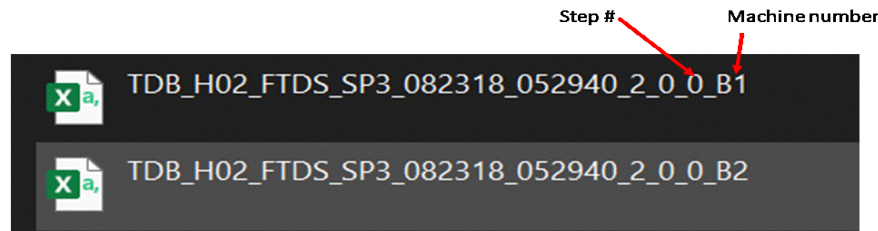


<b>Activity No. 3</b>	
<b>Automation of Statistical Analysis of Signals</b>	
<b>Course Code:</b> CPE027	<b>Program:</b> CpE
<b>Course Title:</b> Digital Signal Processing Applications	<b>Date Performed:</b> September 19, 2023
<b>Section:</b> CPE41S1	<b>Date Submitted:</b> September 25, 2023
<b>Name/s:</b> Cabanos, Renzo Landayan, Selwyn Charlz Angelo Marbebe, Jerome Musni, Christian Marc Ruiz, Sam Ryan	<b>Instructor:</b> Engr. Cris Paulo Hate
<b>1. Objective:</b>	
This activity aims to introduce the concept of statistical analysis for large-volume signals, such as the mean, standard deviation, noise, and error. This includes programmatically visualizing, calculating, and storing, using a high-level language.	
<b>2. Intended Learning Outcomes (ILOs):</b>	
After completion of this activity the students should be able to: Develop a program that calculates and visualizes the statistical properties of a given signal dataset.	
<b>3. Discussion :</b>	
<ol style="list-style-type: none"> <li>How is automation applied in signal processing? <ul style="list-style-type: none"> <li>Automation in signal processing entails efficiently analyzing and manipulating huge amounts of signal data using programming and computational approaches. It comprises operations like data pretreatment, statistical analysis, feature extraction, and visualization that are automated.</li> </ul> </li> <li>Why is automation crucial in signal processing and big data situations? <ul style="list-style-type: none"> <li>Automation is critical in signal processing and big data situations because it allows for the efficient and consistent analysis of large volumes of data, reduces the risk of human errors, speeds up processing, and allows for scalability to handle vast datasets that would be impractical to process manually. Automation also ensures reproducibility and frees up human resources for higher-level tasks like analyzing results and making data-driven decisions.</li> </ul> </li> </ol>	
<b>4. Resources:</b>	
<p>The activity will require the following software, tools and equipment:</p> <ul style="list-style-type: none"> <li>Personal Computer</li> <li>Google Colab</li> </ul>	
<b>5. Directions:</b>	
A sensor managed to sample data in 0 to 195 steps. You will be given access to a repository of each sample in the form of CSV files. Use only csv files coming from machine B1. There is a total of 196 sample files, each of those files contain 41,666 data points. Disregard the file labeled: Base Noise.	



1. Create a program that can calculate the mean, standard deviation, standard error, for **each** STEP, and consolidate it in a separate (new) csv file.

2. Also, create a program that is able to create and save images showing both the time-series plot, histogram, and the calculated values for each step. This would look similarly like the image below.

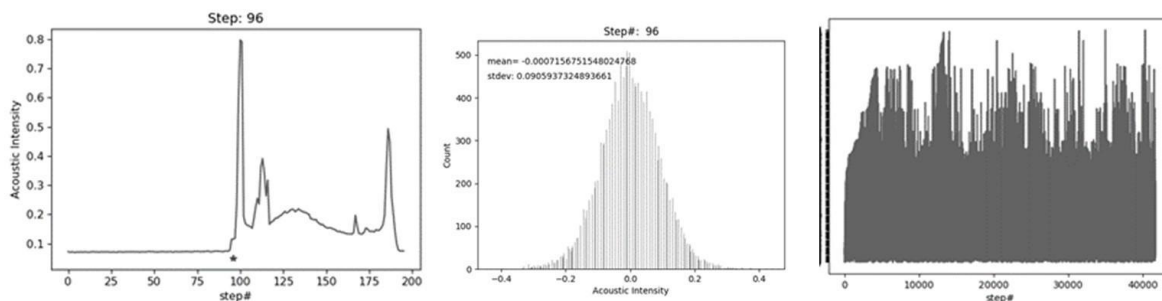


Figure 1 - (Left to Right) Plotting of Means Per Step, Histogram of one (1) step, Time-Series of one(1) step.

