INTERNATIONAL
UNIVERSITY OF
APPLIED SCIENCES

Advance Statistics Course

University of Applied Science - Online

Masters in Data Science (MSDS60ECTS)

Workbook Advanced Statistics(DLMDSAS01)

Gummadi Sai Dheeraj

Matriculation: 3210935

gummadi.saidheeraj@iu-study.org

Advisor: Paul Libbrecht

Realized with input of the Parameter Generator: 96d7a70e97413e93c06cc8c1976834f5e9163ece

Delivery date: July 19, 2024

Contents

I List of Figures **1**

1 Workbook Solutions **2**

1.1 Task 1: Basic Probabilities and Visualizations (1) 2

1.2 Task 2: Basic Probabilities and Visualizations (2) 4

1.3 Task 3: Transformed Random Variables 6

1.4 Task 4: Hypothesis Test 8

1.5 Task 5: Regularized Regression 10

1.6 Task 6: Bayesian Estimates 13

I List of Figures

1 Negative Binomial Distribution of Meteoritess falling till p=0.5% with median and expectation value . . . 3
2 Probability density function to hear owl in hours and Histogram of probability of the hearing the owl at any particular minute . . . 5
3 Density function of T for $\theta = 22.4$ . . . 7
4 T-test with sample 929, 912, 967, 899, 958, 898, 946, 899, 892, 841 . . . 9
5 Visualizing original data points . . . 11
6 Visualizing original data points . . . 11
7 Calculating OLS Estimates and Ridge Estimates . . . 12
8 Visualizing OLS Estimates . . . 12
9 Visualizing Ridge Estimates . . . 13
10 Posterior Distribution . . . 14

1 Workbook Solutions

1.1 Task 1: Basic Probabilities and Visualizations (1)

Problem Statement: Please provide the requested visualization as well as the numerical results. In both cases, please either prove or cite any computation of the proof steps (calculations, code, steps, etc.), and justify why you trust the tools you used. Do not forget to include the scale of each graphics so that the reader can interpret the numbers represented.

The number of meteorites falling into an ocean in a given year can be modeled by one of the following distributions. Provide a graphic showing the probability of one, two, three, etc., meteorites falling (until the probability remains less than 0.5% for any higher number of meteorites; you should also prove it). Calculate the expectation and median and present them in this graphic: If ξ_1 is 2: a negative binomial distribution with an expectation of k = ξ_2 and p = ξ_3

Generated Parameters: $\xi_1 = 2$, $\xi_2 = 30$, $\xi_3 = 0.89$

Solution:

Given k = 30, p = 0.89

The negative binomial distribution with parameters k and p can be described using probability mass function (PMF) as follows:

$$P(X = x) = (\ x\ ) + k - 1 \qquad\qquad x(1 - p)^x p^k$$

where

k is number of success, p is probability of success, x is number of failures.

The Expectation value (mean) of the negative binomial distribution is calculated as follows:

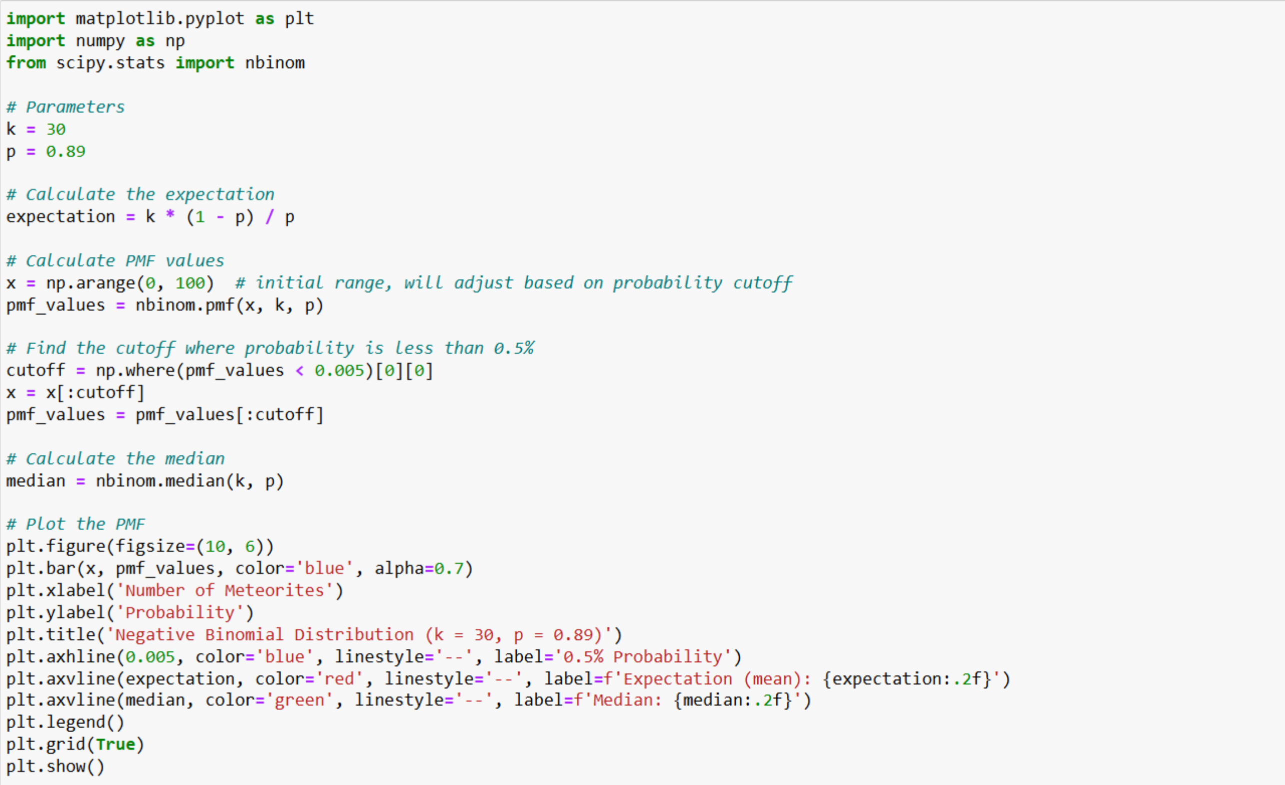
Substituting the given parameters in the above formula:

$$E(X) = \frac{30(1-0.89)}{0.89}$$

$$E(X) = 3.707$$

The code below show the graphical representation of the median and expectation values using PMF negative binomial distribution of meteorites falling.

From Figure.1, the median value of the negative binomial distribution is 3.



```python
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import nbinom

# Parameters
k = 30
p = 0.89

# Calculate the expectation
expectation = k * (1 - p) / p

# Calculate PMF values
x = np.arange(0, 100)  # initial range, will adjust based on probability cutoff
pmf_values = nbinom.pmf(x, k, p)

# Find the cutoff where probability is less than 0.5%
cutoff = np.where(pmf_values < 0.005)[0][0]
x = x[:cutoff]
pmf_values = pmf_values[:cutoff]

# Calculate the median
median = nbinom.median(k, p)

# Plot the PMF
plt.figure(figsize=(10, 6))
plt.bar(x, pmf_values, color='blue', alpha=0.7)
plt.xlabel('Number of Meteorites')
plt.ylabel('Probability')
plt.title('Negative Binomial Distribution (k = 30, p = 0.89)')
plt.axhline(0.005, color='blue', linestyle='--', label='0.5% Probability')
plt.axvline(expectation, color='red', linestyle='--', label=f'Expectation (mean): {expectation:.2f}')
plt.axvline(median, color='green', linestyle='--', label=f'Median: {median:.2f}')
plt.legend()
plt.grid(True)
plt.show()
```

