ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"ESTUDIO, MODELAMIENTO Y SIMULACION DE SISTEMAS MIMO"

TESINA DE SEMINARIO

Previa a la obtención del Título de:

INGENIERO EN ELECTRONICA Y TELECOMUNICACIONES

Presentado por:

VIVIANE EMILY MOLINEROS GUEVARA

GUAYAQUIL – ECUADOR

2010

AGRADECIMIENTO

Al Ingeniero Juan Carlos Avilés y al Ingeniero Hernán Córdova, profesores que nos ayudaron con la resolución de nuestros proyectos y por la disposición que tuvieron para resolver toda inquietud que se presentó en el desarrollo del proyecto.

DEDICATORIA

A Dios porque sin él no estaría sustentando ni entregando esta tesis, a mi familia porque durante toda mi vida siempre confiaron y creyeron en mí, y a mis amigos.

Gracias Totales

TRIBUNAL DE SUSTENTACION

_____	_____
Ing. Sergio Flores M.	Ing. Hernán Córdova
DECANO DE LA FIEC	PROFESOR DE SEMINARIO

_____

Ing. Juan Carlos Avilés

DELEGADO DE LA UNIDAD

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesina de Seminario, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”.

_____

Viviane Molineros G.

RESUMEN

El presente trabajo se enfoca en el desarrollo de dos soluciones que permitan resolver las interferencias co-canales presentes en los sistemas MIMO (Multiple Input Multiple Output), mediante el uso de ecualizadores.

Como se observara en las siguientes secciones, la solución planteada es el Beamforming, el cual consta de dos tipos de ecualizadores para poder hallar la matriz de pesos, la cual será analizada más adelante y veremos cuál de los dos tipos de ecualizadores cumple con el objetivo de eliminar eficazmente las interferencias co-canales.

Las dos soluciones analizadas serán aplicadas en el método Beamforming en un sistema Wireless, en el cual se consideró el efecto del ruido blanco aditivo gausiano y el canal a utilizar es el flat fading Rayleigh.

ÍNDICE GENERAL

Pág.

INDICE DE FIGURAS

ABREVIATURAS

AWGN	Additive White Gaussian Noise
BF	Beamforming
BFSK	Binay Frecuency Shift Keying
BPSK	Binary Phase Shift Keying
CSI	Channel State Information
MIMO	Multiple Input Multiple Output
MMSE	Minimum Mean Square Error
ISI	Inter Symbol Interference
SNR	Signal-to-Noise Ratio
OFDM	Orthogonal Frequency Division Multiplexing
PSD	Power Spectral Density
QAM	Modulation de amplitude en cuadratura
QPSK	Quadrature Phase Shift Keying
ZF	Zero Forcing
ZFBF	Zero Forcing Beamforming

INTRODUCCIÓN

Los sistemas de múltiples entradas y múltiples salidas (MIMO) aprovechan de la propagación multicamino para disminuir la tasa de error e incrementar la tasa de transmisión. El estándar IEEE 802.11n utilizara este tipo de tecnología para obtener velocidades de hasta 600Mbps y también ya está probado que la tecnología celular 4G podrá lograr tasas de transferencia de hasta 100Mbps a una distancia de 200m.

La ventaja que tienen los sistemas MIMO, es que se considera al canal como una matriz, aprovechando sus múltiples antenas para la decodificación correcta de las señales enviadas.

MIMO tiene tres principales categorías entre ellas están: Beamforming, Spatial Multiplexing y Diversidad de código.

En este presente trabajo hablaremos de la tecnología Beamforming, la cual permite una máxima radiación de la señal deseada hacia el usuario y en la dirección de los usuarios interferentes la señal se reduce. Esta técnica utiliza dos tipos de ecualizador para poder hallar la matriz de pesos, entre estos consta: Zero Forcing y el Minimum Mean Square Error.

Estos ecualizadores serán analizados detenidamente y simulados en las secciones siguientes, así podremos observar cual de los se los puede implementar con mayor facilidad.

CAPÍTULO 1

COMPRENSIÓN DEL PROBLEMA

1.1. Definición de interferencias Co- canales

La interferencia co-canal ocurre cuando la misma frecuencia de portadora de dos transmisores separados físicamente (por ejemplo dos estaciones de base), llegan a un mismo receptor al mismo tiempo. Este problema no solo ocasiona una limitación en la capacidad del sistema (en número de usuarios por unidad de superficie), sino también ocasiona una degradación en el BER hasta llegar a niveles desfavorables. La siguiente figura1 muestra un claro ejemplo de las

interferencias co canales y observamos los niveles que queremos (67 dBm), los que no queremos (85 dBm) y de los cuales no debemos de preocupamos (100 dBm).

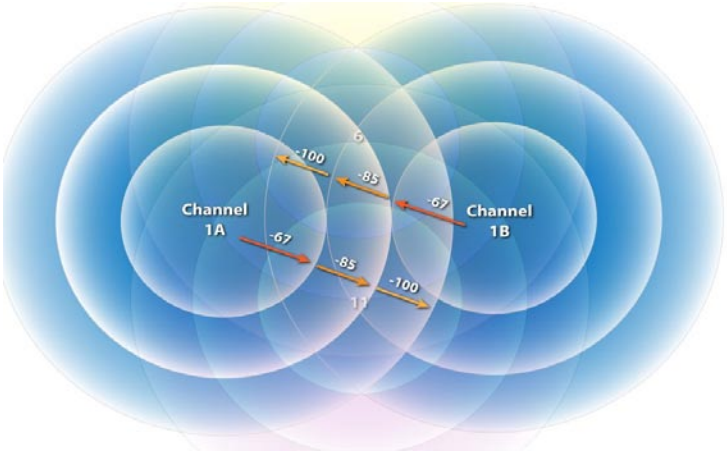

Figura 1.1 interferencia co-canal

Actualmente en muchos sistemas de multiacceso fijo y algunos de telefonía móvil, las estaciones de base se diseñan para una cobertura RF mediante despliegues tipo celular con antenas sectorizadas que usan diferentes frecuencias que efectivamente multiplican el número de canales disponibles. Al aumentar la necesidad de mayor capacidad, se debe reducir el área de cobertura generándose una mayor repetición de las portadoras de cada celda con lo que aumenta la probabilidad de interferencias tipo cocanal.

1.2. Problema de los sistemas MIMO (multiple input multiple output)

El propósito de los métodos de reducción de este tipo de interferencias cocanales apunta a mejorar el BER, el C/I (relación portadora a interferencia) y la capacidad de la red.

MIMO es una técnica que utiliza la diversidad de las señales multitrayectoria (multipath), tradicionalmente calificados como un problema en las comunicaciones inalámbricas [2], para incrementar la habilidad de un receptor de recobrar los mensajes de la señal modulada, aumentar la tasa de transmisión y reducir la tasa de error.

En breves palabras, MIMO aumenta la eficiencia espectral de un sistema de comunicación inalámbrica por medio de la utilización del dominio espacial.

Las interferencias co canales si afectan la capacidad de un sistema MIMO, debido a su naturaleza de múltiples transmisores y receptores como lo expresan en [3], provocando una disminución significativa en esta.

CAPÍTULO 2

DEFINICION DEL PROBLEMA

2.1. Cuál es el problema

Los sistemas MIMO, al ser múltiples transmisores y múltiples receptores, estos llegan a ser afectados por las interferencias co canales, afectando considerablemente la capacidad del sistema.

Por lo tanto para poder disminuir este tipo de interferencias tenemos que hallar el esquema que elimine de una manera más eficaz las interferencias co canales mediante las asunciones que ser presentaran a continuación, las cuales van a ser presentadas en las siguientes

secciones.

2.2. El Escenario

El sistema a utilizar será un sistema MIMO 2x2 donde la interferencia co-canal proviene de otros usuarios cercanos, en el cual se consideran las siguientes asunciones:

- El Transmisor conoce la información de estado del canal (CSI).
- El modelo de canal es el flat fading Rayleigh: donde los componentes de $h_{i,j}$ de la matriz H son independientes e idénticamente distribuidos [2].
- Como el canal es flat fading, el multipath se reduce a un solo tap. Al ser el canal tipo plano (flat) estamos diciendo que todas las bandas de frecuencia sufren de la misma magnitud de desvanecimiento (fading), el cual representa cambios rápidos de la señal sobre pequeñas distancias o intervalos temporales.
- No se toman en cuenta los desplazamientos Doppler debido a que se asume que el sistema inalámbrico es fijo por lo que no existe movimiento relativo entre el transmisor y el receptor.

