Project Report

IAS-NASI-INSA

Summer Research Fellowship Programme 2021

Project

Methods of Estimation in Statistics

Keerti P. Charantimath (MATS513)

Indian Institute of Technology, Kharagpur

Under the guidence of

Prof. Anil Kumar Ghosh

Indian Statistical Institute, Kolkata

Name of SRF:	Keerti P. Charantimath
Registration Number:	MATS513
Worked as research fellow at:	Indian Statistical Institute, Kolkata
Date of Joining:	17 May, 2021
Date of Completion:	12 July, 2021
Name of the Guide:	Prof. Anil Kumar Ghosh
Project Title:	Methods of Estimation in Statistics

Prof. Anil Kumar Ghosh,
Theoretical Statistics and Mathematics Unit,
Indian Statistical Institute, Kolkata,
203, B. T. Road, Kolkata 700108, India
Date: 23 July, 2021

Keerti P. Charantimath
MATS513
Date: 23 July, 2021

Prof. Anil Kumar Ghosh,
Theoretical Statistics and Mathematics Unit,
Indian Statistical Institute, Kolkata ,
203, B. T. Road, Kolkata 700108, India

CERTIFICATE

This is to certify that Ms. Keerti P. Charantimath, second year undergraduate student of the Department of Mathematics, IIT Kharagpur was a bonafide summer research fellow through IAS-NASI-INSA Summer Research Fellowship Programme 2021 at Indian Statistical Institute, Kolkata, and has woked under my guidance on the "Methods of Estimation in Statistics" project, from 17th May 2021 to 12th July 2021.

Prof. Anil Kumar Ghosh

ACKNOWLEDGEMENT

I would like to thank my guide, Prof. Anil Kumar Ghosh[1] for the continuous guidance and patience. Being my 1st summer project I had lot of things to learn and the amount of knowledge and experience I have gained from him is matchless. I owe my sincere thanks to him.

I also thank IAS-NASI-INSA for their support throughout the programme and for giving me this great opportunity. It would have not been possible without this programme.

Keerti P. Charantimath

[1]Professor, Theoretical Statistics and Mathematics Unit, Indian Statistical Institute, Kolkata - akghosh@isical.ac.in

Contents

1 Introduction and Problem Statement

In this fellowship, I have been working on **methods of estimating unknown parameters using statistics**. The approach used in this project is known as **Statistical Inference**.

Statistical inference is the process of making judgment about a population based on sample properties. There are mainly two ways of finding inference of a population from a sample. They are:

- **Estimation :** Here, we do not have any prior idea of the population and we form a complete idea of it based on the available sample and observations.
- **Hypothesis Testing :** Here, we already have a prior idea of the population in the form of equations or statements. We test the validity of these equations/ statements using the sample.

Using the above approach, the real world statistical problems that this summer project aims to solve are estimation problems where we would require to estimate certain parameter or value based on the observations obtained from a sample of the population.

Some of the examples of such problems are:

- **The problem of ranking interview candidates when each candidate gives the interview to a combination of panelists.** Here, we do not have the complete data on how each panelist grades every candidate. In such a case, just averaging the score would be biased as some panelists would be lenient while some would be stringent. Here, we require some method which would rank the candidates while nullifying the biases of the panelists.
- **The problem of estimating the probability of occurrence of the blood group genes** A, B, O **when we just have a small observation set and the information of number of people with blood groups** A, B, AB, O **in that set.** This problem can be seen as a problem with incomplete information and it needs some algorithms and statistical methods to be used to obtain the required parameters which are the probability of occurrence of the blood-type genes.
- **The problem of ranking of teams based on matches where not all teams play against each other and the number of matches between**

teams is not fixed. This is again a case where we do not have complete information of each team playing against the other for a fixed number of matches.We need conquer this lack of information to solve the problem.

While we aim to learn methods to solve all such estimation problems, **our final aim is to develop a ranking metric to rank teams or interviewees when complete data is not available.**

2 Methods and Algorithms

The motivation behind estimation is very simple. When a small sample is from a population described by a pdf or pmf $f(x|\theta)$, knowledge of the parameter/ parameters θ yields knowledge of the entire population. Hence, if we are able to find a good method to estimate θ, we would ultimately have complete knowledge of the population.

Throughout this summer project various methods and algorithms to estimate parameters were thoroughly read, discussed and tried on various problems and scenarios. All of these methods and algorithms are listed below.

2.1 Methods of Estimation

There are various methods of estimation. Not all methods give the right estimate of parameter in all cases. Every method has advantages and disadvantages. All of them are briefly discussed below.

2.1.1 Method of Moments

This is the simplest method of estimation. Here, the idea is to equate the sample moments to the population moments to find the unknown parameters of the population distribution.

Advantages :
This method is simple and almost always yields some sort of estimate.

Disadvantages :
The Method of Moments estimate of a parameter sometimes doesn't coincide with the range of the actual parameter.

Methodology :

- Let $x_1, x_2, ..., x_n$ be a sample from a population with pdf or pmf $f(x|\theta_1, \theta_2, \theta_3, ..., \theta_k)$
- Take the first k raw moments of the sample and equate them to the first k raw moments of the population.

$$m_1 = \frac{1}{n}\sum_{i=1}^{n} x_i^1, m_2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2, ..., m_k = \frac{1}{n}\sum_{i=1}^{n} x_i^k$$

$$r_1 = \mu_1(\theta_1, \theta_2, \theta_3, ..., \theta_k), r_2 = \mu_2(\theta_1, \theta_2, \theta_3, ..., \theta_k), .., r_k = \mu_k(\theta_1, \theta_2, \theta_3, ..., \theta_k)$$

Here,
m_i is the i^{th} sample moment
r_i is the i^{th} raw moment with $\mu_i = E(X^i)$

- Equate m_i to r_i for all $i \in (1, 2, 3, ..., k)$. Solve these k simultaneous equations to obtain $\theta_1, \theta_2, \theta_3, ..., \theta_k$

2.1.2 Method of Maximum Likelihood

This method of estimation is based on obtaining population parameters by defining the likelihood function for every point of the sample and finding the parameters which maximize this likelihood function.

Advantages :
This method overcomes the disadvantage posed by the Method of Moments where the estimates of the parameters sometimes lie outside the range of the respective parameters.