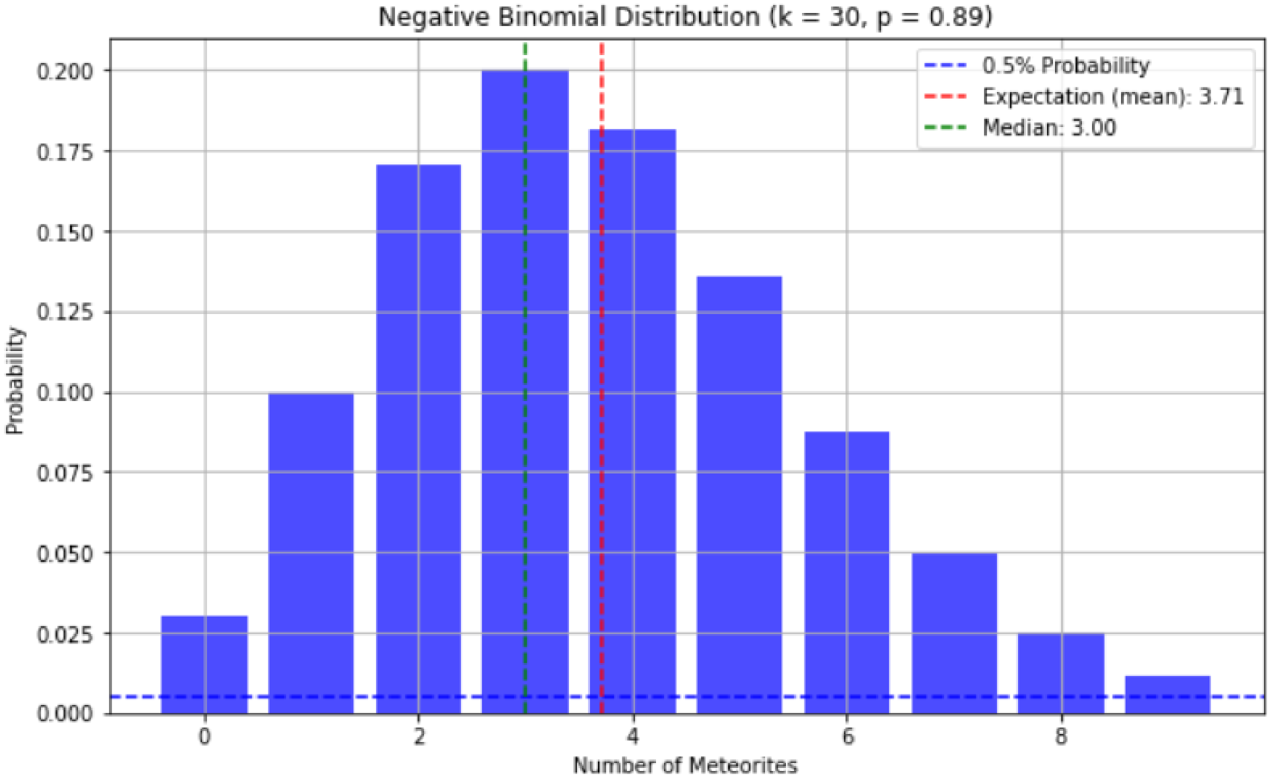


Figure 1: Negative Binomial Distribution of Meteoritess falling till p=0.5% with median and expectation value

1.2 Task 2: Basic Probabilities and Visualizations (2)

Problem Statement: Please provide the requested visualization as well as the numerical results. In both cases, please either prove or cite any computation of the proof steps (calculations, code, steps, etc.), and justify why you trust the tools you used. Do not forget to include the scale of each graphics so that the reader can interpret the numbers represented.

Let Y be the random variable with the time it takes to hear an owl from your room's open window (in hours). Assume that the probability that you still need to wait to hear the owl after y hours is one of the following functions:

Generated Parameters: $\xi_4 = 3, \xi_5 = 0.51, \xi_6 = 6, \xi_7 = 0.48, \xi_8 = 9$

Solution:

Given the probability is given by $0.51e^{-6y^2} + 0.48e^{-9y^2}$

f(y) = $0.51e^{-6y^2} + 0.48e^{-9y^2}$

The probability that you need to wait between two and four hours to hear the owl is given by the probability density function (PDF) of Y within the interval of 2 and 4.

$$P(2 \leq Y \leq 4) = \int_2^4 f(y)\,dy$$
$$P(2 \leq Y \leq 4) = 0.51e^{-6y^2} + 0.48e^{-9y^2}$$

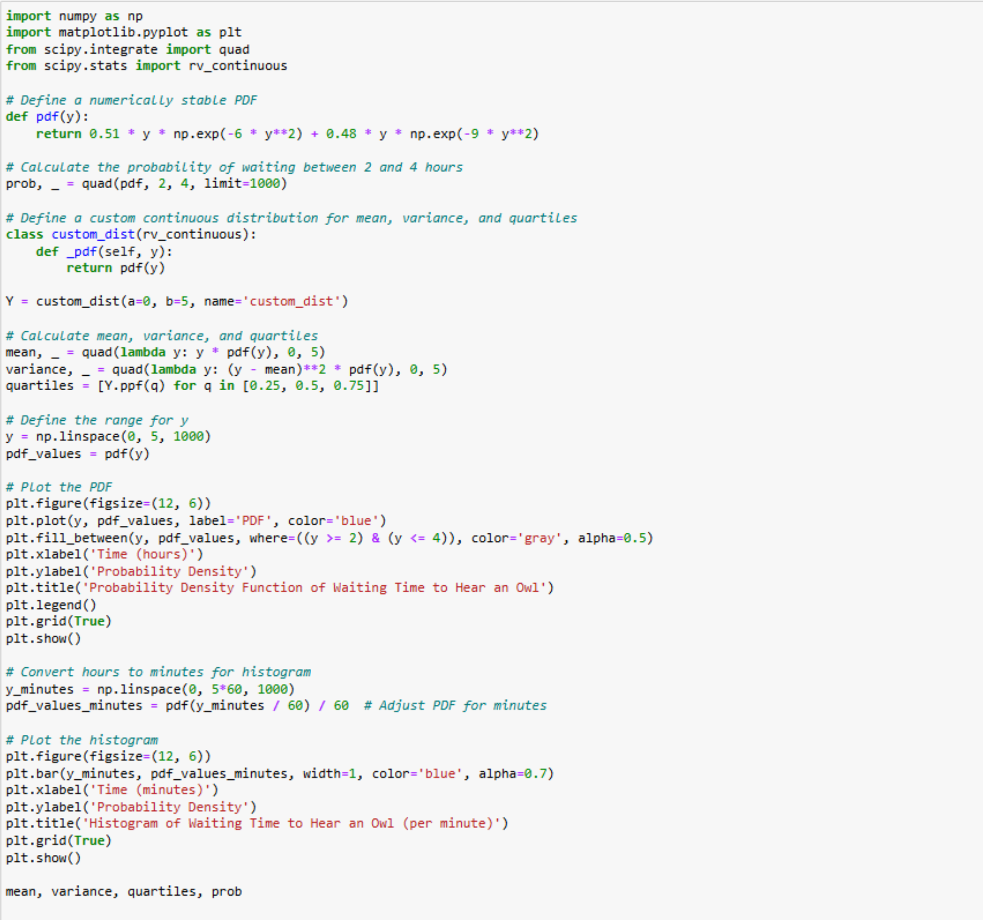
The code in Fig. 2 will help in visualizing PDF graphically and Histogram of probability of the hearing the owl at any particular minute.

