ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

SIMULACIÓN DE LOS MÉTODOS LS Y MMSE PARA ESTIMACIÓN DE CANAL EN SISTEMAS OFDM

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

ESPINOSA NARANJO ELVIS GEOVANNY

elvis.espinosa@epn.edu.ec

DIRECTOR: DR. DIEGO JAVIER REINOSO CHASIGUANO

diego.reinoso@epn.edu.ec

Quito, mayo 2019

AVAL

Certifico que el presente trabajo fue desarrollado por Elvis Geovanny Espinosa Naranjo, bajo mi supervisión.

Dr. Diego Javier Reinoso Chasiguano
DIRECTOR DEL PROYECTO

DECLARACIÓN DE AUTORÍA

Yo, Elvis Geovanny Espinosa Naranjo, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____

Elvis Geovanny Espinosa Naranjo

DEDICATORIA

A mi madre Carlota, por ser mi apoyo y guiarme a ser la persona que soy.

A mis hermanos Alexander y Jefferson.

A la memoria de Mamá Martha, siempre te extrañaremos.

A mí, por demostrar que las metas se pueden alcanzar si en realidad se desea.

AGRADECIMIENTO

A mis padres Vicente y Carlota por su apoyo para realizar mis estudios universitarios.

A mis hermanos, Alexander y Jefferson por ser mi motor para lograrlo.

A mi abuelita Violeta y mi prima Mayra por su cariño y por estar siempre pendientes de mí.

A todos mis compañeros y amigos de la universidad por todas las experiencias vividas juntos, porque siempre estaban ahí cuando necesitaba y porque me han ayudado a crecer para ser la persona que soy ahora. Un agradecimiento especial a Dianita, por ser mi mejor amiga, por ser mi apoyo y por la ayuda brindada tanto en la parte académica como personal. A Soñita y Andrés por todos estos años de amistad, por todas las experiencias divertidas juntos. A mis amigos Jissela, David y Fausto por los momentos compartidos tanto en las aulas como fuera de ellas.

A la Escuela Politécnica Nacional ya que me ha formado para ser un excelente profesional.

A todos mis profesores por sus enseñanzas, así como por sus consejos y su amistad. Un agradecimiento especial al Dr. Diego Reinoso por el apoyo recibido en la realización de este proyecto técnico.

ÍNDICE DE CONTENIDO

RESUMEN

En el presente estudio técnico se presenta la simulación de un sistema de comunicaciones OFDM, en el cual se implementa los algoritmos LS *(Least Square)* y MMSE *(Minimum Mean-Square Error)* para estimación de canal. Los resultados de la simulación se presentan mediante curvas de la BER *(Bit Error Rate)* vs. Eb/No *(energy per-bit to noise power spectral density ratio)* y del MSE *(Mean Square Error)* vs. Eb/No entre el canal real y el canal estimado.

El capítulo 1 presenta la sustentación teórica del proyecto, en él se estudian los fenómenos presentes en un sistema de comunicaciones inalámbrico, así como las características de los sistemas OFDM y de los métodos de estimación de canal LS y MMSE.

El capítulo 2 describe la implementación del sistema de comunicaciones OFDM en Matlab, el cual está compuesto por un transmisor, un receptor y un canal selectivo en frecuencia con una distribución tipo Rayleigh. Se describen brevemente las funciones implementadas en Matlab, así como una breve explicación de la interfaz gráfica desarrollada.

El capítulo 3 muestra los resultados de las simulaciones, es decir las estimaciones de canal realizadas, las curvas de la BER vs. Eb/No y del MSE vs. Eb/No entre el canal real y el canal estimado.

Finalmente, el capítulo 4 presenta las conclusiones y recomendaciones del trabajo realizado.

PALABRAS CLAVE: OFDM, estimación de canal, MMSE, LS, canal Rayleigh

ABSTRACT

In this technical study, the simulation of an OFDM communication system is presented, in which LS (Least Square) and MMSE (Minimum Mean-Square Error) algorithms are implemented for channel estimation. The results of the simulation are presented by curves of the BER (Bit Error Rate) vs. Eb/No (energy per-bit to noise power spectral density ratio) and the MSE (Mean Square Error) vs. Eb/No between the real channel and the estimated channel.

Chapter 1 presents the theoretical support of the project, which studies the phenomena present in a wireless communications system, as well as characteristics of the OFDM systems and the LS and MMSE channel estimation methods.

Chapter 2 describes the implementation of the OFDM communication system in Matlab, which is composed of a transmitter, a receiver and a frequency selective channel with a Rayleigh distribution type. The Matlab implemented functions are briefly described, as well as a brief explanation of the developed interface.

Chapter 3 shows the results of the simulations, that is, the channel estimates made, the BER vs. Eb/No curve and the MSE vs. Eb/No curve between the real channel and the estimated channel.

Finally, chapter 4 presents the conclusions and recommendations of this technical study.

KEYWORDS: OFDM, channel estimation, MMSE, LS, Rayleigh channel

1. INTRODUCCIÓN

En los últimos años se ha producido un crecimiento en la demanda de servicios de comunicaciones móviles y de accesos inalámbricos a Internet. Cada vez, los usuarios solicitan mayores velocidades de transmisión en comunicaciones independientes del lugar, por lo cual es necesario el uso de tecnologías que permitan un acceso rápido y eficiente a la red. Los sistemas multiportadora son una forma eficiente de satisfacer los requerimientos de los usuarios, por lo cual son actualmente utilizados en diferentes estándares. OFDM *(Orthogonal Frequency Division Multiplexing)* es uno de esos sistemas de transmisión multiportadora.

Los sistemas multiportadora y especialmente OFDM tienen como ventaja la robustez en entornos multitrayecto, un entorno muy común en comunicaciones inalámbricas. Esta característica es la que permite a OFDM realizar transmisiones de datos a altas velocidades a pesar del uso de un canal inalámbrico, el cual es dinámico e impredecible. Debido a las características del canal antes mencionadas es necesario el proceso de estimación de canal, ya que conociendo el comportamiento del canal se puede compensar los cambios ocurridos en la señal.

Es común que en simulaciones de sistemas OFDM no se realice el proceso de estimación de canal, debido a que parten del supuesto de un canal inalámbrico conocido. Sin embargo, esto en la práctica no es cierto. De hecho, el no conocer el comportamiento del canal impacta en la eficiencia del sistema de comunicaciones. Por esta razón, en el presente proyecto técnico se propone implementar la simulación de un sistema OFDM en el cual se realizará estimación de canal utilizando los algoritmos LS *(Least Square)* y MMSE *(Minimum Mean-Square Error)* para mejorar el desempeño del sistema.

1.1 Objetivos

El objetivo general de este proyecto técnico es:

- Simular los métodos LS y MMSE para estimación de canal en sistemas OFDM.

Los objetivos específicos de este proyecto técnico son:

- Describir características fundamentales de la arquitectura, parámetros, y componentes de un sistema de comunicación OFDM y de los métodos de estimación de canal LS y MMSE.
- Implementar en MATLAB un sistema OFDM y un canal de comunicaciones inalámbrico.

- Implementar los algoritmos de estimación de canal LS y MMSE usando los esquemas de distribuciones de pilotos *block type* y *comb type*.
- Comparar el desempeño del sistema OFDM usando los algoritmos de estimación de canal LS y MMSE.

1.2 Alcance

El presente proyecto técnico presentará la simulación de un sistema de comunicaciones OFDM, utilizando una interfaz gráfica diseñada en *App Designer* de Matlab. El sistema de comunicaciones mencionado se utilizará para probar diferentes métodos de estimación de canal y constará de un transmisor, un receptor y un canal inalámbrico, los cuales serán implementados a través de una simulación. El diagrama de bloques del sistema OFDM a implementarse puede observarse en la Figura 1.1.

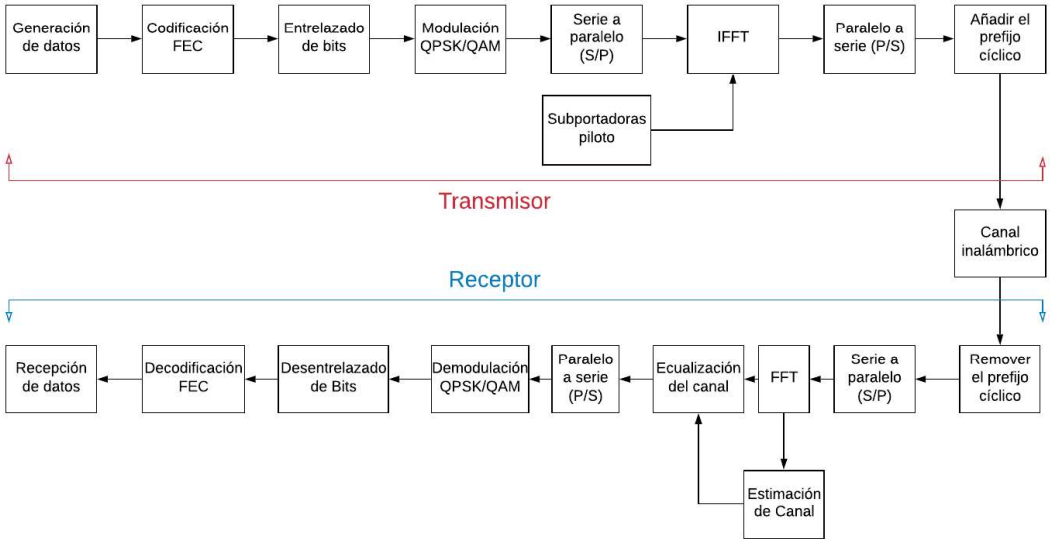

Figura 1.1 Diagrama de bloques del sistema OFDM a implementarse.

Con respecto al transmisor, los datos a transmitirse serán bits que se generarán de forma aleatoria, a estos bits se les realizará una codificación tipo FEC (*Forward Error Correction*) utilizando un codificador convolucional para a continuación efectuar un entrelazado de bits. Posteriormente se realizará la modulación utilizando esquemas de modulación como: QPSK (*Quadrature Phase-Shift Keying*), 16QAM *(Quadrature Amplitude Modulation)* y 64QAM. Luego, se efectuará una conversión de serie a paralelo y se añadirán las subportadoras piloto para después realizar la IFFT (*Inverse Fast Fourier Transform*) y posteriormente añadir el prefijo cíclico al símbolo OFDM que será transmitido [1].

En el receptor se realizará el proceso inverso, es decir primero se removerá el prefijo cíclico, posteriormente la señal que estaba en serie se transformará a paralelo para poder ingresar al bloque que realizará la FFT (*Fast Fourier Transform*) de la señal. Este bloque alimentará tanto a la función de estimación de canal como a la de ecualización del canal. Con la señal ecualizada se realizará la demodulación QAM o QPSK dependiendo del esquema de modulación usado en transmisión. Posteriormente se realizará el desentrelazado de bits y la decodificación FEC y finalmente se tendrá los datos recibidos.

Para realizar la estimación de canal en recepción, es necesario que el transmisor previamente haya insertado las señales piloto. Para el sistema OFDM a implementarse se utilizarán 64 subportadoras de las cuales 12 serán usadas como portadoras virtuales, por lo cual se tendrán 52 subportadoras entre pilotos y subportadoras de datos. En el proceso de inserción de pilotos se utilizará los esquemas: *block type* o *comb type*. En caso de usarse el esquema de inserción de pilotos *block type* se utilizarán todas las subportadoras del primer símbolo del bloque OFDM para enviar pilotos. En cambio, cuando se usa un esquema de transmisión *comb type* se colocarán los pilotos con una separación uniforme. Se probarán 2 esquemas: 1 piloto cada 4 subportadoras y 1 piloto cada 8 subportadoras.

En recepción, para estimar el canal en las posiciones piloto se utilizará el algoritmo LS o MMSE. Una vez realizada la estimación de canal y solo cuando sea necesario se interpolarán las estimaciones de canal en las posiciones piloto para encontrar la respuesta del canal en toda la rejilla tiempo-frecuencia. Al usarse el esquema de inserción de pilotos *block type* se considerará que la respuesta del canal no cambia, por lo cual la estimación realizada para los pilotos del primer símbolo OFDM se usará en todos los demás símbolos del bloque OFDM. Solo en caso de que el transmisor haya usado inserción de pilotos con un esquema *comb type* y el algoritmo LS será necesario interpolar para encontrar la respuesta en frecuencia del canal para todas las subportadoras.

Debido a que este proyecto técnico no está enfocado en sincronización, se considerará que se tiene sincronización perfecta, por lo cual no se realizará el proceso de sincronización en recepción. Con respeto al canal de comunicaciones inalámbrico a considerarse en las simulaciones será un canal configurable en el que se presentará desvanecimiento con una distribución de Rayleigh. En cuanto a las pruebas que se realizarán: se estimará la BER (*Bit Error Rate*) de la comunicación y el MSE (*Mean Square Error*) entre la respuesta real del canal y la estimación realizada, para diferentes configuraciones, usando los diferentes algoritmos y con diferentes valores de Eb/No (*energy per bit to noise power spectral density ratio*).

1.3 Marco Teórico

1.3.1 Comunicaciones Inalámbricas

La búsqueda de una mejor forma de vivir ha sido fundamental para el avance de la civilización humana. En esta búsqueda de progreso, se ha vuelto esencial que los servicios de comunicación estén disponibles en cualquier momento, sin que para ello se necesite que las personas estén conectadas a dispositivos fijos. Hoy en día, gracias al notable progreso en tecnologías inalámbricas, servicios de comunicaciones inalámbricas asequibles se han convertido en una realidad. Los teléfonos móviles conectan a las personas cuando y donde quieran. La transmisión de audio y video digital ofrece a los consumidores programas de alta resolución, con buena calidad de video e incluso programas interactivos. Además, los teléfonos inteligentes con capacidades multimedia y de acceso a Internet se han vuelto algo cotidiano. Para el desarrollo de todos estos avances tecnológicos ha sido necesario el uso de redes inalámbricas con diferentes extensiones de cobertura. Estas redes permiten el acceso a Internet, ya sea en interiores, al aire libre o en áreas rurales o metropolitanas [2].

El Instituto de Ingenieros Eléctricos y Electrónicos (*Institute of Electrical and Electronics Engineers*, IEEE) ha definido varios estándares de redes inalámbricas, desde áreas pequeñas a áreas grandes, como se muestra en la Figura 1.2. La más pequeña es la red de área personal inalámbrica (*Personal Area Network*, PAN), que cubre solo varios metros alrededor de un usuario. Operando en un entorno más grande que la red PAN, se tiene la red de área local (*Local Area Network,* LAN) inalámbrica IEEE 802.11, la cual es el estándar inalámbrico más exitoso y prevalente. En redes LAN inalámbricas se proporciona comunicaciones de corta distancia de hasta 100 metros. La red de área metropolitana (*Metropolitan Area Network*, MAN) extiende su cobertura hasta varios kilómetros. La red de área amplia (*Wide Area Network***,** WAN) es el estándar con mayor cobertura y admite comunicaciones de hasta decenas de kilómetros, incluidos terrenos montañosos y zonas rurales. Con todas estas redes, el acceso a Internet puede estar disponible cuando y donde el usuario desee [2].

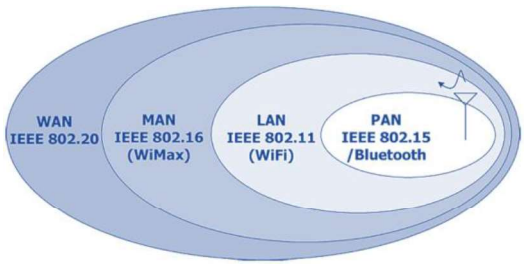

Figura 1.2 Estándares de redes inalámbricas IEEE [2].

1.3.2 El canal inalámbrico

El rendimiento de un sistema de comunicación inalámbrico se encuentra estrechamente relacionado con las características del canal inalámbrico utilizado. A diferencia de un canal alámbrico que es estático y predecible, el canal inalámbrico es dinámico e impredecible, lo cual dificulta el análisis de un sistema de comunicación. En los últimos años, la optimización de los sistemas de comunicaciones inalámbricos se ha vuelto crítica debido al crecimiento rápido de servicios de comunicaciones móviles y servicios de acceso a Internet móvil de banda ancha. La importancia del canal inalámbrico radica en comprender que su funcionamiento es la base para el desarrollo de futuras tecnologías de transmisión [1].

En una comunicación inalámbrica se utiliza la propagación por ondas de radio para enviar la información desde el transmisor hacia el receptor. Las ondas de radio usadas para la transmisión se ven afectadas principalmente por tres fenómenos físicos: reflexión (*reflection*), difracción (*diffraction*) y dispersión (*scattering*) como se puede observar en la Figura 1.3 [3], [4].

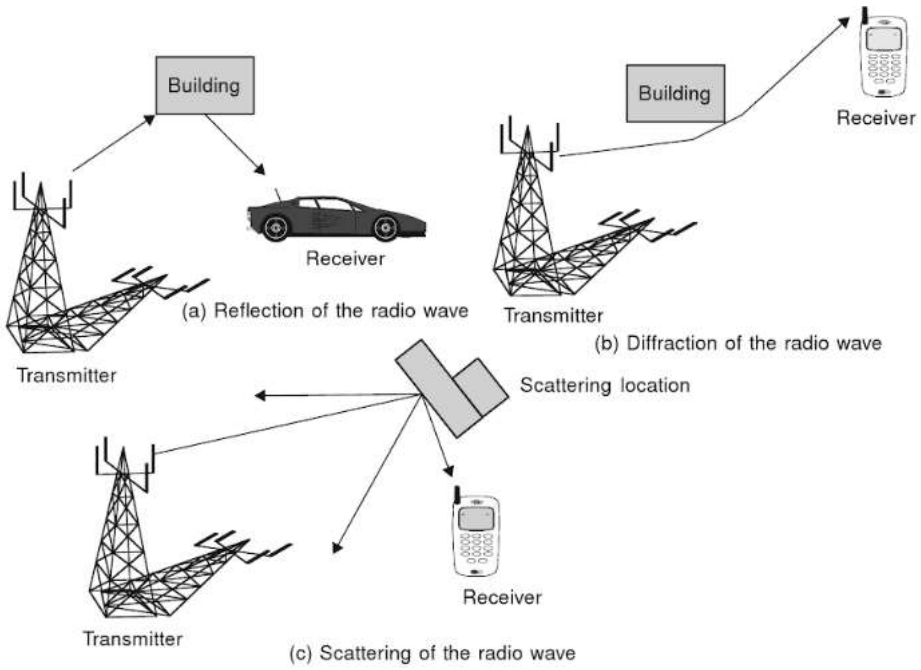

Figura 1.3 Fenómenos de propagación [5].

La reflexión es el fenómeno físico que se produce cuando una onda electromagnética incide sobre un objeto con dimensiones muy grandes en comparación con su longitud de onda, como por ejemplo la superficie de la tierra o un edificio. La amplitud de la onda disminuye debido a la absorción de energía por parte de la superficie reflectora, el ángulo con el que se refleja la onda es igual al ángulo con el que incide sobre la superficie [1], [5].

La difracción es un fenómeno que ocurre cuando el enlace entre el transmisor y el receptor se encuentra obstruido por una superficie que presenta irregularidades agudas o pequeñas aberturas. El efecto de la difracción se aprecia como un conjunto de olas alrededor de los pequeños obstáculos y se extiende fuera de las olas a través de las pequeñas aberturas. Las ondas secundarias generadas por la difracción son útiles para establecer un enlace entre transmisor y receptor, incluso cuando no hay línea de vista. Superficies que pueden causar difracción son paredes edificios o montañas [1], [5].

La dispersión es un fenómeno físico que ocurre cuando una onda choca con uno o más obstáculos locales, con pequeñas dimensiones en comparación con la longitud de onda de la señal transmitida. La onda se divide en señales con menor amplitud que se propagarán en todas las direcciones. Los obstáculos que pueden inducir dispersión son: vegetación, señales de tránsito, postes de luz entre otros [1], [5].

1.3.2.1 Desvanecimiento

Una característica única del canal inalámbrico es un fenómeno llamado "desvanecimiento", que es la variación de la amplitud de la señal en el dominio del tiempo y de la frecuencia. En contraste con el ruido, el desvanecimiento es una fuente de degradación de la señal que se caracteriza como una perturbación de la señal no aditiva en el canal inalámbrico.

El desvanecimiento puede deberse a la propagación por múltiples trayectos denominado desvanecimiento por multitrayectoria (*multipath fading*) o al efecto de sombra (*shadowing*) debido a obstáculos que afectan la propagación de las ondas, denominado desvanecimiento por sombra [1].El desvanecimiento se puede clasificar en dos tipos: desvanecimiento a gran escala y desvanecimiento a pequeña escala.

El desvanecimiento a gran escala se produce cuando el móvil se mueve grandes distancias, por ejemplo, una distancia en el orden del tamaño de una celda [4]. Este tipo de desvanecimiento se debe a la pérdida de la potencia de la señal en función de la distancia (llamado *path loss*) y por la sombra de objetos grandes (*shadowing*) como edificios, terrenos intermedios y vegetación. El efecto de sombra es un proceso de desvanecimiento lento caracterizado por la variación de la pérdida media debido a la distancia entre el transmisor y el receptor en ubicaciones fijas. En otras palabras, el desvanecimiento a gran escala se caracteriza por la pérdida de potencia en función de la distancia y el efecto de sombra promedio en la trayectoria [1].

Por otro lado, el desvanecimiento a pequeña escala es la rápida variación de la potencia de la señal cuando la estación móvil se desplaza distancias cortas. Este desvanecimiento se debe a interferencias constructivas y destructivas debido a múltiples copias de la señal

(multitrayectoria). La relación entre el desvanecimiento a gran escala y el desvanecimiento a pequeña escala se ilustra en la Figura 1.4.

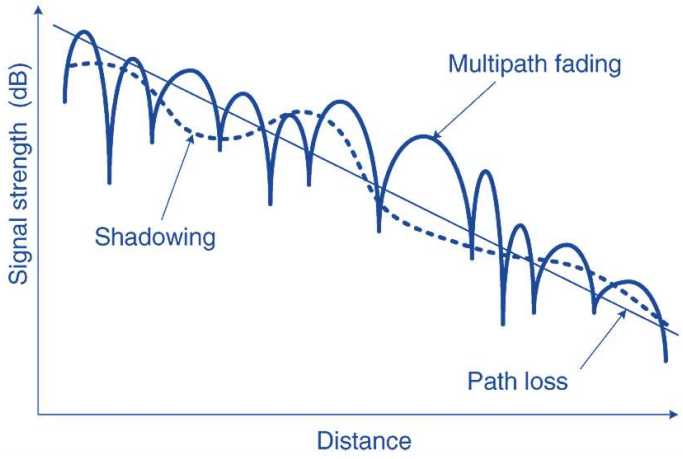

Figura 1.4 Desvanecimiento a gran escala vs. desvanecimiento a pequeña escala [1].

Dependiendo de la extensión del multitrayecto se clasifica la selectividad en frecuencia en respuesta plana en frecuencia o respuesta selectiva en frecuencia. Mientras tanto, dependiendo de la variación en tiempo del canal debido a la velocidad móvil (caracterizada por el desplazamiento Doppler) se clasifican en desvanecimiento rápido o desvanecimiento lento. La Figura 1.5 clasifica los tipos de desvanecimiento [1].

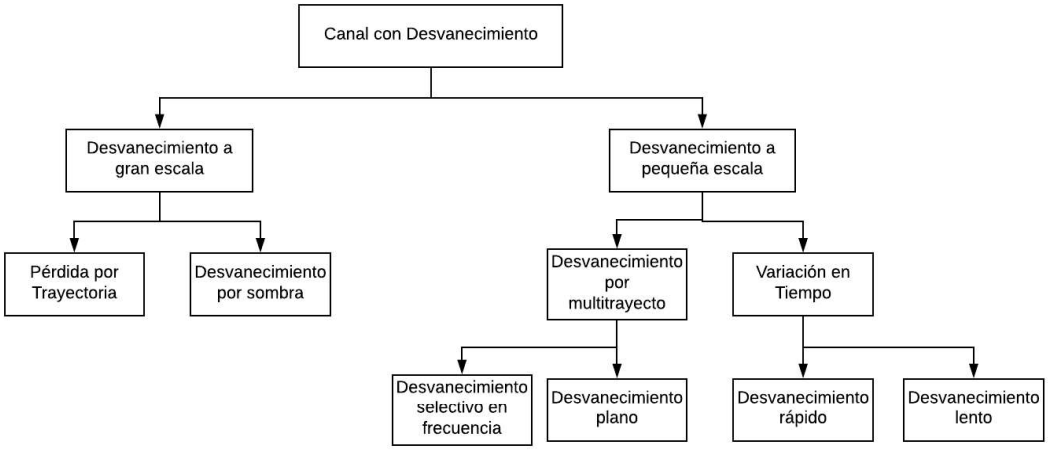

Figura 1.5 Clasificación de los canales con desvanecimiento [1].

La intensidad de la señal puede ser diferente a una misma distancia del transmisor, debido al efecto de sombra causado por obstáculos en el enlace. Además, que algunos de esos obstáculos pueden provocar dispersión causando un desvanecimiento a pequeña escala, el cual produce una variación a corto plazo de la señal que ya experimentó previamente el efecto de sombra.

La pérdida media por trayectoria es un factor determinista que se puede predecir con la distancia entre transmisor y receptor. Por el contrario, el efecto de sombra y el desvanecimiento a pequeña escala son fenómenos aleatorios, lo que significa que sus efectos solo pueden predecirse usando una distribución probabilística. Por ejemplo, el efecto de sombra normalmente se modela mediante una distribución log-normal [1].

Desvanecimiento a gran escala

El modelo de propagación en espacio libre se utiliza para predecir la intensidad de la señal recibida en entornos con línea de vista (*Line-of-Sight*, LOS) donde no hay obstáculos entre transmisor y receptor. La potencia recibida en función de la distancia *d*, $P_r(d)$, se expresa mediante la Ecuación 1.1 [6].

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L}$$

Ecuación 1.1 Ecuación de Friis.

Siendo P_t la potencia de transmisión (vatios), λ es la longitud de onda en metros, *d* la distancia en metros entre el transmisor y el receptor, G_t y G_r las ganancias de las antenas de transmisión y de recepción respectivamente y *L* es el factor de pérdidas del sistema, el cual representa la atenuación o pérdida general en el hardware del sistema, incluida la línea de transmisión, el filtro y las antenas. En general, *L>1*, pero *L=1* asumiendo que no hay pérdida en el hardware [1].

El modelo de pérdidas en espacio libre $PL_F(d)$ se puede obtener a partir de la Ecuación 1.1, en donde se asume que *L=1* G_t=1y G_r=1. Este modelo de pérdidas se define como [1]:

$$\boldsymbol{PL_F(d)[dB] = 20log\left(\frac{4\pi d}{\lambda}\right)}$$

Ecuación 1.2 Modelo de pérdidas en el espacio libre.

Se puede construir una forma más generalizada del modelo de pérdidas por trayectoria modificando la ecuación anterior con el exponente de pérdida por trayectoria *n* que varía de 2 a 6 dependiendo del entorno como se observa en la Tabla 1.1. Esto se conoce como el modelo de pérdida por trayectoria *log-distance*, en el que la pérdida de trayectoria a la distancia *d* se define como:

$$\boldsymbol{PL_{LD}(d)[dB] = PL_F(do) + 10nlog\left(\frac{d}{do}\right)}$$

Ecuación 1.3 Modelo de pérdida por trayectoria *log-distance* [1].

Tabla 1.1 Exponente de pérdidas en diferentes ambientes [7].

Ambientes	Exponente de pérdidas (*n*)
Espacio libre	2
Radio celular de área urbana	2.7-3.5
Radio celular de área sombreada	3-5
Línea de vista en edificio	1.6-1.8
Obstruido en edificio	4-6
Obstruido en fábricas	2-3

Donde *do* es una distancia de referencia que se determina para diferentes entornos de propagación. Por ejemplo, *do* se establece como 1 km para un sistema celular con una gran cobertura (por ejemplo, un sistema celular con un radio de celda superior a 10 km). Sin embargo, podría ser de 100 m para un sistema macrocelular con un radio de 1 km o podria ser de 1 m para un sistema microcelular con un radio extremadamente pequeño [8].

A pesar de que la distancia entre transmisor y receptor sea igual en dos enlaces, en cada enlace se puede tener diferentes pérdidas. Los modelos de pérdida por trayectoria mencionados anteriormente no consideran esta situación particular. Un modelo de pérdidas *log-normal* con sombra es útil cuando se trata de una situación más realista. Siendo X_σ una variable aleatoria gaussiana con media cero y desviación estándar σ, entonces, el modelo de pérdidas *log-normal* con sombra se define como [1]:

$$PL(d)[dB] = PL_F(do) + 10nlog\left(\frac{d}{do}\right) + X_\sigma$$

Ecuación 1.4 Modelo de pérdidas *log-normal* con sombra.

Este modelo ilustra claramente el efecto aleatorio de la sombra que se impone sobre la naturaleza determinista del modelo de pérdida por trayectoria *log-distance*.

Desvanecimiento a pequeña escala

El desvanecimiento a pequeña escala es la variación rápida del nivel de la señal recibida en un corto plazo, a medida que el terminal se mueve una distancia corta. Este desvanecimiento se debe al efecto de múltiples trayectos de la señal, que causan interferencia cuando llegan a la antena receptora con fases variables (es decir, interferencia constructiva con la misma fase e interferencia destructiva con una fase diferente). Cada uno de los múltiples trayectos de la señal puede sufrir cambios dependiendo de las velocidades de la estación móvil y de los objetos circundantes. En resumen, el desvanecimiento a pequeña escala se atribuye a la multitrayectoria, la velocidad del móvil, la velocidad de los objetos circundantes y al ancho de banda de la señal [1].

Las características de un canal con desvanecimiento por multitrayectoria a menudo se especifican mediante un *Power Delay Profile* (PDP). La Tabla 1.2 presenta como ejemplo el PDP específico para el modelo de canal peatonal UIT-R, en el que se tiene cuatro trayectos de la señal caracterizados por su retraso relativo y potencia promedio. El retraso relativo es un retardo en exceso con respecto a la primera ruta, se asume que esta ruta no presenta retraso y la potencia promedio de cada ruta se normaliza con respecto a la primera ruta (*tap*) [9].