Disadvantages :
Finding global maxima and verifying it is sometimes tough when functions are not easily differentiable. Numerical sensitivity also affects the finding of maxima.

This disadvantage is overcome in some cases by **maximising log of the likelihood function** as the global maxima doesn't vary even if we apply log to a function. This method is known as the **Method of Log Likelihood.**

Methodology:

- Let $x_1, x_2, ..., x_n$ be a sample from a population with pdf or pmf $f(x|\theta_1, \theta_2, \theta_3, ..., \theta_k)$
- Obtain the likelihood function of the whole sample by multiplying the individual likelihoods with respect to the assumed distribution.

$$L(\theta) = L(\theta_1, \theta_2, \theta_3, ..., \theta_k | x_1, x_2, ..., x_n) = \prod_{i=1}^{n} f(x_i|\theta_1, \theta_2, \theta_3, ..., \theta_k)$$

  Here, $L(\theta)$ is the likelihood function for the sample set.

- Maximize $L(\theta)$ with respect to the parameters to estimate, to obtain the Maximum Likelihood Estimates of $\theta_1, \theta_2, \theta_3, ..., \theta_k$.
- If it is difficult to directly maximize the likelihood function, therefore $log(L(\theta))$ i.e. Log Likelihood function can be maximized instead to obtain the Maximum Likelihood Estimates of $\theta_1, \theta_2, \theta_3, ..., \theta_k$.

2.1.3 Method of Least Squares

This method estimates the required parameters by minimising the squared error between the the estimated sample values and actual sample values.

When a straight line that reduces the squared error is fit on the sample data, this method is known as **Linear Regression.**

Advantages :
This method is easily differentiable hence it is easy to find the parameters which minimise the squared error. It gives a stable single estimate for the parameters.

Disadvantages :
It is very sensitive to outliers i.e. it is not very robust.

Methodology :

- Let us consider a data-set on a 2D plain with $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ as the sample points.

– Let m be the slope of the line to be fit and c be the y-intercept of the same line. Both m c are known and have to be determined using the Method of Least Squares

$$y_i^{'} = mx_i + c$$

Here $y_i^{'}$ is the estimated value of y_i based on the line fit according to the Method of Least Squares.

– Let e_i be the error of the ith term defined as

$$e_i = y_i^{'} - y_i$$

Now, the sum of squared errors is defined as

$$S = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i^{'} - y_i)^2 = \sum_{i=1}^{n} (mx_i + c - y_i)^2$$

– Minimize S with respect to m and c to obtain the values of the unknown parameters that that give the least square error for the sample set.

2.1.4 Method of Least Absolute Deviations

This method estimates the required parameters by minimising the absolute error between the estimated sample values and actual sample values.

Advantages :
This method is robust against outliers.

Disadvantages :
It doesn't gives a stable single estimate for the parameters. Also, this method is not very easy to minimize and the obtained function of that parameters is not differentiable. Hence, it is difficult to find the parameters which minimise the absolute error directly.

Methodology :

– Let us consider a data-set on a 2D plain with $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ as the sample points.

– Let m be the slope of the line to be fit and c be the y-intercept of the same line. Both m c are known and have to be determined using the Method of Least Absolute Deviations

$$y_i^{'} = mx_i + c$$

Here $y_i^{'}$ is the estimated value of y_i based on the line fit according to the Method of Least Absolute Deviations.

– Let e_i be the error of the ith term defined as

$$e_i = y_i^{'} - y_i$$

Now, the sum of absolute errors is defined as

$$S^{'} = \sum_{i=1}^{n} |e_i| = \sum_{i=1}^{n} |y_i^{'} - y_i| = \sum_{i=1}^{n} |mx_i + c - y_i| \tag{1}$$

– Minimize $S^{'}$ with respect to m and c to obtain the values of the unknown parameters that that give the least absolute error for the sample set.

Note:
The methodologies for Method of Least Absolute Deviations and Method of Least Squares given above are explained only for examples in 2D plane with x_i of power $= 1$. This same methodology can be used for higher powers and dimensions. The only difference would be that the number of unknown parameters to be determined would increase with the increase in powers of x_i or dimensions of x_i and we would be no more fitting lines. With increase in power of x_i, we would be fitting in curves of respective maximum degree and with increase increase in dimension we would be fitting planes or surfaces for 3D case and so on.

2.2 Algorithms used for Estimation

The algorithms are used to iteratively estimate the values of the parameters using the above described methods of estimation when they are difficult to do so directly. The algorithms that are being described below have been coded and tried out for various problems.Also, the convergence for all these algorithms has been discussed and proved.

2.2.1 Iteratively Re-Weighted Least Squares for LAD regression

This algorithm aims to iteratively find the **Least Absolute Deviation (LAD) regression parameters as it is difficult to do so directly.** This as explained in the previous sub-section is because the obtained sub of absolute errors is not differentiable. This is one of the many algorithms that are used in LAD regression.

Algorithm :

- Let us consider the sum of absolute errors $S^{'}$ given in equation 1. According to the LAD regression, we need to minimize $S^{'}$ to obtain the unknown parameters m and c.

- Randomly initialize $m = m_0$ and $c = c_0$.
  (Here, m_0 and c_0 are some constants)
  Let us define $S^{'}$ with respect to the initialized values m_0 and c_0 as follows:

$$S^{'} = \sum_{i=1}^{n} (y_i^{'} - y_i)^2 \frac{1}{|y_i^{'} - y_i|} = \sum_{i=1}^{n} (mx_i + c - y_i)^2 \frac{1}{|m_0 x_i + c_0 - y_i|} \quad (2)$$

- Now, the problem is transformed to sum of weighted squared errors as $\frac{1}{|m_0 x_i + c_0 - y_i|}$ is a constant. The transformed $S^{'}$ in 2 is now easily differentiable. We minimize 2 to obtain m_1 and c_1.

- The iterative updates can be generalized as given below:

$$(m_{k+1}, c_{k+1}) = min_{(m,k)} (\sum_{i=1}^{n} (mx_i + c - y_i)^2 \frac{1}{|m_k x_i + c_k - y_i|})$$

  Iteratively update m and c according to the above equation until convergence.

