ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

SIMULACIÓN DE LOS DETECTORES SQRD Y MMSE-SQRD EN UN SISTEMA OFDM CON UN CANAL VARIANTE EN EL TIEMPO USANDO MATLAB

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

ALLISON MONSERRATH ALAVA GARCÍA

DIRECTOR: DIEGO JAVIER REINOSO CHISAGUANO

Quito, noviembre 2021

AVAL

Certifico que el presente trabajo fue desarrollado por Allison Monserrath Alava García, bajo mi supervisión.

DIEGO JAVIER REINOSO CHISAGUANO
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Allison Monserrath Alava García, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado 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 dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

ALLISON MONSERRATH ALAVA GARCÍA

DEDICATORIA

A la pequeña Allison de hace unos años,

nos perdimos, nos encontramos,

creímos en nosotras, lo logramos.

Ahora, vamos por más.

AGRADECIMIENTO

A mis pilares fundamentales en todo este proceso de estudios y crecimiento, mis padres. Gracias por ser mi piedra de fortaleza en tiempos de cansancio, por ser mi motivación diaria para ser cada vez mejor, por su sensatez entregada a mí en palabras y gestos. A ustedes que jamás dejaron de creer en mí y a su cansancio ausente con tal de darme todo. Sin ustedes esto no hubiera sido posible. Este es el fruto de todos sus esfuerzos y dedicación, es lo mínimo con lo que puedo agradecerles, pero sin duda no será lo último con lo que prometo enorgullecerlos. Son mi vida, hoy y siempre. Los amo.

Al ñaño. Porque uno pide muchas cosas, pero la vida es sabia para darte lo que necesitas. Mi inocencia de la infancia pedía un hermano mayor, pero mi propósito era ser la mayor. A ti, pequeño, porque no hay mejor referente que lo que ves día a día y por eso pido al cielo me dé la mejor guía para ir abriéndote camino. Porque espero poder aprender y entender lo que más pueda para poder pasarte la receta de vida a ti. Sé que tendrás que vivir tus errores para aprender, pero en medida de lo que pueda, aquí estaré para apoyarte y ayudarte a ser mejor. Gracias por tu inocencia y amor, eres mi motivación, te quiero mucho.

Al Dios de amor que ahora conozco. Al que siempre ha estado para darme paz, para guiarme, para hacerme fuerte. Sin ti, no hubiera podido.

A mi familia materna, que siempre me ha dado su amor y apoyo incondicional. Mis papitos queridos, quienes me han consentido y aconsejado de la mejor manera. A mi mamita querida por el amor más puro y grande que haya podido existir. A mi papito querido por ser el superhéroe encarnado. Gracias por siempre estar en buenas y malas. A mi ñaña mayor y ejemplo, Liss. Gracias por ser amiga, apoyo y modelo a seguir. A Walter, por ser apoyo, oídos y comprensión en momentos de confusión y tempestad. A mi hermanita de travesuras y cómplice de infancia, Gaby.

A mi familia paterna, que me ha dado fortaleza y carácter. A mis abuelitos, Dios les pague porque en su hogar jamás faltó un delicioso plato de comida y una mesa para compartirlo juntos. A mi abuelita por su ejemplo de mujer aguerrida. A mis tíos por su cariño. A mi primo, el más alto y a los pequeñines.

A mis mejores amigas, de quienes aprendí mucho y a quienes admiro un montón por la grandeza de su corazón y la tenacidad de sus capacidades. Mi Geme y mi Mushu, gracias

por estar para mí, gracias por tanto. A la familia que uno elige, los amigos, mis “Chachos”. Gracias por hacer de los días en la universidad, una aventura divertida y de momentos inolvidables. A Fredicin, por siempre ayudarme a ser mejor, por cuidarme y ser un amigo incondicional. A Brayitan por ser un apoyo constante para buenas y malas, por su compañía en mi camino por entender la vida y por ayudarme a brillar siempre. A Mari y Mishu, mis ‘chicas superpoderosas’ por ser ejemplar de mujeres, por el cariño y apoyo. A mis amigos: Jarita, Huguito, Javi, Frankie, Leñito, y Monito, por el compañerismo, por las risas y el acolite. A mis ‘controlitos’ por llenarme de alegrías y motivarme a superarme igual que ellos lo hacen. A Juanito por enseñarme que sea lo que sea ‘all iz well’ y a motivarnos a todos a ser nuestra mejor versión. A Edgarin por los abrazos llenos de paz en medio de los estudios y su linda amistad. A Luigi, gracias por brindarme un poco de su paz y buena onda, por ayudarme a desplegar velas. A Carlitos, Pablito y Dieguito, por darme la oportunidad de aprender de ellos, crecer profesionalmente y ser amigos de los que llegan al corazón. Y a todos quienes en el camino me dieron un poquito de su esencia para poder ser quien soy ahora.

Agradezco al Dr. Diego Reinoso por su apoyo y paciencia constantes en la realización del presente proyecto. Por ser un excelente guía en lo académico, por su búsqueda de forjar en sus estudiantes valores que hagan de nosotros excelentes profesionales y mejores personas. Gracias por ser un maestro exigente sin dejar de lado la calidez de un ser humano hacia sus estudiantes.

A la EPN por abrirme sus puertas al estudio de lo que hoy en más será mi profesión. No fue sencillo, pero nada que valga la pena lo es. Gracias ‘poli’ por los desafíos y el conocimiento. Gracias a todos quienes conforman la comunidad politécnica y día a día buscan hacer de la universidad un lugar mejor para las futuras generaciones.

Y a todos quienes han creído y creen en mí, ¡lo logramos!

Ally

ÍNDICE DE CONTENIDO

RESUMEN

En el presente proyecto de titulación se realiza la simulación de un canal variante en el tiempo, utilizando la técnica OFDM *(Orthogonal Frequency-Division Multiplexing)* y empleando en la recepción los detectores ZF *(Zero-Forcing)*, MMSE *(Minimum Mean Squared Error)*, QRD (*QR Decomposition*), SQRD *(Sorted QR Decomposition)* y MMSE-SQRD. Con la finalidad de determinar el detector que representa un mejor rendimiento, se obtienen los resultados de la detección de cada algoritmo en forma gráfica visibilizando la comparación entre el BER *(Bit Error Rate)* vs Eb/No *(Energy per Bit to the Spectral Noise Density)*. Además, se realiza un análisis del costo computacional de cada detector, basándose en el tiempo de ejecución de cada uno de ellos.

En el capítulo 1 se detalla la teoría relacionada con la técnica OFDM, el canal inalámbrico y sus efectos, el estándar IEEE 802.11p, las características y algoritmos de los detectores ZF, MMSE, SQRD y MMSE-SQRD.

En el capítulo 2 se presenta la metodología empleada. Se explica cómo ha sido desarrollado el código para la implementación del canal y los detectores.

En el capítulo 3 se detalla los resultados obtenidos. Se presentan las gráficas BER vs Eb/No y un análisis comparativo. También se presenta una recopilación de los tiempos de ejecución de cada detector y su análisis.

Finalmente, en el capítulo 4 se muestran las conclusiones obtenidas posterior a la realización del trabajo de titulación y las recomendaciones sugeridas para posteriores trabajos.

PALABRAS CLAVE: OFDM, ZF, MMSE, QRD, SQRD, MMSE-SQRD

ABSTRACT

In this final project, the simulation of a time-varying channel is carried out, using the OFDM *(Orthogonal Frequency-Division Multiplexing)* technique and using the detectors ZF *(Zero-Forcing)*, MMSE *(Minimum Mean Squared Error)*, QRD (*QR Decomposition*), SQRD *(Sorted QR Decomposition)* and MMSE-SQRD. In order to determine the detector that represents the best performance, the results are graphically presented by displaying the BER *(Bit Error Rate)* vs Eb/No *(Energy per Bit to the Spectral Noise Density)* comparison. In addition, there is a computational cost analysis based on the time of execution of each of the detectors.

Chapter 1 details the theory related to OFDM technique, the wireless channel and its effects, the IEEE 802.11p standard, the characteristics and algorithms of the ZF, MMSE, SQRD and MMSE-SQRD detectors.

Chapter 2 presents the methodology used. There is an explanation of how the code was done to create de channel and the detector.

Chapter 3 details the results obtained. The graphics about BER vs Eb/No and a comparative analysis are presented. A compilation of the execution times of each detector and its analysis is also presented.

Finally, chapter 4 shows the conclusions obtained after completing the degree work and the suggested recommendations for subsequent works.

KEYWORDS: OFDM, ZF, MMSE, QRD, SQRD, MMSE-SQRD

1. INTRODUCCIÓN

En la actualidad, la demanda por comunicaciones móviles es desbordante, por lo cual ciertas comunicaciones alámbricas han sido reemplazadas por las comunicaciones inalámbricas. Esto tomando en cuenta que las comunicaciones inalámbricas permiten que exista movilidad gracias a la ausencia de un cable o medio de transmisión físico. Un ejemplo claro que encontramos en el diario vivir es la conexión de laptops, tablets, celulares y demás dispositivos al router del hogar. Resultaría muy incómodo tener todos esos dispositivos conectados por un cable al router. Eso sin mencionar que no se podría conectar tantos dispositivos al router sin la necesidad de un switch. La ventaja de la movilidad nos permite estar en cualquier lugar de la casa y movernos indistintamente sin perder conexión a Internet.

Sin embargo, en el área de las comunicaciones inalámbricas se conoce que el problema radica fundamentalmente en los efectos que se producen en el canal, principalmente si se trata de un canal variante en el tiempo. Uno de los escenarios claros donde se puede encontrar un canal variante en el tiempo, es en un entorno donde se tienen comunicaciones vehiculares o V2V que utilizan estándares como el IEEE 802.11p. En este caso, se pueden tener diversas situaciones respecto del transmisor y el receptor ya que estos pueden estar o no en movimiento.

El movimiento de los elementos del sistema provocará efectos severos en el canal dificultando una comunicación efectiva. Además de los efectos que de por sí presenta el canal de comunicaciones, la movilidad causa algunos efectos como lo son el efecto Doppler y la Interferencia Entre Subportadoras *(Intercarrier Interference – ICI)* [1].

En un ambiente de comunicaciones inalámbricas, se requieren velocidades altas y se presentan problemas como el desvanecimiento multitrayectoria y el desvanecimiento selectivo en frecuencia, para lo cual se emplean técnicas OFDM *(Orthogonal Frequency-Division Multiplexing)* que solucionen dichos aspectos. Entre las ventajas de utilizar OFDM están la alta eficiencia espectral, la habilidad para mitigar el desvanecimiento multitrayectoria selectivo en frecuencia y moderar la interferencia [2]. OFDM es una técnica bastante eficiente pero los efectos producidos por un canal variante en el tiempo degradan la señal al momento de su recepción. Por ello, es necesario considerar las complicaciones que se tendrá en el receptor al momento de hacer la detección de la señal. Existen varios métodos que permiten facilitar la detección de las señales, estos se clasifican en detectores lineales y no lineales.

Los detectores lineales se caracterizan por su baja complejidad computacional, pero esto no garantiza que sean los más eficientes para detectar la señal. Por otro lado, los detectores no lineales resultan mucho más eficientes que los detectores lineales, pero su complejidad computacional es mucho más alta. Dentro de los detectores lineales se tienen detectores como el de Forzamiento a Cero *(Zero Forcing - ZF)* y el de Error Cuadrático Medio Mínimo *(Minimum Mean Square Error - MMSE)* [3].

Por otra parte, existen detectores no lineales como el de Máxima Verosimilitud *(Maximum Likelihood – ML)*, la Descomposición QR ordenada *(Sorted QR Decomposition – SQRD)* y la Descomposición QR ordenada MMSE *(MMSE Sorted QR Decomposition – MMSE SQRD)*. El método ML es el más eficiente de los detectores existentes, pero su complejidad computacional es extremadamente alta, por lo que no podrían implementarse ya que su procesamiento sería muy lento. A este le sigue el MMSE – SQRD y el SQRD, siendo el último el de menor complejidad computacional [4] [5]. Por lo tanto, es importante identificar cuál de los métodos resulta más adecuado considerando las condiciones del canal y las ventajas de cada uno de los detectores.

1.1. OBJETIVOS

El objetivo general de este Proyecto Técnico es simular los detectores SQRD y MMSE-SQRD en un sistema OFDM con un canal variante en el tiempo usando MATLAB.

Los objetivos específicos del Proyecto Técnico son:

- Describir qué son las técnicas de detección lineal y no lineal.
- Implementar un sistema OFDM con un canal variante en el tiempo en MATLAB.
- Simular las técnicas de detección en el sistema OFDM.
- Realizar pruebas de funcionamiento de cada detector.
- Analizar los resultados obtenidos en términos de rendimiento y costo computacional.

1.2. ALCANCE

Se implementará una simulación a través de MATLAB de un sistema OFDM donde se realizará en el lado del receptor dos métodos de detección no lineal. Para el sistema ya descrito se debe considerar los siguientes factores:

- Se considerará sincronización perfecta.
- No se va a tomar en cuenta la etapa de codificación FEC (*Forward Error Correction*).

- Se tomará en cuenta que ya se conoce el estado del canal.
- El sistema OFDM que se utilizará cuenta con un solo receptor y un solo transmisor.
- El canal tendrá desvanecimiento además de ruido blanco. El desvanecimiento tendrá una distribución de tipo Rayleigh de dos tipos: selectivo en frecuencia y variante en el tiempo.

Se plantea la implementación de un sistema con un transmisor. El flujo de datos pasará por una etapa de modulación digital que emplea tres tipos de modulaciones: QPSK, 16QAM y 64QAM. Posteriormente, se utiliza la técnica OFDM para transmitirlos mediante la antena hacia el canal. Todo esto, tomando en cuenta que el canal es variante en el tiempo, por lo que el transmisor y el receptor pueden estar en movimiento uno con respecto del otro. En recepción se cuenta con una sola antena. Se recibe entonces las señales enviadas por el transmisor bajo los efectos del canal como el ICI y el efecto Doppler. Se realiza el proceso de demodulación OFDM y demodulación digital, para luego proceder con la etapa final de detección. Para esta etapa se implementarán cuatro tipos de detectores. Los detectores no lineales a utilizarse son principalmente el SQRD y el MMSE - SQRD. Con el propósito de comparar su rendimiento, se simularán también los detectores lineales ZF, MMSE y *one-tap*.

En los bloques de modulación y demodulación OFDM se trabajará con el estándar IEEE 802.11p que especifica los siguientes parámetros: 64 subportadoras totales de las cuales 48 subportadoras son de información, cuatro subportadoras son piloto y 12 subportadoras son nulas.

Para el análisis de los resultados, el programa arrojará gráficas de BER (*Bit Error Rate*) vs Eb/No (*Energy per Bit to the Spectral Noise Density*) para cada uno de los detectores utilizados. También se realizará un análisis del costo computacional que representa cada detector para con esto poder identificar cuál de ellos resulta mejor tomando también en cuenta su rendimiento según las gráficas que se obtengan.

No existe producto final demostrable ya que solo se implementará una simulación en MATLAB.

1.3. MARCO TEÓRICO

En esta sección se presentan los conceptos fundamentales para la realización de este Proyecto Técnico. Entre ellos, se tiene la técnica OFDM y cómo funciona. Además, se abordan los parámetros del estándar IEEE 802.11p y el canal inalámbrico junto con los efectos que se tiene dentro del mismo. Además, se detallan los algoritmos one-tap, ZF, MMSE, SQRD y MMSE-SQRD.

1.3.1. CANAL INALÁMBRICO

Uno de los parámetros elementales para determinar el rendimiento de las comunicaciones inalámbricas es el entorno del canal inalámbrico. A diferencia de los canales alámbricos, el canal inalámbrico es dinámico e impredecible por lo cual resulta difícil hacer un análisis exacto del sistema de comunicaciones inalámbrico.

1.3.1.1. Comportamiento de las ondas de radio

En el ámbito de un canal inalámbrico, la radio-propagación se refiere al comportamiento de las ondas de radio cuando son propagadas del transmisor al receptor. En el curso de la propagación, las ondas son afectadas principalmente por tres distintos fenómenos físicos: reflexión, difracción y dispersión (*Scattering*) [3].

- **Reflexión**

La reflexión es el fenómeno físico que se produce cuando la onda electromagnética que está siendo propagada incide en un objeto cuyas dimensiones son grandes en comparación con la longitud de onda, por ejemplo, la superficie terrestre, un edificio o un muro. Estos obstáculos forzan a que la potencia de la señal transmitida sea reflejada de vuelta a su origen y no continúe su curso normal hasta el receptor [3]. La onda reflejada puede sufrir de atenuaciones provocadas cuando el medio reflector absorbe parte de la energía de las ondas de radio propagadas [6].

- **Difracción**

La difracción se refiere al fenómeno que se suscita cuando en el trayecto entre el transmisor y el receptor existe una obstrucción por una superficie con irregularidades o pequeñas aberturas. Dichos obstáculos pueden ser paredes, edificios, montañas y demás. Se produce una especie de traspaso curvo de las ondas a través de estos objetos o sus aberturas. Las ondas secundarias que se producen a partir de la difracción son útiles para establecer una trayectoria entre el transmisor y el receptor, incluso si no existe línea-de-vista (LOS – *Line of Sight*) [3] [6].

- **Dispersión**

La dispersión es el fenómeno físico que forza a que la radiación de la onda electromagnética se desvíe de una trayectoria recta a causa de uno o más obstáculos cuyas dimensiones son pequeñas comparadas con la longitud de onda transmitida. Esto produce que la energía de la señal sea reducida y se extienda en todas las direcciones. Se conoce como dispersores (*Scatters*) a los objetos que provocan esta dispersión, los cuales pueden ser la vegetación, las señales de tránsito o postes de luz [3] [6].

Por todos los fenómenos mencionados, se conoce bien que la propagación de una onda de radio es un proceso complicado y muy poco predecible ya que es delimitado por dichos fenómenos y la intensidad de los mismos dependiendo de los diferentes entornos en los diferentes escenarios [3]. En la Figura 1.1 se muestran de manera gráfica los tres fenómenos mencionados anteriormente.

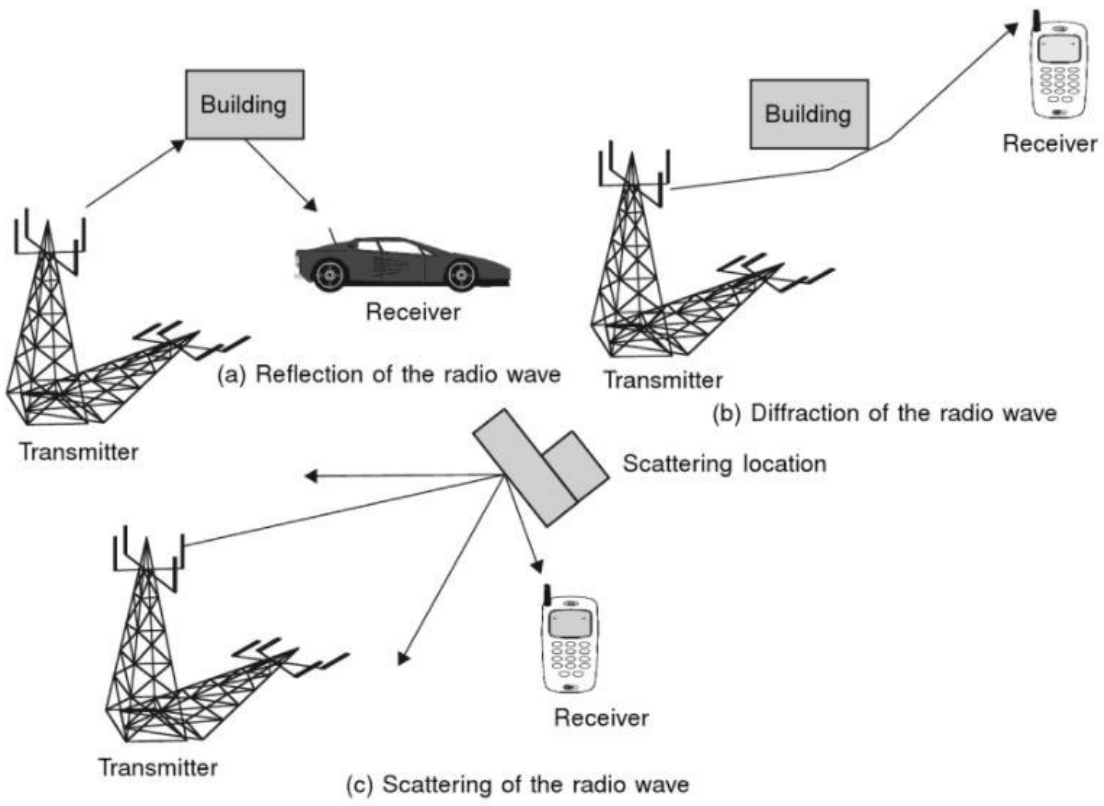

Figura 1.1 Reflexión, difracción y dispersión de las ondas de radio [6]

1.3.1.2. Multitrayectoria

En las áreas urbanas y no urbanas, se presenta la problemática relacionada a la altura de las antenas dado que en ocasiones las estructuras u objetos a los alrededores de las

antenas son de mayor altura. Esto causa que no exista una línea de vista hacia la estación base, incluso si existiera dicha línea de vista, la multitrayectoria aún ocurriría debido a las diferentes reflexiones con el piso y las demás estructuras.

El desvanecimiento en pequeña escala es ocasionado por la interferencia entre dos o más versiones de la señal transmitida, de las cuales, cada una llega al receptor en tiempos ligeramente diferentes. Estas versiones de la señal transmitida, son ondas conocidas como ondas multitrayectoria.

Las ondas que llegan a la antena de recepción llegan de diferentes direcciones y con diferentes retardos de propagación. En la antena de recepción las ondas son combinadas para obtener una señal resultante que puede variar ampliamente en cuanto a su amplitud y fase. Incluso si el receptor es estático, la señal recibida puede sufrir desvanecimiento debido al movimiento de los elementos que se encuentran a los alrededores [7].

1.3.1.2.1. Perfil de Retardo de Potencia

Como resultado del multitrayecto, la señal recibida se percibe como un conjunto de copias de la señal que fue transmitida originalmente. En la Figura 1.2 se puede observar de forma ilustrativa lo mencionado, donde $\mathrm{p}(\tau)$ representa la potencia de la señal recibida mediante cada trayectoria por la que viajó la señal y $\tau_n$representa el retardo de cada señal enviada por cada trayectoria. Cada una de estas trayectorias son conocidas como rayos o taps [8].

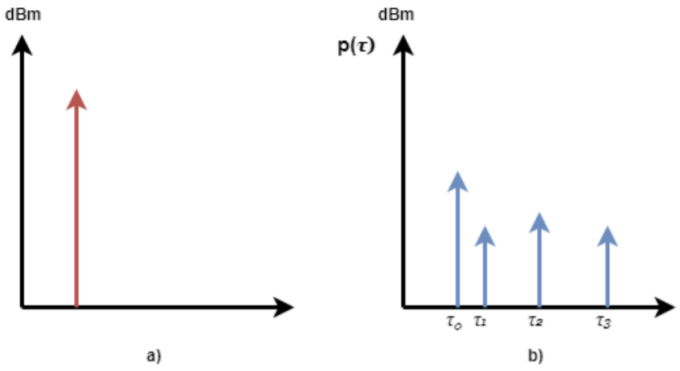

Figura 1.2 a) Señal enviada, b) Señal recibida [8]

La propagación por multitrayectoria ocasiona una dispersión severa de la señal transmitida. El grado esperado de dispersión es determinado mediante la medición del perfil de retardo de potencia (PDP – *Power Delay Profile)*. El PDP da una idea de la dispersión o de la distribución de la potencia transmitida mediante varias trayectorias en un canal que sufre de propagación por multitrayectoria [9].

La IEEE ha obtenido diferentes PDP para definir estadísticamente el modelo de canales variantes en el tiempo como lo son los canales para comunicaciones V2V (*Vehicle to Vehicle*), basándose en mediciones que fueron realizadas de manera independiente por la Universidad de Lund, CohdaWirelees y la Universidad de Berkeley. Gracias a la realización de este estudio, se pudo obtener el perfil PDP y el desplazamiento Doppler para cada rayo o *tap* considerando los siguientes escenarios [10]:

- ***Rural* LOS (Entorno rural con línea de vista)**

  Este escenario involucra ambientes amplios en donde no se encuentran otros vehículos, edificios ni tampoco vallas. Como se puede apreciar en la Figura 1.3, existe línea de vista entre el transmisor y el receptor, de ahí el nombre de este escenario. Dado que no se tiene ningún obstáculo sino más bien LOS, este escenario se convierte en el menos severo [10].

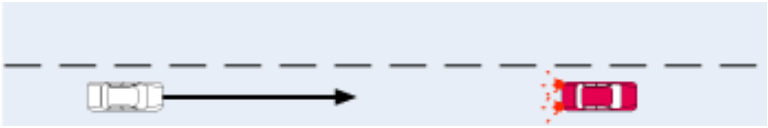

Figura 1.3 *Rural* LOS [10]

En la Tabla 1.1 se muestra el perfil PDP de este escenario.

Tabla 1.1 *Rural* LOS [10]

	Tap1	***Tap2***	***Tap3***	***Units***
Power	0	-14	-17	dB
Delay	0	83	183	ns
Doppler	0	492	-295	Hz
Profile	*Static*	*HalfBT*	*HalfBT*	

- ***Urban Approaching* LOS (Entorno urbano con línea de vista y proximidad)**

  Como su nombre lo indica, para este escenario se tienen dos vehículos aproximándose uno al otro en un entorno urbano en el cual existen edificios en los alrededores [10]. Sin embargo, como se muestra en la Figura 1.4 a pesar de tener construcciones alrededor se tiene línea de vista [8].

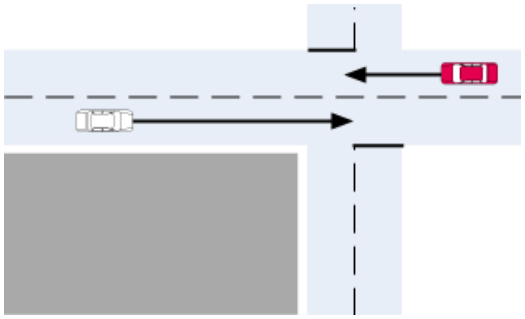

Figura 1.4 *Urban Approaching* LOS [10]

En la Tabla 1.2 se muestra el perfil PDP de este escenario.

Tabla 1.2 *Urban Approaching* LOS [10]

	Tap1	***Tap2***	***Tap3***	***Tap4***	***Units***
Power	0	-8	-10	-15	dB
Delay	0	117	183	333	ns
Doppler	0	236	-157	492	Hz
Profile	*Static*	*HalfBT*	*HalfBT*	*HalfBT*	

- ***Highway* LOS (Autopista con línea de vista)**

  En este escenario se tienen dos vehículos, uno siguiendo al otro en una autopista donde se tienen varios carriles. En este entorno se presentan señales de tránsito, pasos elevados, laderas y mayor cantidad de tráfico circulando tanto en el mismo sentido como en sentido contrario [10]. Como se puede ver en la Figura 1.5, a pesar de que se tienen más vehículos en la autopista que puedan causar reflexiones en la señal, aún se mantiene la línea de vista [8].

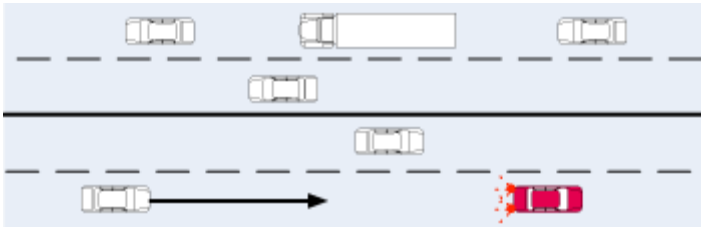

Figura 1.5 *Highway* LOS [10]

En la Tabla 1.3 se muestra el perfil PDP de este escenario.

Tabla 1.3 *Highway* LOS [10]

	Tap1	***Tap2***	***Tap3***	***Tap4***	***Units***
Power	0	-10	-15	-20	dB
Delay	0	100	167	500	ns
Doppler	0	689	-492	886	Hz
Profile	*Static*	*HalfBT*	*HalfBT*	*HalfBT*	

- ***Highway* NLOS (Autopista sin línea de vista)**

  Este es el escenario más severo dado que no se tiene línea de vista, considerando que en la autopista se tiene la presencia de un camión entre los vehículos [10]. Además, existen también más vehículos que puedan provocar reflexiones de la señal, como se muestra en la Figura 1.6 [8].

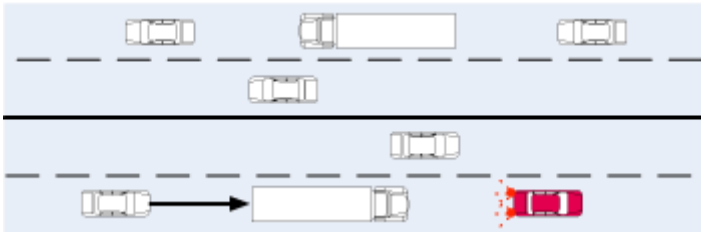

Figura 1.6 *Highway* NLOS [10]

En la Tabla 1.4 se muestra el perfil PDP de este escenario.

Tabla 1.4 *Highway* NLOS [10]

	Tap1	***Tap2***	***Tap3***	***Tap4***	***Units***
Power	0	-2	-5	-7	dB
Delay	0	200	433	700	ns
Doppler	0	689	-492	886	Hz
Profile	*Static*	*HalfBT*	*HalfBT*	*HalfBT*	

1.3.1.3. Desvanecimiento

Uno de los efectos característicos de los canales inalámbricos es el desvanecimiento, la variación de la amplitud de la señal respecto del tiempo y la frecuencia. El desvanecimiento, en contraste con la fuente más común de degradación de la señal como lo es el ruido aditivo, es otra fuente de degradación de la señal que se caracteriza por ser una perturbación de señal no-aditiva en el canal inalámbrico.

En los años 1950 y 1960, el fenómeno de desvanecimiento en el canal inalámbrico de comunicaciones fue inicialmente modelado para las bandas de HF (*High Frequency* 3~30

MHz), UHF (Ultra HF, 300~3000 GHz), y SHF (Super HF, 3~30 GHz). En la actualidad, los modelos de canales inalámbricos más comunes se han establecido de 800 MHz a 2.5 GHz.

El fenómeno de desvanecimiento se clasifica en dos tipos: a gran escala (*Large scale fading*) y a pequeña escala (*Small scale fading*). El primero ocurre a medida que el móvil se desplaza en distancias largas. Se produce debido a la pérdida de la trayectoria de la señal como una función de la distancia y al efecto sombra (*Shadowing*). Este efecto es producido por objetos grandes como edificios, terrenos intermedios y vegetación, que provocan la pérdida de la trayectoria entre el transmisor y el receptor.

Por otro lado, se tiene el desvanecimiento a pequeña escala que se refiere a la variación significativa de los niveles de la señal debido a la interferencia de señales multitrayectoria cuando la estación móvil se desplaza en distancias cortas. Dependiendo de la extensión de la multitrayectoria, se puede caracterizar al canal como selectivo en frecuencia (*Frequency-selective*) o de frecuencia plana (*Frequency flat*). Mientras que, según la variación que se tenga en tiempo, relacionado a la velocidad del móvil y por ende al efecto Doppler, se puede tener desvanecimiento rápido (*Fast Fading*) o desvanecimiento lento (*Slow Fading*). La Figura 1.7 muestra la clasificación de los canales de desvanecimiento mencionada anteriormente [3].

