ENGO363: Estimation and Statistical Testing

Term 2024 – Lab 2

Title: Single and multivariate statistics, and confidence intervals

Course instructor: Dr. Ivan Detchev

Teaching Assistant: Michael Blois

Date: February 13, 2024

Student: Wesam Omran

Student ID: 30133992

Academic Integrity Statement:

I, Wesam Omran, certify that this is my own work, which has been done expressly for this course, either without the assistance of any other party or where appropriate I have acknowledged the work of others. Further, I have read and understood the section in the university calendar on plagiarism/cheating/other academic misconduct and I am aware of the implications thereof. WO

Table of Contents

List of Figures iii
List of Tables iii
List of Equation: iv
List of Matrices iv
1 Introduction 1
2 Methodology 1
2.1 C++ Code 1
2.1.1 Calculations.h 1
2.1.2 Calculations.cpp 1
2.1.3 FileHandling.h 3
2.1.4 FileHandling.cpp 3
2.1.5 Main.cpp 4
2.2 Python Code (plot.py) 4
2.3 Bash Script (run.sh) 5
3 Results 6
3.1 Tables 6
3.2 Figures 9
3.3 Matrices 19
4 Discussion 20
4.1 Questions 20
5 Conclusion 21
5.1 Learning Outcomes 21
6 References 22
7 Appendix 23
7.1 Code 23

List of Figures

Figure Name	Figure Number
Observations A against Time	Figure 3.2-1
Observations B against Time	Figure 3.2-2
Probability Density of Residuals A	Figure 3.2-3:
Probability Density of Residuals B	Figure 3.2-4:
Scatter Showing the Correlation of Speed and Goals	Figure 3.2-5
Scatter Showing the Correlation of Weight and Goals	Figure 3.2-6
Scatter Showing the Correlation of Weight and Height	Figure 3.2-7
Scatter Showing the Correlation of Weight and Speed	Figure 3.2-8
Scatter Showing the Correlation of Height and Goals	Figure 3.2-9
Scatter Showing the Correlation of Height and Speed	Figure 3.2-10

List of Tables

Table Name	Table Number
Observations A Results	Table 3.1-1
Observations B Results	Table 3.1-2
Wights Results	Table 3.1-3
Height Results	Table 3.1-4
Speed Results	Table 3.1-5
Goals Results	Table 3.1-6

List of Matrices

Matrix Name	Matrix Number
Variant-Covariant Matrix	Matrix 3.3-1
Correlation Coefficient Matrix	Matrix 3.3-2

List of Equation:

$$Range = l_{max} - l_{min} \quad (1)$$

$$Mean\ \bar{X} = \frac{\sum_{i=1}^{n} l_i}{n} \quad (2)$$

$$Variance\ \sigma_l^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})}{n-1} \quad (3)$$

$$Standard\ Deviation\ \sigma_l = \sqrt{\sigma_l^2} = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})}{n-1}} \quad (4)$$

$$Standard\ Deviation\ of\ the\ Mean = \sigma_{\bar{X}} = \sqrt{\sigma_{\bar{X}}{}^2} = \frac{\sigma_l}{\sqrt{n}} \quad (5)$$

$$Residual\ \hat{v}_i = \bar{X} - l_i = f(x_0, c) - l_i \quad (6)$$

$$Weight\ of\ Observation = P_A = \frac{1}{\sigma_{\bar{X}_A}^2}, P_B = \frac{1}{\sigma_{\bar{X}_B}^2} \quad (7)$$

$$Weighted\ Mean\ \bar{X}_{WM} = \frac{\sum_{k=1}^{n} P_k \bar{X}_k}{\sum_{k=1}^{n} P_k} \quad (8)$$

$$Standard\ Deviation\ of\ Weighted\ Mean\ \sigma_{WM} = \sqrt{\frac{\sum_{k=1}^{n} P_k \hat{v}_k^2}{(k-1)\sum_{k=1}^{n} P_k}} \quad (9)$$

$$where\ \hat{v}_k = \hat{X}_{WM} - \bar{X}_k$$

$$Confidence\ Interval\ CI = \bar{X} \pm z\frac{\sigma}{\sqrt{n}} \quad (10)$$

$$Covariance\ \sigma_{ab} = \frac{1}{n-1}\sum_{i=1}^{n}(\hat{v}_{a_i} \cdot \hat{v}_{b_i}) \quad (11)$$

$$Variance - Covariance\ Matrix\ C = \begin{bmatrix} {\sigma_{a_1}}^2 & \sigma_{a_1 a_2} & \cdots & \cdots & \sigma_{a_1 a_n} \\ & {\sigma_{a_2}}^2 & \cdots & \cdots & \sigma_{a_2 a_n} \\ & & \ddots & \cdots & \vdots \\ & & & \ddots & \vdots \\ Symm & & & & {\sigma_{a_n}}^2 \end{bmatrix} \quad (12)$$

$$where\ \sigma_{a_n a_m} = \frac{1}{n-1}\sum_{i=1}^{n}(\hat{v}_{n_i} \cdot \hat{v}_{m_i})$$

$$where\ \hat{v}_{n_i} = \bar{n} - n_i$$

$$Correlation\ Coefficient\ \rho_{ab} = \rho_{ba} = \frac{\sigma_{ab}}{\sigma_a \cdot \sigma_b} \quad (13)$$

$$Correlation\ Matrix\ R = \begin{bmatrix} 1 & \rho_{ab} & \cdots & \cdots & \rho_{an} \\ & 1 & \cdots & \cdots & \rho_{bc} \\ & & \ddots & \cdots & \vdots \\ & & & \ddots & \vdots \\ Symm & & & & 1 \end{bmatrix} \quad (14)$$

1 Introduction

This lab aims to explore the implementation of statistical equations in C++ language and visualization in Python. Using a series of calculations to find the confidence intervals and statistics to calculate and visualize the results.

2 Methodology

The program was divided into 3 main programs. The first using C++ to read the observations, compute the needed calculations, and write the results to files that will be handled by the Python script to plot graphs for better visualizations of the result. The code is run through the terminal (command line or Linux system) using a Bash script.

2.1 C++ Code

The C++ code is divided into 3 classes, each responsible for its own functions; Calculations, File Handling, and Main.

2.1.1 Calculations.h

This is the header file for Calculations class. It is responsible for making sure functions exist in its respective C++ file. The header file is included in the Main file to make sure it can utilize the functions.

2.1.2 Calculations.cpp

This file contains all the functions to calculate results.

2.1.2.1 Sort (sort)

The Sort function is responsible for sorting vectors in an ascending order. It takes in a vector as a parameter. First, assign the vector size to an integer variable. This will be used when looping through the vector. Then, it goes into a nested loop, the first to which is used to loop through the vector whether changes are made or not. The second checks the index it is at and compares it to the next one. If the next value is bigger than the current value, the loop goes through the next index. If the next value is smaller, then it swaps the values of the indices. Finally, the loop ends, and the function returns the same vector but sorted.

This method was developed assuming that the data is not sorted in ascending order. It makes it easier to write functions to find the range and median of the vector. Additionally, another purpose would be to make it easier for the developer or the user to read the values of the vector when printing the content of the vector out.

2.1.2.2 Range (range)

The Range function is responsible for finding the range of a given vector. The way it does it by assigning the first value of the vector to a variable *(first)* and then the last value of the vector to a variable *(last)* and then it returns the value of the last value minus the first value.