## 6. Procedures

### Step 1: Accessing the Dataset

We created a shortcut of the signal dataset we wanted to analyze in our own Google Drive so we could have access to it.

### Step 2: Setting Up Google Colab

We created a new Colab notebook and mounted our Google Drive folders.

### Step 3: Defining the Directory and Filtering Files

We defined the directory path where our dataset was located within Google Drive and used code to filter and select specific CSV files that end with B1 from the dataset.

### Step 4: Performing Statistical Analysis

We utilized Python libraries such as Pandas, NumPy, and Matplotlib to perform statistical analysis on the selected signal data. We calculated the mean and standard deviation.

### Step 5: Visualizing the Data

We created visualizations such as time series plots and histograms for the signal data using Matplotlib.

### Step 6: Storing Results

We created an output directory within Google Drive to store the results and visualizations.

## 7. Results(sample)

... > Dataset > output ▾

Type ▾ People ▾ Modified ▾




































Name ▾	Owner	Last modified ▾	File size	
 step_statistics.csv	 me	1:35 PM me	13 KB	    
 mean_values.png	 me	1:35 PM me	54 KB	
 195_timeseries.png	 me	1:32 PM me	48 KB	
 195_histogram.png	 me	1:32 PM me	18 KB	
 194_timeseries.png	 me	1:32 PM me	47 KB	
 194_histogram.png	 me	1:32 PM me	17 KB	
 193_timeseries.png	 me	1:32 PM me	49 KB	
 193_histogram.png	 me	1:32 PM me	17 KB	
 192_timeseries.png	 me	1:32 PM me	48 KB	
 192_histogram.png	 me	1:32 PM me	18 KB	

Figure 7.1: Outputs of the csv, images file in the directory

←  step\_statistics.csv

	A	B	C	D
1	Step Number	Mean	Std Deviation	Std Error
2	0	-0.0002132241396	0.06477799647	0.0003173486244
3	1	-0.0006970991696	0.06384584152	0.0003127819701
4	2	-0.0002491454423	0.06467102872	0.0003168245776
5	3	-0.0002141049057	0.06420572238	0.0003145450332
6	4	-0.0003065058801	0.0639037275	0.0003105506201
7	5	-0.0002605727692	0.06416536736	0.0003143473332
8	6	-0.0002534805357	0.06385756919	0.0003128394242
9	7	-0.0002444568953	0.06438543241	0.000315425436
10	8	-0.0002723492536	0.06375169637	0.0003123207513
11	9	-0.0001853919263	0.06493378149	0.0003181118084
12	10	-0.0002670141602	0.06454328187	0.0003161987434
13	11	-0.0002365467287	0.06392665427	0.0003131778733
14	12	-0.000685499952	0.06304775169	0.0003088721131
15	13	-0.0006750891931	0.06440091828	0.0003155013015
16	14	-0.0006003611098	0.06423587866	0.0003146927693
17	15	-0.0006712042913	0.06495127995	0.0003181975336
18	16	-0.0006978049489	0.06413321785	0.0003141898321
19	17	-0.0006852066433	0.06492900898	0.0003180884277
20	18	-0.0001741421783	0.06353499192	0.0003112591121
21	19	-0.0006863809101	0.06411805971	0.0003141155721
22	20	-0.0006857331397	0.06395756321	0.0003133292968
23	21	-0.0007260449527	0.06473466898	0.0003171363524
24	22	-0.0007122166035	0.06498385422	0.0003183571156
25	23	-0.0006932908847	0.06457691799	0.0003163635274
26	24	-0.0007562256516	0.06447914502	0.0003158845358
27	25	-0.0007824139106	0.06325376215	0.0003098813622
28	26	-0.0006684501032	0.06376340813	0.0003123781274
29	27	-0.0007156162339	0.06410794365	0.0003140660134
30	28	-0.0003812160515	0.06277667675	0.000307544112
31	29	-0.0008321043777	0.06444385433	0.0003157116461