2.2.2 Step-Wise Algorithm for LAD regression

This algorithm aims to iteratively find the **Least Absolute Deviation (LAD) regression parameters as it is difficult to do so directly.** This as explained in the previous sub-section is because the obtained sub of absolute errors is not differentiable. This is also, one of the many algorithms that are used in LAD regression.

Algorithm :

- Let us consider the sum of absolute errors $S^{'}$ given in equation 1. According to the LAD regression, we need to minimize $S^{'}$ to obtain the unknown parameters m and c.

- Randomly initialize $m = m_0$, where m_0 is a constant.
  Obtain c_0 from the equation below:

$$c_0 = Median[(m_0 x_i - y_i)]_{i=1}^{n}$$

Here, $Median[.]_{i=1}^{n}$ stands for the median of all the observations running from $i = 1$ to n.

– Update m_1 according to the equation below:

$$m_1 = Median[\ |x_i| \diamond \frac{(c_0 - y_i)}{x_i}\ ]_{i=1}^{n}$$

Here, $a \diamond b$ stands b taken a times.

– The iterative updates of m and c can be generalized as:

$$c_k = Median[(m_k x_i - y_i)]_{i=1}^{n}$$

$$m_{k+1} = Median[\ |x_i| \diamond \frac{(c_k - y_i)}{x_i}\ ]_{i=1}^{n}$$

We iteratively update m and c according to the above equations until convergence.

Note:
The algorithms for Iteratively Re-Weighted Least Squares for LAD regression and Step-Wise Algorithm for LAD regression given above are explained only for examples in 2D plane with x_i of power = 1. This same algorithms can be used for higher powers and dimensions. The only difference would be that the number of unknown parameters to be determined would increase with the increase in powers of x_i or dimensions of x_i and we would be no more fitting lines. With increase in power of x_i, we would be fitting in curves of respective maximum degree and with increase increase in dimension we would be fitting planes or surfaces for 3D case and so on.

Also, **there is no unique value of parameters for LAD regression,** hence every run of Iteratively Re-Weighted Least Squares for LAD regression and Step-Wise Algorithm for LAD regression would results in a different value set for parameters according to the initial random initialization. But, **the convergence of both the algorithms to a optimal parameter set is guaranteed.**

2.2.3 Expectation-Maximization (EM) Algorithm

This algorithm aims to find the **Maximum Likelihood estimate of the parameters when it is difficult to do so directly.** A complete data set is assumed and the likelihood function of this complete data is calculated. The parameters to be estimated are then initiated and the expectation of the likelihood function

given the known data is maximised to obtain the next estimate of the wanted parameters. These expectation and maximization steps are repeated until convergence.

Algorithm :

- Let us consider a problem where X is a set of the known/given values and Y is a set of unknown data/values. Set (X, Y) constitutes of the complete data. Also, let θ be a vector of parameters that have to be estimated.
- Let the sample likelihood function based on complete data i.e. (X, Y) be $L_c(\theta)$.
- Initiate θ with some initial constant values θ_0
- **Expectation step :** Calculate expectation of $L_c(\theta)$ given X at θ_0.

$$E_{\theta_0}(L_c(\theta)|X) = \int L_c(\theta) f_{\theta_0}(y|x)dy = Q(\theta, \theta_0)$$

  Here, $f_{\theta_0}(y|x)$ is the conditional density of y given x at θ_0
- **Maximization step :** Maximize $Q(\theta, \theta_0)$ with respect to θ to obtain θ_1
- Repeat the expectation step i.e. calculating $Q(\theta, \theta_k)$ and maximization step i.e. maximizing $Q(\theta, \theta_k)$ to obtain θ_{k+1} until convergence.

Note:
For EM algorithm, the example uses only the complete likelihood function $L_c(\theta)$, but in many cases we can also use the complete log likelihood function $log(L_c(\theta))$ in place of $L_c(\theta)$ according to the ease of maximization.

The convergence of EM algorithm is guaranteed but it might get stuck in local optima in cases of non-uni-modal functions. This can be overcome by methods like genetic algorithms[2]or simulated annealing[3].

[2]Genetic algorithms are randomized search algorithms that have been developed in an effort to imitate the mechanics of natural selection and natural genetics. They follow the principle of the survival of the fittest and hence give out the fittest or the most optimal answer

[3]Simulated Annealing mimics the Physical Annealing process and is used for optimizing parameters in a model. This process is very useful for situations where there are a lot of local minima but the aim is to reach the global minima

2.2.4 MM Algorithm

MM Algorithm is not an algorithm, but a prescription or principle for constructing optimization algorithms.

For **minimization** problems, MM stands for **Majorize-Minimize.**
For **maximization** function, MM stands for **Minorize-Maximize.**

MM algorithm works by creating **surrogate function** (majorization function in the case of minimization problem and minorization function in case of maximization problem). When surrogate function is optimized, the objective function is driven towards the required optimum point.

Algorithm for Maximization problem:

- Let $f(x)$ be a function of vector x which needs to be maximized with respect to x
- Initiate x with some initial constant values x_0
- Find a **minorization function** f_0 such that,

$$f_0(x) \leq f(x) \forall x$$

$$and$$

$$f_0(x_0) = f(x_0)$$

- Find x_1 that maximizes $f_0(x)$
- Generalized minorization function f_k is defined as a function which satisfies

$$f_k(x) \leq f(x) \forall x$$

$$and$$

$$f_k(x_k) = f(x_k)$$

  Maximizing $f_k(x)$ with respect to x would give x_{k+1}. We repeatedly perform this update until convergence.

Algorithm for Minimozation problem:

- Let $f(x)$ be a function of vector x which needs to be minimized with respect to x
- Initiate x with come initial constant values x_0
- Find a **majorization function** f_0 such that,

$$f_0(x) \geq f(x) \forall x$$

$$and$$

$$f_0(x_0) = f(x_0)$$

- Find x_1 that minimizes $f_0(x)$
- Generalized majorization function f_k is defined as a function which satisfies

$$f_k(x) \geq f(x) \forall x$$

$$and$$

$$f_k(x_k) = f(x_k)$$

  Minimizing $f_k(x)$ with respect to x would give x_{k+1}. We repeatedly perform this update until convergence.