2.1.2.3 Median (median)

The Median function is responsible for finding the median value of the vector given as a parameter. It starts by initializing a variable that will later hold the value of the median. Then, using the mod operator, it will check if the number of values in the vector is even or odd. This will allow the function to perform 2 different approaches to finding the median. If the vector size is even, then it

assigns the 2 middle values of the vector to *(lower)* and *(upper)* and then it adds them up and divides them by 2 and then return that value.

2.1.2.4 Mean (mean)

The Mean function is responsible for finding the mean value of the vector given as a parameter. Using equation (2), it starts by initiating variable that will later hold the value of the mean *(sum)*. Then, by looping through the vector, it adds each index value to the sum. After finishing with the loop, it assigns the value of the mean to *(meanValue)* by dividing the sum by the size of the vector. Finally, it returns the mean value.

2.1.2.5 Residuals (residuals)

The Residuals function is responsible for finding the residuals vector of a given vector with its mean. Using equation (6), it starts by initiating a variable that will hold the value of each index's residual *(residual)*. Then it loops through the vector, and assigns the residual value as mean minus the value of the index of the vector. Then, it switches the value of the index to the residual value. After the loop is done, it returns the vector with the updated values.

2.1.2.6 Variance (variance)

The Variance function is responsible for finding the variance of a given vector and its mean as its parameters. Using equation (3), it starts by initializing a variable with value 0 *(sum)*. Then, it loops around the vector and adds the value of the index value minus the mean all squared and adds it to the sum. Then assigns the value of the sum divided by the vector size – 1 and returns it.

2.1.2.7 Standard Deviation (standardDeviation)

The Standard Deviation function is responsible for finding the standard deviation of a vector given its variance as a parameter. By using equation (4), it finds the square root of the variance value and returns it.

2.1.2.8 Mean Standard Deviation (meanSD)

The Mean Standard Deviation function is responsible for finding the mean standard deviation of a given vector and its standard deviation. Using equation (5), it divides the standard deviation by the square root of the vector size and returns it.

2.1.2.9 Sum (sum)

The Sum function is responsible for finding the sum of a given vector. It does that by looping through the values of the vector and adding them together. Then returns the value of the sum.

2.1.2.10 Weight (weight)

The Weight function is responsible for finding the weight of an observation. Using equation (7), it computes the inverse of the mean standard deviation squared and return its value.

2.1.2.11 Weighted Mean (weightedMean)

The Weighted Mean function is responsible for finding the weighted mean value of a given vector of weights vector *(weights)* and means vector *(means)*. Using equation (8), it initiates and assigns a value of 0 to the sum of the numerator *(sum1)* and the sum of the denominator *(sum2)*. Then using a loop, it adds the value of the index of weights vector times the index of means vector to the numerator sum; and adds the value of the index of weights vector to the sum of the denominator. Finally, it returns the value of the sum of the numerator divided by the sum of the denominator.

2.1.2.12 Standard Deviation of the Weighted Mean (weightedMeanSD)

The Standard Deviation of the Weighted Mean function is responsible for finding the standard deviation of the weighted mean using weights vector *(weights)*, mean vector *(means)*, and their value of the weighted mean *(wm)* as parameters. Using equation (9), it initiates and assigns a value of 0 to the sum of the numerator *(sum1)* and the sum of the denominator *(sum2)*. Then using a loop, it adds the value of the index of weights vector and multiplies it by weighted mean – the index of means squared; and adds the value of the index of weights vector to the sum of the denominator. Finally, it returns the value of the square root of the numerator sum divided by the denominator sum multiplied by the number of observations – 1.

2.1.2.13 Transpose (transpose)

The transpose function is responsible for transposing a given matrix. This function loops through the matrix first and its columns secondly. Then it swaps the values of row index with the column index. Finally, it returns the transposed matrix.

2.1.2.14 Covariance (covariance)

The Covariance function is responsible for finding the covariance between 2 given vectors. Using equation (11), it initiates and assigns the value of the mean of each vector to 2 different variables and initiates a sum variable. Then, loops through the vectors and adds the value of multiplying the residuals of each vectors. Finally, it returns the value of the inverse of the vectors sizes times the sum.

2.1.2.15 Variant Covariant Matrix (variantCovariant)

The Variant Covariant Matrix function is responsible for finding the variant covariant matrix by taking in an empty matrix and residuals matrix of all given single variate samples. Using equation (12), it loops through the rows and columns of the empty matrix and uses the Covariant function to assign each index of the matrix. Finally, it returns the adjusted matrix.

2.1.2.16 Correlation Coefficient Matrix (correlationCoefficient)

The Correlation Coefficient Matrix function is responsible for finding the correlation coefficient matrix by taking in an empty matrix that will hold the values of correlation, the variant covariant matrix, and a vector of each single variate standard deviation. as parameters. Using equation (13), it loops through the empty matrix and executes the calculation by dividing the index value of the variant-covariant matrix by the standard deviation of the first single variate multiplied by the standard deviation of the second single variate. Finally, it returns the adjusted matrix.

2.1.3 FileHandling.h

This is the header file for FileHandling class. It is responsible for making sure functions exist in its respective C++ file. The header file is included in the Main file to make sure it can utilize the functions.

2.1.4 FileHandling.cpp

This file contains all the functions to read and write the observations and results.

2.1.4.1 Read File (readFile)

The Read File function is written with 2 different parametrizations.

2.1.4.1.1 readFile 1

The first one will be responsible for reading the contents of the Observations A and Observations B files. Since these files have 2 different columns each responsible for different categories. It takes in file

name and a vector as parameters. It loops through the file and assigns the second column value to the vector.

2.1.4.1.2 readFile 2

The second one is responsible for reading the contents of the single variate files. It takes in the file name and returns a vector of the values of the file's content. It loops through the file and assign each value of each line to a vector and returns it.

2.1.4.2 Write to File (writeToFile)

The Write to File function is written with 3 different parametrizations.

2.1.4.2.1 writeToFile 1

The first one is responsible for writing the results of the calculation computed from Observations A and Observations B. It takes in the file name, vector that will hold the category name, and a vector that holds the values respectively. By looping through the vectors, it writes the values of the keys vector and the value for the results vector separated by a tab.

2.1.4.2.2 writeToFile 2

The second one is responsible for writing the residuals to a file. Using the file name and the residuals vector passed as parameters, it loops through the residuals vector and adds each value to a line in the file.

2.1.4.2.3 writeToFile 3

The third one is responsible for writing the variance-covariance and correlation coefficients matrices. By passing the file name and a 2D vector, it loops through the 2D vector and outcomes the value of the index at row x column. Each value followed by a tab. After it's done iterating through each row, it inserts a new line character to allow for the second row to be written to a new line.

2.1.5 Main.cpp

The Main file is responsible for running the code. It performs the functions from FileHandling and Calculations classes to read the files, find the results and write them to files that will be used in the python code for visualizations. It is also used to print out the contents of the results and debug any errors happening in the code.

2.2 Python Code (plot.py)

The Python file uses the Numpy, Matplotlib, and Scipy libraries to visualize the results computed in the C++ file.

For the first task, it starts by reading the observations files and assigning the time values and the observations to their respective lists. Then, it reads the results file and assigns each key and its value to a Python dictionary. After that, it calculates the degrees of freedom by subtracting the number of observations by 1. Then, it creates a list of upper and lower bounds of the confidence interval of each observation. This is done by using equation (10), where the lower bound is equal to mean minus 2 standard deviations (for 95% confidence interval) and the upper bound is equal to mean plus 2 standard deviations. Finally, it graphs the observations against time with the mean, weighted mean, and the confidence interval.