A continuación describiremos como está compuesta una señal MIMO 2x2 ya que ese es el sistema que vamos a utilizar en nuestra

transmisión que luego va a verse modificada en las siguientes secciones.

La señal que recibe la antena 1 viene dada por la siguiente ecuación:

$$y_1 = h_{1,1}x_1 + h_{1,2}x_2 + n_1$$

$$y1 = [h_{1,1} \quad h_{1,2}] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + n_1$$

La señal que recibe la antena 2 viene dada por la siguiente ecuación:

$$y_2 = h_{2,1}x_1 + h_{2,2}x_2 + n_2$$

$$y2 = [h_{2,1} \quad h_{2,2}] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + n_2$$

La señal enviada vendría dada por:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} \\ h_{2,1} & h_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}$$

Se puede representar matricialmente la señal que llega al receptor como:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$$

Donde la matriz H representa los distintos caminos de propagación que son creados por la cantidad de antenas existentes tanto en el transmisor como en el receptor; n es ruido blanco Gaussiano con media cero y covarianza $\sigma^2 I$ y x es el vector de transmisión.

CAPITULO 3

COMPRENSION DE SOLUCIONES

Para poder resolver las interferencias co-canales existentes en los sistemas MIMO, se han encontrado distintos métodos que se pueden emplear; entre los cuales solo se analizara el método beamforming, que puede utilizar los siguientes ecualizadores para poder hallar la matriz de pesos: el MMSE (mínimum mean square error) y el Zero Forcing.

3.1. Beamforming

Beamforming es una técnica que en el transmisor, la salida de cada dipolo del arreglo se pondera por un factor de pesos, cuyo valor es asignado mediante algún algoritmo escogido, esta técnica permite una máxima radiación de la señal deseada hacia el usuario (lóbulo principal) y en las direcciones de los suscriptores interferentes el nivel de potencia transmitida se reduce, sin la necesidad de usar una señal de referencia temporal.

La figura 3 presentada a continuación representa una estructura sencilla del beamforming.

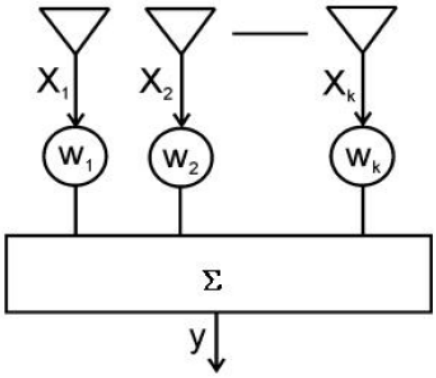

Figura 3.1 simple estructura del beamforming

La señal que será transmitida viene dada por la siguiente expresión [1]:

$$r = \sum_{n=0}^{N-1} w_n^* x_n = \mathbf{w}^H \mathbf{x} \quad (\mathbf{1})$$

Donde $w^H = [w_0^*, w_1^*, \ldots\ldots\ldots w_{N-2}^*, w_{N-1}^*] = [w^T]^*$ es la transpuesta conjugada compleja (Hermitian) del vector de ponderaciones w, y x representa el vector de la señal deseada (de longitud N).

La señal recibida es una suma de las señales de varios usuarios [1], donde se designa a la señal "deseada" como:

$$\mathbf{x} = \alpha \mathbf{h_0} + \mathbf{n}, \quad (2)$$

$$\mathbf{n} = \sum_{m=1}^{M} \alpha_m \mathbf{h_m} + \mathbf{noise} \quad (3)$$

Reemplazamos la ecuación (3) en (2) y obtenemos la siguiente ecuación:

$$x = \alpha h_0 + \left[\sum_{m=1}^{M} \alpha_m h_m + noise\right] \quad (4)$$

En donde α es un término relacionado con cancelación de la interferencia y el ruido.

Nuestro objetivo es estimar la amplitud de la señal y de las observaciones de x usando un conjunto de pesos de Beamforming w(t), donde la salida del Beamforming de banda estrecha está dada por la siguiente ecuación:

$$y = \sum_{n=0}^{N-1} w_n^* x_n = w^H x \quad (5)$$

Reemplazamos la ecuación (2) en (5) y obtenemos la siguiente fórmula:

$$y = w^H(\alpha h_0 + n) \quad (6)$$

$$y = w^H \left[\alpha h_0 + \left(\sum_{m=1}^{M} \alpha_m h_m + noise\right)\right] \quad (7)$$

$$y = \alpha w^H h_0 + w^H \sum_{m=1}^{M} \alpha_m h_m + w^H noise \quad (8)$$

$$y = \alpha\, w_0 h_0 + \sum_{m=1}^{M} \alpha_m w_m h_m + \text{noise} \quad (9)$$

Finalmente la señal transmitida viene dada por la ecuación descrita anteriormente (5) en donde la matriz de los pesos w^H se la multiplica por la señal x, en donde se reemplazo la ecuación (4) en (6), que al descomponer dicha ecuación, finalmente se obtiene la ecuación (9); donde se observa que la señal ***y*** está compuesta por $\alpha w_0 h_0$ siendo estos los valores de la señal deseada, mientras que $\sum_{m=1}^{M} \alpha_m w_m h_m$ son las señales interferentes no deseadas. Los pesos pueden variar dependiendo de los datos recibidos.

3.1.1 Minimum Mean Square Error (MMSE)

Para calcular los pesos (w^H) del beamforming se pueden utilizar algunos algoritmos pero aquí daremos una pequeña descripción del MMSE (Minimum mean square error). El criterio principal es el de minimizar el error con respecto a una señal de referencia d(t), la cual debe de ser conocida. El MMSE considera la reducción de la potencia media de la señal de error, que es la diferencia entre una señal de referencia conocida en la estación base receptora, y la señal de salida [1].

La señal de error viene dada por la siguiente ecuación, en donde y(t) es la señal de salida del transmisor y d(t) es la señal de referencia en la estación base receptora:

$$e(t) = y(t) - d(t) \quad (10)$$

$$e(t) = w^H x(t) - d(t) \quad (11)$$

Para poder entender de una mejor manera de donde obtenemos la señal de error, se la mostrara en el grafico presentado a continuación en la figura 3.2

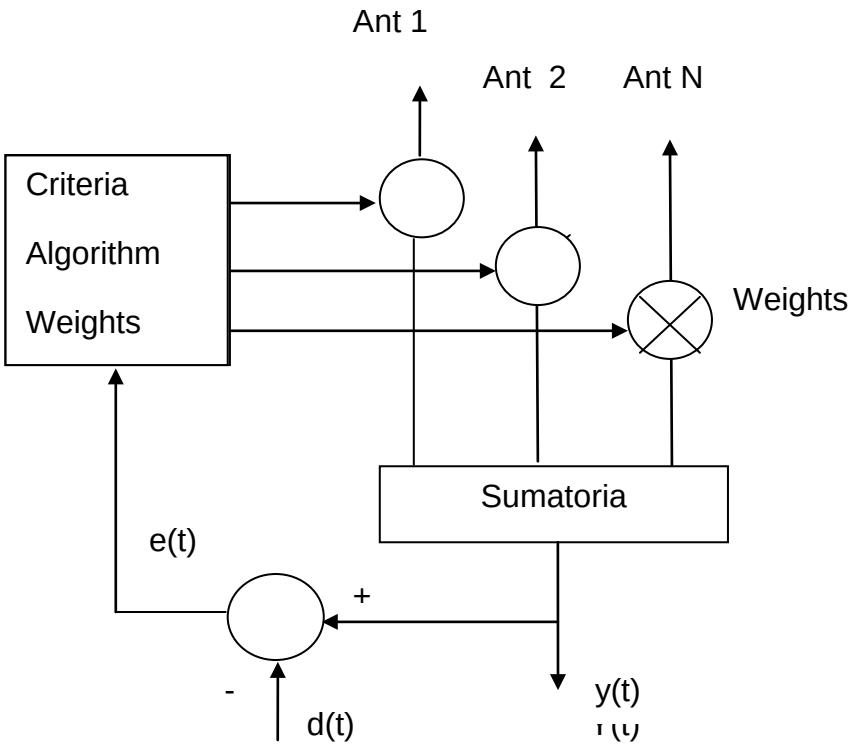

Figura 3.2

Luego para poder encontrar los pesos de la matriz w debemos de hallar el mínimo de la señal de error e(t):

$$\mathrm{w_{MMSE}} = \arg\min_{\mathrm{w}} \mathrm{E}\{|\mathrm{e(t)}|^2\} \quad (12)$$

$$\mathrm{E}\{|\mathrm{e(t)}|^2\} = \mathrm{E}\{|\mathrm{w^H\, x(t) - d(t)}|^2\} \quad (13)$$

$$E\{|e(t)|^2\} = E\{|w^H x(t) - d(t)||w^H x(t) - d(t)|\} \quad (13.\mathrm{a})$$

