UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA:

SIMULACIÓN DE CÓDIGOS DE LÍNEA DESTINADA A TRANSMISIONES DE DATOS EN FORMA DIGITAL

Previa la obtención del Título

INGENIERO EN TELECOMUNICACIONES

ELABORADO POR:
Miguel Santiago Fajardo Brito

Guayaquil, 30 de Agosto del 2014

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por el Sr. **Miguel Santiago Fajardo Brito** como requerimiento parcial para la obtención del título de INGENIERO EN TELECOMUNICACIONES.

Guayaquil, 30 de Agosto del 2014

DIRECTOR

_____

MsC. Edwin F. Palacios Meléndez

REVISADO POR

_____

Ing. Juan Gonzales.
Revisor Metodológico

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

INGENIERÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

MIGUEL SANTIAGO FAJARDO BRITO

DECLARÓ QUE:

El proyecto de tesis denominado **"Simulación de Códigos de Línea destinada a Transmisiones de Datos en forma Digital"**ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros conforme las citas que constan al pie de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Guayaquil, 30 de Agosto del 2014

EL AUTOR

MIGUEL SANTIAGO FAJARDO BRITO

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

INGENIERÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, MIGUEL SANTIAGO FAJARDO BRITO

Autorizó a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del proyecto titulado: **"Simulación de Códigos de Línea destinada a Transmisiones de Datos en forma Digital"**, cuyo contenido, ideas y criterios es de mi exclusiva responsabilidad y autoría.

Guayaquil, 30 de Agosto del 2014

EL AUTOR

MIGUEL SANTIAGO FAJARDO BRITO

DEDICATORIA

Mi tesis se la dedico con todo mi amor, mi cariño y mi aprecio para todas las personas que arrimaron el hombro y creyeron en mi sueño, a ellos les dedico.

A mi madre que me dio la vida, luchando junto a mi en la distancia, sacrificando el tiempo de nosotros, brindándome la sabiduría, amor y paciencia que tanto me sirvió para lograr un sueño planteado que fue duro pero con la firmeza de cumplirlo, su ejemplo de lucha y valentía es el estandarte de mi diario vivir.

Violeta Brito Ruiz

A mi tía quien desinteresadamente con corazón de madre me acompaño a lo largo de todo este sueño, luchó y apoyó en todo sentido, sin duda alguna sin su ayuda yo no hubiera logrado cumplirlo, su ejemplo, caridad y amor hacia el prójimo es un legado en mi vida que deseo continuarlo.

Miguelina Fajardo

A mis hermanos que son mis ejemplos a seguir, motivándome, siendo participes en cada momento de mi carrera sin dejarme caer, buscando la manera de lograr cada paso de mi sueño, dejando de lado su tiempo y objetivos personales, cuanta ayuda recibí de ustedes, nunca me dejaron solo, los amo.

Cristina Solano de la Sala

Diana Fajardo Brito

Andrea Fajardo Brito

Juan Carlos Fajardo Brito

EL AUTOR

MIGUEL SANTIAGO FAJARDO BRITO

AGRADECIMIENTO

Un gracias no es suficiente para expresar el sentimiento de gratitud y reconocimiento de la gran ayuda que me brindaron a lo largo de mi carrera profesional, a mi madre quien trabajo incansablemente Dios le pague madre amada, a mi tía quien hizo posible que pueda lograrlo Dios le pague por tanto cariño y apoyo.

Violeta Brito

Miguelina Fajardo

A mi hermana Diana por su amor y cariño muchas gracias, a mi hermana Andrea quien me permitió conseguir recursos para culminar mi sueño, sin ella no hubiera logrado nada. A mi hermano Juan Carlos quien con su ejemplo, conocimientos aportó hasta el final y gestionó que mi sueño culmine con éxito. A mi hermana Cristina por la paciencia y ayuda que fueron vitales.

Los amo a todos.

EL AUTOR

MIGUEL SANTIAGO FAJARDO BRITO

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

Capítulo 4

Índice de Tablas

Capítulo 2

Resumen

El presente proyecto de titulación al inicio se describen las generalidades del trabajo, tales como Antecedentes, Planteamiento del Problema de Investigación, Objetivo General, Objetivos Específicos, Hipótesis y la Metodología de Investigación utilizada para el desarrollo satisfactorio de las simulaciones mediante programación gráfica GUI bajo la plataforma MatLab.

El propósito del trabajo de titulación fue emplear algunas técnicas de codificación (códigos de líneas) muy utilizados en los sistemas de comunicaciones que transmiten señales digitales. Para lo cual en el Capítulo 2 se revisaron todas las técnicas existentes. En el capítulo 3 se explica la programación de interfaz gráfica de usuario (GUI o GUIDE) que pertenece a la plataforma MatLab. En el capítulo 4 se muestra el diseño de GUIs y a la vez se presentan los resultados obtenidos a través de las simulaciones de algunas técnicas de codificación.

CAPÍTULO 1: GENERALIDADES DEL TRABAJO DE TITULACIÓN

1.1. Antecedentes.

Los códigos de línea o codificación, consiste en representar la señal digital para ser transportados por una amplitud y señal de tiempo discreto que está óptimamente ajustado para propiedades específicas del canal físico y del equipo de recepción. Después de la aparición de la tecnología VLSI (Integración en escala muy grande), los investigadores continuamente tratan de reducir los requisitos de potencia, la optimización de área y conseguir un menor retardo de propagación mediante el uso de diversos algoritmos.

Los códigos o línea de codificación son bloques fundamentales para todo sistema de comunicación, en el que tanto “1” y “0” se traducen en la secuencia de la tensión y el pulso de corriente que se puede propagar a través de medios físicos, como el cable coaxial, fibra óptica, etc. Las señales de banda base digital, emplean a menudo estás técnicas para proporcionar determinadas características espectrales de un tren de impulsos.

La mayoría de los códigos de línea populares son el retorno a cero (RZ) y no retorno a cero (NRZ). Todos ellos son, ya sea en formato codificado unipolar o polar. La elección de los códigos de línea depende siempre de:

a. La presencia o ausencia de nivel de corriente continua (CC),
b. La densidad espectral de potencia (PSD)

c. Los requisitos de ancho de banda,
d. El rendimiento de tasa de error de bit,
e. La facilidad de recuperación de la señal de reloj o ausencia de la propiedad de detección inherente.

1.2. Definición del Problema a Investigar.

Los sistemas de comunicaciones emplean técnicas de codificación, tales como los códigos de línea, en la carrera de Ingeniería en Telecomunicaciones estos han sido tratados de manera teórica y nada de práctica experimental, tanto física como virtual. Esto ha sido difícil para el aprendizaje de los estudiantes, inclusive para poder graficar las densidades espectrales de potencia. Por lo expuesto, surge la necesidad de desarrollar herramientas de simulación bajo alguna plataforma de simulación y programación como MatLab en la mayoría de sistemas empleados en las Telecomunicaciones.

1.3. Objetivos del Problema de Investigación.

Una vez planteada la definición del problema, se redactan el objetivo general y los objetivos específicos del proyecto de titulación.

1.4.1. Objetivo General.

Desarrollar una interfaz de programación gráfica para la simulación de códigos de línea destinada a transmisiones de datos en forma digital.

1.4.2. Objetivos Específicos.

- Describir el Estado del Arte de la codificación lineal o de códigos de línea necesarios en los Sistemas Electrónicos de Comunicaciones.
- Examinar la funcionalidad de la interfaz de programación gráfica GUI-MatLab necesaria para el desarrollo de las simulaciones.
- Preparar el diseño de la interfaz gráfica de las técnicas de codificación de línea y de la densidad espectral de potencia.
- Verificar los resultados obtenidos para que cumplan con los principios básicos de funcionalidad.

1.4. Hipótesis.

Mediante las técnicas de codificación o códigos de líneas seleccionadas, se logrará validar las mismas a través de una plataforma de interfaz gráfica robusta como GUI-MatLab. Esto permitirá incrementar los conocimientos adquiridos en los sistemas de comunicaciones y del modelado de ciertos procesos o tecnologías utilizados en la actualidad en las Telecomunicaciones.

Para el trabajo de titulación se escogieron como variable dependientelas técnicas de codificación lineal y la variable independiente sería las señales digitales o tren de bits aleatorios para diferentes códigos de línea. Dichas variables ayudarán a que el modelo matemático de los códigos de líneas escogidos se desempeñe correctamente como si estuvieran operando en la realidad.

1.5. Metodología de Investigación.

El proyecto de titulación tiene dos alcances, el Exploratorio y el Explicativo. El primero porque se examinará un problema poco estudiado, debido a que existen muchas dudas o porque el tema en cuestión no se ha abordado en temas de titulación. Mientras que el segundo alcance, se dirige a expresar las causas en lo que ocurre el fenómenos físico mediante la programación de interfaz gráfica, GUI de MatLab. Finalmente, el modelo empleado de investigación es Empírico-Analítico.

CAPÍTULO 2: ESTADO DEL ARTE DE CÓDIGOS DE LÍNEA

2.1. Introducción a Redes de Comunicaciones de Datos.

La línea de la terminología de codificación se originó en la telefonía con la necesidad de transmitir información digital a través de una línea telefónica de cobre; más concretamente, los datos binarios a través de una línea de repetidores digitales. El concepto de la codificación de línea, sin embargo, se aplica fácilmente a cualquier línea de transmisión o canal.

En un sistema de comunicación digital, existe un conjunto conocido de símbolos a transmitir. Estos pueden ser designados como {mi}, i = 1, 2,…, N, con una probabilidad de ocurrencia {Pi}, i = 1, 2,…, N, donde los símbolos transmitidos secuencialmente son asumidos generalmente para ser estadísticamente independientes. La conversión o la codificación de estos símbolos abstractos en formas de onda reales y temporales a transmitir en la banda base es el proceso de codificación de línea.

Dado que el tipo más común de codificación de línea es para los datos binarios, una forma de onda de este tipo puede ser de manera sucinta denominado un formato directo para bits en serie. La concentración de esta sección será la línea de codificación para datos binarios. Diferentes características del canal, así como las diferentes aplicaciones y requisitos de desempeño, han proporcionado el impulso para el desarrollo y el estudio de los distintos tipos de línea de codificación [1, 2].

Por ejemplo, el canal podría acoplarse a corriente ac y por lo tanto, no podía apoyar un código de línea con una componente de corriente continuo o alto contenido de corriente dc. Los requisitos de recuperación de sincronización o de temporización pueden requerir un componente discreto en la velocidad de datos. El ancho de banda del canal y limitaciones de diafonía pueden dictar el tipo de codificación de línea ocupada.

Incluso los factores tales como la complejidad del codificador y la economía del decodificador podrían determinar el código de línea elegida. Cada código de línea tiene sus propias características distintivas.Dependiendo de la aplicación, una propiedad puede ser más importante que el otro. En lo que sigue, se describe, en general, las características más deseables que se consideran al elegir un código de línea.

Es comúnmente aceptado [1, 2, 5, 8] que las consideraciones dominantes que afectan la elección de un código de línea son: 1) de temporización, 2) el contenido dc, 3) espectro de potencia, 4) monitoreo del desempeño, 5) la probabilidad de error, y 6) la transparencia. Cada uno de estos se detallan en los siguientes párrafos:

1)**Momento:** La forma de onda producida por un código de línea debe contener información de tiempo suficiente de modo que el receptor puede sincronizarse con el transmisor y decodificar la señal recibida correctamente. El contenido de temporización debe ser relativamente independiente de

estadísticas de fuente, es decir, una larga cadena de 1s o 0s no debe dar lugar a la pérdida de la oportunidad o la oscilación en el receptor.

2) **Contenido de dc:** Dado que los repetidores utilizados en la telefonía son acoplados en ac, es deseable tener cero corriente dc en la forma de onda producida por un código de línea dado. Si entra una señal con un contenido significativo dc se utiliza en líneas de corriente alterna acoplada, causará vagar en la forma de onda recibida. Es decir, la línea de base de la señal recibida variará con el tiempo.

Las líneas de teléfono no pasan dc debido al acoplamiento de corriente alterna con los transformadores y condensadores para eliminar bucles de tierra dc. Debido a esto, el canal telefónico causa una caída en señales constantes. Esto causa dc vagar. Puede ser eliminado por los circuitos dc de restauración, los sistemas de retroalimentación, o con códigos de línea especialmente diseñados.

3) **Espectro de potencia:** El espectro de potencia y ancho de banda de la señal transmitida debe corresponder a la respuesta de frecuencia del canal para evitar la distorsión significativa. Además, el espectro de potencia debe ser tal que la mayor parte de la energía está contenida en el ancho de banda tan pequeño como sea posible. Cuanto menor es el ancho de banda, mayor es la eficiencia de transmisión.

4) **Supervisión del rendimiento:** Es muy deseable detectar errores causados por un canal de transmisión ruidosa. La capacidad de detección de errores a su vez permite la supervisión del rendimiento, mientras que el canal está en uso (es decir, sin procedimientos de prueba elaborados que requieren el uso de suspensión del canal).

5) **Probabilidad de error:** La probabilidad de error promedio debe ser tan pequeña como sea posible para una potencia de transmisión dada. Esto refleja la fiabilidad del código de línea.

6) **Transparencia:** Un código de línea debe permitir que todos los posibles patrones de 1s y 0s. Si un cierto patrón no es deseable debido a otras consideraciones, se debe asignar a un patrón la única alternativa.

2.2. Formatos de Códigos de Línea comunes.

Una línea de formato de codificación consiste en una definición formal de la línea de código que específica cómo una cadena de dígitos binarios se convierte en una forma de onda de código de línea. Hay dos clases principales de códigos de línea: códigos binarios y códigos de nivel de transición.

Los códigos de nivel llevan información en su nivel de tensión, que puede ser alta o baja para un período completo de bits o parte del período de bit.

Los códigos de nivel son en general instantáneos, ya que típicamente codifican un dígito binario en una forma de onda distinta, independiente de cualquier dato binario pasado. Sin embargo, algunos códigos de nivel no presentan memoria. Los códigos de transición llevan la información en cambio en el nivel que aparece en la forma de onda del código de línea.

Los códigos de transición pueden ser instantáneos, pero generalmente tienen memoria, a partir de datos binarios pasados para dictar la presente forma de onda. Hay dos formas comunes de códigos de línea de nivel: uno se llama retorno a cero (RZ) y el otro se llama retención a cero o de no retorno a cero (NRZ). En la codificación RZ, el nivel del pulso vuelve a cero para una porción del intervalo de bit. En la codificación NRZ, el nivel del impulso se mantiene durante todo el intervalo de bit.

Los formatos de codificación de línea se clasifican de acuerdo a la polaridad de los niveles de tensión utilizados para representar los datos. Si sólo se utiliza una polaridad de nivel de tensión, es decir, positivo o negativo (además del nivel cero), entonces se llama señalización unipolar.Si se utilizan ambos niveles de tensión positivos y negativos, con o sin un nivel de voltaje cero, entonces se llama señalización polar.

El término de la señalización bipolar es utilizado por algunos autores para designar a un régimen específico de codificación acorde con niveles

positivos, negativos y de tensión cero. Esto se describirá en detalle más adelante en esta sección.

La definición formal de cinco códigos de línea comunes se da también más adelante, junto con una forma de onda representativa de la densidad espectral de potencia (PSD), la probabilidad de error, y una discusión de las ventajas y desventajas. En algunos casos se observaron aplicaciones específicas.

2.2.1. Código Unipolar NRZ *(Binary On-Off Keying)*.

En este código de línea, un 1 binario está representado por un nivel de tensión distinta de cero y un 0 binario está representado por un nivel de voltaje cero como se muestra en la figura2.1 (a). Este es un código de nivel instantáneo. La densidad espectral de potencia (PSD) de este código con igual probabilidad 1s y 0s está dada por [5, 8]:

$$s_1(f) = \frac{V^2 T}{4}\left(\frac{\sin \pi f T}{\pi f T}\right)^2 + \frac{V^2}{4}\delta(f)$$

donde$V$es el nivel de voltaje de 1 binario,$T = \frac{1}{R}$es la duración de bit, yR es la tasa en bits por segundo. El espectro unipolar NRZ se representa en la figura 2.2. Este PSD es un espectro aún más por las dos caras, aunque sólo la mitad del gráfico muestra la eficiencia de la presentación. Si la probabilidad de que un “1” binario es$p$, y la probabilidad de que un “0” binario es$(1 - p)$.