After that, it loads the values of the residuals by reading the content of the residuals file. It then calculates the confidence interval. Having the lower bound equal to mean minus 3 standard deviations (for 99% confidence interval) and upper bound equal to mean plus 3 standard deviations (for 99% confidence interval). Finally, it creates a histogram of residuals probability density with the mean of residuals, and the confidence interval.

For the second task, it reads in the content of each single variate file and assigns the values in respective lists. Finally, it plots each single variate against another in a scatter plot.

2.3 Bash Script (run.sh)

Since the program was executed through the terminal and not using an IDE, a bash script was written to provide faster compilation of the code. It starts by compiling the C++ code into class and object files. Then links them into a compiled Main executable program. Finally, it runs the Main.exe file using "./Main" command.

3 Results

The data was collected by utilizing a C++ algorithm to analyze the statistics of two observations, their residuals, and single variate samples. Subsequently, the calculated values were compared using Python libraries.

3.1 Tables

Table 3.1-1

Observations A Results

Keys	Values
Range	0.1450 [m]
Median	137.4150 [m]
Mean	137.4137 [m]
Variance	0.0011 [m^2]
Standard Deviation	$\pm$0.0325 [m]
Mean Standard Deviation	$\pm$0.0059 [m]
Residuals Range	0.1450 [m]
Residuals Median	-0.0013 [m]
Residuals Sum	0 [m]
Residuals Mean	0 [m]
Residuals Variance	0.0011 [m^2]
Residuals Standard Deviation	$\pm$0.0325 [m]
Weighted Mean	137.4129 [m]
Weighted Mean Standard Deviation	$\pm$0.0039 [m]

This table gives us information about Observation A. It shows how closely the measurements are around the mean. It also shows that the range, variance, the standard deviation, and the mean standard deviation of the residuals is equal to those of the observations.

Table 3.1-2

Observations B Results

Keys	Values
Range	0.7900 [m]
Median	137.4000 [m]
Mean	137.3944 [m^2]
Variance	0.0242 [m]
Standard Deviation	$\pm$0.1554 [m]
Mean Standard Deviation	$\pm$0.0210 [m]
Residuals Range	0.7900 [m]
Residuals Median	-0.0056 [m]
Residuals Sum	0 [m]
Residuals Mean	0 [m]
Residuals Variance	0.0242 [m^2]
Residuals Standard Deviation	$\pm$0.1554 [m]
Weighted Mean	137.4129 [m]
Weighted Mean Standard Deviation	$\pm$0.0039 [m]

This table gives us information about Observation B. It shows more spread of observations due to significantly higher standard deviation compared to Observation A. Indicating that the values are more spread out than those in Observation A.

Table 3.1-3

Weights Results

Keys	Values
Mean	76.64 [Kg]
Variance	32.4649 [Kg^2]
Standard Deviation	5.6978 [Kg]

Table 3.1-3 shows the calculated statistics for weight.

Table 3.1-4

Heights Results

Keys	Values
Mean	1.7725 [m]
Variance	0.0073 [m^2]
Standard Deviation	0.0853 [m]

Table 3.1-4 shows the calculated statistics for height.

Table 3.1-5

Speed Results

Keys	Values
Mean	6.5750 [m/s]
Variance	1.2508 [m^2/s^2]
Standard Deviation	1.1184 [m/s]

Table 3.1-5 shows the calculated statistics for speed.

Table 3.1-6

Goals Results

Keys	Values
Mean	10.4000 [g]
Variance	16.7111 [g^2]
Standard Deviation	4.0879 [g]

Table 3.1-6 shows the calculated statistics for goals.

3.2 Figures

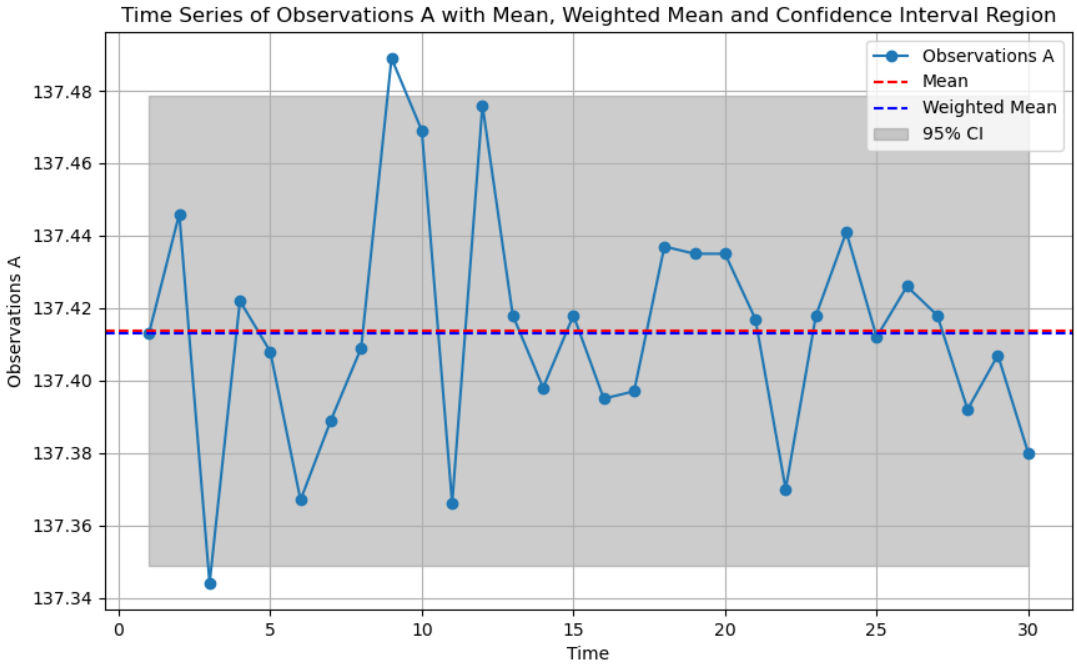

Figure 3.2-1

Plot showing the 95% CI, Mean, and Weighted Mean of the time series of Observations A

Figure 3-2-1 illustrates the proximity between the weighted mean and the mean of A, indicating a stronger influence of the weight of A compared to Figure 3-2. Additionally, the graph demonstrates the standard deviation, indicating the closer alignment of each value.

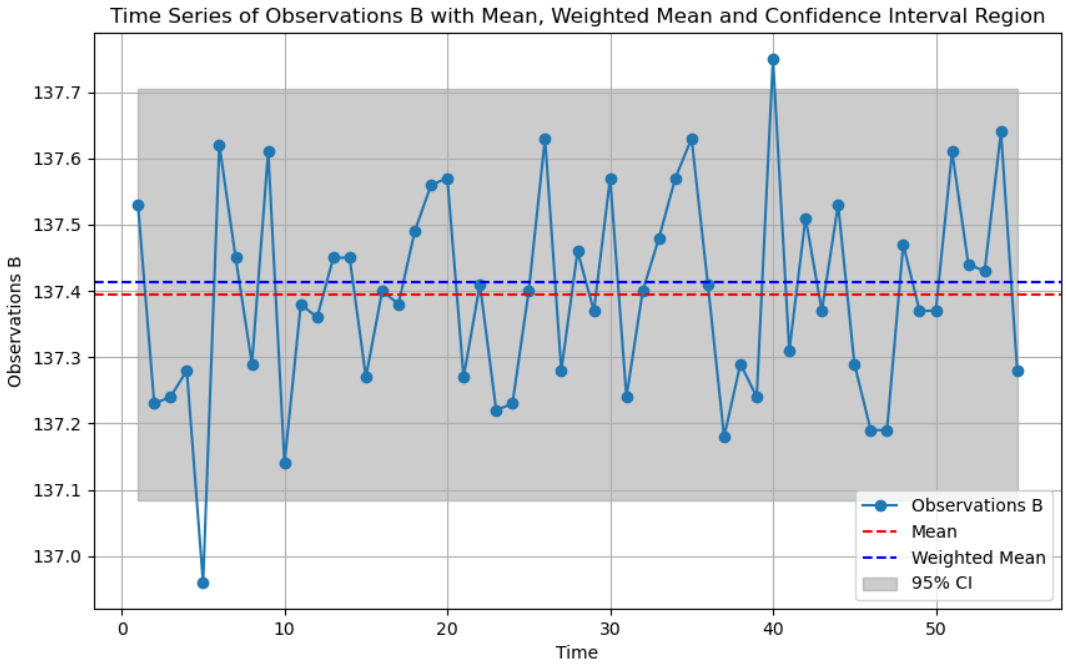

Figure 3.2-2

Pot showing the 95% CI, Mean, and Weighted Mean of the time series of Observations B