Note:
The convergence of MM algorithm is guaranteed but it might get stuck in local optima in cases of non-uni-modal functions. This can be overcome by methods like genetic algorithms or simulated annealing, just like in the case of EM algorithm.

Iteratively Re-Weighted Least Squares for LAD regression and Expectation-Maximization Algorithm are both special cases of MM algorithm.

3 Codes and Outputs

Many problems were tried, discussed and solved using the methods of estimation explained in the Section 2.1. Some of those problems needed the application of the algorithms explained in Section 2.2. These algorithms were coded out and outputs of these programs were recorded. All the codes with the respective problem statements and outputs are recorded below.

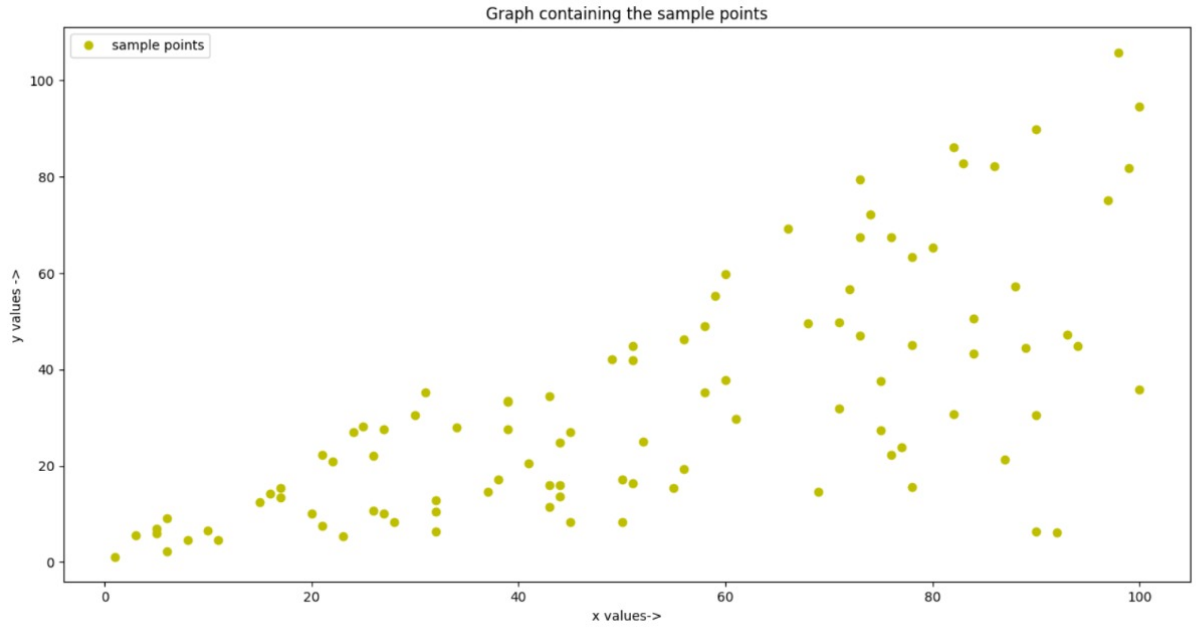

Figure 1: A plot of the sample points generated for LAD regression

3.1 LAD Regression

Problem Setup :

We have 100 points in the 2D plane with the x values being random integers between 1 and 100, and y values being determined by the equation:

$$y_i = 0.01(i)(x_i) + 10(error)$$

where the $error$ is a random floating point number in the range [0.0, 1.0) and subscript i represents the i^{th} sample point.

Figure 1 contains the graph of these sample points.

The unknown parameters are the slope m and the y-intercept c which have to be determined using the LAD regression.

Code :

The unknown parameters were estimated using both Iteratively Re-Weighted Least Square Algorithm for LAD Regression and Step-Wise Algorithm run for 250 iterations each.

The language used to code the simulation and algorithm is python. The complete code is given here.[4]

Output:

Both graphs and terminal outputs were recorded for the Iteratively Re-Weighted Least Square Algorithm for LAD Regression and Step-Wise Algorithm.

Terminal Output:

```
Iteratively Re-Weighted Least Squares

sum of absolute differences before minimization =  2141.817785863387
sum of absolute differences after minimization =  1355.1130867303107
estimated c, m values =  0.9980768825502854 0.59100902240049
```

Figure 2: Terminal output for Iteratively Re-Weighted Least Square Algorithm for LAD regression

```
Step-Wise Algorithm

sum of absolute differences before minimization =  2141.817785863387
sum of absolute differences after minimization =  1355.1099802214353
estimated c, m values =  0.9539644565154681 0.5916303241756291
```

Figure 3: Terminal Output for Step-Wise Algorithm for LAD regression

In the terminal outputs (Figure 2, Figure 3) of Iteratively Re-Weighted Least Square Algorithm for LAD Regression and Step-Wise Algorithm,

– *"sum of absolute differences before minimization"* stands for the value of absolute errors calculated for the random initialization of the y-intercept c and slope m

[4] https://github.com/keertipc7/SRFP_2021_Summer_Project_in_Statistical_Estimation/blob/master/Codes/LADsolvingAlgo.py

– *"sum of absolute differences after minimization"* stands for the value of absolute errors calculated for the values of the y-intercept c and slope m estimated after running the Iteratively Re-Weighted Least Square Algorithm for LAD Regression and Step-Wise Algorithm respectively

– *"estimated c, m values"* stands for the values of the y-intercept c and slope m estimated after running the Iteratively Re-Weighted Least Square Algorithm for LAD Regression and Step-Wise Algorithm respectively

Graphs :

