Práctica 1: PROGRAMACIÓN EN RADIO DEFINIDA POR SOFTWARE (GNURADIO)

Anderson David Hernandez Rodriguez - 2202930
Karen Daniela Rangel Uribe - 2191709
Sharon Catalina Vargas Cortes - 2211259

https://github.com/COM2-GRUPO2/COM2$_B$1$_G$2

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones
Universidad Industrial de Santander

27 de febrero 2025

Resumen

In this practice, we examined the use of GNU Radio for software-defined radio (SDR) by developing and testing blocks for signal accumulation, differentiation, and statistical analysis. Our goal was to reconstruct signals affected by Gaussian noise, which allowed us to identify limitations in signal precision due to noise amplification and suboptimal initial conditions. Although the developed blocks performed their intended functions, the recovered signals exhibited discrepancies compared to the originals, highlighting the need for more robust algorithms and improved noise management strategies. Future enhancements include implementing smoothing filters, optimizing block definitions, and utilizing vector-based signal representations to improve accuracy and signal reconstruction quality..

1. Introducción

La radio definida por software (SDR, por sus siglas en inglés) se ha convertido en una tecnología fundamental en el campo de las telecomunicaciones gracias a su versatilidad y capacidad para adaptarse a distintos estándares y protocolos de comunicación, son configuradas por software, lo que facilita la actualización y optimización de los sistemas sin necesidad de modificar los componentes físicos. GNU Radio es una herramienta clave para la simulación y análisis de señales en SDR. Mediante la programación de bloques funcionales en Python, es posible implementar procesos como la acumulación, la diferenciación y el análisis estadístico de señales, permitiendo evaluar su comportamiento y recuperación en distintos escenarios. Además, el uso de plataformas colaborativas como GitHub favorece el desarrollo conjunto de estos sistemas con un enfoque práctico para el diseño y validación de arquitecturas de comunicación basadas en SDR.[1]

2. Metodología

Inicialmente se creó la rama práctica 1 en GitHub para trabajar en el informe, en esta rama se crearon dos carpetas: GNURadio (para guardar archivos relacionados al programa) e Informe. Posteriormente se ingresó a GNURADIO y se realizaron los distintos diagramas de bloques planteados en la guía (acumulador, diferenciador y el bloque para mostrar la estadística vista en clase y se crearon ramas adicionales correspondientes a cada integrante del grupo con el objetivo de que cada uno pudiera realizar el trabajo independiente.

2.1. Bloque acumulador

Utilizando el libro guía como referencia, se obtuvo el código del bloque acumulador. Posteriormente, este código fue implementado en un bloque de Python dentro de GNU Radio. Se realizaron modificaciones en algunas líneas de código con el objetivo de analizar las gráficas generadas por la aplicación y determinar si los resultados eran correctos o si era necesario realizar ajustes para obtener la respuesta esperada.

2.2. Bloque diferenciador

Siguiendo el mismo procedimiento que con el bloque acumulador, se obtuvo el código del bloque diferenciador y se implementó en un bloque de Python dentro de

GNU Radio. Se realizaron modificaciones en algunas líneas de código para observar las gráficas generadas y analizar si los resultados eran correctos o si era necesario hacer ajustes para obtener la respuesta adecuada.

2.3. Bloque estadístico

Se revisó el código de un Python Block desarrollado para las prácticas de comunicaciones digitales, cuyo objetivo era calcular y entregar simultáneamente varios promedios de tiempo de una señal de entrada, incluyendo la media, la media cuadrática, el valor RMS, la potencia promedio y la desviación estándar. El bloque fue diseñado con una entrada y cinco salidas, cada una correspondiente a uno de estos parámetros, y todo el código se implementó dentro de un solo Python Block. Además, las salidas fueron asignadas utilizando la estructura output items para distribuir correctamente los valores calculados.

2.4. Aplicación

Después de implementar un Python Block para mostrar las estadísticas estudiadas en clase (aplicando los conceptos aprendidos) se propuso una posible aplicación basada en estos principios y se recreó una visualización de los resultados obtenidos, permitiendo analizar su comportamiento y utilidad en un contexto práctico.