Figure 7.2: CSV File Output

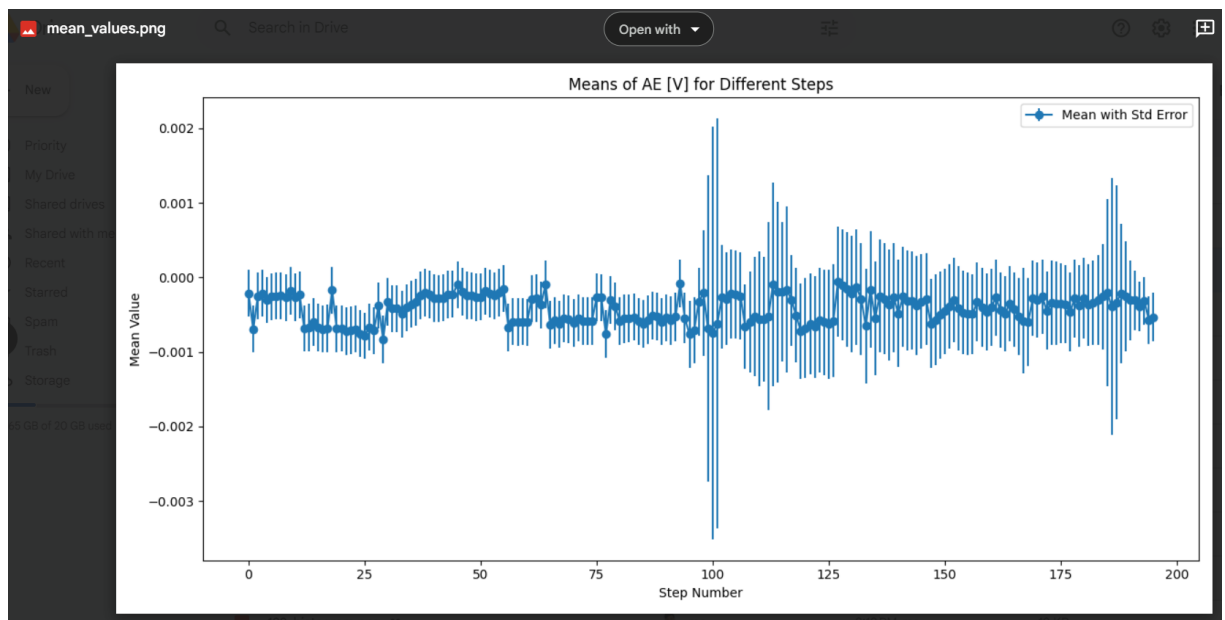


Figure 7.3: Plotting of Means Per Step

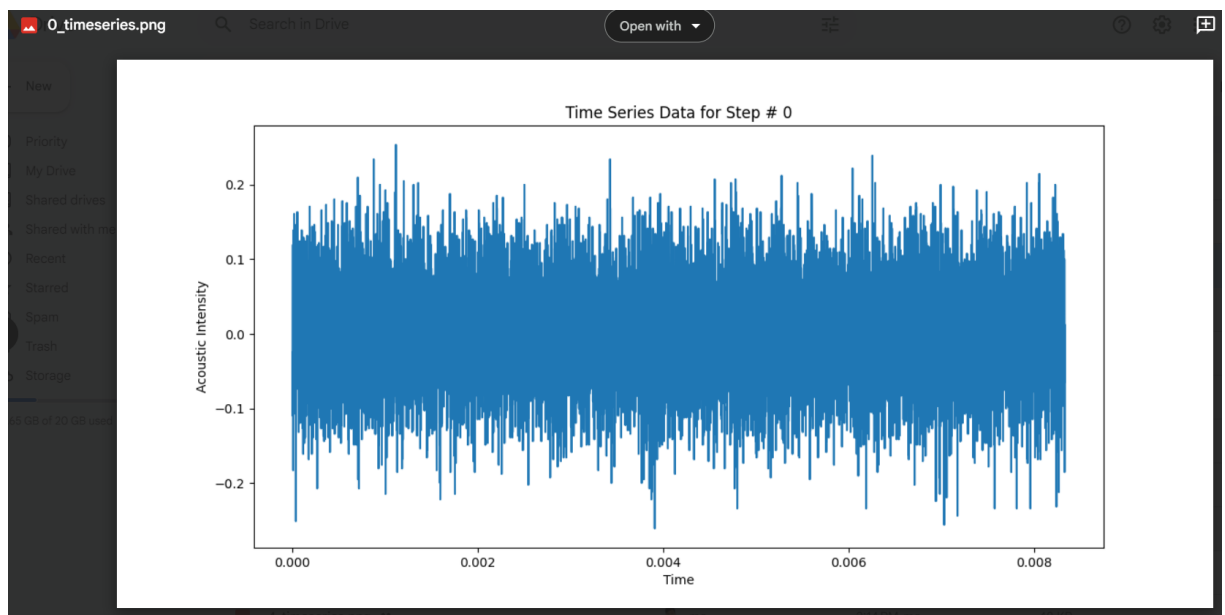


Figure 7.4: Sample Time-Series of one(1) step

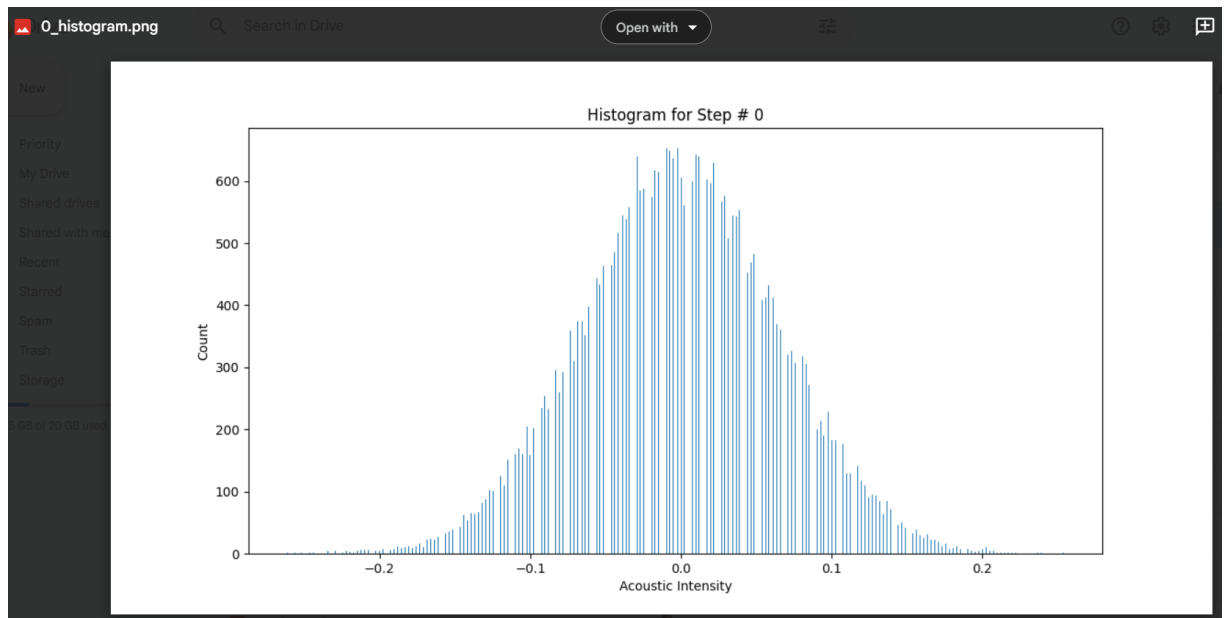


Figure 7.5: Sample Histogram of one (1) step

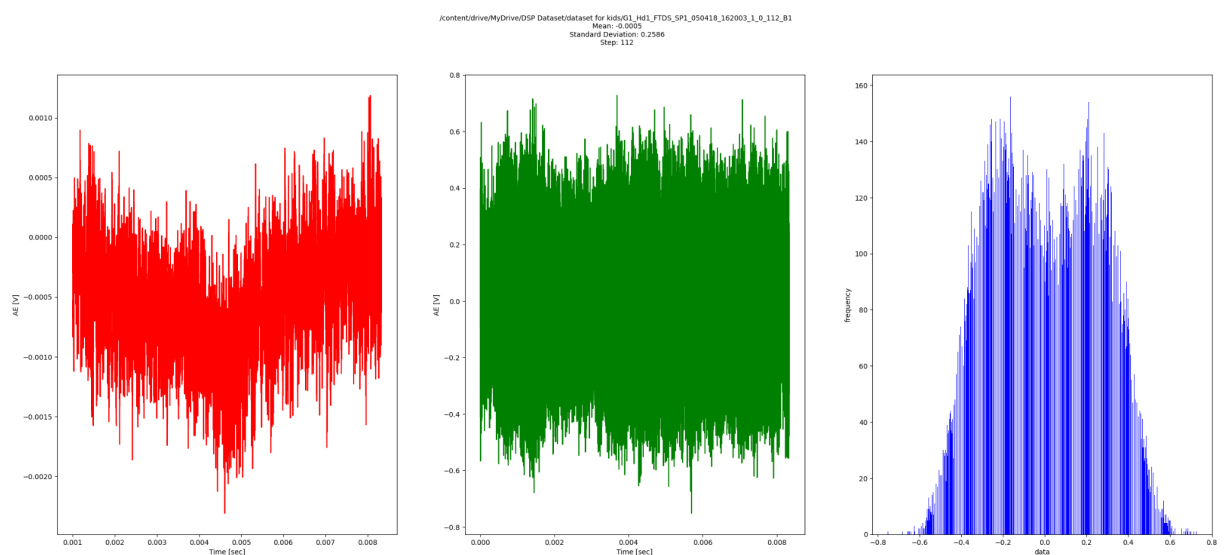


Figure 7.6: Sample Mean, Time-Series and Histogram of one (1) step

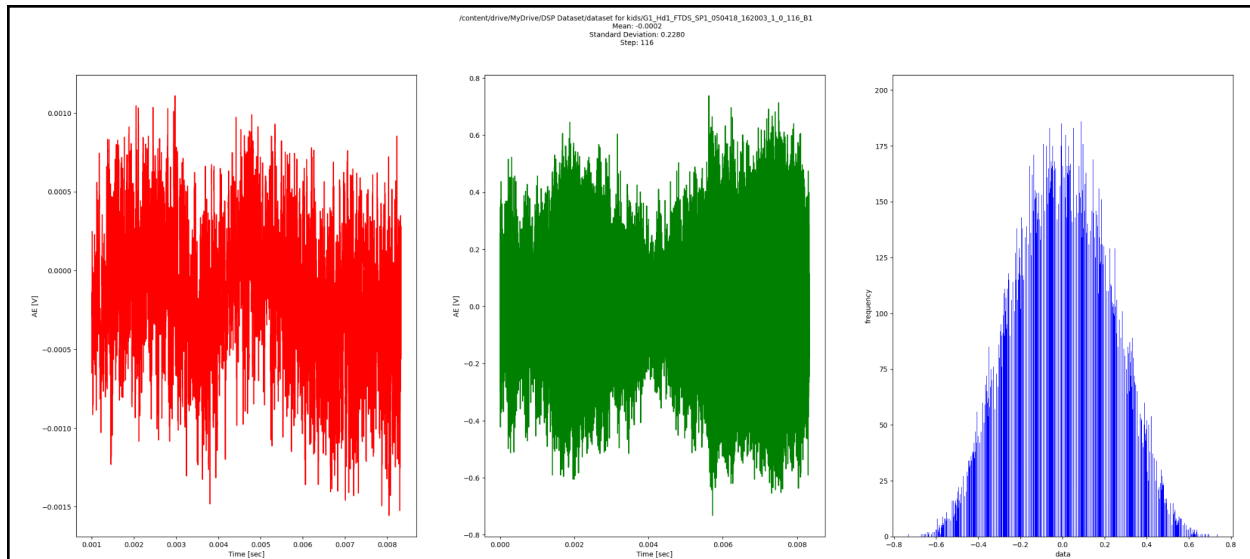


Figure 7.7: Sample Mean, Time-Series and Histogram of one (1) step

#### IPYNB File:

<https://colab.research.google.com/drive/1nt1eHwLOVdQDb4EAveXOWmtv5xzKOSE3?usp=sharing>  
<https://colab.research.google.com/drive/1NGZ9u6MFsQEQtzclBcnp0f6qpoJWEAnG>

#### Output Directory:

[https://drive.google.com/drive/folders/1y8HHc8TCko0p45o2oNnw39tB3Bxjc\\_tN?usp=sharing](https://drive.google.com/drive/folders/1y8HHc8TCko0p45o2oNnw39tB3Bxjc_tN?usp=sharing)

## 8. Data Analysis

According to the output of the given dataset, there is a majority output of a bell shaped curve for all datasets until output 195. This only means that there is a normal distribution of data for all. In some instances, there are bimodal outputs wherein you see two peaks in one visualization graph. This means that there are two modes that the dataset has detected. When it comes to the time series, it can be seen that as voltages show different values the ohms stays in one line meaning there is no change at all and remains at the value of 0.