The Expectation E[Y] can be calculated as

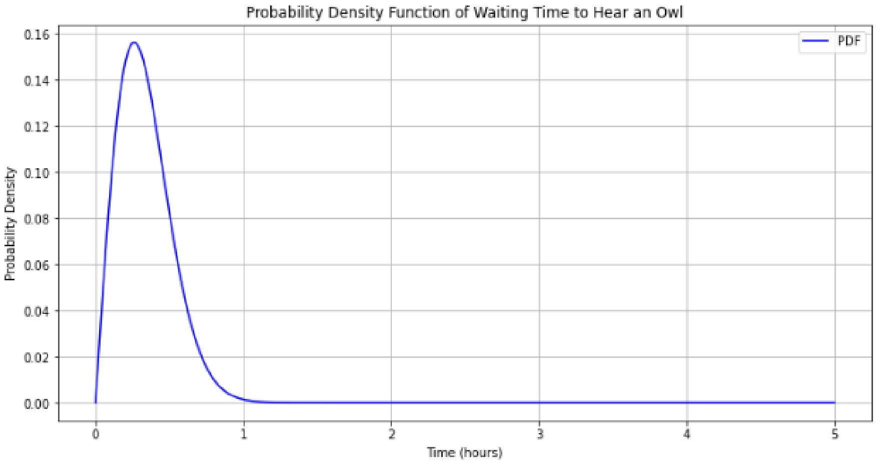
$$E[Y] = \int_0^\infty y(f(y))dy$$
$$E[Y] = \int_0^\infty 0.51e^{-6y^2}dy + \int_0^\infty 0.48e^{-9y^2}dy$$
$$E[Y] = 0.0232$$

The Variance value is given by:

$$Var[Y] = E[Y^2] - (E[Y])^2$$
$$Var[Y] = 0.009$$



```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import quad
from scipy.stats import rv_continuous

# Define a numerically stable PDF
def pdf(y):
    return 0.51 * y * np.exp(-6 * y**2) + 0.48 * y * np.exp(-9 * y**2)

# Calculate the probability of waiting between 2 and 4 hours
prob, _ = quad(pdf, 2, 4, limit=1000)

# Define a custom continuous distribution for mean, variance, and quartiles
class custom_dist(rv_continuous):
    def _pdf(self, y):
        return pdf(y)

Y = custom_dist(a=0, b=5, name='custom_dist')

# Calculate mean, variance, and quartiles
mean, _ = quad(lambda y: y * pdf(y), 0, 5)
variance, _ = quad(lambda y: (y - mean)**2 * pdf(y), 0, 5)
quartiles = [Y.ppf(q) for q in [0.25, 0.5, 0.75]]

# Define the range for y
y = np.linspace(0, 5, 1000)
pdf_values = pdf(y)

# Plot the PDF
plt.figure(figsize=(12, 6))
plt.plot(y, pdf_values, label='PDF', color='blue')
plt.fill_between(y, pdf_values, where=((y >= 2) & (y <= 4)), color='gray', alpha=0.5)
plt.xlabel('Time (hours)')
plt.ylabel('Probability Density')
plt.title('Probability Density Function of Waiting Time to Hear an Owl')
plt.legend()
plt.grid(True)
plt.show()

# Convert hours to minutes for histogram
y_minutes = np.linspace(0, 5*60, 1000)
pdf_values_minutes = pdf(y_minutes / 60) / 60  # Adjust PDF for minutes

# Plot the histogram
plt.figure(figsize=(12, 6))
plt.bar(y_minutes, pdf_values_minutes, width=1, color='blue', alpha=0.7)
plt.xlabel('Time (minutes)')
plt.ylabel('Probability Density')
plt.title('Histogram of Waiting Time to Hear an Owl (per minute)')
plt.grid(True)
plt.show()

mean, variance, quartiles, prob
```



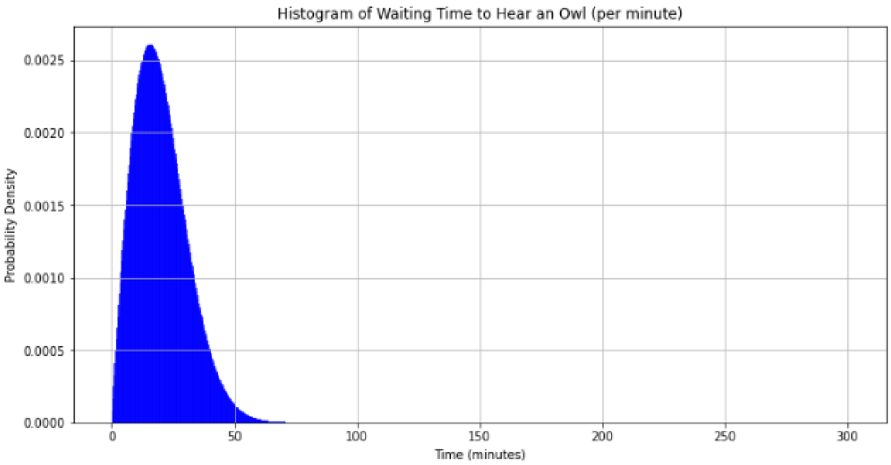


Figure 2: Probability density function to hear owl in hours and Histogram of probability of the hearing the owl at any particular minute

1.3 Task 3: Transformed Random Variables

Probleem Statement: A type of network router has a bandwidth total to first hardware failure called S expressed in terabytes. The random variable S is modeled by an exponential distribution whose density is given by one of the following functions: if $\xi_9 = 1 : f(s) = \frac{1}{24\theta^5} s^4 e^{-\frac{s}{\theta}}$ with a single parameter θ. Consider the bandwidth total to failure T of the sequence of the two routers of the same type (one being brought up automatically when the first is broken). Express T in terms of the bandwidth total to failure of single routers S_1 and S_2. Formulate realistic assumptions about these random variables. Calculate the density function of the variable T. Given an experiment with the dual-router-system yielding a sample $T_1, T_2, \ldots, T_n$, calculate the likelihood function for θ. Propose a transformation of this likelihood function whose maximum is the same and can be computed easily. An actual experiment is performed, and the infrastructure team has obtained the bandwidth totals to failure given by the sequence $\xi_1 0$ of numbers. Estimate the model-parameter with the maximum likelihood and compute the expectation of the bandwidth total to failure of the dual-router-system.
Generated Paramenters: $\xi_9 = 1, \xi_{10} = 98, 4, 38, 43, 41$
Solution:
Given that S_1 and S_2 are independent and identically distributed (i.i.d.) random variables representing the bandwidth total to failure of each router, the total bandwidth T to failure of the dual-router system (where the second router is used after the first one fails) can be expressed as:

$$\mathrm{T} = S_1 + S_2$$

Assuming S_1 and S_2 follow the given exponential distribution with the density function:

$$f(s) = \frac{1}{24\theta^5} s^4 e^{-\frac{s}{\theta}}$$

based on the standard exponential function:

$$f(x) = \lambda e^{-\lambda x}$$

comparing $f(x)$ and $f(s)$: $\lambda = \frac{1}{\theta}$ Convolution can be used, as shown below, to find the density of the sum of these two random variables.

$$f_T(t) = \int_0^\infty s_1(t) s_2(t - \tau) d\tau$$

The density function of T is the convolution of the density functions of S_1 and S_2. Let $f_{S_1}(s) and f_{S_2}(s)$ be the density functions of S_1 and S_2:

$$f_T(t) = \int_0^t f_{s_1}(s) f_{s_2}(t - s) ds$$
$$f_T(t) = (\frac{1}{24\theta^5})^2 * t^8 * e^{-\frac{2t}{\theta}}$$

The gamma distribution is utilized for various purposes in statistical analyses. Initially, it can be used to estimate the accuracy of a Gaussian distribution by considering prior information. Furthermore, it can also be utilized to approximate the rate parameter of a exponential distribution. This distribution has two parameters: the shape parameter and the scale parameter. Furthermore, T's density function is the moment-generating function of the Gamma(2,) distribution. The function is as follows: for t greater than 0, it equals S to the power of 4, times e to the power of -S, and divided by 24 to the power of 5. Yet, if t is not greater than 0, the function results in 0 instead.

