Jilin Zheng // UID 205258796

EE3581 Homework #9

Problem 1.

(a) $\mathbb{E}[T] = \sum_{i=1}^{n} \mathbb{E}[T_i] = \sum_{i=1}^{100} \mathbb{E}[T_i] = \sum_{i=1}^{100} 3 = 100 \cdot 3 = 300$

T_i is an uniformly distributed random var. $\rightarrow f_T(t) = \begin{cases} \frac{1}{5-1}, & 1 \le t \le 5 \\ 0, & \text{otherwise} \end{cases} = \begin{cases} 1/4, & 1 \le t \le 5 \\ 0, & \text{otherwise} \end{cases}$

$\mathbb{E}[T_i] = \frac{1+5}{2} = 3$

(b) $\text{Var}[T] = n\,\text{Var}[T_i] = 100 \cdot \frac{4}{3} = \frac{400}{3}$

$\text{Var}[T_i] = \frac{(5-1)^2}{12} = \frac{16}{12} = \frac{4}{3}$

(c) $\mathbb{P}[T > 330] = 1 - \mathbb{P}[T \le 330] = 1 - \Phi\left(\frac{330-300}{\sqrt{\frac{400}{3}}}\right) = 1 - \Phi(2.598) \approx 0.0046874$

$T \rightarrow \text{Gaussian}\left(100 \cdot 3, 100 \cdot \frac{4}{3}\right) = \text{Gaussian}\left(300, \frac{400}{3}\right)$

Problem 2.

(a) let Y_i be Discrete Uniform $(1,3) \rightarrow \mathbb{E}[Y_i] = \frac{1+3}{2} = 2$, $\text{Var}[Y_i] = \frac{(3-1+1)^2-1}{12} = \frac{8}{12} = \frac{2}{3}$

$\lim_{n\to\infty} F_{Y_n}(y) = \Phi(y)$ let $Y_n = \frac{\sum_{i=1}^{36}(Y_i - 2)}{\sigma\sqrt{n}} \rightarrow \text{Gaussian}\left(n\mathbb{E}[Y_i], n\,\text{Var}[Y_i]\right) = \text{Gaussian}\left(36 \cdot 2, 36 \cdot \frac{2}{3}\right) = \text{Gaussian}(72, 24)$

$\mathbb{P}[Y \le 60] = \Phi\left(\frac{60-72}{\sqrt{24}}\right) \approx 0.0071529$

(b) $\lim_{n\to\infty} \mathbb{P}[Y_n = 2n] = 1$

Yes because as $\lim_{n\to\infty}$, Y_n becomes $\text{Gaussian}\left(n\mathbb{E}[Y_i], n\,\text{Var}[Y_i]\right)$, where $\mathbb{E}[Y_n] = n\mathbb{E}[Y_i] = 2n$

Problem 3.

(a) $\hat{X}_{LLSE} = \mathbb{E}[X] + \frac{\text{Cov}[X,Y]}{\text{Var}[Y]}\left(y - \mathbb{E}[Y]\right) = 2 + \frac{1/3}{4/3}(y-3) = 2 + \frac{1}{4}(y-3) = \frac{1}{4}y + \frac{5}{4}$

$\mathbb{E}[X] = \frac{1+3}{2} = 2$ $\quad \text{Var}[X] = \frac{(3-1)^2}{12} = \frac{1}{3}$

$\mathbb{E}[Y] = \mathbb{E}[X+V] = \mathbb{E}[X] + \mathbb{E}[V] = 2 + 1 = 3$

$\text{Var}[Y] = \text{Var}[X+V] = \text{Var}[X] + \text{Var}[V] + 2\text{Cov}[X,V] = \frac{1}{3} + 1 = \frac{4}{3}$

$\text{Cov}[X,V] = \emptyset$ since X and V are independent

$\text{Cov}[X,Y] = \text{Cov}[X, X+V] = \text{Var}[X] + \emptyset + (1+\emptyset)\,\text{Cov}[X,V] = \text{Var}[X] = \frac{1}{3}$