Tabla 1.2 PDP para un canal peatonal (ITU-R) [9].

Trayecto	Retraso relativo(ns)	Potencia promedio (dB)
1	0	0.0
2	110	-9.7
3	190	-19.2
4	410	-22.8

El *mean excess delay* y el *RMS delay spread* son parámetros de canal útiles que proporcionan una referencia para comparar diferentes canales con desvanecimiento por multitrayectoria. Siendo τ_k el retardo del canal en el trayecto k-ésimo mientras que a_k y $P(\tau_k)$ son la amplitud y la potencia, respectivamente. Entonces, el *mean excess delay* $\bar{\tau}$ se define como [1]:

$$\bar{\tau} = \frac{\sum_k a_k^2 \tau_k}{\sum_k a_k^2} = \frac{\sum_k \tau_k P(\tau_k)}{\sum_k P(\tau_k)}$$

Ecuación 1.5 *Mean excess delay*.

Mientras que el *RMS delay spread* σ_T es [1]:

$$\sigma_\tau = \sqrt{\overline{\tau^2} - (\bar{\tau})^2}$$

Ecuación 1.6 RMS delay spread.

Donde $\overline{\tau^2}$ se obtiene como [1]**:**

$$\overline{\tau^2} = \frac{\sum_k a_k^2 {\tau_k}^2}{\sum_k a_k^2} = \frac{\sum_k {\tau_k}^2 P(\tau_k)}{\sum_k P(\tau_k)}$$

Ecuación 1.7 Segundo momento central del PDP.

En general, el ancho de banda de coherencia, denotado como B_c, es inversamente proporcional al *RMS delay spread*, es decir [1]:

$$B_C \approx \frac{1}{\sigma_\tau}$$

Ecuación 1.8 Ancho de Banda de Coherencia.

El tipo específico de desvanecimiento depende tanto del esquema de transmisión como de las características del canal. El esquema de transmisión se especifica mediante parámetros de la señal como su ancho de banda y su período de símbolo. Mientras tanto, el canal inalámbrico se caracteriza con los parámetros *RMS delay spread* y el desplazamiento Doppler, los cuales causan dispersión en tiempo y dispersión en frecuencia, respectivamente. Dependiendo de la extensión de la dispersión en tiempo o de la dispersión en frecuencia, se induce un desvanecimiento selectivo en frecuencia o un desvanecimiento selectivo en tiempo respectivamente [1].

- **Canal con desvanecimiento en frecuencia**

Debido a la dispersión en tiempo, una señal puede sufrir desvanecimiento en el dominio de la frecuencia de manera selectiva o no selectiva, lo que se conoce como desvanecimiento selectivo en frecuencia o desvanecimiento plano, respectivamente. La selectividad en frecuencia depende del ancho de banda de la señal. La respuesta del canal varía con la frecuencia debido a la dispersión en tiempo causada por las múltiples trayectorias. Una señal está sujeta a un desvanecimiento no selectivo en frecuencia cuando el ancho de banda de la señal es suficientemente estrecho para que se transmita a través de la respuesta plana del canal. Por otro lado, una señal está sujeta a un desvanecimiento selectivo en frecuencia cuando el ancho de banda de la señal es suficientemente amplio para que la señal se filtre debido al ancho de banda finito del canal [1].

Un ancho de banda estrecho implica que el período de símbolo T_s es mayor que el *delay spread* τ del canal. Mientras que T_s sea mayor que τ el símbolo actual no afecta demasiado al símbolo siguiente, lo cual implica que la interferencia entre símbolos (*intersymbol interference*, ISI) no es significativa, aunque en el canal con desvanecimiento plano la amplitud varíe lentamente en el tiempo. Resumiendo, una señal está sujeta a un desvanecimiento plano en las siguientes condiciones [1]:

$$B_s \ll B_c \quad y \quad T_s \gg \sigma_\tau$$

Ecuación 1.9 Condiciones para un desvanecimiento plano en frecuencia.

Donde B_s y T_s son el ancho de banda y el período de símbolo de la señal respectivamente. Mientras que B_c y σ_τ son el ancho de banda de coherencia y el *RMS delay spread*, respectivamente [1].

Como se mencionó anteriormente, la señal sufre un desvanecimiento selectivo en frecuencia cuando el canal inalámbrico tiene una respuesta de fase lineal y amplitud constante dentro de un ancho de banda del canal (ancho de banda de coherencia B_c) menor que el ancho de banda de la señal B_s. Debido a la corta duración del símbolo (T_s) en comparación con el *RMS delay spread* σ_τ, las múltiples copias con retardo de la señal se superponen significativamente con el símbolo subsiguiente produciendo ISI. Resumiendo las observaciones anteriores, la señal está sujeta a un desvanecimiento selectivo en frecuencia en las siguientes condiciones [1]:

$$B_s > B_c \quad y \quad T_s < \sigma_\tau$$

Ecuación 1.10 Condiciones para un desvanecimiento selectivo en frecuencia.

- **Canal con desvanecimiento en tiempo**

Dependiendo de la extensión del desplazamiento Doppler, la señal recibida sufre un desvanecimiento rápido o lento. En un canal con desvanecimiento rápido, el tiempo de coherencia T_c es más pequeño que el período del símbolo T_s, por lo tanto, la respuesta impulsiva del canal varía rápidamente dentro del período del símbolo. La variación en el dominio del tiempo está estrechamente relacionada con el movimiento del transmisor o del receptor, lo que produce una expansión en el dominio de la frecuencia, conocido como desplazamiento Doppler (f_m). El ancho de banda del espectro Doppler, denotado como B_d, se define como $B_d=2f_m$. El tiempo de coherencia es inversamente proporcional al desplazamiento Doppler, es decir [1]:

$$T_c \approx \frac{1}{f_m}$$

Ecuación 1.11 Tiempo de coherencia.

$T_s > T_c$ implica que $B_s < B_d$ por tanto la señal está sujeta a un desvanecimiento rápido en las siguientes condiciones:

$$T_s > T_c \quad y \quad B_s < B_d$$

Ecuación 1.12 Condiciones para un desvanecimiento rápido [1].

En cambio, cuando la respuesta impulsiva del canal varía lentamente en comparación con la variación de la señal en banda base, se tiene un canal que no cambia durante la duración de uno o más símbolos, el cual se denomina canal estático. Lo que implica que el Desplazamiento Doppler es mucho menor que el ancho de banda de la señal en banda base. En conclusión, la señal está sujeta a un desvanecimiento lento cuando[1]:

$$T_s \ll T_c \quad y \quad B_s \gg B_d$$

Ecuación 1.13 Condiciones para un desvanecimiento lento.

Distribuciones de desvanecimiento

- **Distribución tipo Ricean**

Una distribución tipo Ricean (ver Figura 1.6) se usa para caracterizar el desvanecimiento a pequeña escala en un entorno con línea de vista (Line of Sight, LOS) en el cual se tiene una señal dominante y el resto son más débiles [5]. Su función de Distribución de probabilidad (Probability density function, PDF) es [5]:

$$p(r) = \frac{r}{\sigma^2} e^{-\left(\frac{r^2+A^2}{2\sigma^2}\right)} * I_o * \left(\frac{A * r}{\sigma^2}\right) \text{ para } A \geq 0\,, r \geq 0$$

Ecuación 1.14 PDF de la distribución tipo Ricean.

Donde: *A* es la amplitud pico de la señal dominante, I_o es la función de Bessel modificada de orden cero, $r^2/2$ es la potencia instantánea y σ es la desviación estándar de la potencia.

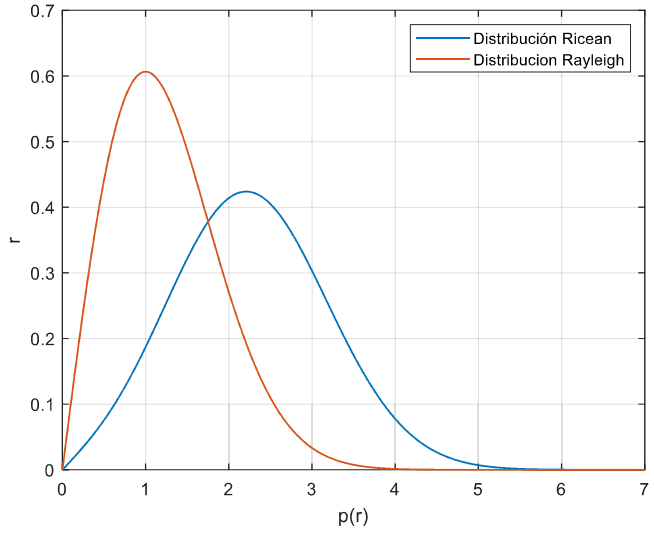

Figura 1.6 Distribuciones tipo Rayleigh y Ricean [5].

Una distribución Ricean se puede describir en términos del factor Ricean (*K*) dado por [5]:

$$K = 10 \log \frac{(A^2)}{2\sigma^2} \; [dB]$$

Ecuación 1.15 Factor Ricean.

Cuando la amplitud de la componente dominante tiende a 0, entonces la distribución tipo Ricean se convierte en una distribución tipo Rayleigh.

- **Distribución tipo Rayleigh**

La distribución tipo Rayleigh (ver Figura 1.6) se usa para describir estadísticamente las variaciones en tiempo de un canal que presenta un desvanecimiento plano con variación de una componente debido al multitrayecto. Este tipo de distribución se usa en entornos urbanos y suburbanos donde no existe línea de vista (Non Line of Sight, NLOS) [5]. La distribución tipo Rayleigh se caracteriza como:

$$p(r) = \frac{r}{\sigma^2} e^{-\left(\frac{r^2}{2\sigma^2}\right)} \qquad 0 \leq r \leq \infty$$

Ecuación 1.16 PDF de la distribución tipo Rayleigh [5].

Donde: σ es el valor RMS de la señal recibida, $r^2/2$ es la potencia instantánea y σ^2 es la potencia promedio recibida antes de la detección de la envolvente.

1.3.3 OFDM

OFDM es un esquema de transmisión que se ha adoptado por muchos sistemas de comunicaciones inalámbricas con altas tasas de velocidad, como son los estándares: IEEE 802.11n, IEEE 802.11ac e IEEE 802.16e, ya que es adecuado para canales con un desvanecimiento selectivo en frecuencia [10], [11]. Opuesto a una transmisión monoportadora, en donde la información se envía en un solo flujo de datos, OFDM divide los símbolos en diferentes flujos, los cuales se transmiten utilizando subportadoras que son paralelas y ortogonales entre sí. Con el uso de esta transmisión en paralelo, el periodo del símbolo se incrementa por lo cual la señal se vuelve más robusta frente a la ISI [12].

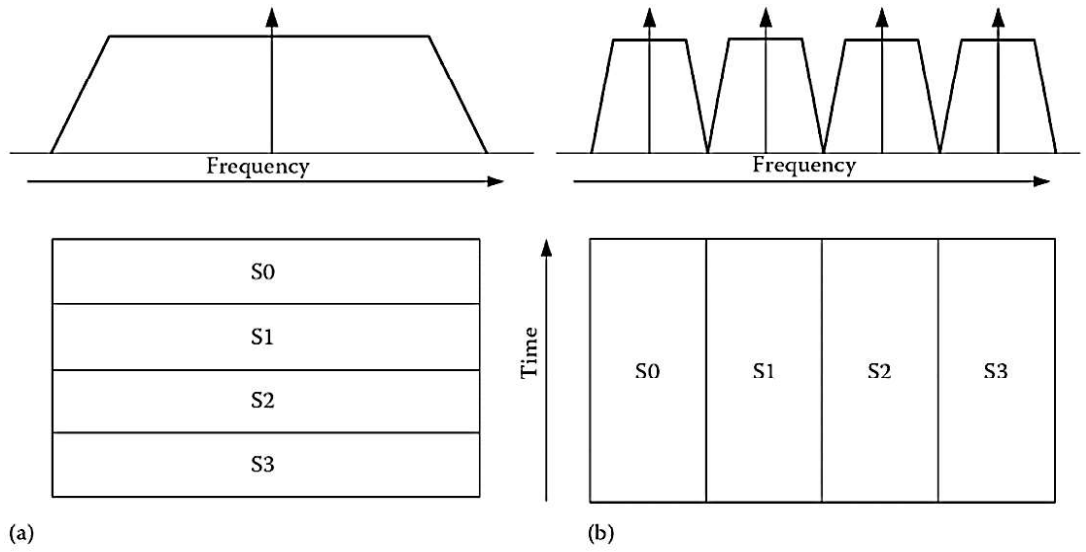

Figura 1.7 Espectro de monoportadora (a) vs. OFDM (b) [12].

La ortogonalidad entre subportadoras puede asegurarse con el uso de la transformada discreta de Fourier (*Discrete Fourier Transform,* DFT) y su inversa (*Inverse Discrete Fourier Transform,* IDFT). Tanto la DFT como la IDFT se pueden implementar de manera eficiente mediante el uso de la transformada rápida de Fourier (*Fast Fourier Transform,* FFT) y su inversa (*Inverse Fast Fourier Transform,* IFFT), respectivamente [1]. Por lo cual las señales OFDM se generan usando una IFFT de N puntos, donde la entrada de la IFFT es la representación en frecuencia de la señal OFDM y su salida es la representación en tiempo de la misma señal. Una IFFT de N puntos se define como un símbolo OFDM. El símbolo OFDM en el dominio del tiempo este compuesto de N subportadoras, a diferencia de una transmisión que usa una sola portadora como se observa en la Figura 1.7 [12]. A pesar de que las subportadoras paralelas se encuentran separadas en el dominio de la frecuencia, en el dominio del tiempo se encuentran sumadas, conformando una sola señal [12].

En una señal OFDM en la que el flujo de símbolos se divide en varios subflujos paralelos (cada uno asociado a una diferente subportadora) la ISI puede ocurrir dentro de los diferentes subflujos. Para mitigar los efectos de la ISI causados por el *delay spread* del canal se inserta un intervalo de guarda en el dominio del tiempo, llamado prefijo cíclico (*Cyclic prefix*, CP). El prefijo cíclico consiste en una simple copia de los últimos N_G símbolos en el dominio del tiempo, donde el número de muestras (N_G) debe ser igual o mayor al máximo retardo del canal. La inserción del prefijo cíclico se ilustra en la Figura 1.8 [1], [12].

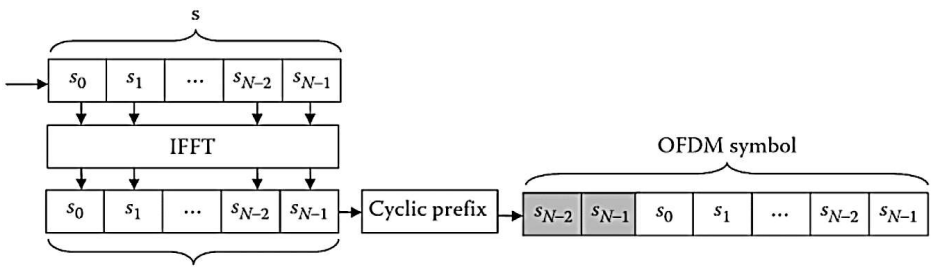

Figura 1.8 Inserción del Prefijo Cíclico [12].

Una señal OFDM puede incurrir en radiación fuera de banda, lo que provoca interferencia de canal adyacente (*adjacent channel interference*, ACI). Esto se produce ya que el espectro de potencia de la señal OFDM tiene lóbulos laterales bastante altos, como observa en la Figura 1.9, en la que el primer lóbulo lateral no es tan pequeño en comparación con el lóbulo principal. Para mitigar este efecto, en OFDM se usan las subportadoras externas como bandas de guarda, a estas subportadoras se las denominadas portadoras virtuales (*virtual carriers*, VC) o portadoras nulas. Las portadoras virtuales son subportadoras alrededor de la banda de frecuencia en las cuales no se transmite ningún dato. La reserva de la banda de guarda ayuda a reducir la emisión de

potencia fuera de misma, facilitando los requisitos de los filtros del transmisor. Sin embargo, la adopción de la banda de guarda desperdicia algo del ancho de banda asignado y disminuye la eficiencia espectral del sistema OFDM [1],[2].

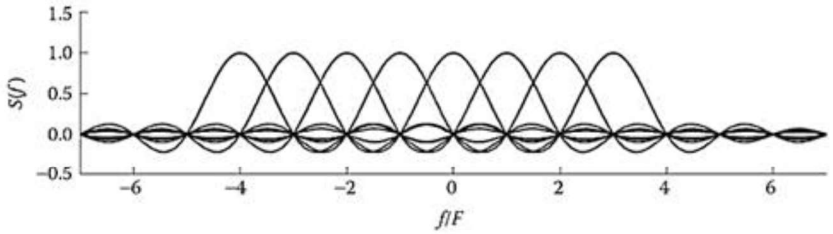

Figura 1.9 Subportadoras individuales

1.3.3.1 Esquema de un sistema OFDM

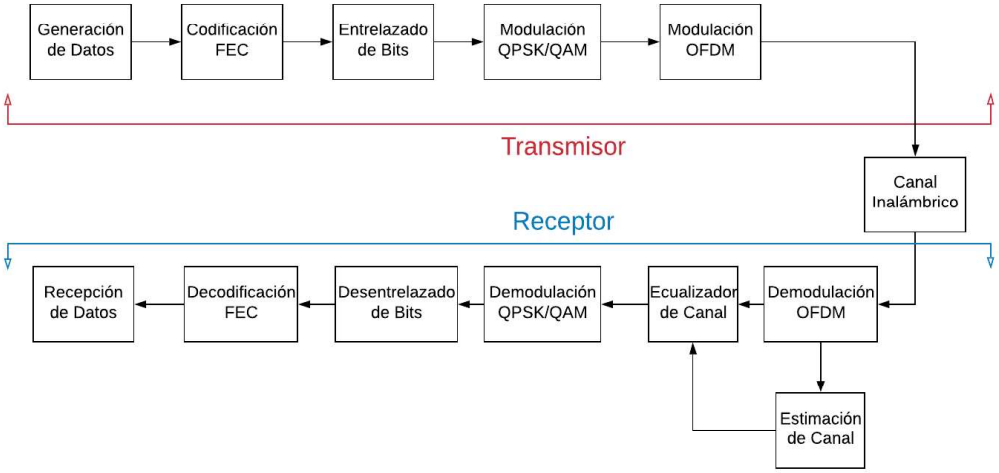

Figura 1.10 Esquema de un sistema OFDM

El esquema de un sistema OFDM se presenta en la Figura 1.10, en la cual se pueden observar los siguientes bloques:

Codificación/Decodificación FEC

Es un mecanismo corrector de errores que utiliza un código corrector de errores, la codificación se realiza bit a bit y presenta una tasa máxima de corrección. Ayuda a reducir el número de errores y de retransmisiones en entornos ruidosos.

Entrelazado/Desentrelazado de bits

Este bloque ayuda a combatir errores en ráfagas, consiste en convertir los errores agrupados en aislados con lo cual se logra un mejor desempeño del código corrector de errores.

Modulación/Demodulación QPSK/QAM

Este bloque realiza la modulación digital tanto en amplitud como en fase, utilizando alguno de los siguientes esquemas de modulación BPSK, QPSK, 16QAM o 64QAM. En recepción se realiza el proceso contrario.

Modulación/Demodulación OFDM

Realiza el ensamblaje del símbolo OFDM que consiste en la conversión de serial a paralelo, la asignación del esquema de portadoras, la inserción de portadoras piloto y portadoras virtuales, la IFFT, la conversión de paralelo a serial y la inserción del prefijo cíclico.

Estimación de canal

Utilizando los símbolos enviados en las subportadoras piloto se encuentra la respuesta en frecuencia del canal para posteriormente utilizarla en el proceso de ecualización. Este proceso se realiza en el receptor.

Ecualización

La ecualización compensa los efectos del canal en la señal OFDM usando un valor complejo por cada una de las subportadoras. El valor complejo necesario para la ecualización se obtiene en el proceso de estimación de canal. La ecualización se realiza en recepción previo a la detección de símbolos.

1.3.3.2 Ventajas de OFDM

El esquema de transmisión OFDM es un tipo de sistema multicanal, que emplea múltiples subportadoras. Como se muestra en la Figura 1.9, los espectros de las diferentes subportadoras se encuentran superpuestos aumentando la eficiencia espectral [1].

Ya que la velocidad de cada subportadora es N veces más lenta que en la señal original, el nivel de la ISI resultante debido al multitrayecto es mucho menor. Otra ventaja de OFDM es que debido al uso del entrelazado (*interleaving)*, los errores pueden ser fácilmente recuperados usando técnicas de corrección de errores [12].

Si la longitud del Prefijo Cíclico es mayor que la longitud del canal, la convolución lineal se convierte en convolución circular, como resultado de esto, el efecto de la ISI y de la interferencia entre subportadoras (*Inter-carrier interference*, ICI) se remueve fácilmente [12].

En señales convencionales en el dominio del tiempo, el proceso de ecualización consiste en una serie de operaciones de convolución, cuya longitud es proporcional al máximo

retardo del canal. Eso significa que para algunos canales con dispersión en tiempo el receptor puede ser muy complejo. En sistemas OFDM, la ecualización se realiza como una simple multiplicación entre la señal recibida y la respuesta en frecuencia de cada subcanal. Esto representa una gran ventaja en términos de requerimientos de procesamiento [12].

1.3.4 Estimación de canal

En un sistema OFDM, el transmisor modula la secuencia de bits del mensaje en símbolos PSK / QAM, posteriormente realiza la IFFT convirtiendo los símbolos en señales en el dominio del tiempo y los envía a través de un canal (inalámbrico). La señal recibida suele estar distorsionada por las características del canal. Para recuperar los bits transmitidos, el efecto del canal se debe estimarse y compensarse en el receptor [1].

Como se mencionó anteriormente, cada subportadora puede considerarse como un canal independiente, cuando no ocurre ICI, ya que se preserva la ortogonalidad entre ellas. La ortogonalidad permite que cada subportadora de la señal recibida se exprese como el producto de la señal transmitida y la respuesta en frecuencia del canal de esa subportadora. Por lo tanto, la señal transmitida puede recuperarse estimando la respuesta del canal de cada subportadora [1].

En general, el canal puede estimarse utilizando un preámbulo o símbolos piloto conocidos tanto por el transmisor como por el receptor. Para elegir la técnica de estimación de canal para un sistema OFDM se deben tener en cuenta muchos aspectos como: el rendimiento requerido, la complejidad computacional y la variación temporal del canal [1].

1.3.4.1 Distribución de pilotos

Dependiendo de la disposición de los pilotos, se consideran tres tipos de distribuciones de pilotos: *block type, comb type* y *lattice type* [1].

Block Type

En la Figura 1.11 se muestra una disposición de pilotos *block type*. En este tipo de distribución se transmiten periódicamente símbolos OFDM con pilotos en todas sus subportadoras. Usando estos pilotos, se realiza una interpolación en el dominio del tiempo para estimar el canal a lo largo del eje temporal [1].

Con el fin de realizar un seguimiento de las características de un canal variable en el tiempo, los símbolos piloto deben colocarse con un espaciamiento (S_t) menor que el tiempo de coherencia, el cual se relaciona con la frecuencia Doppler de forma inversa, por lo cual la periodicidad en el tiempo de los símbolo piloto (S_t) debe satisfacer la siguiente desigualdad [1]:

$$S_t \leq \frac{1}{f_{Doppler}}$$

Ecuación 1.17 Periodicidad de los símbolos piloto para *block type*.

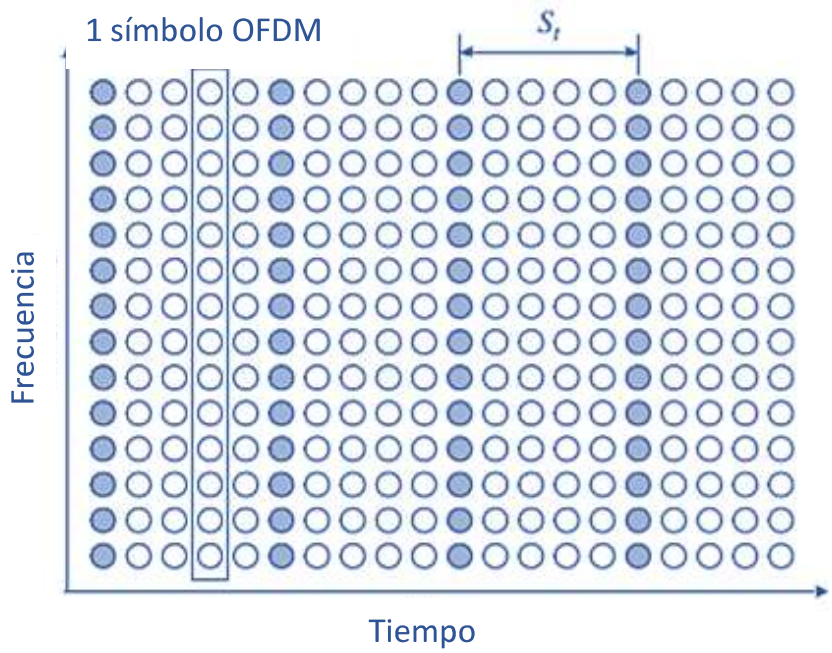

Figura 1.11 Distribución de pilotos *block type* [1].

Dado que los pilotos se insertan en todas las subportadoras del símbolo piloto con un cierto período de tiempo, este tipo de distribución de pilotos es adecuada para canales selectivos en frecuencia. En cambio, en canales con un desvanecimiento rápido se puede incurrir en un exceso de símbolos piloto, debido a que es necesario reducir el período con el que se envían los símbolos pilotos para lograr detectar la variación del canal [1].

Comb Type

La distribución de pilotos *comb type* se muestra en la Figura 1.12. En esta distribución se tiene los pilotos insertados cada cierto número de subportadoras en todos los símbolos OFDM, estos pilotos se usan para una interpolación en el dominio de la frecuencia [1].

Con el fin de realizar un seguimiento de las características de un canal selectivo en frecuencia, los símbolos piloto deben colocarse con un espaciamiento (S_f) menor que el ancho de banda de coherencia. Como el ancho de banda de coherencia es el inverso del máximo *delay spread* σ_{max}, la periodicidad en frecuencia de los símbolo piloto (S_f) debe satisfacer la siguiente desigualdad [1]:

$$S_f \leq \frac{1}{\sigma_{max}}$$

Ecuación 1.18 Periodicidad de los símbolos piloto para *comb type*.

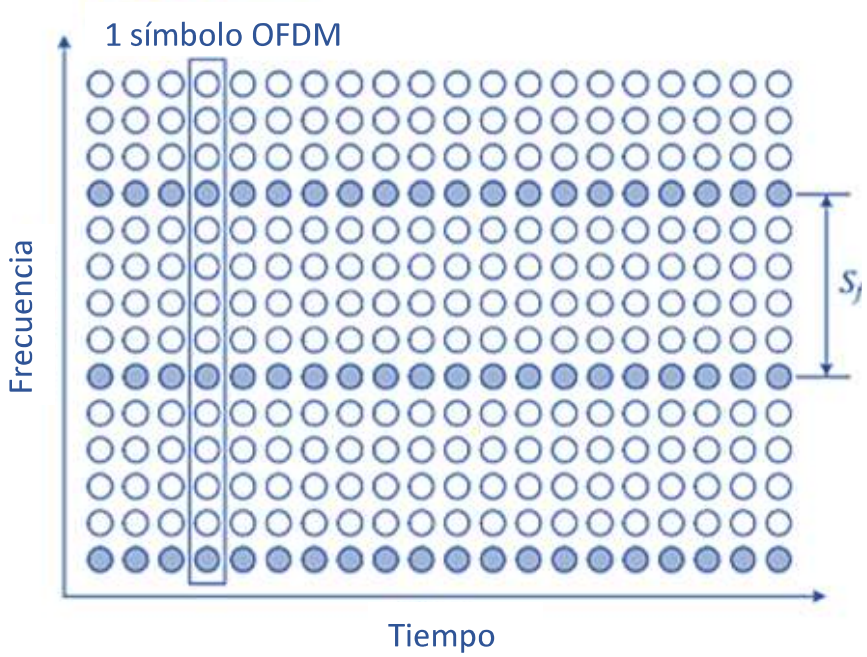

Figura 1.12 Distribución de pilotos *comb type* [1].

A diferencia de la distribución de pilotos *block type*, la distribución *comb type* es adecuada para canales con desvanecimiento rápido en el tiempo, pero no para canales selectivos en frecuencia [1].

Lattice Type

La distribución de pilotos *lattice type* se muestra en la Figura 1.13. En esta distribución, los pilotos se insertan a lo largo de los ejes del tiempo y de la frecuencia con ciertos períodos determinados. Los pilotos que se encuentran dispersos tanto en los ejes del tiempo como de la frecuencia facilitan las interpolaciones en los dos ejes para lograr la estimación del canal [1].

Con el fin de realizar un seguimiento de las características de un canal variable en el tiempo y selectivo en frecuencia, la disposición de los pilotos debe satisfacer la Ecuación 1.17 y la Ecuación 1.18, de modo que [1]:

$$S_t \leq \frac{1}{f_{Doppler}} \; y \;\; S_f \leq \frac{1}{\sigma_{max}}$$

Ecuación 1.19 Periodicidad de los símbolos piloto para *lattice type*.

Donde: S_t y S_f son la periodicidad en tiempo y en frecuencia de los símbolo piloto, respectivamente $f_{Doppler}$ y σ_{max} representan la frecuencia Doppler y el máximo *delay spread*, respectivamente [1].