Maximum Likelihood Estimation (MLE) can be used to ascertain the most likely distribution parameters when previous information about the dataset's distribution is known. Using MLE, the likelihood function L() is optimized in relation to the unknown parameter . Based on n observations T1, T2,..., Tn, the probability function of may be defined as follows:

$$L(\theta) = \Pi_{i=1}^n f_t(T_i)$$
$$L(\theta) = \frac{1}{\theta^2} \Pi_{i-1}^n t_i e^{\Sigma_{i=1}^n \frac{t_i}{\theta}}$$

The log likelihood function is defined as:

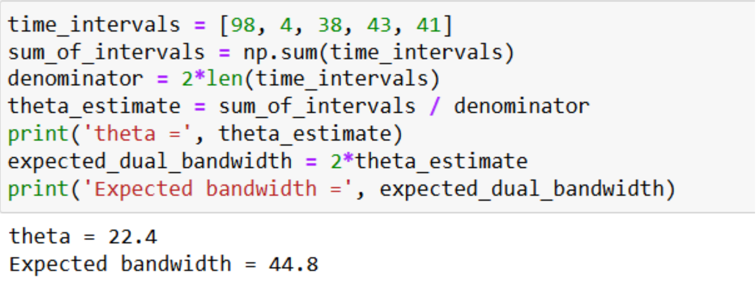
$$log(L(\theta)) = -2nlog(\theta) \Sigma_{n=1}^n log(t_i) - \Sigma_{n=i}^n \frac{t_i}{\theta}$$

The maximum likelihood is defined as:

$$\frac{\delta logL(\theta)}{\delta\theta} = \frac{-2n}{\theta} + \Sigma_{n=1}^{i} \frac{t_i}{\theta^2} = 0$$

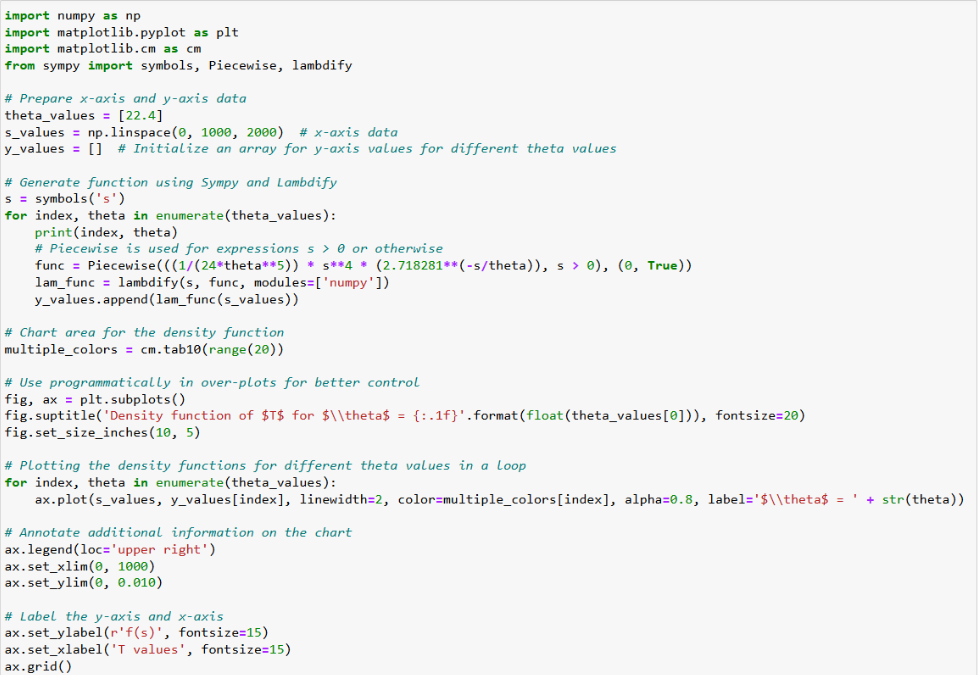
$$\hat{\theta} = \frac{\Sigma_{i=1}^{n} t_i}{2n}$$

Code in Fig: 3 helps in calculating the predicted bandwidth total until the dual router system fails, we must compute the highest likelihood estimate of the model parameter.



```
time_intervals = [98, 4, 38, 43, 41]
sum_of_intervals = np.sum(time_intervals)
denominator = 2*len(time_intervals)
theta_estimate = sum_of_intervals / denominator
print('theta =', theta_estimate)
expected_dual_bandwidth = 2*theta_estimate
print('Expected bandwidth =', expected_dual_bandwidth)
```

```
theta = 22.4
Expected bandwidth = 44.8
```



```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
from sympy import symbols, Piecewise, lambdify

# Prepare x-axis and y-axis data
theta_values = [22.4]
s_values = np.linspace(0, 1000, 2000)  # x-axis data
y_values = []  # Initialize an array for y-axis values for different theta values

# Generate function using Sympy and Lambdify
s = symbols('s')
for index, theta in enumerate(theta_values):
    print(index, theta)
    # Piecewise is used for expressions s > 0 or otherwise
    func = Piecewise(((1/(24*theta**5)) * s**4 * (2.718281**(-s/theta)), s > 0), (0, True))
    lam_func = lambdify(s, func, modules=['numpy'])
    y_values.append(lam_func(s_values))

# Chart area for the density function
multiple_colors = cm.tab10(range(20))

# Use programmatically in over-plots for better control
fig, ax = plt.subplots()
fig.suptitle('Density function of $T$ for $\\theta$ = {:.1f}'.format(float(theta_values[0])), fontsize=20)
fig.set_size_inches(10, 5)

# Plotting the density functions for different theta values in a loop
for index, theta in enumerate(theta_values):
    ax.plot(s_values, y_values[index], linewidth=2, color=multiple_colors[index], alpha=0.8, label='$\\theta$ = ' + str(theta))

# Annotate additional information on the chart
ax.legend(loc='upper right')
ax.set_xlim(0, 1000)
ax.set_ylim(0, 0.010)

# Label the y-axis and x-axis
ax.set_ylabel(r'f(s)', fontsize=15)
ax.set_xlabel('T values', fontsize=15)
ax.grid()
```

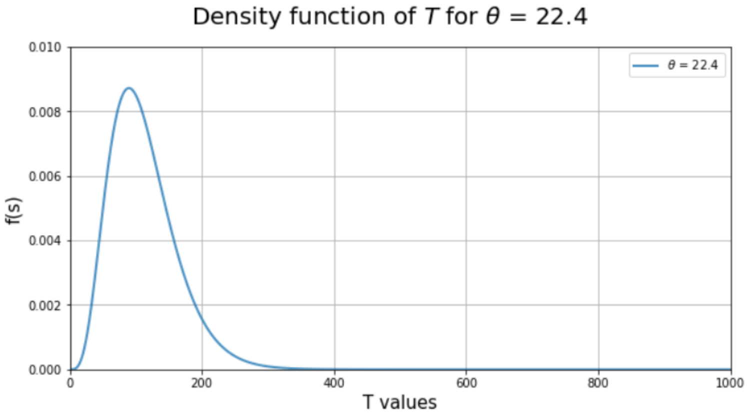


Figure 3: Density function of T for θ = 22.4

1.4 Task 4: Hypothesis Test

Probleem Statement: Over a long period of time, the production of 1,000 high-quality hammers in a factory seems to have reached a weight with an average of ξ_{11} (in g) and standard deviation of ξ_{12} (in g). Propose a model for the weight of the hammers including a probability distribution for the weight. Provide all the assumptions needed for this model to hold (even the uncertain ones). What parameters does this model have? (if ξ_{13} = 1): Does the new system make lower weights? To answer this question, a random sample of newly produced hammers is evaluated yielding the weights in ξ_{14}. What hypotheses can you propose to test the question? What test and decision rule can you make to estimate whether the new system answers the given question? Express the decision rules as logical statements involving critical values. What error probabilities can you suggest and calculate? Perform the test and draw the conclusion to answer the question. **Generated Paramenters:** $\xi_{11} = 912, \xi_{12} = 74.1, \xi_{13} = 1, \xi_{14} = 929, 912, 967, 899, 958, 898, 946, 899, 892, 841$

Solution:

Given Mean weight(μ) = 912 grams

Standard Deviation (σ) = 74.1 grams

Model Assumptions:

- **Normal Distribution:** We assume that the weights of the hammers follow a normal distribution.
- **Independent Observations:** We assume that the weights of individual hammers are independent of each other.
- **Constant Variance:** The variance of hammer weights is constant across the production period.