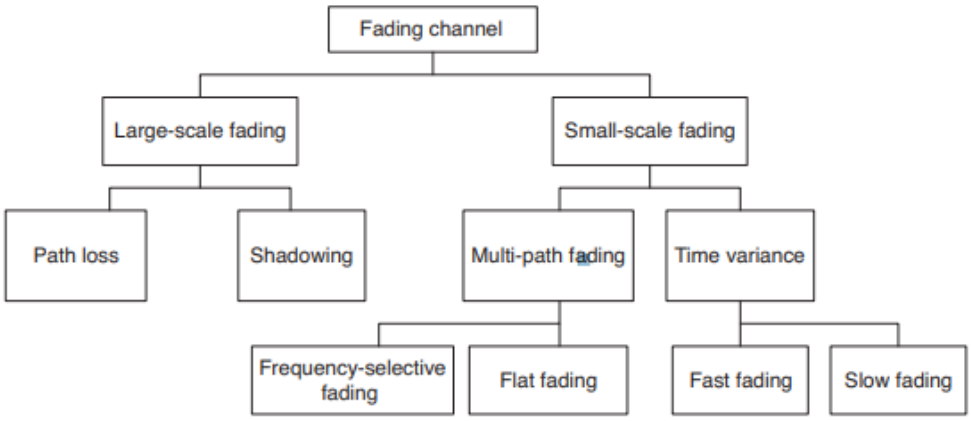

Figura 1.7 Clasificación de los canales de desvanecimiento [3]

1.3.1.4. Efecto Doppler

Cuando se habla de un canal variante en el tiempo, es importante considerar que la frecuencia de la señal transmitida será diferente a la frecuencia de la señal que se recibe. Esto se debe al movimiento que existe entre el receptor y el transmisor, escenario característico de un canal variante en el tiempo. Este desplazamiento en frecuencia que se produce, es conocido como desplazamiento Doppler (*Doppler Shift*).

La frecuencia recibida puede resultar mayor o menor a la frecuencia transmitida, dependiendo del movimiento que se produce entre el transmisor y el receptor. Si se tiene que el transmisor y el receptor se mueven acercándose entre sí, la frecuencia de la señal recibida resultará mayor a la frecuencia de la señal transmitida. En cambio, en el caso de que el transmisor y el receptor se muevan en direcciones opuestas alejándose uno del otro, la frecuencia de la señal recibida resultará menor a la frecuencia de la señal transmitida. Este efecto puede calcularse mediante [11].

$$\nabla \mathrm{f}_c = \pm f_o \frac{\nabla_v}{c} \tag{1.1}$$

donde:

- $\nabla \mathrm{f}_c$ es la diferencia que existe entre la frecuencia recibida y transmitida;
- f_o es la frecuencia de la señal transmitida;
- ∇_v es la diferencia de velocidad que existe entre el transmisor y el receptor;
- c es la velocidad de la luz.

1.3.1.5. Interferencia Interportadora - ICI

Como se ha mencionado antes, al considerarse un canal variante en el tiempo donde el transmisor o el receptor se encuentran en movimiento, se tiene desplazamiento Doppler. Esto produce la pérdida de la ortogonalidad de las subportadoras, lo cual ocasiona interferencia interportadora (*Inter-Carrier Interference - ICI)*. La ICI es la presencia de un desplazamiento de fase en las subportadoras. Como se puede ver en la Figura 1.8, se pierde la ortogonalidad debido a este desplazamiento de fase.

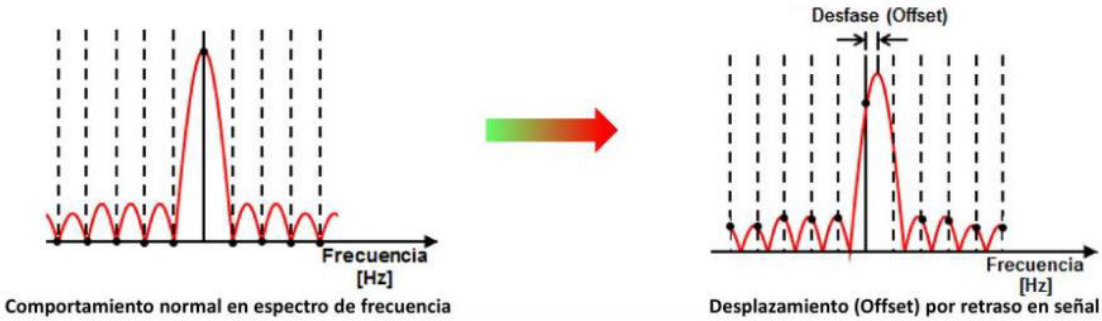

Figura 1.8 Interferencia Interportadora [12]

1.3.2. ESTÁNDAR IEEE 802.11p

El Instituto de Ingenieros Eléctricos y Electrónicos (*Institute of Electrical and Electronics Engineers* – IEEE) ha elaborado una serie de estándares para el Acceso Inalámbrico en

Ambientes Vehiculares (*Wireless Access in Vehicular Environment* - WAVE) que son considerados como las tecnologías más prometedoras para las redes vehiculares. El diseño de la capa física de WAVE está basado en el estándar IEEE 802.11. Sin embargo, considerando el ambiente de operatividad de las redes vehiculares, se hizo una modificación del estándar conocida ahora como estándar IEEE 802.11p [13].

De manera similar al protocolo IEEE 802.11a, el estándar IEEE 802.11p emplea la técnica OFDM para transmisión. Utiliza también una taza de datos de entre 3 a 27 Mbps a una distancia máxima de 1000 metros. Este estándar opera en la banda de 5 GHz. A continuación, en la Tabla 1.5 se encuentran detallados todos los parámetros principales de este estándar [14].

Tabla 1.5 Parámetros del estándar IEEE 802.11p

Parámetro	**Valor**			
Modulaciones	BPSK	QPSK	16-QAM	64-QAM
Factor de normalización	1	1/√2	1/√10	1/√42
No. de subportadoras de datos	48			
No. de subportadoras piloto	4			
No. de subportadoras null	12			
No. de subportadoras total	64			
Tamaño de FFT/IFFT	64			
Tamaño del PC	1.6 [us]			
Duración del símbolo OFDM	8 [us]			
Ancho de banda	10 [MHz]			

1.3.3. OFDM

La Multiplexación por División Ortogonal de Frecuencia es un caso particular de transmisión multiportadora. Este esquema de transmisión emplea la transmisión de datos en paralelo. Consiste en la transmisión de un flujo de datos sobre cierto número de subportadoras de baja velocidad. Esto representa una gran ventaja frente a los sistemas que emplean una única subportadora ya que, con un solo desvanecimiento o interferencia, se caería todo el enlace. Mientras que, al emplear múltiples subportadoras solo un pequeño porcentaje de estas se vería afectado y el enlace continuaría levantado.

OFDM se considera una técnica de multiplexación que se emplea para aumentar la robustez de la señal frente a los efectos del desvanecimiento selectivo en frecuencia y la

interferencia de banda estrecha. En los sistemas clásicos de transmisión de datos en paralelo, la banda de frecuencia total de la señal se dividía en N subcanales de frecuencia que no se solapan entre sí. Esto parecía ser muy eficiente para eliminar la ICI, sin embargo, se tenía un uso ineficiente del espectro disponible. Para contrarrestar esta ineficiencia, surgieron ideas para implementar datos de forma paralela y la técnica de multiplexación por división de frecuencia (FDM – *Frequency Divisió Multiplexing*) con subcanales que se solapan entre sí. La Figura 1.9 muestra la diferencia entre la técnica convencional de subportadoras que no se solapan comparada con la técnica de modulación de subportadoras que sí se solapan. Como se puede observar, se ahorra al menos un 50% del ancho de banda.

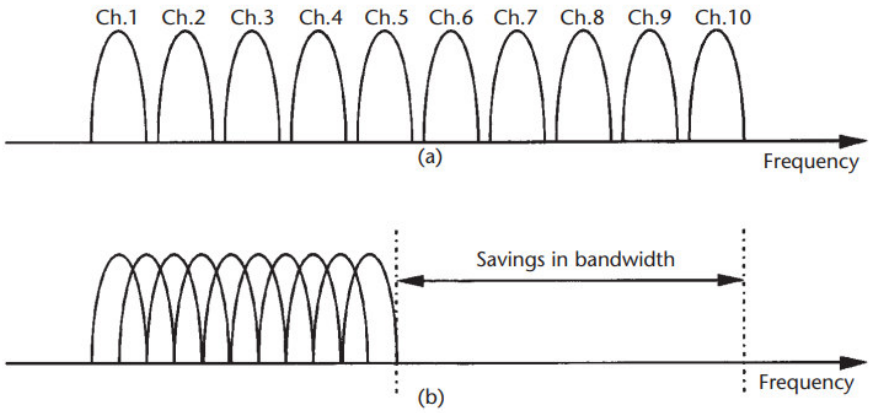

Figura 1.9 Concepto de una señal OFDM: (a) técnica convencional multiportadora, y (b) técnica de modulación ortogonal multiportadora [15]

Sin embargo, para implementar esta técnica se necesita también reducir el *cross talk*, para lo cual se debe implementar la ortogonalidad entre las diferentes portadoras moduladas. Esta característica de ortogonalidad, indica que existe una relación matemática entre las diferentes frecuencias de las portadoras que conforman el sistema. De este modo, se puede compactar las subportadoras en una señal OFDM de forma que las bandas de guarda de las subportadoras se solapen y aun así las señales sean recibidas sin interferencia interportadora adyacente [15].

1.3.3.1. Efectos de un canal variante en tiempo sobre OFDM

OFDM divide su ancho de banda de transmisión en varios subcanales que serán transmitidos en paralelo. Esto ocasiona que la duración del símbolo incremente ocasionando que OFDM sea susceptible frente a los efectos ocasionados por el desplazamiento Doppler. Dichos efectos degradan la eficiencia de OFDM destruyendo la

ortogonalidad entre subportadoras y produciendo como resultado el ICI [16]. Se ha demostrado que la potencia del ICI depende fundamentalmente del producto entre la frecuencia máxima de Doppler y la duración de un símbolo OFDM [17].

Existen algunos métodos para la detección del canal, sin embargo, es importante tomar en cuenta si las características del canal cambian para cada periodo de un símbolo OFDM. Cuando el transmisor y el receptor de un canal se mueven rápidamente, se producen cambios en el canal, en donde el periodo largo de un símbolo OFDM tiene un efecto más severo en el desempeño de la detección del canal. Un canal que es variante en el tiempo puede destruir la ortogonalidad entre las subportadoras en el receptor, produciendo así ICI. El efecto del ICI no puede ser compensado con el ecualizador convencional one-tap [3].

Una señal OFDM que se transmite, puede ser representada en el dominio del tiempo como [3]:

$$x[n] = \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}, \qquad n = 0, 1, \cdots, N-1 \tag{1.2}$$

donde:

- N es el número de subportadoras (tamaño de la FFT);
- $x[n]$ es la señal transmitida;
- $X[k]$ es la señal en el dominio de la frecuencia de la subportadora k.

La señal recibida mediante un canal inalámbrico por medio de L trayectorias se puede expresar como [3]:

$$y[n] = \sum_{i=0}^{L-1} h_i[n] x[n - \tau_i] + w[n] \tag{1.3}$$

de donde $h_i[n]$ representa la respuesta impulsiva, τ_i representa el retardo para la i-ésima trayectoria del canal variante en el tiempo y $w[n]$ es el ruido AWGN (*Additive White Gaussian Noise*). La señal recibida en el dominio de la frecuencia se obtiene mediante la FFT (*Fast Fourier Transform*) de $\{y[n]\}$, así [3]:

$$Y[k] = \frac{1}{N} \sum_{n=0}^{N-1} y[n] e^{-j2\pi kn/N}$$

$$= \sum_{n=0}^{N-1} \sum_{i=0}^{L-1} X[m] H_i[k-m] e^{-\frac{j2\pi im}{N}} + W[k], \qquad k = 0, 1, \cdots, N-1 \tag{1.4}$$

donde $W[k]$ es la FFT de $\{w[n]\}$ que es el ruido aditivo AWGN y $H_i[k]$ denota la FFT de la respuesta impulsiva $\{h_i[n]\}$. De este modo, $H_i[k]$ se puede representar como [3]:

$$H_i[k] = \frac{1}{N} \sum_{n=0}^{N-1} h_i[n] e^{-j2\pi kn/N} \tag{1.5}$$

Adicionalmente, la ecuación (1.5), que representa el efecto del canal, se puede representar mediante una matriz H conocida como la matriz del canal. De este modo, la ecuación (1.4) puede ser expresada como:

$$Y = HX + W \tag{1.6}$$

donde:

- Y es el vector de la señal recibida;
- H es la matriz del canal en el dominio de la frecuencia;
- X es el vector de la señal transmitida;
- W es el vector del ruido.

Si la respuesta del canal $h_i[n]$ se mantiene constante en lo que dura el periodo de un símbolo OFDM, la matriz del canal H se convertirá en una matriz diagonal teniendo que $a_{k,m} = 0, \ \forall \ k \neq m$. De ser este el caso, la señal transmitida puede recuperarse de manera sencilla empleando un ecualizador simple así:

$$X = H^{-1} Y \tag{1.7}$$

Sin embargo, cuando se trata de un canal variante en el tiempo, $h_i[n]$ varía en lo que dura el periodo del símbolo OFDM, haciendo que H no sea una matriz diagonal, lo cual complica la resolución de la ecuación (1.6) [3].

Con el objetivo de simplificar las expresiones mencionadas anteriormente, podemos escribir una expresión similar a la ecuación (1.2). Por lo que las señales OFDM en el dominio del tiempo también pueden escribirse como:

$$x_n = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} s_k e^{j2\pi \frac{kn}{N}} \ , -N_p \leq n \leq N \tag{1.8}$$

Donde N es el número de subportadoras, N_p es la longitud del prefijo cíclico, x_n es la señal de la subportadora n antes de ser transmitida, y s_k es la subportadora k [18].

Una forma más sencilla de representar a la ecuación (1.8) se la puede escribir tomando en cuenta que x es la señal en el dominio del tiempo generada en el transmisor, F es la matriz de la transformada discreta de Fourier (DFT) y s es el vector formado por los elementos del símbolo OFDM. De modo que x se obtiene al multiplicar la transpuesta hermitiana de F por s [8].

$$x = F^H s \tag{1.9}$$

Tomando la ecuación (1.6) y llevándola al dominio del tiempo. Al reemplazar la ecuación (1.9) en la ecuación resultante se obtiene la ecuación de la señal recibida.

$$y = hF^H s + w \tag{1.10}$$

Para pasar la señal recibida y al dominio de la frecuencia, se debe multiplicar a toda la ecuación por F, teniéndose que:

$$Fy = FhF^H s + Fw \tag{1.11}$$

dado que en la ecuación (1.11) s es el vector de elementos del símbolo OFDM y w es el ruido, entonces el canal puede ser representado mediante la ecuación (1.12) en el dominio de la frecuencia [8].

$$H = FhF^H \tag{1.12}$$

1.3.4. TÉCNICAS DE DETECCIÓN DE LA SEÑAL

La detección, también es conocida como ecualización. El objetivo de emplear detección en el receptor es recuperar los símbolos enviados por el transmisor. OFDM emplea comúnmente el detector one-tap ya que este funciona correctamente cuando se trata de canales estáticos en el tiempo. Sin embargo, para canales variantes en el tiempo, el detector one-tap ya no tiene un buen rendimiento.

Existen dos tipos de detectores: detectores lineales y detectores no lineales.

- **Detección lineal**

Los detectores lineales presentan una complejidad reducida en términos de su implementación y ejecución. Para facilitar la detección de la señal deseada, se invierte el efecto del canal mediante la matriz W de pesos. Los métodos estándar de detección lineal

son la técnica de One-tap, ZF (*Zero-Forcing*) y la técnica de MMSE (*Minimum Mean Square Error*) [3].

- **Detección no lineal**

  Los detectores no lineales presentan una complejidad mayor a la de los detectores lineales. Para obtener la matriz del canal realizan operaciones más complejas como lo es la descomposición QR y derivados de la misma. Algunos de los métodos de detección no lineal son QRD (*QR Decomposition*), SQRD (*Sorted QR Decomposition*) y MMSE-SQRD.

1.3.4.1. Técnica de Detección One-Tap

Cuando no se presenta una variación de tiempo en el canal, la respuesta impulsiva del canal (CIR-*Channel Impulsive Response*) es constante a lo largo del tiempo, por lo cual la matriz del canal en el dominio del tiempo se vuelve circulante y constante para todos los bloques OFDM. En este escenario, el prefijo cíclico no solamente se encarga de eliminar la interferencia intersímbolo (ISI-*Intersymbol Interference*) sino que también induce una convolución circular en el dominio del tiempo de la señal transmitida con CIR. Como consecuencia, la matriz del canal en el dominio de la frecuencia es una matriz diagonal [19].

Al tenerse una matriz diagonal es muy sencillo aplicar el detector one-tap. Para realizar la estimación de los símbolos transmitidos simplemente se divide los símbolos recibidos para la matriz del canal o en su defecto, multiplicar los símbolos recibidos por la inversa de la matriz del canal [3]. Considerando que la inversa de una matriz diagonal se forma como otra matriz diagonal con los inversos de la diagonal principal, el cálculo se vuelve mucho más sencillo.

Sin embargo, cuando se trata de un escenario con un canal variante en el tiempo, ya no se puede aplicar los detectores convencionales como lo es el one-tap. Esto se debe a que este detector no toma en cuenta el ICI generado por la variación en tiempo del canal. Además, para este caso la matriz del canal ya no es una matriz diagonal, por lo que el cálculo ya no es tan sencillo como se mencionaba anteriormente.

1.3.4.2. Técnica de Detección Zero Forcing (ZF)

La técnica ZF anula la interferencia empleando de forma general la siguiente matriz de pesos:

$$W_{ZF} = (H^H H)^{-1} H^H \tag{1.13}$$

Donde H representa la matriz del canal que puede o no ser una matriz cuadrada y $(.)^H$ denota la operación de transposición Hermitiana. Para el caso en el que la matriz H del canal sea una matriz cuadrada, la ecuación general se simplifica y se reduce a lo siguiente:

$$W_{ZF} = \mathrm{H}^{-1} \tag{1.14}$$

Para obtener el vector de símbolos transmitidos $\tilde{x}_{ZF}$, se multiplica la matriz de pesos mencionada anteriormente por el vector de símbolos recibidos y como se muestra en la siguiente ecuación: [3]

$$\tilde{x}_{ZF} = W_{ZF} y \tag{1.15}$$

1.3.4.3. Minimum Mean Squared Error (MMSE)

Con el propósito de maximizar la relación señal/interferencia & ruido (SINR) en la etapa post-detección, la matriz de pesos MMSE está dada por [20]:

$$W_{MMSE} = (H^H H + \sigma_z^2 I)^{-1} H^H \tag{1.16}$$

La detección MMSE permite reducir el impacto del ruido de fondo. Como se puede notar, el detector MMSE es similar al detector ZF, pero la diferencia está en que éste utiliza además la varianza del ruido representada por σ_z^2. Para obtener este último parámetro mencionado, es importante recordar la relación que tiene el SNR (*Signal-to-Noise Ratio*) con el ruido. Esta relación se presenta en la ecuación (1.17):

$$SNR_{veces} = \frac{S}{N} \tag{1.17}$$

donde S es la potencia de la señal transmitida antes de entrar al canal donde se le adiciona el ruido y N es la potencia del ruido. El cálculo del SNR se realiza mediante la ecuación (1.18)

$$SNR_{dB} = Eb/No + 10log_{10}\left(k\frac{subp.piloto + subp.de\ datos}{Nfft}\right) \tag{1.18}$$

donde:

- Eb/No es la relación energía por bit respecto de la densidad espectral de potencia del ruido;
- k número de bits por símbolo;
- $subp.piloto$ número de subportadoras piloto;

- $subp.\ de\ datos$ número de subportadoras de datos;
- $Nfft$ tamaño de la FFT.

de donde se puede obtener el SNR en veces, como se muestra en la ecuación (1.19):

$$SNR_{veces} = 10^{\frac{SNR_{dB}}{10}} \tag{1.19}$$

Pero para obtener la varianza del ruido se requiere información estadística, por lo que para este proyecto de titulación se realizó un cálculo del promedio de las potencias con cien iteraciones. De donde se obtuvo como potencia de una subportadora el valor de 0.0127, que multiplicado por las 64 subportadoras resulta en una potencia de 0.8128 para cada símbolo OFDM. Una vez calculado el SNR_{veces} y la potencia S, se puede obtener la varianza del ruido simplemente despejando N de la ecuación (1.17).

Ya que se obtiene la varianza del ruido, la señal transmitida se puede obtener empleando la matriz de pesos del detector MMSE como se muestra a continuación [20]:

$$\tilde{x}_{MMSE} = W_{MMSE} y \tag{1.20}$$

1.3.4.4. Sorted QR Decomposition (SQRD)

El algoritmo SQRD es una variante de la descomposición QR. La matriz del canal puede ser expresada como:

$$H = QR \tag{1.21}$$

donde Q es una matriz ortogonal y unitaria, de modo que:

$$(Q^H Q) = I \tag{1.22}$$

y R es una matriz triangular superior.

El vector de símbolos recibidos se puede expresar como:

$$u = Hx + z \tag{1.23}$$

Reemplazando la ecuación (1.21) en la ecuación (1.23), se tiene:

$$u = QRx + z \tag{1.24}$$

Al multiplicar la ecuación (1.24) por Q^H, la transpuesta Hermitiana de Q, esta expresión se reduce a:

$$y = Q^H u = Rx + v \tag{1.25}$$

Donde:

u Vector de símbolos recibidos

H Matriz del canal

x Vector de símbolos transmitidos

z Vector de ruido blanco gaussiano

y Señal de recepción

v Representa a $Q^H z$

El k-ésimo elemento de y corresponde a la capa más baja y se puede expresar como:

$$y_k = r_{k,k} x_k + v_k + d_k \tag{1.26}$$

donde d_k es la interferencia definida como:

$$d_k = \sum_{i=k+1}^{N_T} r_{k,i} x_i \tag{1.27}$$

Como se puede ver, d_k es dependiente de los símbolos de las capas superiores $x_{1,} \dots, x_{k-1}$.El k-ésimo símbolo transmitido se obtiene considerando una interferencia nula. Por lo cual la señal transmitida y cuantizada para la subportadora N_T se obtiene despejando de (1.26), aplicando la operación de cuantización $Q[.]$, así:

$$\hat{x}_{N_T} = Q\left[\frac{y_{N_T}}{r_{N_T,N_T}}\right] \tag{1.28}$$

Mientras que los demás símbolos transmitidos se calculan de forma regresiva empleando la interferencia diferente de cero y el elemento cuantizado del símbolo anterior. De este modo, los demás elementos cuantizados se calculan, despejando de (1.26) así:

$$\hat{x}_k = Q\left[\frac{y_k - d_k}{r_{k,k}}\right] \tag{1.29}$$

Una vez obtenido el símbolo cuantizado, la interferencia se calcula empleando $\hat{x}_i$. Por lo cual se reformula la ecuación (1.27), teniéndose que:

$$d_k = \sum_{i=k+1}^{N_T} r_{k,i}\hat{x}_i \tag{1.30}$$

El orden de detección de los símbolos es crucial dado que la descomposición QR es propensa a la propagación de errores. Esto se debe a que, si un error se genera en una capa inferior, este se propagara a las capas superiores lo que ocasiona que se incremente la probabilidad de error. El algoritmo SQRD se basa en el algoritmo modificado Gram-Schmidt el cual calcula la matriz R por filas desde la parte superior hacia la inferior. La matriz Q se calcula por columnas comenzando en la parte izquierda y empezando por establecer que $Q = H$. El algoritmo SQRD ordena las columnas de Q en base a su norma[1], seleccionado la que posee la menor norma. Este ordenamiento permite reducir el riesgo de una propagación de errores. El seudocódigo del algoritmo se muestra a continuación [5] [21].

- **Algoritmo para el detector SQRD**

1. $\boldsymbol{R = 0, Q = H, p = (1, \dots, N_T)}$
2. $\boldsymbol{for}\ i = 1, \dots, \boldsymbol{N_T}\ \boldsymbol{do}$
3. $\quad \boldsymbol{norm_i = \left|\left|q_i\right|\right|^2}$
4. $\boldsymbol{end\ for}$
5. $\boldsymbol{for}\ i = 1, \dots, \boldsymbol{N_T}\ \boldsymbol{do}$
6. $\quad \boldsymbol{k_i = arg\ min_{l=i,\dots,col_s} norm_l}$
7. $\quad exchange\ columns\ i\ y\ k_i\ in\ \boldsymbol{R, Q, norm, p}$
8. $\quad \boldsymbol{r_{i,i} = \sqrt{norm_i}}$
9. $\quad \boldsymbol{q_i \coloneqq q_i / r_{i,i}}$
10. $\quad \boldsymbol{for}\ l = i + 1, \dots, N_T\ \boldsymbol{do}$
11. $\quad\quad \boldsymbol{r_{i,l} = q_i^H q_l}$
12. $\quad\quad \boldsymbol{q_l \coloneqq q_l - r_{i,l} q_i}$
13. $\quad\quad \boldsymbol{norm_l \coloneqq norm_l - \left|\left|r_{i,l}\right|\right|^2}$
14. $\quad \boldsymbol{end\ for}$
15. $\boldsymbol{end\ for}$
16. $\boldsymbol{y = Q^H u}$
17. $\boldsymbol{for}\ k = N_T, \dots 1\ \boldsymbol{do}$

[1] La norma de un vector se obtiene sacando la raíz cuadrada de la suma de los cuadrados de todos sus componentes [34].

18. $\hat{d} = \sum_{i=k+1}^{N_T} r_{k,i}\hat{x}_i$

19. $\hat{x}_k = \boldsymbol{Q}[(y_k - \hat{d})/r_{k,k}]$

20. $\boldsymbol{end\ for}$

21. $Permutate\ \hat{x}\ according\ to\ \boldsymbol{p}$

1.3.4.5. Detector MMSE-SQRD

El algoritmo MMSE-SQRD es similar al algoritmo anteriormente mencionado, SQRD. Sin embargo, este algoritmo introduce la desviación estándar del ruido como parte del cálculo.

Para este algoritmo se tiene definida a $\underline{H}$ como la matriz extendida del canal, de la siguiente forma:

$$\underline{H} = \begin{pmatrix} H \\ \sigma_z I_{N_T} \end{pmatrix} \tag{1.31}$$

donde σ_z representa a la desviación estándar del ruido en la ecuación anterior. Del mismo modo, se define al vector extendido de símbolos recibidos del siguiente modo:

$$\underline{u} = \begin{pmatrix} u \\ 0_{N_T,1} \end{pmatrix} \tag{1.32}$$

Para este algoritmo, la matriz extendida del canal $\underline{H}$ se obtiene agregando a la matriz del canal la desviación estándar del ruido como se muestra en la ecuación (1.31). De este modo, similar al algoritmo SQRD, los parámetros: y, el k-ésimo elemento de y y d_k pueden calcularse del mismo modo presentando anteriormente.

Como se puede notar, la única diferencia que se presenta entre los algoritmos MMSE-SQRD y SQRD radica al emplear la matriz extendida $\underline{H}$ y el vector extendido $\underline{u}$ [21].

2. METODOLOGÍA

En el capítulo a continuación se describe el código del programa implementado para el desarrollo del canal variante en el tiempo, la técnica OFDM, así como cada uno de los detectores implementados. El programa implementado se realizó utilizando el software de simulación MATLAB, empleando particularmente la *Communications Toolbox* de dicho software. La mencionada *toolbox* posee algoritmos y aplicaciones propias para el análisis, diseño, simulación y verificación de sistemas de comunicaciones [22].

Por medio de los objetos que provee la toolbox se realiza la modulación y demodulación de datos empleando OFDM. Para ello se emplean las funciones *qammod*, *qamdemod*, *comm.OFDMModulator* y *comm.OFDMDemodulator*. Además de la implementación de un canal con distribución Rayleigh o Rician según corresponda al escenario que se va a simular. En los escenarios que no cuentan con línea de vista se emplea una distribución de Rayleigh. Para ello se emplea la función *comm.RicianChannel* considerando un *factor K* de cero.

Para el desarrollo de este proyecto técnico se implementó un transmisor, el canal inalámbrico variante en el tiempo considerando desvanecimiento de Rayleigh y Rician y el ruido AWGN, receptor con algoritmos de detección one-tap, ZF, MMSE, QRD, SQRD, MMSE-SQRD y por supuesto la generación de gráficas necesarias para el análisis de los resultados obtenidos. Para esto se tiene principalmente los siguientes scripts cuyos nombres son: OFDM_fading, Channel, Detector, Cancelador y Cuantificador. Pero también se implementaron funciones específicas para ciertos detectores como lo son las funciones one_tap, SQRD y MMSE-SQRD.

En la Figura 2.1 se puede apreciar el diagrama de flujo general de la simulación, en el cual se observa a breves rasgos el funcionamiento del programa principal. Del mencionado diagrama se pueden destacar tres secciones. En la primera sección se establecen los parámetros necesarios que caracterizan al escenario a partir del cual se realizará la simulación. Estos parámetros son el preámbulo, las subportadoras piloto, las funciones para el multiplexor y demultiplexor OFDM. Así como también la generación del canal, el SNR y la potencia de una subportadora. La segunda sección está conformada por dos bucles principales. Uno de ellos considera el número de iteraciones necesarias para obtener mejores resultados y el otro considera cada valor de SNR. Dentro de esta sección se realiza la transmisión de datos, se aplica el canal, se emplea el método de detección que se haya seleccionado y se realiza la recepción de la señal. En la tercera y última

sección se realiza el cálculo del BER y la gráfica de los resultados comparando el parámetro Eb/No con el BER.

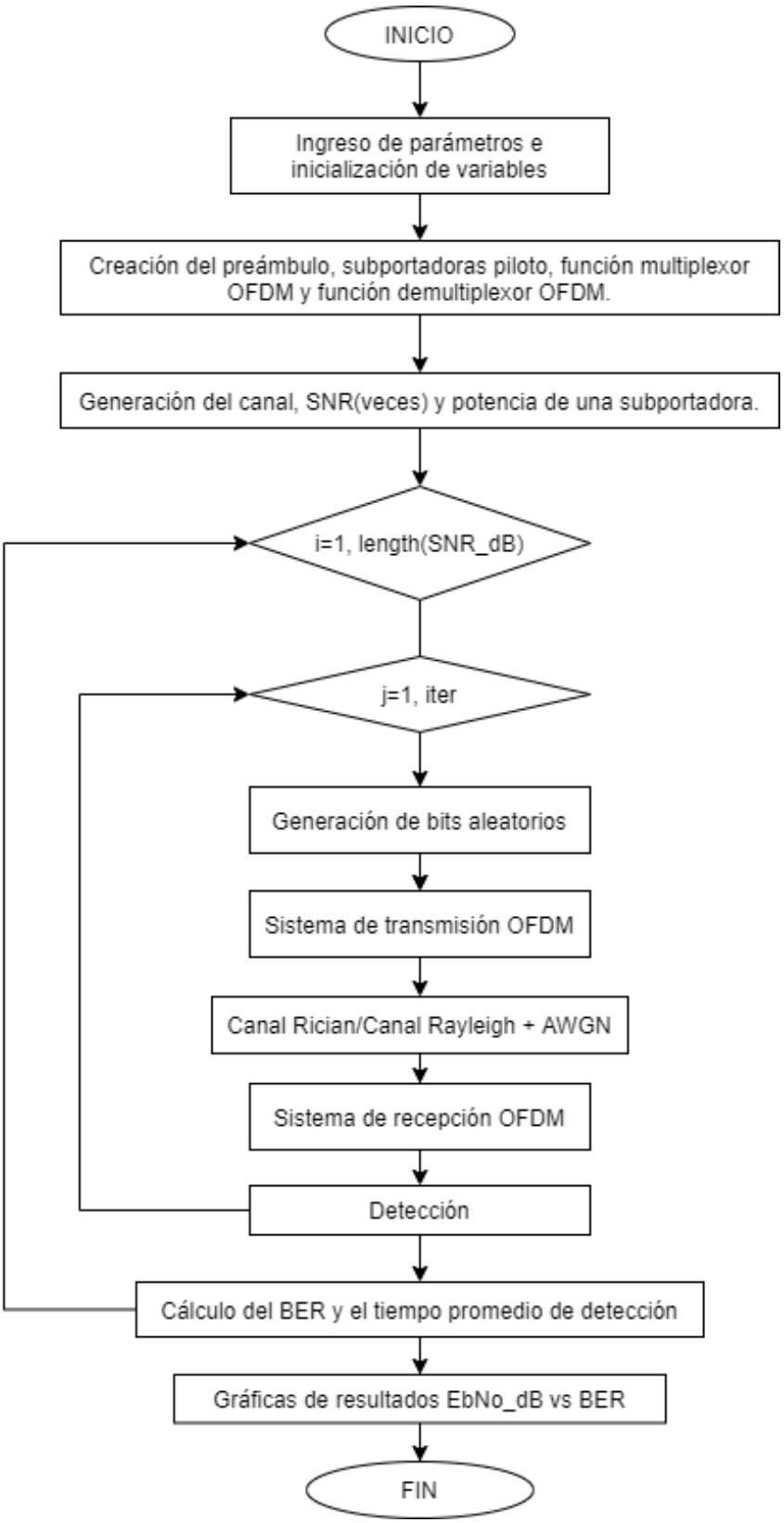

Figura 2.1 Diagrama de flujo general de la simulación

2.1. PROGRAMA PRINCIPAL

2.1.1. INICIALIZACIÓN DE VARIABLES

Al principio de todo el programa es necesario definir el valor de algunas variables en base a los parámetros seleccionados para la realización de este proyecto de titulación. En este caso se decidió trabajar con el estándar IEEE 802.11p en donde se tiene definido el número de subportadoras, los tipos de modulaciones que se pueden emplear, número de subportadoras de datos y número de subportadoras piloto.