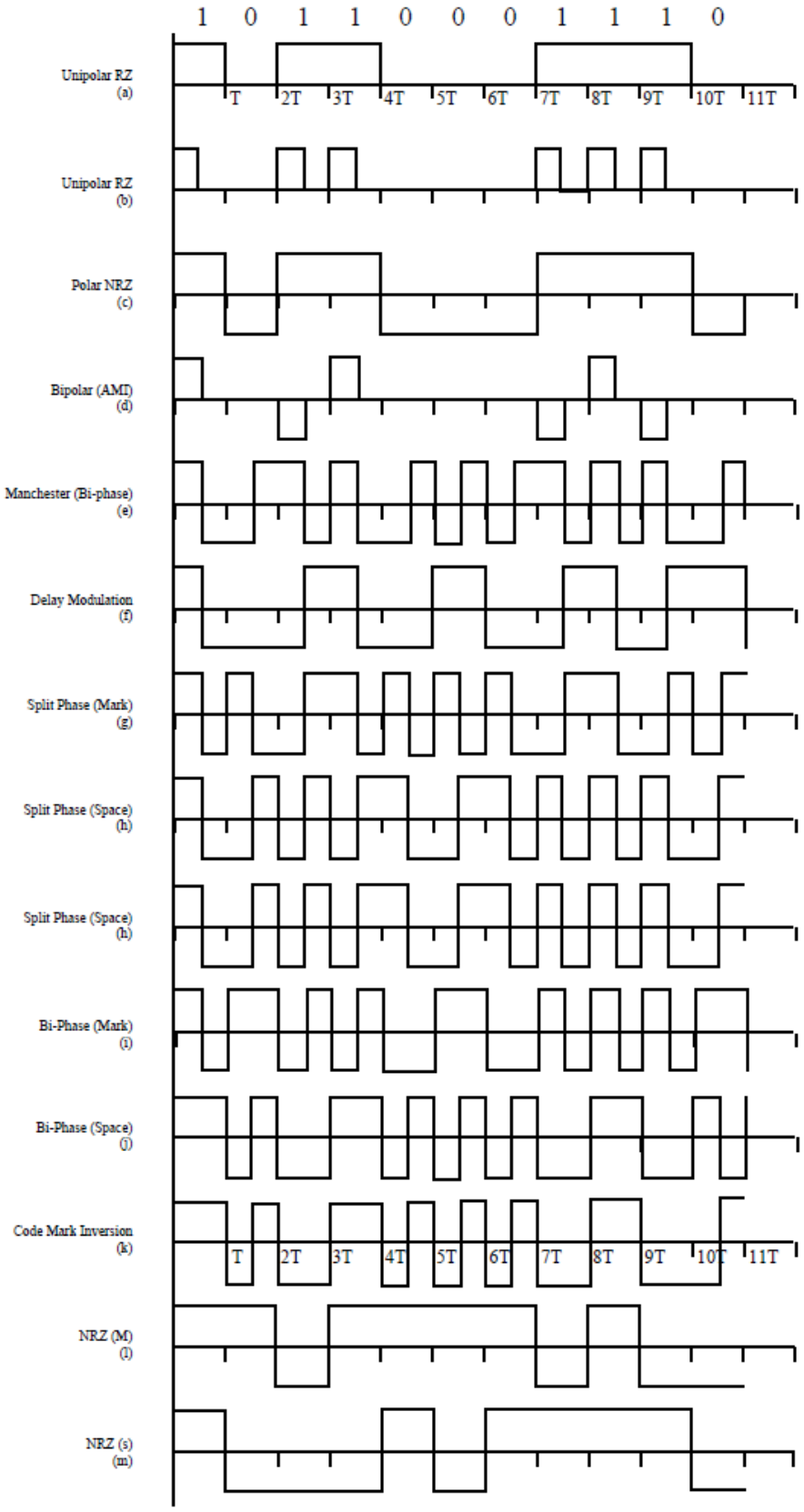

Figura 2. 1: Formas de onda para diferentes códigos de línea.
Fuente: Chitode J., S. (2009).

A continuación, el PSD, en el caso más general, resulta ser $4p(1-p)S_1(f)$. Teniendo en cuenta la frecuencia de la primera nula espectral como el ancho de banda forma de onda, el ancho de banda unipolar NRZ es R en hercios.

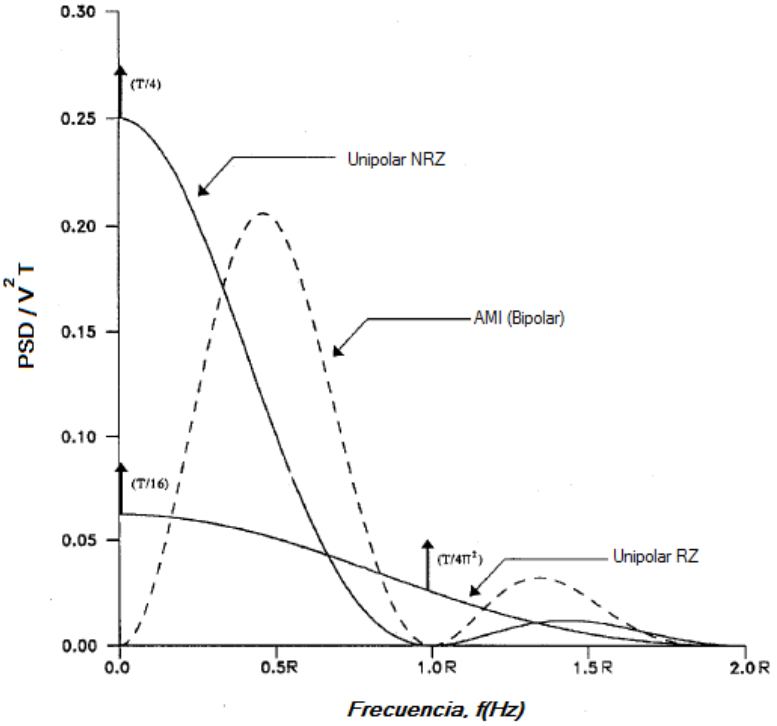

Figura 2. 2: Densidad espectral de potencia de códigos de línea Unipolar NRZ, AMI (Bipolar) y Unipolar RZ, donde $\boldsymbol{R} = \frac{\mathbf{1}}{\boldsymbol{T}}$ es la tasa de bits.
Fuente: Chitode J., S. (2009).

La tasa de error de rendimiento de este código por igual a los datos probables, con Ruido Gaussiano Blanco Aditivo *(Additive White Gaussian Noise, AWGN)*óptimo, es decir, filtro adaptado, la detección viene dada por [1, 5]:

$$P_e = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{2N_0}}\right)$$

DondeE_b/N_0 es una medida de la relación señal/ruido (SNR) de la señal recibida. En general, $E_b$es la energía por bit y$N_0/2$ es el PSD doble cara de la AWGN. Más específicamente, para NRZ unipolar, Eb es la energía en un 1 binario, que es V 2T. La tasa de error de rendimiento de este código NRZ unipolar se representa en la figura2.4

Las ventajas principales de unipolar NRZ esla facilidad de generación, ya que requiere únicamente de una fuente de alimentación, y un ancho de banda relativamente bajo de R Hz. Hay un buen número de desventajas de este código de línea. Una pérdida de sincronización y temporización de fluctuación de fase puede dar lugar con una larga secuencia de 1s o 0s porque ninguna transición de impulso está presente.

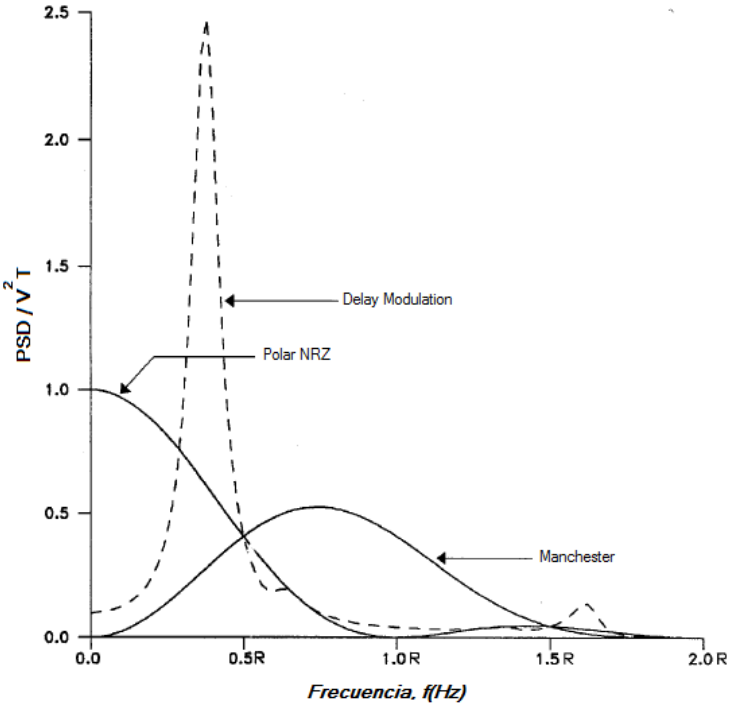

Figura 2. 3: Densidad espectral de potencia de códigos de línea Delay Modulation, Polar NRZ, y Manchester, donde $\boldsymbol{R = \frac{1}{T}}$ es la tasa de bits.
Fuente: Chitode J., S. (2009).

El código no tiene capacidad de detección de errores y, por lo tanto, el rendimiento puede no ser monitoreado. Hay un significativo componente de corriente continua, así como un contenido de dc. La tasa de error de rendimiento no es tan buena como la de los códigos de línea polares.

2.2.2. Código Unipolar RZ.

En este código de línea, un “1” binarioestá representado por un nivel de voltaje distinto de cero durante una parte de la duración de bit, por lo general por medio del período de bits, y un nivel de tensión cero para el resto de la duración del bit. Un “0” binario está representado por un nivel de voltaje cero durante toda la duración del bit. Por lo tanto, este es un código de nivel instantáneo. La figura 2.1 (b) ilustra una forma de onda Unipolar RZ en la cual el 1 está representado por un nivel de voltaje medio distinto de cero para el período de bits. La PSD de este código de línea, con los dígitos binarios de igual probabilidad, está dada por [5, 6, 8]:

$$+\frac{V^2}{4\pi^2}\left[\frac{\pi^2}{4}\delta(f)+\sum_{n=-\infty}^{\infty}\frac{1}{(2n+1)^2}\delta(f-(2n+1)R)\right]$$

donde de nuevo Ves el nivel de tensión de un “1” binario, y $T=\frac{1}{R}$es el periodo de bits. El espectro de este código se mostró en la figura 2.3. En el caso más general, cuando la probabilidad de “1” es$p$, la parte continua de la PSD en la ecuación anterior, es escalada por el factor de$4p(1-p)$ y la porción discreta es escalada por el factor de $4p^2$.

El primer ancho de banda nulo unipolar RZ es $2R\ Hz$. El rendimiento de la tasa de error para este código, es el mismo que el de Unipolar NRZ, con tal de aumentar el nivel de tensión de este código de tal manera que la energía en formato binario sea 1, E_b, es la misma para ambos códigos. La probabilidad de error es dada por la ecuación $P_e = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_b}{2N_0}}\right)$ e identificada en la figura 2.4.

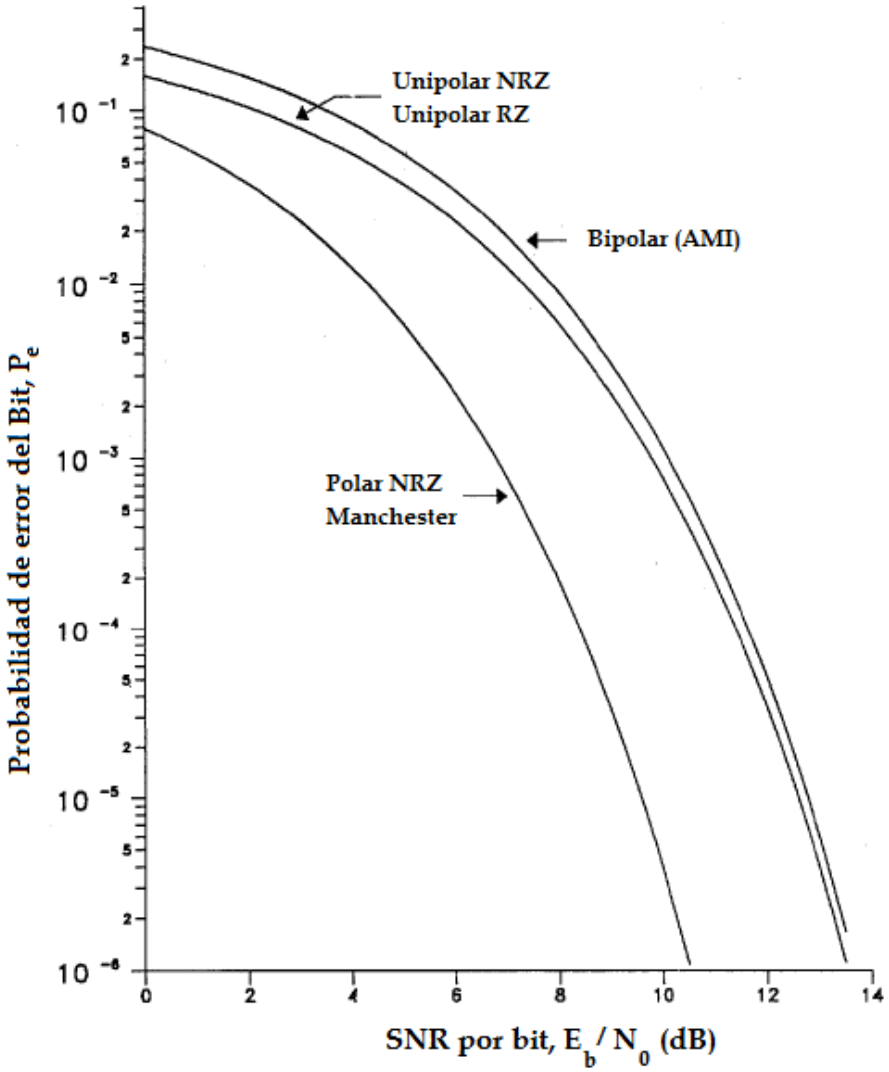

Figura 2. 4: Probabilidad de error de bits para códigos de línea Polar NRZ, Manchester, Bipolar (AMI) y Unipolares NRZ y RZ.
Fuente: Chitode J., S. (2009).

Si el nivel de tensión y período de bits son los mismos para los códigos unipolar NRZ y RZ, entonces la energía en código binario “1” para unipolar RZ será$V^2T/2$ y la probabilidad de error empeoraa los 3 dB.Las principales ventajas de unipolar RZ son, de nuevo, la facilidad de generación, ya que requiere una única fuente de alimentación y la presencia de un componente espectral discreto en la velocidad de símbolo, que permite la recuperación de temporización sencilla. Una serie de inconvenientes existe para este código de línea. Tiene un componente de corriente dc distinto de cero y contenido de corriente dc distinto de cero, lo que puede dar lugar a divagar en dc.

A lo largo de la cadena de 0s carecerán transiciones de impulsos y podría conducir a la pérdida de sincronización. No hay capacidad de detección de error y, por lo tanto, la supervisión del rendimiento no es posible. El requisito de ancho de banda (2R Hz) es más alta que la de las señales NRZ. La tasa de error de rendimiento es peor que la de los códigos de línea polares.

Tanto los códigos Unipolar NRZ así como Unipolar RZ son ejemplos del tipo de señalización de pulso/no pulso. En este tipo de señalización, el pulso para un “0” binario, $g_2(t)$es cero y el pulso para un “1” binario se especifica de forma genérica como$g_1(t) = g(t)$. Usando $G(f)$como la transformada de Fourier de$g(t)$, el PSD de la señalización de pulso/nopulso se da como [6, 7, 10]:

$$S_{PNP}(f) = p(1-p)R|G(f)|^2 + p^2R^2 \sum_{n=-\infty}^{\infty} |G(nR)|^2 \delta(f - nR)$$

donde$p$es la probabilidad de un “1” binarioy$R$ es la tasa de bits.

2.2.3. Código Polar NRZ

En este código de línea, un “1” binario está representado por un voltaje $+V$y para un “0” binario está representado por un voltaje $-V$durante el período de bit completo. Este código también se conoce como NRZ (L), ya que un bit se representa mediante el mantenimiento de un nivel (L) durante la totalidad de su periodo. Una forma de onda del código Polar NRZ se muestra en la figura 2.1 (c).

