# Practice 1: SOFTWARE-DEFINED RADIO PROGRAMMING (GNU RADIO)

Hailen Andrea Chacón López - 2210409
Laura Tatiana Monsalve Ríos - 2211551
Daniel Jeshua Morelos Villamizar - 2200515

https://github.com/DanielMorelos/CommII_A1_G6

School of Electrical, Electronic, and Telecommunications Engineering
Industrial University of Santander

February 28, 2025

**Abstract**

In this practical case, several signal processing blocks were implemented, such as an accumulator, a differential block, and a time-averaging block, in addition to exploring the use of the accumulator as a simple low-pass filter. To this end, digital communications concepts were reinforced by generating, manipulating, and analyzing discrete signals, bringing theory into a practical environment. The accumulator acts as a discrete integrator; the differential block emulates the derivative; and the average over time allows evaluating the average level or general trend of the samples.

**Key words: GNU Radio, Accumulator, Differentiator**

## 1 Introduction

Signal processing is a crucial aspect of modern communications engineering, providing the foundation for efficient transmission, manipulation, and analysis of data. In this practical exercise, we focused on the implementation of several key signal processing blocks, such as an accumulator, a differentiator, and a time-averaging block [1]. In addition, we explored the application of the accumulator as a simple low-pass filter.

The objectives of this project were to develop abilities that enable interaction with the programming of functions in GNU Radio,identify the fundamental aspects of real-time systems and software-defined radio, generate functions from implementation code blocks and evaluate the results with other blocks, and use the implemented blocks to produce a specific application for real signals.

The accumulator block was designed to act as a discrete integrator, adding each incoming sample with all previous samples. The differentiator block emulated the derivative by calculating the difference between consecutive samples. The time-averaging block allowed us to evaluate the average level or general trend of the samples over time.

The following sections detail the methodology, results, and conclusions derived from this practical exercise.

## 2 Process

1. We started this practice by establishing a new repository named `Practica_1` on GitHub. This repository served as the storage location for all the files related to the project.
2. We developed a Python block that will be essential for subsequent phases.
3. New branches were created from `Practica_1` so each member of the group could contribute individually to the project.
4. We developed both a derivative block, an accumulator block and a time averaging block by making use of three custom Python blocks.
5. Afterwards, we made a low-pass filter making use of the derivate block as a practical application of the concepts we learned during this project.
6. We used GitHub commands, such as `git add`, `git commit -m`, `git push`, etc. as well as our tokens, to send our files to the repository.

## 3 Results

The accumulator block, in the discrete domain, acts as an integrator, as shown in Figure 1 because at each sampling instant it adds the current sample to all previous samples, thus reproducing the integration function seen in the continuous domain.

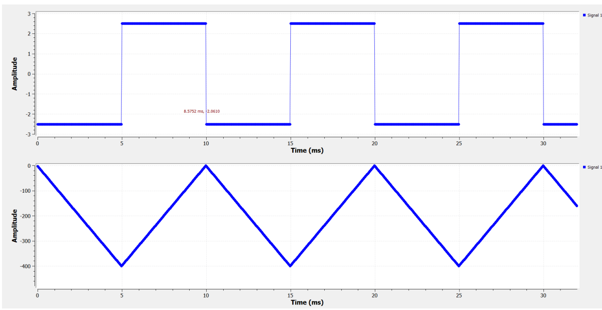


Fig. 1: Results obtained from the accumulator block.

The differentiator block calculates the difference between consecutive values of the input signal, reproducing the concept of derivative in the discrete domain. At each sampling instant, the previous sample is subtracted from the current sample, resulting in an output that reflects the rate of change of the signal, as seen in Figure 2

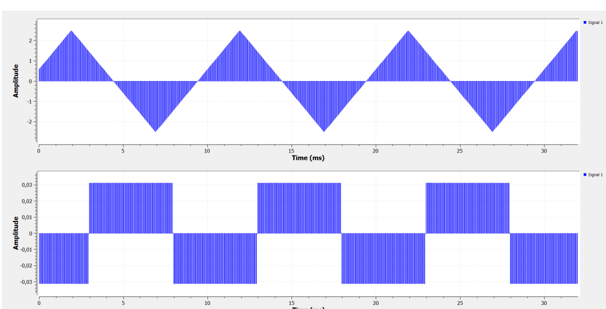


Fig. 2: Results obtained from the differentiator block.

The average power allows determining the amount of energy that a signal carries over time. RMS is an effective measure to evaluate the true amplitude of a signal. The standard deviation indicates the degree of variability of the signal, helping to identify the presence of noise. The arithmetic mean represents the average value of the signal. The root mean square is closely related to the energy of the signal.

| **Parameter** | **Value** |
|---|---|
| RMS | 1.1414214 |
| Quadratic Mean | 2 |
| Average Power | 2 |
| Standard Deviation | 1.247234 |
| Mean | 0.666667 |

Tab. 1: Statistical results of the signal in GNU RADIO

The use of the accumulator block was intended to reduce the presence of high-frequency components in the signal, aligning with the principles of a low-pass filter. By performing an integration process, the block progressively smooths the waveform by summing each sample with the previous ones. This process minimizes rapid variations, which are typical of high-frequency components. However, unlike a traditional FIR filter with a well-defined cutoff frequency, this approach does not strictly eliminate high-frequency signals and may introduce drift when the input contains a DC offset or asymmetrical noise. Despite these limitations, the accumulator serves as a straightforward method for reducing high-frequency noise to some extent, making it functionally similar to a basic low-pass filter.

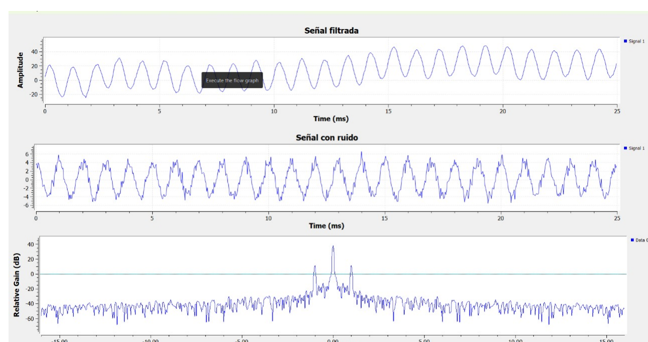


Fig. 3: Results of the application of an accumulator.

## 4 Conclusions

The code provided by the guide book was fixed, since when it was implemented, it was noticed that the code blocks didn't fulfill their determined function; therefore, as a group, a space was taken to identify the logic of the error in order to obtain blocks that did fulfill their function (to accumulate and derive correctly).

It was noted that in GNU Radio, signals are handled as streams represented by arrays rather than being processed sample by sample. Because of this, the variability in the dimension of these arrays allows for efficient real-time processing. Thanks to this structure, it's possible to develop systems that modulate signals or apply filters such as the Figure 3 dynamically, optimizing signal processing in communications.

With the modified blocks working correctly, a low-pass filter was implemented using what was learned in class. A signal with added noise was analyzed, and its behavior was reviewed before and after modulation and after applying the filters.

The analysis of a signal through metrics such as average power, RMS, standard deviation, arithmetic mean and root mean square helps us to better understand its behavior and characteristics in different applications.

## References

[1] H. O. Boada and O. M. R. Torres, *Comunicaciones Digitales basadas en radio definida por software*, primera ed. Bucaramanga, Colombia: Editorial UIS, 2019.