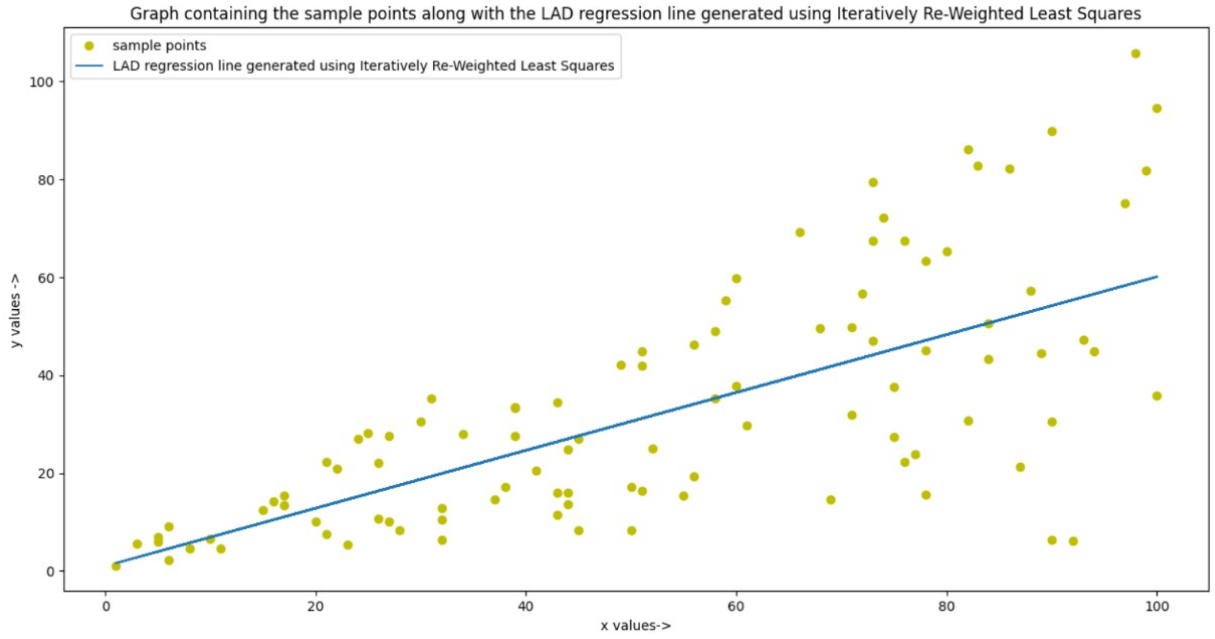

Figure 4: Graph containing the LAD regression line estimated using Iteratively Re-Weighted Least Square Algorithm

We see that the estimated values of y-intercept c and slope m through both the algorithms are in agreement with each other and are very close to each other as well, as seen in Figure 7
Hence, The outputs of the codes are correct and the algorithms converge to the correct values of parameters to be estimated using LAD regression.

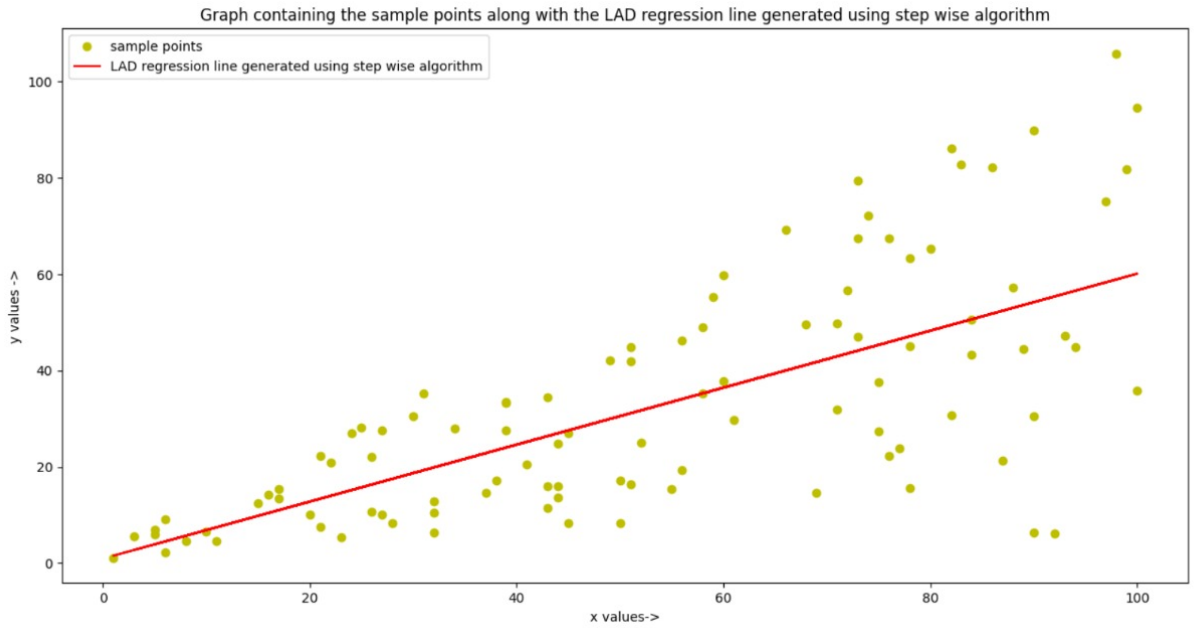

Figure 5: Graph containing the LAD regression line estimated using Step-Wise Algorithm