La aplicación realizada permitió analizar el comportamiento de una señal en presencia de ruido, utilizando un acumulador con límite configurable. La señal principal era una onda cosenoidal generada con una frecuencia de 500 Hz, mientras que el ruido se modeló con una señal gaussiana de baja amplitud. Ambas señales se combinaron para simular una medición real afectada por interferencias.

Se hizo uso del bloque acumulador que realizó la acumulación progresiva de la señal combinada, permitiendo observar cómo se incrementaba su valor a lo largo del tiempo. Para evitar que la amplitud de la señal se amplificara de forma indefinida, se agregó el bloque Multiply Const, que multiplicó la señal acumulada por 0.1, lo cual permitió recuperar la amplitud original de la señal, facilitando su análisis.

La aplicación realizada resultó útil en aplicaciones como la metrología, para medir energía acumulada, y en el procesamiento de señales digitales, para realizar integraciones o filtrado de señales ruidosas.

3. Resultados

En primer lugar, se evaluó el bloque Acumulador empleando un vector de tres datos que se repetía de manera cíclica. El comportamiento esperado de este proceso es la generación de una recta con pendiente positiva, debido al incremento continuo del valor acumulado. Sin embargo, al completarse el ciclo del vector y reiniciarse, la recta también se restablece, lo que da lugar a un comportamiento característico de una onda de dientes de sierra con pendiente positiva, tal como se observa en la Figura 7.

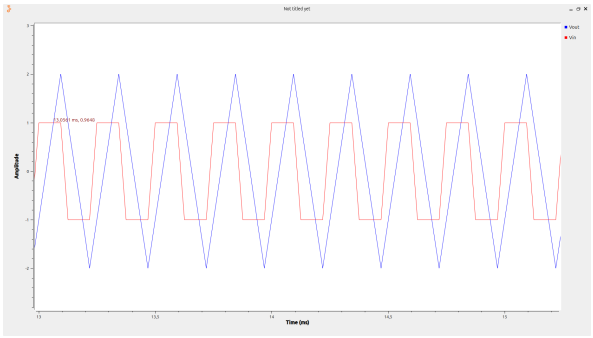

figura 1 .Simulación-verificación del bloque acumulador.

En la Figura 2 se muestra la respuesta del bloque Diferenciador ante una señal de entrada periódica. La señal de entrada (Vin, en rojo) corresponde a una onda cuadrada, mientras que la señal de salida (Vout, en azul) representa la derivada de la entrada.
Como se observa, la salida presenta picos positivos y negativos en los instantes en los que la señal de entrada cambia abruptamente de nivel. Este comportamiento es característico de un diferenciador, ya que su función es calcular la tasa de cambio de la señal de entrada.
En este caso, los picos corresponden a los momentos en los que la onda cuadrada transita entre sus valores máximos y mínimos, evidenciando cómo el bloque responde a variaciones bruscas de la señal de entrada.

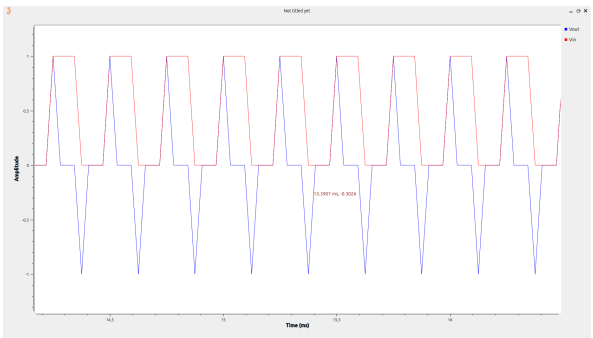

figura 2 .Simulación-verificación del bloque

diferenciador.

Además de los bloques Acumulador y Diferenciador, se empleó un bloque estadístico denominado "Promedio de tiempos". La función principal de este bloque es calcular y proporcionar métricas estadísticas de la señal de entrada, tales como la media, la media cuadrática, el valor RMS, la potencia promedio y la desviación estándar.

Para verificar el correcto cálculo de estos valores, se implementó el esquema mostrado en la Figura 5, en el cual se utilizó una onda cuadrada periódica de amplitud 1 y sin offset como señal de entrada. A partir de esta configuración, se analizaron las estadísticas obtenidas, cuyos resultados se presentan en la Figura 3.

