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

Juan Pablo Gamboa Durán - 2210416
Didier Julián Moreno Ortíz - 2202932
Karla Vanessa Ruíz González - 2202808

https://github.com/JuKevCom/Comu2_C1_judika/tree/Practica_I/Practica_I

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

8 de Septiembre de 2024

Abstract

In this practice, we explored the capabilities of GNU Radio for software-defined radio (SDR) by developing and testing blocks for signal accumulation, differentiation, and statistical analysis. Aimed at reconstructing signals affected by Gaussian noise, our approach revealed limitations in signal precision due to noise amplification and suboptimal initial conditions. Despite the functionality of the developed blocks, the recovered signals showed discrepancies compared to the original, highlighting the need for more robust algorithms and noise management techniques. Future improvements include implementing smoothing filters, refining block definitions, and employing vector-based signal definitions to enhance accuracy and signal recovery.

1. INTRODUCCIÓN

La radio definida por software (SDR, por sus siglas en inglés) ha emergido como una tecnología clave en el ámbito de las telecomunicaciones, debido a su flexibilidad y capacidad de adaptación a diferentes protocolos y estándares de comunicación. A diferencia de las radios tradicionales, donde el hardware define las funcionalidades, en SDR estas son determinadas mediante software, lo que permite a los ingenieros actualizar y modificar sistemas de comunicación sin cambiar componentes físicos.[1][2]

A través de la programación de bloques funcionales de GNU Radio en Python, se pretende replicar y analizar diferentes procesos de manipulación y recuperación de señales, como la acumulación, la diferenciación y el cálculo de métricas estadísticas. Este enfoque proporciona una experiencia práctica en el diseño, simulación y validación de sistemas de comunicación basados en SDR, utilizando entornos colaborativos de desarrollo como GitHub para gestionar el trabajo en equipo.

2. METODOLOGÍA

La experimentación fue desarrollada en dos etapas, la primera en lo concerniente a generar el espacio de trabajo de GitHub y la segunda enfocada a la aplicación de la temática vista en clase.

Durante la primera fase, se generó una rama (Practica_I) a partir de la principal, para trabajar en ella todo lo relacionado a este primer informe, en esta rama se creó la carpeta "Práctica_I", dentro de la cuál se crearon otras dos carpetas: GNU_RADIO, donde se guardarían todos los archivos relacionados en dicho aplicativo, e INFORME, donde se irían guardando evidencias fotográficas de lo realizado y se desarrollaría el informe (ver Figura 1).

Fig 1. Creación de carpeta "Práctica_I." en GitHub.

A partir de la rama Practica_I, se creó una rama para cada miembro del grupo, de forma que cada uno podría ir aportando, de manera independiente, distintas eviden-

cias y, a su vez, realizar sus contribuciones independientes al informe previo su entrega (ver Figura 2).

Fig 2. Creación de ramas del equipo en GitHub.

Durante la segunda fase, se utilizaron los códigos en Python proporcionados por el libro guía de la asignatura [2], y, haciendo ciertas modificaciones a estos para que funcionaran de acuerdo a las implementaciones requeridas, se crearon los bloques del acumulador (ver Figura 3) y el diferenciador.

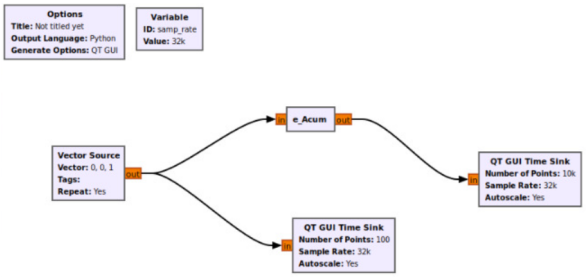

Fig 3. Diagrama de bloques para la verificación del bloque acumulador.

Asimismo, se comprobó su funcionamiento colocando como señal de entrada un vector para el acumulador y una onda triangular para el diferenciador (ver Figura 4).

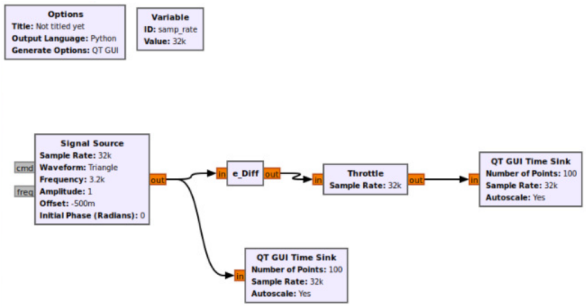

Fig 4. Diagrama de bloques para la verificación del bloque diferenciador.

