***ABSTRACT:***  
In the field of engineering, Power Spectral Density (PSD) is a crucial tool for data acquisition and signal analysis, offering insights into how energy is distributed across frequencies. This communications laboratory focuses on understanding the fundamentals of PSD calculation using GNU Radio, where random bipolar signals, images, audio, and binary signals are analyzed. The practice involves generating patterns that facilitate PSD computation and exploring how PSD changes under various sample per symbol (Sps) configurations. Additionally, it covers the construction of custom code blocks to produce averaging vectors and the combination of existing GNU Radio blocks for specific applications involving random signals. By evaluating signals in both the time and frequency domains, the results highlight the effects of signal characteristics and block configurations on the PSD, providing a comprehensive understanding of signal behavior in real-world applications.

***INTRODUCCION:***El análisis de señales desempeña un papel crucial en los campos relacionados con las comunicaciones y el procesamiento de datos. La capacidad de examinar y entender el comportamiento de las señales en el dominio del tiempo y la frecuencia permite a los ingenieros optimizar la transmisión y recepción de información, identificar patrones relevantes y diseñar sistemas de comunicación más eficientes. Dentro de este contexto, la Densidad Espectral de Potencia (PSD) se destaca como una herramienta esencial para evaluar cómo la energía de una señal se distribuye en función de la frecuencia.

Este informe tiene como objetivo profundizar en el análisis de diversos tipos de señales, que abarcan desde señales aleatorias bipolares y unipolares hasta señales de audio, utilizando la plataforma de simulación GNU Radio. A través de esta herramienta, se estudian las características espectrales de las señales y se implementan funciones personalizadas que permiten al estudiante generar vectores de promedio y visualizar las diferencias en la PSD bajo distintas configuraciones y parámetros. La capacidad de interpretar estos resultados y ajustar la configuración de bloques como la interpolación y la modulación es crucial para aplicaciones específicas que requieren un control detallado del espectro.

En esta práctica, se pretende adquirir la capacidad de construir funciones a partir de bloques de código y de combinar estos bloques en GNU Radio para crear esquemas más complejos y eficientes. Esto proporciona una visión integral de cómo analizar y manipular señales en diferentes escenarios, permitiendo una comprensión más profunda de las aplicaciones de la PSD y su utilidad en el diseño de sistemas de comunicaciones avanzados.

***MARCO TEORICO:***

La Densidad Espectral de Potencia (PSD) es una medida clave en el análisis de señales en el dominio de la frecuencia. La PSD describe cómo se distribuye la potencia de una señal a través de las distintas frecuencias que la componen, proporcionando una representación clara de la estructura espectral de dicha señal. Esta herramienta es fundamental en áreas como las comunicaciones, el procesamiento de señales y la ingeniería de control, donde se necesita entender cómo varía la energía de una señal en función de la frecuencia para optimizar su transmisión y procesamiento. La PSD se calcula tomando la transformada de Fourier de la autocorrelación de la señal, lo que permite obtener su distribución espectral. [1][2]

***PROCEDIMIENTO: (BORRADOR)***

1. Creación del entorno de trabajo en GitHub

El primer paso fue la creación de un repositorio en GitHub para gestionar los desarrollos de la práctica. Se creó una rama nueva a partir de la rama principal llamada Practica\_2, donde se organizaron los archivos en dos carpetas: una para los desarrollos de GNU Radio y otra para el informe de resultados. Este enfoque permitió llevar un control detallado de los avances de la práctica y facilita la colaboración en grupo mediante el registro de cambios.

2. Configuración del flujo en GNU Radio

El siguiente paso consistió en abrir y analizar el flujograma proporcionado (randombinaryrectsignal.grc) en GNU Radio, el cual genera una señal binaria aleatoria bipolar de forma rectangular. Este flujo fue la base para el análisis de la Densidad Espectral de Potencia (PSD) de señales aleatorias bajo distintas configuraciones. Los bloques utilizados incluyen:

* Random Source: Genera una secuencia de bits aleatoria que representa la señal binaria.
* Multiply Const: Convierte los bits aleatorios en una señal bipolar con valores entre -1 y 1.
* Thottle: Controla el flujo de la señal y evita la sobrecarga de procesamiento en GNU Radio.
* WX GUI FFT Sink: Visualiza la PSD en tiempo real, mostrando cómo se distribuye la energía de la señal a través de las frecuencias.