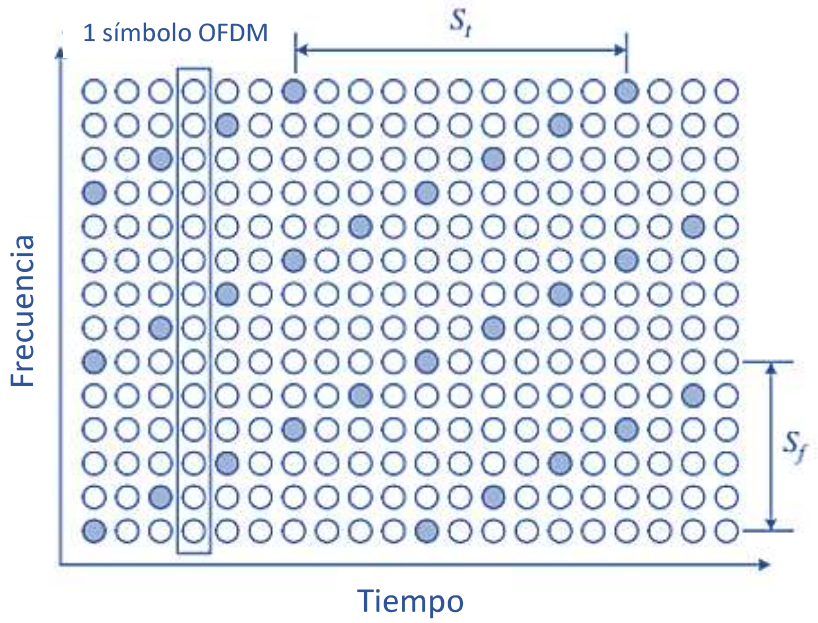

Figura 1.13 Distribución de pilotos *lattice type* [1].

1.3.4.2 Técnicas de estimación de canal

Se pueden usar símbolos de entrenamiento (preámbulo o símbolos piloto) para estimar el canal, lo cual generalmente proporciona un buen rendimiento. Sin embargo, su uso reduce la eficiencia de la transmisión. Al disponer de símbolos de entrenamiento se puede utilizar diferentes técnicas de estimación de canal entre las que se destacan: *Least-Square* (LS) y *Minimum-Mean-Square-Error* (MMSE) [1].

Si todas las subportadoras son ortogonales (es decir, sin ICI), los símbolos de entrenamiento para N subportadoras se pueden representar mediante la siguiente matriz diagonal [1] :

$$\boldsymbol{X} = \begin{bmatrix} X[0] & 0 & \cdots & 0 \\ 0 & X[1] & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \ldots & 0 & X[N-1] \end{bmatrix}$$

Ecuación 1.20 Representación de los símbolos de entrenamiento transmitidos.

Donde $X[k]$ es el piloto en la subportadora k-ésima, con $E\{X[k]\} = 0$ y con una $Var\{X[K]\} = \sigma^2_X$, $k=0,..,N-1$. $\boldsymbol{X}$ se define como una matriz diagonal, ya que se asume que todas las

subportadoras son ortogonales. Dado que la ganancia del canal es *H[k]* para cada subportadora *k*, la señal de entrenamiento recibida *Y[k]* se puede representarse como [1]:

$$\boldsymbol{Y} \triangleq \begin{bmatrix} Y[0] \\ Y[1] \\ \vdots \\ Y[N-1] \end{bmatrix} = \begin{bmatrix} X[0] & 0 & \cdots & 0 \\ 0 & X[1] & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \ldots & 0 & X[N-1] \end{bmatrix} \begin{bmatrix} H[0] \\ H[1] \\ \vdots \\ H[N-1] \end{bmatrix} + \begin{bmatrix} Z[0] \\ Z[1] \\ \vdots \\ Z[N-1] \end{bmatrix}$$

$$= \boldsymbol{XH} + \boldsymbol{Z}$$

Ecuación 1.21 Representación de los símbolos de entrenamiento recibidos.

Donde $\boldsymbol{H}$ es el vector del canal dado como $\boldsymbol{H}$*=[H[0],…,H[N-1]]*^{T} y $\boldsymbol{Z}$ es el vector de ruido dado como $\boldsymbol{Z}$*=[Z[0],…,Z[N-1]]*^{T} con *E{Z[k]}=0* y con una *Var{Z[K]}=*σ^2_Z**,** *k=0,..,N-1*. Para la siguiente sección se supondrá que $\widehat{\boldsymbol{H}}$ denota la estimación del canal $\boldsymbol{H}$ [1].

Estimación de canal tipo LS

El método de estimación de canal usando mínimos cuadrados (*Least Square,* LS) encuentra la estimación de canal $\widehat{\boldsymbol{H}}$ de tal manera que se minimiza la siguiente función de costo [1]:

$$\boldsymbol{J}(\widehat{\mathrm{H}}) = \left\| \boldsymbol{Y} - \boldsymbol{X}\widehat{\mathbf{H}} \right\|^2$$

Ecuación 1.22 Función costo de la estimación de canal tipo LS.

Al realizar la derivada de la función anterior ecuación con respecto a $\widehat{\boldsymbol{H}}$ e igualarla a cero se tiene $\boldsymbol{X^H X\widehat{H}=X^H Y}$, lo cual soluciona la estimación de canal tipo LS [1]:

$$\widehat{\mathbf{H}}_{\boldsymbol{LS}} = (\boldsymbol{X^H X})^{-1}\boldsymbol{X^H Y} = \boldsymbol{X^{-1}Y}$$

Ecuación 1.23 Estimación de canal tipo LS.

Siendo cada componente del canal LS estimado $\widehat{\mathrm{H}}_{\mathbf{LS}}$ dado por $\widehat{\mathrm{H}}_{\mathrm{LS}}$[k], k=0,..,N-1. Como se supone que $\boldsymbol{X}$ es diagonal debido a la condición libre de ICI, la estimación del canal $\widehat{\mathbf{H}}_{\mathbf{LS}}$ se puede escribir para cada subportadora como [1]:

$$\widehat{\mathrm{H}}_{LS} = \frac{Y[K]}{X[K]}, \qquad 0,1,2,..,\mathrm{N}-1$$

Ecuación 1.24 Estimación de canal tipo LS para cada subportadora.

El error cuadrático medio (Mean Squared Error, MSE) de la estimación de canal tipo LS se define como [1]:

$$MSE_{LS} = \frac{{\sigma_z}^2}{{\sigma_x}^2}$$

Ecuación 1.25 Error cuadrático medio del estimador tipo LS.

Hay que tener en cuenta que el MSE en la Ecuación 1.25 es inversamente proporcional a la SNR, lo que implica que el MSE de esta estimación de canal será considerable para valores de SNR bajos. Sin embargo debido a su simplicidad, el método LS se ha utilizado ampliamente para la estimación de canal [1].

Estimación de canal tipo MMSE

El estimador tipo MMSE se define como [1]:

$$\widehat{\mathbf{H}} \triangleq \boldsymbol{W}\widetilde{\boldsymbol{H}}$$

Ecuación 1.26 Estimador tipo MMSE.

Donde $\widetilde{\boldsymbol{H}}$ es la solución del estimador tipo LS de la Ecuación 1.23 y $\boldsymbol{W}$ es una matriz de pesos. El estimador consta de la estructura mostrada en la Figura 1.14 [1].

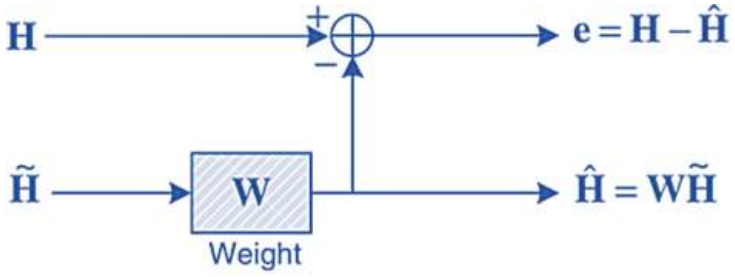

Figura 1.14 Estimador tipo MMSE [1].

El método de estimación de canal tipo MMSE presenta un mejor desempeño que el método LS, ya que minimiza el MSE de la Ecuación 1.27 [1].

$$J(\widehat{\mathrm{H}}) = E\{\|e\|^2\} = E\{\left\|\boldsymbol{H} - \widehat{\mathbf{H}}\right\|^2\}$$

Ecuación 1.27 Función costo de la estimación de canal tipo MMSE.

Para minimizar el MSE de la ecuación anterior se utiliza el principio de ortogonalidad entre el vector de error de la estimación ($\boldsymbol{e} = \boldsymbol{H} - \widehat{\mathbf{H}}$) y la estimación de canal tipo LS ($\widetilde{\boldsymbol{H}}$), demostrando que [1]:

$$\boldsymbol{E}\{\boldsymbol{e}\widetilde{\boldsymbol{H}}^{\boldsymbol{H}}\} = \boldsymbol{R}_{\boldsymbol{H}\widetilde{\boldsymbol{H}}} - \boldsymbol{W}\boldsymbol{R}_{\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{H}}} = 0$$

Ecuación 1.28 Minimización del MSE del estimador MMSE.

De la anterior ecuación se obtiene [1]:

$$\boldsymbol{W} = \boldsymbol{R}_{H\widetilde{H}} - \boldsymbol{R}_{\widetilde{H}\widetilde{H}}{}^{-1}$$

Ecuación 1.29 Matriz de pesos (***W***).

En donde $R_{H\widetilde{H}}$ (ver Ecuación 1.30) es la matriz de correlación cruzada entre el vector de la respuesta real del canal (***H***) y el vector de la estimación de canal LS ($\widetilde{\boldsymbol{H}}$) y $\boldsymbol{R}_{\widetilde{H}\widetilde{H}}$ es la matriz de autocorrelación de $\widetilde{\boldsymbol{H}}$ (ver Ecuación 1.31**)** [1].

$$\boldsymbol{R}_{H\widetilde{H}} = E\{\boldsymbol{H}\widetilde{\boldsymbol{H}}^{\boldsymbol{H}}\}$$

Ecuación 1.30 Matriz de correlación cruzada entre $\boldsymbol{H}$ y $\widetilde{\mathbf{H}}$

$$\boldsymbol{R}_{\widetilde{H}\widetilde{H}} = E\{\boldsymbol{H}\boldsymbol{H}^{\boldsymbol{H}}\} + \frac{{\sigma_z}^2}{{\sigma_x}^2}\boldsymbol{I} = \boldsymbol{R}_{\boldsymbol{HH}} + \frac{{\sigma_z}^2}{{\sigma_x}^2}\boldsymbol{I}$$

Ecuación 1.31 Matriz de autocorrelación de $\widetilde{\mathbf{H}}$.

Usando las tres ecuaciones anteriores en la Ecuación 1.26, la estimación de canal tipo MMSE se puede obtener como [1]:

$$\widehat{\boldsymbol{H}} = \boldsymbol{W}\widetilde{\boldsymbol{H}} = \boldsymbol{R}_{H\widetilde{H}}\boldsymbol{R}_{\widetilde{H}\widetilde{H}}{}^{-1}\widetilde{\boldsymbol{H}} = \boldsymbol{R}_{H\widetilde{H}}\left(\boldsymbol{R}_{\boldsymbol{HH}} + \frac{{\sigma_z}^2}{{\sigma_x}^2}\boldsymbol{I}\right)^{-1}\widetilde{\boldsymbol{H}}$$

Ecuación 1.32 Ecuación del estimador de canal tipo MMSE.

Los elementos de $\boldsymbol{R}_{H\widetilde{H}}$ y $\boldsymbol{R}_{\boldsymbol{HH}}$ en la Ecuación 1.32 son [1]:

$$E\{h_{k,l}\,\widetilde{h}_{k',l'}\} = E\{h_{k,l}\,h^*{}_{k',l'}\} = r_f[k-k']r_t[l-l']$$

Ecuación 1.33 Elementos de las matrices de correlación cruzada y autocorrelación.

Donde *k* y *l* denotan el índice de subportadora y el índice del símbolo OFDM, respectivamente.

Usando el PDP de un canal con multitrayecto donde las rutas disminuyen su potencia exponencialmente, la correlación en el dominio de la frecuencia $r_f[k]$ se obtiene como [1]:

$$r_f[k] = \frac{1}{1 + j2\pi\tau_{rms}k\Delta f}$$

Ecuación 1.34 Correlación en el dominio de la frecuencia para un PDP exponencial.

Donde $\Delta f = 1/T_{sub}$ y representa el espaciado entre subportadoras para una FFT de longitud igual a T_{sub}.

Para estimar la respuesta del canal de las subportadoras de datos, la estimación para las subportadoras piloto debe interpolarse. Los métodos populares de interpolación incluyen interpolación lineal, interpolación polinomial de segundo orden e interpolación *spline* cúbica [1].

1.3.4.3 Métodos de interpolación

En muchos problemas de ingeniería, el valor de una función se conoce solo en algún conjunto discreto de valores de la variable o variables independientes. Esto puede surgir porque la función se ha calculado o evaluado por el usuario solo en un conjunto discreto de valores, o porque se ha proporcionado un conjunto de valores al usuario solo para un número discreto de valores de la variable independiente. Un problema numérico clásico que surge para esos casos es cómo obtener un valor aproximado de la función en otros valores de la variable independiente [13].

La interpolación es el proceso de estimar en una función un valor intermedio a partir de una tabla de valores conocida, encontrando valores *y* y *x* que no existían en la tabla. Una situación física donde ocurren estas tablas es el uso de datos experimentales [13].

Interpolación lineal

La interpolación lineal es el método más simple. Este método se usa por programas graficadores de funciones, donde se interpola usando líneas rectas entre una serie de puntos. Un ejemplo de interpolación lineal se ilustra en la Figura 1.15, donde la curva de puntos representa una relación funcional desconocida entre los valores (*x,y*), y los 6 puntos de datos representan los pares de datos tabulados. Además, en esa misma figura se representa la interpolación lineal mediante segmentos de línea sólidos dibujados entre los puntos de datos adyacentes [13].

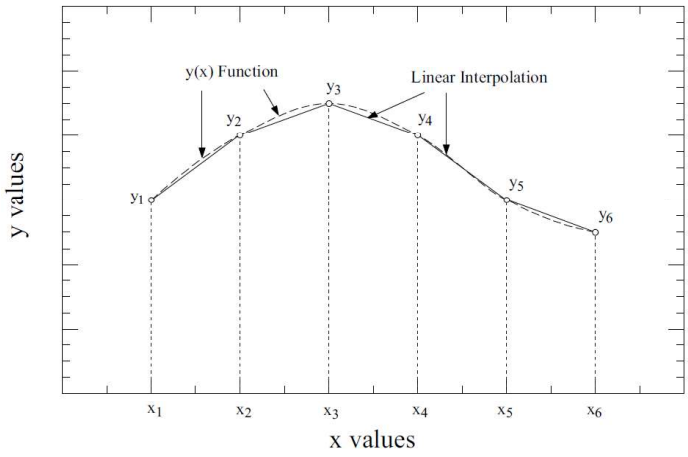

Figura 1.15 Interpolación lineal [13].

La idea básica de la interpolación consiste en conectar los 2 puntos dados en x_i, es decir (x_1, y_1) y (x_2, y_2). La función interpolante en este tipo de interpolación es una línea recta entre los dos puntos. Para cualquier punto entre los dos valores de x_1 y x_2 se debe seguir la ecuación de la recta [14]:

$$y = y_1 + (y_2 - y_1) * \frac{x - x_1}{x_2 - x_1}$$

Ecuación 1.35 Ecuación de la recta.

Donde se asume que $x_1 < x < x_2$, de otra forma se conoce como extrapolación. Si se tienen más de dos puntos para la interpolación, es decir $N > 2$, con puntos $x_1, x_2, \ldots, x_N$, simplemente se concatena la interpolación lineal entre pares de puntos continuos [14].

La interpolación lineal funciona mejor cuando se tiene un gran número de datos y la variación entre los puntos de datos es un pequeño porcentaje del rango general de la función. Cuando los datos tabulados están muy espaciados por lo general se utiliza técnicas de interpolación más avanzadas como la que se mostrará a continuación [13].

Interpolación *spline* cúbico

El objetivo de los *splines* cúbicos es generar una función de interpolación usando un polinomio cúbico que tenga una primera derivada suavizada y una segunda derivada continua, tanto dentro de los intervalos como en los puntos x_k como se puede observar en la siguiente Figura [14].

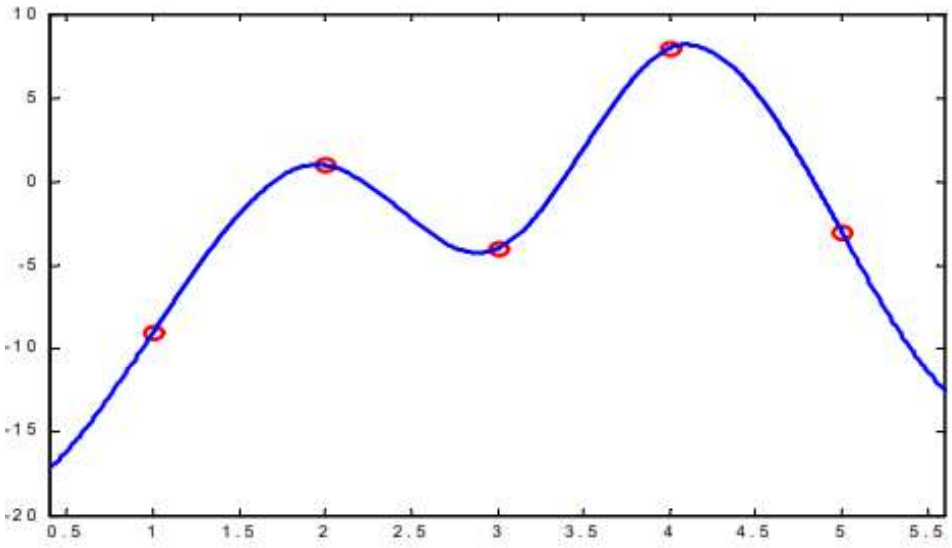

Figura 1.16 Interpolación *spline* cúbico [15].

Para lograr este tipo de interpolación se utiliza:

$$y = Ay_j + By_{j+1} + Cy_j'' + Dy_{j+1}''$$

Ecuación 1.36 Interpolación *spline* cúbico [16].

donde se define:

$$A = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2 \qquad D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

Ecuación 1.37 Coeficientes de la función *spline* [16].

Se puede observar en la Ecuación 1.37 que la variable independiente *x* se encuentra relacionada de manera lineal con *A* y *B* y de forma cubica con *C* y *D* [16].

El termino interpolación *spline* puede extenderse más allá de la idea presentada, ya que existen métodos multidimensionales, splines de mayor orden *(high-order)* con nodos variables y *smoothing splines* [14].

2. METODOLOGÍA

El capítulo 2 se centra en el desarrollo de la aplicación que realizará la estimación de canal, para lo cual se ha utilizado Matlab como software de simulación. El sistema de comunicaciones implementado consta de: transmisor OFDM, canal Rayleigh, receptor OFDM, así como de los dos métodos de estimación de canal (LS y MMSE) que se aplicarán en el presente proyecto técnico. Además, en este capítulo se explicará brevemente como se realizó la programación de las diferentes funciones, así como del programa principal y una breve explicación de los elementos de la interfaz gráfica que se utilizará.

Para el presente proyecto se utilizó tanto la investigación explicativa como la investigación aplicada. La primera se puede observar en el capítulo 1 en el cual se determina las causas de los fenómenos del canal inalámbrico. En cambio, el segundo tipo de investigación está presente en la simulación del sistema de comunicaciones inalámbrico.

El diagrama de flujo del sistema OFDM implementado se muestra en la Figura 2.1, en este se observa que se tiene dos bucles: el primero se refiere al valor de la Eb/No de la señal y el segundo es respecto al número de transmisiones que se realizará por cada Eb/No. En el segundo bucle se tiene el transmisor OFDM, el canal tipo Rayleigh y el receptor OFDM. Ya que la estimación de canal es un proceso que se realiza en recepción se explicará con detalle este proceso en la sección que trata acerca del receptor OFDM implementado.

2.1 Generalidades de Matlab

Matlab, acrónimo de *MATrix LABoratory,* es un software que permite realizar cálculos científicos y tecnológicos complejos usando una representación de valores basada en matrices [17]. Matlab integra un conjunto de librerías y comandos de alto nivel muy versátil y completo que lo hace idóneo para trabajar con simulaciones dentro de múltiples ámbitos de la ingeniería. Esta herramienta de software ofrece un entorno interactivo muy sencillo que permite tanto la programación de algoritmos, la entrada y salida de datos de una forma versátil y funcional, así como la visualización de estos de una forma sencilla y cómoda. Matlab facilita la resolución de problemas de una forma más cómoda que mediante el uso de lenguajes de propósito general como C y C++ al integrar un poderoso conjunto de librerías [17].

Matlab ofrece excelentes capacidades gráficas para la representación de datos y su lenguaje de operación propio (Lenguaje M) es facil de emplear, pudiendo interaccionar con herramientas como Excel asi como con otros lenguajes como C o Java.

Para el presente proyecto técnico se ha utilizado la versión R2018B de Matlab. En el desarrollo de la programación se utilizó tanto funciones predefinidas de Matlab, así como sistemas de objetos [1] y además se creó diferentes funciones. Adicionalmente se desarrolló una aplicación utilizando *App Designer*, del cual se tratará más detalladamente a continuación.

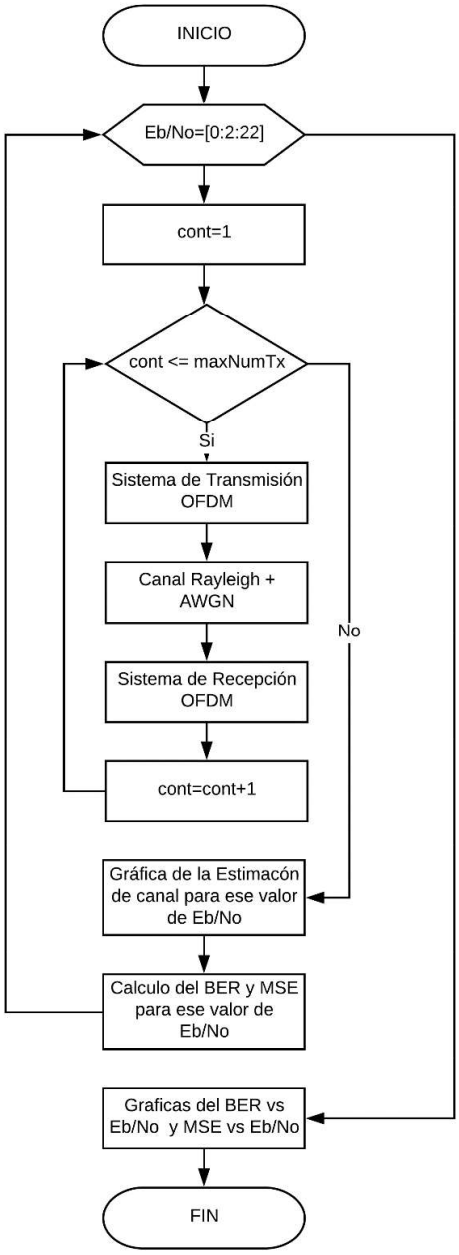

Figura 2.1 Diagrama de flujo del programa principal.

[1] Sistema de Objetos: Es un objeto especial de Matlab creado para simular sistemas con entradas dinámicas, este objeto necesita primero ser creado para posteriormente ser ejecutado.

2.1.1 *App Designer*

App Designer es un entorno para crear aplicaciones en Matlab. Este entorno simplifica el proceso de diseño de los componentes visuales de la interfaz de usuario, ya que incluye un conjunto completo de componentes para su diseño [18]. *App Designer* integra las dos tareas principales de la creación de aplicaciones que son: diseñar los componentes visuales de la interfaz gráfica de usuario (*Graphical User Interface*, GUI) (ver Figura 2.2 izquierda) y programar el comportamiento de la aplicación (ver Figura 2.2 derecha).

En *App Designer* para el diseño de la interfaz de usuario basta con arrastrar y soltar los componentes visuales en la interfaz de diseño. *App Designer* generará automáticamente un código orientado a objetos que especifica el diseño y la distribución de la aplicación. El usuario puede usar una versión integrada del editor de Matlab para definir el comportamiento de la aplicación [18].

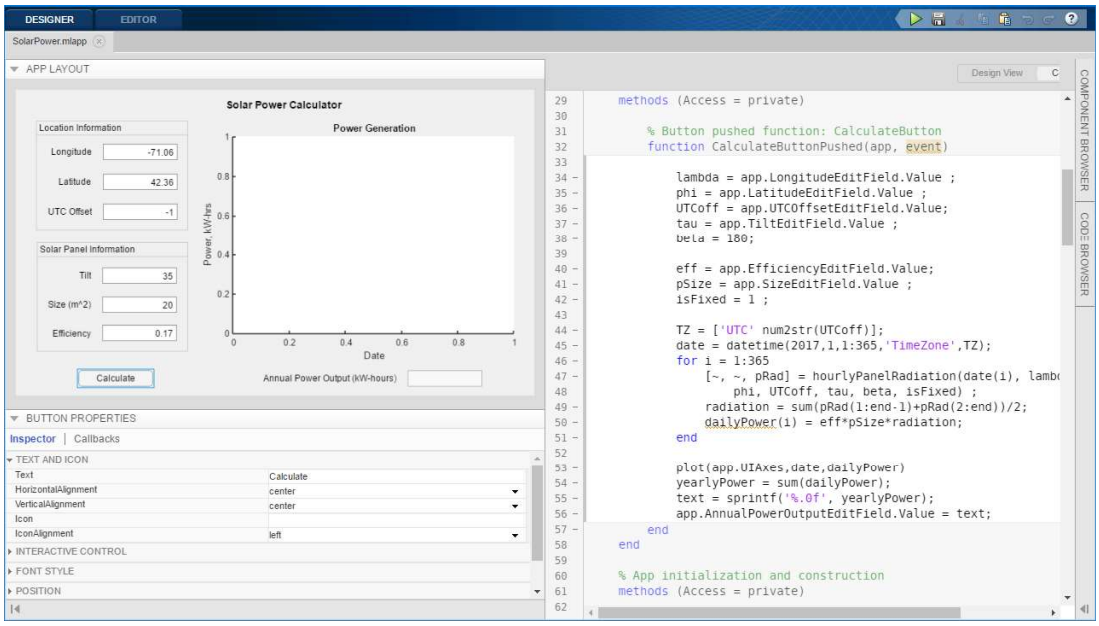

Figura 2.2 Interfaz de *App Designer* [18].

Además de componentes estándar como: botones, casillas de verificación, árboles y listas desplegables, *App Designer* proporciona controles como: medidores, lámparas, perillas e interruptores que permiten replicar la apariencia y acciones de paneles de instrumentación. Igualmente, se puede utilizar gráficos en 2D y 3D, así como tablas para presentar los resultados en la aplicación. También se puede usar componentes de contenedor como pestañas y paneles para organizar la interfaz gráfica de usuario [18].

2.1.1.1 Características de *App Designer*:

- Un entorno de diseño mejorado respecto a GUIDE (*GUI Development Environment*) y una nueva estructura de generación de código [18].
- Un conjunto completo de componentes para el diseño de interfaces de usuario. Además de un selector de fecha, una tabla mejorada, componentes para crear paneles de control e interfaces hombre-máquina [18].
- Capacidad de implementación en la web para compartir la aplicación con cualquier persona de la organización o ejecutarla en *Matlab online* para un acceso más fácil [18].

2.2 Interfaz Gráfica

Para el diseño de la interfaz de la aplicación (ver Figura 2.3) se ha utilizado diferentes elementos que ofrece *App Designer* tales como paneles, grupos de pestañas, botones, menús desplegables, casillas de verificación, *axes, spinner, radio button group, list box*, campos editables y áreas para mostrar texto.

Figura 2.3 Interfaz gráfica implementada.

Con respecto a la interfaz se la ha dividido en dos áreas principales que son el panel de ingreso de datos (ver Figura 2.3 izquierda) y el panel de resultados (ver Figura 2.3 derecha). En la parte inferior del panel de ingreso de datos se tiene tres botones los cuales son Deshacer, Graficar y Borrar. En este mismo panel se presenta dos pestañas, la primera

pestaña permite acceder a las opciones de la transmisión, mientras que la segunda permite al usuario la configuración del canal inalámbrico. En el área de resultados se tiene tres pestañas las cuales permiten la visualización de: la curva de la BER vs. Eb/No, la curva del MSE vs. Eb/No y diferentes estimaciones de canal para Eb/No que van desde 0 dB hasta 22 dB en pasos de 2 dB.

Las opciones presentes en la pestaña de transmisión (ver Figura 2.4) son: si se realizará la estimación perfecta de canal, usar codificación convolucional con tasa ½ e interleaving, seleccionar el número de transmisiones a realizarse, el método de estimación, el tipo de interpolación y el esquema de pilotos que se utilizará en la transmisión.

Figura 2.4 Opciones de la pestaña Transmisión.

En la pestaña de ingreso de las configuraciones del canal (ver Figura 2.5) se puede seleccionar si se utilizará un canal con 3 rayos o con 4 rayos, se puede ingresar la potencia promedio de los rayos y el retardo que tendrá cada uno de ellos. Además, en la parte inferior se presenta una nota que advierte al usuario que para usar la estimación de canal tipo MMSE el canal ingresado deberá ser de tipo exponencial.

Figura 2.5 Opciones de la pestaña Canal.

2.3 Configuración del sistema OFDM

Previo a realizar el proceso de transmisión es necesario definir ciertos parámetros y configurar ciertos aspectos de la comunicación. Además, es necesario generar ciertos valores que serán constantes para toda la transmisión. Para realizar lo anteriormente mencionado se utilizó las configuraciones que se observan en el diagrama de flujo de la Figura 2.6, en el cual se ilustra que previo a la transmisión es necesario: fijar los parámetros del sistema de transmisión (ver Segmento de código 2.1), recolectar las opciones ingresadas por el usuario (ver Segmento de código 2.2.), generar los pilotos dependiendo del esquema de pilotos seleccionado por el usuario (ver Segmento de código 2.3), calcular la SNR en función de la Eb/No (ver Segmento de código 2.4) , configurar los moduladores y demoduladores (QAM y OFDM) y el canal tipo Rayleigh+AWGN (ver Segmento de código 2.5).