Se trata nuevamente de un código de nivel instantáneo. Alternativamente, un “1” se puede representar por un nivel de voltaje$-V$ y un “0” por un nivel de voltaje $+V$, sin necesidad de cambiar las características espectrales yrendimiento del código de línea. La PSD de este código de línea con igual probabilidad de bits se indica por [5, 8]:

$$S_3(f) = V^2T\left(\frac{\sin \pi fT}{\pi fT}\right)^2$$

Esta ecuación se representa gráficamente en la figura 2.3. Cuando la probabilidad de un “1” es$p$, yp no es 0.5, existe una componente de corriente continua, y el PSD se convierte en [10]:

$$S_3(f) = 4V^2Tp(1-p)\left(\frac{\sin \pi fT}{\pi fT}\right)^2 + V^2(1-2p)^2\delta(f)$$

El primer ancho de banda nulo para este código de línea es de nuevo$R\ Hz$, independiente de$p$. La probabilidad de error de este código de línea cuando$p = 0.5$ viene dada por [1, 5]:

$$P_e = \frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right)$$

El rendimiento de polar NRZ se representa en la figura 2.4. Esto es mejor que la característica de error de los códigos unipolar de 3 dB.Las ventajas del código polar NRZ incluyen el requisito del ancho de banda bajo, R Hz, comparable a unipolar NRZ, muy buena probabilidad de error, y reducido en gran medida debido a que la forma de onda de corriente continua tiene un componente cero dc cuando p = 0.5 aunque el contenido dc nunca es cero.

Unas pocas desventajas notables son que no hay capacidad de detección de error, y que una cadena larga de 1s o 0s podría resultar en la pérdida de la sincronización, ya que no hay transiciones durante la duración de cadena. Se requieren dos fuentes de alimentación para generar este código.

2.2.4. Polar RZ [Bipolar, Inversión de marcas alternadas (AMI), o Pseudoternario]

En este esquema, un “1” binario está representado por la alternancia de los niveles de tensión positivos y negativos, que devuelven a cero para

una parte de la duración de bit, generalmente la mitad del período de bits. Un “0” binario está representado por un nivel de voltaje cero durante toda la duración del bit.

Este esquema de codificación de línea a menudo se denomina inversión de marca alternada *(Alternate Mark Inversion, AMI)* ya que 1s (marca) están representados por la alternancia de impulsos positivos y negativos. También se la denomina pseudoternario, ya que tres niveles de tensión diferentes se utilizan para representar datos binarios.

Algunos autores designan este código de línea como Bipolar RZ (BRZ). Una forma de onda AMI se muestra en la figura 2.1 (d). Hay que tener en cuenta que este es un código de nivel con memoria. El código AMI es bien conocido por su uso en la telefonía. El PSD de este código de línea con la memoria está dado por [1, 2, 7]:

$$S_{4p}(f) = 2p(1-p)R|G(f)|^2\left(\frac{1-\cos 2\pi fT}{1+(2p-1)^2+2(2p-1)\cos 2\pi fT}\right)$$

donde$G(f)$es la transformada de Fourier del pulso utilizado para representar un “1” binario, y $p$es la probabilidad de un “1” binario. Cuando$p = 0.5$ y mediante pulsos cuadrados con amplitud ±V cuya duración es $\frac{T}{2}$ son utilizados para representar 1s binarios, el PSD se convierte:

$$S_4(f) = \frac{V^2T}{4}\left[\frac{\sin\left(\frac{\pi fT}{2}\right)}{\frac{\pi fT}{2}}\right]^2 \sin^2(\pi fT)$$

Este PSD para este código se muestra en la figura 2.2. El primer ancho de banda nulo de esta forma de onda es $R\,Hz$. Esto es cierto para los impulsos rectangulares RZ, independientes del valor de$p$en la ecuación representada por $S_{4p}(f)$. El comportamiento de la tasa de error de este código de línea es la misma probabilidad para los datos binarios viene dada por [5]:

$$P_e \approx \frac{3}{4}\operatorname{erfc}\left(\sqrt{\frac{E_b}{2N_0}}\right), \qquad \frac{E_b}{N_0} > 2$$

Esta curva se representa en la figura. 6.3 y se ve que es no más de 0.5 dB peor que los códigos unipolares. Las ventajas del código Polar RZ (o AMI, ya que es más comúnmente llamado) superan las desventajas. Este código no tiene ningún componente de corriente continua y el contenido dc es cero, evitando por completo el problema de fluctuación lenta de fase dc.

La recuperación de sincronismo o temporización es bastante fácil, ya que la cuadraturao rectificación de onda completa para este tipo de señal produce una forma de onda Unipolar RZ con un componente discreto en la velocidad de bits, R Hz.Debido a los impulsos de polaridad alterna para 1s binarios, este código dispone de detección de errores y, por lo tanto, la capacidad de supervisión del rendimiento.

Tiene un bajo requerimiento de ancho de banda $R\,Hz$, comparable al código Unipolar NRZ. La desventaja obvia, es que el rendimiento de tasa de

error de rendimiento es peor que la de las formas de onda unipolares y polares. Una larga serie de 0s podría resultar en la pérdida de sincronización, y por lo tanto se requieren dos fuentes de alimentación para este código.

2.2.5. Codificación Manchester (Fase dividida o Bifásico digital)

En esta codificación, un “1” binario está representado por un pulso que tiene un voltaje positivo durante la primera mitad de duración del bit y tensión negativa durante la segunda mitad de la duración del bit. Un “0” binario está representado por un pulso que es negativo durante la primera mitad de la duración del bit y positiva durante la segunda mitad de la duración del bit.

La mitad del flanco descendente negativo o positivo indica un “1” binario o “0” binario, respectivamente.Por lo tanto, un código de Manchester se clasifica como un código de transición instantánea, es decir, que no tiene memoria.El código también se llama bifásico, porque una onda cuadrada con una fase de 0° se utiliza para representar un “1” binario y una onda cuadrada con una fase de 180° se utiliza para representar un “0” binario; o viceversa.

Este código de línea se utiliza en Redes de Área Local (LAN) Ethernet. La forma de onda, es la misma probabilidad para codificación Manchester que se muestra en la figura 2.1 (e). El PSD de una forma de onda de Manchester con igual probabilidad de bits se representa por [5, 8]:

$$S_5(f) = V^2T\left[\frac{\sin\left(\frac{\pi fT}{2}\right)}{\frac{\pi fT}{2}}\right]^2 \sin^2\left(\frac{\pi fT}{2}\right)$$

donde$\pm V$son utilizados como los niveles de tensión positivos y negativos para este código. Su espectro se puede visualizar en figura 2.3. Cuando la probabilidad$p$ de un “1” binario, no es igual a la mitad de 1, la porción continua del PSD se reduce en componentes de amplitud y aparecen componentes discretosen múltiplos enteros de la velocidad de bits $R = \frac{1}{T}$. El PSD resultante es [6, 10]:

$$S_{5p}(f) = V^2T4p(1-p)\left[\frac{\sin\left(\frac{\pi fT}{2}\right)}{\frac{\pi fT}{2}}\right]^2 \sin^2\left(\frac{\pi fT}{2}\right) + V^2(1-2p)^2 \sum_{n=-\infty,\ n\neq 0}^{\infty} \left(\frac{2}{n\pi}\right)^2 \delta(f-nR)$$

El primer ancho de banda nulo de la forma de onda generada por un código Manchester es$2R\ Hz$. El rendimiento de la tasa de error de rendimiento de esta forma de onda, cuando$p = 0.5$ es la igual al del código Polar NRZ, dada por la función $S_4(f)$ y se representa en la figura 2.4.

Las ventajas de este código incluyen un contenido de corriente continua nula (cero) en forma de impulsos individuales, por lo que hay un patrón de bits que pueden ocasionar la acumulación de corriente continua (dc); la mitad de transiciones de los bits están siempre presentes, lo que sería fácil de extraer información de la sincronización (temporización); y esto

provoca un buen comportamiento de la tasa de error, idéntico al código Polar NRZ. La principal desventaja de este código es un ancho de banda más grande que cualquiera de los otros códigos comunes. Además, no tiene capacidad de detección de errores y, por lo tanto, la supervisión del rendimiento no es posible.

La codificación Polar NRZ y Manchester son ejemplos del uso de la señalización polar puro, donde el pulso para un “0” binario, $g_2(t)$es el pulso negativo para un “1” binario, es decir, que $g_2(t) = -g_1(t)$. Esto también se conoce como un conjunto de señales antipodales. Para esta ampliaclase de código de línea polar binario, el PSD viene dado por [10]:

$$S_{BP}(f) = 4p(1-p)R|G(f)|^2 + (2p-1)^2R^2 \sum_{n=-\infty}^{\infty} |G(nR)|^2 \delta(f - nR)$$

donde$|G(f)|$ es la magnitud de la transformada de Fourier de cualquiera de $g_1(t)$o$g_2(t)$.

Una generalización adicional de la PSD de códigos de línea binarios se puede dar, donde en un espectro continuo y un espectro discreto es evidente. Dejamos un “1” binario, con probabilidad$p$, ser representado por $g_1(t)$ sobre el segundo intervalo de bit $T = \frac{1}{R}$; y dejar que un “0” binario, con probabilidad$1-p$, ser representado por$g_2(t)$ durante el mismo intervalo de $g_1(t)$. El PSD a doble cara para este código de línea binaria general es [10]:

$$S_{GB}(f) = p(1-p)R|G_1(f) - G_2(f)|^2$$

$$+ R^2 \sum_{n=-\infty}^{\infty} |pG_1(nR) + (1-p)G_2(nR)|^2 \delta(f - nR)$$

donde la transformada de Fourier de $g_1(t)$ y $g_2(t)$ están dadas por $G_1(f)$y $G_2(f)$, respectivamente.

2.3. Códigos de línea alternativos.

La mayoría de los códigos de línea discutida hasta ahora fueron los códigos de nivel instantáneo. Aunque sólo AMI tenía memoria, y el Manchester era un código de transición instantánea. Los códigos de línea alternativos presentados en esta sección, tienen memoria.

Los cuatro primeros son los códigos de transición, donde los datos binarios se representan como la presencia o ausencia de una transición, o por la dirección de la transición, es decir, de positivo a negativo o viceversa. Los últimos cuatro códigos descritos en esta sección son los códigos de línea nivel con memoria.

2.3.1. Retardo de Modulación o Delay Modulation (Código Miller)

En este código de línea, un “1” binario está representado por una transición en la posición media del bit, y un 0 binario está representado por ninguna transición en la posición media del bit. Si el “0” es seguido por otro

“0”, no obstante la transición de la señal también se produce al final del intervalo de bit, es decir, entre las dos 0s.

Un ejemplo de retardo de modulación se muestra en la figura 2.1 (f), es evidente que el retraso de modulación es un código de transición con memoria. Este código logra el objetivo de ofrecer un buen contenido de temporización sin sacrificar el ancho de banda. La PSD del código Miller para los datos de la misma probabilidad está dada por [10]:

$$S_6(f) = \frac{V^2T}{2(\pi fT)^2(17 + 8\cos 2\pi fT)} \times (23 - 2\cos \pi fT - 22\cos 2\pi fT - 12\cos 3\pi fT + 5\cos 4\pi fT + 12\cos 5\pi fT + 2\cos 6\pi fT - 8\cos 7\pi fT + 2\cos 8\pi fT)$$

Este espectro fue mostrado en la figura 2.3, una ventaja de este código es que requiere relativamente de poco ancho de banda, y la mayor parte de la energía está contenido en menos de $0.5R$. Sin embargo, no existen diferentes espectros nulos dentro de la banda$2R - Hz$. Tiene un bajo contenido de corriente continua y sin componente de continua.

Finalmente, este código tiene muy buencontenido de temporización y de seguimiento de portadora, siendo más sencillo que la codificación Manchester. El rendimiento de la tasa de error, es comparable a la de los códigos de línea comunes. Una desventaja importante es que no tiene

capacidad de detección de errores y, por lo tanto, el rendimiento puede no ser monitoreado.

2.3.2. Fase Dividida*(Split Phase)*, Mark

Este código es similar al de Manchester, en el sentido de que siempre hay transiciones de medio bit. Por lo tanto, este código es relativamente fácil de sincronizar y no tiene corriente dc. Sin embargo, a diferencia de Manchester, la fase (marca) dividida codifica un dígito binario en un medio bit, la transición depende de la transición del medio bit en el período del bit anterior [12].

Específicamente, un “1” binario produce una reversión de transición del medio bit relativa a la transición del medio bit anterior. Para un “0” binario no se produce reversión de la transición del medio bit. Ciertamente se trata de un código de transición con memoria.Un ejemplo de una fase de división se muestra la forma de onda codificada en la figura 2.1 (g), en donde se elige la forma de onda en el primer período de bits de forma arbitraria.

Dado que este método codifica diferencialmente los bits, no existe 180° de ambigüedad de fase asociado con algunos los códigos de línea. Esta ambigüedad de fase puede no ser un problema en la mayoría de los enlaces de banda base, pero es importante si se modula el código de línea. La fase dividida de espacio (space), es muy similar a la fase dividida de marca (mark), donde se intercambian las funciones de un “1” binarioy un “0”binario.

Un ejemplo de una forma de onda de fase dividida (espacio) codificada se da en la figura 2.1 (h); de nuevo, la primera forma de onda de bits es arbitraria.

2.3.3. Bifásico (Mark)

Este código, designado como Bi ϕ-M, es similar al código de línea Miller (ver sección 2.3.1) en que un “1” binario está representado por una transición de medio bit, y un “0” binario no tiene ninguna transición de medio bit. Sin embargo, este código tiene siempre una transición al comienzo de un periodo de bit [10]. Por lo tanto, el código es fácil de sincronizar y no tiene corriente dc.

Un ejemplo de Bi ϕ-M se ha mostrado en la figura 2.1 (i), donde elegimos arbitrariamente la dirección de la transición en$t = 0$. El código Bifásico (espacio) o Bi ϕ-S es similar a Bi ϕ-M, excepto que el rol de los datos binarios se invierte. Aquí un “0” binario (espacio) produce una transición de medio bit, y un “1” binario no tiene una transición de medio bit. Un ejemplo de forma de onda del código Bi ϕ-S se muestra en la figura 2.1 (j). Tanto Bi ϕ-Scomo Bi ϕ-M son códigos de transición pero con memoria.

2.3.4. Código en Banda Base o CMI.

Este código de línea espropuesto por el Comité Consultivo Internacional Telegráfico y Telefónico (CCITT) como una interfaz de multiplexación y es muy similar a Bi ϕ-S. Un“1” binario se codifica como el código de línea de no retorno a cero (NRZ) con polaridad alternada, es decir,

$+V$ o$-V$. Un“0” binario se codifica con una transición de medio bit definitivo (o fase de onda cuadrada) [1].

Un ejemplo de esta forma de onda se muestra en la figura 2.1 (k) en la que la transición positiva a negativa (o 180° fase) se utiliza para un “0” binario. El nivel de tensión del primer “1” binario en este ejemplo se elige arbitrariamente. Esta forma de onda de ejemplo es idéntico al Bi ϕ-Sque se visualizó en la figura 2.1 (j), excepto para el último bit. CMI tiene buenas propiedades de sincronización y no tiene corriente dc.

2.3.5. Código NRZ (I).

Este tipo de código de línea utiliza una inversión (I) para designar dígitos binarios, específicamente, un cambio en el nivel o ningún cambio. Hay dos variantes de este código, NRZ (M) y NRZ (S) [5, 12]. En NRZ (M), se utiliza un cambio de nivel para indicar un “1” binario, y ningún cambio de nivel se utiliza para indicar un binario 0. En NRZ (S) se utiliza un cambio de nivel para indicar un “0” binario, y el cambio de nivel se utiliza para indicar un “1” binario.

Las formas de onda para NRZ (M) y NRZ (S) se representaron en las figuras2.1 (l) y 2.1 (m), respectivamente, donde se elige el nivel de tensión delprimer “1” binario para el ejemplo de forma de onda arbitraria. Estos códigos son códigos de nivel con memoria.

En general, los códigos de línea que utilizan como una codificación diferencial, como NRZ (I), son insensibles a la ambigüedad de fase a 180°.La recuperación de reloj con NRZ (I) no es particularmente buena, y la fluctuación de fase de dc es lenta, siendo un gran problema. Su ancho de banda es comparable al código polar NRZ.

2.3.6. Binario N Zero sustitución (BNZS)

El código común bipolar AMI tiene muchas propiedades deseables de un código de línea. Su mayor limitación, sin embargo, es que una cadena larga de ceros puede conducir a la pérdida de sincronización y la fluctuación de fase de tiempo porque no hay pulsos en la forma de onda para períodos relativamente largos de tiempo. Por eso, el Binario N de sustitución cero (BNZS) intenta mejorar el código AMI sustituyendo un código especial de longitud N para todas las cadenas de N ceros.