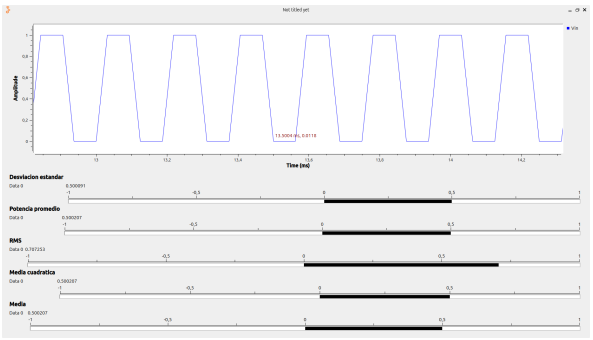

figura 3 .Simulación-verificación del bloque Promedio de tiempos.

A partir de los cálculos obtenidos y su comparación con los resultados teóricos, se confirmó el correcto funcionamiento del bloque Promedio de tiempos y sus métricas estadísticas. Con base en este análisis, se planteó el desarrollo de un sistema para la recuperación de señales, el cual permite comparar las métricas estadísticas antes y después del proceso de reconstrucción.

En este esquema se integran los bloques previamente analizados y, adicionalmente, se incorporó el bloque "Noise Source"para introducir ruido gaussiano en la señal de entrada, simulando así condiciones más realistas.

El diagrama representa un flujo de procesamiento de señales en GNU Radio, donde una señal cosenoidal de 500 Hz y una tasa de muestreo de 32 kHz se combina con ruido gaussiano mediante un bloque de suma. Posteriormente, la señal resultante es escalada por una constante de 100m y visualizada en diferentes etapas a través de bloques QT GUI Time Sink. Este diseño permite analizar el efecto del ruido en una señal y su procesamiento, siendo útil en aplicaciones de telecomunicaciones y control.

Posteriormente, la señal resultante es procesada por el bloque e-Acum, que acumula los valores en el tiempo. Luego, el bloque Multiply Const escala la señal mediante un factor de 100m antes de ser visualizada en otro QT GUI Time Sink.

Este diseño permite evaluar el efecto del ruido en el procesamiento de señales y analizar su evolución tras aplicar operaciones de acumulación y escalamiento.

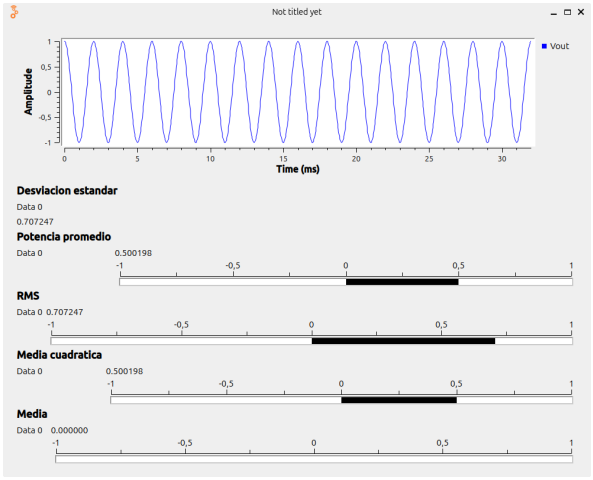

figura 4 .Señal original.

La Figura 4 muestra la representación gráfica de una señal senoidal junto con sus métricas estadísticas obtenidas mediante el bloque Promedio de tiempos en GNU Radio.

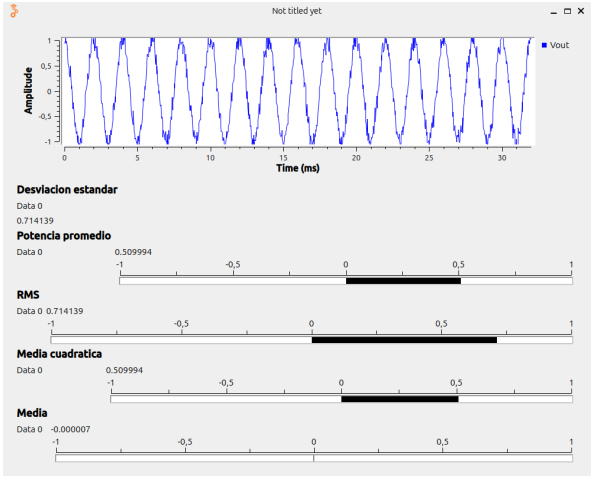

figura 5 .Señal ruido gaussiano.