To determine whether the new system produces hammers with a different average weight, we can perform a hypothesis test.

Null Hypothesis (H_0): The mean weight of hammers produced by the new system is equal to the mean weight of hammers produced by the old system. (μ=912 grams)

Alternative Hypothesis (H_1): The mean weight of hammers produced by the new system is not equal to the mean weight of hammers produced by the old system. ($\mu \neq$ 912 grams)

One-sample t-test can be used here as it is used to compare an unknown population mean from a specific value.

$$t = \frac{\overline{x}-\mu}{\frac{s}{\sqrt{n}}}$$

where $\overline{x} = samplemean$

$s = samplestandarddeviation$

$n = samplesize$

$Giventhesampleweights : 929, 912, 967, 899, 958, 898, 946, 899, 892, 841.$

$Herewewillconductthet - test, andifthepvalueislessthanorequalsignificancelevel(\alpha)$ of 0.05, the null hypothesis is rejected else we will fail to reject the null hypothesis.

Sample Size = 10

Calculating Sample mean weight ($\overline{x}$):

$$\overline{x} = \frac{929+912+967+899+958+898+946+899+892+841}{10}$$

$$\overline{x} = 914.1$$

Calculating Sample Standard Deviation:

$$s = \sqrt{\frac{\Sigma_{i=1}^{1}0(x_i-\overline{h})^2}{10-1}}$$

$$s = 37.2$$

Performing T-Test:

$$H_0 : \mu_n ew = 912 \text{ and } H_1 : \mu_n ew \neq 912$$

$$z = \frac{\overline{x} - \mu}{\frac{s}{\sqrt{n}}}$$
$$t = \frac{914.1 - 912}{\frac{37.2}{\sqrt{10}}}$$
$$t = 0.178$$

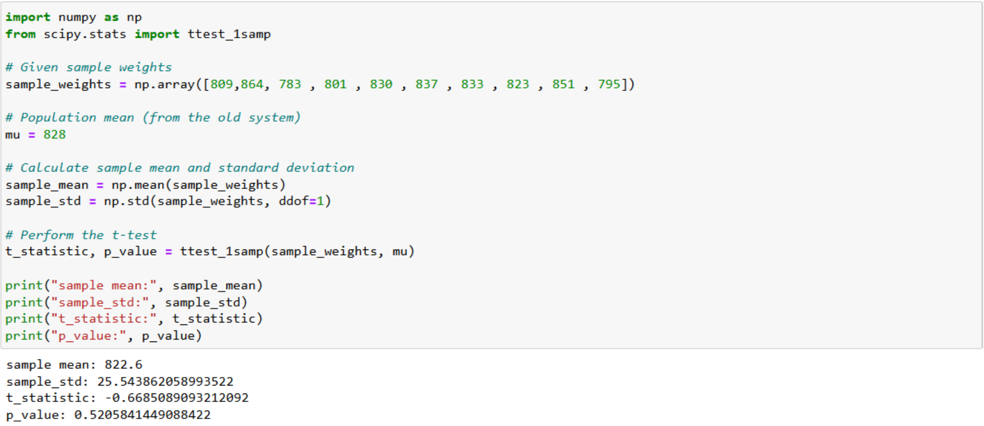
Calcualting P-Value:

degrees of freedom(df) = 10-1 = 9

CDF for t = 0.182 with df=9 gives approximately 0.571.

$$p - value = 2 \times (1 - CDF(t))$$
$$p - value = 2 \times (1 - 0.571)$$
$$p - value = 2 \times 0.429$$
$$p - value = 0.85$$

The following code will help in calculating the mean, standard deviation, t-test of given sample weights:



```
import numpy as np
from scipy.stats import ttest_1samp

# Given sample weights
sample_weights = np.array([809,864, 783 , 801 , 830 , 837 , 833 , 823 , 851 , 795])

# Population mean (from the old system)
mu = 828

# Calculate sample mean and standard deviation
sample_mean = np.mean(sample_weights)
sample_std = np.std(sample_weights, ddof=1)

# Perform the t-test
t_statistic, p_value = ttest_1samp(sample_weights, mu)

print("sample mean:", sample_mean)
print("sample_std:", sample_std)
print("t_statistic:", t_statistic)
print("p_value:", p_value)
```

```
sample mean: 822.6
sample_std: 25.543862058993522
t_statistic: -0.6685089093212092
p_value: 0.5205841449088422
```

Figure 4: T-test with sample 929, 912, 967, 899, 958, 898, 946, 899, 892, 841

The decision rule is either if p-value is greater than significance value (0.05), we fail to reject null hypothesis or if $|z|$ is less than equal to critical value ($z_{0.05/2}$), we fail to reject the null hypothesis. In the current scenario, both p-Value (0.85) is greater than 0.05 and z score (0.178) is less than $z_{0.05/2}$(1.96). So, we fail to reject null hypothesis. thus conclude that there is not enough evidence to say that the new system produces more constant weights.

1.5 Task 5: Regularized Regression

Problem Statement: Given the values of an unknown function f: $\mathbb{R} \rightarrow \mathbb{R}$ at some selected points, we try to calculate the parameters of a model function using OLS as a distance and a ridge regularization: (if $\xi_1 5 = 0$): a polynomial model function of twelve α_i parameters: $f(x) = \alpha_0 + \alpha_1 x + ... + \alpha_1 2x^1 2$.

Calculate the OLS estimate, and the OLS ridge-regularized estimates for the parameters given the sample points of the graph of f given that the values are (x, y) each of the elements of $\xi_1 6$. What weight do you give to the penalties? What are the qualities of each of the solutions? Remember to include the steps of your computation, which are more important than the actual computations. If you calculate the solution with a program, make sure that you trust and cite the core functions used and that you sketch the mathematical path in a way that is coherent with the program.

Generated Paramenters: $\xi_{15} = 0, \xi_{16} = (11, 32766575994387.79), (12, 87792488207053.36), (5, 2545633558.93), (-11, 302$

Solution:

from , minimizing sum of square residuals will give the OLS estimate.

$$\Sigma_{i=1}^{n}(y_i - f(x_i))^2$$

Ridge Regularized version of sum of squared residuals is denoted as follows:

$$\Sigma_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\Sigma_{j=0}^{12}\alpha_j^2$$

The OLS estimate of the parameters:

$$\hat{\alpha}_{OLS} = (X^T X)^{-1} X^T y)$$

where X will be an $n \times 13$ matrix where each row represents a data point and each column represents a power of x.