## 9. Summary and Conclusions

In this group activity, our objective was to further explore statistical analysis for large-volume signals, focusing on essential concepts like mean, standard deviation, noise, and error. We successfully accomplished this objective through implementing our knowledge in programming, enabling us to visualize, calculate, and store these statistical properties programmatically. As a result, our intended learning outcomes were achieved. We now possess the skills and knowledge required to develop programs that can effectively calculate and visually represent the statistical properties of diverse signal datasets. This activity has not only expanded our understanding of statistical analysis but also equipped us with valuable tools for data-driven decision-making and problem-solving in various fields.

## 10. Learnings and Contributions of each member

### **Cabanos**

For this activity, I tried to help a little bit with the code and focused more on the documentation. With this activity, I was able to grasp the concept of time series and analyze how data distribution would work for currents like this. We can see that the guide values we have are the mean, standard deviation and error. As the values show, it is easier to understand the visualization part of the code since we now know how the dataset would be interpreted.

### **Landayan**

I contributed to the code of this activity, such as getting the B1 CSV files, making a script that automatically visualizes all the B1 CSV files in time series and histograms, and also getting the mean and standard deviation. In this activity, I learned to statistically analyze the large volume of signals, such as the mean standard deviation, noise, and error. In dealing with large volumes of signals and datasets, you have to write your code in a loop so that you can save time in manually visualizing and computing the statistics of each file.

### **Marbebe**

I majorly contributed to the development of the code that selects only the files that have B1 in their file names. In addition, I also create a visualization code for each file or step using Matplotlib and Numpy. Consequently, throughout this activity, I learned how to automate statistical analysis of a large dataset, where each file contains thousands of rows. Visualizing the time series of each signal helps me gain insight into how each step affects its nature. Also, with the help of statistical analysis such as mean, standard deviation, and error, I was able to determine the commonalities and differences of each signal per step.

### **Musni**

In this activity, I was able to code and contribute to documentation. Through this activity, I gained a deeper understanding of data analysis, especially how to programmatically calculate and visualize statistical properties such as mean, standard deviation, noise, and error. I also learned how to create time-series plots and histograms to explore data trends and distributions on a given dataset. Overall, this activity successfully met its objectives, allowed me to develop my ability to analyze and visualize diverse signal datasets and improved my statistical analysis capabilities.

### **Ruiz**

I actively participated in this activity by using my programming skills to evaluate large datasets using statistical methods such as averages, variance, and error handling. I used Python to generate charts and save data automatically. This experience improved my understanding of statistics and provided me with practical skills for working with real-world data. I also learned how to use Google Colab, a tool for online coding with others that made our job easier and faster. Overall, I enhanced my ability to write algorithms that evaluate signal data and gained a better understanding of these crucial statistical concepts.