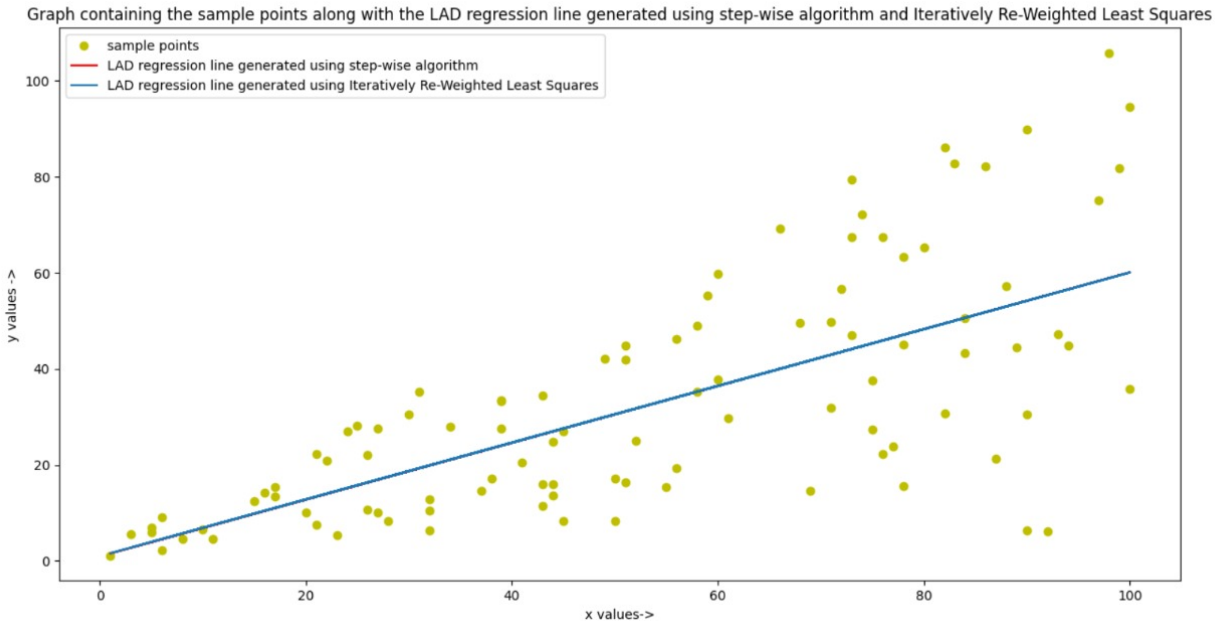

Figure 6: Graph containing the LAD regression lines estimated using Iteratively Re-Weighted Least Square Algorithm and Step-Wise Algorithm

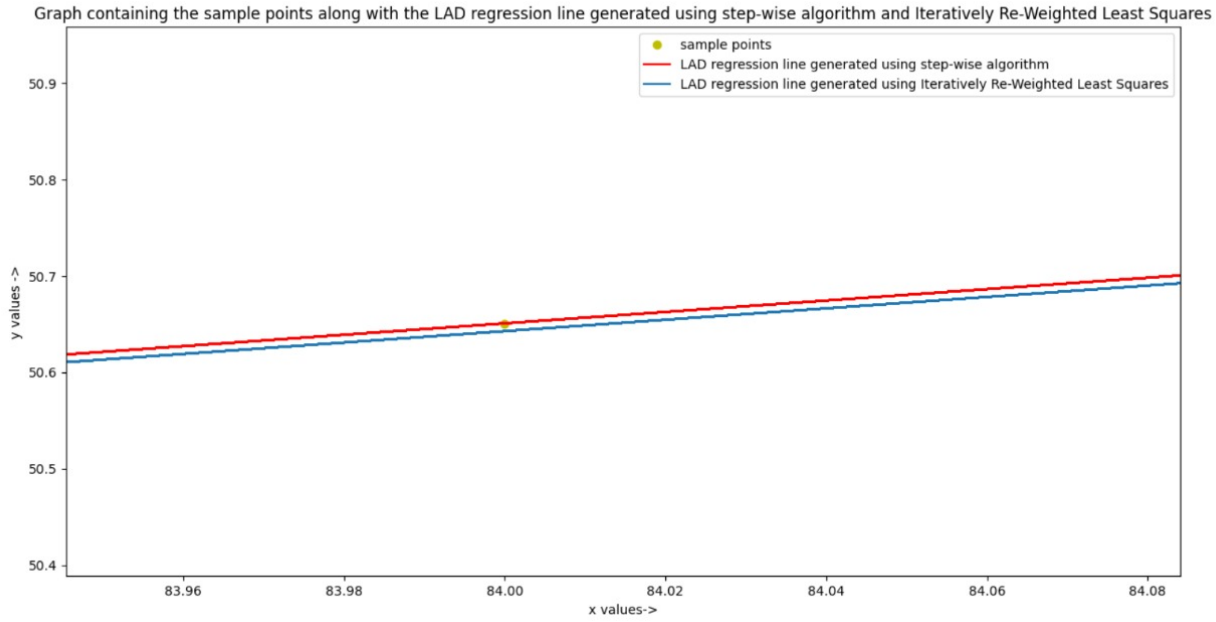

Figure 7: Graph containing a zoomed part of LAD regression lines estimated using Iteratively Re-Weighted Least Square Algorithm and Step-Wise Algorithm

3.2 ABO Blood Group Problem

Problem Setup :

We have Genotype A with probability p, Genotype B with probability q and Genotype O with probability r such that $p+q+r = 1$. It is known that Genotype A and Genotype B are co-dominant while Genotype O is recessive.

Two genes combine and give out one phenotype in a human.

Genes	**Phenotype**
AA	A
AO	A
BB	B
BO	B
OO	O
AB	AB

Given data : We have a total of n samples(humans) out of which n_A are of Phenotype A, n_B are of Phenotype B, n_{AB} are of Phenotype AB and n_O are of Phenotype O.

Parameters to be estimated : We need to estimate p (probability of Genotype A), r (probability of Genotype B) and r (probability of Genotype O).

Known data (X) **:** We know the value of n (total number of samples), n_A (the number of samples with genes AA and AO), n_B (the number of samples with genes BB and BO), n_{AB} (the number of samples with genes AB) and n_O (the number of samples with genes OO).

Unknown data (Y) **:** We do knot know n_{AA} (the number of samples with genes AA), n_{AO} (the number of samples with genes AO), n_{BB} (the number of samples with genes BB) and n_{BO} (the number of samples with genes BO).

Algorithm used : To solve this problem we use the Expectation-Maximization algorithm as we clearly have a set of known data and another of unknown data.

Complete log likelihood function $log(L_c(p, q, r))$ is used here. Expectation of $L_c(p, q, r)$ given the known data X at paramater values p_k, q_k, r_k i.e. $E_{p_k,q_k,r_k}(log(L_c(p, q, r))|X)$ is calulated and maximized to obtain the next estimates of p, q, r which are $p_{k+1}, q_{k+1}, r_{k+1}$.

This is repeated till convergence. Here, k in the number of iteration.

Simulation : We created a sample with sample size $n = 1000$. The values of $p,\ q,\ r$ were set to $0.3,\ 0.3,\ 0.4$ respectively for generating samples. A random number generator based in uniform distribution was used to divide the 1000 s into $n_A,\ n_B,\ n_{AB}$ and n_O. Once these samples were divided into different phenotypes, the set value of $p,\ q,\ r$ was forgotten and only $n_A,\ n_B,\ n_{AB}$ and n_O were used to estimate $p,\ q,\ r$ using EM algorithm.

Code :

The unknown parameters were estimated using Expectation-Maximization algorithm run for 100 iterations.