La señal generada fue configurada para tener distintas tasas de muestreo y número de muestras por símbolo (Samples per Symbol, Sps). Los valores utilizados para Sps fueron 4, 8, 16 y 1. Esto permitió analizar cómo cambia la PSD a medida que se varía el número de muestras por símbolo, proporcionando una visión clara de cómo estos parámetros afectan la distribución espectral de la señal.

3. Análisis del ruido blanco

El siguiente paso fue analizar el comportamiento del ruido blanco en el dominio del tiempo y la frecuencia. Para ello, se utilizó una fuente de ruido en GNU Radio, configurando bloques de generación de ruido blanco aditivo (AWGN). Se visualizaron las formas de onda y la PSD del ruido, lo cual permitió observar que, como era de esperarse, el ruido blanco tiene una PSD teóricamente infinita en el espectro de frecuencia.

Se llevaron a cabo varias pruebas para comparar las características del ruido blanco con la señal binaria aleatoria. Las visualizaciones obtenidas a través del bloque FFT Sink mostraron una distribución de potencia más homogénea en el espectro para el ruido blanco, en contraste con las señales binarias aleatorias, cuyas PSDs exhiben lóbulos específicos relacionados con su estructura temporal.

4. Uso de señales del mundo real

Luego de analizar señales aleatorias y ruido blanco, se procedió a reemplazar la fuente de bits aleatoria con señales provenientes del mundo real, como imágenes y archivos de audio, para estudiar cómo estas señales afectan la PSD. Este análisis fue realizado de la siguiente manera:

Imágenes (archivo .jpg): Se utilizó un bloque File Source que leyó el archivo rana.jpg. La señal binaria extraída de la imagen fue procesada para analizar su comportamiento tanto en el dominio del tiempo como en la frecuencia. Se configuró el flujo de GNU Radio para visualizar su PSD, permitiendo observar cómo las características de la imagen afectan la estructura espectral de la señal.

Audio (archivo .wav): De manera similar, se utilizó el bloque File Source para leer el archivo de sonido sonido.wav. Se realizaron experimentos para observar cómo la señal de audio se comporta en el dominio temporal y frecuencial al ser representada como una señal binaria bipolar.

Estos experimentos revelaron que las señales del mundo real presentan PSDs más complejas y diferentes comparadas con las señales aleatorias. Mientras que las señales aleatorias tienen una PSD más predecible y simétrica, las señales del mundo real tienen variaciones más irregulares que reflejan las características particulares de la fuente (imagen o audio).

***ANALISIS DE RESULTADOS (PREGUNTAS DE CONTROL)***

***CONCLUCIONES***

* La práctica permitió profundizar en la Densidad Espectral de Potencia (PSD) como una herramienta fundamental para el análisis de señales. Se observó cómo la PSD proporciona información crítica sobre la distribución de energía de una señal a través de diferentes frecuencias, lo que es esencial para optimizar sistemas de comunicación.
* A medida que se variaron los parámetros de muestras por símbolo (Sps), se notó un impacto significativo en la representación espectral de la señal. Específicamente, se comprobó que un mayor número de muestras por símbolo mejora la resolución en el dominio de la frecuencia, permitiendo un análisis más preciso de la PSD y revelando patrones que podrían no ser evidentes con configuraciones de Sps más bajas.

***REFERENCIAS***

*[1] Proakis, J. G., & Manolakis, D. G. (2013). Digital Signal Processing: Principles, Algorithms, and Applications (4th ed.). Prentice Hall.*

*[2] Oppenheim, A. V., & Schafer, R. W. (2009). Discrete-Time Signal Processing (3rd ed.). Prentice Hall.*