Figure 3-2-2 reveals that the values at 5 seconds and 40 seconds fall outside the confidence interval (CI), which is expected given the higher standard deviation. Consequently, this graph highlights the deviation present in each value.

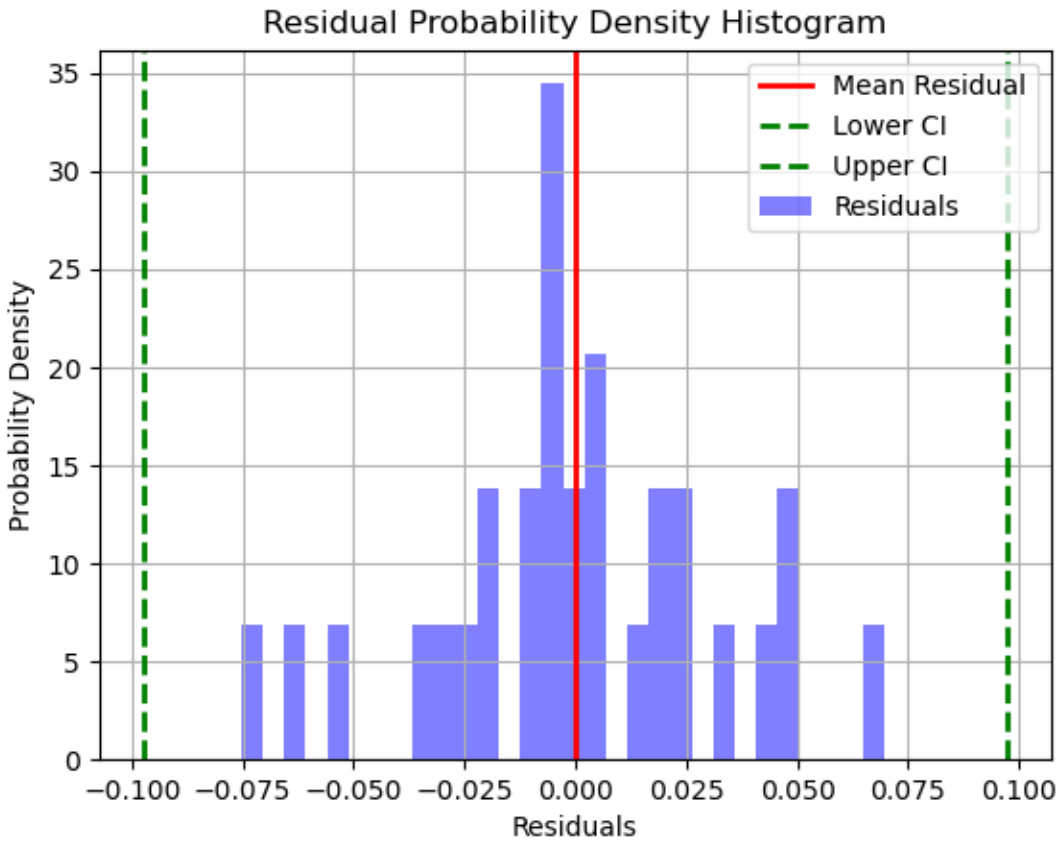

Figure 3.2-3:

Histogram showing the 99% CI, Mean, and weighted Mean of the time series of Residuals A

In Figure 3-2-3, we see a graph that shows how often different leftover values occur after making observations. It looks a bit like a normal curve, with most leftovers clustered around zero, meaning they're close to what we expected. However, there are some leftovers that are further away from zero, showing that sometimes our observations don't match our expectations perfectly.

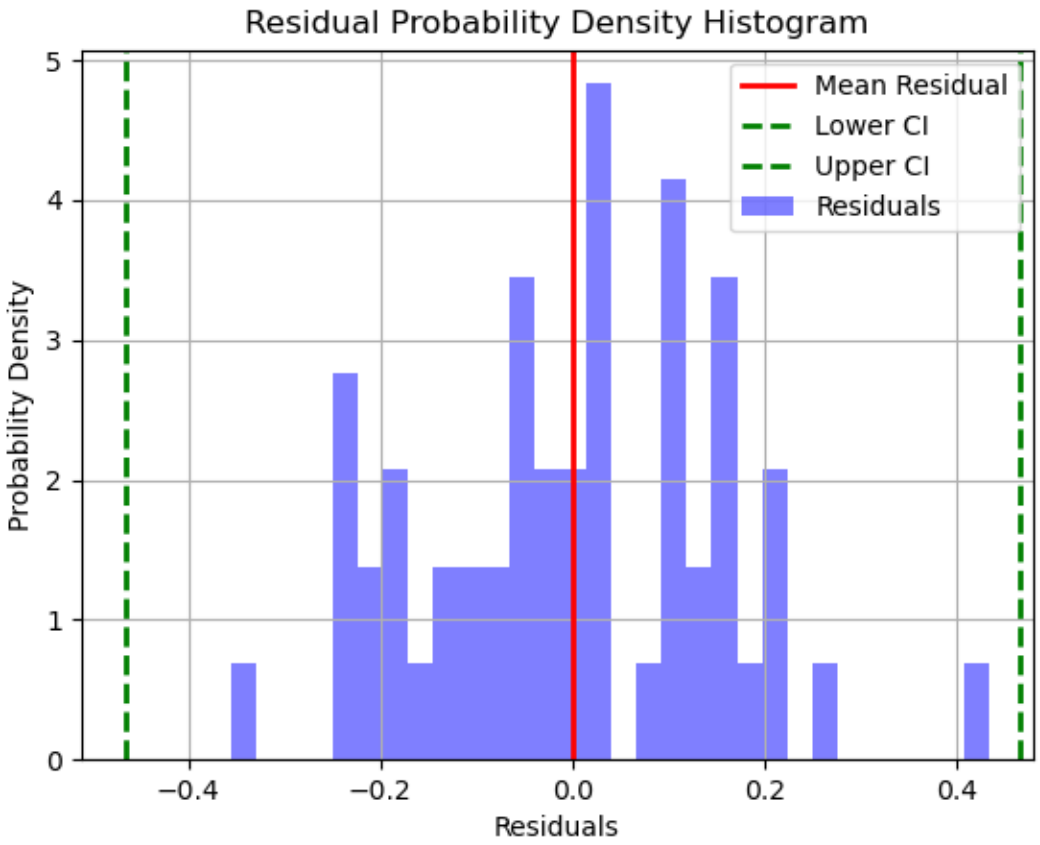

Figure 3.2-4:

Histogram Showing the 99% CI, Mean, and Weighted Mean of the time series of Residuals

Figure 3-2-4 displays a graph representing how often different leftover values occur after making observations B. This graph resembles a normal curve, with the majority of leftovers centered around zero, indicating they are close to what was expected. However, there are more residuals that deviate from zero compared to the graph for observations A, suggesting that observations in B have more variability or differences from the expected values.