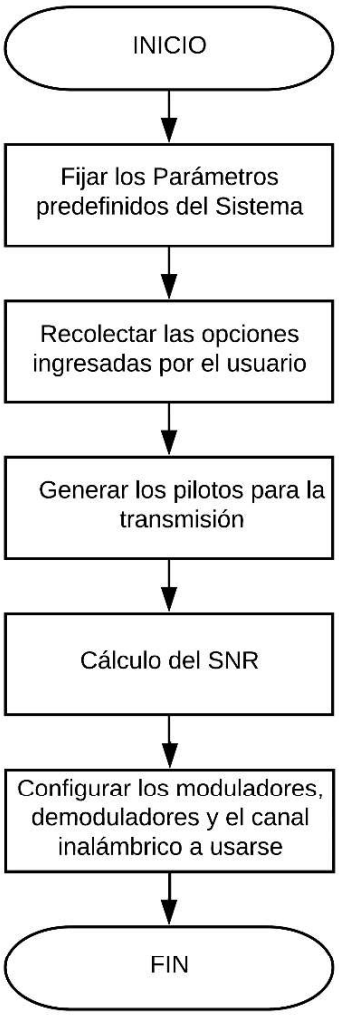

Figura 2.6 Configuración del Sistema OFDM

2.3.1 Parámetros predefinidos del sistema OFDM

Los parámetros que se han predefinido para el sistema OFDM se especifican en la Tabla 2.1.

Tabla 2.1 Parámetros del sistema.

PARÁMETRO	**VALOR**
Número de subportadoras nulas (VC)	12
Número de subportadoras de datos+pilotos	52
Número de subportadoras totales	64
Tamaño de la IFFT	64
Longitud del prefijo cíclico	16

Además de los parámetros especificados en la tabla anterior se ha definido las posiciones de las subportadoras nulas (VC) que estarán ubicadas en (1,2,3,4,5,6,33,60,61,62,63,64) y se ha definido un tamaño de bloque de 3 símbolos OFDM en el caso de usarse un esquema de portadoras *block type*, esto se observa en el Segmento de código 2.1.

```
nFFT = 64;                                  % Longitud de la IFFT
cpLen = nFFT/4;                             % Longitud del prefijo cíclico
nullIdx = [1:6 33 60:64].';                 % Subportadoras nulas
BlockSize=3;                                % Tamaño del bloque
trellis = poly2trellis(7,[171 133]);        % Codificador convolucional
tbl = 32;                                   % Codificador convolucional
rate = 1/2;                                 % Tasa de codificación
```

Segmento de código 2.1 Parámetros predefinidos para la transmisión.

2.3.2 Opciones ingresadas por el usuario

Se requiere que el sistema OFDM a implementarse sea configurable para lo cual se ha utilizado la interfaz de la Figura 2.3. Las opciones que el usuario puede seleccionar para la transmisión se las resume en la Tabla 2.2.

Tabla 2.2 Opciones del sistema OFDM.

OPCIONES DEL SISTEMA OFDM	
Estimación perfecta de canal	Si, No
Codificación convolucional (Tasa 1/2) + *Interleaving*	Si, No
Número de transmisiones	Seleccionable por el usuario
Esquema de modulación	QPSK, 16QAM, 64QAM
Método de estimación	LS, MMSE
Tipo de interpolación	Lineal, *Spline*
Esquema de pilotos	*Comb* cada 8 subportadoras *Comb* cada 4 subportadoras *Block*

Para recolectar las opciones de la transmisión ingresadas por el usuario en la interfaz (ver Figura 2.4) se utiliza la función especificada en el Segmento de código 2.2.

```
function
[modOrd,maxNumTx,EsqPort,EstPerfCanal,TipoInterp,TipoEstimacion,cod]=ingreso(app)
app.Modulacion.ItemsData = [2 4 6];   % Orden de Modulación (QPSK=2,16QAM=4,64QAM=6)
modOrd= app.Modulacion.Value;
maxNumTx=app.Transmisiones.Value;     % Número maximo de transmisiones
app.Pilotos.ItemsData = [1 2 3];      % Esquema de Pilotos (Comb8=1,Comb4=2,Block=3)
EsqPort=app.Pilotos.Value;
cod=app.Codificacion.Value;           % ON(1) OFF(0) Codificación e Interleaving
app.Estimacion.ItemsData = [1 2];     % Tipo de Estimación (LS=1 o MMSE=2)
TipoEstimacion= app.Estimacion.Value;
TipoInterp= app.Interpolacion.Value;  % Tipo de Interpolación Usada(lineal o spline)
s=app.Perfecta.SelectedObject.Text;   % Estimación Perfecta de Canal
switch s
    case 'No'
        EstPerfCanal=0;
    case 'Si'
        EstPerfCanal=1;
end
end
```

Segmento de código 2.2 Opciones de transmisión ingresadas por el usuario.

2.3.3 Generación de los pilotos

En la modulación OFDM se realizará la inserción de las subportadoras pilotos para lo cual es necesario que previamente los pilotos se hayan generado usando alguna de las estructuras de piloto especificadas en las tablas 2.3-2.5. Los pilotos son necesarios, ya que serán utilizados en el proceso de estimación de canal. Para la generación de las subportadoras se ha utilizado el Segmento de código 2.3, en el cual se utiliza los índices de las estructuras de pilotos especificadas en las tablas 2.3-2-5 y se generan 1L que serán modulados en BPSK (*Binary Phase Shift Keying*).

Tabla 2.3 Estructura de los pilotos para la distribución *Comb Type* cada 8 subportadoras.

# Port	1-6	7	15	23	31	33	39	47	55	60-64
	VC	P	P	P	P	VC	P	P	P	VC

Tabla 2.4 Estructura de los pilotos para la distribución *Comb Type* cada 4 subportadoras.

# Port	1-6	7	11	15	19	23	27	31	33	35	39	43	47	51	55	59	60-64
	VC	P	P	P	P	P	P	P	VC	P	P	P	P	P	P	P	VC

Tabla 2.5 Estructura de los pilotos para la distribución *Block Type.*

# Port	1-6	7-32	33	34-59	60-64
	VC	P/D	VC	P/D	VC

donde P=Piloto, VC= Subportadora Nula (*Virtual Carrier)* y D=Subportadora de Datos.

```
function [pilotIdx,pilots,numTones]=Generatepilots(app,EsqPort,nFFT,nullIdx)
switch EsqPort
    case 1 %Comb8
        pilotIdx = (7:8:55)';                              % Índice de los pilotos
        pilots = pskmod(ones(1,7).',4);                    % Generar los pilotos
        numTones = nFFT-length(nullIdx)-length(pilotIdx); % Num subportad de datos
    case 2 %Comb4
        pilotIdx = (7:4:59)';                              % Índice de los pilotos
        pilots = pskmod(ones(1,14).',4);                   % Generar los pilotos
        numTones = nFFT-length(nullIdx)-length(pilotIdx); % Num subportad de datos
    case 3 %Block
        pilotIdx = [(7:32),(34:59)]';                      % Índice de los pilotos
        pilots = pskmod(ones(1,52).',4);                   % Generar los pilotos
        numTones = nFFT-length(nullIdx);                   % Num subportad de datos
end
end
```

Segmento de código 2.3 Generar los Pilotos.

2.3.4 Cálculo de la SNR

Se utiliza la relación Eb/No para el cálculo de la SNR tanto en el caso que se usa codificación convolucional como cuando no se usa. En el caso de usar codificación convolucional se considera la tasa de codificación para el cálculo de la SNR como se puede observar en el Segmento de código 2.4.

```
if cod==0
    snrVec = EbNoVec + 10*log10(modOrd) + 10*log10(numTones/nFFT);
else
    snrVec = EbNoVec + 10*log10(modOrd*rate) + 10*log10(numTones/nFFT);
end
```

Segmento de código 2.4 Configuración de parámetros de la transmisión.

2.3.5 Configuración de los moduladores y canal inalámbrico

La función configurar (ver Segmento de código 2.5) realiza la creación y la configuración de los moduladores y demoduladores QAM con potencia ponderada, además configura el canal AWGN de manera que la relación Eb/No pueda ser ingresada de manera externa. En este mismo código se define parámetros para el canal tipo Rayleigh como son: el ancho de banda y la frecuencia de muestreo de la señal de entrada. Finalmente, se obtiene el PDP ingresado por el usuario (ver Figura 2.5) y se calcula el *RMS delay spread* utilizando las ecuaciones 1.4, 1.5 y 1.6.

```
function [qam,qamdem,chanAWGN,chan,symOffset,tau_rms]=configurar(app,modOrd,cpLen)
% Creación del modulador y demodulador QAM con potencia ponderada
qam =
comm.RectangularQAMModulator('ModulationOrder',2^modOrd,'BitInput',true,'Normalizati
onMethod','Average power');
qamdem =
comm.RectangularQAMDemodulator('ModulationOrder',2^modOrd,'BitOutput',true,'Normaliz
ationMethod','Average power');
% Config el canal AWGN de manera que se pueda ingresar de manera externa la Eb/No
chanAWGN = comm.AWGNChannel('NoiseMethod','Variance','VarianceSource','Input port');
% Canal Rayleigh
maxDopp = 0;       % Ancho de banda de Doppler
sRate = 20e6;      % Frecuencia de muestreo de la señal de entrada
s2=app.Rayleigh.SelectedObject.Text;
switch s2
    case '3 Rayos'
        pathGains=[app.PDP1_3.Value,app.PDP2_3.Value,app.PDP3_3.Value];  % Potencias
        pathDelays=[app.Delay1_3.Value,app.Delay2_3.Value,app.Delay3_3.Value];%Delay
        pathDelays=pathDelays*1e-9;
    case '4 Rayos'
        pathGains=[app.PDP1_4.Value,app.PDP2_4.Value,...
            app.PDP3_4.Value,app.PDP4_4.Value];                          % Potencias
        pathDelays=[app.Delay1_4.Value,app.Delay2_4.Value,...
            app.Delay3_4.Value,app.Delay4_4.Value];                       % Delays
        pathDelays=pathDelays*1e-9;
end
chan =
comm.RayleighChannel('PathGainsOutputPort',true,'MaximumDopplerShift',maxDopp,...
    'PathDelays',pathDelays,'AveragePathGains',pathGains,'SampleRate',sRate);
sampIdx = round(pathDelays*sRate) + 1;
symOffset = min(max(sampIdx),cpLen); % Determine el desface del muestreo del símbolo
% Respuesta impulsiva del canal
posicion=round(pathDelays(2:end)*sRate);
hImp=zeros(1,posicion(end));
hImp(posicion)=pathGains(2:end);
hImp=[pathGains(1) hImp];
% Obtener RMS delay spread
k=0:length(hImp)-1;
hh = hImp*hImp';
tmp = hImp.*conj(hImp).*k;
r = sum(tmp)/hh;             % Mean excess delay (ecuación 1.4)
r2 = tmp*k.'/hh;             % Segundo Momento central del PDP (ecuación 1.6)
tau_rms = sqrt(r2-r^2);      % RMS delay spread (ecuación 1.5)
end
```

Segmento de código 2.5 Configuración de los moduladores y del canal *Rayleigh+AWGN.*

2.4 Sistema de transmisión OFDM

El sistema de transmisión implementado se muestra en la Figura 2.7 en donde la codificación FEC y el entrelazado de bits se realizarán solo en el caso de que el usuario lo haya seleccionado.

Figura 2.7 Diagrama de bloques del transmisor OFDM.

Para la implementación del transmisor se ha utilizado el Segmento de código 2.6 en el cual se genera un número de bits a transmitirse que depende de: el número de portadoras de datos, del esquema de modulación que se utilizará (QPSK, 16QAM, 64QAM) y de si se utilizará o no codificación convolucional. Posterior a lo cual se realizará la codificación convolucional y el entrelazado de bits, solo en caso de que el usuario haya seleccionado esta opción. Finalmente se realiza las modulaciones QPSK/QAM y OFDM.

```
if cod==0
% Generar los datos a tx
dataIn = randi(s1,[0 1],modOrd*numTones,1);
% Modulación QPSK/QAM y OFDM
txSig=transmision(app,qam,dataIn,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Blo
ckSize);
else
% Generar los datos a tx
dataIn = randi(s1,[0 1],modOrd*numTones*rate,1);
dataCod=codifyinterl(app,dataIn,trellis,EsqPort,modOrd);
% Modulación QPSK/QAM y OFDM
txSig=transmision(app,qam,dataCod,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Bl
ockSize);
end
```

Segmento de código 2.6 Transmisor.

2.4.1 Codificación FEC

Para la codificación FEC se ha utilizado la función dataEnc incluida en el *toolbox* de comunicaciones de Matlab, en la cual es necesario el ingreso de los datos a codificar y la estructura de trellis a utilizarse, esta estructura fue previamente especificada en el Segmento de código 2.1. La codificación FEC con tasa ½ se puede observar en la segunda línea del Segmento de código 2.7.

```
function dataCod=codifyinterl(app,dataIn,trellis,EsqPort,modOrd)
dataEnc = convenc(dataIn,trellis); %Codificación
%Interleaving
NCBPS=length(dataEnc);
switch EsqPort
    case 1       % Comb8
        Ncol=15; % Número de columnas
    case 2       % Comb4
        Ncol=19; % Número de columnas
    case 3       % Block
        Ncol=13; % Número de columnas
end
Nrow=NCBPS/Ncol; % Número de filas
% Primera Permutación
k=[0:NCBPS-1];
i=(Nrow)*mod(k,Ncol)+floor(k/Ncol);                % Indice i
Tx=ordenar(app,i,dataEnc);
% Segunda Permutación
s=max(modOrd/2,1);
i=[0:NCBPS-1];
j=s*floor(i/s)+mod((i+NCBPS-floor(Ncol*i/NCBPS)),s); % Indice j
dataCod=(ordenar(app,j,Tx))';
end
```

Segmento de código 2.7 Codificación FEC e interleaving.

2.4.2 Entrelazado de bits

Se ha implementado un esquema similar al presente en el estándar IEEE 802.11 en el cual se realizan dos entrelazados, el primero evita que bits continuos se transmitan en la misma subportadora y el segundo evita que bits consecutivos se envíen en un mismo símbolo QAM o QPSK. Este entrelazado de bits es de tipo bloque con un tamaño de bloque igual al número de bits en un símbolo OFDM (N_{CBPS})[19].

Sea *k* el índice de los bits a codificar antes de la primera permutación, *i* el índice después de la primera permutación y *j* el índice después de la segunda permutación. La primera permutación que asegura que bits codificados adyacentes sean asignados a subportadoras no adyacentes se define como [19], [20]:

$$\mathrm{i} = (N_{row}) * \mathrm{mod}(\mathrm{k}, \mathrm{N}_{col}) + \mathrm{floor}\left(\frac{\mathrm{k}}{\mathrm{N}_{col}}\right)$$

Ecuación 2.1 Índice *i* para el primer entrelazado.

Donde *k= 0, 1,…, N_{CBPS} -1*. N_{col} es el número de columnas definido como 15, 19 y 13 para los esquemas de portadoras *comb type* cada 8 portadoras, *comb type* cada 4 portadoras y *block type* respectivamente. N_{row} es el número de filas, el cual se obtiene dividiendo el

número de bits para el numero de columnas. La operación módulo se denota como *mod()* y la función *floor ()* representa el entero más grande que no excede el parámetro.

La segunda permutación evita las cadenas de bits de baja fiabilidad, por lo cual asigna bits adyacentes a diferentes símbolos en la constelación. Esta permutación se define como [19], [20]:

$$j = s * \text{floor}\left(\frac{i}{s}\right) + \text{mod}\left(\left(i + N_{CBPS} - \text{floor}\left(N_{col} * \frac{i}{N_{CBPS}}\right)\right), s\right)$$

Ecuación 2.2 Índice *j* para el segundo entrelazado.

La variable *i* puede tomar valores *i= 0, 1, …, N_{CBPS} -1* y siendo *s* el desplazamiento en el segundo entrelazado [19], [20].

$$s = \max\left(\frac{\text{modOrd}}{2}, 1\right)$$

Ecuación 2.3 Desplazamiento en el segundo entrelazado.

Donde modOrd es el número de bits por símbolo que es: 2 en QPSK, 4 en16QAM y 6 en 64QAM. La implementación del entrelazado se observa en el Segmento de código 2.7.

2.4.3 Modulación QAM/QPSK

Para la modulación QAM/QPSK se han utilizado los moduladores previamente configurados en el Segmento de código 2.5, por lo cual únicamente se necesita especificar los datos de entrada, como se puede observar en el Segmento de código 2.8.

```
function
txSig=transmision(app,qam,dataIn,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Blo
ckSize)
if EsqPort <=2 % Comb8 y Comb4
    qamTx = qam(dataIn);                                        % Modular en QAM
    txSig = ofdmmod(qamTx,nFFT,cpLen,nullIdx,pilotIdx,pilots); % Modular en OFDM
else           % Block
    if (rem(cont,BlockSize))==1 % Los simbolos son los pilotos
        qamTx=pilots;
    else                        % Los símbolos son datos
        qamTx = qam(dataIn);                                    % Modular en QAM
    end
    txSig = ofdmmod(qamTx,nFFT,cpLen,nullIdx);                  % Modular en OFDM
end
end
```

Segmento de código 2.8 Transmisión.

2.4.4 Modulación OFDM

Para la implementación de la modulación OFDM se ha utilizado la función del *toolbox* de comunicaciones de Matlab ofdmmod, la cual permite realizar todo el procesamiento OFDM (conversión serial paralelo, IFFT, insertar las subportadoras piloto, conversión paralelo serial, insertar el prefijo cíclico).

En la función ofdmmod se define la estructura de portadoras mostrada en la Figura 2.8, donde las primeras y últimas subportadoras son las bandas de guarda. La numeración de las subportadoras es desde 1 hasta *n* donde *n* es el número total de subportadoras. El uso de la función ofdmmod se puede observar en el Segmento de código 2.8.

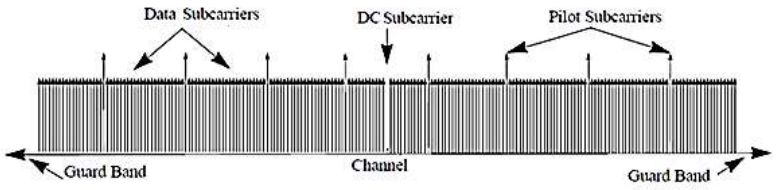

Figura 2.8 Distribución de portadoras definida en la función ofdmmod

2.5 Canal Rayleigh

El canal Rayleigh se ha implementado siguiendo la estructura especificada en el diagrama de flujo de la Figura 2.9, en la cual se hace uso de una función que reinicia el canal (ver Segmento de código 2.9) y otra función que pasa la señal a través del canal previamente creado (ver Segmento de código 2.10).

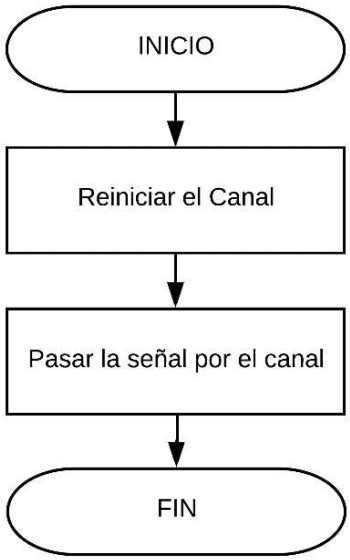

Figura 2.9 Implementación del Canal Rayleigh

2.5.1 Reiniciar el canal

La función Channel (ver Segmento de código 2.9) reinicia el canal en cada transmisión cuando se utiliza un esquema de pilotos tipo *comb type*, en cambio al usar un esquema de pilotos *block type* mantiene el mismo canal por un número de transmisiones especificado en el Segmento de código 2.1 y cuando llega a ese número que es 3 se reinicia el canal. Además de reiniciar el canal, esta función obtiene la respuesta en frecuencia del canal que se usará posteriormente para obtener el MSE.

```
function [H,Hdat]=Channel(app,nFFT,cpLen,symOffset,chan,EsqPort,cont,BlockSize)
if EsqPort <=2                  % Comb8 y Comb4
    reset(chan);                % Reiniciar el canal
else
    if (rem(cont,BlockSize))==1 % Block
        reset(chan);            % Reiniciar el Canal
    end
end
% Obtener el canal
datos= ones(nFFT,1);                          % Generar los datos a Tx
l=ofdmmod(datos,nFFT,cpLen);                  % Modular en OFDM
filtrado=chan(l);                             % Aplicar a la Señal el Canal
H=ofdmdemod(filtrado,nFFT,cpLen,symOffset);   % Demodulación OFDM
Hdat=seleccion(app,H,EsqPort);                % Selec solo las portadoras de datos
end
```

Segmento de código 2.9 Reiniciar el canal.

2.5.2 Aplicar el Canal

La función AplyChannel aplica a la señal el canal Rayleigh previamente creado en el Segmento de código 2.5. Posteriormente esta función calcula la potencia de la señal y el *noise variance* utilizado por la función chanAWGN. La función chanAWGN pasa la señal a través del canal AWGN. Con todo el procedimiento descrito se obtiene un canal Rayleigh+AWGN como se puede observar en el Segmento de código 2.10.

```
function rxSig=AplyChannel(app,txSig,chan,chanAWGN,snr)
fadSig = chan(txSig);               % Aplicar el canal RAYLEIGH
powerDB = 10*log10(var(fadSig));    % Calcular la potencia de la señal Tx
noiseVar = 10.^((powerDB-snr)/10);  % Calular el noise variance
rxSig = chanAWGN(fadSig,noiseVar);  % Aplicar el canal AWGN
end
```

Segmento de código 2.10 Aplicar el canal.

2.6 Sistema de recepción OFDM

El sistema de recepción implementado se muestra en la Figura 2.10, donde la decodificación FEC y el desentrelazado de bits solo se realizarán en caso de que en transmisión se hayan realizado los procesos inversos.

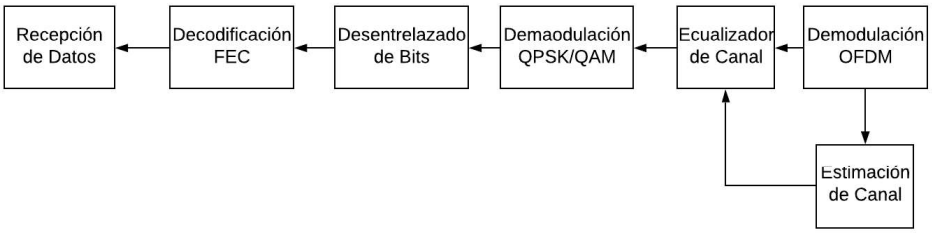

Figura 2.10 Diagrama de Bloques del receptor OFDM.

Para la implementación del receptor se ha utilizado el Segmento de código 2.11, en el cual primero se realiza la demodulación OFDM, posteriormente se realiza la estimación de canal usando los pilotos obtenidos del proceso de demodulación OFDM y luego se realiza la interpolación para obtener la respuesta en frecuencia para las subportadoras de datos. En caso de usarse una distribución tipo bloque no se realizará la interpolación, tampoco se realizará en caso de usarse el algoritmo de estimación de canal tipo MMSE. Usando la respuesta en frecuencia se realizará la ecualización y posterior a esta se realizará la demodulación QPSK/QAM. Finalmente se realizará el desentrelazado y la decodificación en caso de que el usuario seleccionó esta opción.

```
   if EsqPort <=2
            % Demodulación OFDM
            [x,pilotosRx] = ofdmdemod(rxSig,nFFT,cpLen,symOffset,nullIdx,pilotIdx);
        else
            % Demodulación OFDM
            x= ofdmdemod(rxSig,nFFT,cpLen,symOffset,nullIdx);
        end
        if EsqPort==3 && (rem(cont,BlockSize))==1
            pilotosRx=x;
        end
hpilotos=estimacion(app,pilotosRx,pilots,TipoEstimacion,numTones,tau_rms,snr,EsqPort
);
[h,hdat]=interpolacion(app,pilotIdx,hpilotos,nFFT,EsqPort,TipoInterp,TipoEstimacion)
;
% Ecualización y Demodulación QPSK o QAM
dataOut=ecualizacion(app,x,hdat,Hdat,qamdem,EstPerfCanal);
if cod==1
% Decodificación y Desentrenlazado
dataOut=decodifyinterl(app,dataOut,trellis,tbl,EsqPort,modOrd);
end
```

Segmento de código 2.11 Receptor.

2.6.1 Demodulación OFDM

Para la implementación de la demodulación OFDM se ha utilizado la función ofdmdemod del toolbox de comunicaciones de Matlab, la cual es la función reciproca a la usada en transmisión ofdmmod. Ofdmdemod realiza la extracción del prefijo cíclico, la conversión serial paralelo, separa las subportadoras piloto de los datos y realiza la conversión de paralelo a serial. El uso de la función ofdmdemod se puede observar en el Segmento de código 2.11.

2.6.2 Estimación de Canal

Para realizar la estimación de canal se ha utilizado la función estimación, la cual se puede observar en el Segmento de código 2.12. Se ha implementado los algoritmos de estimación de canal: LS y MMSE.

Para la estimación usando el algoritmo LS simplemente se ha realizado la división entre los pilotos recibidos y los pilotos trasmitidos, con lo cual se obtiene la respuesta en frecuencia del canal para las subportadoras piloto. Cabe recalcar que al usar este método no se considera los efectos del ruido, por lo cual en la sección de resultados se podrá observar que este método presenta bastantes errores para relaciones Eb/No bajas.

En caso de realizar la estimación de canal usando el algoritmo MMSE, primero se realiza la división de las subportadoras recibidas para las transmitidas, de la misma forma que se realizó para el método LS (estimación de canal tipo LS). Luego de lo cual, se define el espaciado que existen entre subportadoras piloto. Para el caso de *comb type* cada 8 subportadoras el espaciado es de 8, para *comb type* cada 4 subportadoras el espaciado es de 4 y para *block type* el espaciado es de 1. El espaciado de 1 en *block type* se debe a que en todas las subportadoras del símbolo OFDM se envía pilotos.

Posteriormente, se utiliza la Ecuación 1.33 para encontrar las variables auxiliares K_1,K_2,K_3,K_4 las cuales junto con el *mean excess delay* que se obtuvo en el Segmento de código 2.5 y en conjunto con las ecuaciones 1.33, 1.29. 1.30 permiten el cálculo de: la correlación para un canal con una distribución exponencial, la matriz de autocorrelación de la estimación tipo LS y la matriz de correlación cruzada entre el canal y la estimación de canal tipo LS.

Finalmente, usando la matriz de autocorrelación, la matriz de correlación cruzada y la matriz con la estimación de canal tipo LS se obtiene la estimación de canal tipo MMSE.

```
function
hpilotos=estimacion(app,pilotosRx,pilots,TipoEstimacion,numTones,tau_rms,snr,EsqPort
)
if TipoEstimacion==1                        % Estimación de canal LS
    hpilotos =pilotosRx./pilots;
else                                        % Estimación de canal MMSE
    H_tilde = (conj(pilotosRx./pilots))'; % Estimación tipo LS
    switch EsqPort
        case 1      % Comb8
            Nps=8;  % Espaciado entre pilotos
            Np=length(pilotosRx);
        case 2      % Comb4
            Nps=4;  % Espaciado entre pilotos
            Np=length(pilotosRx);
        case 3      % Block
            Nps=1;  % Espaciado entre pilotos
            Np=length(pilotosRx)+1;
            H_tilde=[H_tilde(1:26),H_tilde(26:52)];
    end
    snrlineal = 10^(snr*0.1);
    K1 = repmat((0:1:52).',1,Np); % 52 subportadoras entre datos y pilotos
    K2 = repmat((0:Np-1),52+1,1);
    K3 = repmat((0:Np-1).',1,Np);
    K4 = repmat((0:Np-1),Np,1);
    df = 1/numTones;
    j2pi_tau_df = 1i*2*pi*tau_rms*df;
    % Correlación para un PDP exponencial(ecuacion 1.33)
    rf2 = 1./(1+j2pi_tau_df*Nps*(K3-K4));
    % Correlación cruzada entre H y H_tilde(ecuación 1.29)
    Rhp = 1./(1+j2pi_tau_df*(K1-K2*Nps));
    % Matriz de autocorrelación de H_tilde(ecuación 1.30)
    Rpp = rf2 + eye(length(H_tilde),length(H_tilde))/snrlineal;
    % Estimador de canal tipo MMSE(ecuación 1.31)
    hpilotos = transpose((Rhp/Rpp)*H_tilde.');
end
end
```

Segmento de código 2.12 Estimación de canal.

2.6.2.1 Interpolación

En algunos tipos de estimación es necesario realizar interpolación luego de haber encontrado la respuesta en frecuencia para las subportadoras piloto. Al utilizarse el algoritmo de estimación de canal tipo MMSE no es necesario realizar interpolación, ya que este algoritmo permite encontrar las componentes de frecuencia para todas las subportadoras. Igualmente, si se utiliza una distribución *block type* no es necesario realizar interpolación temporal ya que se asumió que la respuesta del canal no cambia para un número de símbolos OFDM igual al tamaño del bloque, posterior a lo cual se recalculará la respuesta en frecuencia del canal.

Los tipos de interpolación implementados son lineal y *spline* como se observa en el Segmento de código 2.13.

```
function
[h,hdat]=interpolacion(app,pilotIdx,hpilotos,nFFT,EsqPort,TipoInterp,TipoEstimacion)
if TipoEstimacion==1 % Estimación LS
    if EsqPort <=2   % Comb8 y Comb4
        h = interp1(pilotIdx,hpilotos,(1:1:nFFT)',TipoInterp,'extrap');  % Interpola
    else             % Block
        h = [NaN(6,1);(hpilotos(1:1:26));NaN(1,1);(hpilotos(27:52));NaN(5,1)];
    end
else                 % Estimación MMSE
    h=[NaN(6,1);conj(hpilotos');NaN(5,1)];  % No interpola
end
hdat=seleccion(app,h,EsqPort);
end
```

Segmento de código 2.13 Interpolación.

2.6.3 Ecualización

Para el proceso de ecualización se ha utilizado la función descrita en el Segmento de código 2.14 en la cual se divide la señal demodulada de OFDM para la estimación de canal previamente calculada.

```
function dataOut=ecualizacion(app,x,hdat,Hdat,qamdem,EstPerfCanal)
if EstPerfCanal==0
    eqH = conj(hdat)./(conj(hdat).*hdat);
else
    eqH = conj(Hdat)./(conj(Hdat).*Hdat);    % Estimación perfecta de canal
end
eqSig = eqH.*x;                              % Ecualización
dataOut = qamdem(eqSig);                     % Demodulación QAM o QPSK de la señal
end
```

Segmento de código 2.14 Ecualización.

2.6.4 Demodulación QPSK/QAM

Para la demodulación QPSK/QAM se han utilizado los demoduladores previamente configurados en el Segmento de código 2.5, por lo cual únicamente es necesario especificar los datos de entrada, como se puede observar en la penúltima línea del Segmento de código 2.14.

