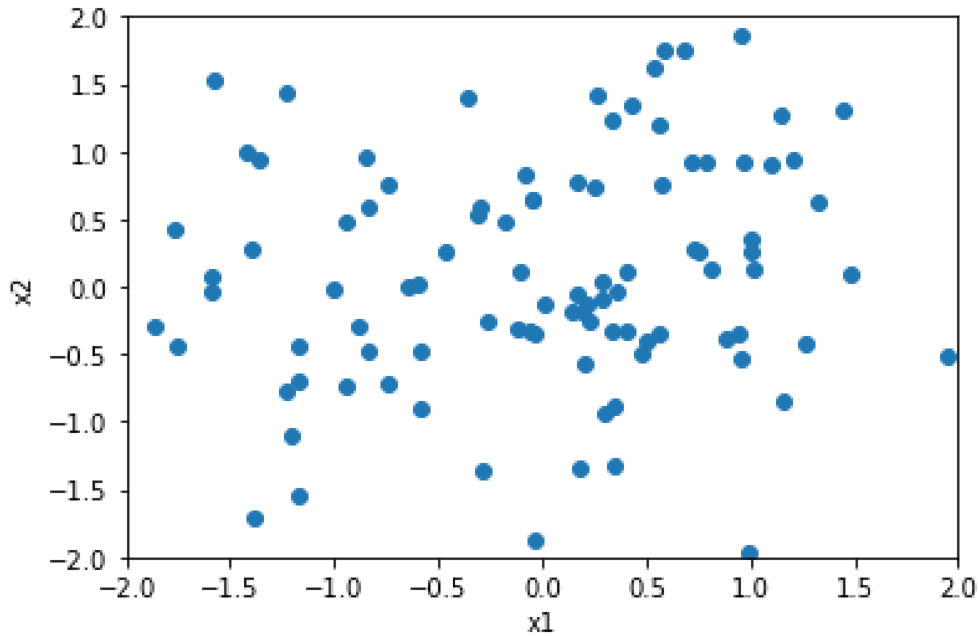

Figure 1: Scatter plot with 100 samples of (X_1, X_2)

(c)
```
# Q1 (c)
count = 0.0
for i in range(N):
    if((x_1[i]**2+x_2[i]**2) < 0.5**2):
        count+=1
print(count/N)
```

$$P(x_1^2 + x_2^2 < 0.5^2) \approx 0.11$$

(d)
```
# Q1 (d)
mu, sigma = 0, 1 # mean and standard deviation
N = 100
x_1 = np.random.normal(mu, sigma, N)
x_2 = np.random.normal(mu,sigma,N)
x_3 = np.random.normal(mu,sigma,N)
count = 0.0
for i in range(N):
    if((x_1[i]**2+x_2[i]**2+x_3[i]**2) < 0.5**2):
        count+=1
print(count/N)
```

$$P(x_1^2 + x_2^2 + x_3^2 < 0.5^2) \approx 0.02$$

(e)
```
# Q1 (e)
random_vars = []
n = 1000
mu, sigma = 0, 1 # mean and standard deviation
N = 100
count = 0.0
for i in range(n):
    random_vars.append(np.random.normal(mu, sigma, N))
for i in range(N):
    squareSum = 0.0
    for j in range(n):
        squareSum +=random_vars[j][i]**2
    if(squareSum >=0.5**2):
        continue
    else:
        count+=1
print(count/N)
print(count)
```

$$P(x_1^2 + x_2^2 + x_3^2 + ... + x_1000^2 < 0.5^2) \approx 0.0$$

(f) As $n \to \infty$, $P(x_1^2 + x_2^2 + x_3^2 + ... + x_n^2 < 0.5^2) \to 0$.
To prove this, we first need to understand the distribution of X_i^2. One can show that X_i^2 is a Chi-square distribution with 1 degree of freedom, where mean $\mu = 1$ and variance is $\sigma^2 = 2$.
Let $Z_n = \sum_{i=1}^{n} X_i^2$. By Central Limit Theorem

$$\begin{aligned}
\lim_{n\to 0} P[Z_n < 0.5^2] &= \lim_{n\to 0} P[\frac{Z_n - n\mu}{\sqrt(n)\sigma} < \frac{0.25 - n\mu}{\sqrt(n)\sigma}] \\
&= \lim_{n\to 0} P[\frac{Z_n - n}{\sqrt{2n}} < \frac{0.25 - n}{\sqrt{2n}}] \\
&= \lim_{n\to 0} \Phi(\frac{0.25 - n}{\sqrt{2n}}) \\
&= \Phi(-\infty) = 0
\end{aligned}$$

Where $\Phi(x)$ is the standard normal cdf evaluated at x.