Además, para este proyecto se ha decidido hacer la transmisión de 10 símbolos OFDM, así como 5000 iteraciones, número determinado como óptimo tras varias pruebas para una adecuada simulación y obtención de resultados. Dichos parámetros se los encuentra a continuación en la Tabla 2.1.

La parte del código donde se establecen estos valores se presenta en el Segmento de código 2.1.

```
clc, close all, clear all
% tic
Nfft=64;
iter=5000;
M =4;
k = log2(M);
EbNo_dB=0:2:25;
n_sym=10;
data_sub=48;
pilot_sub=4;
escenario=1;
D=6;
```

Segmento de código 2.1 Inicialización de variables

Tabla 2.1 Inicialización de variables

Nombre de la variable	Descripción	Valor
Nfft	Número de subportadoras	64
iter	Número de iteraciones	5000
k	Número de bits por símbolo	$log_2 M$
EbNo_dB	Intervalo de simulación	0 a 25 en pasos de 2
n_sym	Número de símbolos OFDM	10
data_sub	Número de subportadoras de datos	48
pilot_sub	Número de subportadoras piloto	4

2.1.2. INGRESO DE PARÁMETROS

Existen también ciertos parámetros que el usuario puede seleccionar al inicio del programa según el caso que se requiera simular. Estos parámetros son: el tipo de modulación, el escenario del canal de desvanecimiento y el detector. En la Tabla 2.2 se puede observar el nombre de dichos parámetros y las opciones posibles entre las cuales el usuario puede escoger.

La parte del código donde se establecen estos valores se presenta en el Segmento de código 2.1.

Tabla 2.2 Ingreso de parámetros

Nombre de la variable	Descripción	Opciones
M	Tamaño de la constelación	4(QPSK) 16(16QAM) 64(64QAM)
escenario	Tipo escenario	1, 2, 3, 4
D	Detector	1(Detector one-tap) 2(Detector Zero Forcing) 3(Detector MMSE) 4(Detector QRD) 5(Detector SQRD) 6(Detector MMSE-SQRD)

2.1.3. PREÁMBULO

Para la construcción del símbolo OFDM es necesario agregar el preámbulo al inicio. El preámbulo es el que marca el inicio de la trama al enviar los datos. Mediante el envío del preámbulo se sincroniza y entrena al receptor de modo que se pueda tener una demodulación adecuada y corregir el offset de tiempo [11].

El preámbulo está formado a partir de la repetición de secuencias conocidas. Se utiliza el mismo preámbulo para todos los símbolos transmitidos. Dentro de las secuencias que conforman al preámbulo se encuentran las secuencias cortas de entrenamiento conocidas como STS (*Short Term Symbol*) y también las secuencias largas de entrenamiento conocidas como LTS (*Long Term Symbol*) [23].

Para este proyecto se empleó la siguiente secuencia para el preámbulo:

$$preámbulo = [1,1,1,1,-1,-1,-1,1,-1,-1,-1,-1,1,1,-1,1,-1,-1,1,1,-1,1,1,-1,$$

$$1,1,1,1,1,-1,1,1,1,-1,1,1,-1,-1,1,1,1,-1,1,-1,-1,-1,1]$$

En el Segmento de código 2.2 se puede visualizar la definición del preámbulo en el código. Cabe recalcar que para la simulación implementada se considera sincronización perfecta por lo que el preámbulo es descartado en el receptor.

```
15 -    preambulo = [1,1,1,1,-1,-1,-1,1, -1,-1,-1,-1, 1,1,-1,1, -1,-1,1,1, -1,1,1,-1,...
16          1,1,1,1, 1,1,-1,1,1,1,-1,1, 1,-1,-1,1, 1,1,-1,1, -1,-1,-1,1];
17 -    pilotos=repmat([1;1;1;1],1,n_sym+1);
```

Segmento de código 2.2 Definición del preámbulo y las subportadoras piloto

2.1.4. PILOTOS

Para el presente trabajo, las subportadoras piloto se implementan a partir de una repetición de una secuencia de unos de tamaño 4x11. Siendo la dimensión de las filas el número de subportadoras piloto que se utilizan según el estándar y la dimensión de las columnas el número de símbolos aumentado en uno. Esto se puede observar en el Segmento de código 2.2.

Como se describe en la sección 1.3.2, respecto del estándar IEEE 802.11p, se tendrán 4 subportadoras piloto. En la Figura 2.2 se indica la ubicación correspondiente a las subportadoras de datos OFDM y las subportadoras piloto en el dominio del tiempo después de salir de la IFFT [24].

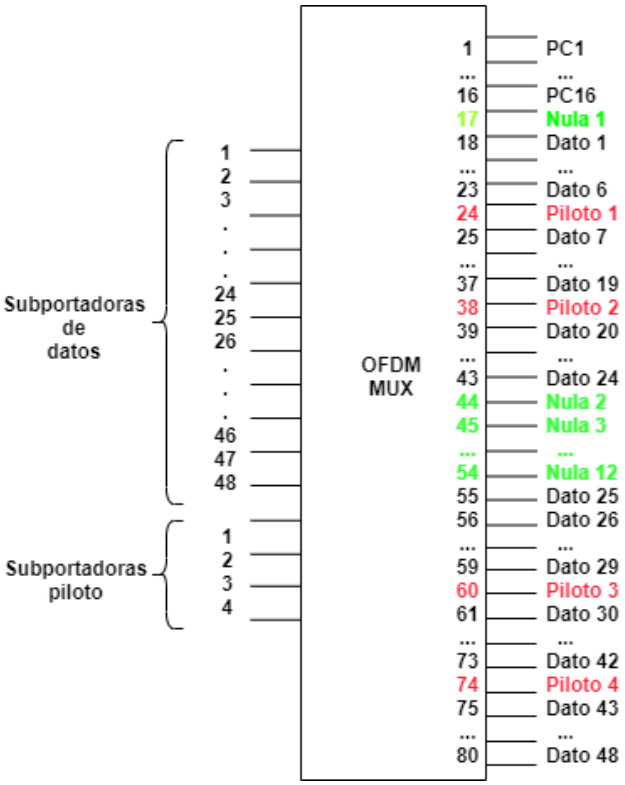

Figura 2.2 Entradas y salidas de la IFFT

Dado que para la multiplexación de los datos se aplica la IFFT, las posiciones de las subportadoras piloto se traducen en las posiciones 24, 38, 60 y 74. Esto se debe a que el comando comm.OFDMModulator construye el símbolo OFDM ubicando primero las 16 subportadoras del prefijo cíclico, luego una subportadora nula que corresponde a la componente DC, luego reordena el espectro ubicando primero las 24 subportadoras de datos del espectro negativo, luego la parte positiva del espectro intercalados con las subportadoras piloto y en el medio las 11 subportadoras nulas. Las 80 subportadoras se mapean según lo indicado en la Figura 2.2[25].

2.1.5. MULTIPLEAXIÓN OFDM Y DEMULTIPLEXACIÓN OFDM

Para el desarrollo de este trabajo se utilizó la *Communications Toolbox* donde se incluyen objetos que permiten implementar el multiplexor OFDM y el demultiplexor OFDM. Dichos objetos son: *comm.OFDMModulator* y *comm.OFDMDemodulator*. El multiplexor OFDM aplica la IFFT a la señal transmitida y agrega todas las características de OFDM para construir el símbolo OFDM como tal. Es decir, agrega las 12 subportadoras nulas, las 4 subportadoras piloto y la componente DC. Mientras que el demultiplexor OFDM realiza la FFT y retira las características de OFDM que agregó el multiplexor obteniéndose netamente las 48 subportadoras de datos de la señal recibida.

Dentro de los parametros que se pueden configurar para el uso de este objeto, se tiene:

- *FFTLength*

  Este parámetro permite indicar el número de puntos de la Transformada Rápida de Fourier (*FFT-Fast Fourier Transform*) como un número entero positivo. El tamaño de la FFT equivale al número de subportadoras totales que se van a utilizar [26]. Para el presente proyecto y según el estándar IEEE 802.11p empleado en el mismo, se especifica este parámetro en 64 subportadoras.

- *PilotInputPort*

  Este parámetro acepta un número o un valor booleano, es decir un 0(*false*) o un 1(*true*). Si esta propiedad está configurada en 1(*true*), se puede asignar subportadoras individuales para la transmisión de los pilotos. Por otra parte, si se configura con 0(falso), se asume que la información de las pilotos viene embebida en los datos de entrada [26]. Para el presente proyecto y según el estándar IEEE 802.11p empleado en el mismo, se especifica este parámetro en *true* ya que sí se va a emplear subportadoras piloto.

- *PilotCarrierIndices*

  Cuando el parámetro *PilotInputPort* está configurado en 1(verdadero), se puede especificar los índices de las subportadoras piloto. Dichos índices se deben especificar en forma de vector columna. Para este proyecto se seleccionaron los índices [12 ;26; 40; 54].

- *InsertDCNull*

  Este parámetro acepta un número o un valor booleano, es decir un 0(*false*) o un 1(*true*) para indicar si se va o no a insertar la componente DC nula. La subportadora DC es el centro de la banda de frecuencia [26]. Para el presente proyecto se especificó este parámetro en *true* dado que sí se va a emplear una subportadora nula.

- *CyclicPrefixLength*

  Este parámetro especifica la longitud del prefijo cíclico, la cual puede expresarse como un entero positivo [26]. Para el presente proyecto se especificó este valor en 16 dado que el prefijo cíclico toma la cuarta parte de las 64 subportadoras que conforman el símbolo OFDM.

- *NumSymbols*

  Este parámetro especifica el número de símbolos OFDM que se va a transmitir, se lo especifica mediante un número entero positivo [26]. Para el presente proyecto se especificó este valor en *n_sym*+1, que resulta en 11 símbolos. Esto considerando que se transmitirán 10 símbolos OFDM de información y 1 símbolo del preámbulo al inicio de la secuencia.

```
18 -    OFDM_mux = comm.OFDMModulator('FFTLength',64, ...
19                                    'PilotInputPort',true, ...
20                                    'PilotCarrierIndices',[12; 26; 40; 54], ...
21                                    'InsertDCNull',true, ...
22                                    'CyclicPrefixLength',16, ...
23                                    'NumSymbols',n_sym+1);
```

Segmento de código 2.3 Implementación del multiplexor OFDM

Como se puede observar en el Segmento de código 2.3, se crea el multiplexor OFDM con cada uno de los parámetros especificados. Una vez creado el multiplexor OFDM, se puede crear el demultiplexor OFDM a partir del multiplexor OFDM previamente definido, utilizando los mismos parámetros ya definidos.

```
24 -    OFDM_demux = comm.OFDMDemodulator(OFDM_mux);
25 -    [canal_fading,sampIdx]=Channel(escenario);
26
27 -    SNR_dB=EbNo_dB+10*log10(k*(pilot_sub+data_sub)/Nfft);
28 -    SNR_veces=(10.^(SNR_dB/10));
29 -    pot_s=0.0127*64;     %Potencia promedio
30
31 -    BER=zeros(1,length(SNR_dB));
32 -    elapsedT=zeros(1,iter);
33 -    iter_promT=zeros(1,length(SNR_dB));
```

Segmento de código 2.4 Implementación del demultiplexor OFDM, el canal y el SNR

En el segmento de código anterior se muestra también en la línea 25 la generación del canal, pero esta función se discutirá más a detalle en la sección 2.2. Además, se tiene en las líneas 27 a 29, el cálculo del SNR en decibelios, SNR en veces y la potencia promedio obtenida como se explicó previamente en la sección 1.3.4.3. En la línea 31 se designa el tamaño del vector en donde se almacenarán los valores del BER obtenido luego del número de iteraciones definido en la variable *iter*. Y finalmente, en las líneas 32 y 33 se define el tamaño de los vectores donde se almacena el tiempo de detección en *elapsedT* y el tiempo promedio de todas las iteraciones para un SNR determinado en el vector *iter_promT*.

2.1.6. BUCLE PRINCIPAL

Dentro del bucle principal se encuentra la esencia de todo este proyecto de titulación. El bucle involucra un *for* que va de 1 hasta el número de iteraciones. Para el presente proyecto se realizaron varias pruebas con diferente cantidad de iteraciones que fueron desde 100, 500, hasta decidir que el número adecuado de iteraciones sería 5000. Con dicho número de iteraciones se puede obtener unos resultados bastante claros en cuanto a las gráficas de los detectores para cada uno de los escenarios propuestos.

En la línea 36 del Segmento de código 2.5 se generan bits aleatorios que conformarán la señal de transmisión. Dicha señal es modulada mediante el comando *qammod* en el cual se especifica que los datos de entrada, *InputType,* son bits y que la señal de salida que se obtendrá tendrá una potencia promedio mediante el parámetro *UnitAveragePower*. Una vez modulada la señal, es necesario reorganizar la información que se va a transmitir, para lo cual se usa una matriz de 48x10, tomando en cuenta que se tienen 48 subportadoras de información y 10 símbolos OFDM. Sin embargo, es importante recordar que se debe agregar el preámbulo al inicio por lo que la matriz definitiva será de 48x11.

```
34 -  for i=1:length(SNR_dB)
35 -      for j=1:iter
36 -          data_bit = randi([0 1],48*k*n_sym,1);            %Generacion de bits
37 -          data_sym = qammod(data_bit,M,'InputType','bit','UnitAveragePower',true);
38 -          data_sym_mat=reshape(data_sym,data_sub,n_sym);
39 -          data_sym_mat=[preambulo',data_sym_mat];     %unir preambulo y datos
40 -          txSig = step(OFDM_mux,data_sym_mat,pilotos);    %se aplica OFDM
41 -          reset(canal_fading);                                %Resetear
42 -          [y,pathgains] = step(canal_fading,txSig);      %Filtrar la señal
```

Segmento de código 2.5 Implementación del bucle principal

La matriz mencionada ingresa al multiplexor OFDM previamente definido en conjunto con las subportadoras piloto y almacena el resultado en la variable *txSig*. Como se puede observar en la línea 41 es importante resetear el canal para garantizar que el canal cambie en cada iteración. Hasta aquí se tiene todo lo que corresponde al transmisor. Posteriormente, se ingresa la señal al canal, el mismo que estará definido en base al escenario que se seleccione al inicio, recordando que los escenarios propuestos para este proyecto están descritos en base a los PDP descritos en la sección 1.3.1.2.1.

En la línea 43 del Segmento de código 2.6 se realiza el cálculo de la potencia de la señal, valor que sirvió para definir la potencia de una subportadora, tal como se especifica en la sección 1.3.4.3. Luego se calcula el ruido a partir de la potencia y el SNR correspondiente. Como se conoce que el canal sufre de efectos como el ruido, se utiliza la función *awgn* para añadir ruido blanco a la señal para luego pasar a la etapa de recepción en donde se utiliza el demultiplexor OFDM creado anteriormente. Como siguiente se emplea la función *Detector*, la cual se aplica según el detector que se haya seleccionado al inicio de la simulación, que puede ser el detector one-tap, ZF, MMSE, QRD, SQRD o MMSE-SQRD. La función *Detector* será explicada con mayor detalle en la sección 2.3. Finalmente, se vuelve a acomodar la matriz obtenida luego de la detección en un arreglo de 480x1 para demodularlo mediante la función *qamdemod*.

Para poder observar el desempeño de los detectores se obtiene el BER mediante la función *biterr* de Matlab, valor que se irá acumulando en el vector BER para cada una de las iteraciones.

```
43      %         pot(j)=mean(abs(y.^2));    % Se obtiene la potencia de la senal antes de anadir el ruido
44 -        ruido=pot_s/SNR_veces(i);  % Se calcula el ruido
45 -        rxSig=awgn(y,SNR_dB(i),'measured');
46 -        rx_data_mat= step(OFDM_demux,rxSig);  % Proceso inverso OFDM
47 -        tic
48 -        rx_data_eq=Detector(rx_data_mat,rxSig,n_sym,pathgains,sampIdx,D, SNR_dB, y, ruido, M);
49 -        elapsedT(j)=toc; %Guardar en una matriz los valores promedio de tiempo que tarda el detector
50          %  disp(j)
51 -        rx_data=reshape(rx_data_eq,n_sym*data_sub,1);
52 -        rx_bit = qamdemod(rx_data,M,'OutputType','bit','UnitAveragePower',true);
53 -        [number,ratio] = biterr(data_bit,rx_bit);
54 -        BER(i)=BER(i)+ratio;
55 -    end
56      %     pot_prom(i)=mean(pot);
57 -       iter_promT(i) = mean(elapsedT);
58
59 -    BER(i)=BER(i)/iter;
60 - end
```

Segmento de código 2.6 Implementación del programa principal

A la salida del bucle principal, se realiza el cálculo de la potencia promedio y se hace un promedio del BER obtenido para el correspondiente SNR en la línea 56 del Segmento de código 2.6. También se obtiene el tiempo promedio de detección en la línea 57 de segmento de código anterior. Este tiempo de detección promedio corresponde a las 5000 iteraciones para un determinado valor de SNR. El bucle principal se encarga de realizar el cálculo para los 13 valores de SNR generados al inicio. Como se puede ver en el Segmento de código 2.7, las líneas finales del programa principal, son dedicadas netamente para graficar los resultados y almacenarlos para su posterior explicación en el Capítulo 3. Sin embargo, en el Segmento de código 2.7 se realiza también el cálculo del tiempo promedio de detección para los 13 valores de SNR.

```
61      % t_total=toc;
62 -    prom_T = mean(iter_promT);
63 -    semilogy(EbNo_dB,BER)
64 -    ylim([0.0001 0.5])
65 -    xlim([0 EbNo_dB(end)])
66 -    grid on
67 -    xlabel('E_b/N_0 [dB]');
68 -    ylabel('BER')
69
70      % Para graficar
71 -    e=num2str(escenario);
72 -    D=num2str(D);
73 -    M=num2str(M);
74
75 -    filenm1= sprintf('Resultado escenario %s detector %s modulac %s.mat', e, D, M);
76 -    save(filenm1, 'EbNo_dB','BER');
77
78 -    filenm2= sprintf('Tiempo escenario %s detector %s modulac %s.mat', e, D, M);
79 -    save(filenm2, 'prom_T');
```

Segmento de código 2.7 Obtención de la gráfica y resultado de la simulación

2.2. FUNCIÓN CHANNEL

Esta función esta creada con el fin de seleccionar el canal y establecer los parámetros bajo los cuales se quiere regir al mismo. La única entrada de esta función es el valor *C*, mediante el cual se selecciona el tipo de escenario que puede ser: *Rural* LOS, *Urban Approaching* LOS, *Highway* LOS o *Highway* NLOS. Como se muestra en el Segmento de código 2.8, las salidas de esta función son el canal generado bajo los criterios del escenario seleccionado y las posiciones de los retardos.

```
function [ canal,sampIdx ] = Channel( C )
% Seleccionar del canal y establecer parámetros
% C             Opcion del canal seleccionado
% sampIdx       Retardos normalizados
% canal         Canal generado
% Pot           Potencia en dB de cada rayo.
% Delay         Retardo de cada rayo.
% DP            Desplazamiento Doppler de cada rayo.
% El parámetro FactorR tiene el siguiente criterio:
% alto - pure Rician(DIRECTO)
% cero - pure Rayleigh(Reflexiones).
```

Segmento de código 2.8 Función *Channel*

Estos escenarios están detallados en la sección 1.3.1.2.1 y tienen sus correspondientes valores de potencia, retardo y efecto Doppler para cada uno de los rayos en la Tabla 1.1, Tabla 1.2, Tabla 1.3 y Tabla 1.4. En el Segmento de código 2.9, dentro de cada consideración del comando *case*, se asignan estos parámetros en los vectores *Pot*, *Delay* y *DP*. Adicional a cada escenario se ha definido el vector *FactorR*, el mismo que identifica si la transmisión de cada rayo es resultado de si existió línea de vista o si no se tuvo línea de vista y el rayo es una de las diferentes reflexiones de la señal. Es decir, se identifica si se trata de una distribución de Rician o de Rayleigh. Según sea el caso, se asigna un valor de cero o cercano a cero para el caso de ser una distribución Rayleigh o un valor en el orden de $1x10^{12}$ en caso de ser una distribución de Rician.

```
switch C
    case 1  %Canal = Rural-LOS
        Pot = [0 -14 -17];
        Delay = [0 100 200]*1e-9;
        DP = [0 492 -295];
        FactorR = [1e12 0 0];
    case 2  %Urban-LOS
        Pot = [0 -8 -10 -15];
        Delay = [0 100 200 300]*1e-9;
        DP = [0 236 -157 492];
        FactorR = [1e12 0 0 0];
    case 3  %Autopista-LOS
        Pot = [0 -10 -15 -20];
        Delay = [0 100 200 500]*1e-9;
        DP = [0 689 -492 886];
        FactorR = [1e12 0 0 0];
    case 4  %Autopista-NLOS
        Pot = [0 -2 -5 -7];
        Delay = [0 200 400 700]*1e-9;
%         DP = [0 689 -492 886];
        DP = [0 100000 -50000 150000];
        FactorR = [0.1 0 0 0];
end
```

Segmento de código 2.9 Selección del canal

2.2.1. ESPECTRO DOPPLER

Una vez obtenidas las características del escenario, en el Segmento de código 2.10, se muestra la obtención de la frecuencia máxima de Doppler en la línea 39. Este parámetro es importante para la posterior creación del espectro Doppler. Para esto, se emplea la función *Doppler* que permite construir el espectro especificando el tipo de espectro que se requiere.

```
% Establecer la frecuencia Doppler máxima
fd = max(abs(DP));
% Crear arreglo para el espectro Doppler
FED = cell(size(DP));
% Crear la forma del espectro Doppler
FED{1} = doppler('Asymmetric Jakes', [-.02 .02]);
for j = 2:length(DP)
    FED{j} = doppler('Asymmetric Jakes', sort([0 DP(j)/fd]));
end
```

Segmento de código 2.10 Implementación de la Frecuencia máxima y la forma del espectro Doppler

Entre las opciones de modelos que se pueden emplear se tiene el modelo asimétrico de Jakes que será explicado en la sección 2.2.2. Para emplear este modelo, se debe especificar, dentro de los parámetros de la función *Doppler*, el tipo de modelo y en este caso particular del modelo asimétrico de Jakes, se debe especificar la frecuencia mínima y máxima normalizadas.

En la línea 43 del Segmento de código 2.10 se crea la forma del espectro Doppler para el primer rayo. Considerando que, para todos los escenarios en la transmisión del primer rayo se considera que el transmisor y el receptor se encuentran estáticos. Por lo tanto, no se tiene efecto Doppler y la frecuencia seria nula. Sin embargo, se emplean los valores de -0.02 y 0.02 porque MATLAB especifica que se debe tener un espaciamiento mayor a 0.02.

Para el resto de los rayos se crea la forma del espectro en la línea 45 del Segmento de código 2.10. Aquí se considera una frecuencia mínima nula y como frecuencia máxima se obtiene la frecuencia máxima normalizada dividiendo la frecuencia de Doppler en cuestión para la frecuencia máxima que se obtiene en la línea 39.

2.2.2. MODELO DE JAKES

El modelo de Jakes es un método para simular el efecto Doppler que se presenta en las comunicaciones donde se tiene un canal variante en el tiempo. El funcionamiento de este modelo radica en asumir que al receptor llegan rayos de todas las direcciones que tienen un espaciamiento de $2\pi/N$ entre sí, cada uno con un desplazamiento Doppler diferente [27]. El modelo de Jakes se basa en la suma de señales sinusoidales y emplea la ecuación (2.1):

$$s(f) = \frac{1}{\pi f_d \sqrt{1-(f/f_d)^2}}\ , |f| \leq f_d \tag{2.1}$$

donde s es la forma del espectro, f es la frecuencia y f_d es la frecuencia máxima de Doppler [8] [28]. Sin embargo, se tiene también una variación del modelo de Jakes conocida como el espectro de Jakes asimétrico. Este último se diferencia del modelo de Jakes normal ya que permite especificar el rango de frecuencias del espectro. Para este modelo se usan las siguientes ecuaciones [28]:

$$s(f) = \frac{A_a}{\pi f_d \sqrt{1-(f/f_d)^2}}\ , -f_d \leq f_{min} \leq f \leq f_{max} \leq f_d \tag{2.2}$$

$$A_a = \frac{1}{\left[sin^{-1}\left(\frac{f_{max}}{f_d}\right) - sin^{-1}\left(\frac{f_{min}}{f_d}\right)\right]} \tag{2.3}$$

Donde f_{min} y f_{max} son las frecuencias mínima y máxima que conforman los extremos del rango de frecuencia.

2.2.3. CREACIÓN DEL CANAL

Para crear el canal se empleó la función *comm.RicianChannel*. Los parámetros de la Tabla 2.3 se configuraron para el uso de esta función. [8]

Tabla 2.3 Argumentos de la función *comm.RicianChannel()*

Argumento	Descripción	Valor
SampleRate	Frecuencia de muestreo de la señal de entrada	sRate
PathDelays	Retardos de cada trayecto	Delay
AveragePathGains	Ganancias promedio de cada trayecto	Pot
KFactor	Factor de Rician	FactorR
DirectPathDopplerShift	Desplazamiento Doppler de la trayectoria con línea de vista	DP
DirectPathInitialPhase	Fase inicial de la trayectoria con línea de vista	0
MaximumDopplerShift	Frecuencia máxima de Doppler	fd
DopplerSpectrum	Forma del espectro Doppler	FED
PathGainsOutputPort	Opción para habilitar la salida Path Gains	'true'

```
47      % Definir la frecuencia de muestreo según el AB del estándar
48 -    sRate = 10*1e6;
49      % Crear el canal
50 -    canal = comm.RicianChannel (...
51          'KFactor',                  FactorR, ...
52          'DirectPathDopplerShift',   DP, ...
53          'DirectPathInitialPhase',   zeros(size(FactorR)), ...
54          'DopplerSpectrum',          FED, ...
55          'SampleRate',               sRate, ...
56          'PathDelays',               Delay, ...
57          'AveragePathGains',         Pot, ...
58          'MaximumDopplerShift',      fd, ...
59          'PathGainsOutputPort',      true ...
60          );
61      % Calcular sampIdx
62 -    sampIdx = round(Delay/(1/sRate)) + 1;
63
64 -    end
```

Segmento de código 2.11 Cálculo de las posiciones de los retardos

Como parte final de la función *Channel* se tiene el cálculo de las posiciones de los retardos de cada uno de los rayos que se puede ver en la línea 62 del Segmento de código 2.11. Para esto, se toma el valor redondeado que resulta de la division de los retardos de cada

rayo en el vector *Delay*, para la fecuencia de muestreo. Considerando que la frecuencia de muestreo es equivalente al ancho de banda, es decir $10x10^6\,Hz$. Además, se ha incrementado en uno este cálculo dado que en Matlab se trabaja con posiciones de uno en adelante, sin tomar en cuenta la posicion cero.

2.3. FUNCIÓN DETECTOR

Esta función ejecuta el detector seleccionado según el valor de la variable *D*. Las entradas de esta función se pueden observar en la Tabla 2.4.

Tabla 2.4 Parámetros de la función *Detector*

Parámetros	Tipo de Dato	Descripción
rx_data_mat	Entrada	Matriz de símbolos recibidos después de OFDM
rxSig	Entrada	Matriz de símbolos recibidos antes de OFDM
n_sym	Entrada	Número de símbolos
pathgains	Entrada	Respuesta del canal
sampIdx	Entrada	Retardos normalizados de cada rayo
D	Entrada	Detector seleccionado
SNR_dB	Entrada	Parámetro SNR(dB)
y	Entrada	Señal recibida antes de considerar el ruido
ruido	Entrada	Varianza del ruido
M	Entrada	Tamaño de la constelación
data_det	Salida	Señal detectada

La función *Detector* se define de la siguiente forma, como se muestra en el Segmento de código 2.12.

```
1  function data_det = Detector(rx_data_mat,rxSig,n_sym,pathgains,sampIdx,D, SNR_dB, y, ruido, M)
2  % Esta funcion permite seleccionar el detector y obtener la senal transmitida
3   % rx_data_mat       Señal recibida despues del demodulador OFDM
4   % rxSig             Señal recibida considerando el ruido
5   % n_sym             Numero de simbolos OFDM
6   % pathgains         Respuesta del canal
7   % sampIdx           Retardos normalizados
8   % D                 Detector seleccionado
9   % SNR_dB            SNR (dB)
10  % y                 Señal recibida sin ruido
11  % ruido             Varianza del ruido
12  % M                 Tamaño de la constelacion
13  % data_det          Señal x(transmitida)/Señal detectada
```

Segmento de código 2.12 Función *Detector*

Mediante la variable *D* se realiza una comparación con un *if* para acceder al algoritmo correcto. Dentro de cada condicional se ejecuta el detector correspondiente o se llama a la función implementada para el detector. Sin embargo, es importante recalcar que la función *Detector* se divide en dos secciones debido a la consideración realizada para generar la matriz del canal *H*. En primera instancia se compara si D es 1, para lo cual se ejecuta una función adicional creada particularmente para el detector one-tap. Dentro de la función *Detector* se llama a la función *one_tap* como se muestra en la línea 18 del Segmento de código 2.13.

```
18 -    if D==1  %One tap
19 -       data_det=one_tap(rx_data_mat,n_sym,pathgains,sampIdx);
20
```

Segmento de código 2.13 Función *one_tap* dentro de la función *Detector*

En cambio, si 1<D<7, se realiza el cálculo de la matriz *h* en el dominio del tiempo, que será utilizada para los detectores ZF, MMSE, QRD, SQRD y MMSE-SQRD. Esta condición se muestra en la línea 21 del Segmento de código 2.14. Eximiendo al detector one-tap, se necesita demultiplexar manualmente la señal recibida en vez de ser pasada por el demultiplexor OFDM ya que este obtiene como salida las 80 subportadoras de datos de cada símbolo. Sin embargo, para los detectores ZF, MMSE, QRD, SQRD y MMSE-SQRDP se necesita considerar las 64 subportadoras sin el prefijo cíclico ya que se procesa la matriz *H* del canal obteniendo una matriz con dimensiones 64x64.