2.6.5 Desentrelazado de bits

La implementación del desentrelazado se puede observar en el Segmento de código 2.15, en donde el número de columnas se lo ha definido como 15, 19 y 13 para los esquemas de portadoras *comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type* respectivamente.

La primera permutación del desentrelazador que invierte la segunda permutación realizada por el entrelazador se define como:

$$i = s * \text{floor}\left(\frac{j}{s}\right) + \text{mod}\left(\left(j + \text{floor}\left(N_{col} * \frac{j}{N_{CBPS}}\right)\right), s\right)$$

Ecuación 2.4 Índice *i* para el primer desentrelazado [19], [20].

La segunda permutación del desentrelazador que invierte la primera permutación realizada por el entrelazador se define como:

$$k = N_{col} * i - (N_{CBPS} - 1) * \text{floor}\left(\frac{i}{N_{row}}\right)$$

Ecuación 2.5 Índice *k* para el segundo desentrelazado [19], [20].

```
function dataDecod=decodifyinterl(app,dataOut,trellis,tbl,EsqPort,modOrd)
% Deinterleaving
NCBPS=length(dataOut);
switch EsqPort
    case 1       % Comb8
        Ncol=15; % Número de columnas
    case 2       % Comb4
        Ncol=19; % Número de columnas
    case 3       % Block
        Ncol=13; % Número de columnas
end
Nrow=NCBPS/Ncol; % Número de filas
% Inverso de la Segunda Permutación
j=[0:NCBPS-1];
s=max(modOrd/2,1);
i=s*floor(j/s)+mod((j+floor(Ncol*j/NCBPS)),s); % Indice i
Rx=ordenar(app,i,dataOut);
% Inverso de la Primera Permutación
i=[0:NCBPS-1];
k=Ncol*i-(NCBPS-1)*floor(i/Nrow);              % Indice k
Rx2=(ordenar(app,k,Rx))';
dataDecod = vitdec(Rx2,trellis,tbl,'cont','hard'); % Decodificación
end
```

Segmento de código 2.15 Deinterleaving y Decodificación FEC.

2.6.6 Decodificación FEC

Para la decodificación FEC se ha utilizado la función vitdec incluida en el toolbox de comunicaciones de Matlab, en la cual es necesario el ingreso de los datos a decodificar, el número de símbolos de retraso(*tbl*) y la estructura de trellis a utilizarse. Esta estructura fue previamente especificada en el Segmento de código 2.1. La decodificación FEC se puede observar en el Segmento de código 2.15.

2.7 Cálculo de errores

El cálculo de errores para el presente trabajo consiste en obtener la BER y el MSE entre el canal real y el canal estimado.

2.7.1 Cálculo de la BER

Para el cálculo de la BER se ha utilizado la función errorRate de Matlab como se puede observar en el Segmento de código 2.16.

```
if EsqPort <=2
   if cod==0
      errorStats = errorRate(dataIn,dataOut,0);
    else
      errorStats=errorRate(dataIn(1:end-tbl),dataOut(tbl+1:end),0);
    end
else
    if (rem(cont,BlockSize))~= 1
           if cod==0
             errorStats = errorRate(dataIn,dataOut,0);
     else
         errorStats=errorRate(dataIn(1:end-tbl),dataOut(tbl+1:end),0);
           end
    end
end
cont=cont+1;
mse=immse(abs(Hdat),abs(hdat));
MSEVec(m)=MSEVec(m)+mse;
```

Segmento de código 2.16 Cálculo de la BER y MSE.

2.7.2 Cálculo del MSE

Para el cálculo del MSE se ha utilizado la función immse de Matlab como se puede observar en la penúltima línea del Segmento de código 2.16.

3. RESULTADOS Y DISCUSIÓN

En este capítulo se presentan los resultados de las simulaciones mediante: curvas del MSE (entre el canal real y la estimación de canal) versus la relación Eb/No para diferentes esquemas de modulación. También se presentan las curvas de la BER vs. Eb/No para diferentes esquemas de modulación, tanto sin codificación ni *interleaving*, así como con codificación convolucional e *interleaving*, para diferentes esquemas de pilotos (*comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type*).

3.1 Escenario de simulación

Las simulaciones se realizan utilizando la interfaz que se creó en *App Designer* de Matlab (ver Figura 2.3). El sistema OFDM implementado consiste en un transmisor, un receptor y un canal inalámbrico. El número de transmisiones por cada Eb/No se configuró en 10000 con el fin de obtener curvas con trazos definidos. El canal inalámbrico que se usó es un canal tipo exponencial, el mismo que se encuentra especificado en la Tabla 3.1.

Tabla 3.1 PDP del canal utilizado

Trayecto	Retraso Relativo(ns)	Potencia Promedio (dB)
1	0	0.0
2	100	-3
3	200	-6

En el sistema OFDM implementado se asume sincronización perfecta entre transmisor y receptor. Se ha implementado 3 tipos de esquemas de pilotos: *comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type*. En el caso de usar el esquema de pilotos *block type* se asumió que el canal no presenta variaciones en el tiempo, por lo cual no es necesaria la interpolación, simplemente el canal se mantendrá constante para un bloque de símbolos OFDM que para la simulación se ha definido en 3. Al utilizar el algoritmo MMSE tampoco se necesita realizar interpolación ya que este permite obtener la respuesta en frecuencia del canal para todas las subportadoras del símbolo OFDM. Los resultados de las simulaciones se presentan mediante curvas (MSE vs Eb/No y BER vs. Eb/No) en las cuales se compara los algoritmos LS y MMSE para diferentes esquemas de pilotos (*comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type*).

3.2 Pruebas de funcionamiento

Con el fin de comprobar que el estimador de canal implementado funcione correctamente, es decir que el canal estimado sea similar al canal real, se presentan curvas comparando el canal real versus la estimación de canal realizada. Además de graficar las estimaciones de canal, cuando se utiliza una distribución *comb type* y el algoritmo LS se grafica los

pilotos. Esto se realiza para observar cómo los métodos de interpolación afectan a la estimación de canal. Ya que los valores de Eb/No configurados en el sistema van desde 0 dB hasta 22 dB se seleccionó el valor intermedio de 10 dB para realizar las pruebas del correcto funcionamiento de la estimación de canal.

3.2.1 *Comb type* cada 8 subportadoras

En las Figuras 3.1 a 3.3 se presenta la respuesta del canal real (en azul) comparada con la estimación de canal usando una estructura *comb type* cada 8 subportadoras y las diferentes técnicas (en rojo). En las figuras se compara el canal real versus: estimación LS con interpolación lineal (Figura 3.1), estimación LS con interpolación *spline* (Figura 3.2) y estimación MMSE (Figura 3.3). En las Figuras 3.1 y 3.2 además se presenta los pilotos recibidos (en amarillo).

Al usar una estructura de pilotos *comb type,* donde se tiene un piloto cada 8 subportadoras, se tiene 7 pilotos y 45 subportadoras de datos en cada símbolo OFDM. Esta estructura se usa en las figuras 3.1-3.3, en las que se puede observar que la mejor estimación de canal es usando el algoritmo MMSE (ver Figura 3.3),seguido por el algoritmo LS con interpolación lineal (ver Figura 3.1) y finalmente la peor es al usar el algoritmo LS con interpolación *spline* (ver Figura 3.2), esto ocurre porque no se tiene los suficientes datos para que la interpolación tipo *spline* se ajuste adecuadamente a las variaciones del canal.

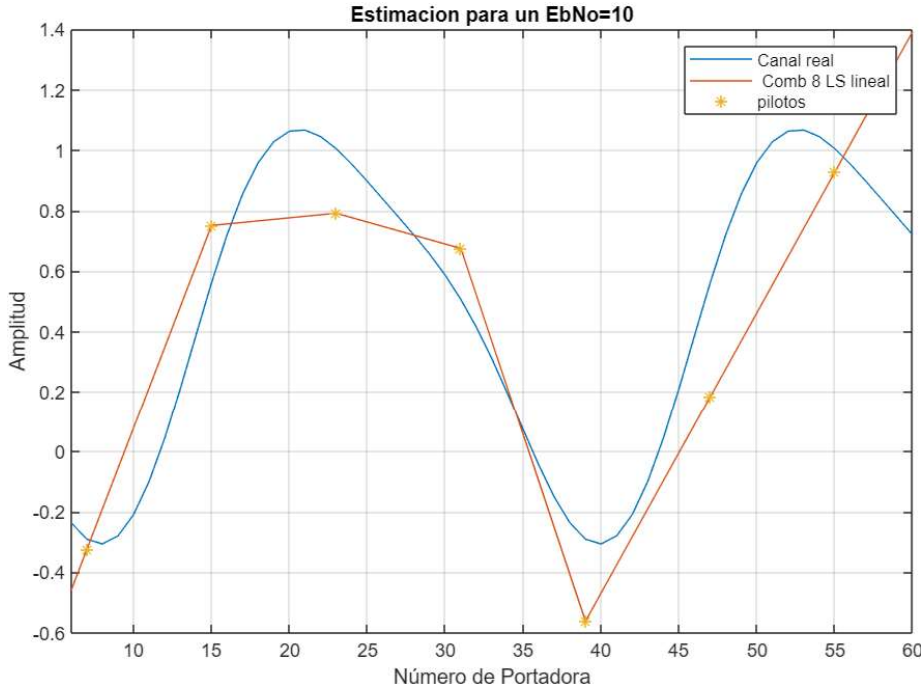

Figura 3.1 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 8 subportadoras, algoritmo LS e interpolación lineal

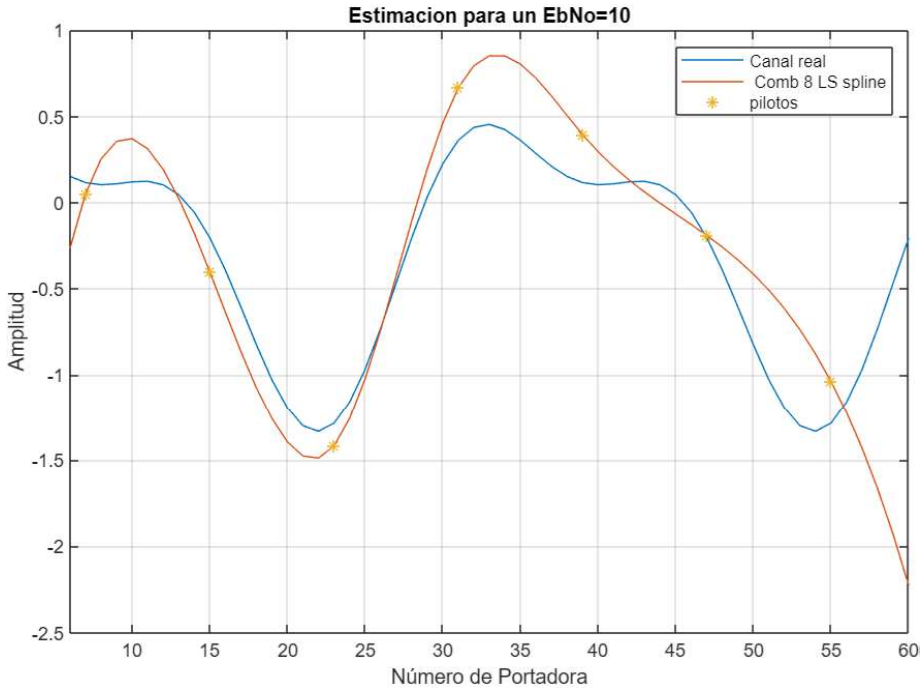

Figura 3.2 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 8 subportadoras, algoritmo LS e interpolación *spline*

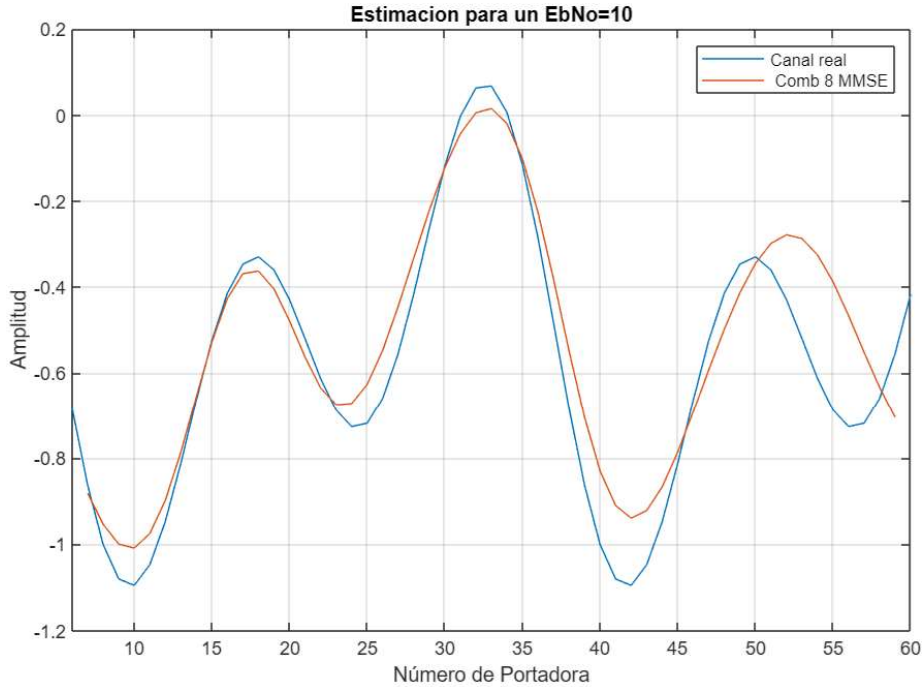

Figura 3.3 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 8 subportadoras y algoritmo MMSE

3.2.2 *Comb type* cada 4 subportadoras

En las Figuras 3.4-3.6 se presenta la respuesta del canal real (en azul) comparada con la estimación de canal usando una estructura *comb type* cada 4 subportadoras y las diferentes técnicas (en rojo. En las figuras se compara el canal real versus: estimación LS con interpolación lineal (Figura 3.4), estimación LS con interpolación *spline* (Figura 3.5) y estimación MMSE (Figura 3.6). En las Figuras 3.4 y 3.5 además se presenta los pilotos recibidos (en amarillo).

Usando una estructura de pilotos *comb type,* en donde se tiene un piloto cada 4 subportadoras se obtiene 14 pilotos y 38 subportadoras de datos en cada símbolo OFDM. Esta estructura de pilotos se usa en las siguientes figuras, en las cuales se observa que la mejor estimación de canal es usando el algoritmo MMSE (ver Figura 3.6), seguido por el algoritmo LS con interpolación *spline* (ver Figura 3.5) y finalmente la peor es al usar el algoritmo LS con interpolación lineal (ver Figura 3.4). Es importante señalar que la diferencia entre la interpolación tipo MMSE y la interpolación tipo LS usando interpolación tipo *spline* no es muy significativa, por tanto, al usar este tipo de distribución de pilotos las dos son buenas opciones de estimación de canal.

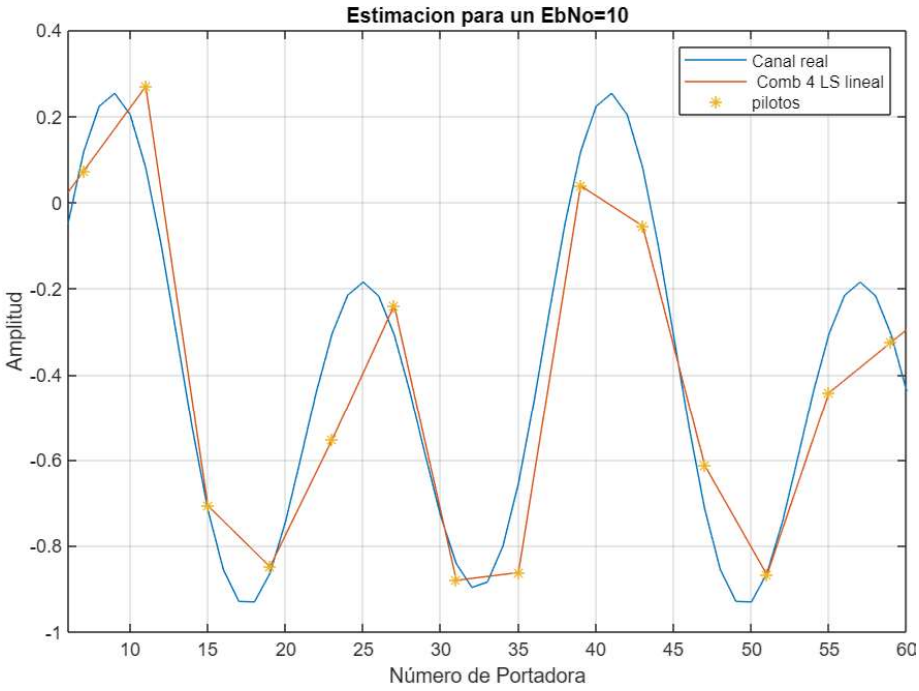

Figura 3.4 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 4 subportadoras, algoritmo LS e interpolación lineal

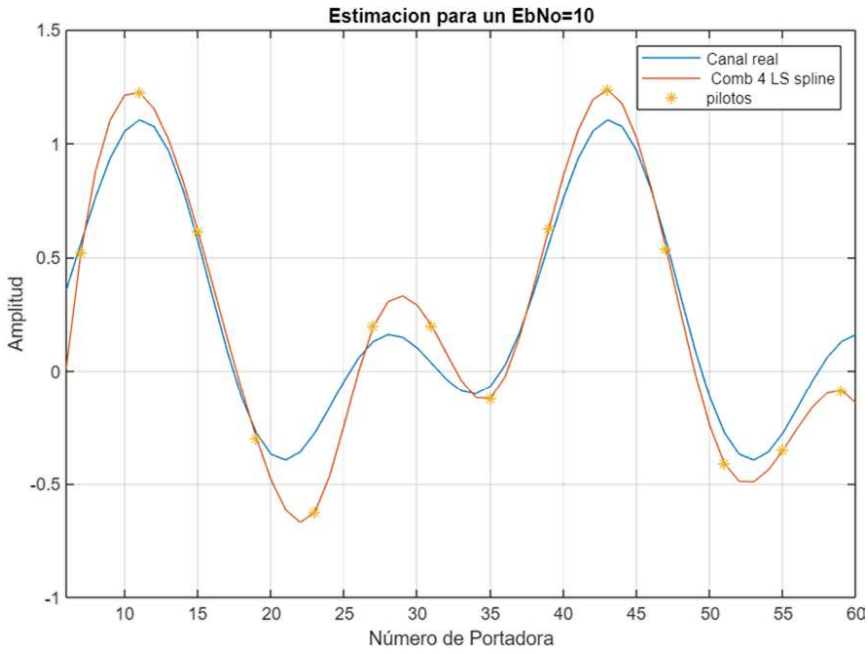

Figura 3.5 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 4 subportadoras, algoritmo LS e interpolación *spline*

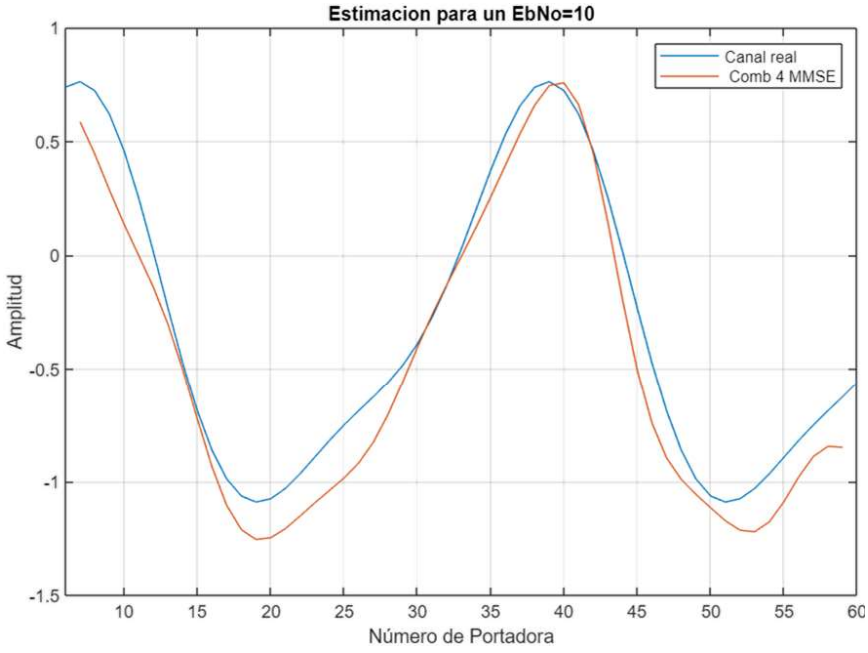

Figura 3.6 Canal real y canal estimado usando una estructura de pilotos *comb type* cada 4 subportadoras y algoritmo MMSE

3.2.3 *Block type*

En las Figuras 3.7 a 3.8 se compara el canal real versus: estimación LS (Figura 3.7) y estimación MMSE (Figura 3.8). En la estructura de pilotos *block type* se tiene el símbolo

OFDM lleno de pilotos, en consecuencia, se tiene la respuesta del canal para todas las subportadoras. En las siguientes figuras se observa que la mejor estimación es usando el algoritmo MMSE (ver Figura 3.8). En cambio, al usar el algoritmo LS (ver Figura 3.7) se observa que la estimación de canal se encuentra afectada por el ruido AWGN.

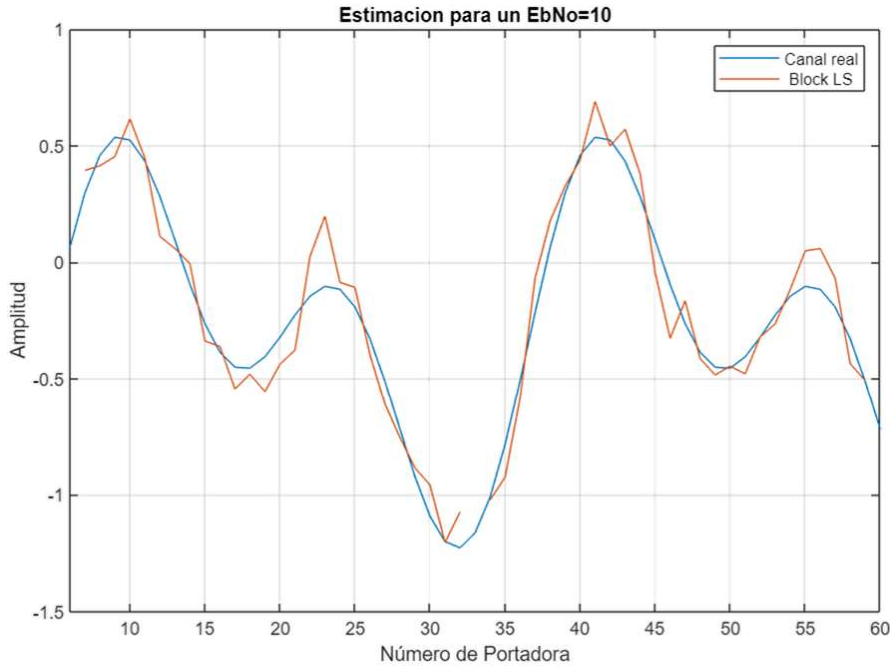

Figura 3.7 Canal real y estimado con estructura de pilotos *block type* y algoritmo LS

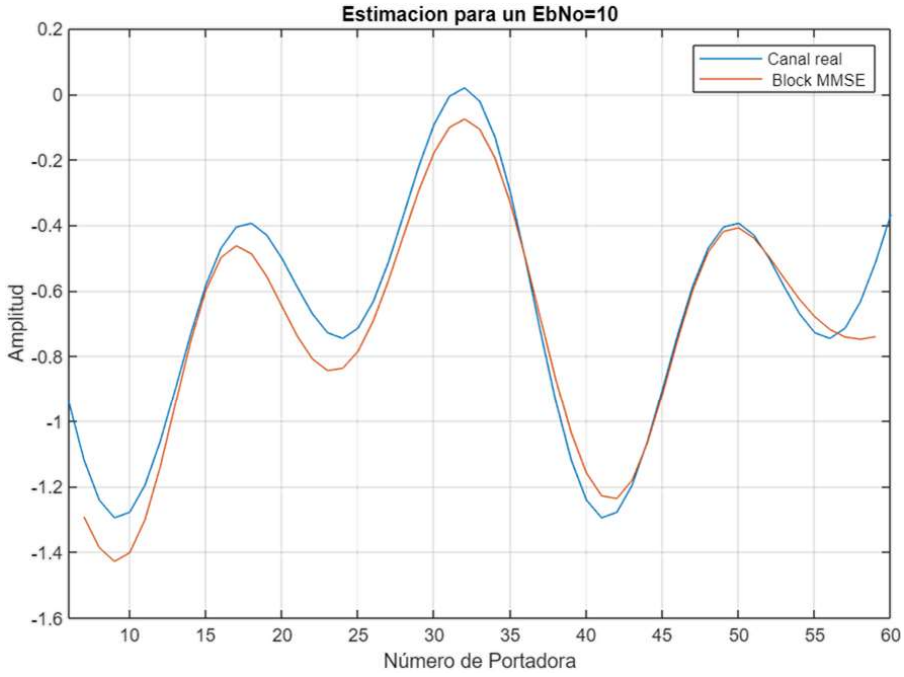

Figura 3.8 Canal real y estimado con estructura de pilotos *block type* y algoritmo MMSE

3.3 MSE vs Eb/No

Se presenta curvas del MSE vs Eb/No para las estructuras de pilotos: *comb type* cada 8 subportadoras (ver Figura 3.9), *comb type* cada 4 subportadoras (ver Figura 3.10) y *block type* (ver Figura 3.11).

En la Figura 3.9 se observa que al usar la estructura de pilotos *comb type* cada 8 subportadoras, la estimación de canal que presenta el menor MSE es cuando se usa el algoritmo MMSE (en violeta), seguido por el algoritmo LS con interpolación lineal (en rojo) y finalmente el que presenta un peor desempeño es el algoritmo LS usando interpolación *spline* (en amarillo).

En la Figura 3.10 se observa que el mejor desempeño se logra usando el algoritmo MMSE (en violeta), seguido por el algoritmo LS con interpolación *spline* (en amarillo), el cual funciona mejor que el algoritmo LS con interpolación lineal (en rojo) para relaciones Eb/No superiores a 10 dB.

En la Figura 3.11 al usar una estructura de pilotos *block type* el menor MSE se presenta usando el algoritmo MMSE (en amarillo), en cambio al usar el algoritmo LS (en rojo) el MSE entre el canal real y el estimado aumenta.

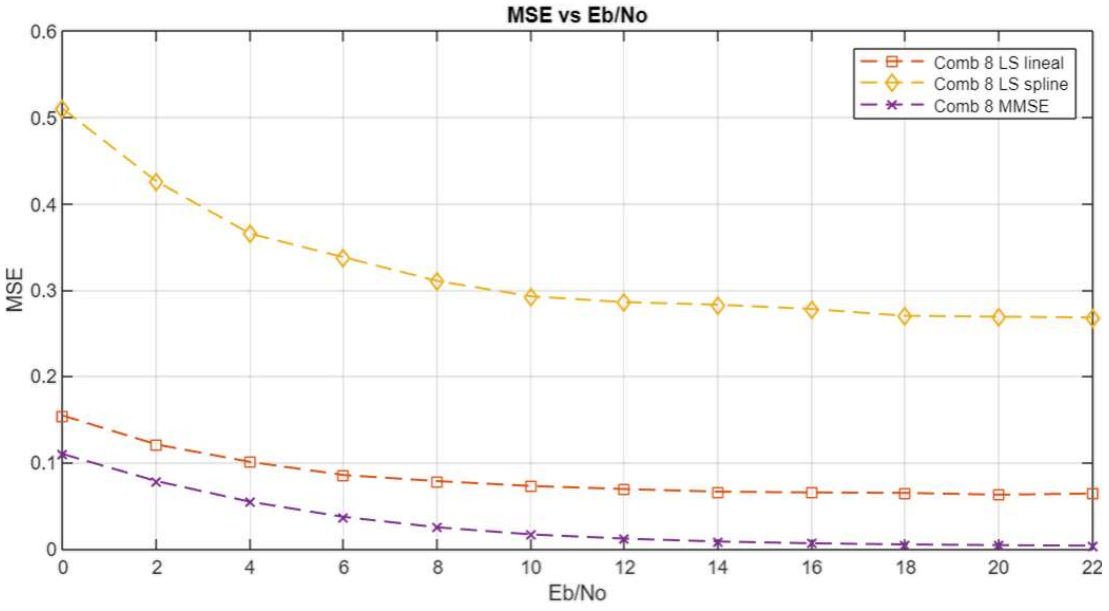

Figura 3.9 MSE vs. Eb/No, modulación 16QAM, distribución de pilotos *comb 8*

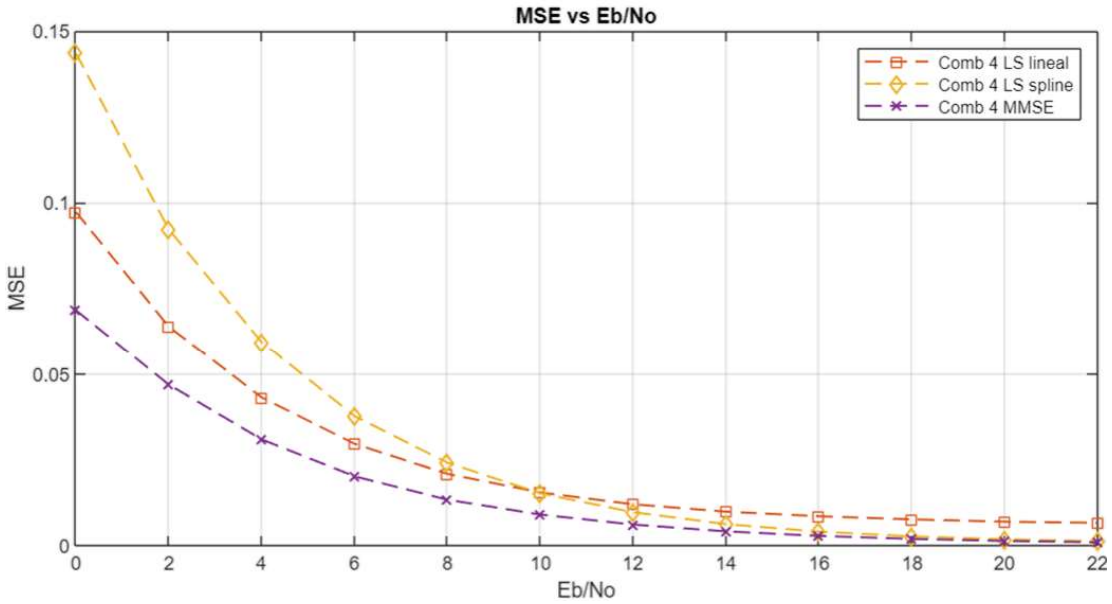

Figura 3.10 MSE vs. Eb/No, modulación 16QAM, distribución de pilotos *comb 4*

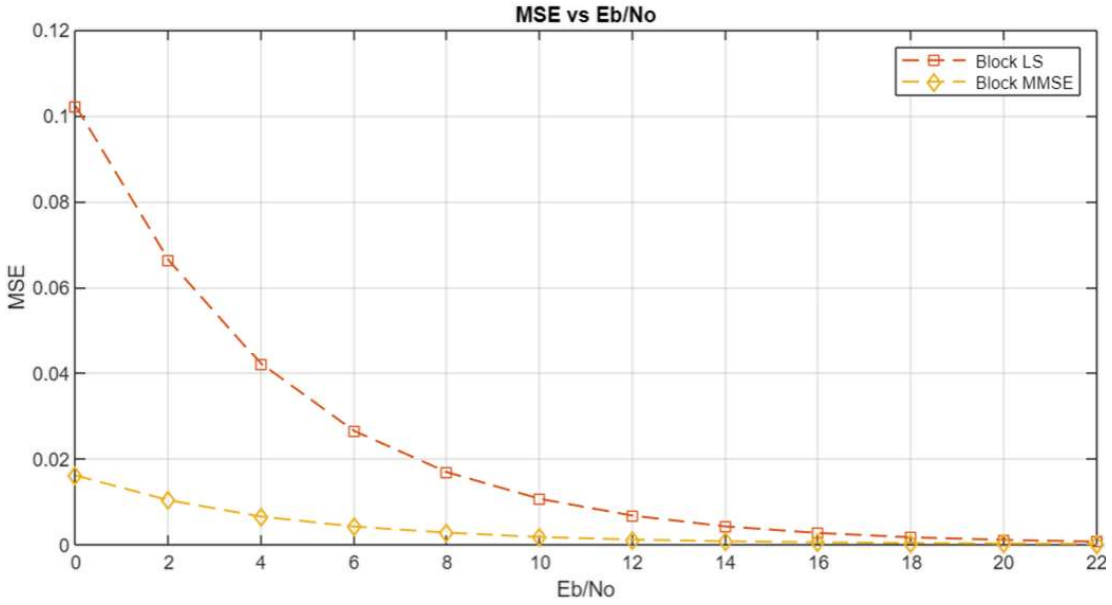

Figura 3.11 MSE vs. Eb/No, modulación 16QAM, distribución de pilotos *block*

3.4 BER vs. Eb/No

Se realizaron simulaciones para tres diferentes esquemas de modulación (QPSK, 16QAM y 64QAM). Para cada uno de los esquemas de modulación se realizó las curvas utilizando los esquemas de portadoras: *comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type*. En cada figura se compara para una misma distribución de pilotos los algoritmos de estimación de canal LS y MMSE.