Este código especial contiene pulsos que se parecen a 1s binarios, pero producen deliberadamente violaciones de la convención del código de pulso AMI. Dos pulsos consecutivos de la misma polaridad violan la convención del código de pulso AMI, independiente del número de ceros entre los dos pulsos consecutivos. Estas violaciones se pueden detectar en el receptor, y el código especial es reemplazados por N ceros.

El código especial contiene pulsos que facilitan la sincronización, incluso cuando los datos originales tienen una larga cadena de ceros. Se

elige el código especial de tal manera que las propiedades deseables o familias de codificación se mantienen a pesar de las violaciones de convenciones del pulso AMI, es decir, la capacidad de detección y error de equilibrio de corriente continua. La única desventaja del código BNZS en comparación con AMI es un ligero aumento de la diafonía debido al aumento del número de impulsos y, por lo tanto, un aumento en la energía media en el código.

La elección de los diferentes valores de N produce diferentes códigos BNZS. El valor de N se elige para satisfacer los requisitos de temporización de la aplicación. En telefonía, hay tres códigos BNZS comúnmente utilizados: B6ZS, B3ZS, y B8ZS. Todos los códigos son niveles de códigos BNZS con memoria.En un código B6ZS, una cadena de seis ceros consecutivos se sustituye por uno de los dos códigos especiales de acuerdo a la regla:

Si el último impulso fue positiva (C), el código especial es: 0 + - 0 - +

Si el último impulso fue negativo (-), el código especial es:0 - + 0 + -

Aquí un cero indica un nivel de tensión cero para el período de bits; una ventaja designa un pulso positivo; y un signo menos indica un impulso negativo.Este código especial hace dos violaciones de pulso AMI: en su segunda posición de bit y en su posición de quinto bit. Estas violaciones se detectan fácilmente en el receptor y ceros sustituidos. Si el número de ceros consecutivos es 12, 18, 24,..., la sustitución se repite 2, 3, 4,... veces.

Dado que el número de violaciones es aún, la forma de onda B6ZS es la misma que la forma de onda fuera de la AMI código especial, es decir, entre las secuencias de código especial.

Hay cuatro impulsos introducidos por el código especial que facilita la recuperación de la temporización. Además, tenga en cuenta que el código especial es equilibrado dc. Un ejemplo del código B6ZS se da de la siguiente manera, donde el código especial se indica mediante los caracteres en negrita.

Los datos originales fueron:0 1 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 1 1

Formato B6ZS:0 + 0 + − 0 − + − + 0 − 0 − + 0 + − + −

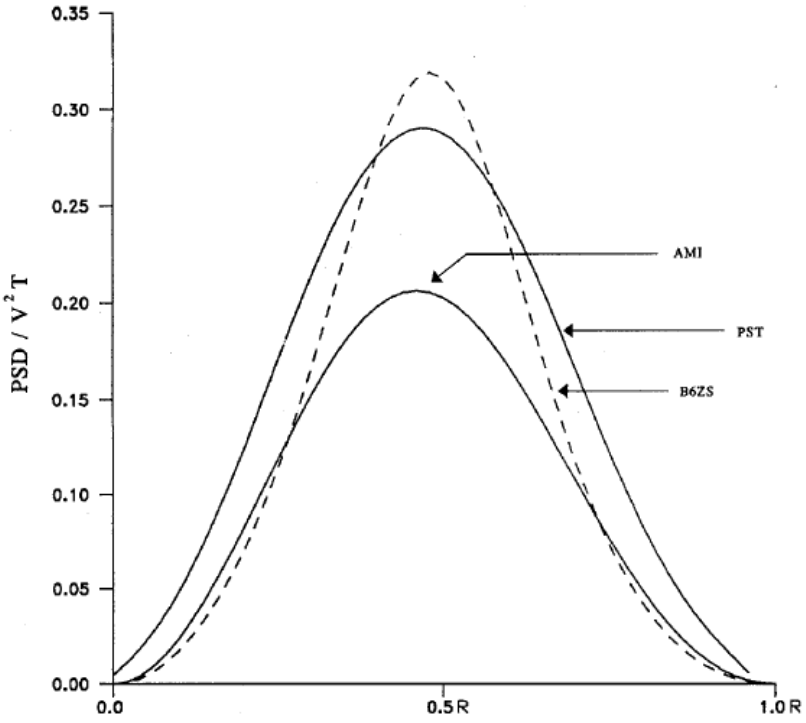

Figura 2. 5: Densidad espectral de potencia para los códigos de línea AMI, PST y B6ZS donde $\boldsymbol{R} = \frac{\mathbf{1}}{\boldsymbol{T}}$ es la tasa de bits.
Fuente: Chitode J., S. (2009).

El cálculo de la PSD de un código B6ZS es tedioso. Su forma se da en la figura. 6.4, a efectos comparativos con AMI, para el caso de los datos igualmente probables.

En un código B3ZS, una cadena de tres ceros consecutivos se sustituye por cualquier código ***B0V*** o ***00V***, donde B se denota como un impulso obedeciendo la convención del código AMI (bipolar) y V denota un pulso que viola la convención del código AMI. Se elige ***B0V*** o ***00V*** de tal manera que el número de (B) entre los pulsos bipolares y de violaciones sea impar. Las reglas B3ZS se resumen en la Tabla 2.1.

Tabla 2. 1: Reglas de Sustitución.

Número de pulsos (B) desde última violación	Polaridad del último pulso (B)	Código de Sustitución	Forma del código de sustitución
Impar (Odd)	Negativo (-)	0 0 -	00V
Impar (Odd)	Positivo (+)	0 0 +	00V
Par (Even)	Negativo (-)	.+ 0 +	B0V
Par (Even)	Positivo (+)	.- 0 -	B0V

Fuente: Chitode J., S. (2009).

De la tabla 2.1 se observa que la violación siempre se produce en la tercera posición de bit del código de sustitución, ypor lo que se puede detectar fácilmente y sustituirlo por el cero realizado en el receptor. Además, la selección de código de sustitución mantiene el equilibrio de corriente continua.Hay uno o dos pulsos en el código de sustitución, lo que facilita la sincronización.

La capacidad de detección de error del código AMI se retiene en B3ZS, porque un solo error de canal haría que la cantidad de pulsos bipolares entre violaciones da a lugar que inclusive sea impar. A diferencia de B6ZS, la forma de onda B3ZS entre violaciones no puede ser la misma que la forma de onda del código AMI.

B3ZS es utilizado en señales digitales-3 (DS-3) de interfaz de señalización en América del Norte, y también para sistemas de larga distancia-4 (LD-4) de transmisión coaxial en Canadá. A continuación se muestra un ejemplo de un código de B3ZS, usando el mismo símbolo descrito en el código B6ZS.

Datos originales:	1	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	1
Formato B3ZS:																		
No. Par de pulsos B:	+	0	0	−	+	**0**	+	−	+	−	**0**	−	0	+	**0**	**0**	+	−
No. Impar de pulsos B:	+	0	0	−	**0**	**0**	−	+	−	+	**0**	+	0	−	**0**	**0**	−	+

El último código BNZS considerado se utiliza $N = 8$, es decir, que dicho código sería B8ZS, muy utilizado para proporcionar canales transparentes para Redes Digitales de Servicios Integrados (RDSI) en las líneas T1 y es similar al código B6ZS. Aquí una cadena de ocho ceros consecutivos se sustituye por uno de los dos códigos especiales de acuerdo a la siguiente regla:

Si el último pulso era positivo (+), el código especial es:	0 0 0 + − 0 − +.
Si el último pulso era negativo (-), el código especial es:	0 0 0 − + 0 + −.

Hay dos violaciones bipolares en los códigos especiales, que ocurren en la cuarta y séptima posición del bit. El código es de corriente dc equilibrada, y se mantiene su capacidad para la detección de errores de AMI. La forma de onda entre sustituciones, es la misma que la de AMI, es decir, que si el número de ceros consecutivos es 16, 24,..., entonces la sustitución se repite 2, 3,..., veces.

2.3.7. De Alta Densidad Bipolar N (HDBN)

Este algoritmo de codificación, es un estándar del CCITT recomendado por la Conferencia Europea de Administraciones de Correos y Telecomunicaciones (CEPT), que es un organismo de normalización europeo. Es bastante similar a la codificación BNZS. Por lo tanto, es un código de nivel con memoria. Siempre que hay una cadena de ceros consecutivos de$N + 1$, éstos son sustituidos por un código especial de longitud$N + 1$que contiene violaciones AMI.

Los códigos específicos son construidos para diferentes valores de N. Un código específico de alta densidad bipolar N (HDBN), como HDB3, se implementan como señales digitales primarias de la CEPT. Es muy similar al código B3ZS. En este código, una cadena de cuatro ceros consecutivos se sustituye por B00V o 000V. Se elige B00V o 000V de tal manera que el número bipolar (B) entre pulsos de violacioneses impar. Las reglas HDB3 se describen en la Tabla 2.2.

Tabla 2. 2: Reglas de Sustitución del código HDB3.

Número de pulsos (B) desde última violación	Polaridad del último pulso (B)	Código de Sustitución	Forma del código de sustitución
Impar (Odd)	Negativo (-)	0 0 0 -	000V
Impar (Odd)	Positivo (+)	0 0 0 +	000V
Par (Even)	Negativo (-)	.+ 0 0 +	B00V
Par (Even)	Positivo (+)	.- 0 0 -	B00V

Fuente: Chitode J., S. (2009).

Aquí la violación siempre se produce en la cuarta posición de bit del código de sustitución, de modo que pueda ser fácilmente detectado y reemplazado de ceroshechos en el receptor. Además, la selección de código de sustitución mantiene el equilibrio de corriente continua. No sea uno o dos pulsos en el código de sustitución facilitando la sincronización. La capacidad de detección de error de AMI se retiene en HDB3 porque un solo error de canal haría que el número de pulsos bipolares entre violaciones incluso en lugar de estar impar.

2.3.8. Codificación ternaria

Muchos esquemas de codificación de línea emplean tres símbolos o niveles de representar sólo un bit de información, como el AMI. En teoría, debe ser posible para transmitir información de manera más eficiente con tres símbolos, específicamente la eficiencia máxima es $\log_2 3 = 1.58$ bits por símbolo.

Alternativamente, la redundancia en el espacio de señal de código puede ser usado para proporcionar un mejor control de error. Se describen a continuación dos ejemplos de codificación ternaria [1, 2]: par seleccionado ternaria (PST) y 4 binarios 3 ternarias (4B3T).

El código PST tiene muchas de las propiedades deseables de códigos de línea, pero su eficiencia de transmisión es todavía de “1” bit por símbolo. Mientras que el código 4B3T también tiene muchas de las propiedades deseables de códigos de línea, pero incrementando la eficiencia de transmisión.

En el código de PST, dos bits consecutivos, denominado par binarios, se agrupan para formar una palabra. Estos pares binarios se asignan palabras de código que consta de dos símbolos ternarios, donde cada símbolo ternario puede ser +, -, o 0, al igual que en el código AMI. Hay nueve posibles palabras de códigos ternarios.

Sin embargo, las palabras de código ternario con elementos idénticos,se evitan, por ejemplo, + +, -, y 00. Las seis palabras de código restantes se transmiten usando dos modos denominados modo positivo (+) y modo negativo (-). Los modos se conmutan cada vez que se transmite una palabra de código con un solo pulso. El código de PST y reglas de cambio de modo se muestran en la tabla 2.3.

Tabla 2. 3: Las palabras de código de asignación PST y las reglas de los cambios de modo.

Par Binario	Palabras de Códigos Ternarios		Modo de Conmutación
	Modo +	Modo –	
11	+ –	+ –	No
10	+ 0	– 0	Si
01	0 +	0 –	Si
00	– +	– +	No

Fuente: Chitode J., S. (2009).

La PST está diseñada para mantener el equilibrio de corriente dc e incluye un fuerte componente de temporización (sincronización).Un inconveniente de este código, es que los bits deben enmarcarse en parejas. En el receptor, una condición de fueradelmarco, se señala cuando las palabras de código ternarias no utilizados son detectadas (++, y 00).

La propiedad de cambio de modo del PST proporciona una capacidad de detección de errores. PDT puede ser clasificado como un código de nivel con la memoria.Si los datos originales para la codificación de PST contienen solamente de 1s o 0s, una secuencia alterna de$+ \ - \ + \ - \cdots$es transmitida.

Como resultado de ello, una condición de fueradelmarco no se puede detectar. Este problema se puede minimizar utilizando el código de PDT modificado como se muestra en la tabla 2.4.

Tabla 2. 4: Modificación de la palabra código de asignación PST y las reglas de los cambios de modo.

Par Binario	Palabras de Códigos Ternarios		Modo de Conmutación
	Modo +	Modo –	
11	+ 0	0 –	Si
10	+ –	+ –	No
01	– +	– +	No
00	0 +	– 0	Si

Fuente: Chitode J., S. (2009).

Es tedioso para derivar el PSD de una forma de onda codificada PST. Una vez más,podemos visualizar en la figura 2.5 la PSD del código PST junto con el PSD de AMI y B6ZS para efectos de comparación, para todos los datos binarios igualmente probables. Obsérvese que la PST tiene más poder que AMI y, por lo tanto, una mayor cantidad de energía por bit, lo que se traduce en ligero aumento de diafonía.

En la codificación 4B3T, las palabras que constan de 4 dígitos binarios se asignan 3 símbolos ternarios. Cuatro bits implican$2^4 = 16$posibles palabras binarias, mientras que tres símbolos ternarios permiten$3^3 = 27$posibles palabras de código ternarios. La conversión binaria a ternario en 4B3T, asegura equilibrio de corriente dc y una fuerte componente de temporización. La asignación específica de palabra de código se muestra en la tabla 2.5.

Tabla 2. 5: Asignación de la palabra de código 4B3T.

Palabras Binarias	Palabras de Códigos Ternarios		
	Columna 1	Columna 2	Columna 3
0000	– – –	0 –	+ + +
0001	– – 0	+ –	+ + 0
0010	– 0 –	– +	+ 0 +
0011	0 – –		0 + +
0100	– – +		+ + –
0101	– + –		+ – +
0110	+ – –		– + +
0111	– 0 0		+ 0 0
1000	0 – 0		0 + 0
1001	0 0 –		0 0 +
1010		0 + –	
1011		0 – +	
1100		+ 0 –	
1101		– 0 +	
1110		+ – 0	
1111		– + 0	

Fuente: Chitode J., S. (2009).

Hay tres tipos de palabras de código (ver tabla 2.5), organizados en tres columnas. Las palabras de código en la primera columna tendráncorriente dc negativa, para las palabras de la segunda columna tiene cero corriente dc, y finalmente los de la tercera columna tienen corriente dc positivo. El codificador controla la variable de número entero, de acuerdo a:

$$I = N_p - N_n$$

donde$N_p$es el número de impulsos positivos de transmisión y$N_n$es el número de impulsos negativos de transmisión. Las palabras en clave se eligen de acuerdo a las siguientes reglas:

a. Si $I < 0$, elegir las palabras de código ternarios de las columnas 1 y 2.
b. Si $I > 0$, elegir las palabras de código ternarios de las columnas 2 y 3.
c. Si $I = 0$, elegir las palabras de código ternarios de la columnas 2, y de la columna 1 si anteriormente $I > 0$ o desde la columna 3 si anteriormente $I < 0$.

Hay que tener en cuenta que la palabra código ternario 000 no se utiliza, pero las 26 palabras de código restantes se utilizan de manera complementaria. Por ejemplo, la palabra de código de la columna 1 para 0001 es – – 0, mientras que la columna 3 de palabra de código es de ++ 0.

La eficiencia de transmisión máxima para el código de 4B3T es 1.33 bits por símbolo en comparación con 1 bit por símbolo para los otros códigos de línea. La desventaja del código 4B3T, es que se requiere de la elaboración y supervisión del rendimiento siendo esto complicado.

El código 4B3T se utiliza en la línea lapso T148, desarrollado por la ITT. Este código permite transmitir 48 canales con sólo el 50% más de ancho de banda que el requerido por las líneas T1, en vez de un 100% más de ancho de banda.

2.4. Multinivel de señalización, señalización de respuesta parcial, y codificación duobinaria.

La codificación ternaria, tal como 4B3T, es un ejemplo de la utilización de más de dos niveles para mejorar la eficiencia de transmisión. Para aumentar la eficiencia de transmisión de otro tipo, se necesita más niveles y/o un mayor procesamiento de la señal. La señalización multinivel, permite una mejora en la eficiencia de la transmisión a expensas de un aumento en la tasa de error, es decir, que se requerirá mayor potencia de transmisión para así mantener una probabilidad de error dada.