Por ello, se realiza la demultiplexación manualmente empleando las 64 subportadoras. Como se puede ver en el Segmento de código 2.14, se va obteniendo pedazos de *rxSig* sin tomar en cuenta las 80 primeras subportadoras que corresponden al preámbulo ni las 16 primeras subportadoras del primer símbolo que corresponden al prefijo cíclico, por lo que se empieza a partir de la subportadora 17. El límite considera en cambio las 64 subportadoras que conforman el símbolo, omitiendo la que ya está considerada en la última posición. Todo esto se ordena en un vector columna temporal llamado *temp_rx*. De este último se obtiene la FFT con el cual se arma la señal recibida y demultiplexada *rx_data*.

```
21 -    elseif D>1&&D<7 %ZF, MMSE, QRD, SQRD, MMSE-SQRD
22 -        temp_rx=zeros(64,1);
23 -        rx_data=zeros(64,10);
24 -          for j=1:n_sym      %Demultiplexado: obtencion de las 64 subportadoras
25 -                temp_rx=rxSig((j)*80+17:(j)*80+17+(64-1),1);
26 -                temp_rx=fft(temp_rx);
27 -                rx_data(:,j)=temp_rx;
28 -          end
```

Segmento de código 2.14 Demultiplexado manual de la señal recibida

2.3.1. OBTENCIÓN DE LA MATRIZ DEL CANAL PARA ZF, MMSE, QRD, SQRD Y MMSE-SQRD

Para obtener la matriz, se considera que el canal varía para la transmisión de cada uno de los símbolos, por lo que *H* será una matriz con tres dimensiones 64x64x10. En el Segmento de código 2.15, en cada una de las iteraciones del *for* de la línea 30 se irá construyendo la matriz 64x64 correspondiente a cada símbolo. Se comienza por definir la variable *ini*, la cual permite dejar de lado el preámbulo (80 subportadoras), prefijo cíclico (16 subportadoras) y los demás símbolos iniciales que se van descartando a medida que se avanza en el bucle.

```
30 -  for i=1:n_sym %Se empieza a armar la matriz h en dominio del tiempo
31 -      ini=i*80+17;
32 -      h1_temp=diag(pathgains(ini:ini+(64-1),1),0);  %Se obtiene la matriz con la diagonal principal
33 -      h_temp=h1_temp;   %Se la asigna temporalmente a h_temp
34
35 -          if length(sampIdx)>1      % Si el canal tiene 2taps
36 -              h2_temp=diag(pathgains(ini+sampIdx(2)-1:ini+sampIdx(2)-1+(64-1),2),-sampIdx(2)+1);
37 -              h_temp=h_temp+h2_temp(1:64,1:64);       %Se une la diagonal del 1er rayo con la obtenida
38 -                  if sampIdx(2)==2        %Para el caso en el que el efecto del segundo rayo sea de 1
39 -                     h_temp(1,64)=pathgains(ini+sampIdx(2)-2,2);  %Se asignan los elementos
40 -                  end
41 -                  if sampIdx(2)==3   %Para el caso en el que el efecto del segundo rayo sea de 2
42 -                     h_temp(1,63)=pathgains(ini+sampIdx(2)-3,2);
43 -                     h_temp(2,64)=pathgains(ini+sampIdx(2)-2,2);
44 -                  end
45 -          end
```

Segmento de código 2.15 Implementación de la matriz del canal

Para formar la matriz *H* se irán tomando los datos de las ganancias de la matriz *pathgains*, para ir formando las diagonales de la matriz *H*. La matriz *pathgains* puede ser de dimensiones 880x3 u 880x4 dependiendo de si se tienen 3 o 4 rayos. La función que nos permite crear la matriz diagonal es la función *diag*, donde se especifican como argumentos: el vector a partir del cual se conformará la diagonal de la nueva matriz y la posición de la diagonal en la matriz donde se ubicarán los valores. Al momento de especificar la posición de la diagonal, es importante considerar que la función *diag* establece a las diagonales positivas desde la diagonal principal hacia la parte superior derecha de la matriz. Mientras que las diagonales negativas, son las diagonales debajo de la diagonal principal en la parte inferior izquierda de la matriz así como se muestra en la Figura 2.3 [29]:

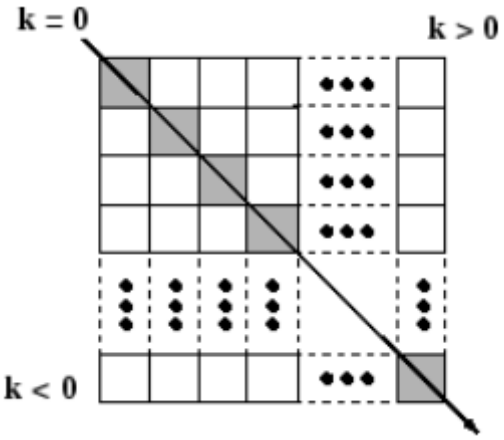

Figura 2.3 Número de diagonales para el comando *diag* [29]

Para el primer rayo, se emplean las primeras 64 ganancias de la primera columna de la matriz *pathgains* descartando el valor *ini* explicado anteriormente. El primer rayo corresponde a una transmisión donde no existe retardo. Por ello, se ubican los 64 elementos en la diagonal principal o diagonal cero. A cada matriz formada por la diagonal correspondiente, se la almacena temporalmente en *h1_temp, h2_temp, h3_temp* o *h4_temp* según el rayo que corresponda. Todas estas matrices con las diagonales obtenidas para cada rayo se irán consolidando en *h_temp*.

Para el vector a partir del cual se conformará la diagonal de la nueva matriz, se consideran dos aspectos: el retardo y la columna de la matriz pathgains que corresponde a cada rayo. El retardo se lo considera empleando el vector *sampIdx*, según el número del rayo se toma el valor de este vector. Por ejemplo, para el primer rayo se debe tomar el primer valor del vector *sampIdx*, para el segundo rayo se debe tomar el segundo valor del vector *sampIdx* y así sucesivamente. El retardo es considerado junto con la variable *ini* explicada anteriormente. Es por ello que se suma *ini* más *sampIdx* restado de uno para considerar el retardo neto y no el retardo incrementado en uno. Mientras que, la columna de la matriz *pathgains* se considera según el rayo. Similar que para *sampIdx*, para el primer rayo se considera la primera columna de la matriz *pathgains*, para el segundo rayo se considera la segunda columna de la matriz *pathgains* y así sucesivamente.

La posición de la diagonal en la matriz depende del retardo del escenario, el cual lo encontramos normalizado e incrementado en uno en la variable *sampIdx*. Estas posiciones de las diagonales irán en la parte inferior a la diagonal principal, es decir serán las diagonales negativas para el comando *diag*. Para ello se emplea *-sampIdx(2)+1*, *-sampIdx(3)+1 o -sampIdx(4)+1* según corresponda al rayo que se esté considerando. Es importante recordar que se suma uno debido a que los retardos almacenados en *sampIdx*

son retardos normalizados que se incrementaron en uno dentro de la función *Channel* como se muestra en el Segmento de código 2.11.

Pero es importante tomar en cuenta que, al sacar las demás diagonales, se crearan matrices *h2_temp, h3_temp* o *h4_temp* con dimensiones mayores que 64x64. Esto se debe a que cada una de las diagonales contiene 64 elementos, estos elementos deben distribuirse en la diagonal para lo cual el comando *diag* agranda la nueva matriz creada de modo que todos los 64 elementos quepan a lo largo de la diagonal. Es decir, si se tiene un retardo de 2, la diagonal a usarse seria la diagonal con posición -2. De donde el primer elemento se ubicaría en la posición (3,1), el segundo elemento en la posición (4,2) y así sucesivamente hasta que el último elemento se ubicaría en la posición (66,64). Por lo tanto, la matriz generada seria de 66x66.

Una vez obtenida la matriz con la diagonal, se consolida con las matrices de los anteriores rayos en *h_temp* considerando solamente la dimensión de 64x64. Pero se debe tomar en cuenta que las diagonales deben ser completadas. Los elementos para completar la diagonal se deben ubicar en la parte superior derecha de la matriz como se puede observar en la Figura 2.4. Por esta razón, se asigna directamente en las posiciones conocidas para cada diagonal, los valores faltantes. Estos valores corresponden a las ganancias dentro de la matriz *pathgains*. Pero consideran el retardo disminuido uno a uno.

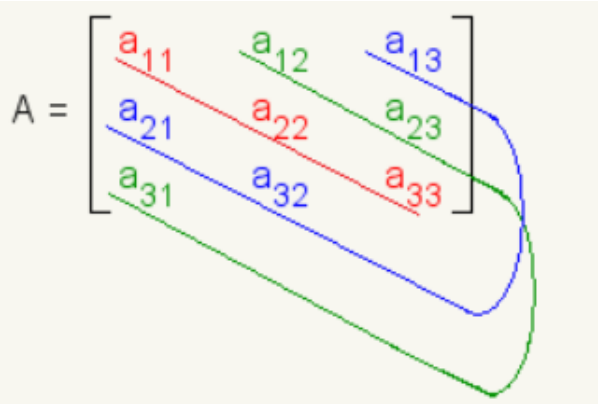

Figura 2.4 Posición de los elementos de las demás diagonales [30]

Para entender de mejor manera, lo explicado anteriormente, se tomará como ejemplo la generación de la matriz con la diagonal para el cuarto rayo considerando que el retardo de *sampIdx* para el cuarto rayo es de 6. Como se puede ver en el Segmento de código 2.16 en la línea 63, la diagonal de la matriz *h4_temp* está formada por los 64 elementos de *pathgains* empezando por *ini* sumado al retardo guardado en *sampIdx(4).* La posición de la diagonal es -5. En *h_temp* se almacena solo los elementos contenidos en las

dimensiones 64x64 de *h4_temp*. Para completar la diagonal -5 se deben ubicar los elementos faltantes en las posiciones (1,60), (2,61), (3,62), (4,63) y (5,64) de *h_temp*. Los valores almacenados en estas posiciones son las ganancias del canal ubicadas en la posición *ini* más *sampIdx(4)* disminuido progresivamente y la columna 4 que corresponde al cuarto rayo. La asignación de estos valores faltantes se puede ver en la línea 71 hasta la línea 75 del Segmento de código 2.16.

```
62 -    if length(sampIdx)>3                % Si el canal tiene 4taps
63 -        h4_temp=diag(pathgains(ini+sampIdx(4)-1:ini+sampIdx(4)-1+(64-1),4),-sampIdx(4)+1);
64 -        h_temp=h_temp+h4_temp(1:64,1:64);
65 -         if sampIdx(4)==4
66 -            h_temp(1,62)=pathgains(ini+sampIdx(4)-4,4);
67 -            h_temp(2,63)=pathgains(ini+sampIdx(4)-3,4);
68 -            h_temp(3,64)=pathgains(ini+sampIdx(4)-2,4);
69 -         end
70 -         if sampIdx(4)==6
71 -            h_temp(1,60)=pathgains(ini+sampIdx(4)-6,4);
72 -            h_temp(2,61)=pathgains(ini+sampIdx(4)-5,4);
73 -            h_temp(3,62)=pathgains(ini+sampIdx(4)-4,4);
74 -            h_temp(4,63)=pathgains(ini+sampIdx(4)-3,4);
75 -            h_temp(5,64)=pathgains(ini+sampIdx(4)-2,4);
76 -         end
77 -         if sampIdx(4)==8
```

Segmento de código 2.16 Generación de la matriz con la diagonal para el cuarto rayo

Una vez formada la matriz *h* en el dominio del tiempo, se obtiene la matriz de la transformada discreta de Fourier *F*. Con *F* y *h_temp* ya obtenidas, se puede calcular la matriz *H* en el dominio de la frecuencia según la ecuación (1.12) como se muestra en el Segmento de código 2.17.

```
88 -    F = fft(eye(64));                %Crear la matriz de Fourier
89 -    H(:,:,i)=(1/64)*F*h_temp*F';     %Obtener la matriz del canal
```

Segmento de código 2.17 Obtención de la matriz del canal *H* en el dominio de la frecuencia

La matriz *H* será utilizada para el algoritmo de los detectores ZF, MMSE, QRD, SQRD y MMSE-SQRD.

2.3.2. FUNCIÓN ONE-TAP

La función *One-tap* realiza lo descrito previamente en la sección 1.3.4.1. Los parámetros de entrada y salida de esta función se encuentran en la Tabla 2.5.

Tabla 2.5 Parámetros de la función *One-tap*

Parámetros	Tipo de Dato	Descripción
rx_data_mat	Entrada	Señal recibida y pasada por el demultiplexor OFDM
n_sym	Entrada	Número de símbolos OFDM
pathgains	Entrada	Respuesta del canal
sampIdx	Entrada	Posiciones de los retardos
data_eq	Salida	Señal detectada

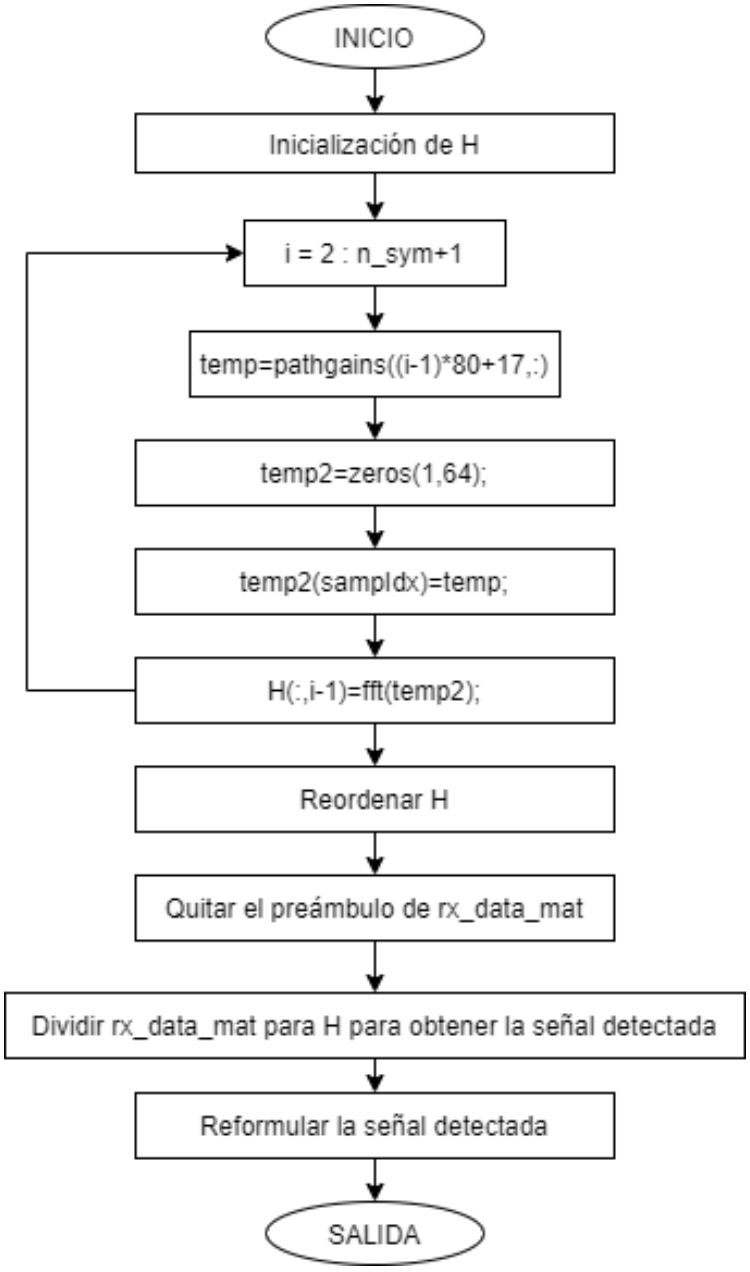

Figura 2.5 Diagrama de flujo del detector One-tap

A diferencia de todos los detectores, el detector one-tap no emplea la matriz del canal generada en la sección 2.3.1 porque no se necesita toda la matriz. Para este detector se emplean las 80 subportadoras. Por ello, se emplea la señal recibida puramente de datos que ya pasó por el demultiplexor OFDM. Para aplicar este detector se considera que la matriz del canal en tiempo, es simplemente una matriz diagonal como se describe en la sección 1.3.4.1. Para un mejor entendimiento se presenta la Figura 2.5 con el diagrama de flujo de la función implementada para el detector one-tap.

```
1  function data_eq=one_tap(rx_data_mat,n_sym,pathgains,sampIdx)
2  % rx_data_mat   Señal recibida despues del demodulador OFDM
3  % n_sym         Numero de simbolos OFDM
4  % pathgains     Respuesta del canal
5  % sampIdx       Retardos normalizados
6  % data_eq       Señal detectada
7  H=zeros(64,n_sym);
8  for i=2:n_sym+1
9      temp=pathgains((i-1)*80+17,:);
10     temp2=zeros(1,64);
11     temp2(sampIdx)=temp; % Se obtiene la diagonal de h
12     H(:,i-1)=fft(temp2);    % Se genera la respuesta de frecuencia del canal
13 end
14 H_eq=[H(39:43,:);H(45:57,:);H(59:64,:);H(2:7,:);H(9:21,:);H(23:27,:)];
15 rx_data_mat=rx_data_mat(:,2:end);
16 rx_data_eq=rx_data_mat./H_eq;            %one-tap
17 data_eq=reshape(rx_data_eq,48*n_sym,1);
18 end
```

Segmento de código 2.18 Implementación de la función para el detector One-tap

Para generar la matriz del canal *H* se van tomando los valores de las ganancias de *pathgains* descartando las 80 subportadoras del preámbulo, las 16 subportadoras del prefijo cíclico y las 80 subportadoras del símbolo ya considerada en cada iteración transcurrida. Estos 3 o 4 elementos, dependiendo de si se tienen 3 o 4 rayos, se almacenan en el vector *temp* como se ve en la línea 9 del Segmento de código 2.18.

Se crea el espacio de 1x64 para almacenar los valores de *temp* en *temp2* considerando las posiciones de los retardos que se tiene en *sampIdx*. De este modo, *temp2* tiene guardado la diagonal de *h* que es la matriz del canal en el dominio del tiempo. Luego se realiza la FFT para obtener *H* que es la matriz del canal en el dominio de la frecuencia. Este proceso se repite para cada símbolo OFDM, de modo que en cada columna de *H* se almacena la diagonal para cada símbolo OFDM.

Una vez obtenida *H*, se reordenan los datos considerando que la IFFT realizada en el multiplexor OFDM los reorganizó previamente para formar el símbolo OFDM. Estos valores definidos para reordenar los datos fueron previamente obtenidos en base a pruebas realizadas obteniendo como resultado las posiciones de las subportadoras que se muestran en la Figura 2.2. De la matriz *rx_data_mat* se toman solamente las subportadoras

recibidas de los 10 símbolos, dejando de lado el primer símbolo que es del preámbulo. Con estas consideraciones, se obtiene entonces la señal detectada dividiendo la señal recibida para la matriz del canal *H* en el dominio de la frecuencia como se muestra en la línea 11 del Segmento de código 2.18. Finalmente, se reformula la señal detectada, en un vector *data_eq* de dimensiones 480x1, el cual es la salida de la función *One-tap*.

2.3.3. DETECTOR ZF

El diagrama de flujo mostrado en la Figura 2.6 contiene el proceso de detección de la señal empleando el algoritmo ZF. En el Segmento de código 2.19 se implementa este detector como parte de la función *Detector*. Con la matriz del canal *H* calculada previamente y *rx_data* que es el vector de datos reordenado y pasado por la FFT, se obtiene la señal detectada *rx_data_eq*. Para ello, se utiliza el comando *pinv* que obtiene la pseudoinversa de la matriz *H* y se la multiplica por *rx_data*. De este modo se aplica la Ecuación (1.15) descrita previamente para el detector ZF. La señal detectada debe obtenerse para cada uno de los símbolos OFDM, por lo que se tendrán 10 iteraciones dentro de este lazo.

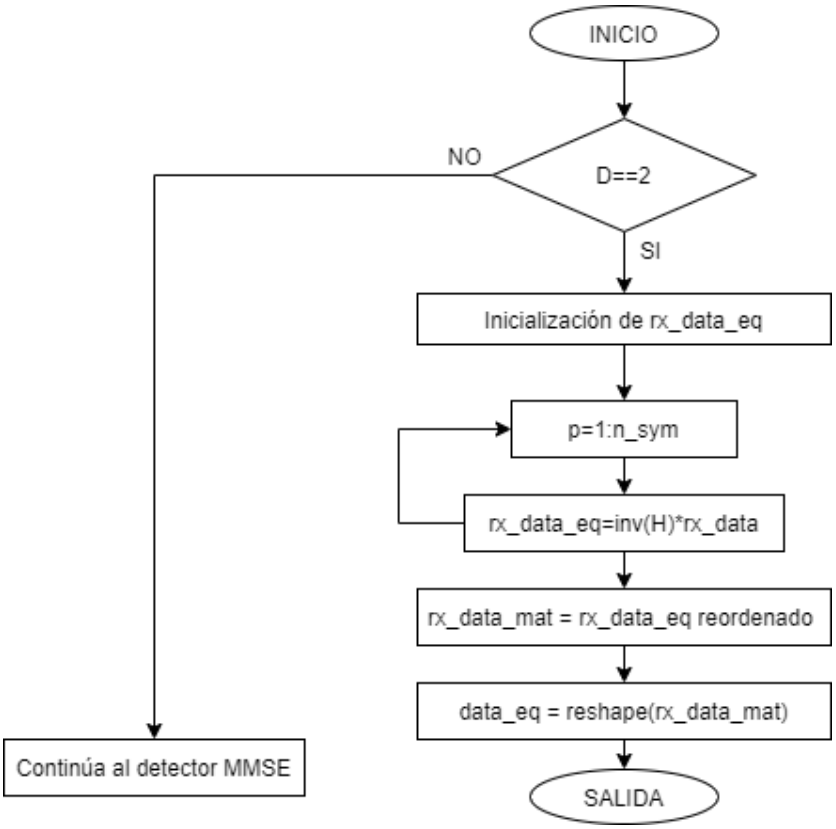

Figura 2.6 Diagrama de flujo del detector ZF

```
101 -     if D==2 %ZF
102 -        x_data_det=zeros(64,10);
103 -           for p=1:n_sym
104 -               x_data_det(:,p)=inv(H(:,:,p))*rx_data(:,p); %para cada simbolo
```

Segmento de código 2.19 Implementación del detector ZF

Cuando ya se ha obtenido la señal detectada final, se ha cumplido con el condicional para el detector ZF, por lo que se sale del condicional hacia las líneas 135 a 136 del Segmento de código 2.20 en la misma función *Detector* donde se reordenan los datos considerando que la IFFT los desordenó previamente. Estos valores definidos para reordenar los datos fueron previamente obtenidos en base a pruebas realizadas. Finalmente, en la línea 137 se reformula la señal detectada, en un vector de dimensiones 480x1, el cual es la salida de la función *Detector*.

```
135 -     x_data_mat=[x_data_det(39:43,:);x_data_det(45:57,:);x_data_det(59:64,:);...
136           x_data_det(2:7,:);x_data_det(9:21,:);x_data_det(23:27,:)];  %Reorganizar
137 -     data_det=reshape(x_data_mat,48*n_sym,1);
```

Segmento de código 2.20 Reordenamiento de la señal detectada

El segmento de código anterior se ejecutará como parte final de cada uno de los detectores ZF, MMSE, QRD, SQRD y MMSE-SQRD.

2.3.4. DETECTOR MMSE

Para el detector MMSE, se emplea un algoritmo un poco más elaborado que el utilizado anteriormente para el ZF. Este detector se encuentra formando parte de la función *Detector*. Como se puede ver en el diagrama de flujo mostrado en la Figura 2.7, cuando la variable *D* que determina el detector que se va a emplear tiene un valor de 3, se ejecuta el algoritmo del detector MMSE. Para ello se define una matriz unitaria en las líneas 108 y 109 del Segmento de código 2.21. Para calcular la matriz de pesos que servirá para obtener la señal detectada, se usa la Ecuación (1.16). En dicha ecuación se emplea la matriz *H* obtenida anteriormente, la varianza del ruido almacenada en la variable *ruido* y la matriz identidad *I* que se acaba de obtener. Una vez que se obtiene la matriz de pesos, se la multiplica por la señal recibida que está almacenada en el vector *rx_data* para obtener la señal detectada *x_data_det*. Esta operación se la realiza cumpliendo con lo mencionado previamente en la Ecuación (1.20). La señal detectada debe ser obtenida para cada símbolo OFDM, por lo que se tendrán 10 iteraciones dentro de este lazo.

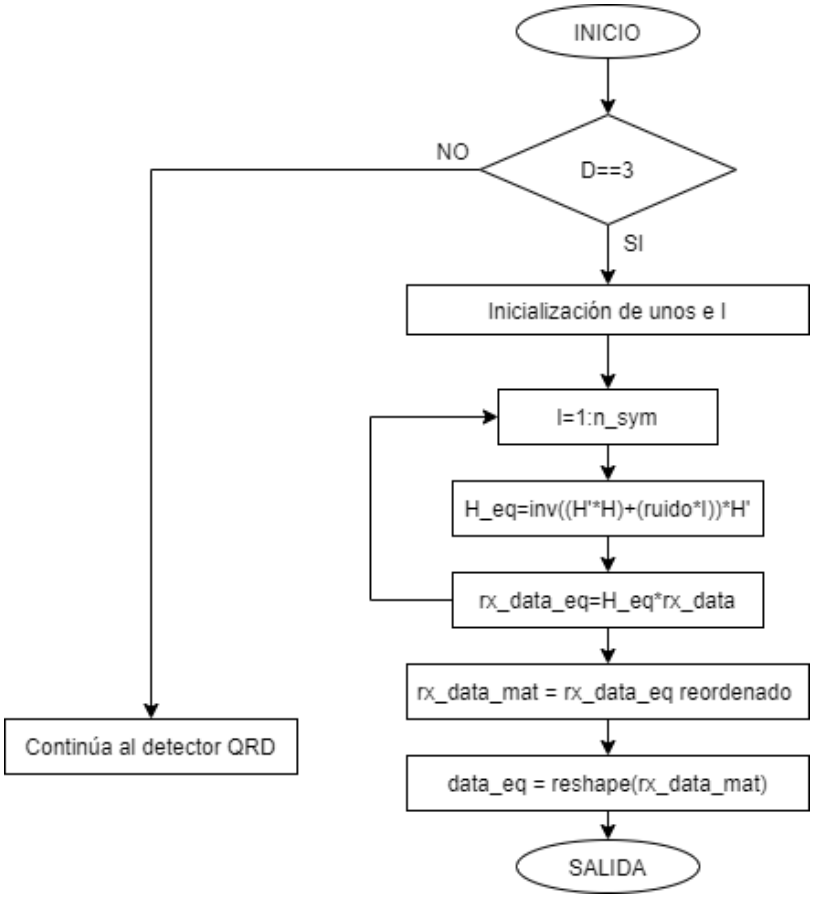

Figura 2.7 Diagrama de flujo del detector MMSE

```
107 -    elseif D==3 %MMSE
108 -       unos=ones(1, 64);
109 -       I=diag(unos);
110 -           for l=1:n_sym
111 -               H_eq(:,:,l)=inv(((H(:,:,l))')*H(:,:,l)+(ruido*I))*(H(:,:,l))';
112 -               x_data_det(:,l)=H_eq(:,:,l)*rx_data(:,l);
113 -           end
```

Segmento de código 2.21 Implementación del detector MMSE

Cuando ya se ha obtenido la señal detectada final, se cumple el condicional para el detector MMSE por lo que la simulación salta hacia las líneas 135 y 136 del Segmento de código 2.20 dentro de la función *Detector*. Por lo cual, del mismo modo que para el detector ZF, se reordenan los datos considerando que la IFFT los desordenó previamente.

2.3.5. DETECTOR QRD

El detector QRD a diferencia de los dos detectores anteriores, emplea la descomposición QR. Este detector está implementado dentro de la función *Detector*. Para emplear este detector, la variable *D* debe tener un valor de 4 y con ello cumple la primera condición como se puede ver en el diagrama de flujo mostrado en la Figura 2.8. Se debe indicar las

posiciones donde irán la subportadoras piloto y las subportadoras nulas. Para esto, en las líneas 116 y 117 del Segmento de código 2.22 se muestran los vectores con estas posiciones. Dentro del bucle principal de este detector se realiza la descomposición QR de la matriz del canal *H*. Esto se obtiene empleando la función *qr* que retorna la matriz triangular superior *R* y la matriz *Q*. De este modo, se obtiene la señal transmitida multiplicando la matriz transpuesta Hermitiana de *Q* por la señal recibida almacenada en el vector *rx_data*. Esta operación da cumplimiento a lo antes mencionado en la Ecuación (1.25).

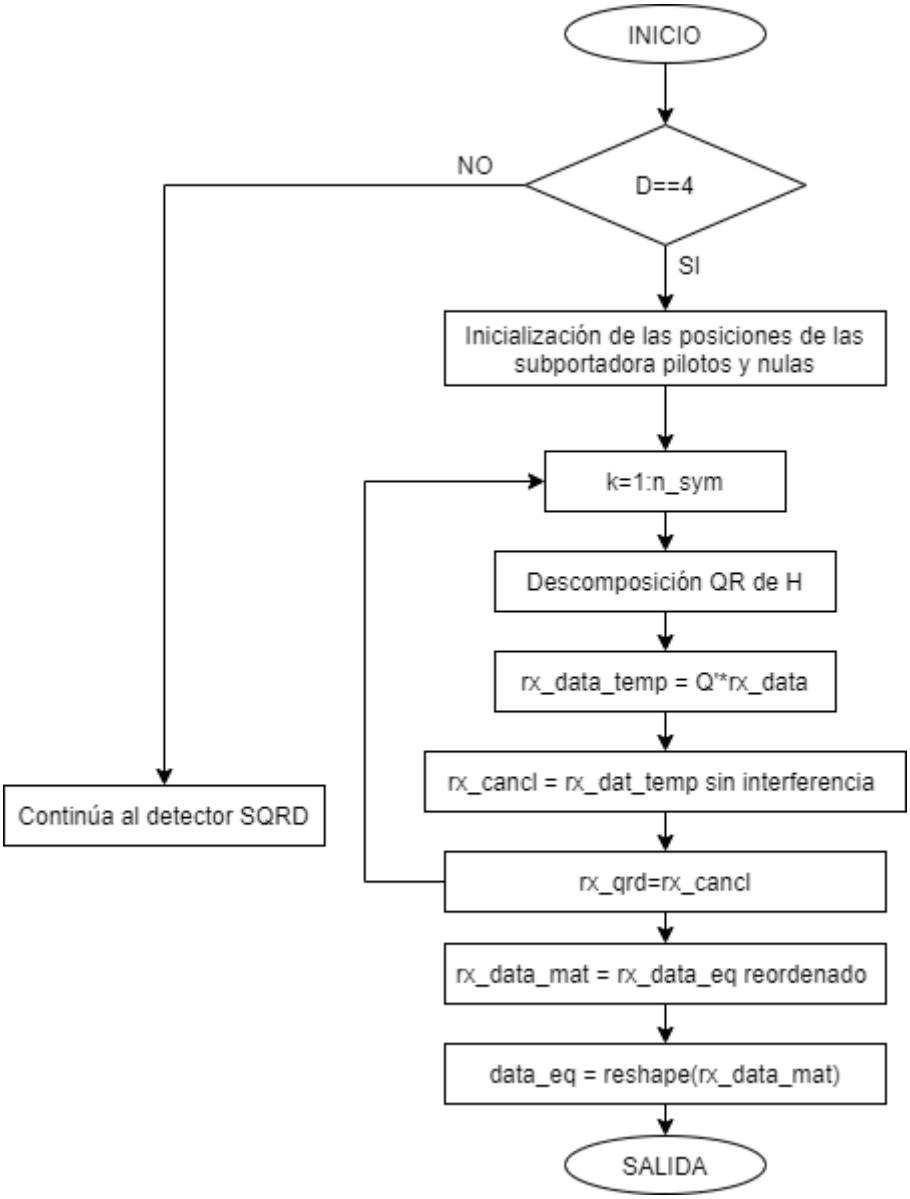

Figura 2.8 Diagrama de flujo del detector QRD

Sin embargo, es importante considerar la propagación de errores a la que es propenso el detector QRD cuando se genera un error en una capa inferior. Por lo que se somete a la

señal transmitida calculada a un cancelador de interferencia mediante la función implementada *Cancelador*. Mediante esta función se elimina la interferencia y se obtiene una señal detectada más confiable en el vector *x_cancl*. Se realiza este procedimiento 10 veces dentro del bucle para cada símbolo OFDM.

```
115 -   elseif D==4 %QRD
116 -       pilot=[8, 22, 44, 58];
117 -       nulls=[1,28:38];
118 -          for k=1:n_sym
119 -               [Q,R] = qr(H(:,:,k));
120 -               x_data_temp=Q'*rx_data(:,k);
121 -               x_data_det(:,k) = cancelador(x_data_temp,R, M,pilot,nulls);
122 -          end
```

Segmento de código 2.22 Implementación del detector QRD

Cuando ya se ha obtenido la señal detectada final, se cumple el condicional para el detector QRD por lo que la simulación salta hacia las líneas 135 y 136 del Segmento de código 2.20 dentro de la función *Detector*. Por lo cual, del mismo modo que para los detectores ZF y MMSE, se reordenan los datos considerando que la IFFT los desordenó previamente.