Para todas las figuras que se presentarán se ha trazado en color azul con línea continua la estimación perfecta de canal, la cual se obtiene al usar los coeficientes del canal tipo Rayleigh para la ecualización (en este caso se tiene el supuesto de un canal inalámbrico conocido).

3.4.1 Modulación QPSK

3.4.1.1 Sin codificación tipo FEC ni entrelazado de bits

Se presentan curvas de la BER vs. Eb/No utilizando modulación QPSK para las estructuras de pilotos: *comb type* cada 8 subportadoras (ver Figura 3.12), *comb type* cada 4 subportadoras (ver Figura 3.13) y *block type* (ver Figura 3.14).

En cada una de las figuras antes mencionadas se observa que se tiene una mejor estimación de canal al usar el algoritmo MMSE con respecto al algoritmo LS, ya que se tiene una menor BER. Si se compara el mismo algoritmo de estimación de canal (LS o MMSE), usando diferentes estructuras de pilotos, se observa que en orden de mejor a peor BER se tiene: la estructura de pilotos *block type* (ver Figura 3.14), seguido por la estructura de pilotos *comb type* cada 4 subportadoras (ver Figura 3.13) y finalmente la estructura de pilotos *comb type* cada 8 subportadoras (ver Figura 3.12).

Como consecuencia de lo anteriormente mencionado se puede afirmar que la mejor BER se obtiene al usar estimación de canal con una estructura de pilotos *block type* en conjunto con el algoritmo MMSE, como se puede observar en la Figura 3.14.

Con respecto a los tipos de interpolación utilizados (lineal, *spline*). En la Figura 3.12, en la cual se usa una estructura de pilotos *comb type* cada 8 subportadoras se puede observar que se tiene una mejor BER usando la interpolación tipo lineal, en cambio en la Figura 3.13 en la cual se usa una estructura de pilotos *comb type* cada 4 subportadoras se observa que se tiene una mejor BER al usarse una interpolación tipo *spline*. La interpolación tipo *spline* funciona mejor que la lineal al usarse un mayor número de pilotos, ya que con un mayor número de datos esta interpolación logra ajustarse de mejor manera a las variaciones de amplitud del canal.

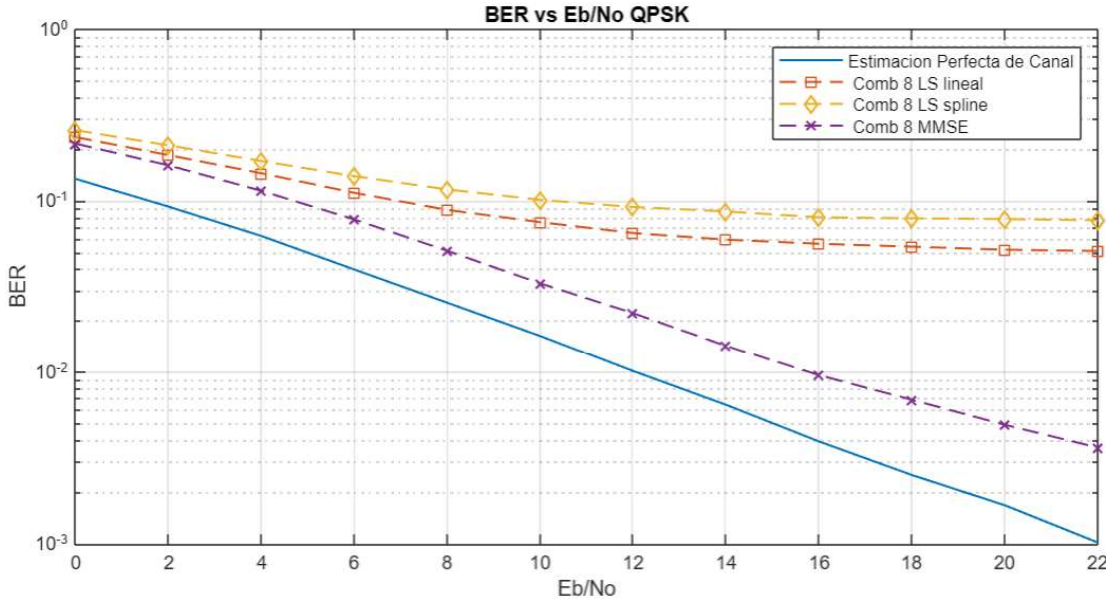

Figura 3.12 BER vs. Eb/No, modulación QPSK, distribución de pilotos *comb 8*

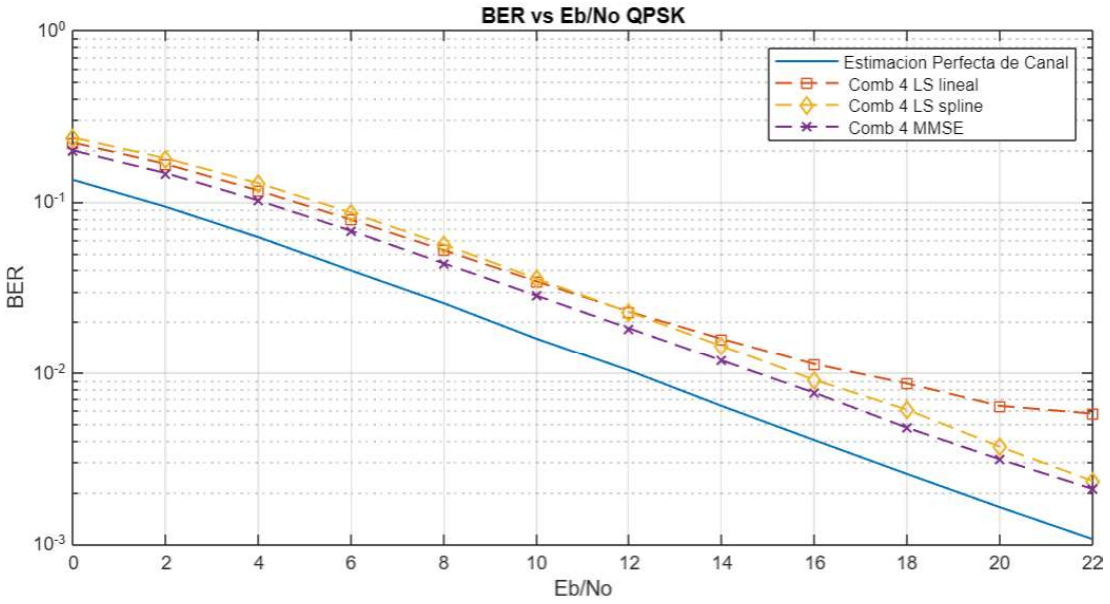

Figura 3.13 BER vs. Eb/No, modulación QPSK, distribución de pilotos *comb 4*

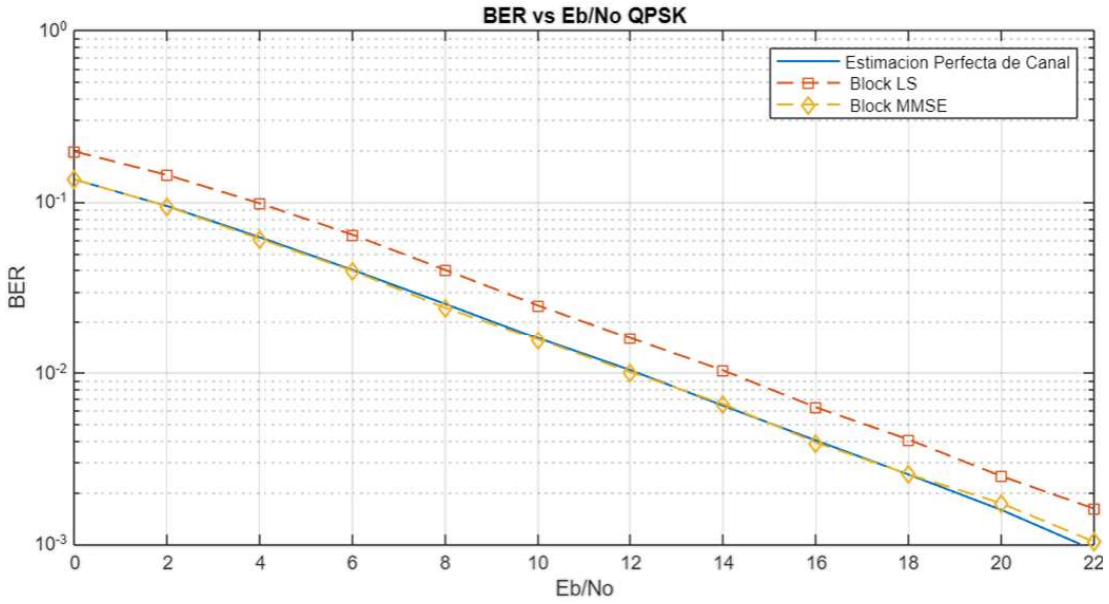

Figura 3.14 BER vs. Eb/No, modulación QPSK, distribución de pilotos *block type*

En la tabla 3.2 se detalla los valores de la Eb/No para una BER de 10^{-1}. Utilizando una distribución de pilotos *comb type* cada 8 subportadoras se observa una mejora en la Eb/No de: 3 dB entre el algoritmo LS *spline* y LS lineal, siendo este último el mejor; 5 dB entre el algoritmo LS *spline* y MMSE, siendo el último mencionado el mejor. Utilizando una distribución de pilotos *comb type* cada 4 subportadoras se observa una mejora en la Eb/No de: 1 dB entre el algoritmo LS spline y LS lineal, siendo este último el mejor; 1,5 dB entre el algoritmo LS *spline* y MMSE, siendo el último mencionado el mejor. Finalmente, utilizando una distribución de pilotos *block type* se observa una mejora en la Eb/No de 2 dB entre el algoritmo LS y MMSE, siendo este último el mejor.

Tabla 3.2 Valores de la relación Eb/No para una BER de 10^{-1}.

		Comb 8	*Comb* 4	*Block*
LS	Sin interpolación	-------	-------	4 dB
	Interpolación lineal	7 dB	4,5 dB	-------
	Interpolación s*pline*	10 dB	5,5 dB	-------
MMSE		5 dB	4 dB	2 dB

3.4.1.2 Con codificación tipo FEC y entrelazado de bits

En las siguientes figuras se observan las curvas de la BER vs. Eb/No para un esquema de modulación QPSK con codificación convolucional con tasa ½ y entrelazado de bits tipo bloque. En las curvas se usará las estructuras: *comb type* cada 8 subportadoras (ver Figura 3.15), *comb type* cada 4 subportadoras (ver Figura 3.16) y *block type* (ver Figura 3.17).

En la Figura 3.15 se puede observar que la BER con estimación perfecta de canal es mejor comparado con la Figura 3.12 donde no se utiliza codificación FEC. Además, se puede notar en la Figura 3.15 que la BER con la estimación tipo LS es malo a pesar de la utilización de codificación FEC. Este resultado se obtiene porque la separación de los pilotos cada 8 subportadoras es excesiva, lo cual genera errores que no pueden ser corregidos con la codificación FEC. En el caso de la estimación usando el algoritmo MMSE, se observa que la BER si mejora.

Con respecto a la Figura 3.16 y la Figura 3.17 si se las compara con la Figura 3.13 y la Figura 3.14 respectivamente, se puede notar una mejora sustancial en la BER con el uso de la codificación. En la Figura 3.16 se puede observar que con pilotos cada 4 subportadoras, el algoritmo LS funciona y se obtiene una BER cercano al algoritmo MMSE. Finalmente, en la Figura 3.17 se puede observar que usando el algoritmo MMSE con una estructura de pilotos *block type*, se obtiene una BER muy similar al obtenido considerando la estimación perfecta del canal.

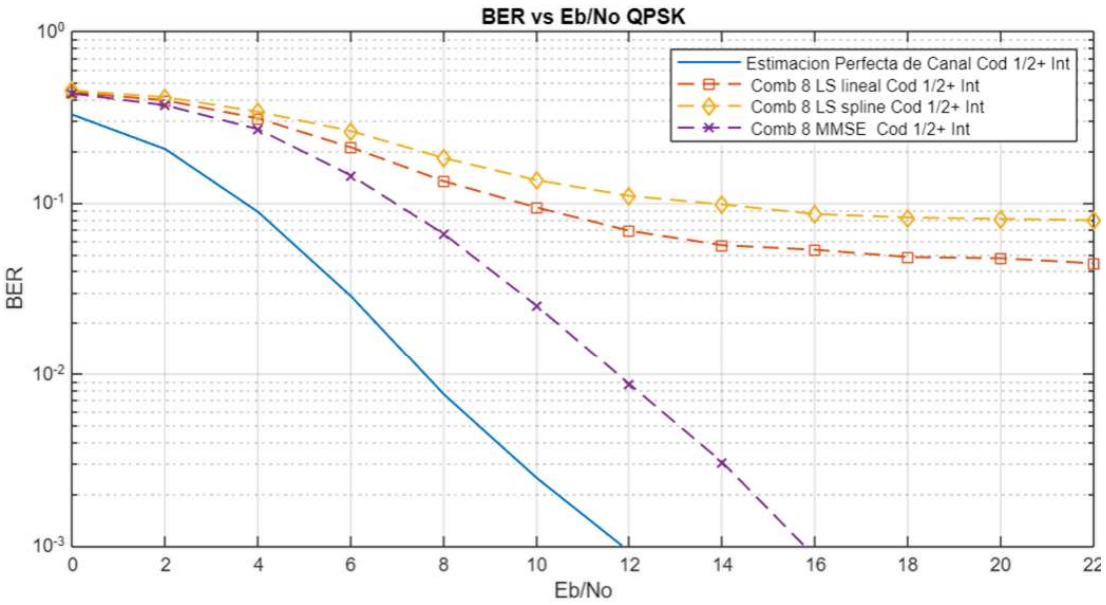

Figura 3.15 BER vs. Eb/No, modulación QPSK, distribución de pilotos *comb 8* (codificación tasa ½ e interleaving)

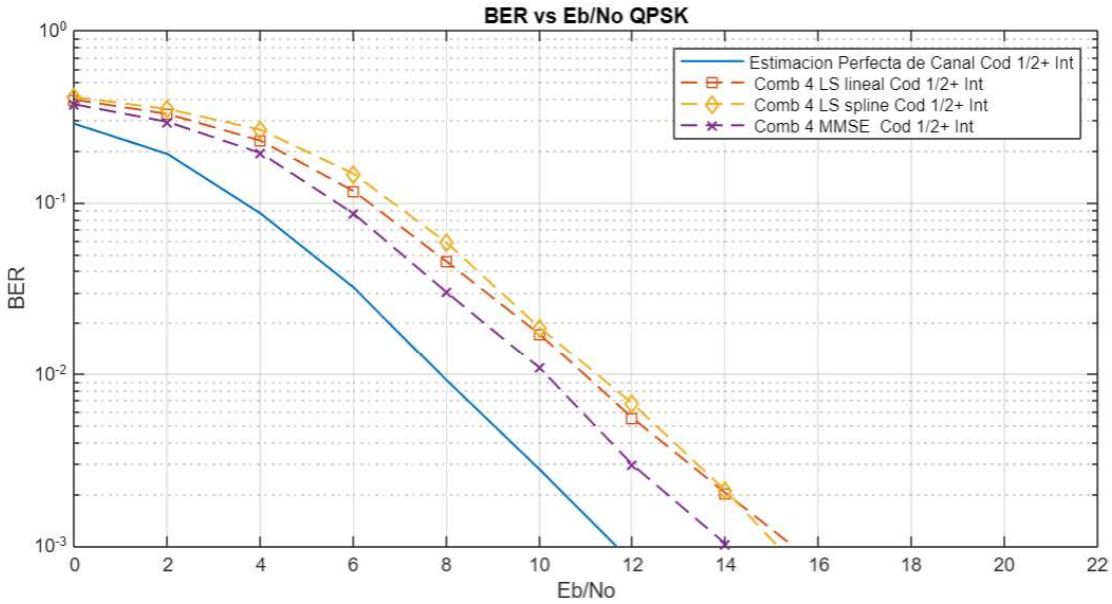

Figura 3.16 BER vs. Eb/No, modulación QPSK, distribución de pilotos *comb 4* (codificación tasa ½ e interleaving)

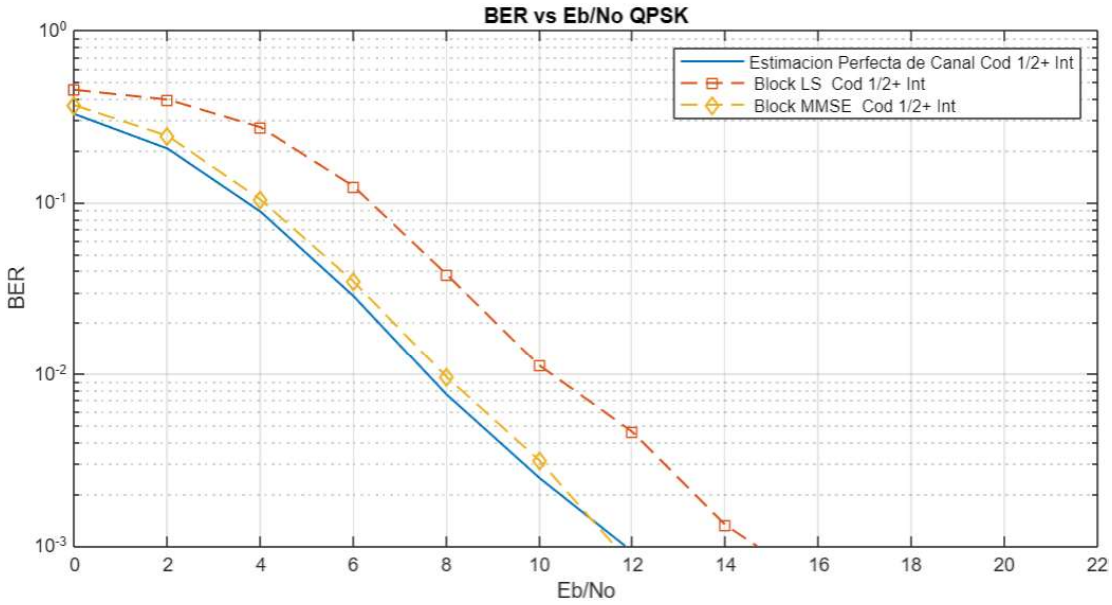

Figura 3.17 BER vs. Eb/No, modulación QPSK, distribución de pilotos *block* (codificación tasa ½ e interleaving)

En la tabla 3.3 se detalla los valores de la Eb/No para una BER de 10^{-1}. Utilizando una distribución de pilotos *comb type* cada 8 subportadoras se observa una mejora en la Eb/No de: 4 dB entre el algoritmo LS *spline* y LS lineal, siendo este último el mejor; 7 dB entre el algoritmo LS *spline* y MMSE, siendo el último mencionado el mejor. Utilizando una distribución de pilotos *comb type* cada 4 subportadoras se observa una mejora en la Eb/No de: 0,5 dB entre el algoritmo LS spline y LS lineal, siendo este último el mejor; 1,5 dB entre el algoritmo LS *spline* y MMSE, siendo el último mencionado el mejor. Finalmente, utilizando una distribución de pilotos *block type* se observa una mejora en la Eb/No de 2,5 dB entre el algoritmo LS y MMSE, siendo este último el mejor.

Tabla 3.3 Valores de la relación Eb/No para una BER de 10^{-1}.

		Comb 8	*Comb* 4	*Block*
LS	Sin interpolación	-------	-------	6,5 dB
	Interpolación lineal	10 dB	6,5 dB	-------
	Interpolación s*pline*	14 dB	7 dB	-------
MMSE		7 dB	5,5 dB	4 dB

3.4.2 Modulación 16QAM

3.4.2.1 Sin codificación tipo FEC ni entrelazado de bits

En las siguientes figuras se presenta las curvas de la BER vs. Eb/No utilizando un esquema de modulación 16QAM para las estructuras de pilotos: *comb type* cada 8 subportadoras (ver Figura 3.18), *comb type* cada 4 subportadoras (ver Figura 3.19) y *block type* (ver Figura 3.20).

En esas figuras se observa que usando el algoritmo MMSE se obtiene una mejor BER que al usar el algoritmo tipo LS. Si en las mismas curvas se compara las diferentes estructuras de pilotos para un mismo algoritmo de estimación de canal, se observa que la mejor BER de canal ocurre cuando se usa la estructura de pilotos *block type*, seguido por la estructura de pilotos *comb type* cada 4 subportadoras y finalmente la estructura de pilotos *comb type* cada 8 subportadoras.

Con respecto a los 2 métodos de interpolación se observa un mejor desempeño de la interpolación lineal en la Figura 3.18, en cambio en la Figura 3.19 se observa un mejor desempeño de la interpolación tipo *spline*, de hecho en esa figura se consigue con el algoritmo tipo LS resultados próximos a los del algoritmo MMSE.

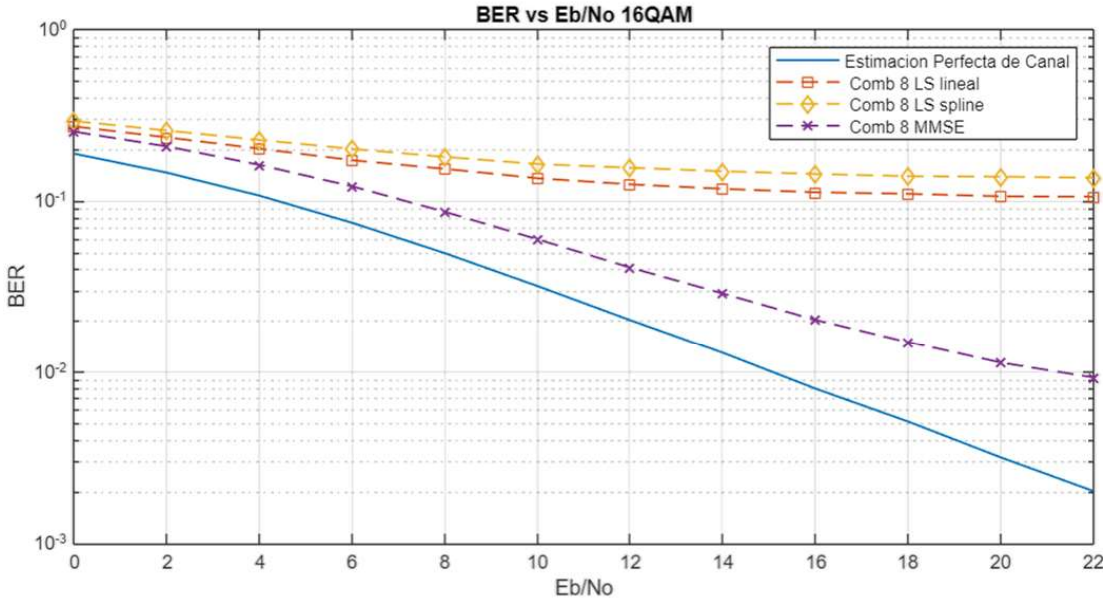

Figura 3.18 BER vs. Eb/No, modulación 16QAM, distribución de pilotos *comb 8*

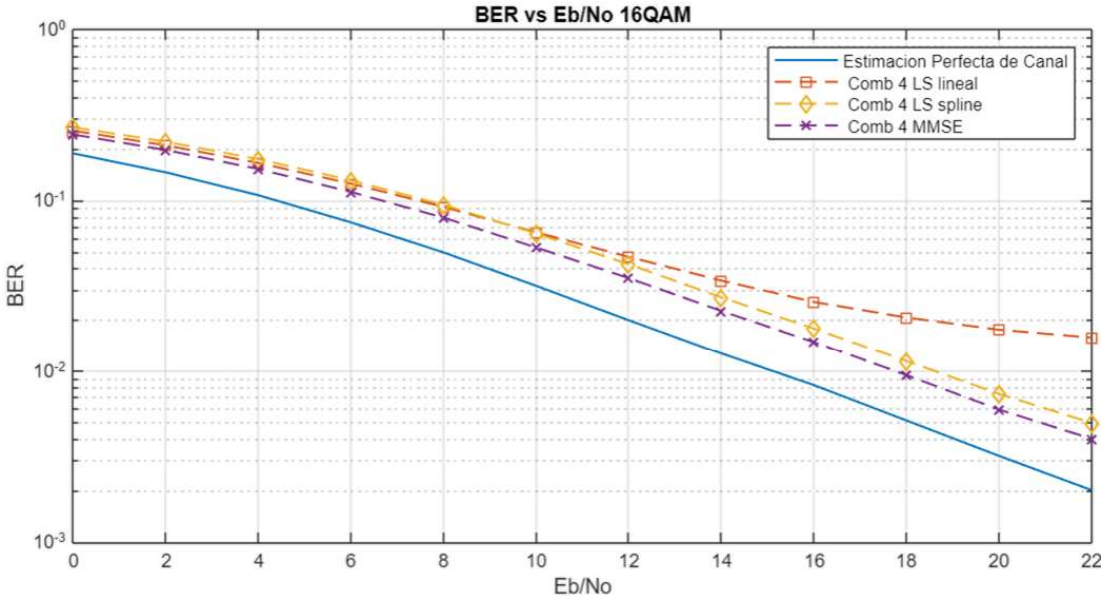

Figura 3.19 BER vs. Eb/No, modulación 16QAM, distribución de pilotos *comb 4*

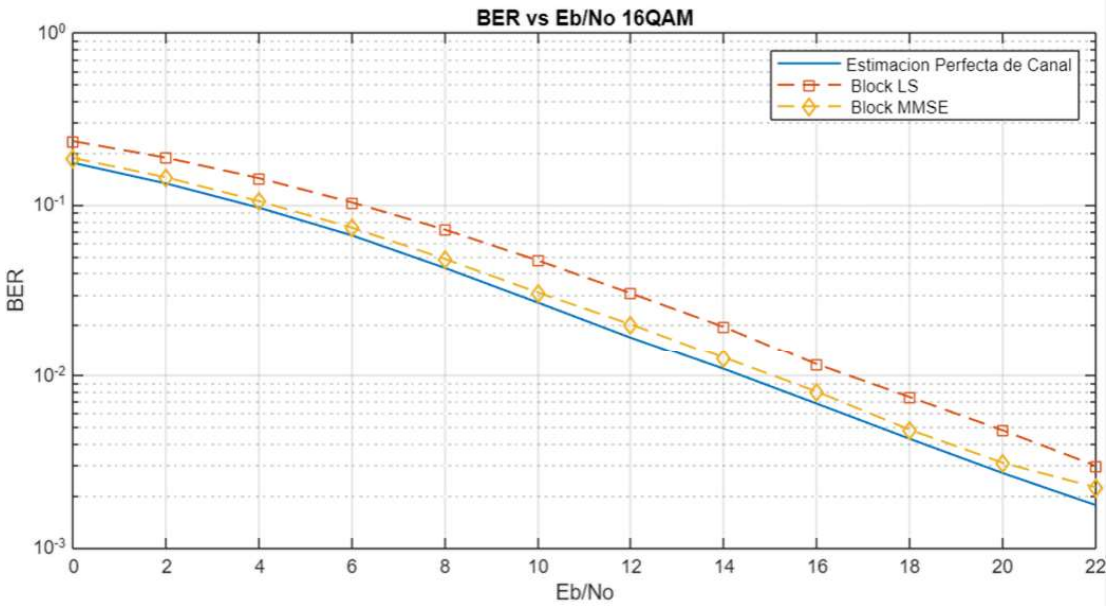

Figura 3.20 BER vs. Eb/No, modulación 16QAM, distribución de pilotos *block*

3.4.2.2 Con codificación tipo FEC y entrelazado de bits

En las siguientes figuras se puede observar las curvas de la BER vs. Eb/No para las estructuras de pilotos presentadas anteriormente (*comb type* cada 8 subportadoras, *comb type* cada 4 subportadoras y *block type*), pero usando codificación convolucional con tasa ½ y entrelazado de bits.