Aplicamos la ley distributiva a la ecuación (13.a), para obtener lo siguiente:

$$E\{|e(t)|^2\} = E\{|w^H x(t) w^H x(t) - w^H x(t)\, d(t) - w^H x(t) d(t) + d(t) d(t)|\} \quad (13.\text{b})$$

Una de las propiedades de las matrices transpuestas es: siendo x una matriz mxn entonces si queremos hallar la matriz x^2, utilizamos la siguiente propiedad $\mathrm{x^H x} = \mathrm{x}^2$. Lo mismo ocurre para hallar la matriz $|\mathrm{w^H}|^2$ y para la señal de referencia $|\mathrm{d(t)}|^2$.

Por lo tanto obtenemos la ecuación (14), la cual se la llama squared error, y se utiliza la propiedad explicada anteriormente.

$$= \mathrm{E\{w^H x x^H w - w^H x d^* - x^H w d + d d^*\}} \quad (14)$$

$$\mathrm{E\{|e(t)|^2\} = E\{w^H x x^H w\} - E\{w^H x d^*\} - E\{x^H w d\} + E\{d d^*\}} \quad (14.\text{a})$$

Como $\mathrm{w^H}$ y w son constantes entonces podemos realizar lo siguiente:

$$\mathrm{E\{|e(t)|^2\} = w^H E\{x x^H\} w - w^H E\{x d^*\} - E\{x^H d\} w + E\{d d^*\}} \quad (14.\text{b})$$

Reemplazamos $\mathrm{r_{xd} = E\{x d^*\}}$ y $\mathrm{R = E\{x x^H\}}$ y obtenemos la siguiente ecuación, la cual es conocida como valor del error esperado (expected error value):

$$= w^H R w - w^H r_{xd} - r_{xd}^H w + d d^* \quad (15)$$

Como queremos hallar el valor mínimo, debemos de derivar la ecuación (15) con respecto a $\mathrm{w^H}$ e igualarla a cero:

$$\frac{\partial E\{|e(t)|^2\}}{\partial w^H} = Rw - r_{xd} = 0 \quad (16)$$

$$\Longrightarrow w_{MMSE} = R^{-1} r_{xd} \quad (17)$$

Finalmente hallamos la ecuación para obtener el vector optimo de los pesos w; pero debemos de tener en cuenta que al emplear este método no siempre es posible hallar la matriz de covarianza de la señal x, al igual el desconocimiento que sabe haber de la señal de referencia d(t) [4]. Por lo tanto a continuación hablaremos de otro método que puede ser mejor y más fácil de emplear que el mencionado.

3.1.2. Zero Forcing beamforming (ZFBF)

Zero forcing beamforming es una técnica muy eficaz para suprimir las interferencias co- canales, ocasionadas por la gran cantidad de usuarios, y el ISI (interferencia entre símbolo) siempre y cuando el canal de bajada (downlink) sea perfectamente conocido y el ángulo de separación sea lo suficientemente grande con el fin de cancelar todas las interferencias existentes y disminuir la cantidad de potencia requerida en la antena [8]; de lo contrario el rendimiento el sistema se puede degradar considerablemente.

Mediante el uso de esta técnica se puede elegir los vectores de pesos

para evitar las interferencias entre los usuarios [5], estos pesos pueden ser encontrados al invertir la matriz del canal compuesto de los usuarios.

El rendimiento del Zero forcing beamforming se degrada cuando la información del canal en el transmisor (CSI) es inexacta [6].

Para eliminar la interferencia co canal, se debe cumplir la siguiente condición: $h_k w_j = 0$ para $j \neq k$ [6]. Donde $H = [h_1^T .. h_K^T]^T$ y $W = [w_1 \ ... \ w_k]$ respectivamente. El óptimo W está diseñado para obtener la máxima relación de señal a ruido (SNR). Por lo tanto para poder encontrar la matriz W que satisfaga la siguiente ecuación: WH = I.

Mediante el uso de esta técnica (ZFBF) obtenemos que la matriz W, la cual satisface la condición de cero interferencias; puede ser implementada como en [7] [6].

$$W(S) = H(S)^{\dagger} \quad (18)$$

$$W(S) = H(S)^{*}(H(S)H(S)^{*})^{-1} \quad (19)$$

Los autores de [6] y [7] proponen que para obtener los vectores de beamforming cumplan con la condición de cero interferencia se debe de cumplir que $h_k w_j = 0$ para $j \neq k$. $S \subset \{1, \ldots, K\}, |S| \leq M$, siendo H(S) y W(S), las submatrices correspondientes a $H = [h_1^T .. h_K^T]^T$ y $W = [w_1 \ ... \ w_k]$. También obtienen la ganancia efectiva del canal la cual

esta expresada como: $\gamma_i = \frac{1}{||w_i||^2} = \frac{1}{[(H(S)H(S)^*)^{-1}]_{i,i}}$ que representa $[A]_{i,i}$ la diagonal del elemento ith de A. En los dos papers presentados al inicio del párrafo utilizan el ZFBF para hallar la matriz W, por lo que no es un método complejo como otros métodos que ellos expresan. Mediante el uso del Zero Forcing Beamforming, los autores tratan de demostrar mediante el uso de otras formulas que el ZFBF puede reemplazar al otro método que ellos proponen cuando existen K usuarios, donde $K \to \infty$; por lo que el ZFBF es más fácil de implementar.

CAPITULO 4

MODELAMIENTO Y SIMULACION

4.1. Modelamiento

Mediante el uso del simulador Matlab, presentaremos la simulación de los métodos vistos en la sección anterior, el cual es zero forcing beamforming y MMSE beamforming, donde se utilizara el sistema MIMO 2x2 con canal flat Rayleigh fading cuatro tipos de modulación BPSK, BFSK, QPSK y 16QAM; ruido AWGN y por ultimo utilizaremos el ecualizador zero forcing y el MMSE. Para el efecto se usaran las siguientes ecuaciones:

Ecualizador Zero Forcing

$$W_n(S_n) = H_n(S_n)^{\dagger}$$

$$W_n(S_n) = H_n(S_n)^{*}(H_n(S_n)H_n(S_n)^{*})^{-1}$$

Mientras que para el ecualizador MMSE utilizamos la siguiente ecuación:

$$W_n(S_n) = [H_n(S_n)^{*} + N_0 I]^{-1} H_n(S_n)^{*}$$

Como lo visto en la sección anterior donde sabemos que para obtener los pesos de la matriz W; a la matriz H se la invierte y eso es lo que el autor de la simulación realiza.

Tomando como constantes las siguientes variables:

- N = 10^6 siendo los números de símbolos q van a hacer enviados.
- nTx y nRx son las variables del transmisor y receptor respectivamente y son igual a dos cada una debido a que el sistema es un MIMO 2x2.