2.3.6. FUNCIÓN PARA EL DETECTOR SQRD

Dentro de la función Detector se tiene la condición para cuando D tiene un valor de 5, donde se ingresa al bucle que ejecuta el detector SQRD. Como se puede ver en el Segmento de código 2.23 de la función Detector, se ingresa a un lazo for que se ejecutará para cada símbolo OFDM. Dentro del mismo se llama a la función SQRD implementada para el presente proyecto.

```
124 -   elseif D==5 %SQRD
125 -      for x=1:n_sym
126 -           x_data_det(:,x) = SQRD(rx_data,H,M,x);
127 -      end
```

Segmento de código 2.23 Implementación del detector SQRD

Para comprender mejor el accionar de esta función se presenta la Figura 2.9.

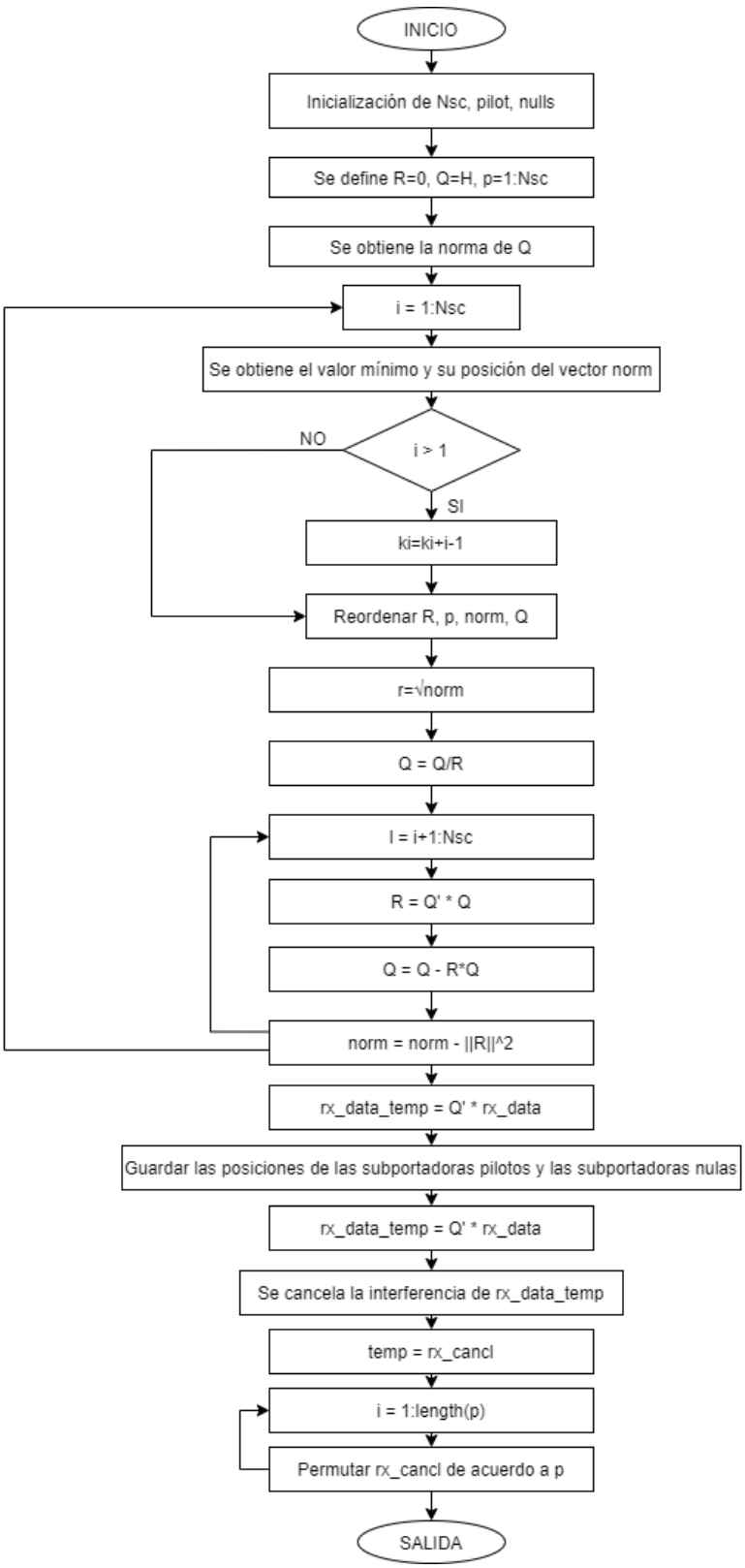

Figura 2.9 Diagrama de flujo del detector SQRD

La función *SQRD* realiza lo descrito previamente en el algoritmo de la sección 1.3.4.4. Los parámetros de entrada y salida de esta función se encuentran en la Tabla 2.6.

Tabla 2.6 Parámetros de la función *SQRD*

Parámetros	Tipo de Dato	Descripción
rx_data	Entrada	Señal recibida
H	Entrada	Matriz del canal en el dominio de la frecuencia
M	Entrada	Tamaño de la constelación
x	Entrada	Número del símbolo detectado
rx_data_eq	Salida	Señal detectada

A diferencia del detector QRD, el detector SQRD utiliza también las matrices *Q* y *R*, pero estas están concebidas de forma muy distinta. Para cumplir con el pseudocódigo previamente expuesto en la sección 1.3.4.4, se define *R* como una matriz nula, *Q* toma los valores de la matriz *H*, se define un vector *p* donde se almacenarán las posiciones de cada subportadora. También se obtiene la norma de *Q* al cuadrado.

Dentro del bucle principal se van tomando las normas mínimas y sus posiciones se almacenan en *ki*. Pero a medida que se continua con las iteraciones, el vector de normas del cual se obtiene el mínimo, se hace más pequeño en cada iteración. Con el propósito de compensar la posición desde la cual se parte para analizar el vector de normas, se define un condicional en la línea 11 del Segmento de código 2.24.

```
1  function rx_data_eq = SQRD(rx_data,H,M,x)
2  Nsc=size(H,1);          % Se define el numero de subportadoras
3  pilot=zeros(1,4);       % Se define el espacio para las posiciones de las subportadoras pilotos
4  nulls=zeros(1,12);      % Se define el espacio para las posiciones de las subportadoras null
5  R=zeros(Nsc,Nsc);       % Se define R como una matriz nula
6  Q=H(:,:,x);             % Se cumple la condicion del algoritmo de que Q=H
7  p=[1:Nsc];              % Se define un vector para almacenar las posiciones
8  norm=sum(abs(Q).^2,1);          %Se obtiene la norma de Q
9  for i=1:Nsc
10     [gk ki]=min(norm(i:Nsc));   %Se obtiene el valor de norma minimo y su posicion
11     if (i>1)
12         ki=ki+i-1;          %Se compensa la posicion del vector porque solo se toma una parte
13     end
```

Segmento de código 2.24 Obtención de la norma mínima y su posición

Una vez consideradas estas posiciones, se va reordenando las columnas de las matrices *R* y *Q* en base a las posiciones de las normas mínimas obtenidas. También se reordenan los elementos del vector de posiciones *p* y del vector de normas *norm*. El reordenamiento de esos elementos se lo puede observar en el Segmento de código 2.25.

```
15 -    rtemp=R(:,i);           %Se reordena la matriz R intercambiando la columna i por ki
16 -    R(:,i)=R(:,ki);
17 -    R(:,ki)=rtemp;
18
19 -    ptemp=p(i);             %Se reordena el vector p intercambiando la columna i por ki
20 -    p(i)=p(ki);
21 -    p(ki)=ptemp;
22
23 -    ntemp=norm(i);          %Se reordena el vector norm intercambiando la columna i por ki
24 -    norm(i)=norm(ki);
25 -    norm(ki)=ntemp;
26
27 -    qtemp=Q(:,i);           %Se reordena la matriz Q intercambiando la columna i por ki
28 -    Q(:,i)=Q(:,ki);
29 -    Q(:,ki)=qtemp;
```

Segmento de código 2.25 Implementación del reordenamiento de *R, p, norm* y *Q*

Continuando con el algoritmo se obtiene una matriz mejorada de *R* que reduzca la propagación de errores. Para ello se obtiene la raíz del valor más pequeño de las normas y se lo almacena en *R*. De esta forma se puede calcular la nueva columna de la matriz *Q* dividiendo *Q* para el valor de *R* calculado en el paso anterior. En el Segmento de código 2.26 se puede observar las operaciones mencionadas así como también un bucle adicional que permitirá ejecutar las operaciones de las líneas 11, 12 y 13 del algoritmo de la sección 1.3.4.4.

```
31 -        R(i,i)=sqrt(gk);            %%%%only norm
32 -        Q(:,i)=Q(:,i)/R(i,i);
33 -        for l=i+1:Nsc
34 -            R(i,l)=(Q(:,i))'*Q(:,l);
35 -            Q(:,l)=Q(:,l)-R(i,l)*Q(:,i);
36 -            norm(l) = norm(l) - abs(R(i,l))*abs(R(i,l));
37 -        end
```

Segmento de código 2.26 Implementación de las matrices *R* y *Q*

Una vez ejecutado el bucle principal donde se ha obtenido las matrices *Q* y *R*, se procede a obtener la señal detectada multiplicando la transpuesta hermitiana de *Q* por los datos recibidos. También se almacenan las nuevas posiciones de las subportadoras piloto y de las subportadoras nulas, obtenidas después del reordenamiento realizado en el Segmento de código 2.25. Para almacenar estas posiciones se emplean los vectores *pilot* y *nulls* como se muestra en las líneas 41 a 56 del Segmento de código 2.27. Como siguiente y, del mismo modo que se realizó para el detector QRD, la señal debe ser sometida al cancelador de interferencia. Como parte final de la función *SQRD*, se utiliza la señal detectada y optimizada para realizar la permutación, tomando en cuenta las posiciones almacenadas en el vector *p*.

```
rx_data_temp=Q'*rx_data(:,x);  %Se obtiene la senal recibida

pilot(1)=find(p==8);            %Se extrae las nuevas posiciones de las portadoras pilotos
pilot(2)=find(p==22);
pilot(3)=find(p==44);
pilot(4)=find(p==58);
nulls(1)=find(p==1);            %Se extrae las nuevas posiciones de las portadoras nulas
nulls(2)=find(p==28);
nulls(3)=find(p==29);
nulls(4)=find(p==30);
nulls(5)=find(p==31);
nulls(6)=find(p==32);
nulls(7)=find(p==33);
nulls(8)=find(p==34);
nulls(9)=find(p==35);
nulls(10)=find(p==36);
nulls(11)=find(p==37);
nulls(12)=find(p==38);

rx_cancl=cancelador(rx_data_temp,R, M, pilot,nulls);    %Se cancela la interferencia

temp=rx_cancl;                  %Se almacena temporalmente la senal
for i=1:length(p)               %Se realiza la permutacion
        rx_cancl(p(i))=temp(i);
end
rx_data_eq=rx_cancl;
end
```

Segmento de código 2.27 Obtención de la señal detectada para el detector SQRD

Finalmente, regresando a la función *Detector*, una vez obtenida la señal final de los símbolos OFDM se debe reordenar los datos y reformularlos del mismo modo que se ha hecho para los detectores anteriores como se muestra en el Segmento de código 2.20.

2.3.7. FUNCIÓN PARA EL DETECTOR MMSE-SQRD

Dentro de la función *Detector* se tiene la condición para cuando *D* tiene un valor de 6, donde se ingresa al bucle que ejecuta el detector MMSE-SQRD. Para un mejor entendimiento, en la Figura 2.10 se presenta el diagrama de flujo de la función *MMSE_SQRD* implementada. Como se puede ver en el Segmento de código 2.28 de la función *Detector*, se ingresa a un lazo *for* que se ejecutará para cada símbolo OFDM. Dentro del mismo se llama a la función *MMSE_SQRD* implementada para el presente proyecto.

```
elseif D==6 %MMSE-SQRD
    for x=1:n_sym
        x_data_det(:,x) = MMSE_SQRD(rx_data,H,ruido,M,x);
    end
```

Segmento de código 2.28 Implementación del detector MMSE-SQRD

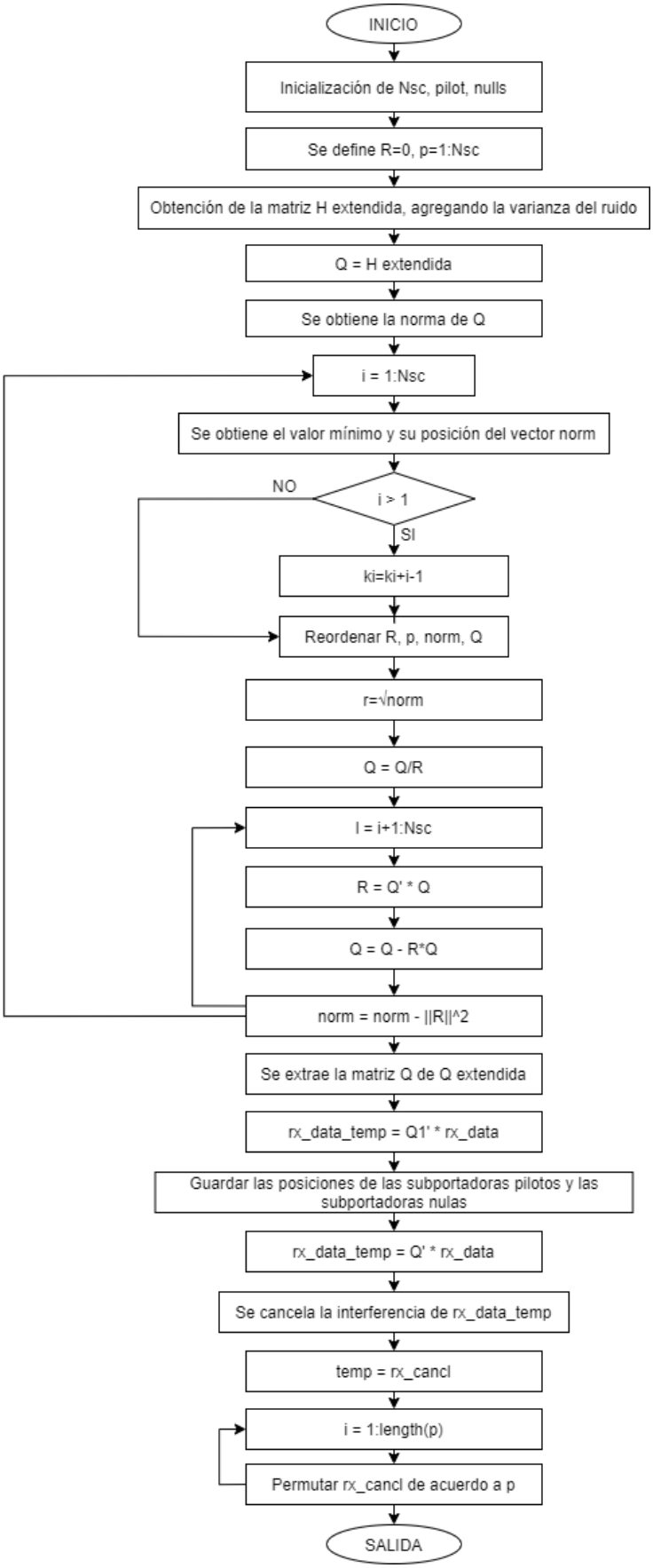

Figura 2.10 Diagrama de flujo del detector MMSE-SQRD

La función *MMSE-SQRD* realiza lo descrito previamente en el algoritmo de la sección 1.3.4.4. Los parámetros de entrada y salida de esta función se encuentran en la Tabla 2.7.

Tabla 2.7 Parámetros de la función *MMSE-SQRD*

Parámetros	Tipo de Dato	Descripción
rx_data	Entrada	Señal recibida
H	Entrada	Matriz del canal en el dominio de la frecuencia
ruido	Entrada	Varianza del ruido
M	Entrada	Tamaño de la constelación
x	Entrada	Número del símbolo detectado
rx_data_eq	Salida	Señal detectada

El detector MMSE-SQRD es similar al detector SQRD, excepto porque este utiliza la matriz ampliada de *H* conocida como $\underline{H}$. Como se definió previamente en la ecuación (1.31), esta matriz no es más que la matriz *H* del canal indexada con una matriz diagonal que contiene los valores de la varianza del ruido en su diagonal principal. De este modo $\underline{H}$ resulta ser una matriz cuyas dimensiones son de 128x64x10. La implementación de $\underline{H}$ se realiza en las líneas 7 a 9 del Segmento de código 2.29.

```
1   function rx_data_eq = MMSE_SQRD(rx_data,H,ruido,M,x)
2   Nsc=size(H,1);          %Se obtiene el tamano de H
3   pilot=zeros(1,4);       %Se define el espacio para las posiciones de las subportadoras pilotos
4   nulls=zeros(1,12);      %Se define el espacio para las posiciones de las subportadoras null
5   R=zeros(Nsc,Nsc);       %Se define R como una matriz nula
6   p=[1:Nsc];              %Se define un vector para almacenar las posiciones
7   for i=1:Nsc
8       H(i+Nsc,i,x)=ruido;  %Se obtiene la matriz extendida de H, agregando la varianza del ruido
9   end
10  Q=H(:,:,x);             %Se cumple la condicion del algoritmo de que Q=H
11  norm=sum(abs(Q).^2,1);          %Se obtiene la norma de Q
```

Segmento de código 2.29 Implementación de la matriz $\underline{H}$

Para cumplir con el pseudocódigo previamente expuesto en la sección 1.3.4.4, se define *R* como una matriz nula, *Q* toma los valores de la matriz $\underline{H}$ y se define un vector *p* donde se almacenarán las posiciones de cada subportadora. También se obtiene la norma de *Q* al cuadrado.

Dentro del bucle principal, de forma similar a lo que se ha hecho en el bucle principal del detector SQRD, se van reordenando *R*, *p*, *norm* y *Q* tomando las normas mínimas y sus posiciones como se muestra en el Segmento de código 2.24 y Segmento de código 2.25. Posteriormente se calculan *R*, *Q* y *norm* del mismo modo que se hizo para el detector SQRD empleando el algoritmo presentado en la sección 1.3.4.4. Es decir, del mismo modo que se realizó en el Segmento de código 2.26. Sin embargo, la diferencia es que para el

detector MMSE-SQRD, es importante considerar que se debe tomar netamente la matriz *Q* y no la matriz *Q* extendida. Por lo que en la línea 42 del Segmento de código 2.30 se extrae en *Q1* netamente la matriz *Q* a partir de la matriz *Q* extendida.

```
42 -    Q1=Q(1:Nsc,:);
43 -    rx_data_temp=Q1'*rx_data(:,x);
44
45 -    pilot(1)=find(p==8);
46 -    pilot(2)=find(p==22);
```

Segmento de código 2.30 Extracción de *Q*, a partir de la matriz *Q* extendida

Como siguiente y, del mismo modo que se realizó para el detector SQRD, se extraen las nuevas posiciones de las subportadoras piloto y nulas como se mostró en el Segmento de código 2.27. Se cancela la interferencia y se cuantifica la señal recibida mediante las funciones implementadas *Cancelador* y *Cuantificador*. Como parte final de la función *MMSE-SQRD*, se utiliza la señal detectada y optimizada para realizar la permutación, tomando en cuenta las posiciones almacenadas en el vector *p,* obteniéndose así la señal detectada.

Una vez ejecutada la función *MMSE-SQRD*, regresando a la función *Detector*, se debe reordenar los datos y reformularlos del mismo modo que se ha hecho para todos los detectores anteriores como se muestra en el Segmento de código 2.20.

2.4. FUNCIÓN CANCELADOR

La función *Cancelador* tiene como objetivo recuperar los símbolos transmitidos a través de la cancelación sucesiva de la interferencia generada por los símbolos de las capas superiores en los detectores que emplean la descomposición QR, para ello ejecuta las operaciones descritas de forma general en el diagrama de flujo que se muestra en la Figura 2.11. Los parámetros de entrada y salida de esta función se encuentran en la Tabla 2.8.

Tabla 2.8 Parámetros de la función *Cancelador*

Parámetros	Tipo de Dato	Descripción
x_data_temp	Entrada	Señal recibida
R	Entrada	Matriz del canal en el dominio de la frecuencia
M	Entrada	Tamaño de la constelación
pilot	Entrada	Vector con posiciones de las subportadoras piloto
nulls	Entrada	Vector con posiciones de las subportadoras nulas
x_cancl	Salida	Señal detectada

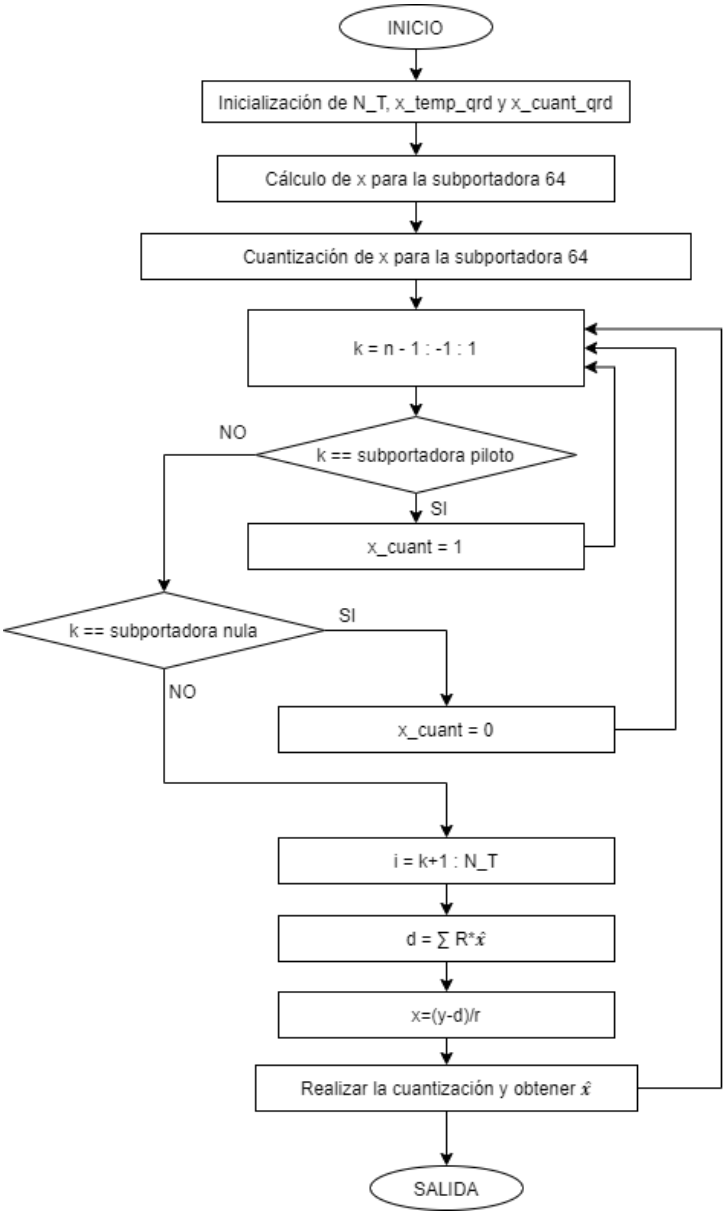

Figura 2.11 Diagrama de flujo de la función *Cancelador*

Como se muestra en el Segmento de código 2.31, se empieza calculando de forma regresiva la k-esima subportadora de la señal detectada. Primero se calcula para la subportadora 64. Para este caso la interferencia de la subportadora 64 es nula ya que según la ecuación (1.27) se la calcula a partir del elemento (64,65) de la matriz *R* y el elemento 65 del vector de la señal detectada. Como estos valores no existen, la interferencia de la subportadora 64 es cero como se designó en la línea 13 de la función *Cancelador*. Una vez obtenida esta interferencia, se necesita calcular el elemento 64 de la señal detectada y cuantizada. Para ello se aplica la ecuación (1.28) en donde se divide el

elemento 64 de la señal recibida para el elemento 64 de la matriz *R*. Este resultado debe cuantizarse, para lo cual se aplica la modulación y la demodulación obteniéndose $\hat{x}_{64}$.

```
function x_cancl = cancelador(x_data_temp,R, M, pilot,nulls)
% Esta funcion permite cancelar la interferencia generada en los detectores
% x_data_temp   Senal recibida
% R             Matriz R de la descomposicion QR
% M             Tamaño de la constelacion
% pilot         Vector de subportadoras piloto
% nulls         Vector de subportadoras nulas
% x_cancl       Senal detectada
% d_k           Interferencia
N_T=64;
x_temp_qrd=zeros(64,1);
% Para k=64
d_64=0;   %Para el la ultima portadora no se tiene interferencia
x_temp_qrd(64)=x_data_temp(64,1)/R(64,64);
temp=qamdemod(x_temp_qrd(64,1),M,'OutputType','bit','UnitAveragePower',true);
x_cancl(64,1)= qammod(temp,M,'InputType','bit','UnitAveragePower',true);
for k=N_T-1:-1:1
    % Se pone todas las subportadoras pilotos en 1
    if k==pilot(1)||k==pilot(2)||k==pilot(3)||k==pilot(4)
       x_cancl(k,1)=1;
    % Se pone todas las subportadoras nulas en 0
    elseif k==nulls(1)||k==nulls(2)||k==nulls(3)||k==nulls(4)||k==nulls(5)...
            ||k==nulls(6)||k==nulls(7)||k==nulls(8)||k==nulls(9)...
            ||k==nulls(10)||k==nulls(11)||k==nulls(12)
       x_cancl(k,1)=0;
    else
        for i=k+1:N_T
            d_temp_interf(i)=R(k,i)*x_cancl(i,1);
            d_interf=sum(d_temp_interf);
        end
        x_temp_qrd(k,1)=(x_data_temp(k,1)-d_interf)/R(k,k);
    %     temp=qamdemod(x_temp_qrd(k,1),M,'OutputType','bit','UnitAveragePower',true);
    %     x_cuant_qrd(k,1)= qammod(temp,M,'InputType','bit','UnitAveragePower',true);
        x_cancl(k,1) = cuantificador(x_temp_qrd,M,k);
    end
end

end
```

Segmento de código 2.31 Implementación de la función *Cancelador*

Una vez obtenido $\hat{x}_{64}$ se debe obtener las interferencias de las demás subportadoras. Para ello se debe considerar que los valores cuantizados de las subportadoras piloto y subportadoras nulas no necesitan ser calculadas puesto que estas tienen valores fijos de 1 para las subportadoras piloto y de 0 para las subportadoras nulas. Para las demás interferencias se realiza la operación de sumatorio según la ecuación (1.27).

Con estas interferencias se procede a calcular los elementos cuantizados de la señal detectada aplicando la división presentada en la ecuación (1.29). Sin embargo, para obtener el elemento cuantizado, se emplea la función *Cuantificador* que será explicada en la sección 2.5.

2.5. FUNCIÓN CUANTIFICADOR

La función *Cuantificador* tiene como objetivo realizar el proceso de cuantificación de la señal detectada para los detectores QRD, SQRD y MMSE-SQRD. Los parámetros de entrada y salida de esta función se encuentran en la Tabla 2.9.

Tabla 2.9 Parámetros de la función *Cuantificador*

Parámetros	Tipo de Dato	Descripción
x_temp_qrd	Entrada	Señal detectada sin interferencia
M	Entrada	Tamaño de la constelación
k	Entrada	Vector con posiciones de las subportadoras nulas
x_cuant	Salida	Señal cuantizada

En el Segmento de código 2.32 se muestra la forma en la que se implementó la función cuantificador.

```
1   function x_cuant = cuantificador(x_temp_qrd,M,k)
```

Segmento de código 2.32 Función *Cuantificador*

Esta función está dividida en 3 secciones, una para cada esquema de modulación. La primera sección realiza la cuantificación para constelación QPSK, la segunda sección realiza la cuantificación para una constelación 16QAM y la última sección realiza la cuantificación para una constelación 64QAM. Cada una de estas modulaciones considera una constelación como las que se muestran en la Figura 2.12.

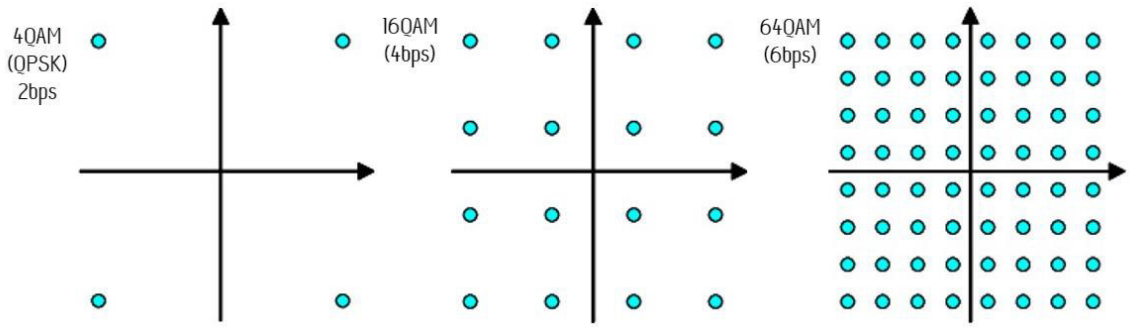

Figura 2.12 Constelaciones QPSK, 16QAM y 64QAM [31]

La función *Cuantificador* toma el valor del k-ésimo elemento de la señal detectada *x_temp_qrd* y lo compara para saber si se encuentra dentro de en un rango cercano al punto de la constelación como se puede ver en cada área sombreada en la Figura 2.13.

Una vez encontrado el punto de la constelación a la que pertenece, se le asigna el valor cuantizado considerando el factor de normalización de la constelación correspondiente que se haya seleccionado al inicio de la simulación en la variable *M*. Para realizar la cuantización se consideran los factores de normalización indicados según el estándar IEEE 802.11p en la Tabla 1.5.