En la Figura 3.21 se observa que la BER con estimación perfecta de canal es mejor que en la Figura 3.18 donde no se usa codificación FEC. Además, se observa en la Figura 3.21 que la BER usando el algoritmo de estimación LS es malo a pesar de la utilización de la codificación FEC. Esto debido a que un piloto cada 8 subportadoras es una separación excesiva, lo cual genera errores que no pueden ser corregidos con la codificación FEC. En el caso de la estimación usando el algoritmo MMSE, se observa que mejora en el BER.

Con respecto a la Figura 3.22 y la Figura 3.23 si se las compara con la Figura 3.19 y la Figura 3.20 respectivamente, se observa una mejora sustancial en la BER como resultado del uso de la codificación. En la Figura 3.22 se observa que con pilotos cada 4 subportadoras, el algoritmo LS funciona y se obtiene una BER cercano al algoritmo MMSE. Finalmente, en la Figura 3.23 se puede observar que usando el algoritmo MMSE y una estructura de pilotos *block type*, se obtiene una BER muy similar al obtenido considerando la estimación perfecta del canal.

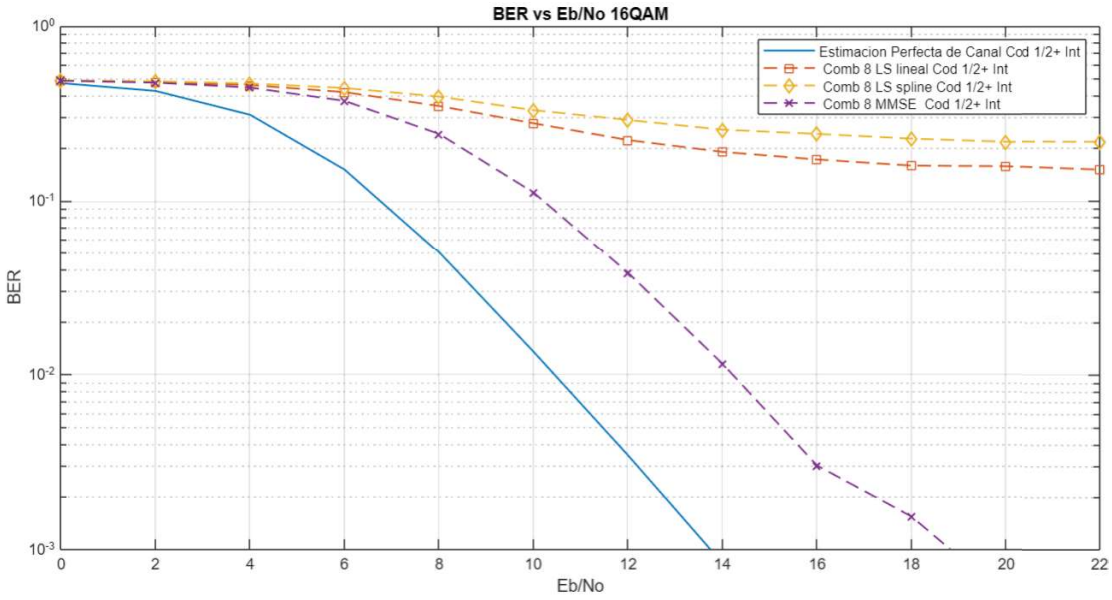

Figura 3.21 BER vs. Eb/No, modulación 16QAM, distribución de pilotos *comb 8* (codificación tasa ½ e interleaving)

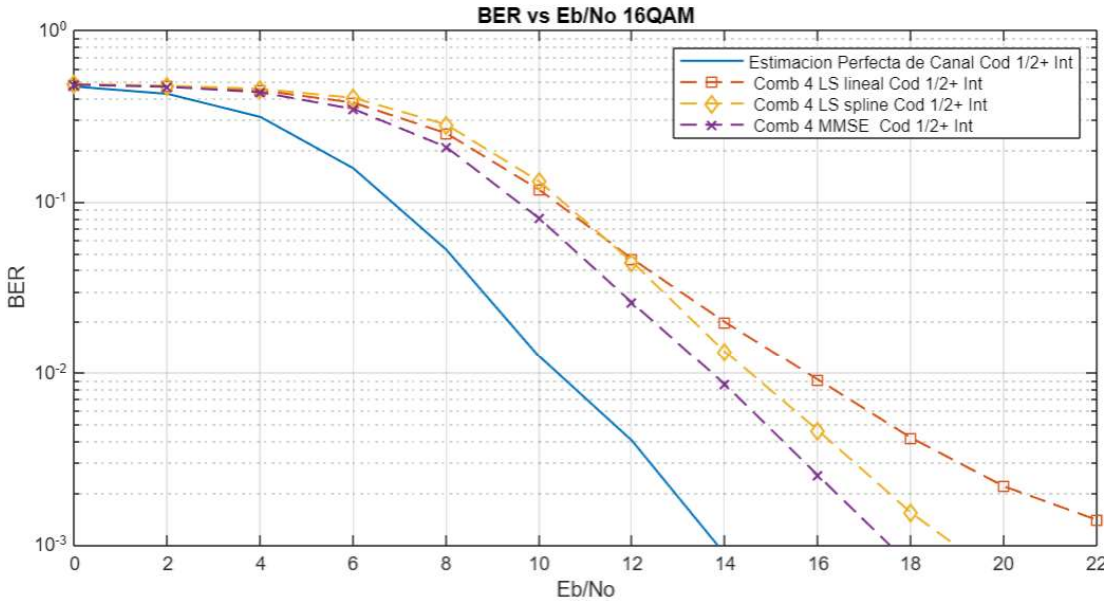

Figura 3.22 BER vs. Eb/No, modulación 16QAM, distribución de pilotos comb 4 (codificación tasa ½ e interleaving)

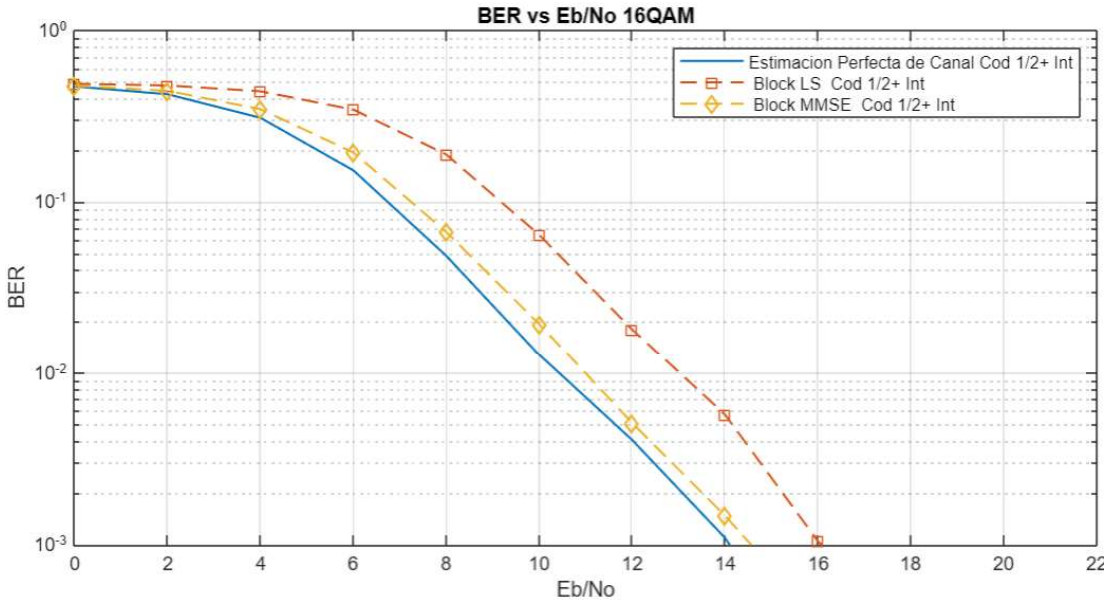

Figura 3.23 BER vs. Eb/No, modulación 16QAM, distribución de pilotos block (codificación tasa ½ e interleaving)

3.4.3 Modulación 64QAM

3.4.3.1 Sin codificación tipo FEC ni entrelazado de bits

En las siguientes figuras se presenta las curvas de la BER vs. Eb/No utilizando un esquema de modulación 64QAM para las distribuciones de pilotos: *block type* con un espaciado entre pilotos de 8 subportadoras (ver Figura 3.24), *block type* con un espaciado entre pilotos de 4 subportadoras (ver Figura 3.25) y *comb type* (ver Figura 3.26).

En las figuras antes mencionadas se observa una menor tasa de bits errados (BER) cuando se usa el algoritmo MMSE que cuando se usa el algoritmo LS. Si se compara el mismo algoritmo de estimación de canal (LS o MMSE), usando diferentes estructuras de pilotos, se observa que en orden de mejor a peor BER se tiene: la estructura de pilotos *block type*, seguido por la estructura de pilotos *comb type* cada 4 subportadoras y finalmente la estructura de pilotos *comb type* cada 8 subportadoras. Como consecuencia de lo anteriormente mencionado se puede afirmar que la mejor BER de canal ocurre al usar una estructura de pilotos *block type* y usando el algoritmo tipo MMSE para la estimación de canal, como se puede observar en la Figura 3.26.

Con respecto a los tipos de interpolación utilizados (lineal, *spline*). En la Figura 3.24, en la cual se usa una estructura de pilotos *comb type* cada 8 subportadoras se observa que se tiene una mejor BER usando la interpolación tipo lineal, en cambio en la Figura 3.13 en la

cual se usa una estructura de pilotos *comb type* cada 4 subportadoras se observa que se tiene una mejor estimación al usarse una interpolación tipo *spline*.

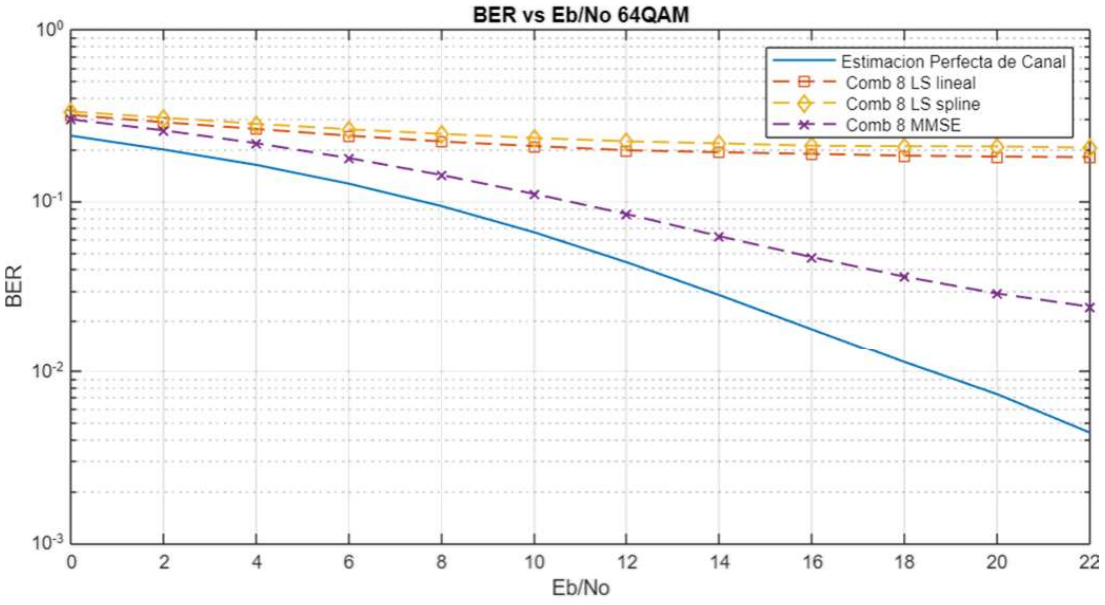

Figura 3.24 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *comb 8*

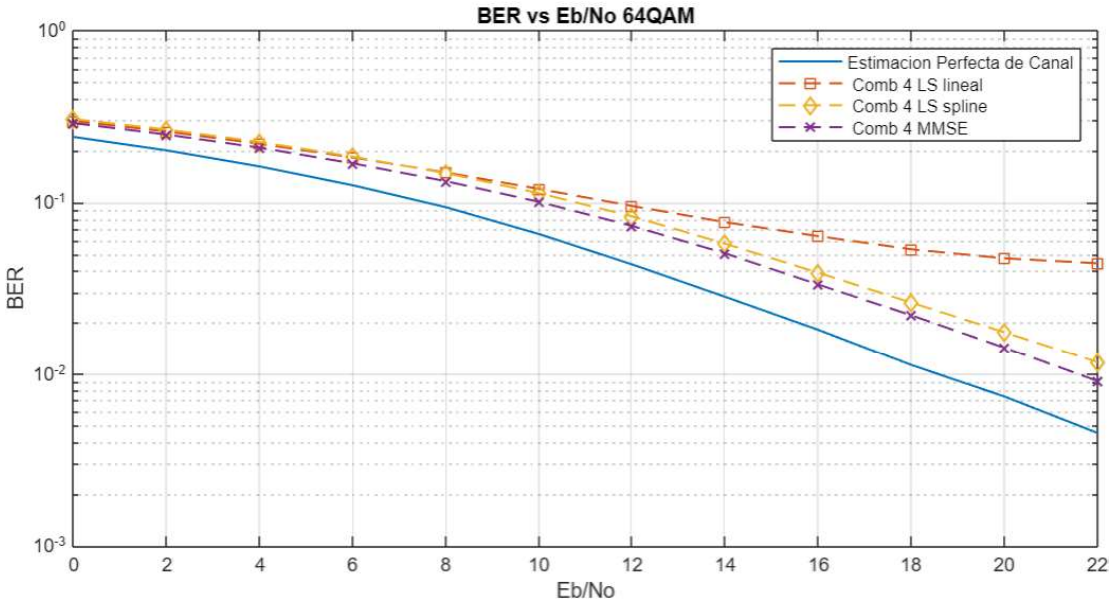

Figura 3.25 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *comb 4*

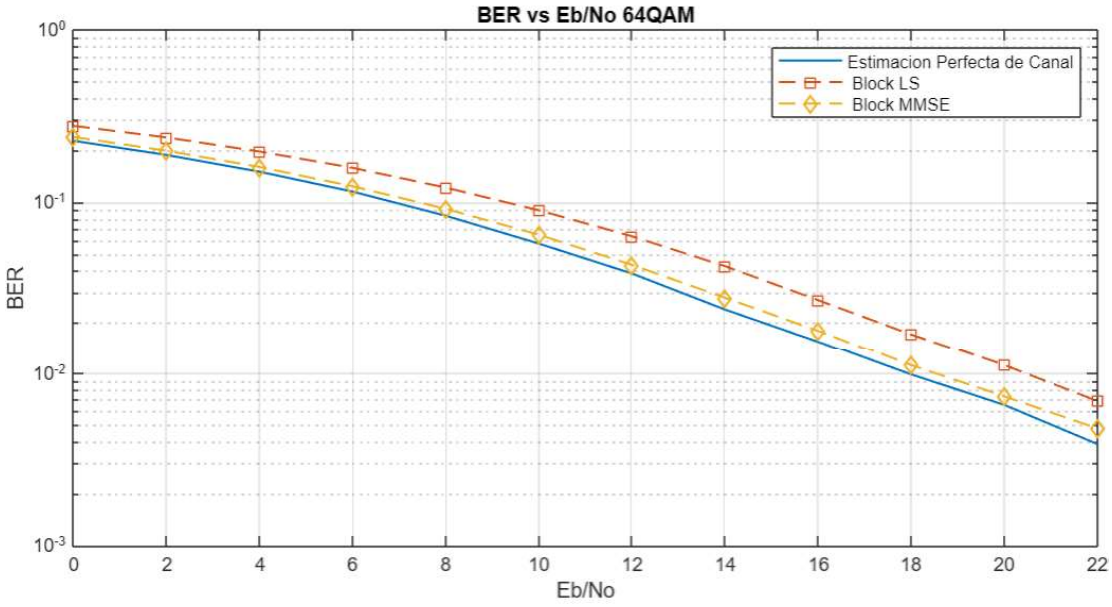

Figura 3.26 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *block*

3.4.3.2 Con codificación tipo FEC y entrelazado de bits

En las siguientes figuras se puede observar las curvas de la BER vs. Eb/No para las estructuras de pilotos: *comb type* cada 8 subportadoras (ver Figura 3.27), *comb type* cada 4 subportadoras (ver Figura 3.28) y *block type* (Figura 3.29) pero usando codificación convolucional con tasa ½ y entrelazado de bits.

En la Figura 3.27 se puede observar que la BER con estimación perfecta de canal es mejor comparado con la Figura 3.24 donde no se utiliza codificación FEC. Además, se puede notar en la Figura 3.27 que la BER con la estimación tipo LS es malo a pesar de la utilización de codificación FEC. Este resultado se obtine porque la separación de los pilotos cada 8 subportadoras es excesiva, lo cual genera errores que no pueden ser corregidos con la codificación FEC. En el caso de la estimación usando el algoritmo MMSE, se observa que la BER si mejora.

Con respecto a la Figura 3.28 y la Figura 3.29 si se las compara con Figura 3.25 y la Figura 3.26 respectivamente, se puede notar una mejora sustancial en la BER con el uso de la codificación. En la Figura 3.28 se puede observar que con pilotos cada 4 subportadoras, el algoritmo LS funciona y se obtiene una BER cercana al algoritmo MMSE. Finalmente, en la Figura 3.29 se puede observar que usando el algoritmo MMSE con una estructura de pilotos *block type*, se obtiene una BER muy similar al obtenido considerando la estimación perfecta del canal.

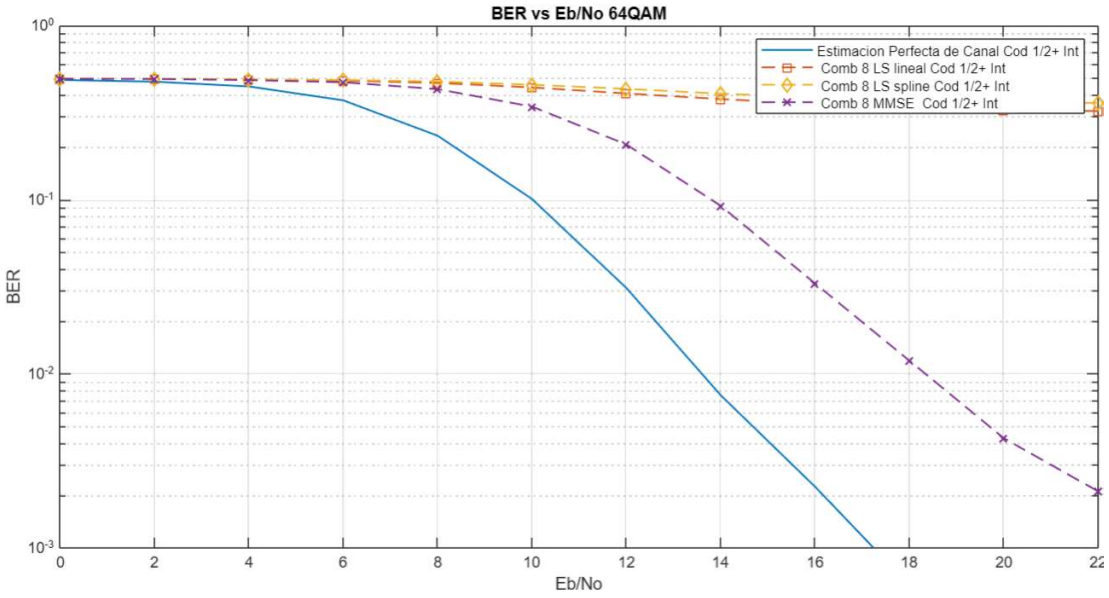

Figura 3.27 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *comb 8* (codificación tasa ½ e interleaving)

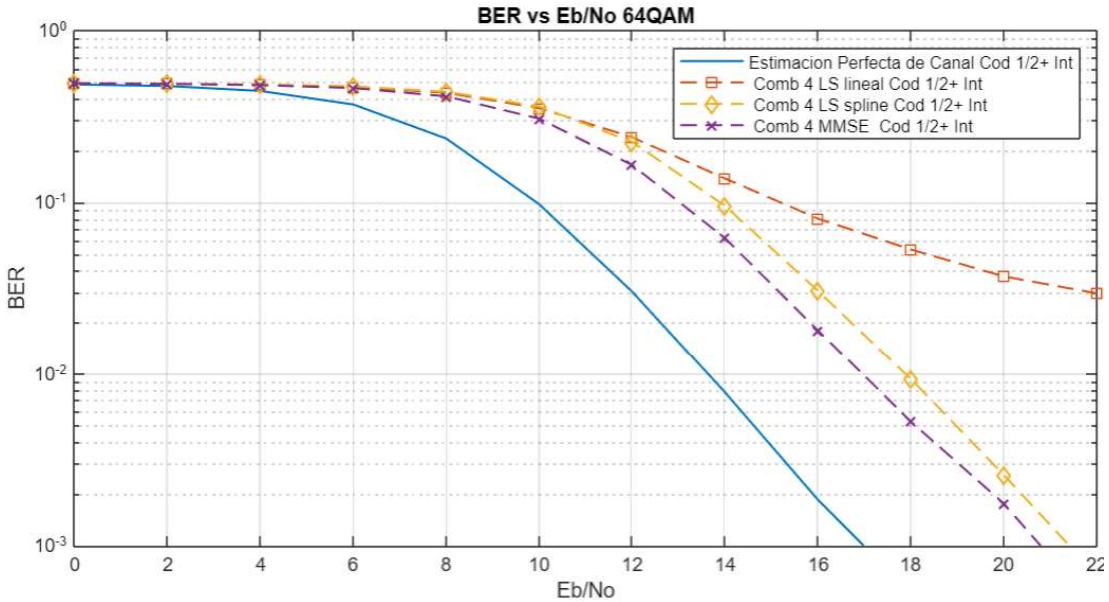

Figura 3.28 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *comb 4* (codificación tasa ½ e interleaving)

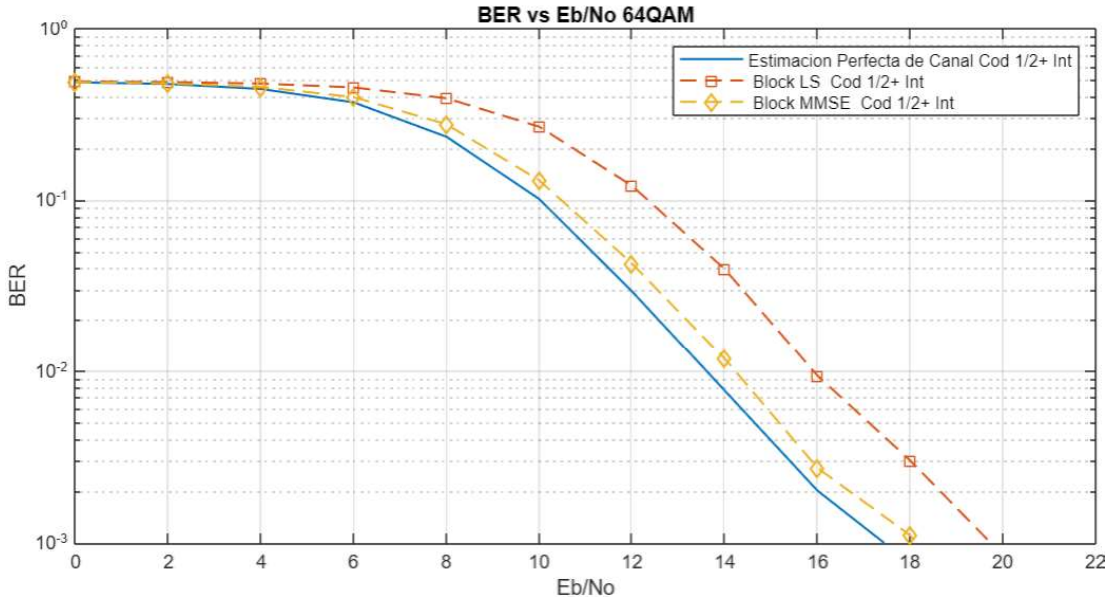

Figura 3.29 BER vs. Eb/No, modulación 64QAM, distribución de pilotos *block* (codificación tasa ½ e interleaving)

3.5 Resultados con un canal no exponencial

Con el fin de enfatizar que el canal ingresado en la aplicación debe ser de tipo exponencial, se mostrará las curvas: MSE vs. Eb/No y BER vs. Eb/No para una estructura de pilotos *comb type* con espaciado entre pilotos de 4 subportadoras. El esquema de modulación usada en los ejemplos será 16QAM y el canal usado para estos ejemplos es el canal especificado en la Tabla 3.4, el cual corresponde al canal presentado en la Tabla 1.2 con ciertas modificaciones en los retrasos con el fin de tener retrasos normalizados.

Tabla 3.4 PDP del canal a utilizarse

Trayecto	Retraso Relativo(ns)	Potencia Promedio (dB)
1	0	0.0
2	100	-9.7
3	200	-19.2
4	400	-22.8

En la Figura 3.30 se observa que el MSE es menor cuando se usa el algoritmo LS. Este resultado es completamente diferente a los resultados usando un canal exponencial, en los cuales al usar el algoritmo MMSE se obtuvo menor MMSE que usando el algoritmo LS. Se tiene este resultado ya que el algoritmo MMSE está diseñado para un canal exponencial. En consecuencia, cuando el canal es diferente, el MSE aumenta.

En la Figura 3.31 se observa una mayor tasa de bits errados al usar el algoritmo MMSE, lo cual contradice ejemplos anteriores en los cuales usando este método se obtuvo menor BER que usando el algoritmo LS. Este resultado es similar al observado en la Figura 3.30 del MSE. Entonces, es importante recalcar que el algoritmo MMSE es mejor que el algoritmo LS sólo cuando el PDP del canal inalámbrico utilizado sea de tipo exponencial.

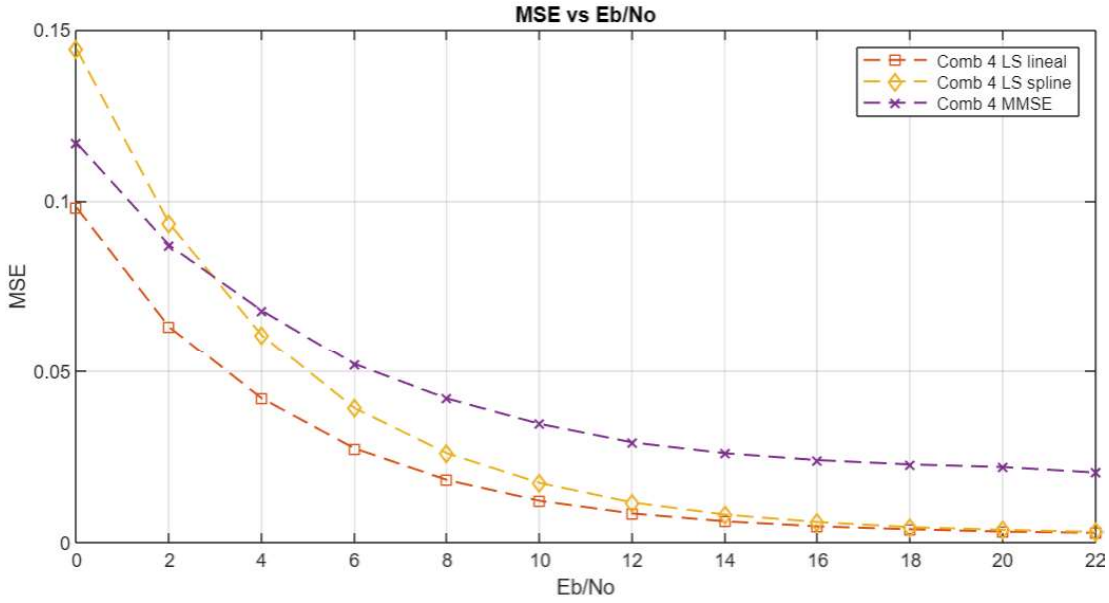

Figura 3.30 MSE vs. Eb/No, modulación 16QAM, distribución de pilotos comb 4

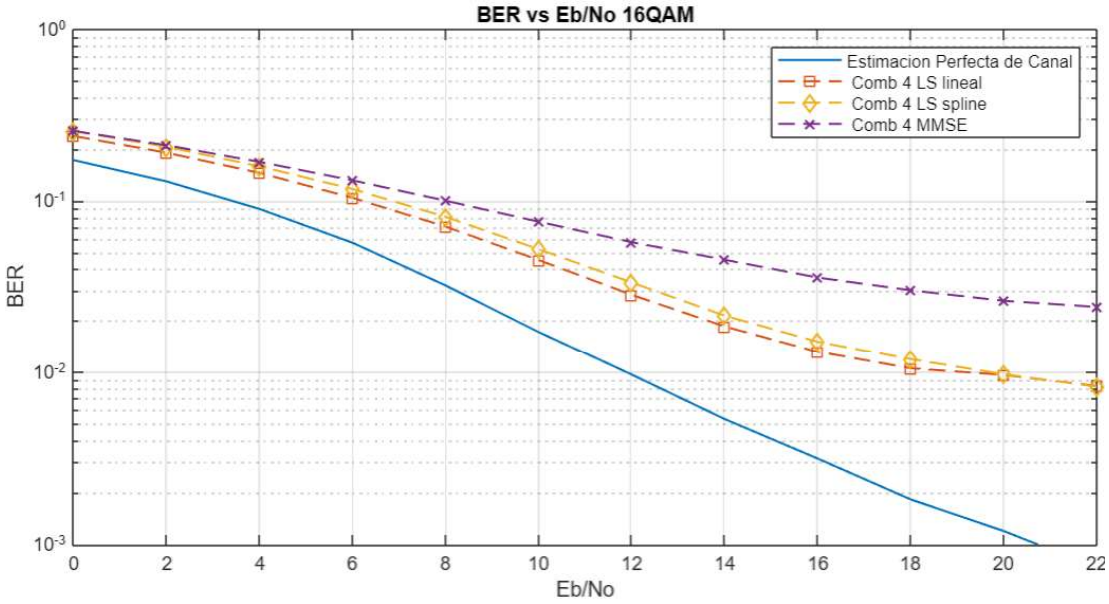

Figura 3.31 BER vs. Eb/No, modulación 16QAM, distribución de pilotos comb 4

4. CONCLUSIONES

En este capítulo se presenta las conclusiones, recomendaciones y posibles trabajos futuros relacionados con el presente proyecto técnico.