La Figura 5 muestra la representación gráfica de una señal senoidal alterada por ruido, junto con sus métricas estadísticas obtenidas mediante el bloque Promedio de tiempos en GNU Radio. Estos valores reflejan la influencia del ruido en la señal original, observándose ligeras variaciones con respecto a los valores teóricos esperados para una onda senoidal pura sin componente DC. La correcta obtención de estos datos valida el adecuado

funcionamiento del bloque de análisis estadístico en presencia de ruido.

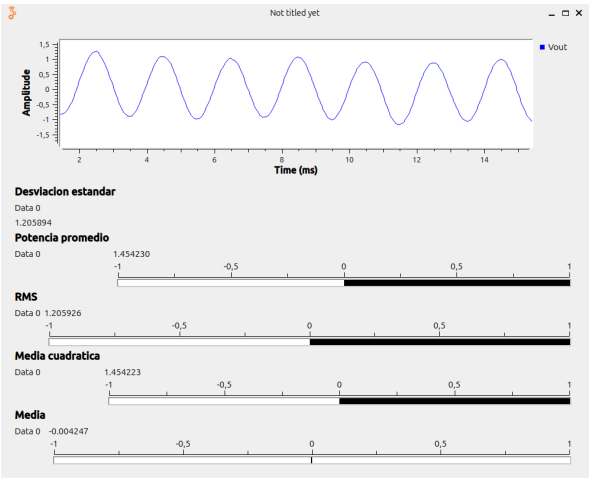

figura 6 .Señal filtrada.

La Figura 6 muestra la representación gráfica de una señal filtrada en función del tiempo y sus métricas estadísticas derivadas. Estos valores permiten evaluar la estabilidad y variabilidad de la señal. La desviación estándar y el valor RMS son mayores a 1, lo que indica que la señal tiene una amplitud significativa y posiblemente contiene una componente de ruido o modulación en su forma.

Este análisis confirma que el bloque "Promedio de tiemposrealiza correctamente el cálculo de métricas estadísticas, proporcionando información clave para evaluar la calidad de la señal procesada.

En la siguiente tabla se presentan los valores de las métricas estadísticas obtenidas para las distintas señales analizadas. Esta comparación permite evaluar el impacto de las modificaciones en la señal, como la adición de ruido o la transformación de la forma de onda.

A partir de los resultados, se observa que la desviación estándar y el valor RMS varían en función de la amplitud y la forma de la señal. Asimismo, la potencia promedio y la media cuadrática reflejan el nivel de energía de la señal en cada caso. La media, en la mayoría de los casos, se mantiene cercana a cero, lo que indica una distribución simétrica de la señal en torno al eje horizontal.
Estos resultados confirman la correcta implementación del bloque "Promedio de tiempos", el cual calcula con precisión las métricas estadísticas y permite un análisis detallado de las señales procesadas.

Estadistica	*Original*	*Ruidosa*	*Filtrada*
DE	0.707247	0.714139	0.406
PP	0.500198	0.509994	1.017
VRMS	0.707247	0.714139	2.08
MC	0.500198	0.509994	2.08
M	0.000000	-0.000007	2.08

Tabla 1. Métricas estadísticas.

4. Conclusiones

- La adición de ruido gaussiano en la señal permitió evaluar la variación de las métricas estadísticas, evidenciando cambios en la desviación estándar y el valor RMS, lo que demuestra la capacidad del sistema para analizar señales en condiciones más realistas.
- La práctica confirmó la utilidad de los bloques diseñados en aplicaciones de procesamiento de señales y metrología. Sin embargo, se identificó la oportunidad de optimizar los algoritmos para una mejor recuperación de señales, lo que podría lograrse mediante el uso de técnicas avanzadas como la representación vectorial de señales y el ajuste de parámetros de los bloques existentes.

Referencias

[1] Wikipedia contributors, “Radio definida por software,” 2025, [Accedido: 25-feb-2025]. [Online]. Available: https://es.wikipedia.org/wiki/Radio_definida_por_software