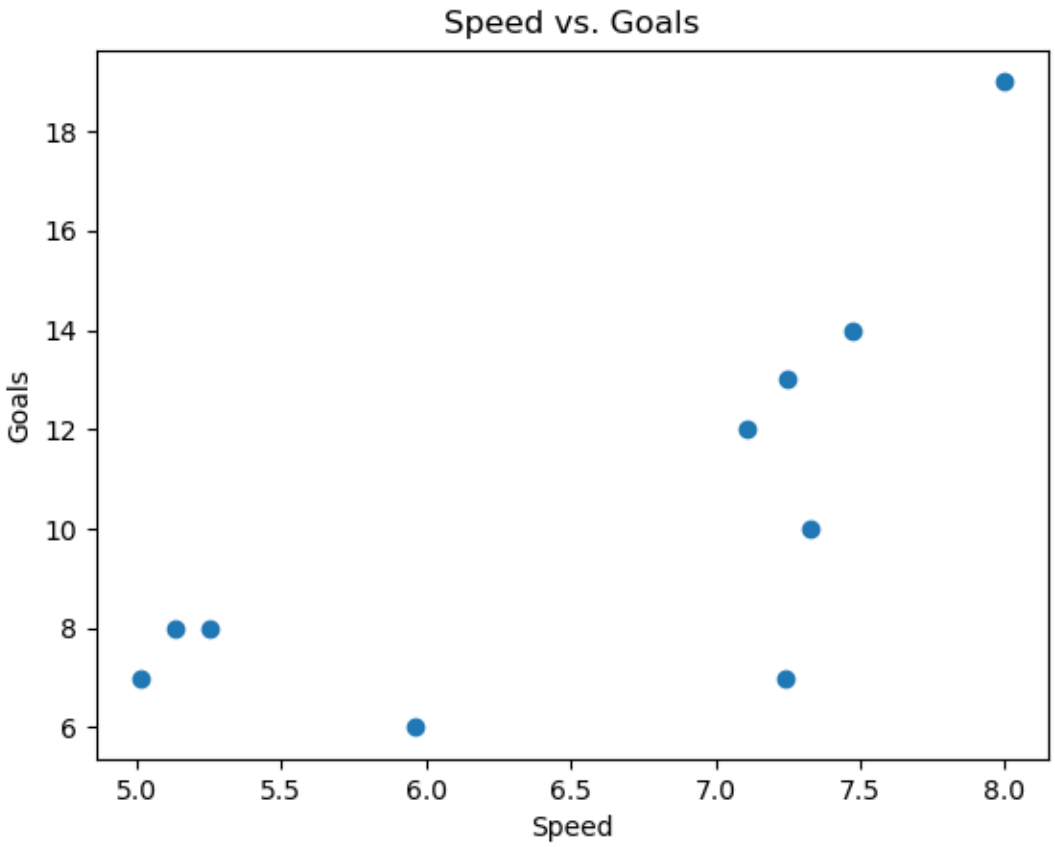

Figure 3.2-5

Scatter Showing the Correlation of Speed and Goals

Figure 3-2-5 illustrates the relationship between a player's speed and the number of goals they score. The graph demonstrates some positive correlation, indicating that as the player's speed increases, their goal-scoring tends to increase as well.

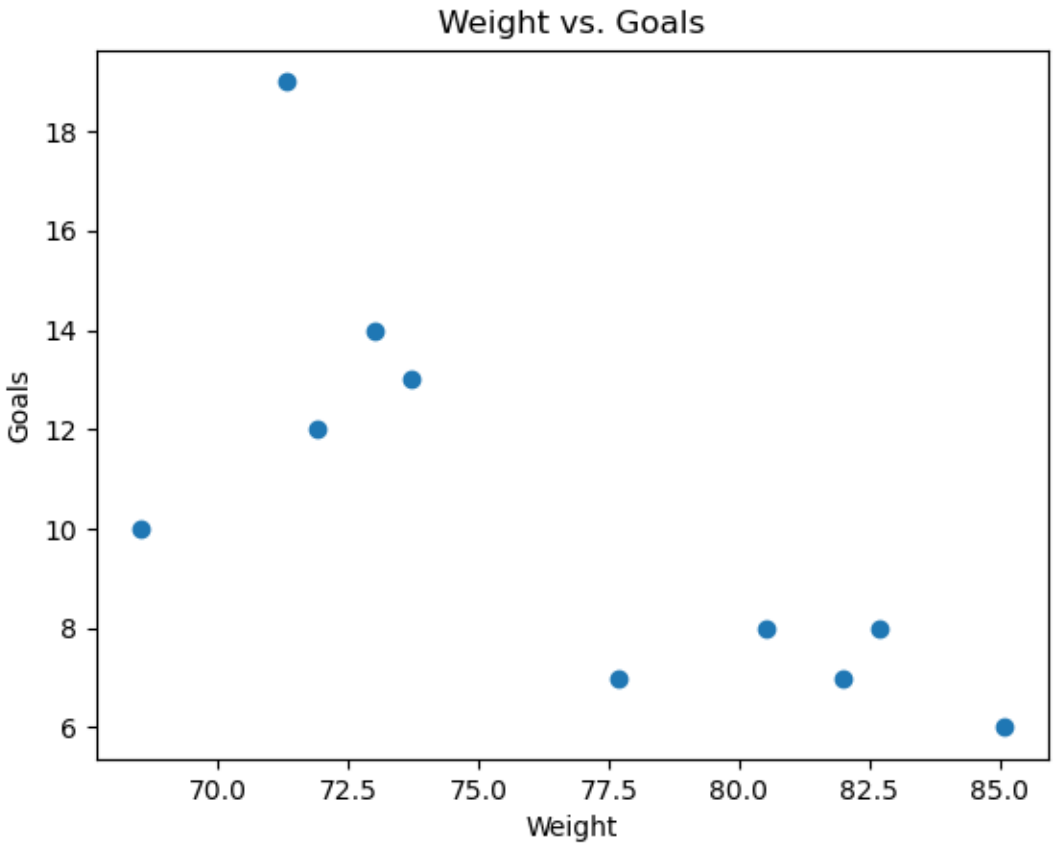

Figure 3.2-6

Scatter Showing the Correlation of Weight and Goals

Figure 3-2-6 illustrates the relationship between a player's weight and the number of goals they score. The graph demonstrates some negative correlation, indicating that as the player's weight increases, their goal-scoring tends to decrease.