4.1 Conclusiones

- Sistemas de comunicaciones actuales como son IEEE 802.11n, IEEE 802.11 ac y utilizan como técnica de transmisión a OFDM ya que presenta múltiples ventajas como son: alta eficiencia espectral, altas velocidades de transmisión, robustez en entornos multitrayecto y un proceso de ecualización en recepción que es fácilmente implementado usando un ecualizador de una sola rama por cada subcanal.

- La ecualización en sistemas OFDM es necesaria para compensar los cambios ocurridos en la señal debido a su paso por el canal. Antes de realizar este proceso es necesario previamente realizar el proceso de estimación de canal, ya que mediante este se conoce el estado del canal.

- La estimación de canal en sistemas OFDM se puede realizar usando estructuras de pilotos como: *block type*, *comb type*, y *lattice type*. El tipo de estructura de pilotos que se usará depende del tipo de desvanecimiento presente en el canal de comunicaciones.

- La estructura de pilotos *block type* es útil cuando el canal presenta un desvanecimiento en frecuencia o para canales con desvanecimiento lento en tiempo, en el cual el canal no varía demasiado en tiempo entre dos muestras consecutivas, cuando el canal presenta variaciones en tiempo grandes esta estructura no es la adecuada, ya que no se tiene pilotos en todos los símbolos OFDM.

- La estructura de pilotos *comb type* es efectiva cuando el canal presenta un desvanecimiento plano en frecuencia o para canales con desvanecimiento rápido en tiempo, ya que las muestras se encuentran separadas en frecuencia, pero en tiempo se tiene pilotos para cada símbolo OFDM, cuando se tiene desvanecimiento selectivo en frecuencia esta estructura no es la más adecuada.

- La mejor estructura de pilotos para el sistema OFDM implementado es *block type* seguida por *comb type* cada 4 subportadoras y finalmente la peor es *comb type* cada 8 subportadoras. A pesar de que la estructura de pilotos *block type* presenta la mejor estimación, se debe considerar que el canal implementado no presenta un desvanecimiento en tiempo, por lo cual para otros canales con este tipo de desvanecimiento la estructura antes mencionada no podría ser tan efectiva.

- Al enviar una gran cantidad de pilotos se logra una mejor estimación de canal, con lo cual se mejoran el MSE y la BER pero al mismo tiempo eso implica que se tiene un menor número de portadoras para datos, con lo cual la eficiencia espectral de la comunicación disminuye.
- El algoritmo MMSE funciona mejor que el LS para los casos en los que el canal ingresado corresponde con las características para las cuales se diseñó el estimador MMSE en el caso de este proyecto técnico, es un canal exponencial.
- La mejora en la BER que se presenta al usar un algoritmo más complejo como es el MMSE respecto a usar un algoritmo más simple como el LS depende del número de portadoras usadas para la estimación de canal. Siendo la mejora más grande al usar pocas portadoras y pequeña al usar un número de portadoras mayor como se puede observar al comparar los resultados de la BER vs. Eb/No para una estructura de pilotos *comb type* donde se incrementa el número de portadoras desde 1 de cada 8 a 1 de cada 4.
- Para el canal simulado la mejor estimación de canal se consigue con la combinación entre el algoritmo de estimación de canal tipo MMSE y la estructura de pilotos *block type*.
- Contrario a lo que se espera, la codificación convolucional tipo FEC no mejora la BER cuando se usa el algoritmo LS y una distribución de pilotos *comb type* cada 8 subportadoras, esto se debe a que se presentan demasiados errores como para que puedan ser corregidos.
- La interpolación *spline* funciona mejor que la interpolación lineal solo cuando se tiene el suficiente número de datos para interpolar como en el caso de la estructura de pilotos *comb type* cada 4 subportadoras, en cambio cuando se tiene un menor número de datos, esta interpolación no logra seguir las variaciones de amplitud del canal como cuando se usa una estructura *comb type* cada 8 subportadoras en la cual funciona mejor la interpolación lineal.

4.2 Recomendaciones

- El tiempo de simulación en Matlab depende de los recursos computacionales utilizados, por lo cual se recomienda el uso de una computadora con buenas características.

- Para utilizar la aplicación implementada en *App Designer* se recomienda el uso de la versión de Matlab 2018 B.
- Para obtener las curvas de la BER vs. Eb/No y del MSE vs. Eb/No se recomienda el uso de un número grande de simulaciones (mayor a 2000 simulaciones) para que los gráficos presenten trazos definidos. En caso de usar codificación convolucional e interleaving se recomienda usar un número de simulaciones más grande que el anterior (mayor a 5000 simulaciones).
- En caso de usarse el algoritmo de estimación de canal MMSE se recomienda ingresar un canal exponencial en la interfaz gráfica, ya que las ecuaciones utilizadas para el cálculo de la autocorrelación y la correlación cruzada son para este tipo de canal.
- Es recomendable el ingreso en la interfaz gráfica de un PDP en el cual los retardos no superen la longitud del prefijo cíclico, esto se realiza con el fin de no introducir ISI en el sistema.
- Para el sistema OFDM implementado se recomienda el ingreso de un *power delay profile* en el cual los retardos se encuentren normalizados, es decir que sean múltiplos de 50 ns.

4.3 Trabajos futuros

- Añadir la estructura de pilotos *lattice type* y compararla con las estructuras *comb type* y *block type* implementadas en el presente proyecto técnico.
- Implementar computación paralela para acelerar la simulación del sistema de comunicaciones OFDM.
- En el presente proyecto técnico se utilizó un canal de 20 MHz con 64 subportadoras, se podría usar un canal de 40 MHz en el cual se tiene 128 subportadoras.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO-OFDM Wireless Communications with MATLAB*. Wiley editorial, 2010.

[2] T. D. Chiueh and P. Y. Tsai, *OFDM Baseband Receiver Design for Wireless Communications*. 2009.

[3] P. M. C. Nathan, *Wireless communications*. Prentice Hall of India, 2008.

[4] B. Sklar, *Digital communications : fundamentals and applications*. Prentice-Hall PTR, 2001.

[5] V. K. Garg, *Wireless communications and networking*. Elsevier Morgan Kaufmann, 2007.

[6] H. T. Friis, "A Note on a Simple Transmission Formula," *Proc. IRE*, vol. 34, no. 5, pp. 254–256, May 1946.

[7] T. S. Rappaport, *Wireless communications : principles and practice*. Dorling Kindersley, 2009.

[8] W. C. Y. Lee, *Mobile communications engineering*. McGraw-Hill, 1982.

[9] "GUIDELINES FOR EVALUATION OF RADIO TRANSMISSION TECHNOLOGIES FOR IMT-2000 (Question ITU-R 39/8) (1997)."

[10] Wonjong Rhee, J. C. Chuang, and L. J. Cimini, "Performance comparison of OFDM and multitone with polyphase filter bank for wireless communications," in *VTC '98. 48th IEEE Vehicular Technology Conference. Pathway to Global Wireless Revolution (Cat. No.98CH36151)*, vol. 2, pp. 768–772.

[11] G. Liu and G. Li, *OFDM-Based Broadband Wireless Networks*. Wiley- Interscience, 2005.

[12] M. M. da Silva, *Cable and wireless networks : theory and practice*. 2016.

[13] J. Hauser, *Numerical Methods for Nonlinear Engineering Models*. Springer, 2009.

[14] Mora Rafael, *Teoria De La Interpolacion*. Instituto Polit{ecnico Universitario de Mariño.

[15] G. Farin, "Cubic Spline Interpolation," in *Curves and Surfaces for Computer-Aided*

Geometric Design, 1993, pp. 133–155.

[16] C. De Boor, *A practical guide to splines : with 32 figures*. Springer, 2001.

[17] Ó. Reinoso, L. Jiménez, L. Payá, A. Gil, and A. Vidal, *MATLAB: conceptos básicos y descripción gráfica*. Universidad Miguel Hérnandez, 2016.

[18] "MATLAB." [Online]. Available: https://www.mathworks.com/products/matlab.html. [Accessed: 27-Jan-2019].

[19] Z. Zhang, B. Wu, Y. Zhou, and X. Zhang, "Low-Complexity Hardware Interleaver/Deinterleaver for IEEE 802.11a/g/n WLAN," *VLSI Des.*, vol. 2012, pp. 1–7, Mar. 2012.

[20] *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012) : IEEE Standard for Information technology--Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access.* IEEE, 2016.

6. ANEXOS

ANEXO I. Programa Principal Implementado en Matlab

ANEXO II. Funciones Implementadas en Matlab.

ANEXO I

Definir parámetros de la transmisión

```
s1 = RandStream('mt19937ar','Seed',12345);  % Generar el tipo de semilla

nFFT = 64;                                  % Longuitud de la IFFT
cpLen = nFFT/4;                             % Longuitud del prefijo cíclico
nullIdx = [1:6 33 60:64].';                 % Subportadoras nulas

BlockSize=3;                                % Tamaño del bloque

trellis = poly2trellis(7,[171 133]);        % Codificador convolucional
tbl = 32;                                   % Codificador convolucional
rate = 1/2;                                 % Tasa de codificación
```

Iniciar vectores

```
EbNoVec = (0:2:22)';               % Vector del SNR
berVec = zeros(length(EbNoVec),3); % Inicializar la BER
errorStats = zeros(1,3);           % Inicializar las matrices estadísticas de error.
MSEVec=zeros(length(EbNoVec),1);   % Inicializar el Vector del MSE

% Ingreso de opciones de la tx
[modOrd,maxNumTx,EsqPort,EstPerfCanal,TipoInterp,TipoEstimacion,cod]=ingreso(app);
% Generar los pilotos
[pilotIdx,pilots,numTones]=Generatepilots(app,EsqPort,nFFT,nullIdx);
% Config moduladores, canal awgn y rayleigh
[qam,qamdem,chanAWGN,chan,symOffset,tau_rms]=configurar(app,modOrd,cpLen);
% Genera el nombre para el título de la gráfica
nombre=nombrar(app,EstPerfCanal,TipoInterp,TipoEstimacion,EsqPort,cod);

% Cálculo del SNR
if cod==0
    snrVec = EbNoVec + 10*log10(modOrd) + 10*log10(numTones/nFFT);
else
    snrVec = EbNoVec + 10*log10(modOrd*rate) + 10*log10(numTones/nFFT);
end

for m = 1:length(EbNoVec)
    snr = snrVec(m);
    % Reinicia el número de errores
    errorRate = comm.ErrorRate('ResetInputPort',true);
    cont=1;

    while cont <= maxNumTx
```

Transmisor

```
        if cod==0
            % Generar los datos a tx
            dataIn = randi(s1,[0 1],modOrd*numTones,1);
            % Modulación QPSK/QAM y OFDM

txSig=transmision(app,qam,dataIn,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Blo
ckSize);
        else
            % Generar los datos a tx
            dataIn = randi(s1,[0 1],modOrd*numTones*rate,1);
            dataCod=codifyinterl(app,dataIn,trellis,EsqPort,modOrd);
            % Modulación QPSK/QAM y OFDM

txSig=transmision(app,qam,dataCod,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Bl
ockSize);
        end
```

Canal

```
        [H,Hdat]=Channel(app,nFFT,cpLen,symOffset,chan,EsqPort,cont,BlockSize);
        % Aplicar el Rayleigh y AWGN a la senal Tx
        rxSig=AplyChannel(app,txSig,chan,chanAWGN,snr);
```

Receptor

```
        if EsqPort <=2
            % Demodulación OFDM
            [x,pilotosRx] = ofdmdemod(rxSig,nFFT,cpLen,symOffset,nullIdx,pilotIdx);
        else
            % Demodulación OFDM
            x= ofdmdemod(rxSig,nFFT,cpLen,symOffset,nullIdx);
        end

        if EsqPort==3 && (rem(cont,BlockSize))==1
            pilotosRx=x;
        end
hpilotos=estimacion(app,pilotosRx,pilots,TipoEstimacion,numTones,tau_rms,snr,EsqPort
);

[h,hdat]=interpolacion(app,pilotIdx,hpilotos,nFFT,EsqPort,TipoInterp,TipoEstimacion)
;
        % Ecualización y Demodulación QPSK o QAM
        dataOut=ecualizacion(app,x,hdat,Hdat,qamdem,EstPerfCanal);

        if cod==1
            % Decodificación y Desentrenlazado
            dataOut=decodifyinterl(app,dataOut,trellis,tbl,EsqPort,modOrd);
        end
```

Recopilar estadísticas de los errores

```
        if EsqPort <=2
            if cod==0
                errorStats = errorRate(dataIn,dataOut,0);
            else
                errorStats=errorRate(dataIn(1:end-tbl),dataOut(tbl+1:end),0);
            end
        else
            if (rem(cont,BlockSize))~= 1
                if cod==0
                    errorStats = errorRate(dataIn,dataOut,0);
                else
                    errorStats=errorRate(dataIn(1:end-tbl),dataOut(tbl+1:end),0);
                end
            end
        end
        cont=cont+1;
        mse=immse(abs(Hdat),abs(hdat));
        MSEVec(m)=MSEVec(m)+mse;
end


drawcanal(app,H,h,pilotIdx,hpilotos,EstPerfCanal,EsqPort,m,TipoEstimacion,nombre);

    MSEVec(m)=MSEVec(m)/cont;
    berVec(m,:) = errorStats;                    % Guardar datos del BER
    errorStats = errorRate(dataIn,dataOut,1); % Restablecer el numero de errores a 0
end
```

ANEXO II

Función ingreso

```
function
[modOrd,maxNumTx,EsqPort,EstPerfCanal,TipoInterp,TipoEstimacion,cod]=ingreso(app)
app.Modulacion.ItemsData = [2 4 6];% Orden de Modulación(QPSK=2,16QAM=4,64QAM=6)
modOrd= app.Modulacion.Value;
maxNumTx=app.Transmisiones.Value;     % Número maximo de transmisiones
app.Pilotos.ItemsData = [1 2 3];  % Esquema de Pilotos (Comb8=1,Comb4=2,Block=3)
EsqPort=app.Pilotos.Value;
cod=app.Codificacion.Value;           % ON(1) OFF(0) Codificación e Interleaving
app.Estimacion.ItemsData = [1 2];     % Tipo de Estimación (LS=1 o MMSE=2)
TipoEstimacion= app.Estimacion.Value;
TipoInterp= app.Interpolacion.Value;  % Tipo de Interpolación(lineal o spline)

s=app.Perfecta.SelectedObject.Text;   % Estimación Perfecta de Canal
switch s
    case 'No'
        EstPerfCanal=0;
    case 'Si'
        EstPerfCanal=1;
end
end
```

Función Generatepilots

```
function [pilotIdx,pilots,numTones]=Generatepilots(app,EsqPort,nFFT,nullIdx)
switch EsqPort
    case 1 %Comb8
        pilotIdx = (7:8:55)';                           % Índice de los pilotos
        pilots = pskmod(ones(1,7).',4);                  % Generar los pilotos
        numTones = nFFT-length(nullIdx)-length(pilotIdx); % Num subportadoras de
datos
    case 2 %Comb4
        pilotIdx = (7:4:59)';                            % Índice de los pilotos
        pilots = pskmod(ones(1,14).',4);                 % Generar los pilotos
        numTones = nFFT-length(nullIdx)-length(pilotIdx); % Num subportadoras de
datos
    case 3 %Block
        pilotIdx = [(7:32),(34:59)]';                    % Índice de los pilotos
        pilots = pskmod(ones(1,52).',4);                 % Generar los pilotos
        numTones = nFFT-length(nullIdx);                 % Num subportadoras de
datos
end
end
```

Función configurar

```
function [qam,qamdem,chanAWGN,chan,symOffset,tau_rms]=configurar(app,modOrd,cpLen)
% Creación del modulador y demodulador QAM con potencia ponderada
qam =
comm.RectangularQAMModulator('ModulationOrder',2^modOrd,'BitInput',true,'Normalizat
ionMethod','Average power');
qamdem =
comm.RectangularQAMDemodulator('ModulationOrder',2^modOrd,'BitOutput',true,'Normali
zationMethod','Average power');
% Configurar el canal AWGN de manera que se pueda ingresar de manera externa la
Eb/No
chanAWGN = comm.AWGNChannel('NoiseMethod','Variance','VarianceSource','Input
port');
% Canal Rayleigh
maxDopp = 0;       % Ancho de banda de Doppler
sRate = 20e6;      % Frecuencia de muestreo de la señal de entrada
s2=app.Rayleigh.SelectedObject.Text;
switch s2
    case '3 Rayos'
        pathGains=[app.PDP1_3.Value,app.PDP2_3.Value,app.PDP3_3.Value]; % Potencias
        pathDelays=[app.Delay1_3.Value,app.Delay2_3.Value,app.Delay3_3.Value]; %
Delays
        pathDelays=pathDelays*1e-9;
    case '4 Rayos'
        pathGains=[app.PDP1_4.Value,app.PDP2_4.Value,...
            app.PDP3_4.Value,app.PDP4_4.Value];                          % Potencias
        pathDelays=[app.Delay1_4.Value,app.Delay2_4.Value,...
            app.Delay3_4.Value,app.Delay4_4.Value];                      % Delays
        pathDelays=pathDelays*1e-9;
end
chan =
comm.RayleighChannel('PathGainsOutputPort',true,'MaximumDopplerShift',maxDopp,...
    'PathDelays',pathDelays,'AveragePathGains',pathGains,'SampleRate',sRate);
sampIdx = round(pathDelays*sRate) + 1;
symOffset = min(max(sampIdx),cpLen);% Determine el desface del muestreo del símbolo
% Respuesta impulsiva del canal
posicion=round(pathDelays(2:end)*sRate);
hImp=zeros(1,posicion(end));
hImp(posicion)=pathGains(2:end);
hImp=[pathGains(1) hImp];

% Obtener RMS delay spread
k=0:length(hImp)-1;
hh = hImp*hImp';
tmp = hImp.*conj(hImp).*k;
r = sum(tmp)/hh;              % Mean excess delay (ecuación 1.4)
r2 = tmp*k.'/hh;              % Segundo Momento central del PDP (ecuación 1.6)
tau_rms = sqrt(r2-r^2);       % RMS delay spread (ecuación 1.5)
end
```

Función transmision

```
function
txSig=transmision(app,qam,dataIn,nFFT,cpLen,nullIdx,pilotIdx,pilots,EsqPort,cont,Bl
ockSize)
if EsqPort <=2 % Comb8 y Comb4
    qamTx = qam(dataIn);                                        % Modular en QAM
    txSig = ofdmmod(qamTx,nFFT,cpLen,nullIdx,pilotIdx,pilots); % Modular en OFDM
else           % Block
    if (rem(cont,BlockSize))==1 % Los simbolos son los pilotos
        qamTx=pilots;
    else                        % Los símbolos son datos
        qamTx = qam(dataIn);                                    % Modular en QAM
    end
    txSig = ofdmmod(qamTx,nFFT,cpLen,nullIdx);                  % Modular en OFDM
end
end
```

Función codifyinterl

```
function dataCod=codifyinterl(app,dataIn,trellis,EsqPort,modOrd)
dataEnc = convenc(dataIn,trellis); %Codificación
%Interleaving
NCBPS=length(dataEnc);
switch EsqPort
    case 1       % Comb8
        Ncol=15; % Número de columnas
    case 2       % Comb4
        Ncol=19; % Número de columnas
    case 3       % Block
        Ncol=13; % Número de columnas
end
Nrow=NCBPS/Ncol; % Número de filas
% Primera Permutación
k=[0:NCBPS-1];
i=(Nrow)*mod(k,Ncol)+floor(k/Ncol);                % Indice i
Tx=ordenar(app,i,dataEnc);

% Segunda Permutación
s=max(modOrd/2,1);
i=[0:NCBPS-1];
j=s*floor(i/s)+mod((i+NCBPS-floor(Ncol*i/NCBPS)),s); % Indice j
dataCod=(ordenar(app,j,Tx))';
end
```

Función ordenar

```
function Y=ordenar(app,i,X) % Ordenar usando los subindices i
Y=zeros(1,length(i));
for cont=1:length(i)
    aux=i(cont)+1;
    Y(cont)=X(aux);
end
end
```

Función Channel

```
function [H,Hdat]=Channel(app,nFFT,cpLen,symOffset,chan,EsqPort,cont,BlockSize)
if EsqPort <=2                  % Comb8 y Comb4
    reset(chan);                % Reiniciar el canal
else
    if (rem(cont,BlockSize))==1 % Block
        reset(chan);            % Reiniciar el Canal
    end
end

% Obtener el canal
datos= ones(nFFT,1);                          % Generar los datos a Tx
l=ofdmmod(datos,nFFT,cpLen);                  % Modular en OFDM
filtrado=chan(l);                             % Aplicar a la Señal el Canal
H=ofdmdemod(filtrado,nFFT,cpLen,symOffset);  % Demodulación OFDM
Hdat=seleccion(app,H,EsqPort);           % Seleccionar solo las portadoras de datos
end
```

Función AplyChannel

```
function rxSig=AplyChannel(app,txSig,chan,chanAWGN,snr)
fadSig = chan(txSig);               % Aplicar el canal RAYLEIGH
powerDB = 10*log10(var(fadSig));    % Calcular la potencia de la señal Tx
noiseVar = 10.^((powerDB-snr)/10);  % Calular el noise variance
rxSig = chanAWGN(fadSig,noiseVar);  % Aplicar el canal AWGN
end
```

Función estimacion

```
function
hpilotos=estimacion(app,pilotosRx,pilots,TipoEstimacion,numTones,tau_rms,snr,EsqPor
t)
if TipoEstimacion==1                        % Estimación de canal LS
    hpilotos =pilotosRx./pilots;
else                                         % Estimación de canal MMSE
    H_tilde = (conj(pilotosRx./pilots))';  % Estimación tipo LS
    switch EsqPort
        case 1      % Comb8
            Nps=8;  % Espaciado entre pilotos
            Np=length(pilotosRx);
        case 2      % Comb4
            Nps=4;  % Espaciado entre pilotos
            Np=length(pilotosRx);
        case 3      % Block
            Nps=1;  % Espaciado entre pilotos
            Np=length(pilotosRx)+1;
            H_tilde=[H_tilde(1:26),H_tilde(26:52)];
    end

    snrlineal = 10^(snr*0.1);
    K1 = repmat((0:1:52).',1,Np); % 52 subportadoras entre datos y pilotos
    K2 = repmat((0:Np-1),52+1,1);
    K3 = repmat((0:Np-1).',1,Np);
    K4 = repmat((0:Np-1),Np,1);
    df = 1/numTones;
    j2pi_tau_df = 1i*2*pi*tau_rms*df;
    % Correlación para un PDP exponencial(ecuacion 1.33)
    rf2 = 1./(1+j2pi_tau_df*Nps*(K3-K4));
    % Correlación cruzada entre H y H_tilde(ecuación 1.29)
    Rhp = 1./(1+j2pi_tau_df*(K1-K2*Nps));
    % Matriz de autocorrelación de H_tilde(ecuación 1.30)
    Rpp = rf2 + eye(length(H_tilde),length(H_tilde))/snrlineal;
    % Estimador de canal tipo MMSE(ecuación 1.31)
    hpilotos = transpose((Rhp/Rpp)*H_tilde.');
end
end
```

Función interpolacion

```
function
[h,hdat]=interpolacion(app,pilotIdx,hpilotos,nFFT,EsqPort,TipoInterp,TipoEstimacion
)
if TipoEstimacion==1 % Estimación LS
    if EsqPort <=2   % Comb8 y Comb4
        h = interp1(pilotIdx,hpilotos,(1:1:nFFT)',TipoInterp,'extrap'); % Interpola
    else             % Block
        h = [NaN(6,1);(hpilotos(1:1:26));NaN(1,1);(hpilotos(27:52));NaN(5,1)]; % No
int
    end
else                 % Estimación MMSE
    h=[NaN(6,1);conj(hpilotos');NaN(5,1)];  % No interpola
end
hdat=seleccion(app,h,EsqPort);
end
```

Función seleccion

```
function hdat=seleccion(app,h,EsqPort)
h=h';
switch EsqPort
    case 1

hdat=[h(8:14),h(16:22),h(24:30),h(32),h(34:38),h(40:46),h(48:54),h(56:59)]';
    case 2
        hdat=[h(8:10),h(12:14),h(16:18),h(20:22),h(24:26),h(28:30),h(32)...
            ,h(34),h(36:38),h(40:42),h(44:46),h(48:50),h(52:54),h(56:58)]';
    case 3
        hdat=[h(7:32),h(34:59)]';
end
end
```

Función ecualizacion

```
function dataOut=ecualizacion(app,x,hdat,Hdat,qamdem,EstPerfCanal)
if EstPerfCanal==0
    eqH = conj(hdat)./(conj(hdat).*hdat);
else
    eqH = conj(Hdat)./(conj(Hdat).*Hdat);    % Estimación perfecta de canal
end
eqSig = eqH.*x;                              % Ecualización
dataOut = qamdem(eqSig);                     % Demodulación QAM o QPSK de la señal
end
```

Función decodifyinterl

```
function dataDecod=decodifyinterl(app,dataOut,trellis,tbl,EsqPort,modOrd)
% Deinterleaving
NCBPS=length(dataOut);
switch EsqPort
    case 1       % Comb8
        Ncol=15; % Número de columnas
    case 2       % Comb4
        Ncol=19; % Número de columnas
    case 3       % Block
        Ncol=13; % Número de columnas
end
Nrow=NCBPS/Ncol; % Número de filas
% Inverso de la Segunda Permutación
j=[0:NCBPS-1];s=max(modOrd/2,1);
i=s*floor(j/s)+mod((j+floor(Ncol*j/NCBPS)),s); % Indice i
Rx=ordenar(app,i,dataOut);
% Inverso de la Primera Permutación
i=[0:NCBPS-1];
k=Ncol*i-(NCBPS-1)*floor(i/Nrow);              % Indice k
Rx2=(ordenar(app,k,Rx))';
dataDecod = vitdec(Rx2,trellis,tbl,'cont','hard'); % Decodificación
end
```

Función nombrar

```
function nombre=nombrar(app,EstPerfCanal,TipoInterp,TipoEstimacion,EsqPort,cod)
switch EsqPort
    case 1
        Port=' Comb 8';
    case 2
        Port=' Comb 4';
    case 3
        Port=' Block';
end
if EstPerfCanal==1
    nombre='Estimacion Perfecta de Canal';
else
    if TipoEstimacion==1
        if EsqPort <=2
            nombre=[Port ' LS ' TipoInterp];
        else
            nombre=[Port ' LS '];
        end
    else
        nombre=[Port ' MMSE '];
    end
end
if cod==1
    nombre=[nombre ' Cod 1/2+ Int'];
end
end
```

Función drawEbNo

```
function drawEbNo(app,EbNoVec,berVec,MSEVec,nombre,EstPerfCanal)
set(groot,'defaultAxesLineStyleOrder',{'-','--s','--d','--x','--*','--p','--v'})
% BER
plot(app.BER,EbNoVec,berVec(:,1),'LineWidth',1,'Tag',nombre);
al=app.BER; al.YScale = 'log'; al.NextPlot='add'; al.XLim = [0 22]; al.YLim =
[0.001 1];
al.LineStyleOrderIndex=al.LineStyleOrderIndex+1;
app.nombreBERIdx=app.nombreBERIdx+1;
app.nombreBERVec(app.nombreBERIdx)=nombre;
legend(app.BER,app.nombreBERVec(1:app.nombreBERIdx));
disableDefaultInteractivity(al);
% MSE
ax=app.MSE;
if EstPerfCanal==0
    plot(app.MSE,EbNoVec,MSEVec,'LineWidth',1,'Tag',nombre);
    ax.NextPlot='add'; ax.XLim = [0 22];
    ax.LineStyleOrderIndex=ax.LineStyleOrderIndex+1;
    app.nombreIdx=app.nombreIdx+1;
    app.nombreVec(app.nombreIdx)=nombre;
    legend(app.MSE,app.nombreVec(1:app.nombreIdx));
    disableDefaultInteractivity(ax);
else
    ax.ColorOrderIndex=ax.ColorOrderIndex+1;
    ax.LineStyleOrderIndex=ax.LineStyleOrderIndex+1;
end
end
```

Función drawcanal

```
function
drawcanal(app,H,h,pilotIdx,hpilotos,EstPerfCanal,EsqPort,m,TipoEstimacion,nombre)
% Gráfico del canal
ejes=[app.EbNo0,app.EbNo2,app.EbNo4,app.EbNo6,app.EbNo8,...
    app.EbNo10,app.EbNo12,app.EbNo14,app.EbNo16,app.EbNo18,app.EbNo20,app.EbNo22];
cla(ejes(m));
plot(ejes(m),real(H),'-');     % Gráfico
legend(ejes(m),'Canal')
ejes(m).NextPlot='add';
ejes(m).XLim = [6 60];
if EstPerfCanal==0
    plot(ejes(m),real(h),'-'); % Graficar el Estimado
    legend(ejes(m),'Canal',nombre)
    if TipoEstimacion==1
        if EsqPort <=2
            plot(ejes(m),pilotIdx,real(hpilotos),'*');  %Graficar los Pilotos
            legend(ejes(m),'Canal',nombre,'pilotos')
        end
    end
end
end
```

ORDEN DE EMPASTADO