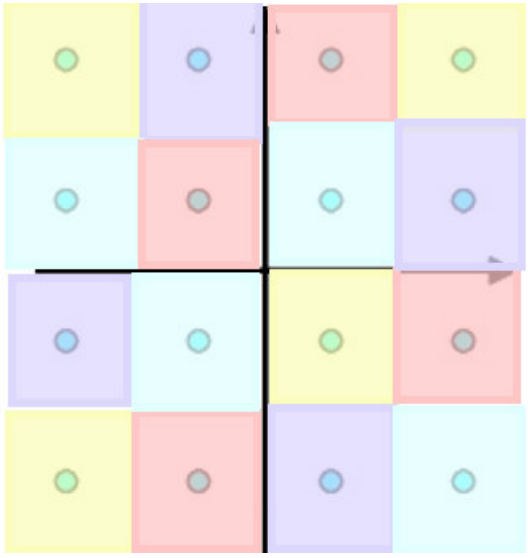

Figura 2.13 Rangos alrededor de cada punto para 16QAM

```
17 -    if M==16 %Para 16 QAM
18 -    norm_factor16=sqrt(10);
19 -    if (real(x_temp_qrd(k,1))>0)
20 -        if (real(x_temp_qrd(k,1))<=2/norm_factor16)
21 -            if (imag(x_temp_qrd(k,1))>0)&&(imag(x_temp_qrd(k,1))<=2/norm_factor16)
22 -                x_cuant=(1+1i)/norm_factor16;
23 -            elseif (imag(x_temp_qrd(k,1))>2/norm_factor16)
24 -                x_cuant=(1+3i)/norm_factor16;
25 -            elseif (imag(x_temp_qrd(k,1))<0)&&(imag(x_temp_qrd(k,1))>=-2/norm_factor16)
26 -                x_cuant=(1-1i)/norm_factor16;
27 -            else
28 -                x_cuant=(1-3i)/norm_factor16;
29 -            end
30 -        else
31 -            if (imag(x_temp_qrd(k,1))>0)&&(imag(x_temp_qrd(k,1))<=2/norm_factor16)
32 -                x_cuant=(3+1i)/norm_factor16;
33 -            elseif (imag(x_temp_qrd(k,1))>2/norm_factor16)
34 -                x_cuant=(3+3i)/norm_factor16;
35 -            elseif (imag(x_temp_qrd(k,1))<0)&&(imag(x_temp_qrd(k,1))>=-2/norm_factor16)
36 -                x_cuant=(3-1i)/norm_factor16;
37 -            else
38 -                x_cuant=(3-3i)/norm_factor16;
39 -            end
40 -        end
41
42 -    else
43 -        if (real(x_temp_qrd(k,1))>=-2/norm_factor16)
44 -            if (imag(x_temp_qrd(k,1))>0)&&(imag(x_temp_qrd(k,1))<=2/norm_factor16)
```

Segmento de código 2.33 Implementación de la constelación 16QAM

Por ejemplo, si la constelación es 16QAM, se ingresa a la segunda sección de la función *Cuantificador* que se muestra en el Segmento de código 2.33. Primero se establece el factor de normalización correspondiente. Con el fin de limitar la cantidad de comparaciones y reducir el tiempo de ejecución de la simulación. Se separó por cuadrantes las comparaciones. En la línea 19 se analiza si la parte real del valor es positiva o negativa. De este modo, si el valor es positivo el valor a cuantizar se encontraría en los cuadrantes I o IV. Mientras que, si el valor es negativo, el valor a cuantizar se encontraría en los cuadrantes II o III.

Para el ejemplo se asumirá que el valor a cuantizar se encuentra en el cuadrante IV en el rango alrededor del punto (1,-3). Una vez cumplida la condición de la línea 19, se delimita la comparación para saber si el valor se encuentra dentro de la primera fila de valores de los cuadrantes I o IV como se muestra en el área sombreada de la Figura 2.14. Para ello se compara si la parte real es menor que 2 dividido para el factor de normalización.

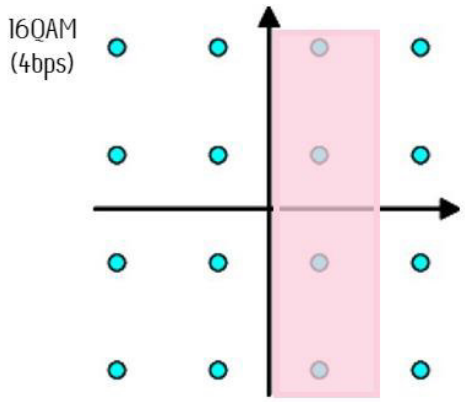

Figura 2.14 Constelación 16QAM

Como sí se cumple con esta condición, se ingresa a una serie de comparaciones que van a partir de la línea 21 del Segmento de código 2.33. Cada una de estas comparaciones va barriendo el área sombreada de la Figura 2.14 de arriba hacia abajo. Para ello se compara la parte imaginaria del valor. Cuando se identifica el área en el que se encuentra el elemento, se asigna el valor cuantizado.

3. RESULTADOS Y DISCUSIÓN

En este capítulo se presentan los resultados obtenidos de las simulaciones realizadas. Las gráficas obtenidas muestran el BER vs Eb/No considerando cada escenario, detector y modulación. Además, se realiza un análisis del costo computacional de cada detector considerando el tiempo que tardaron en ejecutarse los algoritmos respectivos de cada uno de ellos.

3.1. ESTABLECIMIENTO DE PARÁMETROS

Para el análisis de las gráficas obtenidas, se establecieron diferentes números de iteraciones. Cada simulación involucra la transmisión de 10 símbolos OFDM, tomando en cuenta 13 valores de SNR. Como se ha mencionado a lo largo del presente trabajo, los parámetros considerados se basan en el estándar IEEE 802.11p. Por lo cual se emplearon 64 subportadoras de las cuales 48 son de datos, 12 son nulas y 4 son pilotos acorde a lo presentado en la Tabla 1.5.

Los parámetros configurables para la simulación se muestran en la Tabla 2.2. Por lo que para cada simulación se puede ir configurando 3 parámetros: el escenario bajo el cual se plantea realizar la simulación, el detector a emplearse y la modulación. En un principio se empleó 100 iteraciones para cada simulación. Posteriormente se empleó 500 iteraciones. Sin embargo, se consideró que el número adecuado de iteraciones es de 5000 para poder obtener resultados más confiables. Por lo tanto, los resultados que se muestran en este capítulo emplean 5000 iteraciones.

Cada gráfica se la realizó de forma independiente para cada caso específico según los 3 parámetros configurables. Por lo tanto, se obtuvieron 72 gráficas en total. Para visualizar más fácilmente los resultados, se consolidaron las gráficas de modo que se pueda visualizar el rendimiento de cada uno de los detectores para un mismo escenario y una modulación determinada.

De forma simultánea, en cada simulación se obtuvo la gráfica y se guardó en una variable el promedio del tiempo que tardó en ejecutar cada detector.

3.2. GRÁFICAS DE BER vs Eb/No

La discusión de los resultados se la realiza tomando como base los detectores para de este modo compararlos y determinar el que muestra un mejor rendimiento. A continuación, se interpretan cada una de las gráficas para cada detección en un escenario y modulación determinados. Finalmente, se determina el detector de mejor rendimiento previo a un análisis de los resultados obtenidos.

3.2.1. RESULTADO ESCENARIO 1, MODULACIÓN QPSK

Para la obtención de la Figura 3.1 se considera una modulación QPSK y el escenario 1 que corresponde a *Rural* LOS descrito en el primer capítulo. Para este caso se emplearon 5000 iteraciones para cada uno de los 6 detectores. Las 6 curvas obtenidas en cada simulación fueron consolidadas en una sola gráfica presentada a continuación. En la gráfica cada detector es graficado con un color distinto.

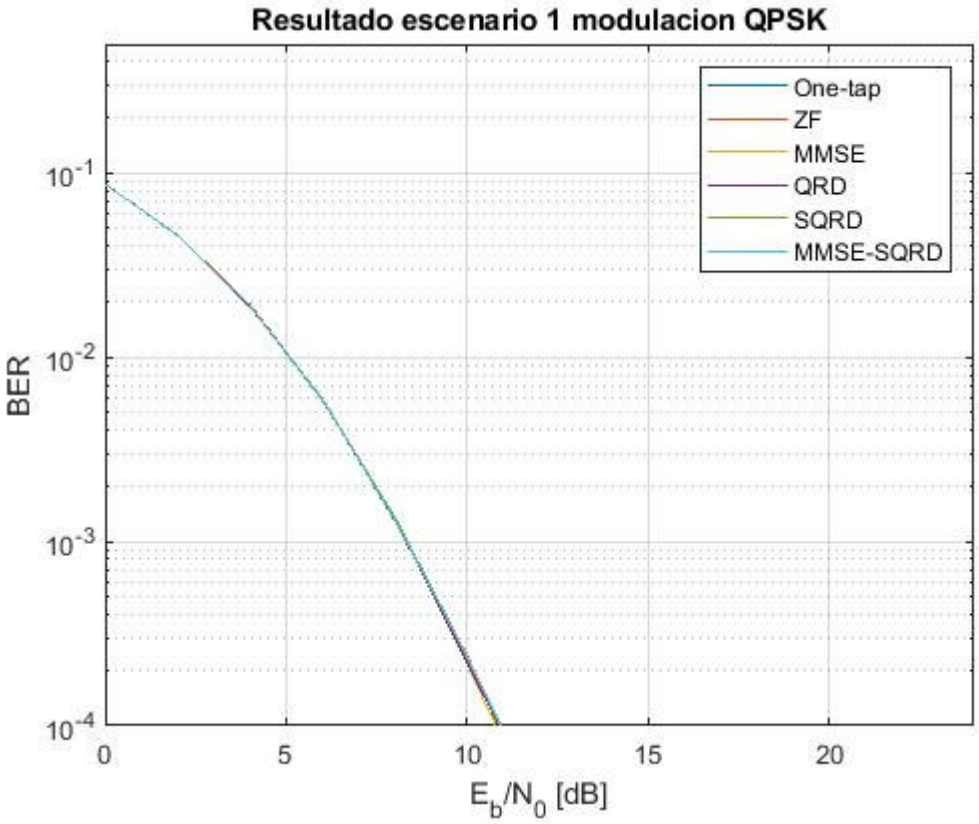

Figura 3.1 BER vs Eb/No del escenario 1, modulación QPSK

Para el análisis de los resultados se ha tomado arbitrariamente un Eb/No de 12dB para poder realizar la comparación en cada uno de los detectores. Como se puede observar en la Figura 3.2 el detector one-tap esta graficado con una curva azul oscura, el detector ZF se identifica por el color rojo y el detector MMSE esta graficado con un color amarillo.

Al tenerse un Eb/No de 12dB, se puede apreciar que para los detectores lineales el detector one-tap tiene un BER de 0.00003521, el detector ZF tiene un BER de 0.00003354 y el MMSE tiene un BER de 0.00003042.

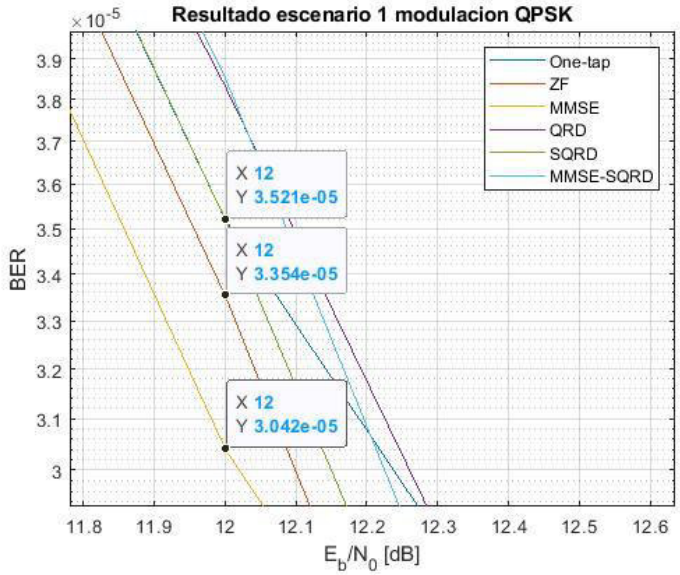

Figura 3.2 Acercamiento centrado en 12dB del escenario 1 con modulación QPSK (One-tap, ZF y MMSE)

Del mismo modo y como se muestra en la Figura 3.3 para los detectores no-lineales, el detector QRD se identifica por su color morado, SQRD de color verde y MMSE-SQRD de color celeste. Considerando los mismos 12dB, el detector QRD tiene un BER de 0.00003833, el SQRD tiene 0.00003521 y el MMSE-SQRD tiene 0.00003854.

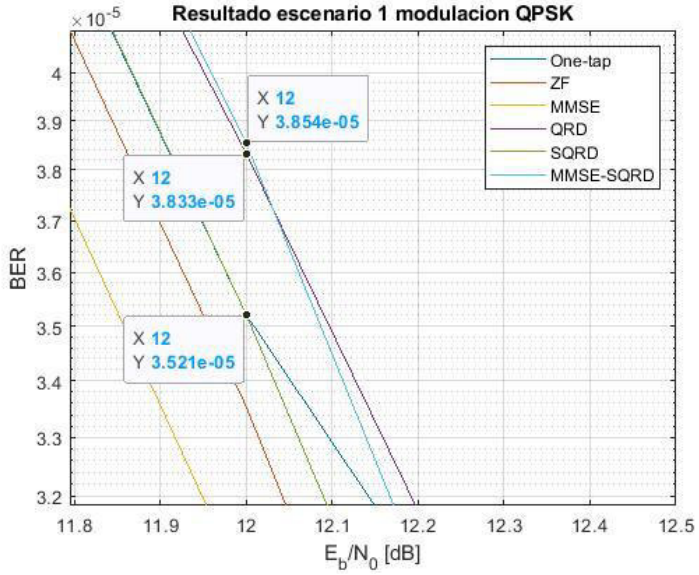

Figura 3.3 Acercamiento centrado en 12dB del escenario 1 con modulación QPSK (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.1 se presentan los valores del BER para todos los detectores. Como se puede observar, el rendimiento de todos los detectores es muy similar. Esto se debe a que el

escenario 1 es el menos severo ya que es puramente de Rician al presentar línea de vista y componentes NLOS de amplitudes pequeñas. Además, se tienen retardos mínimos y un desplazamiento Doppler poco significativo ya que el canal es muy poco variante en el tiempo.

Considerando los datos obtenidos, se puede ver que todos rodean un valor de BER de 0.00003 con diferencias poco significativas entre sí mismos. Esto puede explicarse por el número de iteraciones empleado, pues a pesar de emplearse 5000 iteraciones los resultados permanecen bastante similares entre sí porque la precisión no es suficiente.

Tabla 3.1 Valores de BER para el escenario 1 modulación QPSK

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.00003521
ZF	0.00003354
MMSE	0.00003042
QRD	0.00003833
SQRD	0.00003521
MMSE-SQRD	0.00003854

3.2.2. RESULTADO ESCENARIO 1, MODULACIÓN 16QAM

Para la obtención de la Figura 3.4 se considera una modulación 16QAM y el escenario 1 del mismo modo que en la sección anterior. Para este caso también se emplearon 5000 iteraciones para cada uno de los 6 detectores.

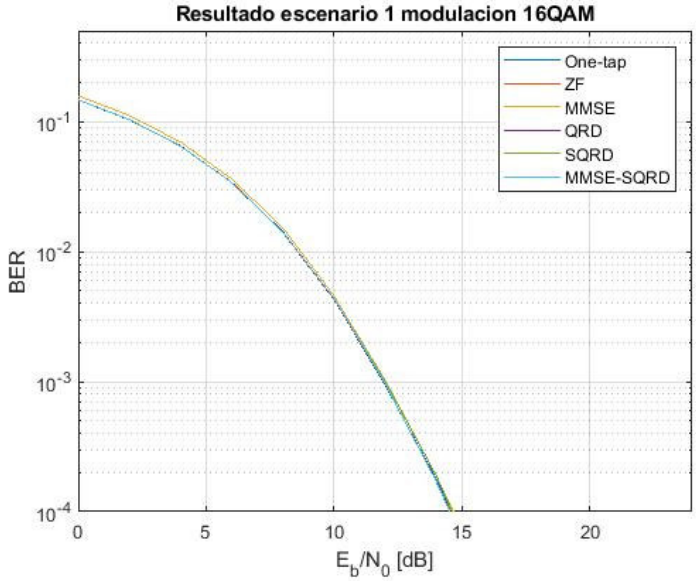

Figura 3.4 BER vs Eb/No del escenario 1, modulación 16QAM

Como se puede observar en la Figura 3.5, cada detector está representado por un color específico. Considerando un valor de Eb/No de 12dB, el detector one-tap tiene un BER de 0.0009431 y corresponde al color azul oscuro. El detector ZF tiene un BER de 0.0009626 identificado por la curva en color rojo y el MMSE tiene un BER de 0.001029 en su curva de color amarillo.

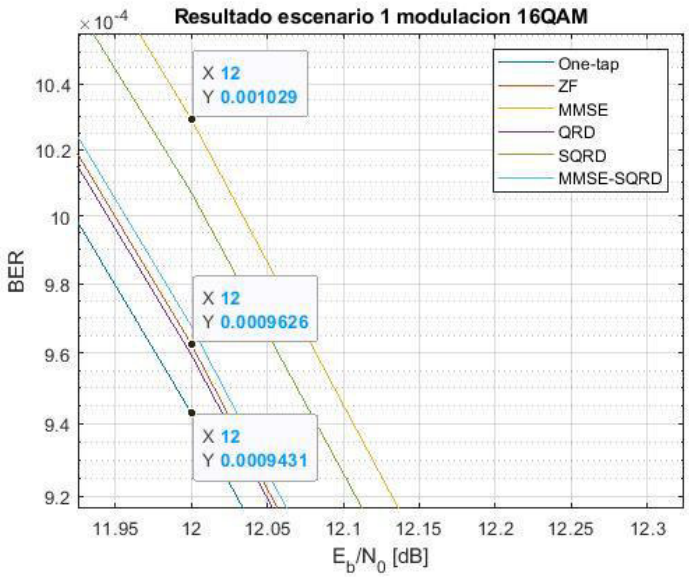

Figura 3.5 Acercamiento centrado en 12dB del escenario 1 con modulación 16QAM (One-tap, ZF y MMSE)

Como se muestra en la Figura 3.6 para los detectores no-lineales, el detector QRD tiene un BER de 0.0009596, el SQRD tiene 0.001007 y el MMSE-SQRD tiene 0.0009679.

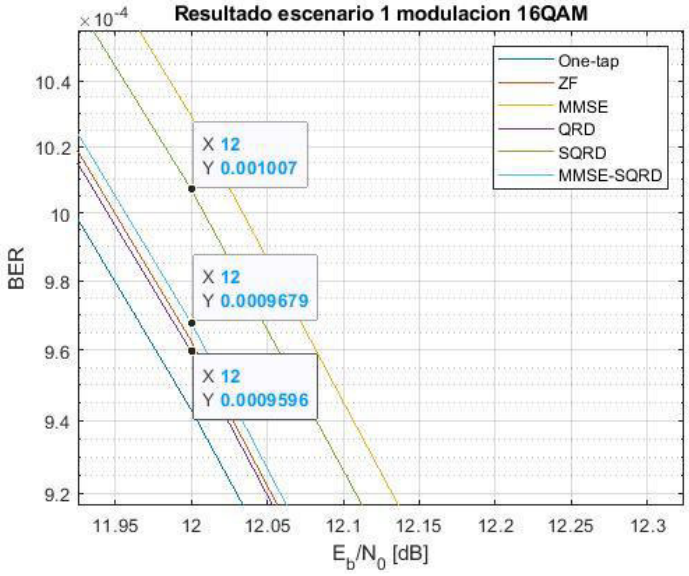

Figura 3.6 Acercamiento centrado en 12dB del escenario 1 con modulación 16QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.2 se muestran los valores del BER obtenidos para todos los detectores. Analizando los resultados, se puede observar que para un mismo Eb/No todos los detectores presentan un BER similar que rodea el valor de 0.0009 y 0.001. Del mismo modo que en la sección anterior, todos tienen un buen rendimiento dado que el canal en el escenario 1 no presenta componentes NLOS significativas, retardos considerables, ni una variación en el tiempo representativa que permita observar la diferencia entre un detector y otro.

Las variaciones entre uno y otro detector son mínimas. Esto puede explicarse por el número de iteraciones empleado, pues a pesar de emplearse 5000 iteraciones la precisión no es suficiente.

Tabla 3.2 Valores de BER para el escenario 1 modulación 16QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.0009431
ZF	0.0009626
MMSE	0.001029
QRD	0.0009596
SQRD	0.001007
MMSE-SQRD	0.0009679

3.2.3. RESULTADO ESCENARIO 1, MODULACIÓN 64QAM

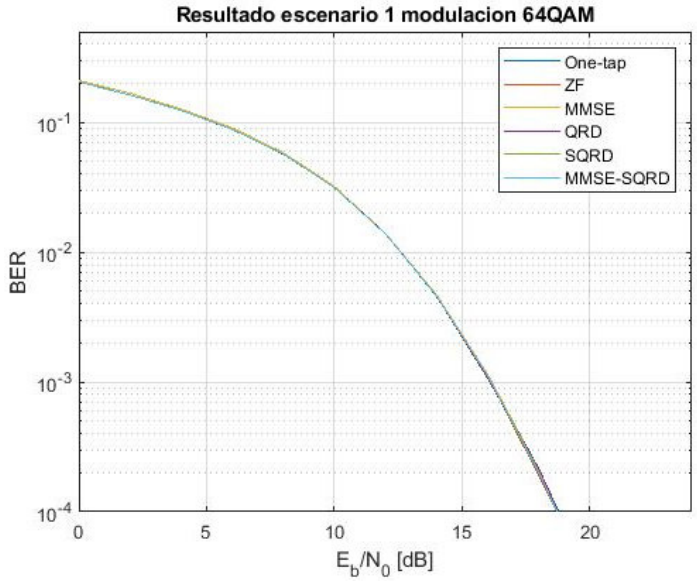

Figura 3.7 BER vs Eb/No del escenario 1, modulación 64QAM

Para obtener la Figura 3.7 se considera la modulación 64QAM para el escenario *Rural* LOS. Se realizaron 5000 iteraciones para cada uno de los detectores bajo el mismo escenario y modulación del mismo modo que para las dos secciones anteriores.

Tomando como referencia 12dB, se tiene que el detector one-tap presenta 0.013907 de BER, el detector ZF, presenta 0.01392 como BER y el detector MMSE presenta 0.01408 como BER. Estos resultados se pueden apreciar en la Figura 3.8.

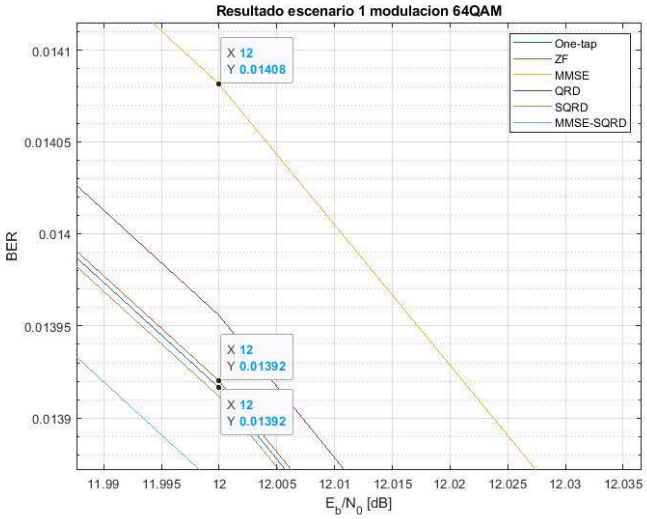

Figura 3.8 Acercamiento centrado en 12dB del escenario 1 con modulación 64QAM (One-tap, ZF y MMSE)

En cuanto a los detectores no-lineales se tiene al QRD con un BER de 0.01396. Mientras que el SQRD tiene un BER de 0.01391. Finalmente, el detector MMSE-SQRD tiene un BER de 0.01386. Estos resultados se pueden apreciar en la Figura 3.9.

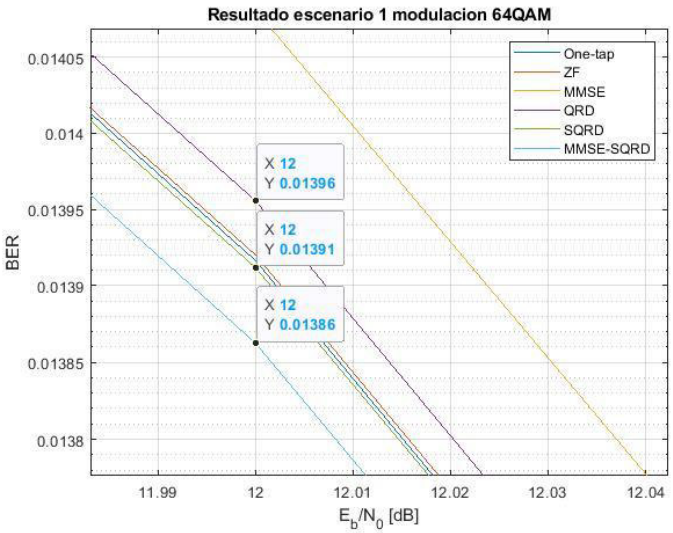

Figura 3.9 Acercamiento centrado en 12dB para del escenario 1 con modulación 64QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.3 se muestran los valores del BER de todos los detectores para un mismo Eb/No. Como se puede apreciar, el rendimiento de los detectores para este escenario es bastante similar. El valor del BER rodea a 0.014. Del mismo modo que en las dos secciones anteriores, esto se debe a que el canal en el escenario 1 es el menos severo como ya se ha explicado anteriormente.

El BER obtenido no varía significativamente de un detector con respecto al otro. Esto puede explicarse también porque la precisión no es suficiente con el número de iteraciones empleado, pues a pesar de emplearse 5000 iteraciones los resultados permanecen bastante similares entre sí.

Tabla 3.3 Valores de BER para el escenario 1 modulación 64QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.013907
ZF	0.01392
MMSE	0.01408
QRD	0.01396
SQRD	0.01391
MMSE-SQRD	0.01386

3.2.4. RESULTADO ESCENARIO 2, MODULACIÓN QPSK

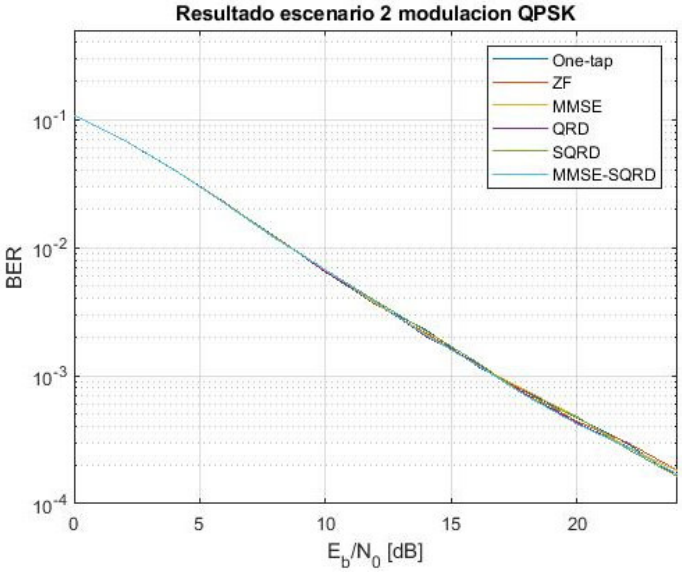

Figura 3.10 BER vs Eb/No del escenario 2, modulación QPSK

Para obtener la gráfica Figura 3.10 se configuraron como parámetros el escenario 2 correspondiente a *Urban Approaching* LOS y una modulación QPSK. Para cada uno de los

6 detectores se ejecutaron 5000 iteraciones. Todas estas gráficas se consolidaron en la figura a continuación donde cada detector está representado por una curva de color diferente.

Como se puede ver en la Figura 3.11, tomando como referencia los 12dB, se tiene que el detector one-tap presenta un BER de 0.003592. El detector ZF presenta un BER de 0.003617. El detector MMSE, presenta un BER de 0.003658.

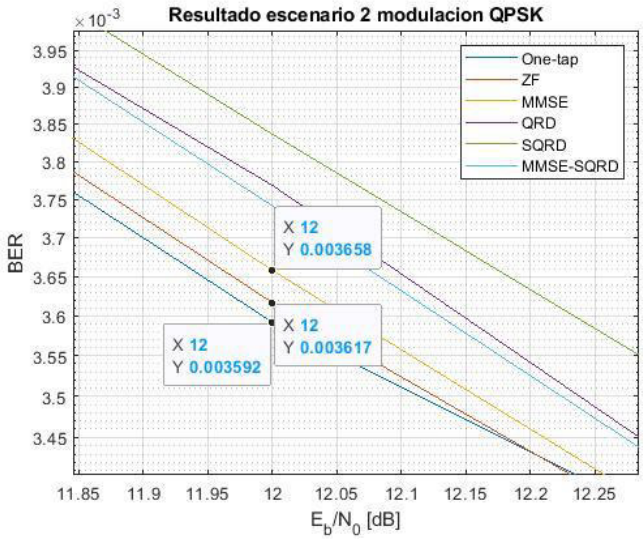

Figura 3.11 Acercamiento centrado en 12dB del escenario 2 con modulación QPSK (One-tap, ZF y MMSE)

Como se puede ver en la Figura 3.12. El detector QRD presenta un BER de 0.003768. El detector SQRD presenta un BER de 0.003836. Finalmente, el detector MMSE-SQRD presenta un BER de 0.003741.

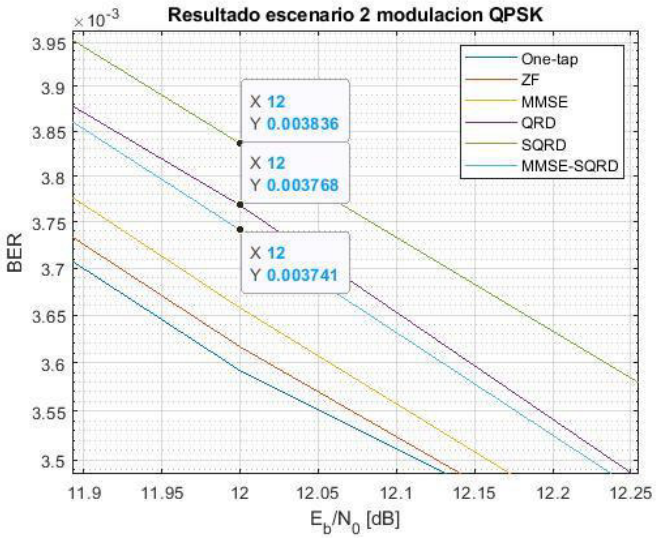

Figura 3.12 Acercamiento centrado en 12dB del escenario 2 con modulación QPSK (QRD, SQRD y MMSE-SQRD)

Como se puede ver, en la Tabla 3.4 se presenta el BER obtenido para todos los detectores considerando un mismo Eb/No. El valor del BER entre todos ellos es muy similar ya que oscila entre los 0.0036 y 0.0037. Sin embargo, se puede ver que todos los detectores presentan un rendimiento bastante bueno considerando también que se realizaron 5000 iteraciones.

Comparando estos resultados con los obtenidos para el escenario 1 y modulación QPSK, se puede ver que el BER para este escenario incrementa debido a que en el escenario 2 las componentes NLOS tienen una mayor amplitud a diferencia del escenario 1, por lo que este escenario es un poco más severo. Del mismo modo, se presentan retardos más significativos y un canal ligeramente más variante en el tiempo que el del escenario 1.