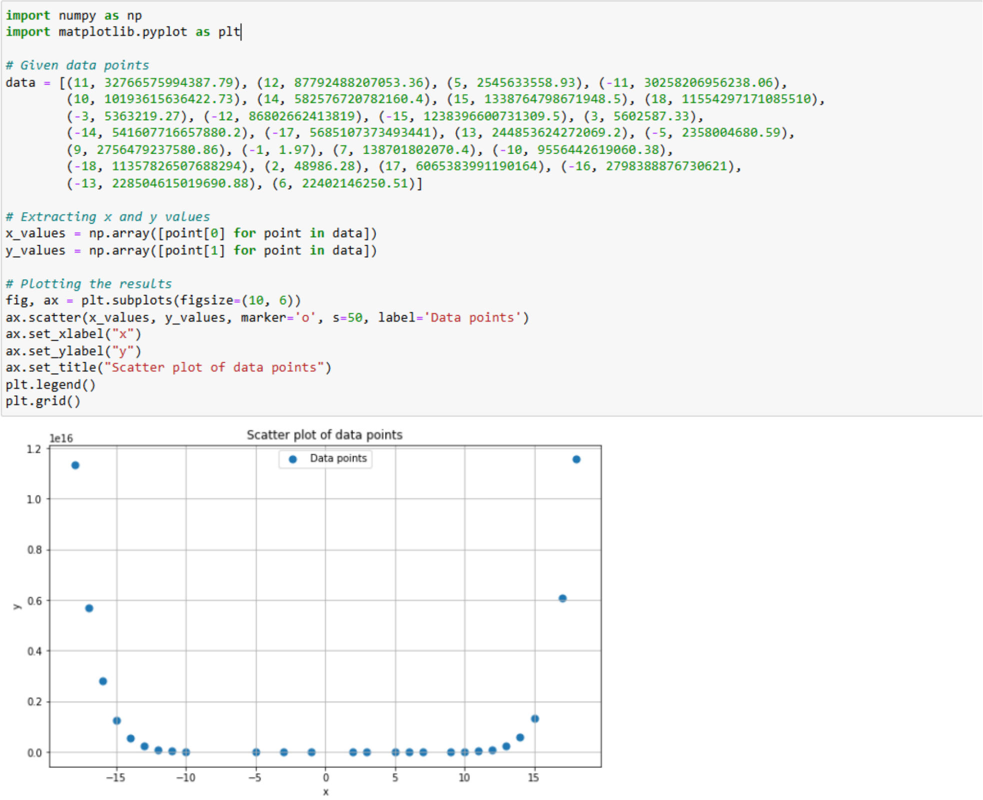
y will be an $n \times 1$ vector of the corresponding y values.

Compute the Ridge-regularized OLS estimate for the parameters:

$$\hat{\alpha}_{Ridge} = (X^T X + \lambda I)^{-1} X^T y)$$

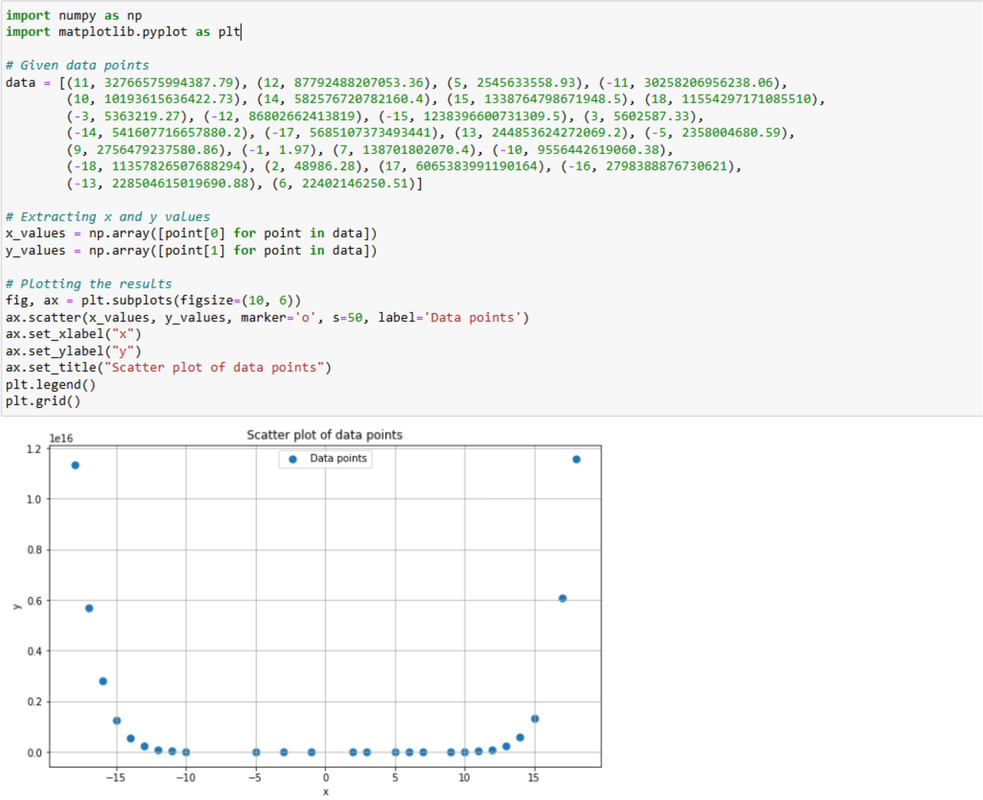
where, λ is regularization parameter.

The following code helps in visualizing the data, OLS Estimates and Ridge Estimates:



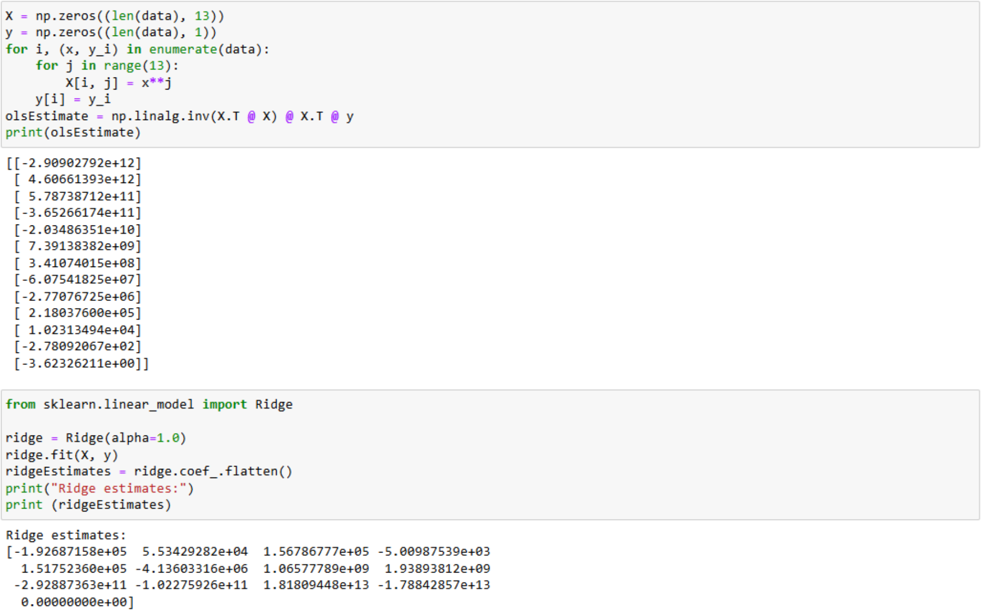
```python
import numpy as np
import matplotlib.pyplot as plt

# Given data points
data = [(11, 32766575994387.79), (12, 87792488207053.36), (5, 2545633558.93), (-11, 30258206956238.06),
        (10, 10193615636422.73), (14, 582576720782160.4), (15, 1338764798671948.5), (18, 11554297171085510),
        (-3, 5363219.27), (-12, 86802662413819), (-15, 1238396600731309.5), (3, 5602587.33),
        (-14, 541607716657880.2), (-17, 5685107373493441), (13, 244853624272069.2), (-5, 2358004680.59),
        (9, 2756479237580.86), (-1, 1.97), (7, 138701802070.4), (-10, 9556442619060.38),
        (-18, 11357826507688294), (2, 48986.28), (17, 6065383991190164), (-16, 2798388876730621),
        (-13, 228504615019690.88), (6, 22402146250.51)]

# Extracting x and y values
x_values = np.array([point[0] for point in data])
y_values = np.array([point[1] for point in data])

# Plotting the results
fig, ax = plt.subplots(figsize=(10, 6))
ax.scatter(x_values, y_values, marker='o', s=50, label='Data points')
ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_title("Scatter plot of data points")
plt.legend()
plt.grid()
```