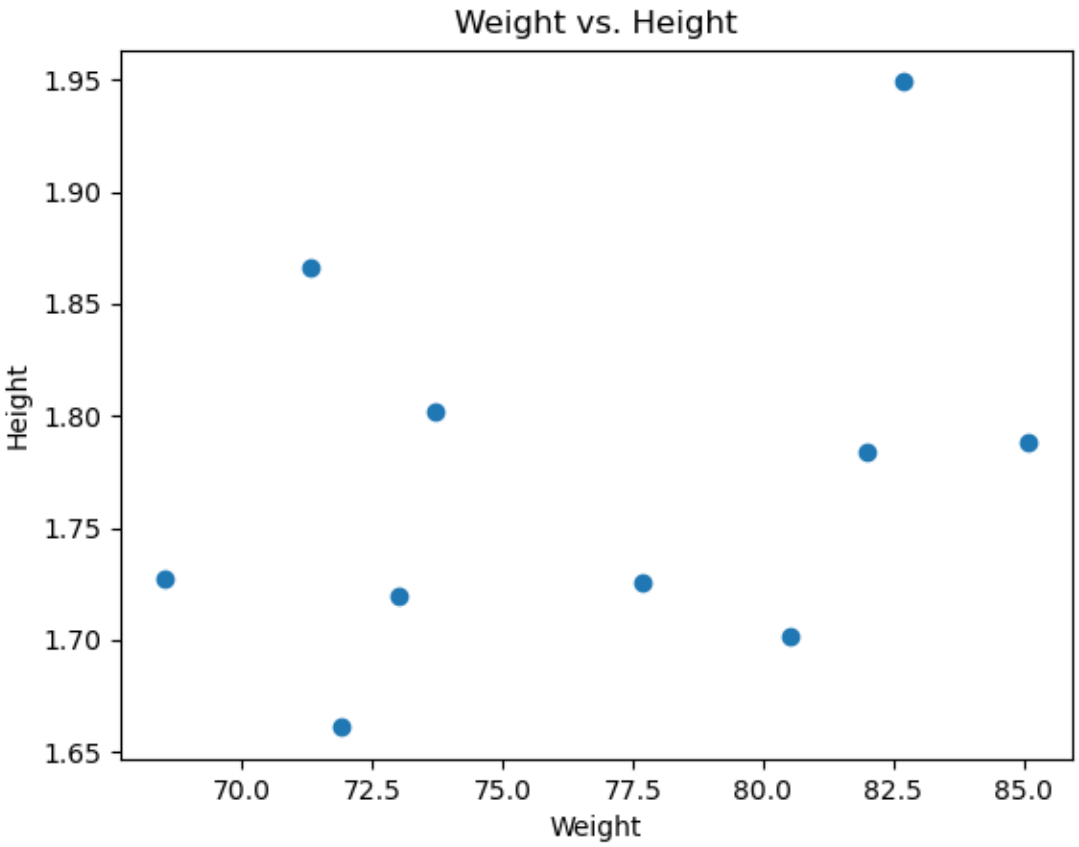

Figure 3.2-7

Scatter Showing the Correlation of Weight and Height

Figure 3-2-7 illustrates the relationship between a player's weight and their height. The graph demonstrates almost no correlation, indicating that the player's height does not affect their height.

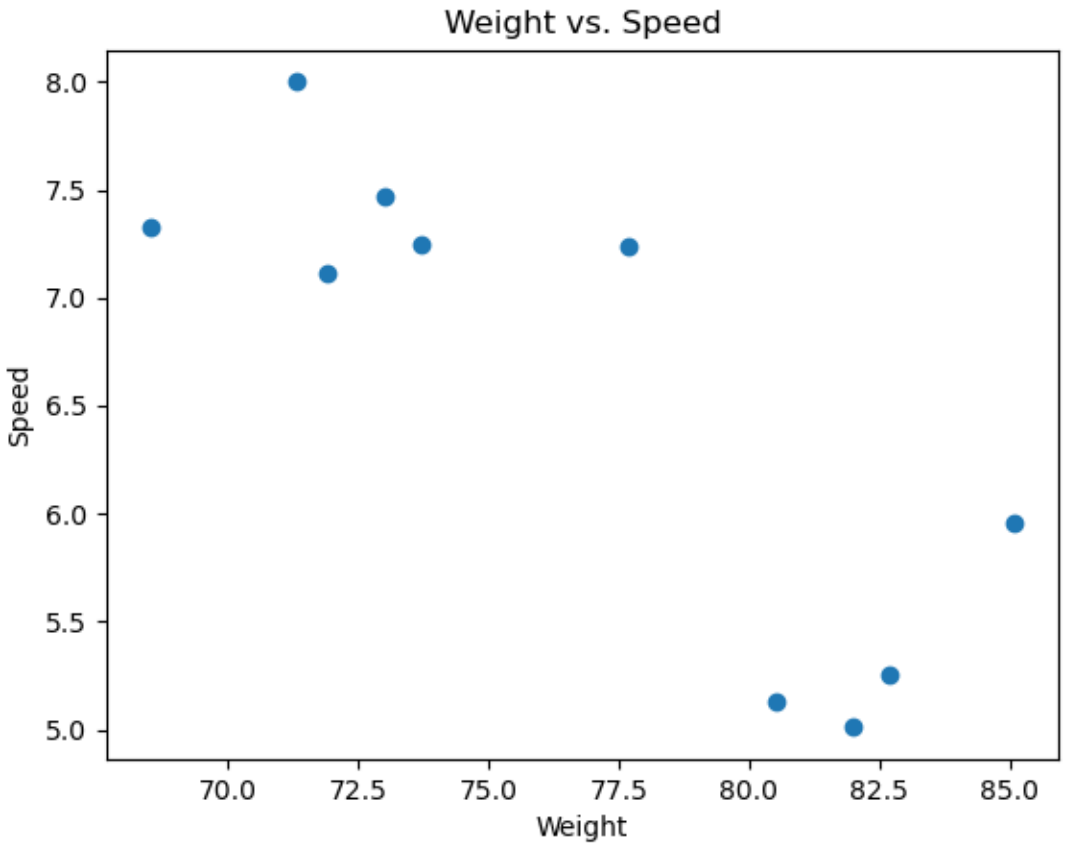

Figure 3.2-8

Scatter Showing the Correlation of Weight and Speed

Figure 3-2-8 illustrates the relationship between a player's weight and their speed. The graph demonstrates some negative correlation, indicating that as the player's weight increases, their speed tends to decrease as well.