$MSE_{LLSE} = \text{Var}[X] - \frac{(\text{Cov}[X,Y])^2}{\text{Var}[Y]} = \frac{1}{3} - \frac{(\frac{1}{3})^2}{4/3} = \frac{1}{3} - \frac{1}{3}\left(\frac{3}{4}\right) = \frac{1}{3} - \frac{1}{12} = \frac{4}{12} - \frac{1}{12} = \frac{1}{4}$

(b) $\mathbb{E}[Y] = \emptyset$ $\quad \text{Var}[Y] = \frac{\text{Var}[X]}{400} = \frac{\left(\frac{(5-(-5))^2}{12}\right)}{400} = 0.0208$

(c) $\mathbb{P}[Y \le 1.9] \approx \Phi\left(\frac{1.9-2}{2}\right) = \Phi(-0.05) \approx 0.480$

$\mathbb{E}[\text{Exponential}(\frac{1}{2})] = \frac{1}{1/2} = 2$ $\quad \text{Var}[\text{Exponential}(\frac{1}{2})] = \left(\frac{1}{1/2}\right)^2 = \frac{1}{1/4} = 4$

Problem 4.

(a) $\mathbb{E}[S_{300}] = 300\,\mathbb{E}[X] = 300 \cdot \left(2 \cdot \frac{1}{2} + 4 \cdot \frac{1}{2}\right) = 300(1+2) = 900$

$\mathbb{E}[X^2] = 2^2 \cdot \frac{1}{2} + 4^2 \cdot \frac{1}{2} = 2 + 8 = 10$ $\quad$ $\mathrm{Var}[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 = 10 - 9 = 1$

$\mathrm{Var}[S_{300}] = 300\,\mathrm{Var}[X] = 300(1) = 300$

(b) $\mathbb{P}\left[|S_{300} - \mathbb{E}[S_{300}]| \geq 40\right] = 1 - \mathbb{P}\left[|S_{300} - \mathbb{E}[S_{300}]| < 40\right]$

$= 1 - \mathbb{P}\left[-40 < S_{300} - \mathbb{E}[S_{300}] < 40\right]$

$= 1 - \mathbb{P}\left[860 < S_{300} < 940\right]$

$= 1 - \left[\Phi\left(\frac{940-900}{\sqrt{300}}\right) - \Phi\left(\frac{860-900}{\sqrt{300}}\right)\right]$

$= 1 - \left[\Phi\left(\frac{4}{\sqrt{3}}\right) - \Phi\left(\frac{-4}{\sqrt{3}}\right)\right]$

(c) $\mathbb{P}\left[\mu - \epsilon \leq M_n \leq \mu + \epsilon\right] = 1 - \alpha = 0.95 \rightarrow \alpha = 0.05$

$\epsilon = \frac{\sigma Q^{-1}\left(\frac{\alpha}{2}\right)}{\sqrt{n}} = \frac{1 \cdot Q^{-1}(0.025)}{\sqrt{300}} = \frac{1.96}{\sqrt{300}} = 0.113$

$\left[M_{300} \pm 0.113\right]$

(d) $V_{300} = 1.21$

$\epsilon = \frac{-\sqrt{V_{300}}\, F^{-1}_{T_{299}}(0.025)}{\sqrt{300}} = \frac{-\sqrt{1.21}\,(-1.9679)}{\sqrt{300}} = 0.125$

$\left[M_{300} \pm 0.125\right]$

Problem 5.

(a) X is Gaussian$(5.02\ \mu m, 0.25\ \mu m^2)$

$\mathrm{Var}[M_{100}] = \frac{\mathrm{Var}[X]}{n} = \frac{0.25}{100} = 0.0025\ \mu m^2$

(b) $M_{100} = 5.10\ \mu m$, $\alpha = 0.05$

$\epsilon = \frac{\sigma Q^{-1}\left(\frac{\alpha}{2}\right)}{\sqrt{n}} = \frac{0.50 \cdot Q^{-1}(0.025)}{\sqrt{100}} = \frac{0.5 \cdot 1.96}{10} = 0.098$

$\left[5.1 \pm 0.098\right] = \left[5.002, 5.198\right]$

No the sample is NOT significantly different @ a significance level of 0.05 because the baseline is within the confidence interval.