4.2. Simulación

En la simulación realizada, se aprecia como la implementación descrita hace uso de cálculos matriciales por medio de funciones especializadas como por ejemplo: kron, reshape, squeeze, etc. Esto hace innecesario el uso de lazos tradicionales de programación como el for y el while, lo cual reduce considerablemente el tiempo de procesamiento y que el código no sea extenso.

En el transmisor, es donde analizaremos como es que se calculan los pesos que luego van a hacer transmitidos. Al principio del código se generan números aleatorios entre 0 y 1; que luego pasan por un modulador BPSK para la primera simulación mientras que las otras son 16 QAM y QPSK. También utilizamos otro tipo de modulación, este es la modulación BFSK la cual consiste en: representar los bits 0's y 1's como señales s_1 y s_2 teniendo frecuencias f_1 y f_2 respectivamente, siendo estas frecuencias ortogonales entre sí. La ecuación general para las señales s_1 y s_2 viene dada por:

$$s_i(t) = \sqrt{\frac{2E}{T}} \cos(2\pi f_i t + \varphi)$$

Podemos observar como en la línea doce utilizan la función kron para multiplicar la matriz s con una matriz de unos, siendo esta almacenada en la variable smod. En la línea de abajo se utiliza el comando reshape

el cual consiste en agrupar en vectores de 2x1, para luego ser transmitidos por las antenas existentes.

Por último para tener la señal que va a ser transmitida por las dos antenas transmisoras se utiliza la función squeeze donde junta todas las matrices de 2x1 y le agrega el ruido AWGN.

En el receptor para formar el ecualizador ZF, se invierte la matriz H, utilizamos la función hcof para poder obtener los valores de (d, a, b, c), los cuales sirven para poder obtener la inversa de la matriz h debido a que no podemos utilizar la función inv() porque la matriz h está compuesta de valores reales e imaginarios.

Lo mismo se emplea para el ecualizador MMSE pero utilizamos una ecuación distinta a la de ZF, la cual esta puesta al principio de esta sección. Por lo que se puede observar que al comparar con la ecuación del ZF. Aparte del término $N_0 I$, las dos ecuaciones son comparables; por lo tanto cuando el término del ruido es igual a cero el ecualizador MMSE puede ser reducido a ZF.

Luego aplicamos el mismo método que se usa en el transmisor pero invertido eso quiere decir que primero aplicamos la función kron y de ahí la función reshape. Por último demodulamos la modulación BPSK, lo mismo sucede con las modulaciones QPSK y 16QAM para la primera simulación, mientras que para la demodulación de BFSK utilizamos un

demodulador coherente, y contabilizamos los errores obtenidos, para realizar los gráficos que se presentaran en la siguiente sección.

El código original fue desarrollado por Krishna Pillai, pero se tuvo que realizar muchas modificaciones para poder adaptar las modulaciones 16QAM, QPSK y BFSK. Finalmente el resultado de la simulación nos permite obtener una gráfica del BER vs SNR. Para más detalles sobre la simulación línea por línea, se encuentra explicada en la sección Anexos.