Figure 5: Visualizing original data points



```python
import numpy as np
import matplotlib.pyplot as plt

# Given data points
data = [(11, 32766575994387.79), (12, 87792488207053.36), (5, 2545633558.93), (-11, 30258206956238.06),
        (10, 10193615636422.73), (14, 582576720782160.4), (15, 1338764798671948.5), (18, 11554297171085510),
        (-3, 5363219.27), (-12, 86802662413819), (-15, 1238396600731309.5), (3, 5602587.33),
        (-14, 541607716657880.2), (-17, 5685107373493441), (13, 244853624272069.2), (-5, 2358004680.59),
        (9, 2756479237580.86), (-1, 1.97), (7, 138701802070.4), (-10, 9556442619060.38),
        (-18, 11357826507688294), (2, 48986.28), (17, 6065383991190164), (-16, 2798388876730621),
        (-13, 228504615019690.88), (6, 22402146250.51)]

# Extracting x and y values
x_values = np.array([point[0] for point in data])
y_values = np.array([point[1] for point in data])

# Plotting the results
fig, ax = plt.subplots(figsize=(10, 6))
ax.scatter(x_values, y_values, marker='o', s=50, label='Data points')
ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_title("Scatter plot of data points")
plt.legend()
plt.grid()
```

Figure 6: Visualizing original data points



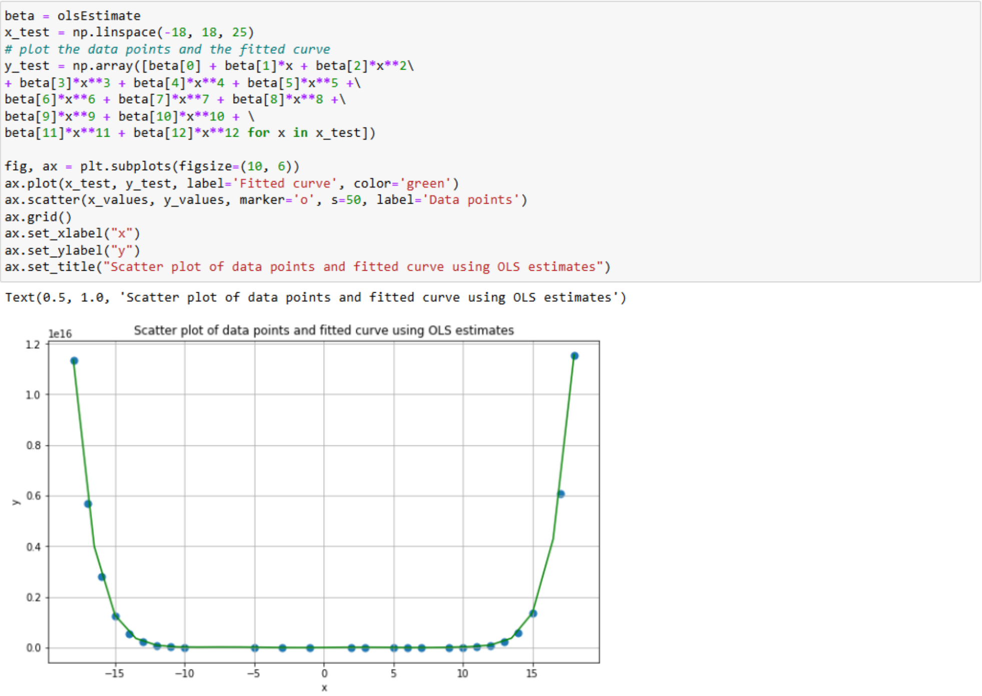
```
X = np.zeros((len(data), 13))
y = np.zeros((len(data), 1))
for i, (x, y_i) in enumerate(data):
    for j in range(13):
        X[i, j] = x**j
    y[i] = y_i
olsEstimate = np.linalg.inv(X.T @ X) @ X.T @ y
print(olsEstimate)
```

```
[[-2.90902792e+12]
 [ 4.60661393e+12]
 [ 5.78738712e+11]
 [-3.65266174e+11]
 [-2.03486351e+10]
 [ 7.39138382e+09]
 [ 3.41074015e+08]
 [-6.07541825e+07]
 [-2.77076725e+06]
 [ 2.18037600e+05]
 [ 1.02313494e+04]
 [-2.78092067e+02]
 [-3.62326211e+00]]
```

```
from sklearn.linear_model import Ridge

ridge = Ridge(alpha=1.0)
ridge.fit(X, y)
ridgeEstimates = ridge.coef_.flatten()
print("Ridge estimates:")
print (ridgeEstimates)
```

```
Ridge estimates:
[-1.92687158e+05  5.53429282e+04  1.56786777e+05 -5.00987539e+03
  1.51752360e+05 -4.13603316e+06  1.06577789e+09  1.93893812e+09
 -2.92887363e+11 -1.02275926e+11  1.81809448e+13 -1.78842857e+13
  0.00000000e+00]
```

Figure 7: Calculating OLS Estimates and Ridge Estimates

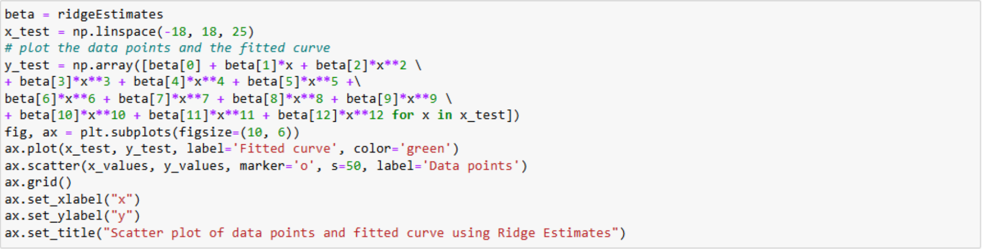


```
beta = olsEstimate
x_test = np.linspace(-18, 18, 25)
# plot the data points and the fitted curve
y_test = np.array([beta[0] + beta[1]*x + beta[2]*x**2\
+ beta[3]*x**3 + beta[4]*x**4 + beta[5]*x**5 +\
beta[6]*x**6 + beta[7]*x**7 + beta[8]*x**8 +\
beta[9]*x**9 + beta[10]*x**10 + \
beta[11]*x**11 + beta[12]*x**12 for x in x_test])

fig, ax = plt.subplots(figsize=(10, 6))
ax.plot(x_test, y_test, label='Fitted curve', color='green')
ax.scatter(x_values, y_values, marker='o', s=50, label='Data points')
ax.grid()
ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_title("Scatter plot of data points and fitted curve using OLS estimates")
```

```
Text(0.5, 1.0, 'Scatter plot of data points and fitted curve using OLS estimates')
```

Scatter plot of data points and fitted curve using OLS estimates
1e16
1.2
1.0
0.8
0.6
0.4
0.2
0.0
y
−15
−10
−5
0
5
10
15
x

