

# Práctica 1: Programación en Radio Definida por Software (GNU Radio)

## I. Introducción

## II. Acumulador

## III. Derivador

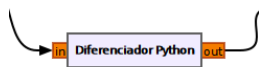


Figura 1: Bloque diferencial

Un bloque diferencial calcula la derivada discreta de una señal, es decir, mide el cambio entre una muestra y la anterior. Matemáticamente se expresa como:

$$y[n] = \frac{x[n] - x[n-1]}{T_s} \quad (1)$$

donde:

- $x[n]$  es la muestra actual de la señal.
- $x[n-1]$  es la muestra anterior.
- $T_s$  es el período de muestreo.
- $y[n]$  es la salida diferencial.

Si la señal es constante ( $x[n] = x[n-1]$ ), la salida es cero. Si hay un cambio, la salida refleja la magnitud y dirección de dicho cambio.

- Configuración del bloque

Este bloque de código para GNU Radio implementa un proceso síncrono que calcula la variación entre muestras consecutivas de una señal, asegurando que no existan saltos o errores cuando los datos se procesan por ráfagas. Para lograr esto, utiliza una variable de memoria que almacena el último valor del paquete anterior y lo inserta al inicio del nuevo grupo de datos, permitiendo que la función de resta sea continua y fluida. Finalmente, actualiza esta memoria con el dato más reciente para estar listo para el siguiente ciclo, manteniendo así la integridad de la señal en todo momento

## A. Configuración del bloque

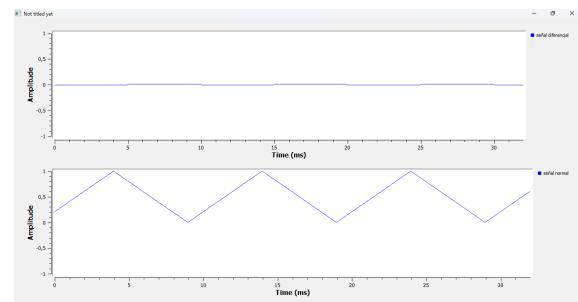


Figura 2: Diferenciador / Derivador

Este bloque de Python para GNU Radio implementa un bloque síncrono diseñado para procesar flujos de datos tipo `float32` de manera continua. Su lógica principal reside en el método `work`, donde utiliza una variable de estado llamada `self.last_sample` para preservar la continuidad entre ráfagas de datos; esto evita errores de cálculo en los bordes de cada paquete al insertar el último valor del ciclo anterior al inicio del vector actual mediante `np.insert`. Finalmente, emplea la función `np.diff` de NumPy para realizar la resta de elementos adyacentes de forma vectorizada, actualiza la memoria del bloque con la última muestra procesada (`in0[-1]`) y devuelve la longitud del vector de salida para mantener la sincronía del flujo en el programa.

## IV. Bloque estadístico

## V. Aplicación

## VI. Conclusiones

## Referencias

- [1] W. Stallings, *Comunicaciones y Redes de Computadores*, 7ma ed. Prentice Hall, 2004.
- [2] GNU Radio Project, "Documentation", [En línea]. Disponible en: <https://www.gnuradio.org/>.