The language used to code the simulation and algorithm is python. The complete code is given here.[5]

Terminal Output:

```
Solution using EM algorithm run for 100 iterations

initial estimates of p,q,r =  0.5142027203026166 0.3988299285486078 0.08696735114877557
final estimates of p,q,r =  0.31855554739360364 0.2869762324893255 0.39446822011707083
nA,nB,nAB,nO,n =  355 311 180 154 1000
actual values of p,q,r =  0.3 0.3 0.4
```

Figure 8: Terminal output for Expectation-Maximization Algorithm used for ABO Blood group Problem

In the terminal output (Figure 8) of Expectation-Maximization Algorithm used for ABO Blood group Problem,

- *"initial estimates of p,q,r"* stands for the initial random initializations of $p,\ q,\ r$.
- *"final estimates of p,q,r"* stands for the value of $p,\ q,\ r$ estimated after running the Expectation-Maximization Algorithm for ABO Blood group Problem.
- *"nA,nB,nAB,nO,n"* stands for values of $n_A,\ n_B,\ n_{AB},\ n_O$ and n which were generated using the simulation. These constitute the known data (X) of the sample.
- *"actual values of p,q,r"* stands for values of $p,\ q,\ r$ which were used to simulate and divide the n samples into $n_A,\ n_B,\ n_{AB}$ and n_O. These values were used to create the sample set and were also used to verify the output of the Expectation-Maximization Algorithm used for ABO Blood group Problem.

We see in terminal output (Figure 8) that the final estimates of $p,\ q,\ r$ generated by running the Expectation-Maximization Algorithm for ABO Blood group Problem are very close to the actual values of $p,\ q,\ r$ which were used for simulation. Hence, the algorithm is successful in converging to the right estimates.

[5] https://github.com/keertipc7/SRFP_2021_Summer_Project_in_Statistical_Estimation/blob/master/Codes/abo_problem.py

3.3 Team Sorting problem

Problem Setup :

We have m teams which play unequal number of matched with each other. We only have information about the win-loss ratio of the teams and total number of matches played between any two teams. We also have some pairs of teams which have not played any match with each other. In such a setup we need to develop a ranking system based on their capacities for these teams.

Given data : We have a total of m teams with n_{ij} being the number of times team i has won over team j. If there is no math played between i and j, both n_{ij} and n_{ji} would be equal to zero. In other cases, $n_{ij} + n_{ji}$ would given the total number of matches played between team i and team j.

Parameters to be estimated : Such problems are solved using the **Bradley-Terry**[6] model according to which each team has a capacity, say i^{th} team has capacity p_i, and the probability of i winning over j is given by $\frac{p_i}{p_i+p_j}$.
We need to find the Bradley-Terry Model capacities $(p_1, p_2, ..., p_m)$ for all the m teams. The more the capacity of the team, the better it is ranked.

Algorithm used : To solve this problem we use the MM (Minorization - Maximization) algorithm.

The objective function to maximize is the log likelihood function

$$f = log(L(p_1, p_2, ..., p_m))$$

We randomly initialize parameters $p_2, ..., p_m$ to $p_2^0, ..., p_m^0$. Parameter p_1 was set to the value 1 and other parameters were estimated with respect to it. We create the minorization function $g^0(p_1 = 1, p_2, ..., p_m)$ at the initial parameter values $p_2^0, ..., p_m^0$ using the concave function properties and Taylor series expansion of logarithmic functions. The general form of minorization function is $g^k(p_1 = 1, p_2, ..., p_m)$ at the initial parameter values $p_2^k, ..., p_m^k$ for some integer k. Maximize the minorization function $g^k(p_1 = 1, p_2, ..., p_m)$ to get the next estimates i.e. $p_2^{k+1}, ..., p_m^{k+1}$.

This is repeated till convergence. Here, k in the number of iteration.

[6]The Bradley–Terry model is a probability model that can predict the outcome of a paired comparison.

Simulation : We simulated a system with $m = 3$ i.e. three team system. The teams were named as A, B, C. The parameters p_A, p_B, p_C were set to $0.2, 0.3, 0.5$ respectively. 100 matched were simulated between A and B, 200 between B and C and 0 between A and C. Winning of one team over the other was determined using a uniform random number generator.Once the number of wins of one team over the other for the respective number of matches played was determined, the set value of p_A, p_B, p_C was forgotten and only n_{ij} i.e. the number of wins of team i over j for all permutations of i, j were used to estimate p_A, p_B, p_C using MM algorithm.

Code : The unknown parameters were estimated using Minorization-Maximization algorithm run until convergence.s

The language used to code the simulation and algorithm is python. The complete code is given here.[7]

Terminal Output:

```
Final estimates of a, b, c =  1 1.7767750381260954 2.6647951213032814
Final relative estimates of a, b, c =  0.18377048732288398 0.3265188146195683 0.48971069805754774
Actual values of a, b, c =  0.2 0.3 0.5
nAB, nBA, nAC, nCA, nBC, nCB =  36 64 0 0 80 120
```

Figure 9: Terminal output for Minorization-Maximization Algorithm used for Team Sorting Problem

In the terminal output (Figure 9) of Minorization-Maximization Algorithm used for Team Sorting Problem,

- *"Final estimates of a, b, c"* stands for the value of p_A, p_B, p_C estimated after running the Minorization-Maximization Algorithm for Team Sorting Problem, with p_A set to be equal to 1.
- *"Final relative estimates of a, b, c"* stands for the value of p_A, p_B, p_C estimated after running the Minorization-Maximization Algorithm such that $p_A + p_B + p_C = 1$. This estimate was included to check if the algorithm was working correctly.

[7] https://github.com/keertipc7/SRFP_2021_Summer_Project_in_Statistical_Estimation/blob/master/Codes/team_sort.py

– *"Actual values of a, b, c"* stands for values of $p_A,\ p_B,\ p_C$ which were used to simulate and find the win ratio of one team over the other. These values were used to create the sample set and were also used to verify the output of the Minorization-Maximization Algorithm used for Team Sorting Problem.

– *"nAB,nBA,nAC,nCA,nBC, nCB"* stands for values of $n_{AB},\ n_{BA},\ n_{AC},\ n_{CA},\ n_{BC},\ n_{CB}$ where n_{ij} is the number of wins of team i over j. These actually constitute the given data of the sample.