Q2 (4 pts)

(a) By definition of conditional probability, $P(A, C) > 0 \Rightarrow P(C)! = 0$.

$$P(B|A, C) = \frac{P(A, B, C)}{P(A, C)}$$

$$= \frac{P(A|B,C)P(B|C)P(C)}{P(A|C)P(C)}$$
$$= \frac{P(A|B,C)P(B|C)}{P(A|C)}$$

(b) First, we need to write out the likelihood function:

$$L(\mu,\sigma) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$
$$= \sigma^{-n}(2\pi)^{-n/2}\exp[-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2]$$

The log-likelihood function is

$$logL(\mu,\sigma) = -n\log(\sigma) - n/2\log(2\pi) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2$$

Taking the partial derivative of the log likelihood with respect to μ, and setting to 0

$$\frac{\partial logL}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{n}(x_i-\mu) \equiv 0$$
$$\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Taking the partial derivative of the log likelihood with respect to σ, and setting to 0

$$\frac{\partial logL}{\partial \sigma} = -\frac{1}{\sigma^3}[-n\sigma^2 + \sum_{i=1}^{n}(x_i-\mu)^2] \equiv 0$$
$$\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^{n}(x_i-\hat{\mu})^2}{n}}$$

(c)
```
# Q2 (c)
import random
# generate data
mu, sigma = random.random(), random.random() # mean and standard deviatio
n, N = 100, 1000
x = np.random.normal(mu, sigma, (n,N))

# initialization
mu_hat, sigma_hat = random.random(), random.random()
lr,count,maxIter = 1,0,10000
print (mu, sigma)

# gradient descent
```

```
while count<maxIter :
    if(count%100 == 0):
        print ('iter',count,':', mu_hat, sigma_hat)
    for i in range(N):
        current_data = x[:,i]
        mu_old, sigma_old = mu_hat, sigma_hat
        mu_hat += lr*1.0/N*(np.sum(current_data)\
        -n*mu_old)/sigma_old**2
        sigma_hat += lr*1.0/N*(-n*sigma_old**2 \
        + np.sum((current_data - mu_old)**2))/sigma_old**3
    count+=1
print (mu_hat, sigma_hat)
```

(d) We know that $\mu \sim Normal(\mu_0, \sigma_0)$ and $X_i|\mu \sim Normal(\mu, \sigma)$. Assume X_is are i.i.d. By Bayes Rule, we have

$$
\begin{aligned}
f(\mu|\mathbb{X}) &\propto f(\mathbb{X}|\mu)f(\mu) = f(\mu)\prod_{i=1}^{n} f(x_i|\mu) \\
&= \frac{1}{\sqrt{2\pi\sigma_0^2}}\ \exp\{-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\}\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi\sigma^2}}\ \exp\{-\frac{(x_i-\mu)^2}{2\sigma^2}\} \\
&= \frac{1}{\sqrt{(2\pi)^{n+1}}\sigma_0\sigma^n}\exp\{-\frac{\mu^2-2\mu\mu_0+\mu_0^2}{2\sigma_0^2}-\sum_{i=1}^{n}\frac{x_i^2-2\mu x_i+\mu^2}{2\sigma^2}\} \\
&= const\times\exp\{-\frac{\mu^2(\sigma^2+n\sigma_0^2)-2\mu(\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2)+\mu_0^2\sigma^2+\sum_{i=1}^n\sigma_0^2x_i^2}{2\sigma_0^2\sigma^2}\} \\
&\propto \exp\{-\frac{\mu^2-2\mu\frac{\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2}{\sigma^2+n\sigma_0^2}+(\frac{\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2}{\sigma^2+n\sigma_0^2})^2}{2\frac{\sigma_0^2\sigma^2}{\sigma^2+n\sigma_0^2}}\}\exp\{\frac{(\frac{\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2}{\sigma^2+n\sigma_0^2})^2}{2\frac{\sigma_0^2\sigma^2}{\sigma^2+n\sigma_0^2}}\} \\
&\propto \exp\{-\frac{(\mu-\frac{\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2}{\sigma^2+n\sigma_0^2})^2}{2\frac{\sigma_0^2\sigma^2}{\sigma^2+n\sigma_0^2}}\}
\end{aligned}
$$