Por otro lado, se creó un bloque adicional para calcular los diferentes promedios de tiempo de una señal, tales como el promedio, la media cuadrática, la RMS, la potencia promedio y la desviación estándar. Y, para verificar su funcionamiento, se construyó un diagrama de bloques como se puede observar en la Figura 5, donde se utilizó una onda cuadrada como señal de entrada y se visualizaron los resultados de cada parámetro calculado.

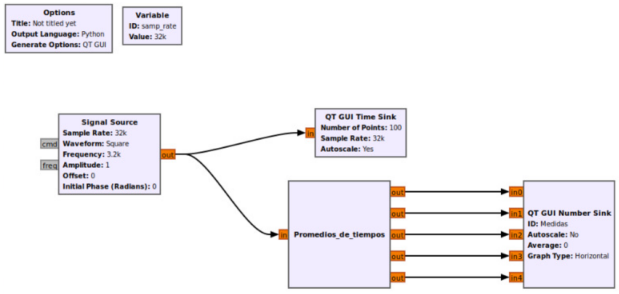

Fig 5. Diagrama de bloques para la verificación del bloque de los promedios de tiempo.

Posteriormente, utilizando los tres bloques desarrollados previamente, se implementó un diagrama de bloques (ver Figura 6) para una aplicación que reconstruía una señal de entrada con ruido gaussiano. En esta, se empleó el bloque diferenciador para diferenciar la señal, el bloque acumulador para recuperarla, y el bloque de promedios de tiempo para visualizar y comparar las estadísticas entre la señal original y la señal recuperada.

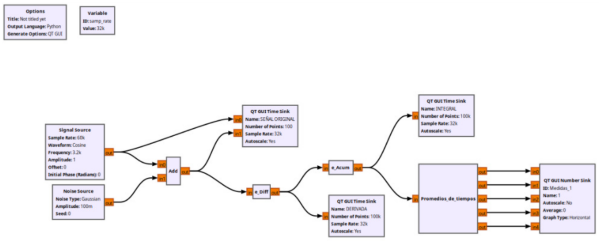

Fig 6. Diagrama de bloques de la aplicación propuesta.

Finalmente, se guardaron todos los cambios realizados en GNURadio y se subieron los archivos a las ramas creadas durante la primera fase, empleando la plataforma de GitHub.

3. RESULTADOS

En primer lugar, antes de realizar cualquier operación con los bloques acumulador y diferenciador proporcionados, se llevó a cabo una pequeña simulación para conocer y verificar el funcionamiento de cada bloque.

Tomando como primer modelo el bloque acumulador, se verificó el correcto funcionamiento del bloque utilizando un vector de 3 datos que se repetía cíclicamente, tal como se muestra en la Figura 3. El resultado esperado de este proceso es una recta con pendiente positiva debido al incremento continuo del valor acumulado. No obstante, al reiniciarse el vector una vez se completa su ciclo, la recta también se reinicia, lo que genera un

comportamiento característico de una onda de dientes de sierra positiva, tal como se presenta en la Figura 7.

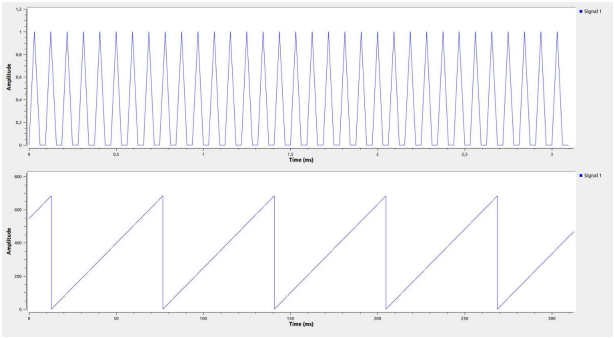

Fig 7. Simulación-verificación del bloque acumulador.