En la señalización de respuesta parcial, la interferencia entre símbolos se introduce deliberadamente, mediante el uso de pulsos que son más anchas y, por lo tanto, requiere menos ancho de banda.La cantidad controlada de interferencia de cada pulso se puede quitar en el receptor. Esto mejora la eficiencia de la transmisión, a expensas de una mayor complejidad.

La codificación duobinaria, un caso especial de la señalización de respuesta parcial, requiere sólo el ancho de banda teórico mínimo de$0.5R\ Hz$. En las siguientes secciones se discuten brevemente estas técnicas con poco más detalle.

2.4.1. Multinivel de señalización.

El número de niveles que se puede utilizar para un código de línea no está limitado a dos o tres. Dado que más niveles o símbolos permiten una

mayor eficiencia de transmisión, la señalización multinivel se puede considerar en aplicaciones de ancho de banda limitado. Específicamente, si la tasa de baudios o tasa de señalización esR_s y el número de niveles utilizados es$L$, entonces la tasa de bits de transmisión equivalenteR_b está dada por:

$$R_b = R_s \log_2(L)$$

Alternativamente, la señalización multinivel se puede utilizar para reducir la velocidad de transmisión, que a su vez puede reducir la diafonía para la misma velocidad de bits en forma equivalente. La sanción, sin embargo, es que la relación señal/ruido (SNR) debe aumentarse para lograr la misma tasa de error.

El sistema de soporte T1G de AT & T utiliza la señalización multinivel con $L = 4$y una velocidad de transmisión de 3.152 mega-símbolos/s al doble de la capacidad del sistema T1C de 48 canales a 96 canales. Además, un esquema de señalización de cuatro niveles en 80-kB se utiliza para lograr 160 kbps como una tasa básica en un bucle de abonado digital (DSL) para RDSI O ISDN.

2.4.2. Señalización de respuesta parcial y Codificación duobinaria

Esta clase de la señalización también se llama codificación correlativa porque introduce a propósito una cantidad controlada o correlacionada de interferencia entre símbolos en cada símbolo. En el receptor, la cantidad

conocida de interferencia se elimina eficazmente de cada símbolo. La ventaja de esta señalización es que los pulsos más amplios pueden ser utilizados requiriendo menos ancho de banda, pero la SNR debe incrementarse para realizar una tasa de error dada. Además, los errores se pueden propagar a menos que se utiliza de precodificación.

Hay muchos esquemas de señalización parciales de uso común, a menudo descrito en términos del operador D retraso, que representa la señalización intervalo de retraso. Por ejemplo, en (1 + D) la señalización del impulso de corriente y el impulso anterior se añaden. El sistema T1D de usos de AT & T (1+D) de señalización con la pre-codificación, se refirió a la señalización como duobinaria, para convertir los datos binarios (de dos niveles) en datos ternarios (tres niveles) en la misma proporción.Esto requiere que el ancho de banda mínimo del canal teórico sin los efectos perjudiciales de la interferencia entre símbolos evite la propagación de errores. Los detalles completos respecto a la codificación duobinaria se encuentran en Lender, 1963 y Schwartz, 1980. Alguna respuesta parcial de esquemas de señalización, tales como (1-D), se utilizan para dar forma a la banda ancha en lugar de controlarlo. Otro ejemplo interesante de la codificación duobinaria es un $(1-D^2)$, que puede ser analizada como el producto (1- D) (1 + D). Es utilizado por los GTE en su sistema de soporte T modificado. AT & T también utiliza $(1-D^2)$ con cuatro niveles de entrada para lograr una velocidad de datos equivalente a 1.544 Mb / s en sólo un ancho de banda 0.5 MHz.

2.5. Ancho de banda de Comparación

Hemos proporcionado las expresiones PSD para la mayoría de los códigos de línea comúnmente utilizados. El requisito de ancho de banda real, sin embargo, depende de la forma del pulso utilizado y la definición de ancho de banda propia. Hay muchas maneras de definir el ancho de banda, por ejemplo, como un porcentaje de la potencia total o la supresión de lóbulo lateral en relación con el lóbulo principal. Usando el primer cero del PSD del código como la definición de ancho de banda, el cuadro 6.6 presenta una comparación ancho de banda útil.

Tabla 2. 6: Comparación del primer ancho de banda nula.

Bandwidth	Codes	
	Unipolar NRZ	BNZS
R	Polar NRZ	HDBN
	Polar RZ (AMI)	PST
$2R$	Unipolar RZ	Split Phase
	Manchester	CMI

Fuente: Chitode J., S. (2009).

La omisión notable en la Tabla 6.6 es la modulación de retardo (código de Miller). No tiene una primera nula en la banda de 2R-Hz, pero la mayor parte de su poder está contenido en menos de 0.5R Hz.

CAPÍTULO 3: INTERFAZ GRÁFICA DE USUARIO – GUI.

3.1. Introducción a la interfaz gráfica de usuario – GUI.

GUI es una interfaz gráfica de usuario, en otras palabras, es una visualización gráfica en una o más ventanas, las mismas disponen de controles, conocidos como componentes, lo que permitea los usuarios realizar tareas interactivas. El usuario de la interfaz gráfica de usuario no tiene que crear un script o escribir comandos en la línea de comandos para realizar las tareas.

A diferencia de los programas de codificación para realizar las tareas, el usuario de una interfaz gráfica de usuario no necesita entender los detalles de cómo se realizan las tareas. Los componentes de los GUIs incluyen menús (pantalla principal), barras de herramientas, botones, botones de opción, cuadros de lista y deslizadores, sólo para nombrar unos pocos.

Los GUIs se desarrollan utilizando la plataforma MatLab, en la cual permite realizar cualquier clase de cálculo, leer y escribir archivos de datos, así como comunicarse con otras interfaces gráficas de usuario, y mostrar datos como tablas o como parcelas. En la figura 3.1 se ilustra una interfaz gráfica de usuario simple que los usuarios (estudiantes, docentes, investigadores, etc.) pueden construir fácilmente.

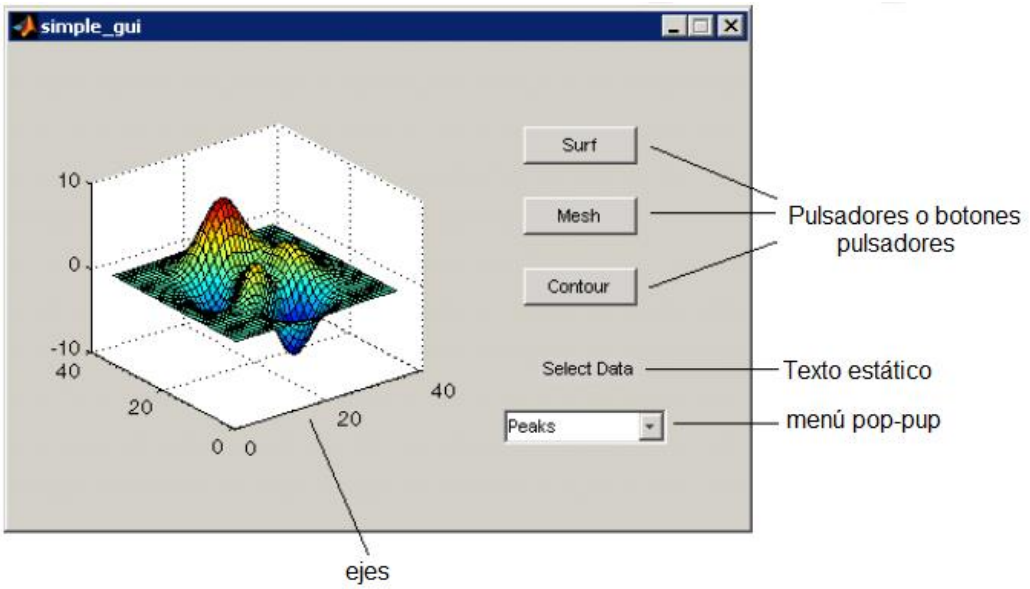

Figura 3. 1: Aplicación realizada en el GUI de MatLab.
Fuente: http://www.mathworks.com/products/matlab/

La interfaz gráfica de usuario contiene:

a. Un componente para los ejes, ya sea 2-D o 3-D.
b. Un menú pop-up, en la cual lista tres conjuntos de datos que corresponden a las funciones de MATLAB.
c. Un componente de texto estático para etiquetar el menú pop-up.
d. Tres botones que proporcionan diferentes tipos de parcelas: superficie, malla, y nivel de curvas.

3.2. Creación de interfaz GUI sencilla.

En esta apartado, se mostrará el desarrollo para la crear interfaces GUIs, tal como se observó la figura 3.1. Es decir, que se guiará en el proceso de creación de GUIs. Si se prefiere ver y ejecutar el código que se ha creado la GUI, establezca una carpeta a uno a los que tiene acceso de

escritura. Copiamos el código de ejemplo y abrimos en el editor el siguiente comando en MATLAB:

```
copyfile(fullfile(docroot, 'techdoc','creating_guis',...
  'examples','simple_gui2*.*')), fileattrib('simple_gui2*.*', '+w');
edit simple_gui2.m
```

3.2.1. Creación de archivos de código de programación de GUIs.

Crear un archivo de función (a diferencia de archivos script, que contiene una secuencia de comandos de MatLab, pero no define funciones):

1. En MATLAB, escribir edit.
2. Escribir la siguiente instrucción en la primera línea del editor:

   ```
   function simple_gui2
   ```

3. A raíz de la declaración de función, escriba estos comentarios, que termina con una línea en blanco.

   ```
   % SIMPLE_GUI2 Select a data set from the pop-up menu, then
   % click one of the plot-type push buttons. Clicking the button
   % plots the selected data in the axes.
   (Leave a blank line here)
   ```

4. Al final del archivo, después de la línea en blanco, agregar una declaración final.
5. Guardar el archivo en la carpeta actual o en un lugar que está guardada en la ruta de MATLAB.

3.2.2. Creación de figuras de interfaz gráfica de usuario simple.

Añada las siguientes líneas antes de la declaración final de su archivo para crear una figura y colocarla en la pantalla. (En el software MATLAB, una interfaz gráfica de usuario es una figura.)

```
%  Create and then hide the GUI as it is being constructed.
f = figure('Visible','off','Position',[360,500,450,285]);
```

Las llamadas funciones de la figura 3.1, se utiliza dos pares de propiedad o valor, que son:

a. La propiedad *"Visible"*, hace que la GUI sea invisible para que el usuario GUI no pueda ver los componentes que se agregan o se inicializan.
b. Cuando la interfaz GUIs tiene todos sus componentes y se inicializa, el ejemplo hace que sea visible. Las propiedades *"Position"* es un vector de cuatro elementos que especifica la ubicación y tamaño de la GUI en la pantalla, bajo la siguiente estructura: [distancia izquierda, la distancia inferior, ancho, alto]. Por defecto las unidades son píxeles.

3.2.3. Agregar componentes de la interfaz gráfica de usuario simple.

En esta sección, se podrá agregar los botones (pulsadores), texto estático, menú pop-up, y los ejes de componentes rectangulares, cilíndricas o esféricas para una interfaz gráfica (GUI), y se debe seguir los siguientes pasos:

1. Para añadir estas declaraciones a su archivo de código, creamos tres componentes de botón pulsador.

```
% Construct the components.
hsurf    = uicontrol('Style','pushbutton',...
             'String','Surf','Position',[315,220,70,25]);
hmesh    = uicontrol('Style','pushbutton',...
             'String','Mesh','Position',[315,180,70,25]);
hcontour = uicontrol('Style','pushbutton',...
             'String','Countour','Position',[315,135,70,25]);
```

Cada instrucción utiliza una serie de pares (propiedad/valor) de ***uicontrol***, para definir un botón o pulsador:

- La propiedad de estilo especifica que el ***uicontrol*** es un botón pulsador.
- La propiedad ***String*** especifica la etiqueta de cada botón: *Surf*, *Mesh*, y *Countour*.
- La propiedad de posición, especificala ubicación y tamaño de cada botón dentro de la GUI en la pantalla, bajo la siguiente estructura: [distancia izquierda, la distancia inferior, ancho, alto]. Las unidades por defecto para pulsadores son píxeles.

2. Añadir el menú pop-up y su etiqueta de texto estática al GUI, mediante la adición de declaraciones en el archivo de código, lasiguiente definición del pulsador. La primera sentencia crea un menú emergente y la segunda sentencia crea un componente de texto que sirve como una etiqueta para el menú pop-pup.

```
hpopup = uicontrol('Style','popupmenu',...
           'String',{'Peaks','Membrane','Sinc'},...
           'Position',[300,50,100,25]);
htext  = uicontrol('Style','text','String','Select Data',...
           'Position',[325,90,60,15]);
```

La propiedad *String*del componente menú pop-pup, utiliza una matriz de celdas para especificar los tres elementos en el menú pop-pup, tal como: *Peaks*, *Membrane* y *Sinc*.La componente del texto estático, la propiedad String especifica instrucciones para el usuario GUI.

Para ambos componentes, la propiedad de posición especifica la ubicación y tamaño de cada componente dentro de la GUI, bajo la siguiente estructura: [distancia izquierda, la distancia parte inferior, ancho, alto]. Unidades por defecto para los componentes son píxeles.

3. Crear los ejes a la interfaz gráfica de usuario mediante la adición de esta declaración en el fichero de código.

```
ha = axes('Units','pixels','Position',[50,60,200,185]);
```

La propiedad *"Units"* especifica las unidades en píxeles, para que los ejes tengan las mismas unidades que los otros componentes.

4. A raíz de todas las definiciones de componentes, agregamos esta línea al archivo de código para alinear todos los componentes, excepto los ejes, a lo largo de sus centros.

```
align([hsurf,hmesh,hcontour,htext,hpopup],'Center','None');
```

5. Añadir este comando después del comando de alineación.

```
%Make the GUI visible.
set(f,'Visible','on')
```

Su archivo de código debería tener este aspecto:

```
function simple_gui2
% SIMPLE_GUI2 Select a data set from the pop-up menu, then
% click one of the plot-type push buttons. Clicking the button
% plots the selected data in the axes.

   %  Create and then hide the GUI as it is being constructed.
   f = figure('Visible','off','Position',[360,500,450,285]);
```

```
%  Construct the components.
hsurf = uicontrol('Style','pushbutton','String','Surf',...
        'Position',[315,220,70,25]);
hmesh = uicontrol('Style','pushbutton','String','Mesh',...
        'Position',[315,180,70,25]);
hcontour = uicontrol('Style','pushbutton',...
        'String','Countour',...
        'Position',[315,135,70,25]);
htext = uicontrol('Style','text','String','Select Data',...
        'Position',[325,90,60,15]);
hpopup = uicontrol('Style','popupmenu',...
        'String',{'Peaks','Membrane','Sinc'},...
        'Position',[300,50,100,25]);
ha = axes('Units','Pixels','Position',[50,60,200,185]);
align([hsurf,hmesh,hcontour,htext,hpopup],'Center','None');

%Make the GUI visible.
set(f,'Visible','on')

end
```

6. Se ejecuta el código escribiendo ***simple_gui2*** en el *prompt*de MatLab, y se mostrará (véase figura 3.2) la GUI creada.

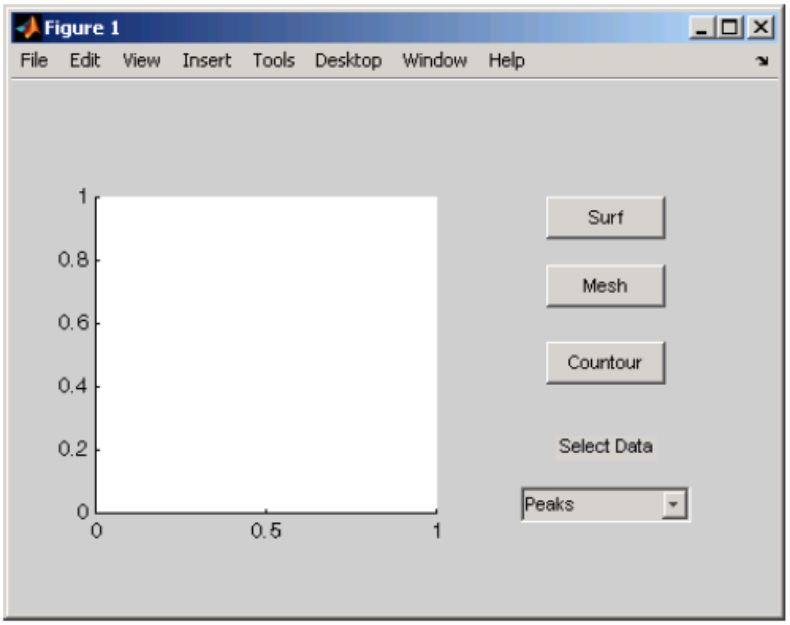

Figura 3. 2: Ventana final desarrollada en GUI de MatLab.
Fuente: http://www.mathworks.com/products/matlab/

Podemos seleccionar un conjunto de datos en el menú pop-pup, y damos clic en los pulsadores (botoneras), sin pasar nada. Esto se debe a que no existe el código ***callback*** (devolución de llamada) en el archivo al servicio en el menú pop-up o de la botonera.