Tabla 3.4 Valores de BER para el escenario 2 modulación QPSK

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.003592
ZF	0.003617
MMSE	0.003658
QRD	0.003768
SQRD	0.003836
MMSE-SQRD	0.003741

3.2.5. RESULTADO ESCENARIO 2, MODULACIÓN 16QAM

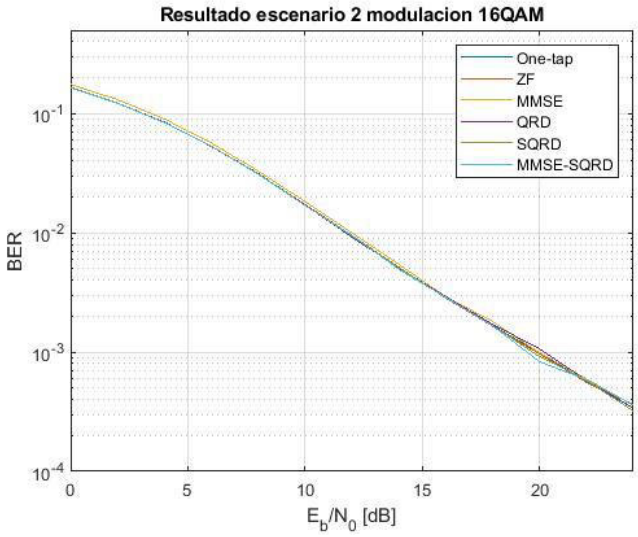

Figura 3.13 BER vs Eb/No del escenario 2, modulación 16QAM

Para obtener la Figura 3.13 se empleó el escenario *Urban Approaching* LOS con una modulación 16QAM. Para este caso se realizaron 5000 iteraciones para cada detector. Se

obtuvieron las 6 curvas y se consolidaron las mismas en una sola gráfica donde cada detector se identifica por un color diferente de curva.

Como se puede ver en Figura 3.14. Para un Eb/No de 12dB se obtuvo un BER de 0.009159 en el caso del detector one-tap. Para el detector ZF se obtuvo un BER de 0.009522. El detector MMSE presenta un BER de 0.009994.

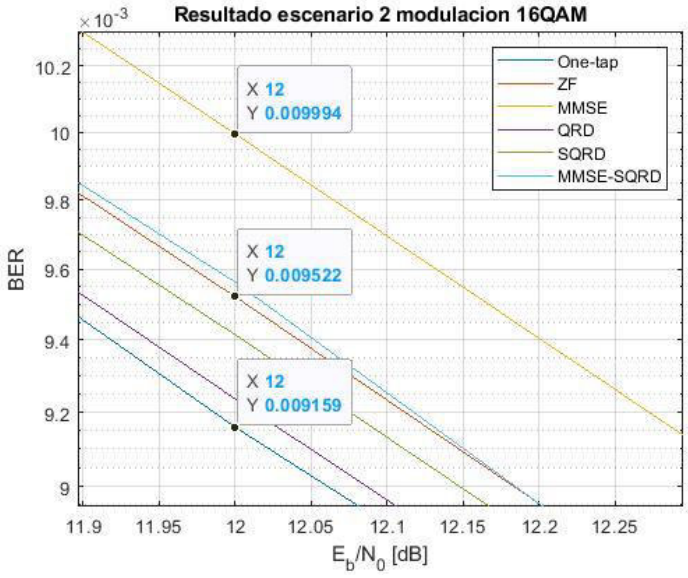

Figura 3.14 Acercamiento centrado en 12dB del escenario 2 con modulación 16QAM (One-tap, ZF y MMSE)

Por otra parte, como se muestra en la Figura 3.15, el detector QRD presenta un BER de 0.009237. El detector SQRD obtuvo un BER de 0.009415. Finalmente, el detector MMSE-SQRD obtuvo un BER de 0.009564.

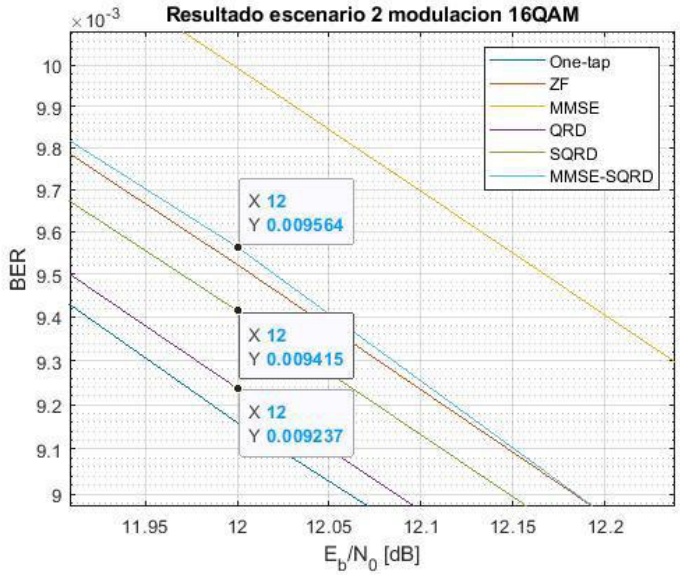

Figura 3.15 Acercamiento centrado en 12dB del escenario 2 con modulación 16QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.5 se presentan los valores del BER obtenidos para un mismo Eb/No para todos los detectores implementados. Como se puede observar, todos los detectores presentan un BER similar entre ellos.

El valor del BER oscila entre 0.0092 y 0.0099. La diferencia entre cada detector es mínima, por lo que todos presentan un rendimiento bastante bueno considerando las 5000 iteraciones empleadas. Del mismo modo que en la sección anterior, esto se debe a que el canal para el escenario 2 no es muy variante en el tiempo, por lo que no se puede observar una diferencia significativa entre uno y otro detector.

Realizando una comparación entre estos resultados y los obtenidos en el escenario 1 con una modulación 16QAM. Se puede ver que el BER para el escenario 2 incrementa debido a que en este caso se presentan componentes NLOS con una potencia más significativa, retardos considerables y un desplazamiento Doppler que hace que el canal sea ligeramente más variante en el tiempo a diferencia del canal en el escenario 1.

Tabla 3.5 Valores de BER para el escenario 2 modulación 16QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.009159
ZF	0.009522
MMSE	0.009994
QRD	0.009237
SQRD	0.009415
MMSE-SQRD	0.009564

3.2.6. RESULTADO ESCENARIO 2, MODULACIÓN 64QAM

La Figura 3.16 se obtuvo empleando el escenario 2 que corresponde a *Urban Approaching* LOS con una modulación 64QAM. Se emplearon 5000 iteraciones para obtener la curva para cada uno de los 6 detectores. Cada una de las curvas obtenidas fueron consolidadas en una misma gráfica para poder visualizar los resultados de mejor manera. En esta gráfica cada detector se encuentra identificado mediante un color distinto, como se puede observar a continuación.

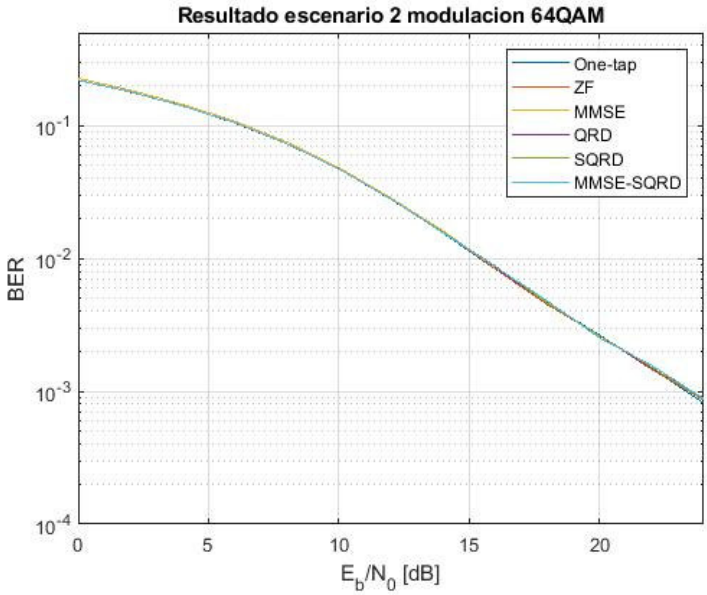

Figura 3.16 BER vs Eb/No del escenario 2, modulación 64QAM

Considerando como referencia un Eb/No de 12dB, se obtuvo como resultado un BER de 0.02828 para el detector one-tap. Para el detector ZF se obtuvo un BER de 0.02801. Para el detector MMSE se obtuvo un BER de 0.02857. Estos resultados se los puede observar en la Figura 3.17.

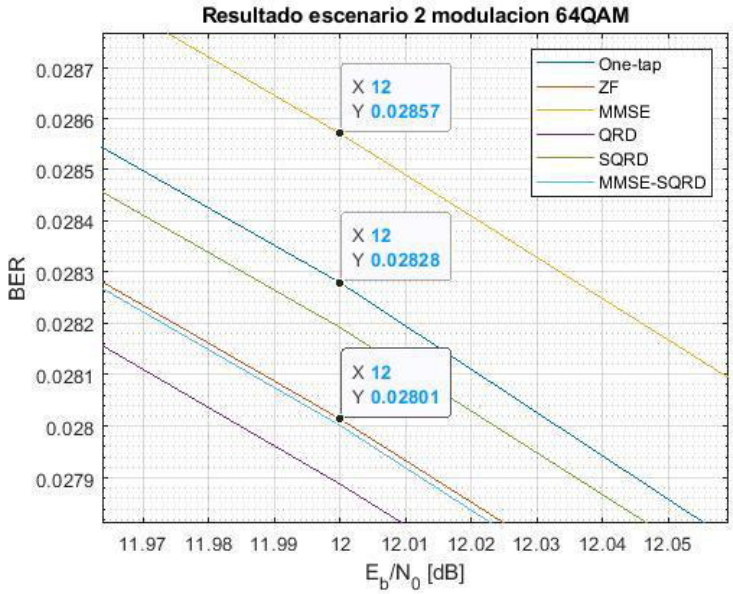

Figura 3.17 Acercamiento centrado en 12dB del escenario 2 con modulación 64QAM (One-tap, ZF y MMSE)

Como se puede ver en la Figura 3.18. Para el detector QRD se obtuvo un BER de 0.02789. Para el detector SQRD se obtuvo un BER de 0.02819. Finalmente, para el detector MMSE-SQRD se obtuvo un BER de 0.028.

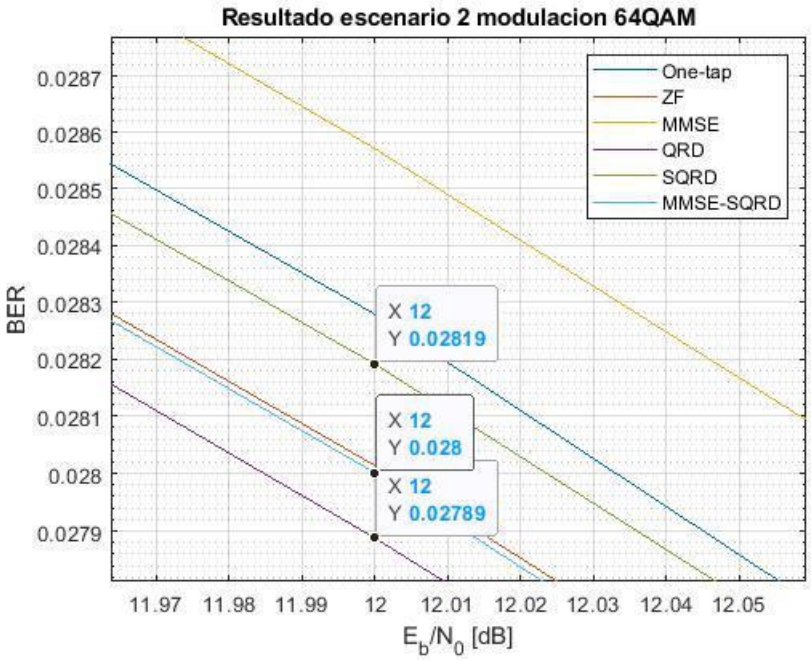

Figura 3.18 Acercamiento centrado en 12dB del escenario 2 con modulación 64QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.6 se muestran los valores del BER obtenido para todos los detectores implementados considerando un Eb/No especifico. Como se puede apreciar, todos obtuvieron un BER muy similar cuyo valor oscila entre 0.02789 y 0.02857. Las variaciones entre un detector respecto de otro son poco significativas considerando que se emplearon 5000 iteraciones. Del mismo modo que para las dos secciones anteriores, se debe tomar en cuenta que el escenario 2 presenta condiciones más severas que las que se consideraron en el escenario 1.

Al comparar estos resultados con los obtenido en el escenario 1 con modulación 64QAM, se puede ver un incremento de más o menos 0.01. Esto se debe a la diferencia que se tiene en el PDP. Puesto que en el escenario 2 se trata de un ambiente urbano en donde las componentes NLOS son más significativas que en el escenario 1, poseen retardos mayores y el canal es ligeramente más cambiante en el tiempo a diferencia del escenario 1.

Tabla 3.6 Valores de BER para el escenario 2 modulación 64QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.02828
ZF	0.02801
MMSE	0.02857
QRD	0.02789
SQRD	0.02819
MMSE-SQRD	0.028

3.2.7. RESULTADO ESCENARIO 3, MODULACIÓN QPSK

Para obtener la Figura 3.19 se consideró el escenario 3 que corresponde a *Highway LOS* y una modulación QPSK. Este escenario es más severo a diferencia de los dos escenarios anteriores. Se realizaron 5000 iteraciones para cada detector, cada gráfica obtenida fue consolidada en una misma gráfica identificando cada detector por un color diferente.

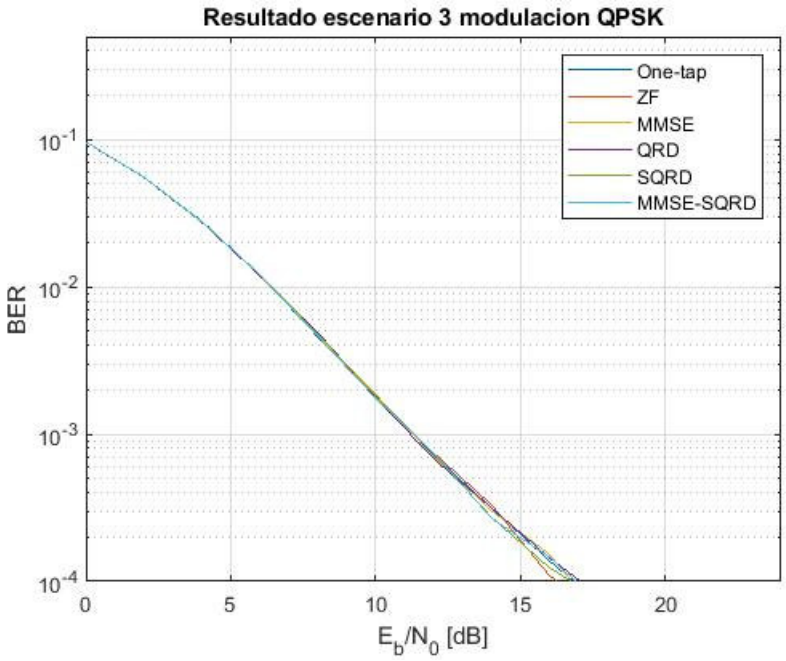

Figura 3.19 BER vs Eb/No del escenario 3, modulación QPSK

Usando como referencia 12dB de Eb/No, con el detector one-tap se obtuvo un BER de 0.0007187. El detector ZF presento un BER de 0.0007481. El detector MMSE obtuvo un BER de 0.0007027. Estos resultados se muestran en la Figura 3.20.

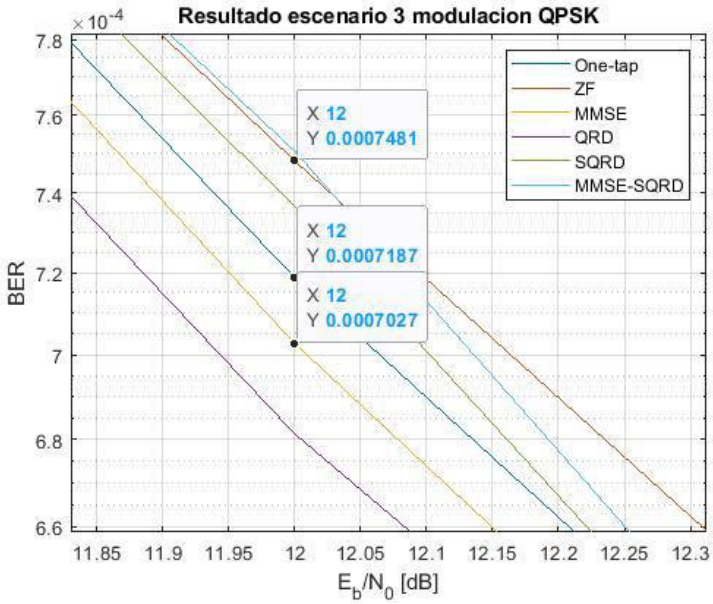

Figura 3.20 Acercamiento centrado en 12dB del escenario 3 con modulación QPSK (One-tap, ZF y MMSE

Para los 12dB se tiene un BER de 0.0006812 con el detector QRD. El detector SQRD obtuvo un BER de 0.0007365. Finalmente, el detector MMSE-SQRD obtuvo un BER de 0.0007506. Estos resultados se muestran en la Figura 3.21.

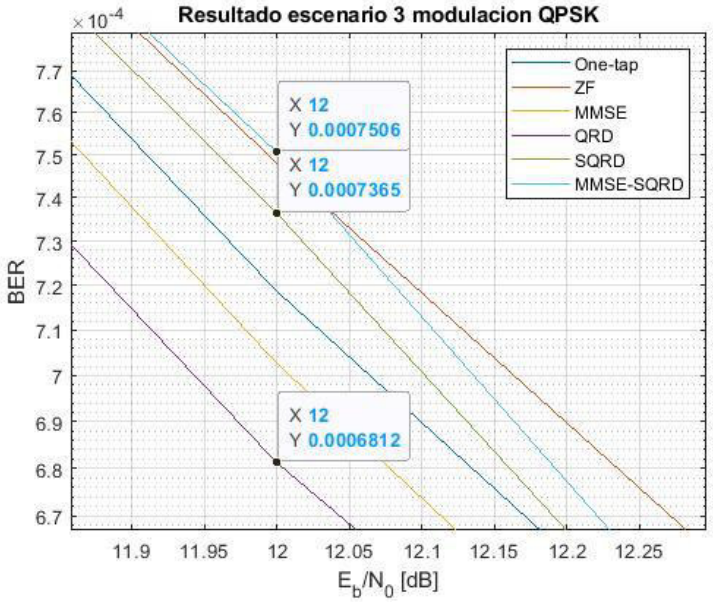

Figura 3.21 Acercamiento centrado en 12dB del escenario 3 con modulación QPSK (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.7 se presentan los valores del BER obtenido para cada uno de los 6 detectores considerando un mismo Eb/No. Como se puede ver, el rendimiento de todos los

detectores es bastante similar dado que presentan un BER que oscila entre 0.0006812 y 0.0007506. Esto se debe a que el canal para el escenario 3 presenta una variación en el tiempo poco notoria, por lo que no se puede observar mayor diferencia al emplear un detector u otro.

Comparando estos resultados con los obtenidos en los escenarios 1 y 2 con la modulación QPSK, se puede notar un incremento del BER. Esto se explica porque el escenario 3 presenta condiciones más severas al ser ligeramente más variante en el tiempo que los dos escenarios anteriores. Sin embargo, el valor del BER se mantiene pequeño por lo que el rendimiento de todos los detectores implementados es bastante bueno.

Tabla 3.7 Valores de BER para el escenario 3 modulación QPSK

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.0007187
ZF	0.0007481
MMSE	0.0007027
QRD	0.0006812
SQRD	0.0007365
MMSE-SQRD	0.0007506

3.2.8. RESULTADO ESCENARIO 3, MODULACIÓN 16QAM

Para obtener la Figura 3.22 se consideró el escenario 3 y una modulación 16QAM. Se emplearon 5000 iteraciones para cada uno de los 6 detectores. Todas estas curvas se consolidaron en una sola gráfica.

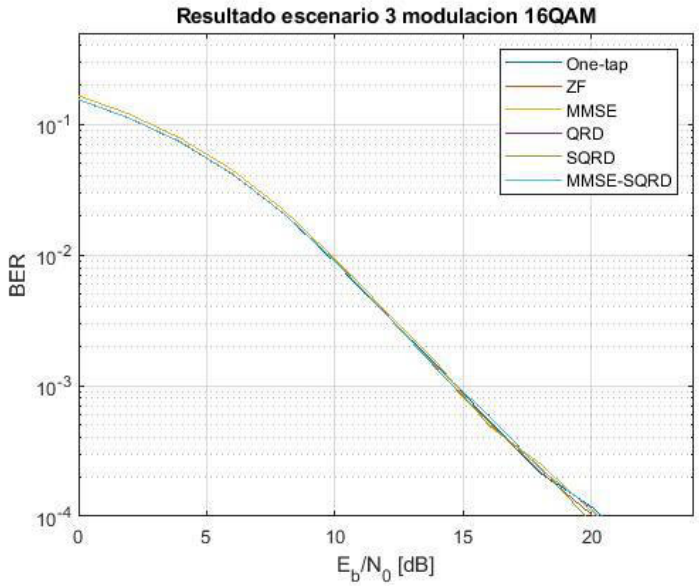

Figura 3.22 BER vs Eb/No del escenario 3, modulación 16QAM

Para 12dB de Eb/No se obtuvo un BER de 0.003499 con el detector one-tap. Con el detector ZF se obtuvo un BER de 0.003566. El detector MMSE obtuvo un BER de 0.003712 como se muestra en la Figura 3.23.

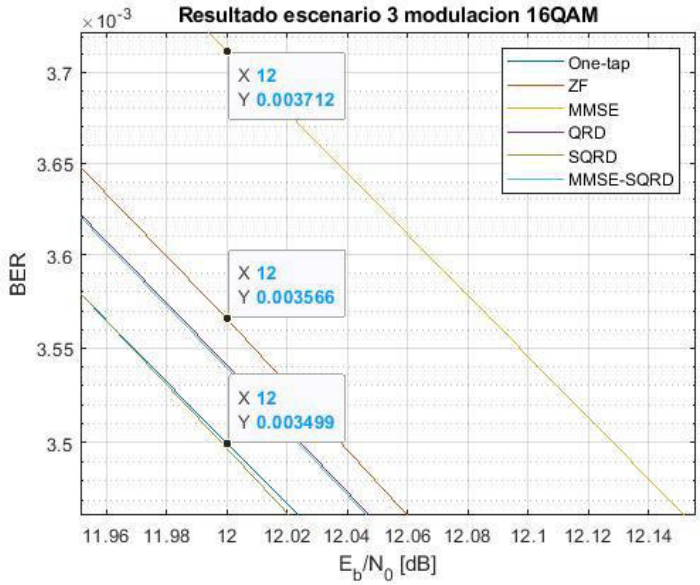

Figura 3.23 Acercamiento centrado en 12dB del escenario 3 con modulación 16QAM (One-tap, ZF y MMSE)

Con el detector QRD se obtuvo un BER de 0.003541. El detector SQRD obtuvo un 0.003497. El detector MMSE-SQRD obtuvo un BER 0.003539 como se muestra en la Figura 3.24.

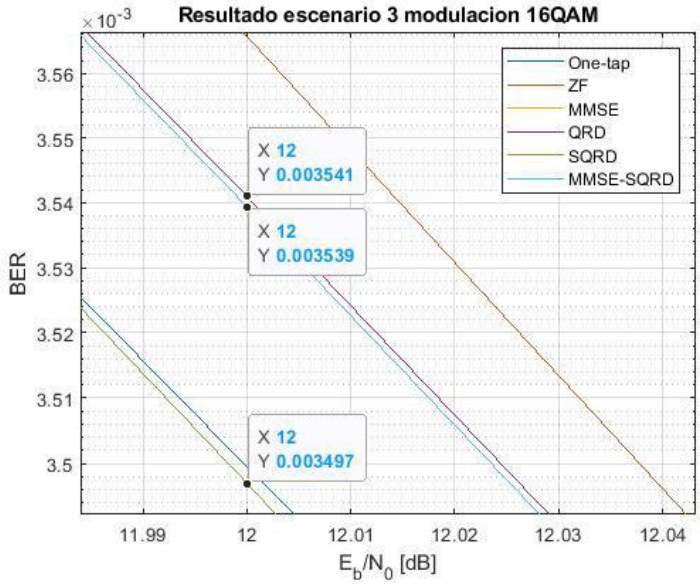

Figura 3.24 Acercamiento centrado en 12dB del escenario 3 con modulación 16QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.8 se muestra los valores de BER obtenidos por cada uno de los 6 detectores para un mismo valor de Eb/No. Como se puede apreciar, todos presentan un rendimiento bastante similar con un valor de BER que oscila entre 0.003497 y 0.003712. Las variaciones entre un detector respecto del otro son poco significativas. Esto se debe a que el escenario 3 presenta un canal poco variante en el tiempo.

Tabla 3.8 Valores de BER para el escenario 3 modulación 16QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.003499
ZF	0.003566
MMSE	0.003712
QRD	0.003541
SQRD	0.003497
MMSE-SQRD	0.003539

Si comparamos estos resultados con los obtenidos en los escenarios 1 y 2 para la misma modulación 16QAM, se puede notar un incremento en el BER. Esto se debe a que a pesar de que el escenario 3 presenta un canal muy poco variante en el tiempo, es ligeramente más variante a diferencia de los otros dos escenarios. Sin embargo, el valor del BER se mantiene aceptable por lo que los 6 detectores poseen un rendimiento bastante bueno.

3.2.9. RESULTADO ESCENARIO 3, MODULACIÓN 64QAM

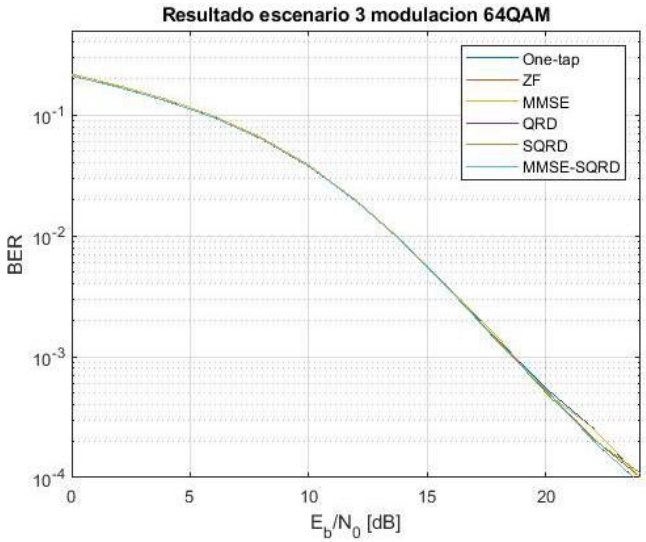

Figura 3.25 BER vs Eb/No del escenario 3, modulación 64QAM

Para obtener la Figura 3.25 se consideró el escenario 3 y una modulación 64QAM. Cada uno de los detectores fueron sometidos a estas condiciones empleando 5000 iteraciones.

Las gráficas de cada uno de los detectores se consolidaron en la gráfica presentada a continuación, se diferencia entre sí porque se emplea un color distinto para cada detector.

Como se puede ver en la Figura 3.26, para un Eb/No de 12dB el detector one-tap obtuvo un BER de 0.01932. El ZF obtuvo un BER de 0.01976. El MMSE obtuvo un BER de 0.01974.

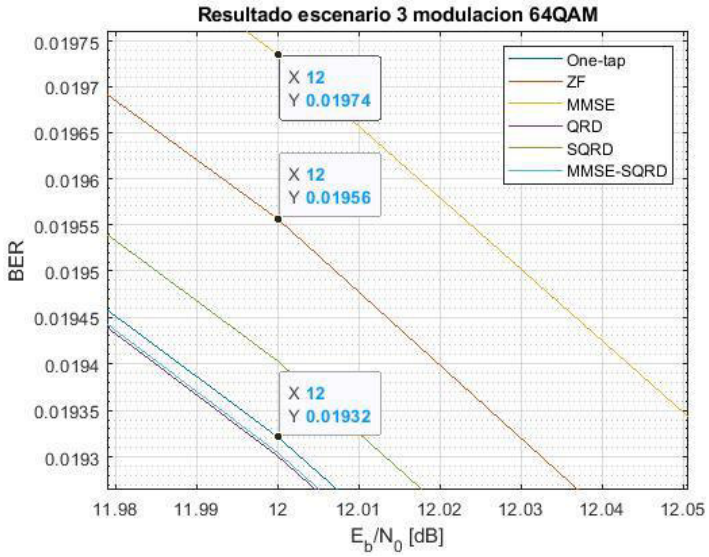

Figura 3.26 Acercamiento centrado en 12dB del escenario 3 con modulación 64QAM (One-tap, ZF y MMSE)

Como se puede ver en la Figura 3.27, el detector QRD obtuvo un BER de 0.0193. El detector SQRD obtuvo un BER de 0.0194. Finalmente, el detector MMSE-SQRD obtuvo un BER de 0.0193025.

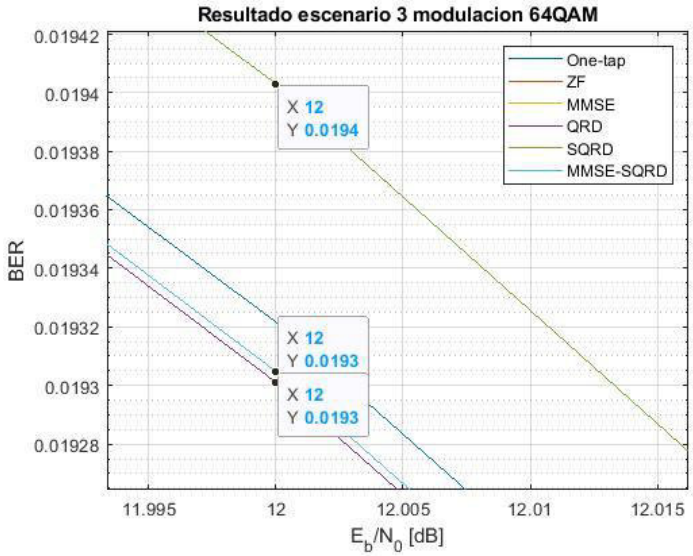

Figura 3.27 Acercamiento centrado en 12dB del escenario 3 con modulación 64QAM (QRD, SQRD y MMSE-SQRD)

En la Tabla 3.9 se presentan todos los valores del BER obtenido para cada uno de los detectores implementados. Como se puede apreciar, se presenta un valor de BER bastante similar entre ellos. El valor del BER oscila entre 0.0193025 y 0.0194 con ligeras variaciones entre un detector respecto de otro. Esto se debe a que el escenario 3 presenta un canal poco variante en el tiempo, por lo que no se puede evidenciar una diferencia significativa entre un detector y otro.

Sin embargo, si se compara estos resultados con los obtenidos en los escenarios 1 y 2 para la misma modulación, se puede ver un incremento del BER. Esto se debe a que, si bien el escenario 3 no es muy variante en el tiempo, si presenta un ligero aumento en cuanto a su variación en el tiempo a diferencia de los otros dos escenarios.

Tabla 3.9 Valores de BER para el escenario 3 modulación 64QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.01932
ZF	0.01976
MMSE	0.01974
QRD	0.0193
SQRD	0.0194
MMSE-SQRD	0.0193025

3.2.10. RESULTADO ESCENARIO 4, MODULACIÓN QPSK

Una vez obtenidos los resultados de los tres escenarios anteriores, dado que en estos no se presenta una variación en el tiempo significativa y no se pudo apreciar una diferencia en los rendimientos con el BER obtenido. Se decidió alterar la frecuencia Doppler del escenario *Highway* NLOS original. Con el objetivo de simular un canal altamente variante en el tiempo, se modificaron las frecuencias Doppler como se presenta en la Tabla 3.10. De este modo, se obtiene el escenario 4, llamado *Highway* NLOS modificado. Con esta modificación se obtuvieron resultados donde se puede notar de forma más evidente las diferencias entre el rendimiento de uno y otro detector.