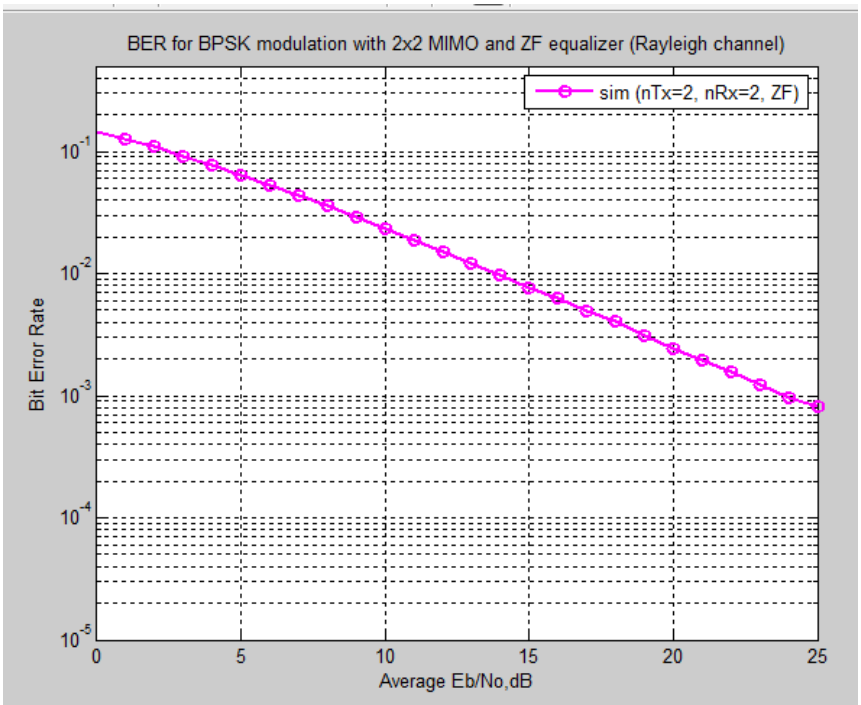

Figura 4

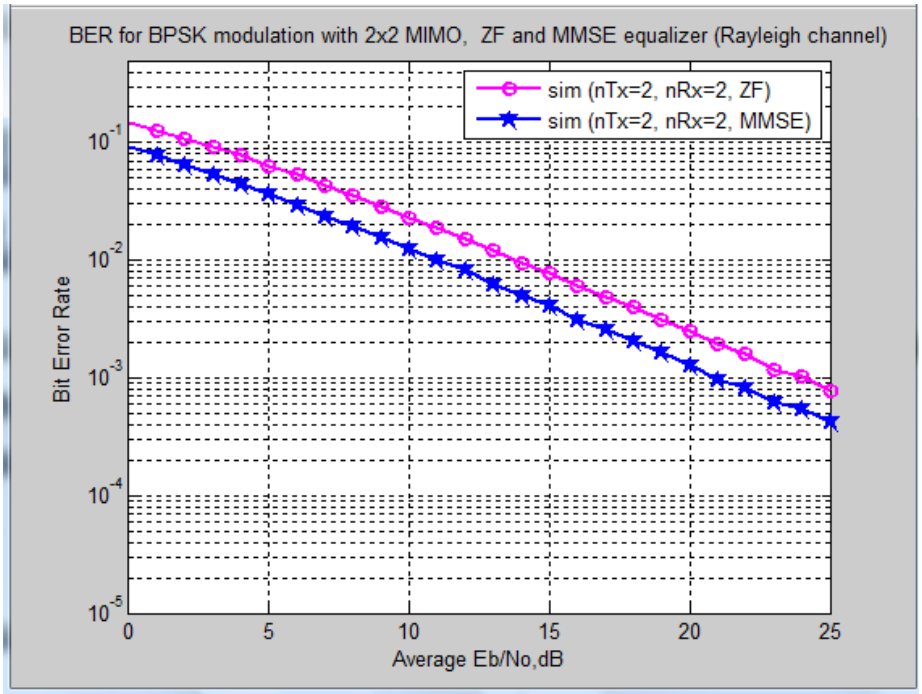

Figura 5

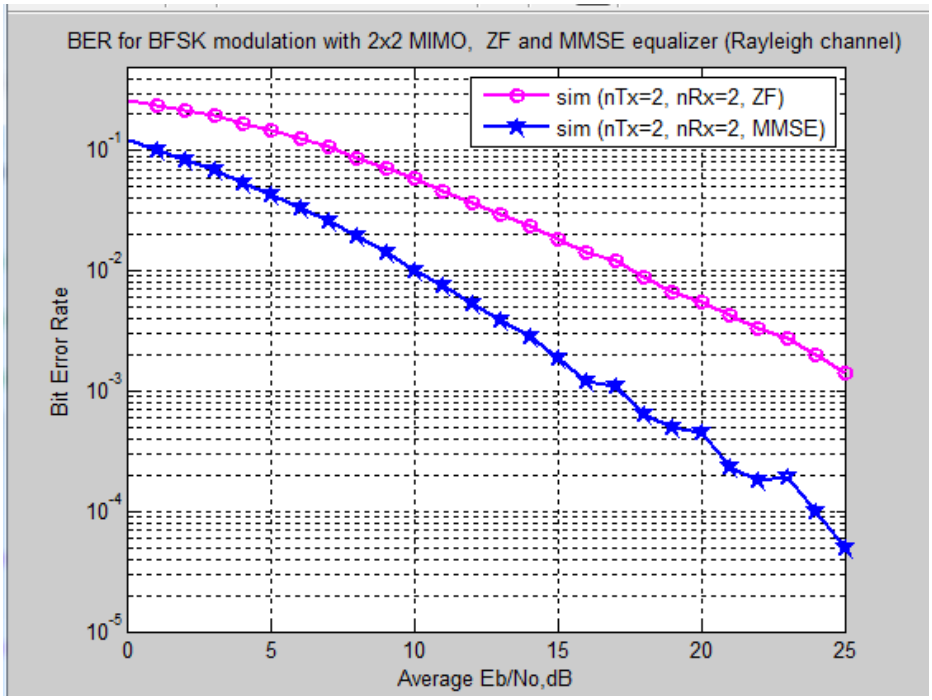

Figura 6

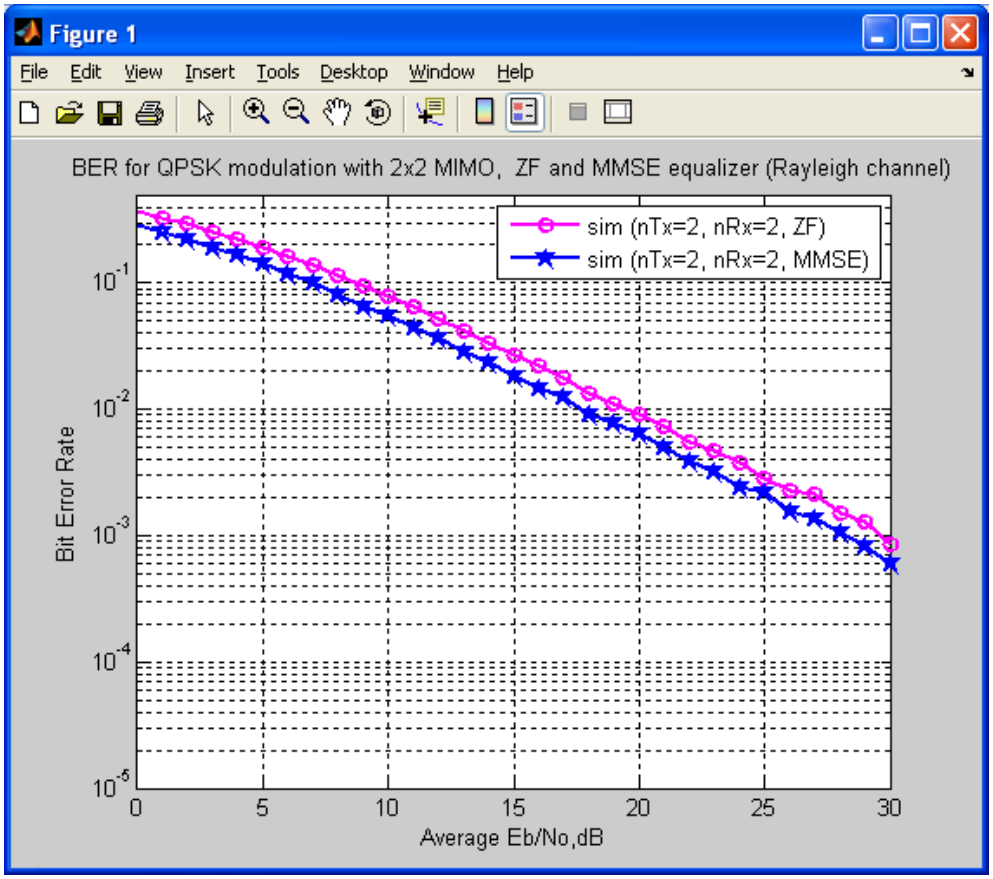
Figure 1
File Edit View Insert Tools Desktop Window Help
BER for QPSK modulation with 2x2 MIMO, ZF and MMSE equalizer (Rayleigh channel)
sim (nTx=2, nRx=2, ZF)
sim (nTx=2, nRx=2, MMSE)
Bit Error Rate
10^-1
10^-2
10^-3
10^-4
10^-5
0
5
10
15
20
25
30
Average Eb/No,dB

Figura 7

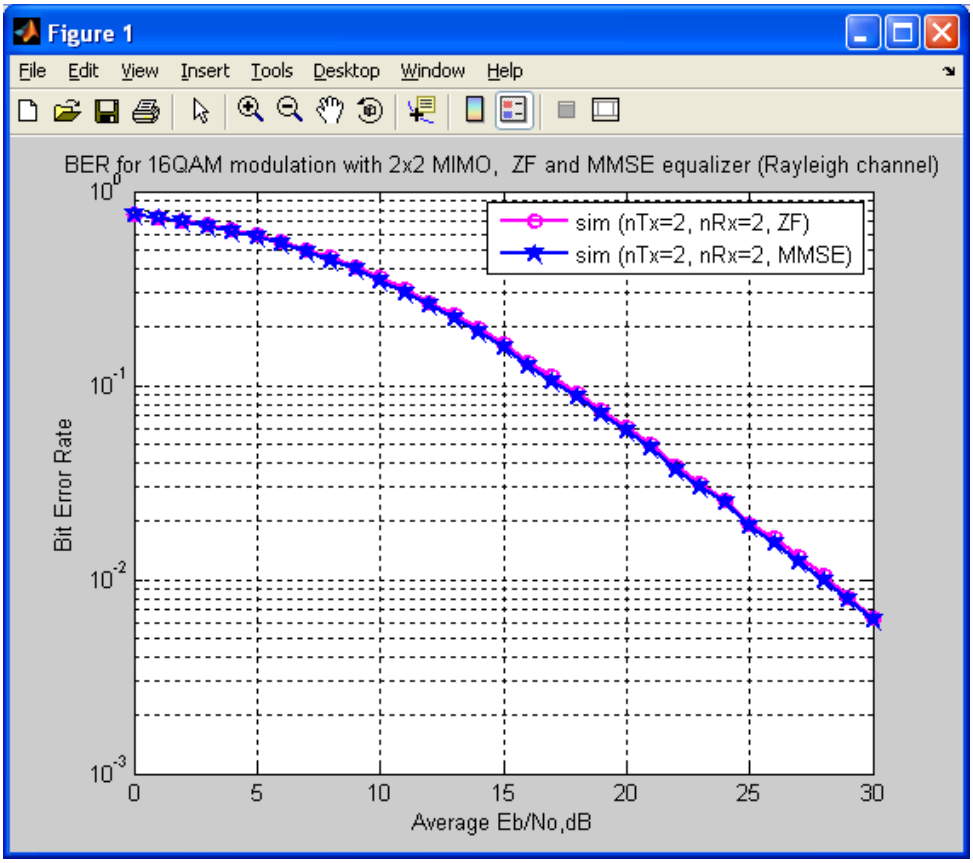

Figura 8

Como se puede observar en la figura 4, el uso del ecualizador zero forcing mejora el BER a medida que aumentan los dB del SNR, es decir que al elevar la potencia vamos a tener una mejora y menos error en la transmisión. La figura 4 representa la simulación original sin hacer algún cambio. En las siguientes graficas se verá los distintos cambios que se realizaron a la simulación original para así ver los distintos resultados al variar la modulación.

En la Figura 5, se implemento la segunda solución la cual es: Minimum Mean Square Error, se empleo la misma cantidad de antenas y las

mismas condiciones del canal al igual que el ZF. Se observa en la grafica que el método MMSE tiene una pequeña mejora al comprarlo con el ZF en donde esta mejora es de aproximadamente 3 – 4 dB en el punto 10^{-3} en la escala del BER.

La ventaja que presenta el ZF al MMSE, se debe a que el ZF es un algoritmo, que no requiere ya sea de la señal de referencia o de la matriz de covarianza a diferencia del método del MMSE que es más complejo de implementar, por lo tanto es más fácil de implementar el ZF, ya que solo se necesita obtener la inversa del canal y conocer completamente la información del canal (CSI).