3.3. Código de programación de un GUI.

El código de programación del menu pop-pup, permite a los usuarios seleccionar datos a la trama. Cuando se selecciona uno de los tres conjuntos de datos del GUI en el pop-pup, MatLab establece la propiedad ***Value***, para el índice de la cadena seleccionada. El menú pop-pup callback, lee la propiedad ***Value***, y así determinar qué elemento se está mostrando actualmente y por consecuencia establecido por current_data.

```
%  Pop-up menu callback. Read the pop-up menu Value property to
%  determine which item is currently displayed and make it the
%  current data. This callback automatically has access to
%  current_data because this function is nested at a lower level.
   function popup_menu_Callback(source,eventdata)
      % Determine the selected data set.
      str = get(source, 'String');
      val = get(source,'Value');
      % Set current data to the selected data set.
   switch str{val};
   case 'Peaks' % User selects Peaks.
      current_data = peaks_data;
   case 'Membrane' % User selects Membrane.
      current_data = membrane_data;
   case 'Sinc' % User selects Sinc.
      current_data = sinc_data;
   end
end
```

CAPÍTULO 4:DISEÑO Y EVALUACIÓN DE LOS CÓDIGOS DE LÍNEA

4.1. Diseño de códigos de línea.

En esta sección se desarrollará una herramienta de interfaz gráfica de usuario (GUI o GUIDE) en la cual se programarán cada uno de los siguientes códigos de línea: Unipolares NRZ y RZ, Polar NRZ, Bipolar RZ, AMI NRZ, AMI RZ y Manchester NRZ; con la única finalidad de simular. Además, se podrá generar aleatoriamente 10 bits y que el usuario elija la codificación, posteriormente se podrá mostrar las densidades espectrales de potencia. En la figura 4.1 se muestran algunos de los códigos de líneas que serán programadas dentro de la GUI.

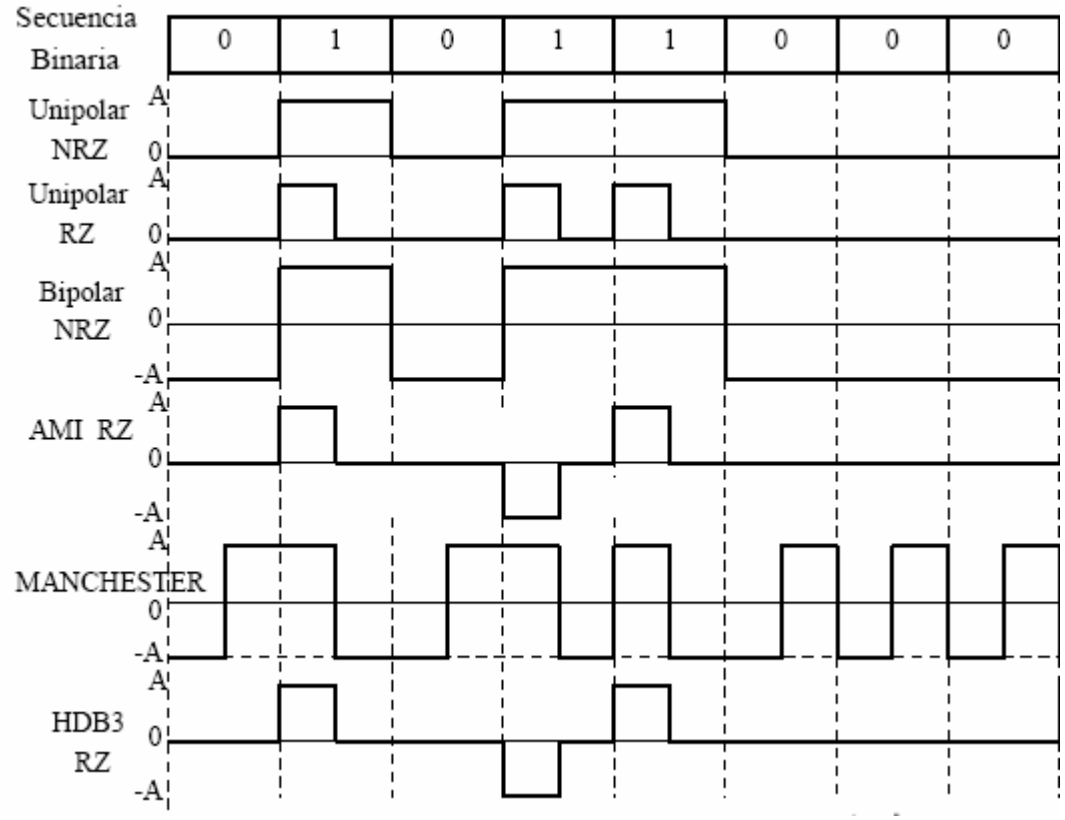

Figura 4. 1: Códigos de línea a simular excepto HDB3 RZ.
Elaborado por el Autor.

4.1.1. Diseño de la GUI principal para los códigos de línea.

En la figura 4.2 se muestra el diseño de la ventana (pantalla) principal, en la cual se observan los datos binarios aleatorios, la selección de los códigos de líneas (ver en Unipolar NRZ), el *axes1* que permitirá graficar el código seleccionado y el botón que permitirá obtener la señal de la densidad espectral de potencia de cada código.

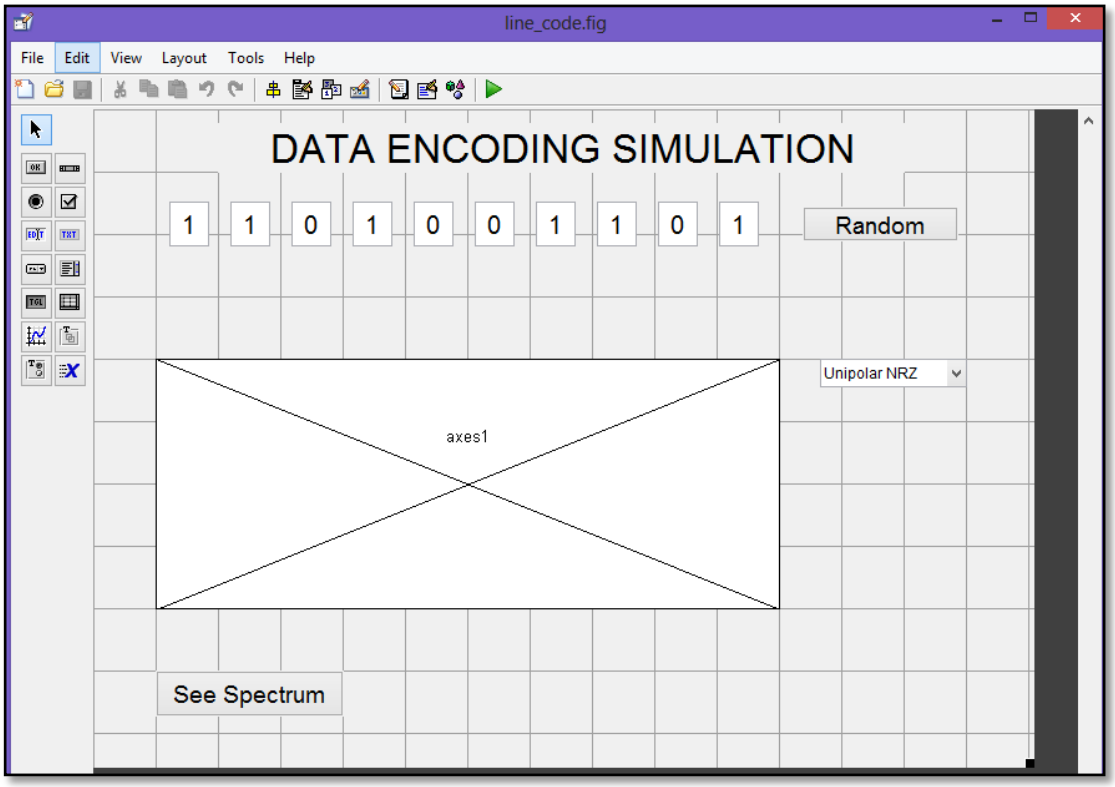

Figura 4. 2: Ventana GUI para códigos de línea.
Elaborado por el Autor.

4.1.2. Diseño de la GUI para generar la densidad de potencia espectral.

En la figura 4.3 se muestra el diseño de la ventana (pantalla) que permitirá obtener la señal de densidad espectral de potencia, en la misma se observan los códigos de líneas que serán seleccionados para así visualizar dichas señales espectrales.

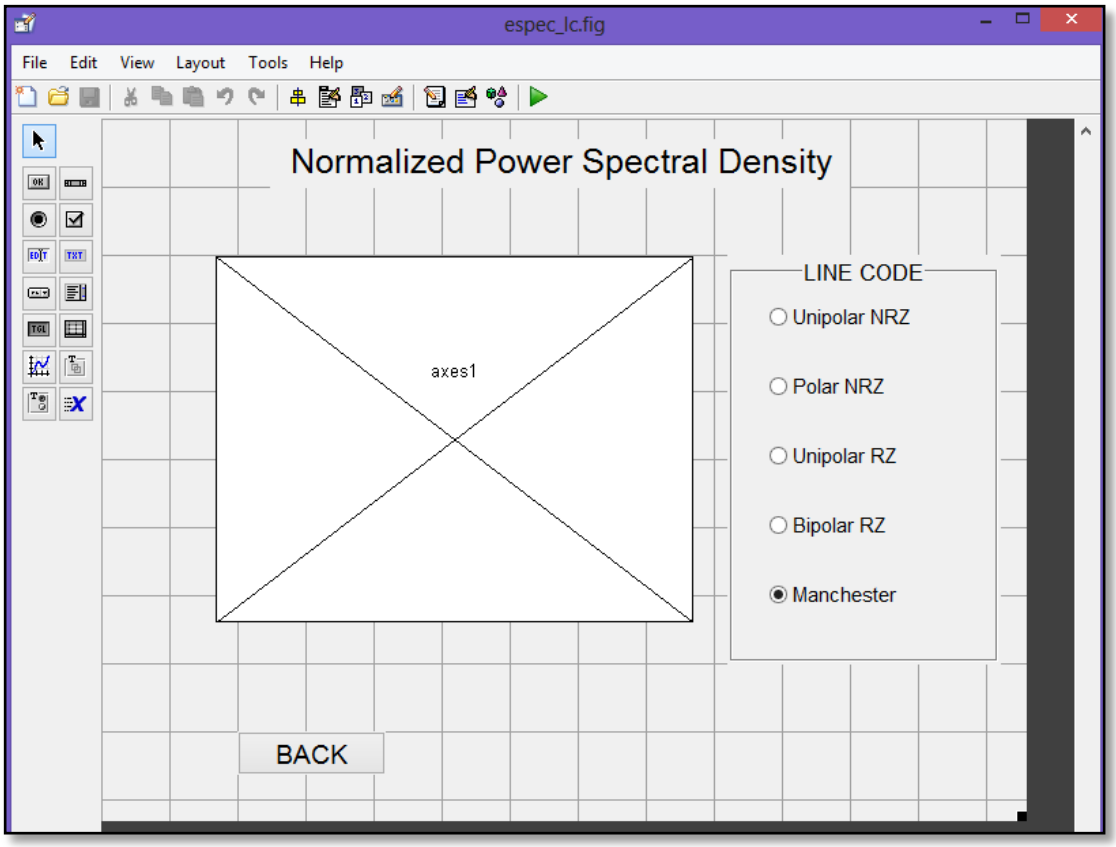

Figura 4. 3: Ventana GUI para generar la densidad de potencia espectral.
Elaborado por el Autor.

4.2. Programación de las GUIs – Códigos de Línea y PSD.

En esta sección se desarrollará la programación de cada GUI que se mostraron en las figuras 4.2 y 4.3, es decir, que se realizará la programación de cómo generar los bits de manera aleatoria, de generar los códigos de líneas escogidos para ser simulados y la obtención de las señales de densidad de potencia espectral.

4.2.1. Programa para generar los Bits aleatorios.

A continuación se muestra el código de programación que permite generar aleatoriamente los 10 bits.

```
function line_code_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to line_code (see VARARGIN)

hold off;
        h=[1 1 0 1 0 0 1 1 0 1];
        n=1;
        h(11)=1;
        while n<=10;
            t=n-1:0.001:n;
        if h(n) == 0
            if h(n+1)==0
                y=(t>n);
            else
                y=(t==n);
            end
            d=plot(t,y);title('Codificación UNIPOLAR NRZ');grid on
            set(d,'LineWidth',2.5);
            hold on;
            axis([0 10 -1.5 1.5]);
        else
            if h(n+1)==0
                y=(t<n)-0*(t==n);
            else
                y=(t<n)+1*(t==n);
            end
            d=plot(t,y);title('Code UNIPOLAR NRZ');grid on;
            set(d,'LineWidth',2.5);
            hold on;
            axis([0 10 -1.5 1.5]);
        end
        n=n+1;
        end
```

Ahora se muestra el código de programación que activará el botón o pulsador *Random* (ver figura 4.3) lo que posteriormente llamará a la selección de la línea de código y mostrarán las gráficas de cada uno de los códigos de líneas escogidas en el presente trabajo de titulación.

```
% --- Executes on button press in random.
function random_Callback(hObject, eventdata, handles)
% hObject    handle to random (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
a=round(rand(1,1));
b=round(rand(1,1));
c=round(rand(1,1));
d=round(rand(1,1));
e=round(rand(1,1));
f=round(rand(1,1));
g=round(rand(1,1));
h=round(rand(1,1));
i0=round(rand(1,1));
j0=round(rand(1,1));
ran=[a,b,c,d,e,f,g,h,i0,j0];
set(handles.uno,'String',ran(1));
set(handles.dos,'String',ran(2));
set(handles.tres,'String',ran(3));
set(handles.cuatro,'String',ran(4));
set(handles.cinco,'String',ran(5));
set(handles.seis,'String',ran(6));
set(handles.siete,'String',ran(7));
set(handles.ocho,'String',ran(8));
set(handles.nueve,'String',ran(9));
set(handles.diez,'String',ran(10));

%-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
handles.bits=[a,b,c,d,e,f,g,h,i0,j0];
cod=get(handles.select_code,'Value');
switch cod
    case 1
        hold off;
        h=handles.bits;
        n=1;
        h(11)=1;
        while n<=10;
            t=n-1:0.001:n;
        if h(n) == 0
            if h(n+1)==0
                y=(t>n);
            else
                y=(t==n);
            end
            d=plot(t,y);title('Code UNIPOLAR NRZ');grid on
            set(d,'LineWidth',2.5);
```

```
        hold on;
        axis([0 10 -1.5 1.5]);
    end
    n=n+1;
    end
case 2
    hold off;
    h =handles.bits;
    n=1;
    h(11)=1;
    while n<=10;
        t=n-1:0.001:n;
    if h(n) == 0
        if h(n+1)==0
            y=-(t<n)-(t==n);
        else
            y=-(t<n)+(t==n);
        end
        d=plot(t,y);title('Code POLAR NRZ');grid on
        set(d,'LineWidth',2.5);
        hold on;
        axis([0 10 -1.5 1.5]);
    else
        if h(n+1)==0
            y=(t<n)-1*(t==n);
        else
            y=(t<n)+1*(t==n);
        end
        d=plot(t,y);title('Code POLAR NRZ');grid on;
        set(d,'LineWidth',2.5);
        hold on;
        axis([0 10 -1.5 1.5]);
    end
    n=n+1;
    end

case 3
    hold off;
    h =handles.bits;
    n=1;
    h(11)=1;
    while n<=10;
        t=n-1:0.001:n;
    %Graficación de los CEROS (0)
        if h(n) == 0
            if h(n+1)==0
                y=(t>n);
```

```
            else
                y=(t==n);
            end
        d=plot(t,y);title('Code UNIPOLAR RZ');grid on
        set(d,'LineWidth',2.5);
        hold on;
        axis([0 10 -1.5 1.5]);
    %Graficación de los UNOS (1)
        else
        if h(n+1)==0
            y=(t<n-0.5);
        else
            y=(t<n-0.5)+1*(t==n);
        end
        d=plot(t,y);title('Code UNIPOLAR RZ');grid on;
        set(d,'LineWidth',2.5);
        hold on;
        axis([0 10 -1.5 1.5]);
        end
    n=n+1;

    end
```