(c) $1 - \alpha = 0.9 \rightarrow \alpha = 0.1$

$\epsilon = \frac{\sigma Q^{-1}\left(\frac{\alpha}{2}\right)}{\sqrt{100}} = \frac{0.5\, Q^{-1}(0.05)}{10} = 0.082$

$\left[5.10\ \mu m \pm 0.082\right]$

In [79]:
```
# Load needed modules
import numpy as np
import matplotlib.pyplot as plt
import math
```

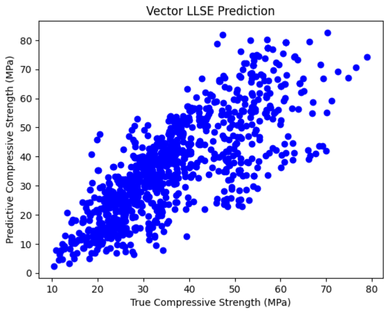
In [80]:
```
#  LLSE(X,Y) produces the LLSE estimate Xhat of X from Y as well as the mean-squared error MSE
#  and the coefficient of determination R2.

#9.6 (a) Complete the function LLSE to compute the LLSE estimate of X given data samples in the mat

def LLSE(X, Y):
    XYstack = np.column_stack((X, Y)) #concatenate X and Y into a matrix with X as its first column

    # Fill in the following:
    #   Compute the average of XYstack,
    #   set the first element of XYstack as mean of X, muX
    #   set the remaining elements of XY stack as the mean vector of Y, muY
    #   Compute the covariance matrix of XYstack.  This covariance will have the shape
    #   [SigmaX   SigmaXY] in the first row, where SigmaX is the scalar variance of X and SigmaXY
    #   is the cross covariance between X and Y.
    #   The rest of rows of the covariance matrix have SigmaYX as the first column, and SigmaY as t
    #   Store the appropriate elements from the computed covariance into SigmaX, SigmaXY, and Sigma
    meanstack = np.mean(XYstack, axis=0)    #mean of X and Y
    covstack = np.cov(XYstack, rowvar=False)     #covariance of X and Y
    muX = meanstack[0]          #extract mean of X from meanstack
    muY = meanstack[1:]          #extract mean of Y from meanstack
    sigmaXY = covstack[0, 1:]      #extract cross-covariance matrix of X and Y
    sigmaY = covstack[1:, 1:]      #extract covariance matrix of Y
    sigmaX = covstack[0, 0]      #extract variance of X

    # Using the above vectors and matrices, for each row j of Y, use the LLSE estimation formula fo
    # Xhat[j] using the LLSE estimation formulas with the vectors and matrices indicated above.
    ndata = Y.shape[0]
    xhat = np.zeros((ndata,))
    sigmaYinv = np.linalg.inv(sigmaY) # always do the inverses outside the loops
    yvector = Y[0,:]
    for j in range(ndata):
        yvector = Y[j,:]
        xhat[j] = muX + (sigmaXY@sigmaYinv)@(yvector - muY)    ### use the LLSE estimate to compute

    # Compute the theoretical LLSE mean squared error using the LLSE error formula:
    sigmaXYtransposed = covstack[1:, 0] #sigmaXY.transpose()
    sigmaE = sigmaX - (sigmaXY@np.linalg.inv(sigmaY)@sigmaXYtransposed)      ## Fill this

    return xhat, sigmaE
```

In [81]:
```
# 9.6 (b)  Compute the empirical performance of the LLSE Estimator from the data and estimates

def ComputePerformance(X, Xhat):
    ndata = X.shape[0]
    muX = np.mean(X)

    #Your code for calculating the mean-squared error between X and Xhat
    sum = 0.
    for ii in range(ndata):
        sum += ((X[ii]-Xhat[ii]) ** 2.)
    MSE = (1./ndata) * sum

    #Your code for calculating the coefficient of determination between X and Xhat.
    sum = 0.
    for ii in range(ndata):
        sum += ((X[ii]-muX) ** 2.)
    denom = (1./ndata) * sum
    R2 = 1. - (MSE/denom)

    return MSE, R2
```