El grafico 6 representa el cambio de modulación; ahora se uso la modulación BFSK, dicha simulación se presenta en el Anexo A, en donde podemos observar que la grafica de color azul del ecualizador MMSE mejora considerablemente en comparación con la figura 5, se usa menos potencia para obtener un mejor BER. En esta grafica el MMSE tiene una diferencia de 10dB en comparación con la grafica del ZF, que al cambiar de modulación quedo igual a la figura 5, no hubo ninguna variación.

Al cambiar la modulación de BPSK a BFSK vimos una considerable mejora en la grafica del MMSE esto se debe que al utilizar BFSK con demodulación coherente, el MMSE mejora su probabilidad de error

debido a que usamos una modulación menor a la BPSK. Pero sabemos que la tasa de transmisión deberá de ser menor, aunque si queremos transmitir a mayores velocidades debemos de utilizar modulaciones más complejas como QAM o QPSK.

En la figura 7 se observa el cambio de modulación que se hizo a la simulación presentada en la grafica 5, esta vez se uso la modulación QPSK, en esta se observa que el ecualizador MMSE sigue siendo mejor que el ZF. Aunque la figura 8, al usar la modulación 16 QAM, parece que la curva del MMSE se encuentra encima de la curva del ZF. Pero si observamos detenidamente la grafica, nos percatamos que no es así, que si hay una muy pequeña separación entre ellas.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

En este trabajo se ha desarrollado, analíticamente y mediante la ayuda del simulador de Matlab, las dos soluciones propuestas en la sección 3.

1. Se comprobó que el zero forcing beamforming tiene un buen rendimiento en el sistema debido a la eliminación de las interferencias ya que a medida que el ruido disminuye, el BER va mejorando.

2. Mediante la primera simulación en donde utilizamos modulación BPSK, el método MMSE, es mejor que el ZFBF en aproximadamente 3 – 4 dB; mientras que para la modulación BFSK la diferencia es de alrededor 10dB, para QPSK esta es de aproximadamente 1dB y para 16 QAM es mucho menor al 1dB.

3. Aunque se mantuvo las mismas condiciones al momento de cambiar la modulación de QPSK y 16QAM se observo que la modulación que presenta un menor BER es BFSK en comparación a las otras 3.

4. Gracias a los graficos del 5 al 8 se comprobó que la curva del MMSE siempre va a hacer mejor que la del ecualizador Zero Forcing, pero las modulaciones 16QAM y QPSK al ser complejas estas necesitan de mayor potencia para que el BER sea menor.

5. Pero con estas dos modulaciones podemos afirmar que el ZF es inmune a las modulaciones BPSK y BFSK es decir que no afecto para nada su grafica.

6. Se comprobó mediante las graficas mostradas en la sección anterior, que el método ZFBF necesita de una mayor cantidad de potencia para que el BER mejore lo cual no ocurre con el método MMSE, solamente con un cambio de modulación comprobamos que el MMSE mejora su probabilidad de error considerablemente.

7. Por lo tanto podemos concluir que al hacer cambio de modulación, usando la modulación BFSK, el mejor método para emplear si queremos que nuestro BER mejore con menor potencia, es con el método MMSE, ya que la probabilidad de error mejora considerablemente.

8. Mientras que en las modulaciones 16QAM y QPSK, la diferencia entre la curva del MMSE y la del ZF es muy pequeña, por lo que se puede utilizar el ecualizador ZF en vez del MMSE ya que

como sabemos este es menos complejo de implementar, debido a que el ZF solo se necesita conocimiento del canal; en cambio para el ecualizador MMSE se necesita conocer la matriz de covarianza y la señal de referencia.

Recomendaciones

1. Se puede variar la velocidad de transmisión para utilizarlo con modulaciones más complejas como es el caso de 64 QAM, 256 QAM, etc.
2. Otro tipo de método que se puede utilizar para resolver las interferencias co-canales es Spatial Multiplexing o la diversidad de código.
3. Existen muchos ecualizadores que sirven para hallar la matriz de pesos, en este proyecto solo se utilizaron dos, pero se pueden implementar otros más, tales como están en la referencia [1], [4] y [7].

ANEXO A

CODIGO MATLAB DE LA SIMULACIÓN

```
% Codigo en Matlab para la modulacion  BPSK con respecto al BER en un canal Rayleigh fading  con  2 Tx, 2Rx siendo este un canal  MIMO y  ecualizador  Zero Forcing.

clear

N = 10^6;    % numero de bits o simbolos

Eb_N0_dB = [0:25];   % multiple Eb/N0 values

nTx = 2;   % numero de antenas transmisoras

nRx = 2;    % numero de antenas receptoras

%En esta primera parte inicializamos las variables que vamos a utilizar más adelante en la programación, los símbolos a transmitir son  10^6 almacenados en la variable N y las antenas transmisoras y receptoras son 2 respectivamente.

for ii = 1:length(Eb_N0_dB)

   % Transmisor

   ip = rand(1,N)>0.5;   % genera  0,1  con igual probabilidad

   s = 2*ip-1; % modulación BPSK donde el  0 -> -1; y  el  1 -> 0

 sMod = kron(s,ones(nRx,1));    %multiplicamos la matriz s por una matriz (2x1) que contiene  unos
```

```
sMod = reshape(sMod,[nRx,nTx,N/nTx]); % agrupamos en la matriz [nRx,nTx,N/NTx ]

h = 1/sqrt(2)*[randn(nRx,nTx,N/nTx) + j*randn(nRx,nTx,N/nTx)]; % canal Rayleigh

% ruido blanco gaussiano, con  0dB de varianza

n = 1/sqrt(2)*[randn(nRx,N/nTx) + j*randn(nRx,N/nTx)];

% Adision del canal y ruido blanco gaussiano

y = squeeze(sum(h.*sMod,2)) + 10^(-Eb_N0_dB(ii)/20)*n;
```

% En la parte del transmisor es donde se generan en la matriz los números aleatorios, siendo estos valores entre 0 y 1 teniendo una igual probabilidad y esta matriz es de 1xN, luego se modulan mediante un modulador BPSK, donde se almacena su respuesta en la variable s. Al salir del modulador, utilizamos la función kron, para multiplicar la señal por una matriz de 1's, que luego esa matriz queda guardada en a variable: sMod.

La siguiente línea consiste en utilizar la función reshape para agrupar en vectores de 2x1. Las variables h y n contienen el canal Rayleigh y el ruido blanco gaussiano, respectivamente, debido a que el transmisor tiene conocimiento del canal (CSI). Finalmente antes de transmitir la señal por las dos antenas transmisoras descritas al principio de esta sección; se utiliza la función squeeze, la cual junta suma las matrices de la variable sMod y h (canal Rayleigh) y al resultado de la función squeeze la multiplica por el ruido AWGN, que queda guardado en la variable y.