Solo se han mostrado las configuraciones de los tres primeros códigos que dispone el GUI, los otros son similares pero con otras características. Finalmente se configura el botón para obtener la PSD.

```
% --- Executes on button press in espectro.
function espectro_Callback(hObject, eventdata, handles)
% hObject    handle to espectro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close line_code;
espec_lc
```

4.2.2. Programa para generar las señales PSD.

En esta sección se desarrollará el programa que permite obtener las señales de la densidad espectral de potencia. A continuación se muestra el código que permite la llamada desde la ventana principal (ver figura 4.2)

```
function varargout = espec_lc(varargin)
%Simulation of Line Codes
%Author: Miguel Santiago Fajardo Brito
%Long Live Judas Priest

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @espec_lc_OpeningFcn, ...
                   'gui_OutputFcn',  @espec_lc_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before espec_lc is made visible.
function espec_lc_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to espec_lc (see VARARGIN)
    hold off;
    A=1;
    Tb=1.5;
    R=1/Tb;
    L=2*R;
    f=0:L/50:L;
    P=(A.^2*Tb)*(sinc(f*Tb/2)).^2.*(sin(pi*f*Tb/2)).^2;
    g=plot(f,P);
    title('ESPECTRAL DENSITY: MANCHESTER NRZ');
    hold on;xlabel('Frequency');ylabel('Normalized Power');
    axis([0 L 0 1.1*Tb]);set(g,'LineWidth',2.5);
    set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
    set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
    set(gca,'XTickLabel',{['R'];['2R']})
    set(gca,'YTickLabel',{['0.5*Tb'];['Tb']})
% Choose default command line output for espec_lc
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes espec_lc wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

A continuación se muestran las líneas de códigos de programación para obtener las densidades de potencias espectrales de cada uno de los códigos de líneas propuestos en el presente trabajo de titulación:

```
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to uipanel1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
if (hObject==handles.Unipolar_NRZ)
    hold off;
    A=sqrt(2);
    Tb=1.5;
    R=1/Tb;
    L=2*R;
    f=0:L/50:L;
    del=0;
    P=(A.^2*Tb)/4*(sinc(f*Tb)).^2*(1+(1/Tb)*del);
    g=plot(f,P);
    title('ESPECTRAL DENSITY: UNIPOLAR NRZ');
    hold on;xlabel('Frequency');ylabel('Normalized Power');
    axis([0 L 0 1.1*Tb]);set(g,'LineWidth',2.5);
    stem(0,(A.^2*Tb)/2,'LineWidth',2.5);hold off;
    axis([0 L 0 1.09*Tb]);set(g,'LineWidth',2.5);
    set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
    set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
    set(gca,'XTickLabel',{'R =';'2R'})
    set(gca,'YTickLabel',{'0.5*Tb';'Tb'})

elseif(hObject==handles.Polar_NRZ)
    hold off;
    A=1;
    Tb=1.5;
    R=1/Tb;
    L=2*R;
    f=0:L/50:L;
    del=0;
    P=(A.^2*Tb)*(sinc(f*Tb)).^2;
```

```
    g=plot(f,P);hold on;xlabel('Frequency');
    ylabel('Normalized Power');
    title('ESPECTRAL DENSITY: POLAR NRZ')
    axis([0 L 0 1.01*Tb]);set(g,'LineWidth',2.5);
    set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
    set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
    set(gca,'XTickLabel',{['R'];['2R']})
    set(gca,'YTickLabel',{['0.5*Tb'];['Tb']})

elseif(hObject==handles.Unipolar_RZ)
    hold off;
    A=2;
    Tb=1;
    R=1/Tb;
    L=2*R;
    f=0:L/50:L;
    del=0;
    P=(A.^2*Tb)/16*(sinc(f*Tb/2)).^2;
    g=plot(f,P);
    title('ESPECTRAL DENSITY: UNIPOLAR NRZ');
    hold on;xlabel('Frequency');ylabel('Normalized Power');
    axis([0 L 0 1.1*Tb]);set(g,'LineWidth',2.5);
    stem([0 R],[(A.^2*Tb)/8 P(26)+0.1],'LineWidth',2.5);hold off;
    set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
    set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
    set(gca,'XTickLabel',{['R'];['2R']})
    set(gca,'YTickLabel',{['0.5*Tb'];['Tb']})
elseif(hObject==handles.Bipolar_RZ)
    hold off;
    A=2;
    Tb=1.5;
    R=1/Tb;
    L=2*R;
    f=0:L/50:L;
    P=(A.^2*Tb)/8*(sinc(f*Tb/2)).^2.*(1-cos(2*pi*f*Tb));
    g=plot(f,P);
    title('ESPECTRAL DENSITY: BIPOLAR RZ');
    hold on;xlabel('Frequency');ylabel('Normalized Power');
    axis([0 L 0 1.1*Tb]);set(g,'LineWidth',2.5);
    set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
    set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
    set(gca,'XTickLabel',{['R'];['2R']})
    set(gca,'YTickLabel',{['0.5*Tb'];['Tb']})

else
    hold off;
    A=1;
    Tb=1.5;
```

```
R=1/Tb;
L=2*R;
f=0:L/50:L;
P=(A.^2*Tb)*(sinc(f*Tb/2)).^2.*(sin(pi*f*Tb/2)).^2;
g=plot(f,P);
title('ESPECTRAL DENSITY: MANCHESTER NRZ');
hold on;xlabel('Frequency');ylabel('Normalized Power');
axis([0 L 0 1.1*Tb]);set(g,'LineWidth',2.5);
set(gca,'XTickMode','manual','XTick',[R,2*R]);grid on;
set(gca,'YTickMode','manual','YTick',[0.5*Tb,Tb]);
set(gca,'XTickLabel',{['R'];['2R']})
set(gca,'YTickLabel',{['0.5*Tb'];['Tb']})