Para verificar el código del diferenciador, se diseñó un esquema distinto, como se muestra en la Figura 4. En dicha figura, se presentan las conexiones de los bloques utilizados para comprobar el correcto funcionamiento del diferenciador. En este caso, se propuso emplear una onda triangular como señal de entrada al bloque diferenciador, de la cual se pretende observar su derivada, esta señal triangular pasa previamente por el bloque Throttle, que ajusta la tasa de muestreo para evitar la saturación de la interfaz visual de GNURadio. El resultado esperado es obtener la derivada de la señal triangular, que corresponde a una onda cuadrada periódica, como se puede apreciar en la Figura 8.

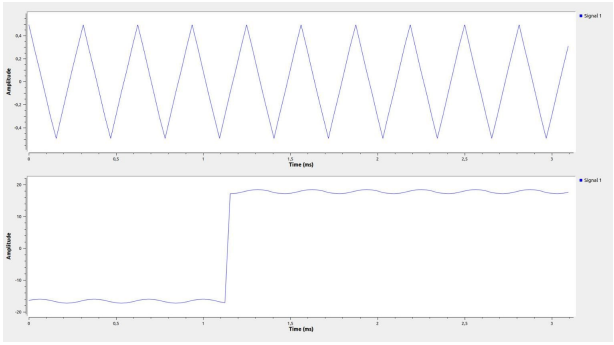

Fig 8. Simulación-verificación del bloque diferenciador.

Además del bloque acumulador y el bloque diferenciador, también se empleó un bloque estadístico denominado “Promedio de tiempos”. La principal función de este bloque es proporcionar valores métricos estadísticos como la media, la media cuadrática, el valor RMS, la potencia promedio y la desviación estándar. Para comprobar estos valores, se construyó el esquema de la Figura 5, en el cual se utiliza una onda cuadrada periódica de amplitud 1 sin offset como señal de entrada para analizar sus estadísticas. Los resultados de estos cálculos se presentan en la Figura 9.

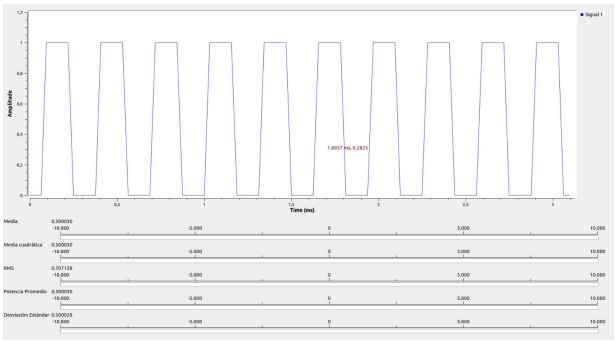

Fig 9. Simulación-verificación del bloque Promedio de tiempos.

Basado en los cálculos obtenidos y comparándolos con los resultados matemáticos reales, se observó el correcto funcionamiento del bloque “Promedio de tiempos” y sus métricas estadísticas. Utilizando este bloque junto con los dos bloques mencionados anteriormente, se planteó el desarrollo de un sistema para la recuperación de una señal, este sistema compara las métricas estadísticas de la señal antes y después del proceso de reconstrucción.

El proceso de reconstrucción de la señal se diseñó utilizando el diagrama de bloques presentado en la Figura 6. En dicho esquema se emplean los bloques mencionados previamente, y, además, con la ayuda del bloque "Noise Source", se logró añadir ruido gaussiano a la señal de entrada para simular situaciones más realistas. Este proceso se realizó con 3 tipos de señales diferentes, en la Figura 10 se presentan las gráficas obtenidas para una señal triangular y en la Tabla I, se condensa la información de las métricas estadísticas de las 3 señales en conjunto.

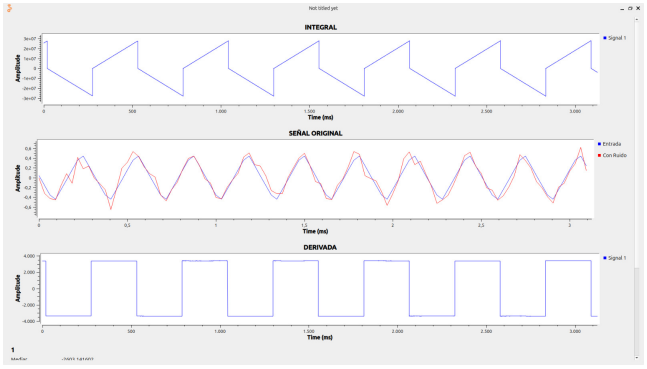

Fig 10. Gráficas de la simulación de reconstrucción para una señal triangular.