Finalmente la variable y es la que va a ser transmitida por las dos antenas de transmisión. Que luego van a hacer receptadas por las dos antenas de recepción.

```
% Receptor
```

```
% Forming the Zero Forcing equalization matrix W = inv(H^H*H)*H^H ,  H^H*H is of dimension [nTx x nTx]. In this case [2 x 2] .  Inverse of a [2x2] matrix [a b; c d] = 1/(ad-bc)[d -b;-c a]

   hCof = zeros(2,2,N/nTx)  ;

   hCof(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1);  % aqui obtenemos el termino d

   hCof(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1);  % se obtiene el termino a

   hCof(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % el termino c

   hCof(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % y por ultimo se obtiene el termino b

   hDen = ((hCof(1,1,:).*hCof(2,2,:)) - (hCof(1,2,:).*hCof(2,1,:))); % aqui multiplicamos y restamos como se muestra a continuacion: ad-bc

   hDen = reshape(kron(reshape(hDen,1,N/nTx),ones(2,2)),2,2,N/nTx);  % formatting for division

   hInv = hCof./hDen; % se obtiene la inversa  inv(H^H*H)

   % Forming the MMSE equalization matrix W = inv(H^H*H+sigma^2*I)*H^H

   % H^H*H is of dimension [nTx x nTx]. In this case [2 x 2]

   % Inverse of a [2x2] matrix [a b; c d] = 1/(ad-bc)[d -b;-c a] obtenemos los terminos a, b, c y d los que nos ayudaran a obtener la matrix inversa

   hCofu = zeros(2,2,N/nTx)  ;

   hCofu(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino d
```

```
hCofu(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino a

hCofu(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % termino c

hCofu(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % termino b

hDenu = ((hCofu(1,1,:).*hCofu(2,2,:)) - (hCofu(1,2,:).*hCofu(2,1,:))); % ad-bc term

hDenu = reshape(kron(reshape(hDenu,1,N/nTx),ones(2,2)),2,2,N/nTx);%formatting for division

hInvu = hCofu./hDenu; % inv(H^H*H) del MMSE

hMod = reshape(conj(h),nRx,N); %se realiza la siguiente operacion H^H

yMod = kron(y,ones(1,2)); % formatting the received symbol for equalization

yMod = sum(hMod.*yMod,1); % H^H * y

yMod = kron(reshape(yMod,2,N/nTx),ones(1,2)); % formatting

yHat = sum(reshape(hInv,2,N).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y

% En el lado del receptor, las primeras líneas sirven para poder obtener la inversa de la matriz h, ya que como vimos en secciones anteriores para poder hallar los pesos de la matriz W tenemos q obtener la inversa de la matriz h, y en vez de usar algoritmos como while o for, todo se hace mediante el uso de matrices. Y usamos primero reshape y luego la función kron para agrupar y  multiplicar matrices y poder obtener la inversa de la matriz H.

% receiver – se toma la decision del decodificador para el ZF
```

```
    ipHat = real(yHat)>0;
% receiver - se toma la decision del decodificador para el MMSE
    ipHatu = real(yHatu)>0;
% contando los errores del Zero Forcing
            nErr(ii) = size(find([ip- ipHat]),2);
 % contando los errores del MMSE
            nErru(ii) = size(find([ip- ipHatu]),2);
end
% La variable ipHat e ipHatu almacenan los valores reales de la matriz yHat y  yHatu,
la cual es una matriz compleja, donde hay valores imaginarios y reales.
simBer = nErr/N; % se simula el  ber
EbN0Lin = 10.^(Eb_N0_dB/10);
close all
figure
semilogy(Eb_N0_dB,simBer,'mo-','LineWidth',2); % grafico del ZF
hold on
semilogy(Eb_N0_dB,simBeru,'bp-','LineWidth',2); % grafico del MMSE
axis([0 25 10^-5 0.5])
```

```
grid on

legend( 'sim (nTx=2, nRx=2, ZF)', 'sim (nTx=2, nRx=2, MMSE)');

xlabel('Average Eb/No,dB');

ylabel('Bit Error Rate');

title('BER for BPSK modulation with 2x2 MIMO,  ZF and MMSE equalizer (Rayleigh channel)');
```

```
% Codigo en Matlab para la modulacion  BFSK con respecto al BER en un canal Rayleigh fading  con  2 Tx, 2Rx siendo este un canal

% MIMO  y  ecualizador  Zero Forcing.

 clear

N = 10^5;    % numero de bits o simbolos

T = 8; % symbol duration

t = [0:1/T:0.99]; % sampling instants

tR = kron(ones(1,N),t); % repeating the sampling instants

 Eb_N0_dB = [0:25];   % multiple Eb/N0 values

nTx = 2;   % numero de antenas transmisoras

nRx = 2; ;   % numero de antenas receptoras

 %En esta primera parte inicializamos las variables que vamos a utilizar más adelante en la programación, los símbolos a transmitir son  10^5 almacenados en la variable N y las antenas transmisoras y receptoras son 2 respectivamente.

 for ii = 1:length(Eb_N0_dB)

    % Transmisor

    ip = rand(1,N)>0.5;   % genera  0,1  con igual probabilidad

    freqM = ip+1; % converting the bits into frequency, bit0 -> frequency of 1, bit1 -> frequency of 2
```

```
freqR = kron(freqM,ones(1,T)); % repeating

s = (sqrt(2)/sqrt(T))*cos(2*pi*freqR.*tR); %generating the FSK modulated signal

sMod = kron(s,ones(nRx,1));  %multiplicamos la matriz s por una matriz (2x1) que contiene  unos

sMod = reshape(sMod,[nRx,nTx,size(freqR,2)/nTx]);  % agrupamos en la matriz [nRx,nTx,N/NTx ]

h=1/sqrt(2)*[randn(nRx,nTx,size(freqR,2)/nTx)+j*randn(nRx,nTx,size(freqR,2)/nTx)]; % canal Rayleigh

n = 1/sqrt(2)*[randn(nRx,size(freqR,2)/nTx) + j*randn(nRx,size(freqR,2)/nTx)]; %ruido blanco gaussiano, con  0dB de varianza

% Adision del canal y ruido blanco gaussiano

y = squeeze(sum(h.*sMod,2)) + 10^(-Eb_N0_dB(ii)/20)*n;

 % Receptor

% Forming the Zero Forcing equalization matrix W = inv(H^H*H)*H^H ,  H^H*H is of dimension [nTx x nTx]. In this case [2 x 2] Inverse of a [2x2] matrix [a b; c d] = 1/(ad-bc)[d -b;-c a]

hCof = zeros(2,2,size(freqR,2)/nTx)  ;

hCof(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1);  % aqui obtenemos el termino d

hCof(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1);  % se obtiene el termino a

hCof(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % el termino c
```

```
    hCof(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % y por ultimo se obtiene el termino b

    hDen = ((hCof(1,1,:).*hCof(2,2,:)) - (hCof(1,2,:).*hCof(2,1,:))); % aqui multiplicamos y
restamos como se muestra a continuacion: ad-bc

hDen=reshape(kron(reshape(hDen,1,size(freqR,2)/nTx),ones(2,2)),2,2,size(freqR,2)/n
Tx);  % formatting for division

hInv = hCof./hDen; % se obtiene la inversa  inv(H^H*H)para el ZF

    % Forming the MMSE equalization matrix W = inv(H^H*H+sigma^2*I)*H^H

    % H^H*H is of dimension [nTx x nTx]. In this case [2 x 2]

    % Inverse of a [2x2] matrix [a b; c d] = 1/(ad-bc)[d -b;-c a] obtenemos los terminos a,
b, c y d los que nos ayudaran a obtener la matrix inversa

    hCofu = zeros(2,2,size(freqR,2)/nTx)  ;

    hCofu(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino d

    hCofu(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino a

    hCofu(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % termino c

    hCofu(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % termino b

    hDenu = ((hCofu(1,1,:).*hCofu(2,2,:)) - (hCofu(1,2,:).*hCofu(2,1,:))); % ad-bc term

hDenu=reshape(kron(reshape(hDenu,1,size(freqR,2)/nTx),ones(2,2)),2,2,size(freqR,2)
/nTx);  % formatting for division

   hInvu = hCofu./hDenu; % inv(H^H*H) del MMSE
```

```
hMod = reshape(conj(h),nRx,size(freqR,2)); %se realiza la siguiente operacion  H^H

yMod = kron(y,ones(1,2)); % formatting the received symbol for equalization

yMod = sum(hMod.*yMod,1); % H^H * y