end
```

4.3. Resultados obtenidos de la simulación de códigos de línea.

Aquí se muestran los resultados que se obtienen de la simulación de los códigos de línea mencionados anteriormente y se obtienen las densidades espectrales de potencia. En la figura 4.4 se muestra la señal del código Unipolar NRZ de los bits aleatorios.

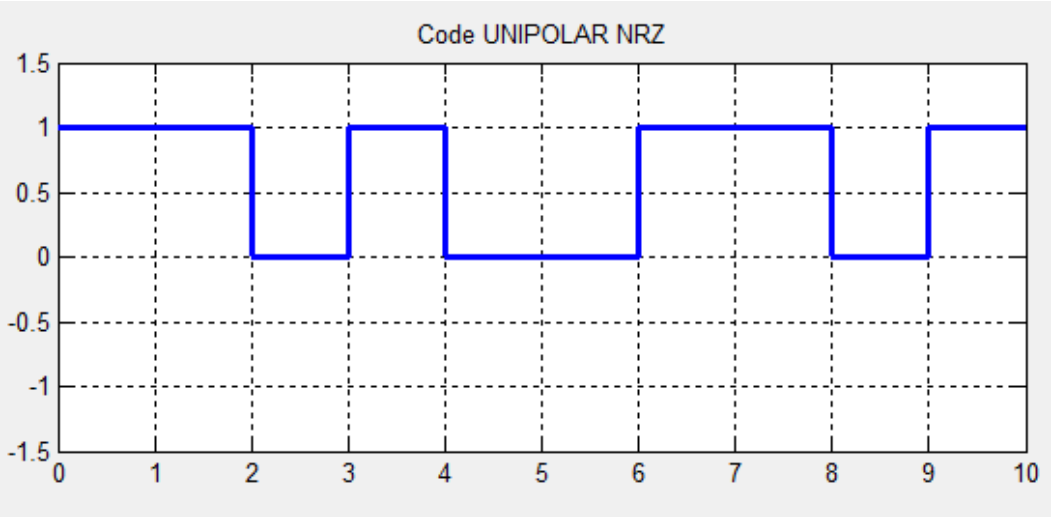

Figura 4. 4: Generación de bits del código Unipolar NRZ.
Elaborado por el Autor.

Una vez generado el código, se debe pulsar *See Spectrum*, para mostrar la densidad espectral de la potencia (véase la figura 4.5).

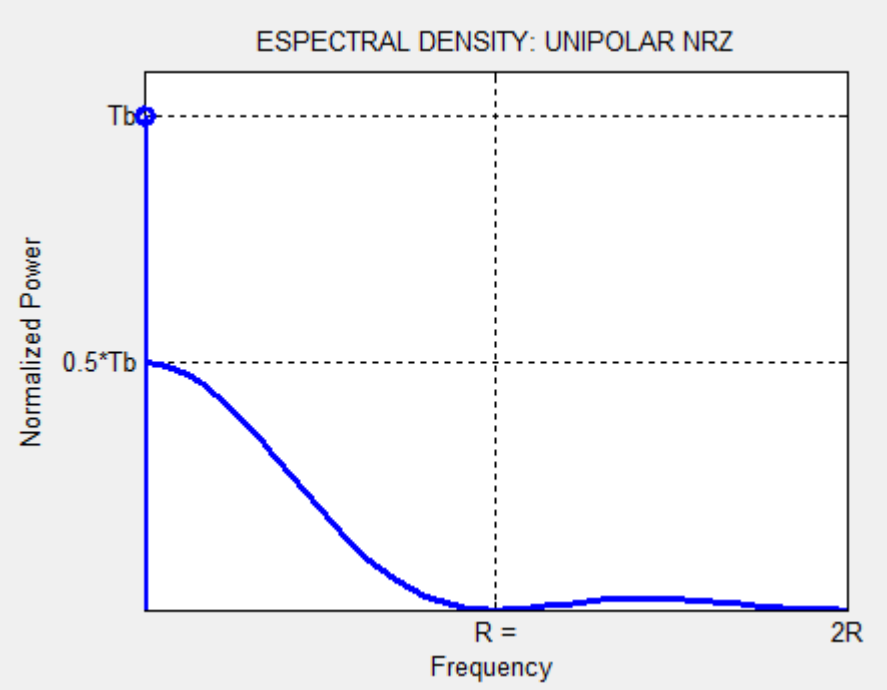

Figura 4. 5: Densidad espectral de potencia para Unipolar NRZ.
Elaborado por el Autor.

En la figura 4.6 se muestra la señal de bits para el código Polar NRZ obtenida aleatoriamente. En la figura 4.7 se muestra la densidad espectral de potencia del código Polar NRZ.

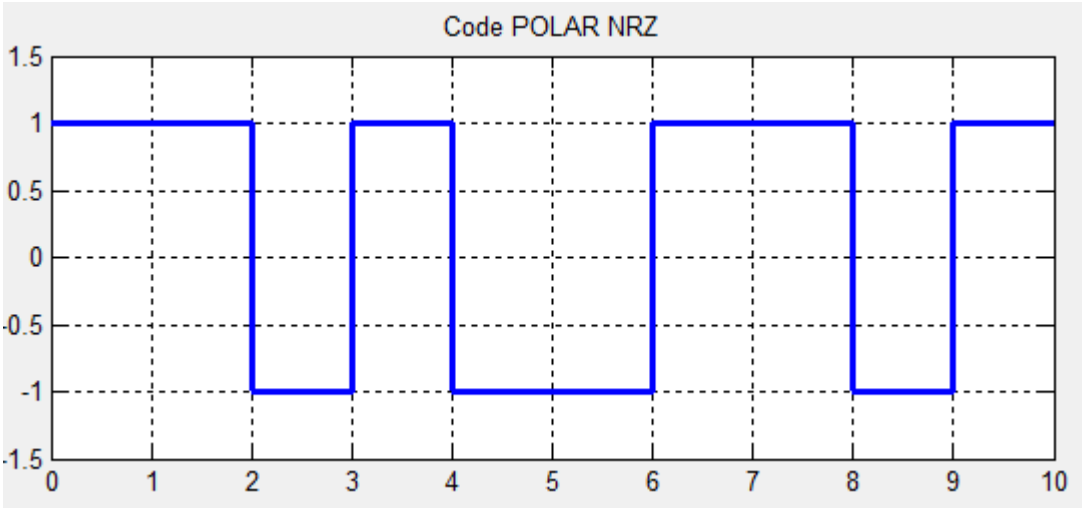

Figura 4. 6: Generación de bits del código Polar NRZ.
Elaborado por el Autor.

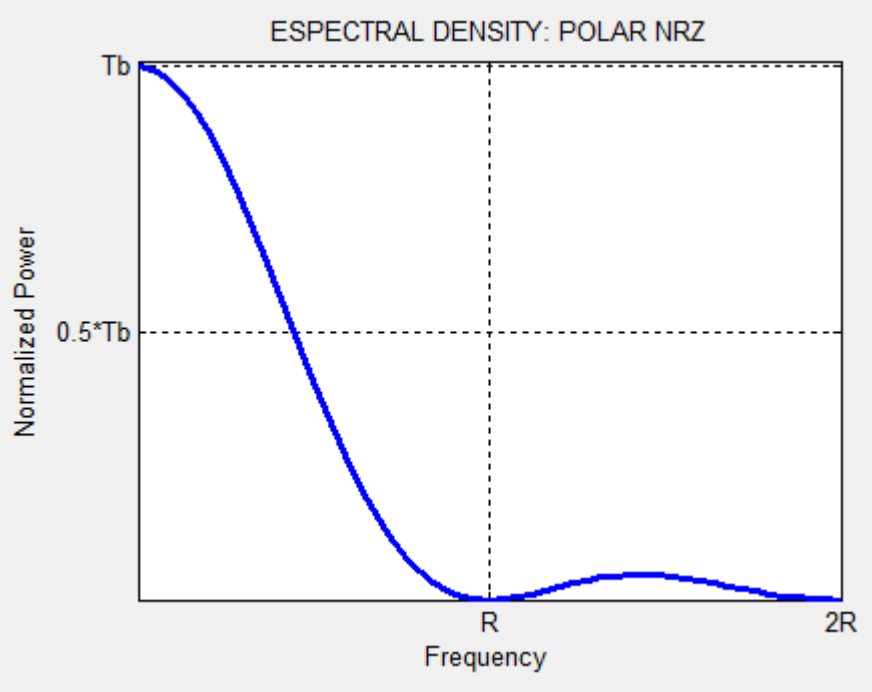

Figura 4. 7: Densidad espectral de potencia para Unipolar NRZ.
Elaborado por el Autor.

La generación de bits de los códigos de líneas restantes se puede visualizar en la figura 4.8 (Unipolar RZ), figura 4.9 (Bipolar RZ), figura 4.10 (AMI NRZ), figura 4.11 (AMI RZ) y figura 4.12 (Manchester).

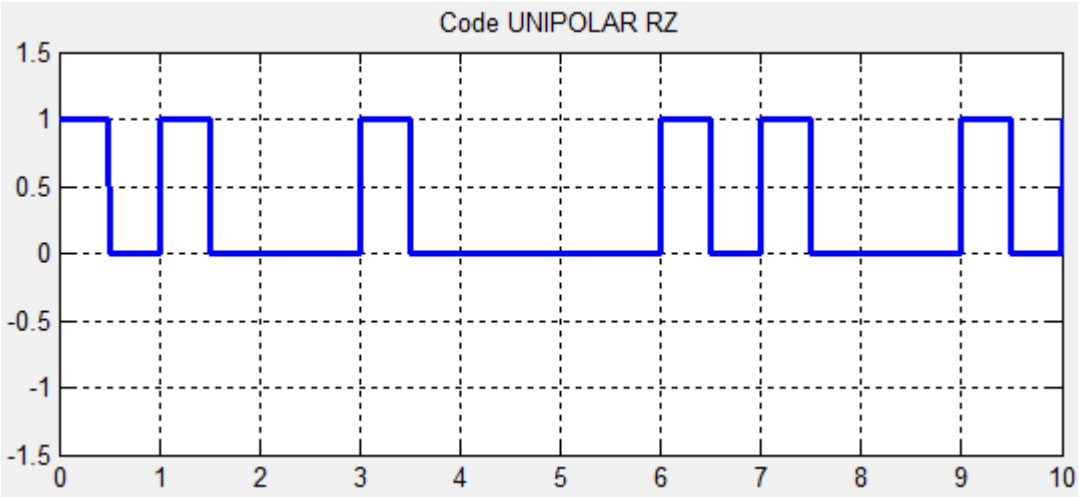

Figura 4. 8: Generación de bits del código Unipolar RZ.
Elaborado por el Autor.

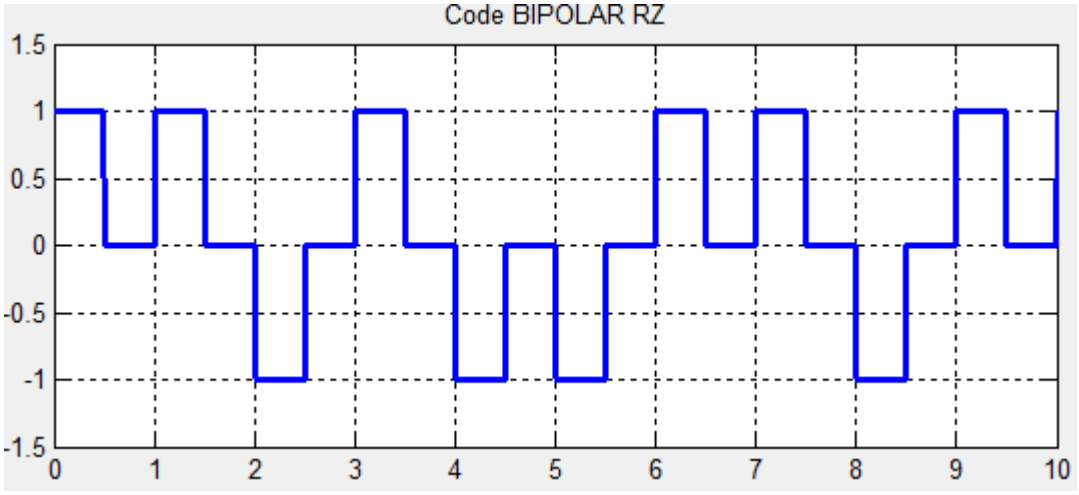

Figura 4. 9: Generación de bits del código Bipolar RZ.
Elaborado por el Autor.

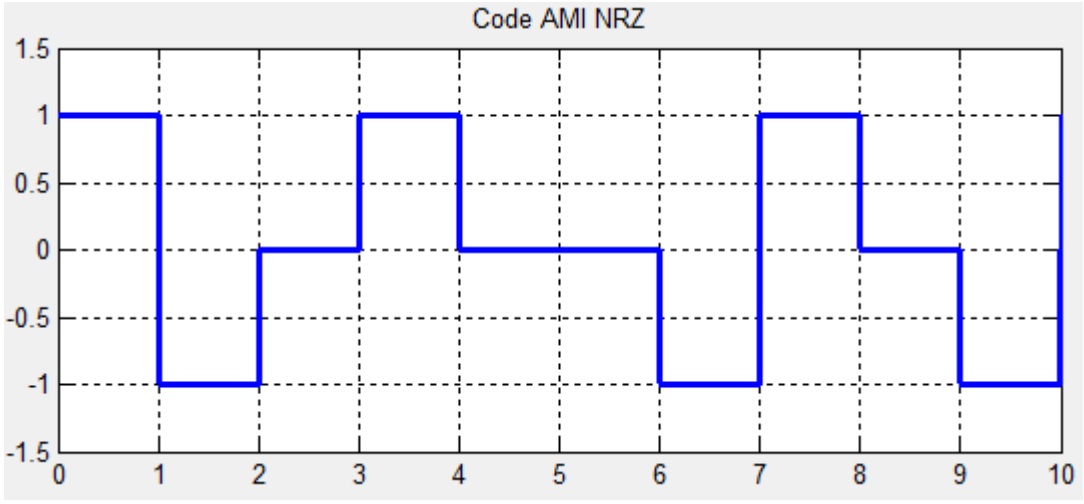

Figura 4. 10: Generación de bits del código AMI NRZ.
Elaborado por el Autor.

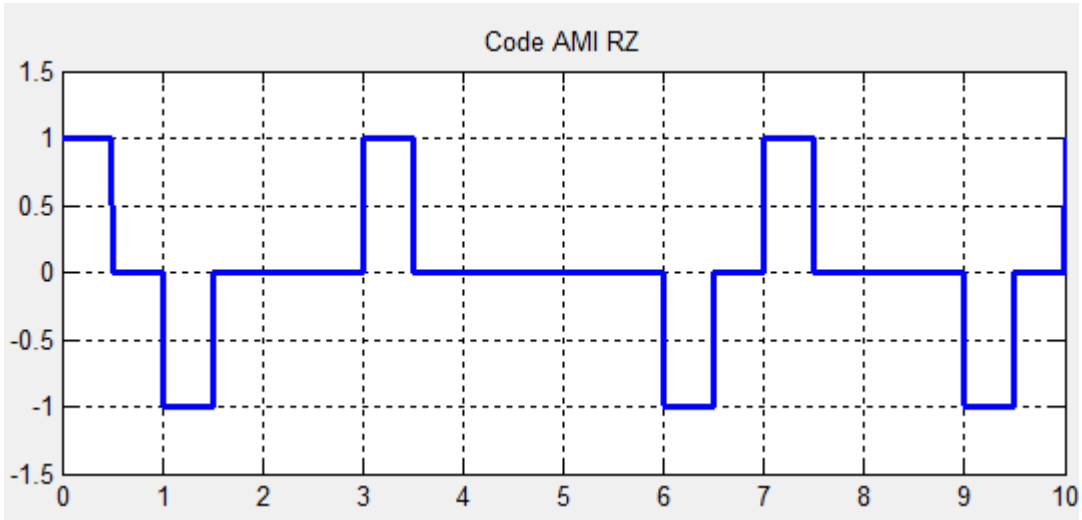

Figura 4. 11: Generación de bits del código AMI RZ.
Elaborado por el Autor.

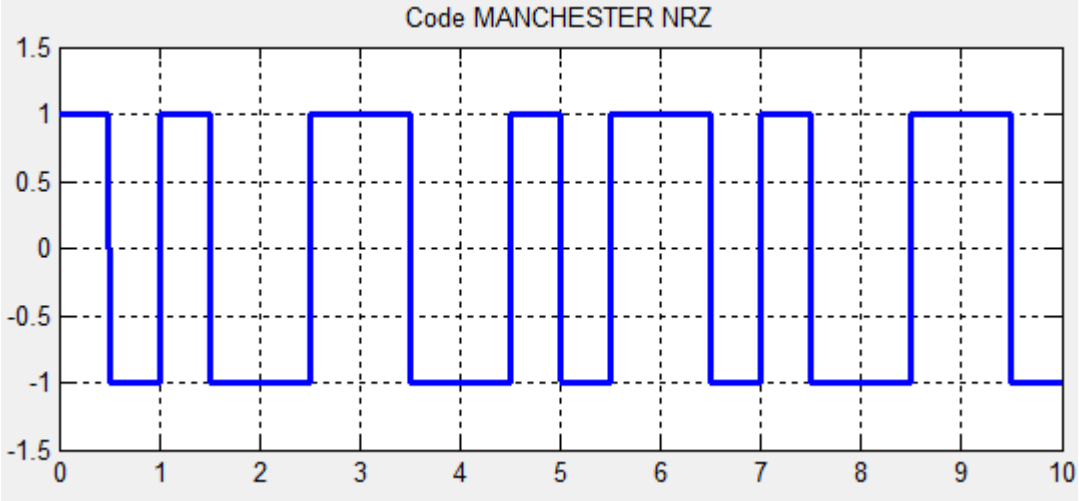

Figura 4. 12: Generación de bits del código Manchester NRZ.
Elaborado por el Autor.

Como sucedió en la figura 4.5, aquí se obtienen las densidades espectrales de potencia tal como se ilustran en las figuras 4.13, 4.14, 4.15 y 4.16.

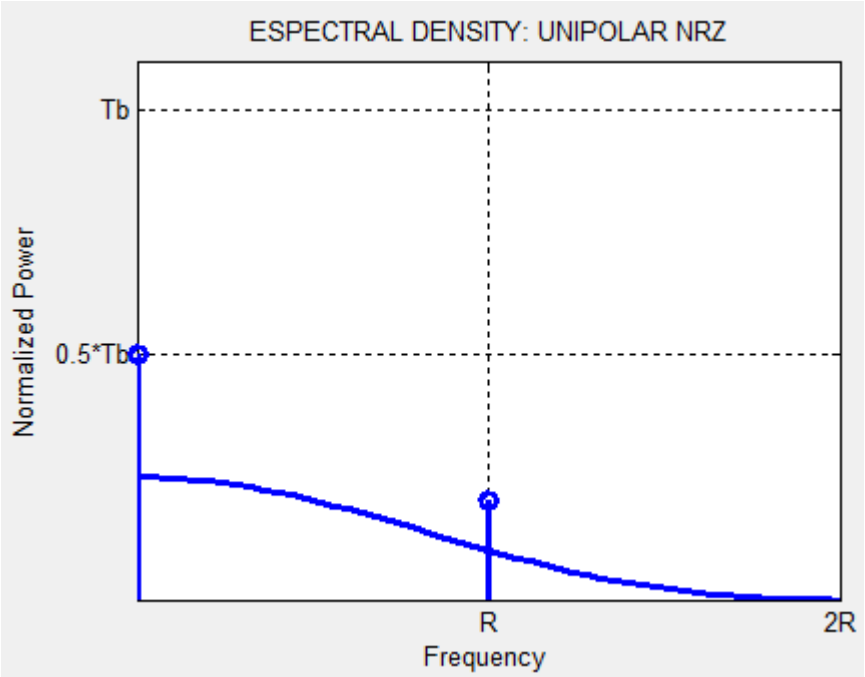

Figura 4. 13: Densidad espectral de potencia para Unipolar RZ.
Elaborado por el Autor.

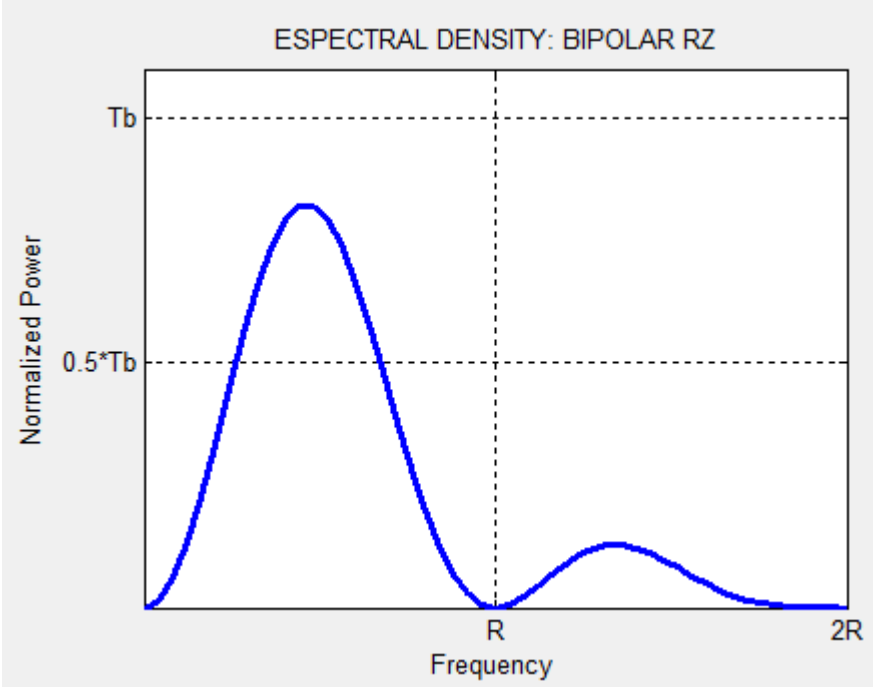

Figura 4. 14: Densidad espectral de potencia para Bipolar RZ.
Elaborado por el Autor.

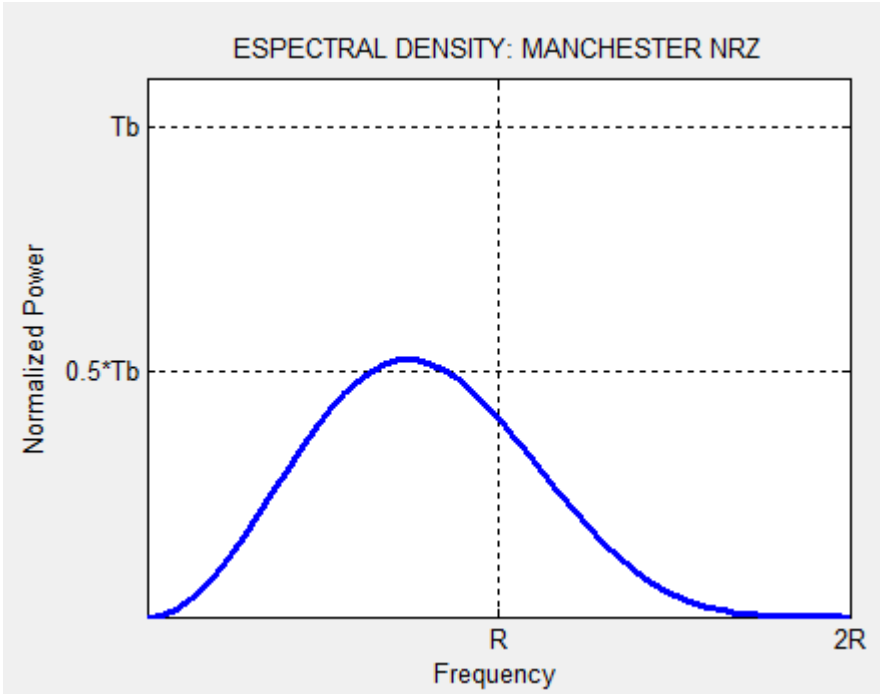

Figura 4. 15: Densidad espectral de potencia para AMI y Manchester NRZ.
Elaborado por el Autor.

CAPÍTULO 5: CONCLUSIONES Y RECOMENDACIONES.

5.1. Conclusiones.

- Una presentación detallada de codificación de línea, particularmente aplicable a la telefonía, se ha incluido en el presente trabajo de titulación. En la cual en el estado del arte se examinaron las características más deseables de códigos de línea. En la cual se presentaron cinco códigos de línea comunes y ocho códigos de línea alternos. Cada línea de código se ilustra con una forma de onda de ejemplo. En la mayoría de casos se dieron y se representan expresiones para el PSD y la probabilidad de error.

- Se examina brevemente la plataforma GUI-MatLab que es la plataforma de programación gráfica, muy utilizada para diferentes aplicaciones en el campo de la Ingeniería en Telecomunicaciones.

- La parte más importante del presente trabajo de titulación fue diseñar una interfaz de programación gráfica GUI, la misma que fue creada para verificar que las técnicas de codificación y la densidad espectral de potencian escogidas sean lo más parecidos a lo descrito en la parte del estado del arte.

5.2. Recomendaciones.

- De acuerdo a diferentes tesis y trabajos de titulación realizados en la Facultad de Educación Técnica para el Desarrollo (FETD)específicamente por los egresados de Ingeniería en Telecomunicaciones, sería factible que la FETD adquiera la plataforma MatLab con licencia profesional, y que sirva como ayuda de aprendizaje en la mayoría de asignaturas disponibles en la malla curricular.

- A los Docentes de la FETD deben realizar cursos de capacitación o certificarse en el manejo de MatLab, para que no solamente quede como obligación de los estudiantes investigar el uso de MatLab, es decir, que los estudiantes requieren de un profesional capacitado para emplear dicha herramienta.

REFERENCIAS BIBLIOGRÁFICAS

Amrinder, K., Mandeep, S., & Balwinder, S. (2011). *VHDL Implementation of Universal Encoder for communication*. ISP Journal of Electronics Engineering,Vol.1,Issue 2,ISSN 2250-0537(online),pp. 37–41 .

Chitode J., S. (2009). *Digital Communication.* 1era Edición. Technical Publications Pune, India.

Coimbra G, E. (2010). *Curso de Sistemas Electrónicos de Comunicaciones*. Apuntes de clases de la Carrera de Ingeniería Electrónica y de Redes y Telecomunicaciones. Universidad Privada de Santa Cruz, Bolivia.

Kulkarni, V., Arya, P. N. & Gaikar, P. V. (2011).*A Multilevel NRZ Line Coding Technique.* International Conference on Technology Systems and Management (ICTSM).

López R., D. A., Reyes A., M., Alejandro A., E., & Tirado M., J. A. (2009). *Educational prototype for line coding*. Proceeding of the6th WSEAS International Conference on Engineering Education,ISSN:1790-2769.

Mandziy,B., Bench,A., &Lipinsky, J. (2009). *Spectral Characteristics of Cyrillic Letters Encoded by Line Codes for Digital Transmission.* XIII International Conference System Modelling and Control. Zakopane, Poland.

Sneha L., V. (2011). *Performance Evaluation of Different Line Codes.* Indian Journal of Computers Science and Engineering (IJCSE). Vol. 2, No. 4. ISSN: 0976-5166.