Figure 8: Visualizing OLS Estimates



```
beta = ridgeEstimates
x_test = np.linspace(-18, 18, 25)
# plot the data points and the fitted curve
y_test = np.array([beta[0] + beta[1]*x + beta[2]*x**2 \
+ beta[3]*x**3 + beta[4]*x**4 + beta[5]*x**5 +\
beta[6]*x**6 + beta[7]*x**7 + beta[8]*x**8 + beta[9]*x**9 \
+ beta[10]*x**10 + beta[11]*x**11 + beta[12]*x**12 for x in x_test])
fig, ax = plt.subplots(figsize=(10, 6))
ax.plot(x_test, y_test, label='Fitted curve', color='green')
ax.scatter(x_values, y_values, marker='o', s=50, label='Data points')
ax.grid()
ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_title("Scatter plot of data points and fitted curve using Ridge Estimates")
```

```
Text(0.5, 1.0, 'Scatter plot of data points and fitted curve using Ridge Estimates')
```

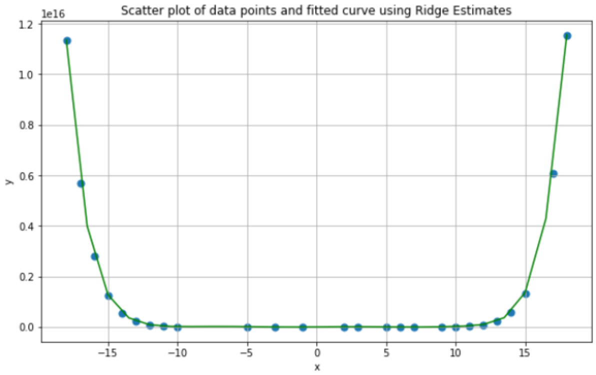


Figure 9: Visualizing Ridge Estimates

1.6 Task 6: Bayesian Estimates

Problem Statement: Following Hogg et al. (2020), exercise 11.2.2: Let $x_1, x_2 \ldots, x_{10}$ be a random sample from a gamma distribution with $\alpha = 3$ and $\beta = 1/\theta$. Suppose we believe that θ follows a gamma-distribution with $\alpha = \xi_{17}$ and $\beta = \xi_{18}$ and suppose we have a trial $x_1, x_2 \ldots, x_n$ with an observed $\overline{x} = \xi_{19}$

- Find the posterior distribution of θ
- What is the Bayes point estimate of θ associated with the square-error loss function?
- What is the Bayes point estimate of θ using the mode of the posterior distribution?

Generated parameters: $\xi_{17} = 69, \xi_{18} = 44, \xi_{19} = 27.79$

Solution:

Part I: Posterior distribution of θ

Let $p(\overline{x}|\theta)$ be likelihood function and $p(\theta)$ be prior distribution.

from GmbH. (2021), the likelihood function is:

$$f(x : \alpha, \beta) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{(\frac{-x}{\beta})}$$

Given $\alpha = 3$ and $\beta = \frac{1}{\theta}$, substituting the values in the above equation.

$$L(\theta|X) = \Pi_{i=1}^{1} 0 \frac{X_i^2 e^- \theta X_i \theta^3}{\Pi(3)}$$

$$L(\theta|X) = (\frac{1}{\Gamma(3)})^{10} \Pi_{i=1}^{1} 0 X_i^2 \theta^{30} e^{-\theta \Sigma_i = 1^{10} X_i + \frac{1}{20}}$$

The prior distribution with $\alpha = 69$ and $\beta = 44$.

$$f(\theta) = \frac{\theta^{68} e^{-\frac{\theta}{44}}}{2^{69} \Gamma(69)}$$

Combining the likelihood function and the prior distribution.

$$p(\theta|x) \propto L(\theta|X) p(\theta)$$

$$p(\theta|x) \propto \theta^{30+69-1} e^{-\theta(\Sigma_{i=1}^{10} X_i + \frac{1}{44})}$$

Given that the sample mean $\overline{x} = 27.7$, we have $\Sigma_{i=1}^{10} X_i = 10 \times 27.79 = 277.9$

$$p(\theta|x) \propto \theta^{98} e^{-\theta(277.9 + \frac{1}{44})}$$

Thus, the posterior distribution of θ is a gamma distribution with parameters $\alpha_{post} = 99, \beta_{post} = \frac{1}{277.9+\frac{1}{44}}$

Part II: Bayes point estimate of associated with the square-error loss function.

The Bayes point estimate of θ associated with the square-error loss function is the mean of the posterior distribution:

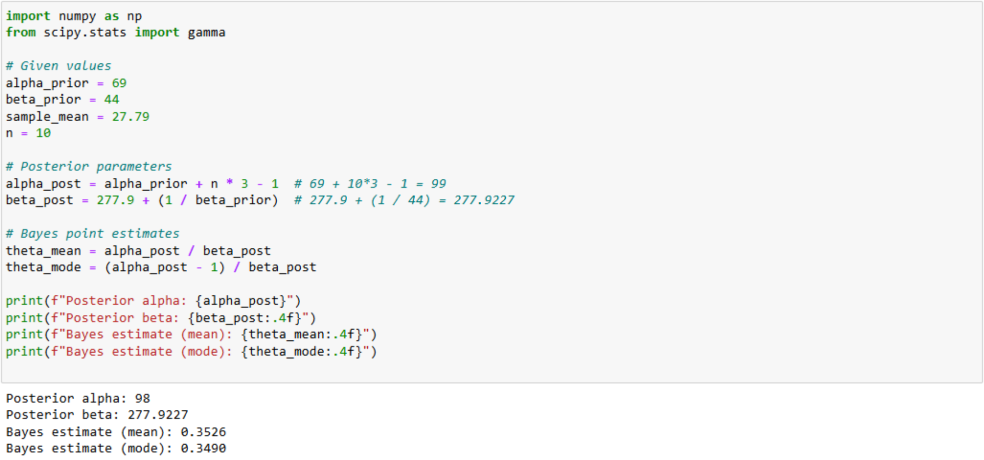
$$\theta_{mean} = \frac{\alpha_{post}}{\beta_{post}}$$
$$\theta_{mean} = \frac{99}{277.9+\frac{1}{44}}$$
$$\theta_{mean} = 0.356$$

Part III: Bayes point estimate of using the mode of the posterior distribution

$$\theta_{mode} = \frac{\alpha_{post}-1}{\beta_{post}}$$
$$\theta_{mode} = \frac{99-1}{277.9+\frac{1}{44}}$$
$$\theta_{mode} = 0.353$$

- The posterior distribution of θ is $\Gamma(99, 0.0036)$.
- The Bayes point estimate of θ associated with the square-error loss function is approximately 0.356.
- The Bayes point estimate of θ using the mode of the posterior distribution is approximately 0.353.

Following code will verify the values that were calculated.



```python
import numpy as np
from scipy.stats import gamma

# Given values
alpha_prior = 69
beta_prior = 44
sample_mean = 27.79
n = 10

# Posterior parameters
alpha_post = alpha_prior + n * 3 - 1  # 69 + 10*3 - 1 = 99
beta_post = 277.9 + (1 / beta_prior)  # 277.9 + (1 / 44) = 277.9227

# Bayes point estimates
theta_mean = alpha_post / beta_post
theta_mode = (alpha_post - 1) / beta_post

print(f"Posterior alpha: {alpha_post}")
print(f"Posterior beta: {beta_post:.4f}")
print(f"Bayes estimate (mean): {theta_mean:.4f}")
print(f"Bayes estimate (mode): {theta_mode:.4f}")
```

```
Posterior alpha: 98
Posterior beta: 277.9227
Bayes estimate (mean): 0.3526
Bayes estimate (mode): 0.3490
```

Figure 10: Posterior Distribution

Bibliography

GmbH., I. (2021): *Advanced Statistics*. Jg. 4, 145–149.

iu INTERNATIONAL
UNIVERSITY OF
APPLIED SCIENCES

Eidesstattliche Erklärung

I hereby certify...

. . . .
Place, date

. . . .
Signature