In [82]:
```
#Homework 9 Python Solution: Read the data

data = np.genfromtxt("concretedata.csv", delimiter = ",")
X = data[1:,8]
nx = X.shape[0]
Y = data[1:,0:8]
ny,dy = Y.shape
```

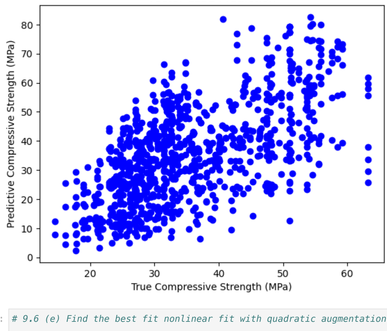
In [83]:
```
# 9.6 (c) Find the best vector fit

vectorXhat, vectorSigmaE = LLSE(X, Y)
vectorMSE, vectorR2 = ComputePerformance(X, vectorXhat)

print(f"Vector LLSE prediction attains theoretical MSE = {vectorSigmaE}, MSE = {vectorMSE}, and R2

## Scatter plot the required data and label the plot and the two axes below...
fig = plt.figure()
plt.scatter(vectorXhat, X, color='b')
plt.xlabel('True Compressive Strength (MPa)')
plt.ylabel('Predictive Compressive Strength (MPa)')
plt.title('Vector LLSE Prediction')
plt.show()
fig.savefig('part_c.png')
```

```
Vector LLSE prediction attains theoretical MSE = 107.30141220321266, MSE = 107.19723607486019, and
R2 = 0.6155198704142721.
```



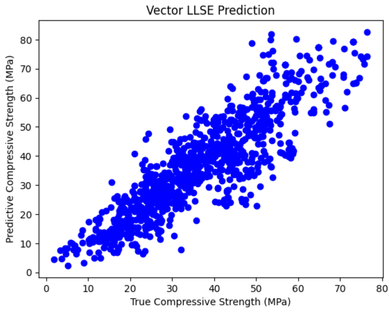
In [84]:
```
# 9.6 (d) Find the best fit with only the first 3 columns of Y

vectorXhat, vectorSigmaE = LLSE(X, Y[:,0:3])
vectorMSE, vectorR2 = ComputePerformance(X, vectorXhat)

print(f"Vector LLSE prediction attains theoretical MSE = {vectorSigmaE}, MSE = {vectorMSE}, and R2

fig = plt.figure()
plt.scatter(vectorXhat, X, color='b')
plt.xlabel('True Compressive Strength (MPa)')
plt.ylabel('Predictive Compressive Strength (MPa)')
plt.title('Vector LLSE Prediction')
plt.show()
fig.savefig('part_d.png')
```

```
Vector LLSE prediction attains theoretical MSE = 166.57286880801084, MSE = 166.41114757615838, and
R2 = 0.40313965240754923.
```



In [85]:
```
# 9.6 (e) Find the best fit nonlinear fit with quadratic augmentation of data

Yaug = np.column_stack((Y, Y**2))
vectorXhat, vectorSigmaE = LLSE(X, Yaug)
vectorMSE, vectorR2 = ComputePerformance(X, vectorXhat)

print(f"Vector LLSE prediction attains theoretical MSE = {vectorSigmaE}, MSE = {vectorMSE}, and R2

fig = plt.figure()
plt.scatter(vectorXhat, X, color='b')
plt.xlabel('True Compressive Strength (MPa)')
plt.ylabel('Predictive Compressive Strength (MPa)')
plt.title('Vector LLSE Prediction')
plt.show()
fig.savefig('part_e.png')

print("This is definitely a better fit to the data than my linear fit from part d as the theoretica
```

Vector LLSE prediction attains theoretical MSE = 63.00867330017962, MSE = 62.94749983094031, and R2 = 0.7742286669108126.



This is definitely a better fit to the data than my linear fit from part d as the theoretical and empirical MSE are both lower, and the R^2 is higher than before. Just visually, I can also see that it is closer to a line than before.