Tabla 3.10 *Highway* NLOS modificado

	Tap1	***Tap2***	***Tap3***	***Tap4***	***Units***
Power	0	-2	-5	-7	dB
Delay	0	200	433	700	ns
Doppler	0	100000	-50000	150000	Hz
Profile	*Static*	*HalfBT*	*HalfBT*	*HalfBT*	

Para obtener la Figura 3.28 se empleó el escenario, *Highway* NLOS modificado, con una modulación QPSK. Este es el escenario más severo, puramente Rayleigh, dado que no se tiene línea de vista y es altamente variante en el tiempo. En esta simulación se emplearon 5000 iteraciones para cada detector. Las curvas obtenidas se consolidaron en la gráfica presentada a continuación, cada detector se diferencia por un color distinto.

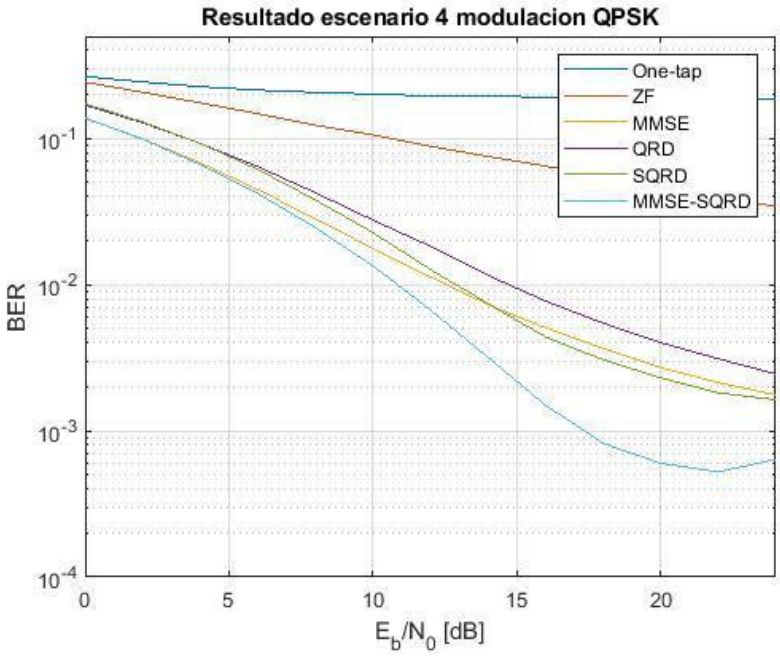

Figura 3.28 BER vs Eb/No del escenario 4, modulación QPSK

En la Tabla 3.11 se presentan los valores del BER obtenidos por cada uno de los detectores. Tomando en cuenta los detectores lineales, se puede ver que el de mejor BER es el MMSE ya que es el valor más pequeño comparado con el del one-tap y ZF. Lo que demuestra que el MMSE es el detector con mejor rendimiento entre los detectores lineales.

Por otra parte, los detectores no-lineales presentan una mejor detección respecto de lo detectores lineales ya que presentan valores de BER menores. El detector con mejor BER de los detectores no-lineales es el MMSE-SQRD ya que el valor obtenido de BER es el menor. De este modo, el detector con mejor rendimiento de todos es el MMSE-SQRD. La diferencia entre cada uno de los detectores para este caso es bastante notoria dado que las condiciones de este escenario involucran un canal altamente variante en el tiempo. El detector MMSE-SQRD es el más adecuado para estas condiciones del canal. Cabe destacar que su mejor rendimiento se debe a que realiza una descomposición QR donde considera los errores por capas. Además, emplea una matriz del canal extendida en la que se considera también la varianza del ruido.

Tabla 3.11 Valores de BER para el escenario 4 modulación QPSK

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.1952
ZF	0.08845
MMSE	0.01131
QRD	0.01834
SQRD	0.01271
MMSE-SQRD	0.006723

3.2.11. RESULTADO ESCENARIO 4, MODULACIÓN 16QAM

Para obtener la Figura 3.29 se empleó el escenario *Highway* NLOS modificado, con una modulación 16QAM. Se emplearon 5000 iteraciones para cada detector.

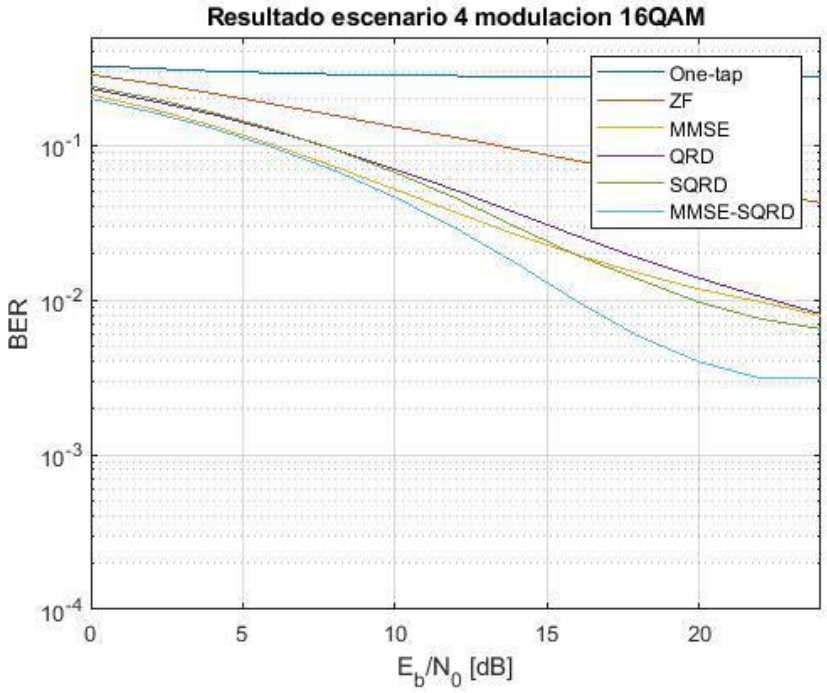

Figura 3.29 BER vs Eb/No del escenario 4, modulación 16QAM

En la Tabla 3.12 se presentan los resultados del BER obtenido para todos los detectores considerando un mismo Eb/No. Como se puede observar, en lo que respecta a los detectores lineales el mejor BER lo obtuvo el detector MMSE. MMSE presenta un valor de BER menor de BER a diferencia del one-tap y ZF. Por lo tanto, el detector MMSE realiza la mejor detección de los detectores lineales por lo que su rendimiento es el mejor. Esto se debe a que a diferencia de los otros dos, MMSE considera la varianza del ruido en su algoritmo.

En cambio, en cuanto a los detectores no-lineales, se puede observar que QRD, SQRD y MMSE-SQRD presentan un valor de BER menor al que presentan los detectores lineales. El detector que presenta un mejor BER, es decir, el que tiene un menor BER es el MMSE-SQRD. A este le sigue el SQRD y el QRD que son detectores no-lineales también diseñados para canales muy variantes en el tiempo como lo es el escenario *Highway* NLOS modificado. Sin embargo, el de mejor rendimiento es el MMSE-SQRD ya que a diferencia del SQRD toma en cuenta la varianza del ruido en su algoritmo.

Tabla 3.12 Valores de BER para el escenario 4 modulación 16QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.2812
ZF	0.1124
MMSE	0.03684
QRD	0.05126
SQRD	0.04564
MMSE-SQRD	0.02909

3.2.12. RESULTADO ESCENARIO 4, MODULACIÓN 64QAM

Para obtener la Figura 3.30, se empleó el escenario *Highway* NLOS modificado con una modulación 64QAM. Cada uno de los detectores fueron sometidos a estas condiciones empleando 5000 iteraciones.

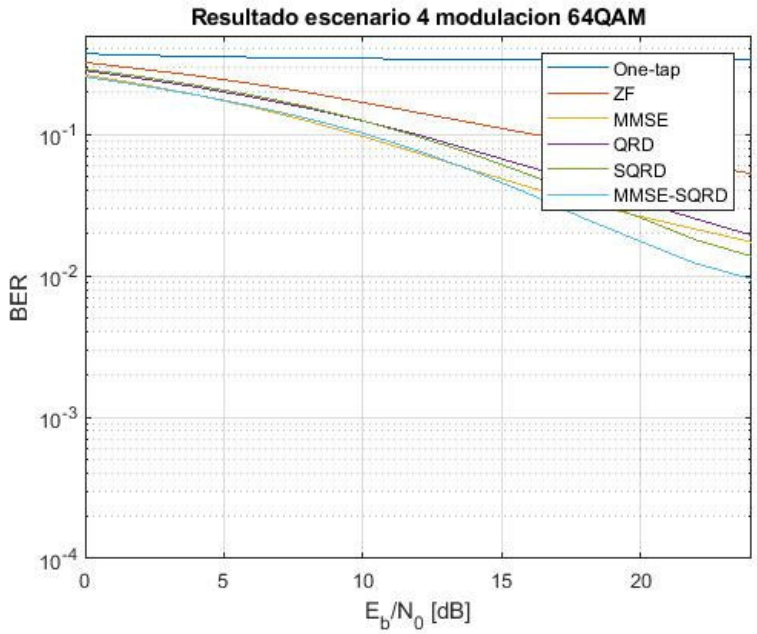

Figura 3.30 BER vs Eb/No del escenario 4, modulación 64QAM

En la Tabla 3.13 se presentan los resultados del BER obtenido para cada uno de los detectores empleando un mismo Eb/No. En cuanto a los detectores lineales, se puede ver que el detector de mejor BER es el MMSE, a este le sigue el ZF y por último el one-tap. Esto se debe a que one-tap emplea un algoritmo simple que resulta eficiente solamente para canales que no varían en el tiempo. Por lo tanto, el MMSE realiza una mejor detección y por ende tiene un mejor rendimiento en cuanto a los detectores lineales.

Sin embargo, los detectores no-lineales obtuvieron valores de BER mucho mejores que los de los detectores lineales. El de menor BER es el MMSE-SQRD, seguido del SQRD y el QRD. De todos los detectores el que presenta un mejor rendimiento es el MMSE-SQRD con un BER mínimo. Si bien estos tres detectores no-lineales emplean una descomposición QR, el MMSE-SQRD tiene una gran ventaja al considerar la varianza del ruido. Como se puede apreciar, la diferencia en los valores del BER entre cada uno de los detectores es más marcada debido a que el canal para este escenario es altamente variante en el tiempo.

Tabla 3.13 Valores de BER para el escenario 4 modulación 64QAM

DETECTOR	BER CORRESPONDIENTE A UN Eb/No=12dB
One-tap	0.3423
ZF	0.1425
MMSE	0.07375
QRD	0.09921
SQRD	0.09641
MMSE-SQRD	0.07668

Para mejor apreciación de los resultados se presenta la Tabla 3.14. Como se puede ver el valor del BER para un mismo escenario y un mismo detector, pero diferente modulación, incrementa directamente con la complejidad de la modulación. Por ejemplo, en el escenario 1 con modulación QPSK se tiene un BER insignificante. Sin embargo, para el mismo escenario con la modulación 16QAM se tiene un incremento del BER y evidentemente en 64QAM se tiene un BER mucho más significativo.

Además, al realizar una comparación entre escenarios, se puede ver que a medida que el canal presenta mayor variación en el tiempo, los detectores obtienen un BER más significativo. En cambio, cuando el escenario no es muy variante en el tiempo el BER obtenido por todos los detectores es similar. No obstante, en el escenario 4 dado que el canal es altamente variante en el tiempo se pueden notar claramente las diferencias entre cada uno de los detectores. Resultando así, que el detector lineal de mejor rendimiento

para el último escenario es el MMSE y el detector no-lineal de mejor rendimiento es el MMSE-SQRD.

Tabla 3.14 Valores de BER para Eb/No=12dB de los escenarios planteados

	One-tap	ZF	MMSE	QRD	SQRD	MMSE-SQRD
Escenario 1, QPSK	0.00003521	0.00003354	0.00003042	0.00003833	0.00003521	0.00003854
Escenario 1, 16QAM	0.0009431	0.0009626	0.001029	0.0009596	0.001007	0.0009679
Escenario 1, 64 QAM	0.013907	0.01392	0.01408	0.01396	0.01391	0.01386
Escenario 2, QPSK	0.003592	0.003617	0.003658	0.003768	0.003836	0.003741
Escenario 2, 16QAM	0.009159	0.009522	0.009994	0.009237	0.009415	0.009564
Escenario 2, 64 QAM	0.02828	0.02801	0.02857	0.02789	0.02819	0.028
Escenario 3, QPSK	0.0007187	0.0007481	0.0007027	0.0006812	0.0007365	0.0007506
Escenario 3, 16QAM	0.003499	0.003566	0.003712	0.003541	0.003497	0.003539
Escenario 3, 64 QAM	0.01932	0.01976	0.01974	0.0193	0.0194	0.0193025
Escenario 4, QPSK	0.1952	0.08845	0.01131	0.01834	0.01271	0.006723
Escenario 4, 16QAM	0.2812	0.1124	0.03684	0.05126	0.04564	0.02909
Escenario 4, 64 QAM	0.3423	0.1425	0.07375	0.09921	0.09641	0.07668

3.3. ANÁLISIS DEL COSTO COMPUTACIONAL

En la Tabla 3.15 se muestran los tiempos promedio de detección para cada detector cuando se emplea una modulación QPSK. Como se puede ver, para un mismo detector el tiempo de detección comparado con cada escenario es bastante similar. Por ejemplo, para el detector ZF, el tiempo de ejecución para la modulación QPSK entre los escenarios 1 a 4 oscilan entre 0.5588 y 0.6321. Del mismo modo para los demás detectores, se puede ver que los tiempos no varían significativamente entre los diferentes escenarios. Los tiempos más pequeños son los del escenario 1 dado que en este se tienen solamente 3 rayos y en los otros escenarios se tiene un rayo más.

También se puede notar que para un mismo escenario el tiempo de detección más pequeño es del one-tap dado que su complejidad es simple. Sin embargo, el tiempo de detección para dicho escenario en cuanto al detector MMSE-SQRD es mucho mayor. Esto se debe a que la complejidad de este detector es mucho más alta que la de los demás detectores. Existe una relación directamente proporcional entre la complejidad del detector y el tiempo de detección.

Tabla 3.15 Tiempos de detección para QPSK

	Escenario 1	**Escenario 2**	**Escenario 3**	**Escenario 4**
One-tap	0.0101 [s]	0.0099 [s]	0.0101 [s]	0.0097 [s]
ZF	0.5588 [s]	0.6321 [s]	0.6201 [s]	0.6023 [s]
MMSE	0.7102 [s]	0.8070 [s]	0.7142 [s]	0.7917 [s]
QRD	1.9901 [s]	2.1239 [s]	1.9986 [s]	2.0869 [s]
SQRD	6.2717 [s]	5.7383 [s]	5.5362 [s]	5.6030 [s]
MMSE-SQRD	19.4485 [s]	20.1979 [s]	19.0464 [s]	19.1532 [s]

En la Tabla 3.16 se muestran los tiempos de detección para todos los detectores para una modulación 16QAM. Del mismo modo que en la tabla anterior, se puede ver que los tiempos de detección para un mismo detector comparando entre cada uno de los escenarios no presenta mayor variación entre sí. Para la modulación 16QAM, el tiempo de mayor detección también lo tiene el MMSE-SQRD dado que es el detector con mayor complejidad por lo que tarda más en realizar la detección.

Tabla 3.16 Tiempos de detección para 16QAM

	Escenario 1	**Escenario 2**	**Escenario 3**	**Escenario 4**
One-tap	0.0097 [s]	0.0096 [s]	0.0097 [s]	0.0099 [s]
ZF	0.5669 [s]	0.6169 [s]	0.6186 [s]	0.5995 [s]
MMSE	0.7248 [s]	0.7381 [s]	0.7301 [s]	0.7937 [s]
QRD	2.0464 [s]	2.0694 [s]	1.9965 [s]	2.1385 [s]
SQRD	5.7382 [s]	5.4734 [s]	5.5292 [s]	5.7396 [s]
MMSE-SQRD	22.5093 [s]	18.9159 [s]	19.0807 [s]	19.0931 [s]

En la Tabla 3.17 se presentan los tiempos de detección para la modulación 64QAM. De forma similar que, para las dos tablas presentadas anteriormente, se puede notar que la diferencia entre los tiempos para un mismo detector comparados entre cada escenario no

es significativa. Sin embargo, es notorio que el tiempo de detección más pequeño lo tiene one-tap mientras que el MMSE-SQRD es el detector que más tarda en realizar la detección.

Tabla 3.17 Tiempos de detección para 64QAM

	Escenario 1	**Escenario 2**	**Escenario 3**	**Escenario 4**
One-tap	0.0103 [s]	0.0101 [s]	0.0097 [s]	0.0101 [s]
ZF	0.5865 [s]	0.6172 [s]	0.611 [s]	0.5811 [s]
MMSE	0.7941 [s]	0.7347 [s]	0.7687 [s]	0.7183 [s]
QRD	2.1115 [s]	2.0470 [s]	2.0381 [s]	2.0208 [s]
SQRD	5.7385 [s]	5.4923 [s]	5.5456 [s]	5.7842 [s]
MMSE-SQRD	19.6280 [s]	18.9315 [s]	19.0421 [s]	19.2128 [s]

Una vez analizados los resultados de los tiempos de detección para cada uno de los detectores, al comparar los tiempos obtenidos para cada una de las modulaciones, se puede ver que no existe una variación determinada para un mismo detector, considerando un mismo escenario, pero distintas modulaciones. Por ejemplo, para el SQRD en el escenario 1, al emplear una modulación QPSK se tarda 6.2717 segundos. Mientras que con 16QAM se tarda 5.7382 segundos y con 64QAM se tarda 5.7385 segundos. Por lo cual, es evidente que el tiempo de detección es independiente del orden de modulación como se puede notar en la Figura 3.31.

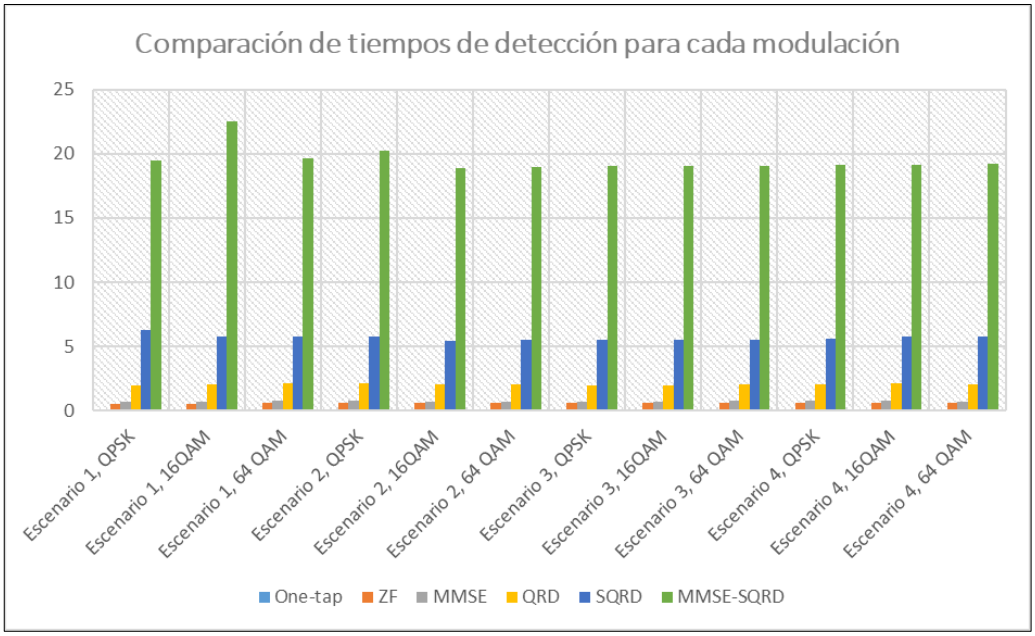

Figura 3.31 Comparación de tiempos de detección para cada modulación

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- Las técnicas de detección permiten atenuar los efectos que se tienen cuando la comunicación se da en un canal variante en el tiempo, siendo los detectores no-lineales los que obtienen un mejor BER debido a que emplean algoritmos complejos que consideran el error por capas, la cuantización y cancelación de interferencia.

- Los detectores lineales tienen una complejidad baja comparados con los detectores no-lineales. Sin embargo, al ser menos complejos, emplean un tiempo de detección bajo comparados con los detectores no-lineales. De modo que si se requiere determinar cuáles son mejores, considerando como más eficientes los detectores que menor tiempo tardan, se elegirían a los detectores lineales.

- La modulación con mejores resultados en cuanto al BER, es la modulación QPSK. Esto se debe a que la modulación QPSK es más robusta en comparación con las modulaciones 16QAM y 64QAM.

- El detector one-tap no es una técnica de detección aplicable para un canal que sea altamente variable en el tiempo ya que presenta un BER elevado. Esto se debe a que en su algoritmo considera una matriz del canal de tipo diagonal. Por lo que solamente divide la señal recibida para la diagonal de la matriz del canal. Pero en los canales variantes en tiempo, la matriz del canal no es una matriz diagonal. Por tanto, al hacer la división, no se está tomando en cuenta correctamente a la matriz del canal.

- En canales que no son muy variantes en el tiempo, la detección-lineal funciona bastante bien y de forma similar a la detección no-lineal. Esto se evidenció en los valores de BER similares entre sí para los 3 primeros escenarios. Las diferencias eran mínimas entre los detectores por lo que para un canal estático en el tiempo se pueden emplear ambos tipos de detección y obtener buenos resultados.

- Al analizar los tiempos de detección de cada detector y compararlo en cada modulación, se puede concluir que el costo computacional de cada detector es independiente al orden de modulación. Esto se evidenció al analizar los tiempos de detección para un mismo detector, considerando un mismo escenario, pero diferente modulación, en donde los tiempos no presentaron ninguna relación con el orden de modulación. Por lo que el tiempo de detección se puede atribuir a la complejidad de cada algoritmo de los detectores, ya sea porque emplean algoritmos

sencillos como dividir la matriz del canal para la señal recibida como en el one-tap o que sean más complejos como el MMSE-SQRD.

- El detector con mejor rendimiento, en términos del BER obtenido, es el MMSE-SQRD. Este detector demostró obtener un BER bajo en comparación con todos los demás detectores, inclusive variando entre las distintas modulaciones. Sin embargo, su desventaja es su complejidad y el alto costo computacional que representa dado el prolongado tiempo que tarda en ejecutarse.

- El detector one-tap es el más eficiente, en términos del costo computacional, dado que su tiempo de detección es el más pequeño. Independientemente del escenario o tipo de modulación, este detector presenta un algoritmo de baja complejidad haciendo que su ejecución sea rápida.

- Los detectores QRD y SQRD tienen un rendimiento bastante bueno, acercándose al rendimiento del MMSE-SQRD. Sin embargo, presentan una ventaja frente a este último, dado que su tiempo de detección es menor. Por lo que representan un costo computacional mejor.

- El detector MMSE-SQRD es el menos eficiente, si se toma en cuenta el costo computacional. Esto se debe a que su algoritmo es más complejo, por lo que se tarda más tiempo en hacer la detección independientemente del orden de modulación o el escenario. Sin embargo, es el de mejor rendimiento cuando se considera el BER obtenido para canales altamente variantes.

4.2. RECOMENDACIONES

- Se recomienda emplear un número más alto de iteraciones, por sobre las 5000 iteraciones. Esto con el propósito de obtener curvas más fiables para las gráficas de Eb/No vs BER.

- Se podría implementar un algoritmo más eficiente para la cuantización en donde no se realicen tantas comparaciones para cuantizar cada uno de los puntos de la constelación. De este modo se optimizaría el tiempo de detección y por ende se reduciría el costo computacional.

- Se sugiere emplear perfiles PDP que representen canales con mayor variación en el tiempo. Considerando que el efecto Doppler sea lo suficientemente notorio entre un escenario y otro, para poder apreciar de mejor manera el rendimiento de los detectores.

- Se podría ampliar el proyecto con la implementación de estas técnicas de detección para un sistema MIMO – OFDM (*Multiple Input Multiple Output* – OFDM). Considerando que, con una cantidad mayor de antenas en transmisión y recepción, sumado a la transmisión paralela de datos, se puede obtener un mejor rendimiento.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] W. L. Abdeldime M.S. Abdelgader, "The Physical Layer of the IEEE 802.11p WAVE," *Proceedings of the World Congress on Engineering and Computer Science,* 22-24 Octubre 2014.

[2] Q. Y. M. W. C.-X. W. Xiang Cheng, "Wideband Channel Modeling and Intercarrier Interference Cancellation for Vehicle-to-Vehicle Communication Systems," *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS/SUPPLEMENT,* 9 Septiembre 2013.

[3] Y. S. Cho, MIMO-OFDM Wireless Communications with MATLAB, Wiley, 2010.

[4] M. P. J. D. L. Archana Doneriya, "Performance Analysis of Linear Precoding Techniques over the Fading Channel for MU-MIMO," *International Conference on Advanced Computation and Telecommunication,* pp. 1-6, 2018.

[5] R. B. V. K. K.-D. K. Dirk Wubben, "MMSE extension of V-BLAST based on Sorted QR Decomposition".

[6] V. K. Garg, Wireless Communications & Networking, San Francisco: Elviser Inc., 2007.

[7] T. S. Rappaport, Wireless Communications: Principles & Practice, New Jersey: Prentice Hall, 2002.

[8] L. Pico, Análisis y Estimación de canal para Comunicaciones Vehiculares con el Estándar 802.11p mediante simulación en Matlab, Quito, 2020.

[9] R. K. Saha, A Handbook on Cellular Mobile Communication Laboratory-A MATLAB based Approach, 2016.

[10] M. Kahn, IEEE 802.11 Regulatory SC DSRC Coexistence Tiger Team V2V Radio Channel Models, 2014.

[11] W. L. Abdeldime Mohammed Salih Abdelgader, "The Physical Layer of the IEEE 802.11p WAVE Communication Standard: The Specifications and Challenges," de *Proceedings of the World Congress on Engineering and Computer Science 2014 Vol II*, San Francisco, 2014.

[12] P. Rodríguez, "SlidePlayer," Noviembre 2017. [En línea]. [Último acceso: 28 Junio 2021].

[13] S. H. S. A. N. F. Shereen A. M. Ahmed, "Overview of Wireless Access in Vehicular," *Indian Journal ofScience and Technology,* vol. 6, p. 8, 2013.

[14] G. L. R. Oscar Orozco Sarasti, "VANET APPLICATIONS FOCUSED ON ENVIRONMENTAL SUSTAINABILITY, A SYSTEMATIC REVIEW," *Cienc. Ing. Neogranad.,* vol. 24, nº 2, p. 22, 2014.

[15] R. Prasad, OFDM for Wireless Communications Systems, Artech House, Inc., 2004.

[16] J.-T. L. Won-Gyu Song, "Channel Estimation and Signal Detection for MIMO-OFDM with Time Varying Channels," *IEEE COMMUNICATIONS LETTERS,* vol. 10, nº 7, 2006.

[17] L. J. C. Ye (Geoffrey) Li, "Bounds on the Interchannel Interference of OFDM in Time-Varying Impairments," *IEEE TRANSACTIONS ON COMMUNICATIONS,* vol. 49, nº 3, 2001.

[18] J. W. C. X. Jun Tao, "Estimation of Channel Transfer Function and Carrier Frequency Offset for OFDM Systems With Phase Noise," *IEEE Transactions on Vehicular Technology,* vol. 58, nº 8, 2009.

[19] G. M. Franz Hlawatsch, Wireless Communications Over Rapidly Time-Varying Channels, Oxford: Elsevier, 2011.

[20] A. V. Ordóñez, Simulación de las Técnica de Detección ZF, MMSE y MLD para un Sistema MIMO - OFDM, Quito, 2019.

[21] D. J. R. Chisaguano, Computational Cost Reduction of the Detection Process of MIMO-OFDM with ESPAR-antenna-assisted Receiver, Nara, 2013.

[22] MathWorks, "Communications Toolbox," [En línea]. Available: https://es.mathworks.com/products/communications.html. [Último acceso: 19 Mayo 2021].

[23] S. P. H. H. E. G. S. Jens Mittag, Enabling Accurate Cross-Layer PHY/MAC/NET Simulation Studies of Vehicular Communication Networks.

[24] X. Flores, Análisis cualitativo y cuantitativo de modelos para el cálculo del PER (Packet Error Rate) para comunicaciones IEEE802.11p, Quito, 2019.

[25] LAN/MAN Committee, IEEE Standard for Information technology Telecommunications and information echange between systems: IEEE Computer Society, New York: IEEE Computer Society, 2007.

[26] MathWorks, comm.OFDMModulator.

[27] G. E. B. a. T. C. P. Dent, "JAKES FADING MODEL REVISITED,» *ELECTRONICS LETTERS,* vol. 29, nº 13, pp. 1162-1163, 1993.

[28] E. T. Michel Mfeze, "Comparative Approach of Doppler Spectra for Fading Channel Modelling by the Filtered White Gaussian Noise Method," *International Journal of Computer Science and Telecommunications,* vol. 6, nº 11, 2015.

[29] MathWorks, "Centro de ayuda," [En línea]. Available: https://la.mathworks.com/help/matlab/ref/diag.html. [Último acceso: 16 Junio 2021].

[30] «Enciclopedia de Todas las Palabras de la Matematicas,» [En línea]. Available: http://www.allmathwords.org/es/d/determinant.html. [Último acceso: 16 Junio 2021].

[31] L. Vázquez, "FLUPROJECT," 20 Noviembre 2019. [En línea]. Available: https://www.flu-project.com/2019/11/teleco-in-nutshell-v87-modulacion-amplitud-cuadratura.html. [Último acceso: 28 Junio 2021].

[32] L. Carvajal, Metodología de la Investgación Científica. Curso general y aplicado, 28 ed., Santiago de Cali: U.S.C., 2006, p. 139.

[33] J. L. Christian Sgraja, "Estimation of Rapid Time-Variant Channels for OFDM using Wiener Filtering," *IEEE International Conference on Communications,* vol. 4, pp. 2390-2395, 2003.

[34] E. Guzmán, Cálculo vectorial en R2, Lima, 2019.

ANEXOS

ANEXO A. Código del programa principal y funciones implementadas en MATLAB (Anexo digital).

ANEXO B. Tabla de comandos empleados en el programa y funciones de MATLAB.

ANEXO A

Anexo Digital.

ANEXO B

Tabla Anexo B. Comandos empleados en el programa principal y funciones.

Programa o función	Comando en MATLAB
Programa principal OFDM_fading.m	clc
	close
	clear
	tic
	log
	repmat
	comm.OFDMModulator
	comm.OFDMDemodulator
	zeros
	length
	for
	randi
	qammod
	reshape
	step
	reset
	awgn
	toc
	qamdemod
	biterr
	mean
	semilogy
	ylim
	xlim
	grid
	xlabel
	ylabe
	num2str
	sprintf
	save
Función Channel.m	switch
	case
	max
	abs
	size
	cell
	doppler
	length
	comm.RicianChannel
	round

Función Detector.m	zeros
	if
	elseif
	for
	fft
	diag
	length
	eye
	inv
	ones
	qr
	reshape
Función one_tap.m	zeros
	for
	fft
	reshape
Función SQRD.m	size
	zeros
	sum
	abs
	for
	min
	if
	sqrt
	find
	length
Función MMSE-SQRD.m	size
	zeros
	for
	abs
	sum
	min
	if
	sqrt
	find
	length
Función cancelador.m	zeros
	qamdemod
	qammod
	for
	if
	elseif
	sum
	else
Función cuantificador.m	if
	real

	imag
	elseif
	elseif
	sqrt

ORDEN DE EMPASTADO