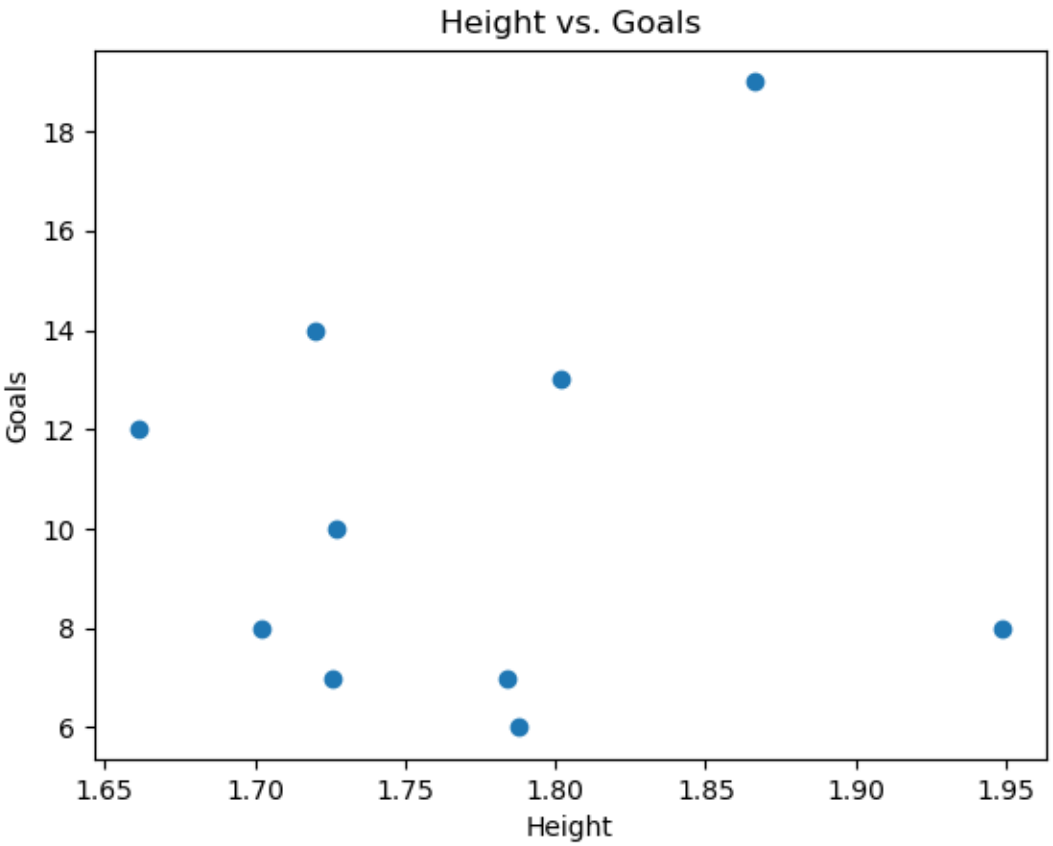

Figure 3.2-9

Scatter Showing the Correlation of Height and Goals

Figure 3-2-9 illustrates the relationship between a player's height and the number of goals they score. The graph demonstrates some positive correlation, indicating that the taller the player is, the more goals they tend to score.

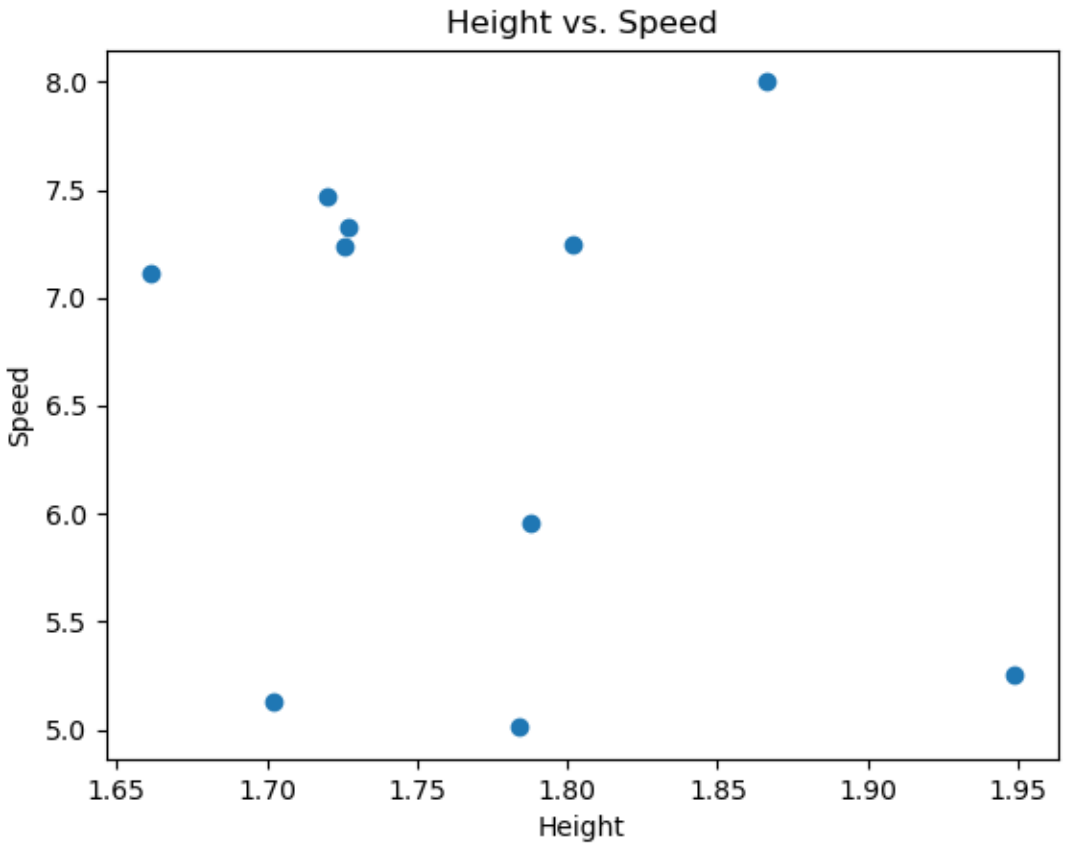

Figure 3.2-10

Scatter Showing the Correlation of Height and Speed

Figure 3-2-10 illustrates the relationship between a player's height and their speed. The graph demonstrates no correlation, indicating that the height of a player wouldn't affect their speed.

3.3 Matrices

$$\begin{array}{c} \\ weight \\ height \\ speed \\ goals \end{array} \begin{array}{c} \begin{array}{cccc} weight & height & speed & goals \end{array} \\ \begin{bmatrix} 32.4649 & 0.4584 & 5.8042 & 22.2378 \\ & 0.0073 & 0.0806 & 0.3423 \\ & & 1.2508 & 3.8611 \\ Symm & & & 16.7111 \end{bmatrix} \end{array}$$

Matrix 3.3-1

Variant-Covariant Matrix

Matrix 3.3-1 shows the Variant-Covariant Matrix indicating the relation between each 2 sets of single variate sample.

$$\begin{array}{c} \\ weight \\ height \\ speed \\ goals \end{array} \begin{array}{c} \begin{array}{cccc} weight & height & speed & goals \end{array} \\ \begin{bmatrix} 1.0000 & 0.3302 & -0.8374 & -0.7340 \\ & 1.0000 & -0.2076 & 0.1045 \\ & & 1.0000 & 0.7340 \\ Symm & & & 1.0000 \end{bmatrix} \end{array}$$

Matrix 3.3-2

Correlation Coefficient Matrix

Matrix 3.3-2 shows the Correlation Coefficient Matrix. Indicating a very powerful correlation between weight and speed, weight and goals, and speed and goals but with a bad solution. However, the correlation between weight and height, height and speed, and height and goals are low with a good solution.

4 Discussion

It is observed that the residuals A and B have identical values for variance, standard deviation, range, and mean standard deviation as observations A and B, respectively. This indicates that the observations at A and B were fairly accurate, since the residuals should show the noise and random fluctuations of data. However, that doesn't mean that the observations were perfect. Looking at the values of standard deviation of observations A and B, we can see that A seems to be more accurate and precise than B because of a significantly lower standard deviation. Though, another reason why the variance and standard deviations of residuals and observations are identical, is because the equation used to calculate the these values (equation (3) and equation (4)) are both using the value of the residuals regardless. Additionally, both observations also have two values outside the 95% confidence interval range, indicating potential outliers.

Additionally, by examining *Figure 3.2-5)*, *Figure 3.2-6)*, and *Matrix 3.3-2)* it becomes evident that the relationships between weight and speed, weight and goals, and speed and goals show stronger correlations compared to other combinations in the dataset. This suggests that lighter football players tend to be faster; a lighter player tends to score more goals; a faster player tends to score more goals.

4.1 Questions

The residual plot for A shows a left skew, which indicates fewer residuals in the negative portion of the histogram, considering it's a probability density histogram. Additionally, two observations fall outside the 95% confidence interval in both plots A and B. Apart from these outliers, no other errors are apparent. The residual histograms align with a normal distribution, and all values fall within their respective confidence intervals.

An outlier is noticeable in the residual plot of B, situated around 0.4. Its proximity to the other values suggests its outlier status. To address this, we can trim the values at the extremes of each observation to remove these outliers.