Therefore, $\mu|\mathbb{X} \sim Normal(\mu_1, \sigma_1)$, where

$$
\begin{aligned}
\sigma_1 &= \frac{\sigma_0^2\sigma^2}{\sigma^2+n\sigma_0^2} = (\frac{1}{\sigma_0^2}+\frac{n}{\sigma^2})^{-1} \\
\mu_1 &= \frac{\mu_0\sigma^2+\sum_{i=1}^n x_i\sigma_0^2}{\sigma^2+n\sigma_0^2} = (\frac{\mu_0}{\sigma_0^2}+\frac{\sum_{i=1}^n x_i}{\sigma^2})/(\frac{1}{\sigma_0^2}+\frac{n}{\sigma^2})
\end{aligned}
$$

Q3 (2 pts)

(a) Let WNV be the event that a particular person is infected by the virus. We can calculate the prior probabilities using the data given in the problem.

$$P(WNV) = 2000/(300 * 10^6)$$
$$= (2/3) * 10^{-5}$$
$$P(\overline{WNV}) = 1 - (2/3) * 10^{-5}$$

The test can return positive results and negative results with certain probabilities.

$$P(Positive|WNV) = 0.95$$
$$P(Positive|\overline{WNV}) = 10^{-4}$$

By applying the total probability theorem we can calculate the probability of a positive result by

$$P(Positive) = P(Positive|WNV) * P(WNV) +$$
$$P(Positive|\overline{WNV}) * P(\overline{WNV}) = 0.00106$$

(i) We need to calculate the probability that Sheldon has WNV given the test returned positive result.
This can be calculated by using bayes theorem.

$$P(WNV|Positive) = P(WNV \cap Positive)/P(Positive)$$
$$= P(Positive|WNV) * P(WNV)/P(Positive) = 0.0595$$

(ii) Now we need to calculate the probability that Sheldon will die this year given different fatal rates.

$$P(Death|WNV) = 0.04 + 0.002 = 0.042$$
$$P(Death|\overline{WNV}) = 0.002$$

By applying the total probability theorem we can calculate the probability of Sheldon's death by

P(Death)=P(Death—Sheldon has WNV)*P(Sheldon has WNV)+
P(Death—Sheldon does not have WNV)*P(Sheldon does not have WNV)
=0.042 * 0.0595 + 0.002 * (1-0.0595)=0.00438

(b) Let A be the event that Alice wins and the event R_i when she rolls i.
Then various scenarios in which A happens is-
The first entry in the tuple belongs to Alice and second entry belongs to Bob
i=1 Not possible
i=2 (2,1)
i=3 (3,1),(3,2)
i=4 (4,1),(4,2),(4,3)
i=5 (5,1),(5,2),(5,3) ,(5,4)
i=6 (6,1),(6,2),(6,3), (6,4), (6,5)

Hence the total number of ways in which event A happens is 1+2+3+4+5=15

P(R_i=3 | A)= P($R_i = 3 \cap A$)/P(A)=2/15

Hence, If Alice just won, the probability that she just rolled a 3 is 2/15

Q4 (3 pts)

Code attached separately

```
import numpy as np
def main():
    # Part a
    map = {'winter' : -1 , 'summer' : 1, 'fall' : 0, 'spring' : 0}
    data = np.genfromtxt('matrix.csv', delimiter = ",", \
        converters = {2 : lambda x : map[x.strip()]}, \
        dtype = int)
    max = np.amax(data, axis = 0)
    result = np.zeros((max[0]+1, max[1] + 1), dtype=int)
    for row in data :
        result[row[0]][row[1]] = row[2]
    print(result)

    # Part b
    b = result[1:3, 9:11]
    print(b)

    # Part c
    u = np.array([[3,4,5]])
    v = np.array([[2,2,-1]])
    c = np.inner(u,v)
    print(u)
    print(v)
    print(c)

main()
```

Q5 (2 pts)

$$A = U\Sigma V^\top$$

Since U and V are unitary,

$$U^\top U = I$$

$$V^\top V = I$$
$$C = AA^\top = U\Sigma V^\top (U\Sigma V^\top)^\top = U\Sigma V^\top V\Sigma^\top U^\top = U\Sigma I\Sigma^\top U^\top = U\Sigma\Sigma^\top U^\top$$

$$B = A^\top A = (U\Sigma V^\top)^\top U\Sigma V^\top = V\Sigma^\top U^\top U\Sigma V^\top = V\Sigma^\top I\Sigma V^\top = V\Sigma^\top \Sigma V^\top$$