Al observar la gráfica superior (La señal recuperada) de la Figura 10, se aprecia que la señal no se recupera de manera óptima. Hay momentos en los que la señal recuperada tiende a parecerse a la señal original (La señal del medio), pero en otros no guardan ninguna similitud.

Al comparar las medidas estadísticas asociadas a cada señal (Ver Tabla I y II), se observa una disparidad significativa entre ellas, lo cual también está influenciado por la incapacidad de recuperar la señal original de forma precisa.

Tabla I. Valores métricos para las señales originales con el ruido gaussiano añadido.

	Triangular	*Diente de Sierra*	*Coseno*
Media	0,000006	−0,000136	−0,000002
Media cuadrática	0,000004	−0,000134	0,000036
RMS	0,002113	0	0,005981
Potencia media	0,000004	0	0,000036
Desviación estándar	0,3006229	0,305507	0,714154

Tabla II. Valores métricos para las señales recuperadas por el sistema de bloques planteado.

	Triangular	*Diente de Sierra*	*Coseno*
Media	−1626,95	2176,45	1110,77
Media cuadrática	120438968	6350300672	72715136
RMS	10974,49	79688,77	8527,31
Potencia media	120438968	6350300672	72715144
Desviación estándar	2545623,25	12207266	2211113,75

Existen varios factores que podrían explicar por qué no se logra recuperar la señal de manera adecuada. El primero de ellos es la incorporación de ruido, al añadir ruido, el proceso de diferenciación e integración se ve afectado, en particular, el uso inicial del bloque diferenciador amplifica el ruido, ya que este bloque tiene un comportamiento similar al de un filtro pasa altas. Como resultado, cuando se aplica el bloque acumulador, el efecto del ruido ya no es despreciable. Además, al añadir ruido a la señal, su media deja de ser cero, lo que complica aún más la tarea de recuperación.

La segunda posible razón está relacionada con la implementación de los bloques de acumulación y diferenciación, los cuales no son una derivada ni una integral estrictamente, lo que puede introducir errores numéricos acumulativos que afectan la señal recuperada. Además, dentro de los códigos utilizados el parámetro “N” lee la longitud de la señal de entrada, pero esta longitud es variable, ya que se trata de una señal de tipo stream, como resultado, los pequeños cambios en la longitud de la señal pueden amplificarse conforme varía el tamaño de la entrada.

Por último, la tercera razón radica en las condiciones iniciales y los extremos de la señal. Si el acumulador no se inicializa correctamente o si hay una falta de sincronización en los bordes de los datos, esto puede afectar el resultado final. Asimismo, es importante considerar las condiciones en los extremos de la señal, ya que un mal manejo de estas podría causar discrepancias en la señal recuperada.

4. CONCLUSIONES

El sistema propuesto para la recuperación de señales afectadas por ruido gaussiano presentó limitaciones en cuanto a la precisión de la señal reconstruida. Aunque los bloques desarrollados cumplieron con su función básica individualmente, cuando se combinaban estos bloques se presentaban inconvenientes que no permitían el correcto funcionamiento del sistema en conjunto.

Basado en los problemas observados durante la práctica, se proponen las siguientes soluciones para mitigar los inconvenientes presentados. En primer lugar, para evitar los cambios bruscos en la señal recuperada, sería ideal utilizar un filtro que suavice la señal antes de procesarla. Además, se podría considerar la modificación interna del código de los bloques para que cumplan rigurosamente con la definición de la derivada y la integral.

Definir la señal de entrada en forma vectorial también ayudaría a evitar errores al calcular las métricas estadísticas, ya que se trabajaría con un tramo temporal definido de la señal, una solución similar sería utilizar un método de ventaneo para limitar la señal, lo que ayudaría a manejar mejor las condiciones en los extremos y así evitar discrepancias en los resultados.

Referencias

[1] L. Couch, *Comunicaciones Digitales basadas en radio definida por software.* Pearson Education, 2008.

[2] H. Ortega and Ó. Reyes, *Comunicaciones Digitales basadas en radio definida por software.* Editorial UIS, 2019.