These correlation values indicate the strength and direction of relationships between different variables. For instance, the weight versus speed correlation shows a weak solution and a strong negative correlation. This implies that errors in data collection could significantly skew the results. Similarly, the correlation between speed and goals, as well as weight and goals, is 0.7340 and -0.7340, indicating a substantial correlation, albeit the weights and goals being negative.

When correlations are higher, like in these cases, it means that solutions are weaker, and more correction calculations are required to approximate values accurately. Conversely, lower correlations, such as weight and height or height and goals, require fewer correction equations to achieve a more approximate estimation.

5 Conclusion

In conclusion, the lab's goal was accomplished, demonstrating the our understanding and skill at using statistical equations to extract values for two tasks. This included multivariate and single-variate statistical problem solving, which resulted in the creation of statistical vectors and variance-covariance and correlation matrices. To help identify possible systematic mistakes and outliers, these computed values were then exported into Python for visualization. The precise calculation of all numbers, which was further supported by the visualizations, is proof that the program was correctly executed.

5.1 Learning Outcomes

Writing code for both simple and complex equations in C++, organizing it into different classes, and applying object-oriented programming (OOP) principles was a valuable experience. It allowed for the application of knowledge gained from previous courses while enhancing skills in code organization and report writing.

In future projects, I aim to prioritize consistency in the program structure and provide more detailed comments throughout the code. Additionally, I plan to adopt a practice of commenting as I write the program, ensuring thorough documentation from the start. This approach will help streamline the development process and avoid spending extra time on commenting after the code is completed.

6 References

[1] GeeksforGeeks. "Vector in C++ STL" [Online]. Available: https://www.geeksforgeeks.org/vector-in-cpp-stl/

[2] C++. "cmath Reference" [Online]. Available: https://cplusplus.com/reference/cmath/

[3] C++. "ifstream - Input file stream" [Online]. Available: https://cplusplus.com/reference/fstream/ifstream

[4] W3Schools. "SciPy Tutorial" [Online]. Available: https://www.w3schools.com/python/scipy/index.php

[5] W3Schools. "Matplotlib Pyplot Tutorial" [Online]. Available: https://www.w3schools.com/python/matplotlib_pyplot.asp

[6] W3Schools. "NumPy Tutorial" [Online]. Available: https://www.w3schools.com/python/numpy/numpy_intro.asp

[7] Stack Overflow. "Creating a bash script to compile a C program" [Online]. Available: https://stackoverflow.com/questions/29090372/creating-a-bash-script-to-compile-a-c

7 Appendix

7.1 Code

```
/*
    ********************************************************

    Written by Wesam Omran - 30133992
    ENGO 363: Estimation and Statistical Methods
    Lab 2

    Calculations.h - Header file for the Calculations class

    ********************************************************
*/

#ifndef CALCULATIONS_H
#define CALCULATIONS_H

#include <vector>
#include <string>

using namespace std;

vector<double> sort(vector<double> v);
double range(vector<double> v);
double median(vector<double> v);
double mean(vector<double> v);
vector<double> residuals(vector<double> v, double mean);
double variance(vector<double> v, double mean);
double standardDeviation(double varianceValue);
double meanSD(vector<double> v, double sd);
double sum(vector<double> v);
double weight(double meanSD);
double weightedMean(vector<double> weights, vector<double> means);
double weightedMeanSD(vector<double> weights, vector<double> means, double wm);
vector<vector<double>> transpose(vector<vector<double>> v);
double covariance(vector<double> v1, vector<double> v2);
vector<vector<double>> variantCovariant(vector<vector<double>> var_covar,
vector<vector<double>> res);
vector<vector<double>> correlationCoefficient(vector<vector<double>> corCoef,
vector<vector<double>> var_covar, vector<double> std);

#endif // CALCULATIONS_H
```

Code 7.1-1: Calculation.h File (Double click on object to view full code)

```cpp
/*
    ********************************************************
    Written by Wesam Omran - 30133992
    ENGO 363: Estimation and Statistical Methods
    Lab 2

    Calculations.cpp - Source file for the Calculations class
    This class is responsible for all calculations needed
    for tasks given in the lab

    ********************************************************
*/

#include "Calculations.h"
#include <iostream>
#include <vector>
#include <string>
#include <cmath>

using namespace std;

// Function responsible for sorting the vector given in ascending order
vector<double> sort(vector<double> v)
{
    int n = v.size();
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (v[j] > v[j + 1])
            {
                double temp = v[j];
                v[j] = v[j + 1];
                v[j + 1] = temp;
            }
        }
    }
    return v;
}
```

Code 7.1-2: Calculation.cpp File (Double click on object to view full code)

```
/*
    ********************************************************

    Written by Wesam Omran - 30133992
    ENGO 363: Estimation and Statistical Methods
    Lab 2

    ReadFile.cpp - Header file for the ReadFile class

    ********************************************************
*/

#ifndef FILEHANDLING_H
#define FILEHANDLING_H

#include <vector>
#include <string>

using namespace std;

void readFile(string s, vector<double> &observations);
vector<double> readFile(string s);
void writeToFile(string file, vector<string> keys, vector<double> &values);
void writeToFile(string file, vector<double> residuals);
void writeToFile(string file, vector<vector<double>> matrix);

#endif // FILEHANDLING_H
```

Code 7.1-3: FileHadnling.h File (Double click on object to view full code)

```
/*
    ********************************************************
    Written by Wesam Omran - 30133992
    ENGO 363: Estimation and Statistical Methods
    Lab 2

    FileHandling.cpp - Source file for the FileHandling class
    This class is responsible for reading from files and
    writing to files

    ********************************************************
*/

#include "FileHandling.h"
#include <iostream>
#include <vector>
#include <string>
#include <fstream>
#include <iomanip>
#include <cmath>

using namespace std;

// Function to read two-column files
void readFile(string file, vector<double> &observations)
{
    // Open the file
    ifstream infile(file);

    // Check if the file opened successfully
    if (!infile.is_open())
    {
        cout << "File failed to open" << endl;
        exit(1);
    }

    double value1, value2;
```

Code 7.1-4: FileHandling.cpp File (Double click on object to view full code)

```
/*
    ********************************************************
    Written by Wesam Omran - 30133992
    ENGO 363: Estimation and Statistical Methods
    Lab 2

    Main.cpp - Source file for the Main class
    Main class used to test, print out, and execute
    all computations needed for result

    ********************************************************
*/

#include "Calculations.h"
#include "FileHandling.h"
#include <iostream>
#include <vector>
#include <string>
#include <iomanip>

using namespace std;

int main()
{
    /*
        Vectors needed to write to files that will be used
        to be read by python file to generate the plots
    */
    vector<string> keys;
    vector<double> valuesA;
    vector<double> valuesB;

    // Load Observations A and B data into vectors
    string obsA = "obsA_2024.txt";
    string obsB = "obsB_2024.txt";
    vector<double> observations1, observations2;
    readFile(obsA, observations1);
    readFile(obsB, observations2);
    // Assuming data isn't sorted, sort the data
    observations1 = sort(observations1);
    observations2 = sort(observations2);
```

Code 7.1-5: Main.cpp File (Double click on object to view full code)

```
#!/bin/bash

# Compile Main.cpp
g++ -c Main.cpp -o Main.o

# Compile Calculations.cpp
g++ -c Calculations.cpp -o Calculations.o

# Compile ReadFile.cpp
g++ -c FileHandling.cpp -o FileHandling.o

# Link the object files
g++ Main.o Calculations.o FileHandling.o -o Main

# Run the program
./Main
```

Code 7.1-6: run.sh File (Double click on object to view full code)