We see in terminal output (Figure 9) that final relative estimates of $p_A,\ p_B,\ p_C$ generated by running the Minorization-Maximization Algorithm for Team Sorting Problem are very close to the actual values of $p_A,\ p_B,\ p_C$ which were used for simulation. Hence, the algorithm is successful in converging to the right estimates.

3.4 Race problem

Problem Setup :

We have m athletes whose combination participate in a race. Not all members run in each race and neither do all athletes run the same number of races. We only have information about the ranking of participants in each race. In such a setup we need to develop a ranking system based on their capacities for these participants.

Given data : We have a total of m participants with the information of their ranking in every race they have participated in.

Parameters to be estimated :
1. Such problems are generally solved using the **Modified Bradley-Terry** model according to which each athlete has a capacity, say i^{th} athlete has capacity p_i, and the probability of i finishing the race before j, k, l is given by $\frac{p_i}{p_i+p_j+p_k+p_l}$.
We need to find the Bradley-Terry Model capacities $(p_1, p_2, ..., p_m)$ for all the m members. The more the capacity of the athlete, the better he/she is ranked.

2. After studying all the methods and approaches, we developed another model which would give similar results as the modified Bradley-Terry model.Our model too associated each athlete with a capacity, say i^{th} athlete has a capacity q_i, and the probability of i finishing the race before j, k, l is given by $\frac{q_i}{q_i+q_j} \cdot \frac{q_i}{q_i+q_k} \cdot \frac{q_i}{q_i+q_l}$.

We need to find the Model capacities $(q_1, q_2, ..., q_m)$ for all the m members. The more the capacity of the athlete, the better he/she is ranked.

Algorithm used : To solve this problem we use the MM (Minorization - Maximization) algorithm.

The objective function to maximize is the log likelihood function

$$f = log(L(p_1, p_2, ..., p_m))$$

We randomly initialize parameters $p_2, ..., p_m$ to $p_2^0, ..., p_m^0$. Parameter p_1 was set to the value 1 and other parameters were estimated with respect to it. We create the minorization function $g^0(p_1 = 1, p_2, ..., p_m)$ at the initial parameter values $p_2^0, ..., p_m^0$ using the concave function properties and Taylor series expansion of logarithmic functions. The general form of minorization function is $g^k(p_1 = 1, p_2, ..., p_m)$ at the initial parameter values $p_2^k, ..., p_m^k$ for some integer k. Maximize the minorization function $g^k(p_1 = 1, p_2, ..., p_m)$ to get the next estimates i.e. $p_2^{k+1}, ..., p_m^{k+1}$.

This is repeated till convergence. Here, k in the number of iteration.

Note : The algorithm used for our model is also the same. Only the parameters to be estimated would be replaced by $(q_1, q_2, ..., q_m)$ and the log likelihood function would vary according to the model.

Simulation : We simulated a system with $m = 4$ i.e. four member system. The members were named as A, B, C, D. 3 races were run in which the first race was run by A, B, C, the second one by all A, B, C, D, and the last race by B, C. The modified Bradley-Terry model parameters of A, B, C, D are p_A, p_B, p_C, p_D respectively. Our model parameters of A, B, C, D are q_A, q_B, q_C, q_D respectively.

Code : The unknown parameters for both the models were estimated using Minorization-Maximization algorithm run until convergence.

The language used to code the simulation and algorithm is python. The complete code is given here.[8]

[8] https://github.com/keertipc7/SRFP_2021_Summer_Project_in_Statistical_Estimation/blob/master/Codes/race.py

Terminal Output:

```
Race 1 = B C A
Race 2 = C D B A
Race 3 = C B

Modified Bradley Terry Approach
a_cap, b_cap, c_cap, d_cap =  1 104.38295550247831 310.7026204055467 207.84713706227598


Our Approach
a_cap, b_cap, c_cap, d_cap =  1 92.74095503665694 326.63631001931896 173.69984834792365
```

Figure 10: Terminal output for Minorization-Maximization Algorithm used for Race Problem

In the terminal output (Figure 10) of Minorization-Maximization Algorithm used for Race Problem,

- *"Race 1 = B C A"* specifies that in the first race, B came first, C second and A came third. Similarly for *Race 2* and *Race 3*.
- *"Modified Bradley Terry Approach a_cap, b_cap, c_cap, d_cap"* stands for the value of modified Bradley-Terry model parameters p_A, p_B, p_C, p_D estimated after running the Minorization-Maximization Algorithm by setting $p_A = 1$.
- *"Our Approach a_cap, b_cap, c_cap, d_cap"* stands for the value of our model parameters q_A, q_B, q_C, q_D estimated after running the Minorization-Maximization Algorithm by setting $q_A = 1$.

We see in terminal output (Figure 10) that the capacity parameter estimates of modified Bradley-Terry model p_A, p_B, p_C, p_D are very close to the capacity parameter estimates of our model q_A, q_B, q_C, q_D. They also are in agreement with the race information given. Hence, the algorithm is successful in converging to the right estimates, and the model developed by us is a valid one.

4 Results

All the codes mentioned in the previous section that can be run to obtain similar results are saved in this[9] GitHub repository. The problems solved above with their respective file names in the mentioned repository are as follows:

Problem	**File Name**
LAD Regression	LADsolvingAlgo.py
ABO Blood Group Problem	abo_problem.py
Team Sorting Problem	team_sort.py
Race Problem	race.py

5 Conclusion

By the end of the fellowship I have been successful in solving real life estimation problems by using estimation methods and algorithms. I was successful in building a methodology to sort and rank teams when incomplete information is given. I also simulated all of the algorithms mentioned and verified that the algorithms as well as the model developed by us gives accurate results and converges to the right estimates.

References

[1] G. Casella and R. Berger. "Statistical Inference". In: 1990.

[2] Yinbo Li and G. Arce. "A Maximum Likelihood Approach to Least Absolute Deviation Regression". In: *EURASIP Journal on Advances in Signal Processing* 2004 (2004), pp. 1–8.

[3] V. Rohatgi and A. Saleh. "An introduction to probability and statistics". In: 1968.

[9] https://github.com/keertipc7/SRFP_2021_Summer_Project_in_Statistical_Estimation/tree/master/Codes