yMod =  kron(reshape(yMod,2,size(freqR,2)/nTx),ones(1,2)); % formatting

yHat = sum(reshape(hInv,2,size(freqR,2)).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y esto es para el ZF

yHatu = sum(reshape(hInvu,2,size(freqR,2)).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y para el MMSE

% En el lado del receptor, las primeras líneas sirven para poder obtener la inversa de la matriz h, ya que como vimos en secciones anteriores para poder hallar los pesos de la matriz W tenemos q obtener la inversa de la matriz h, y en vez de usar algoritmos como while o for, todo se hace mediante el uso de matrices. Y usamos primero reshape y luego la función kron para agrupar y  multiplicar matrices y poder obtener la inversa de la matriz H.

op1 = conv(yHat, sqrt(2/T)*cos(2*pi*1*t)); % correlating with frequency 1

op2 = conv(yHat, sqrt(2/T)*cos(2*pi*2*t)); % correlating with frequency 2

op3 = conv(yHatu, sqrt(2/T)*cos(2*pi*1*t)); % correlating with frequency 1

op4 = conv(yHatu, sqrt(2/T)*cos(2*pi*2*t)); % correlating with frequency 2

% receiver – se toma la decision del decodificador para el ZF

ipHat = [real(op1(T+1:T:end)) < real(op2(T+1:T:end))];
```

```
% receiver - se toma la decision del decodificador para el MMSE
    ipHatu = [real(op3(T+1:T:end)) < real(op4(T+1:T:end))];
    % contando los errores del Zero Forcing
    nErr(ii) = size(find([ip- ipHat]),2);
    % contando los errores del MMSE
    nErru(ii) = size(find([ip- ipHatu]),2);
end
% La variable ipHat almacena los valores reales de la matriz yHat, la cual es una matriz compleja, donde hay valores imaginarios y reales.
simBer = nErr/N; % se simula el  BER del Zero Forcing
simBeru = nErru/N; % se simula el BER para el MMSE
EbN0Lin = 10.^(Eb_N0_dB/10);
close all
figure
semilogy(Eb_N0_dB,simBer,'mo-','LineWidth',2); % grafico del ZF
hold on
semilogy(Eb_N0_dB,simBeru,'bp-','LineWidth',2);% grafico del MMSE
axis([0 25 10^-5 0.5])
```

```
grid on

legend( 'sim (nTx=2, nRx=2, ZF)', 'sim (nTx=2, nRx=2, MMSE)');

xlabel('Average Eb/No,dB');

ylabel('Bit Error Rate');

title('BER for BFSK modulation with 2x2 MIMO,  ZF and MMSE equalizer (Rayleigh channel)');
```

```
%CODIGO MATLAB DE LA SIMULACIÓN QPSK

 % Codigo en Matlab para la modulacion  QPSK con respecto al BER en un canal

% Rayleigh fading  con  2 Tx, 2Rx siendo este un canal  MIMO

% y  ecualizador  Zero Forcing.

 clear

N = 10^5;    % numero de bits o simbolos

T = 8; %  duracion del simbolo

t = [0:1/T:0.99]; % sampling instants

tR = kron(ones(1,N),t); % repeating the sampling instants

 Eb_N0_dB = [0:25];   % multiple Eb/N0 values
```

```
nTx = 2;   % numero de antenas transmisoras

nRx = 2; ;   % numero de antenas receptoras

ipHat = zeros(1,N);

%En esta primera parte inicializamos las variables que vamos a utilizar más adelante en la programación, los símbolos a transmitir son  10^6 almacenados en la variable N y las antenas transmisoras y receptoras son 2 respectivamente.

for ii = 1:length(Eb_N0_dB)

   % Transmisor

       ip = (2*(rand(1,N)>0.5)-1) + j*(2*(rand(1,N)>0.5)-1);

   s = (1/sqrt(2))*ip; % normalization of energy to 1

 sMod = kron(s,ones(nRx,1));    %multiplicamos la matriz s por una matriz (2x1) que contiene  unos

sMod = reshape(sMod,[nRx,nTx,N/nTx]);  % agrupamos en la matriz [nRx,nTx,N/NTx ]

 h = 1/sqrt(2)*[randn(nRx,nTx,N/nTx) + j*randn(nRx,nTx,N/nTx)];

 n = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)]; % white guassian noise, 0dB variance

 % Adision del canal y ruido blanco gaussiano

   y = squeeze(sum(h.*sMod,2)) + 10^(-Eb_N0_dB(ii)/20)*n;

 % Receptor

   % Forming the Zero Forcing equalization matrix W = inv(H^H*H)*H^H
```

```
hCof = zeros(2,2,N/nTx)  ;

hCof(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1);  % aqui obtenemos el termino d

hCof(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1);  % se obtiene el termino a

hCof(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % el termino c

hCof(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % y por ultimo se obtiene el termino b

hDen = ((hCof(1,1,:).*hCof(2,2,:)) - (hCof(1,2,:).*hCof(2,1,:))); % aqui multiplicamos y
restamos como se muestra a continuacion: ad-bc

hDen=reshape(kron(reshape(hDen,1,size(freqR,2)/nTx),ones(2,2)),2,2,size(freqR,2)/n
Tx);  % formatting for división

hInv = hCof./hDen; % se obtiene la inversa  inv(H^H*H)para el ZF

% Forming the MMSE equalization matrix W = inv(H^H*H+sigma^2*I)*H^H

hCof = zeros(2,2,N/nTx)  ;

hCofu(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino d

hCofu(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1) + 10^(-Eb_N0_dB(ii)/10);  % termino a

hCofu(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % termino c

hCofu(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % termino b

hDenu = ((hCofu(1,1,:).*hCofu(2,2,:)) - (hCofu(1,2,:).*hCofu(2,1,:))); % ad-bc term

hDenu=reshape(kron(reshape(hDenu,1,size(freqR,2)/nTx),ones(2,2)),2,2,size(freqR,2)
/nTx);  % formatting for division
```

```
hInvu = hCofu./hDenu; % inv(H^H*H) del MMSE

hMod =  reshape(conj(h),nRx,size(freqR,2)); %se realiza la siguiente operacion  H^H

yMod = kron(y,ones(1,2)); % formatting the received symbol for equalization

yMod = sum(hMod.*yMod,1); % H^H * y

yMod =  kron(reshape(yMod,2,size(freqR,2)/nTx),ones(1,2)); % formatting

yHat = sum(reshape(hInv,2,size(freqR,2)).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y esto es para el ZF

yHatu = sum(reshape(hInvu,2,size(freqR,2)).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y para el MMSE

 % Para ZF

y_re = real(yHat); % real

y_im = imag(yHat); % imaginary

% Para MMSE

y_reu = real(yHatu); % real

y_imu = imag(yHatu); % imaginary

% receiver – se toma la decision del decodificador para el ZF

ipHat(find(y_re < 0 & y_im < 0)) = -1 + -1*j;

ipHat(find(y_re >= 0 & y_im > 0)) = 1 + 1*j;
```

```
ipHat(find(y_re < 0 & y_im >= 0)) = -1 + 1*j;

ipHat(find(y_re >= 0 & y_im < 0)) = 1 - 1*j;

% receiver - se toma la decision del decodificador para el MMSE

ipHatu(find(y_reu < 0 & y_imu < 0)) = -1 + -1*j;

ipHatu(find(y_reu >= 0 & y_imu > 0)) = 1 + 1*j;

ipHatu(find(y_reu < 0 & y_imu >= 0)) = -1 + 1*j;

ipHatu(find(y_reu >= 0 & y_imu < 0)) = 1 - 1*j;

 % contando los errores del Zero Forcing

nErr(ii) = size(find([ip- ipHat]),2);

% contando los errores del MMSE

nErru(ii) = size(find([ip- ipHatu]),2);

end

 % La variable ipHat almacena los valores reales de la matriz yHat, la cual es una matriz compleja, donde hay valores imaginarios y reales.

simBer = nErr/N; % se simula el  BER del Zero Forcing

 simBeru = nErru/N; % se simula el BER para el MMSE

EbN0Lin = 10.^(Eb_N0_dB/10);

close all
```

```
figure

semilogy(Eb_N0_dB,simBer,'mo-','LineWidth',2); % grafico del ZF

hold on

 semilogy(Eb_N0_dB,simBeru,'bp-','LineWidth',2);% grafico del MMSE

 axis([0 25 10^-5 0.5])

grid on

legend( 'sim (nTx=2, nRx=2, ZF)', 'sim (nTx=2, nRx=2, MMSE)');

xlabel('Average Eb/No,dB');

ylabel('Bit Error Rate');

title('BER for QPSK modulation with 2x2 MIMO,  ZF and MMSE equalizer (Rayleigh
channel)');
```

```
%CODIGO MATLAB DE LA SIMULACIÓN 16 QAM

% Codigo en Matlab para la modulacion 16 QAM con respecto al BER en un canal Rayleigh fading con 2 Tx, 2Rx siendo este un canal MIMO y ecualizador Zero Forcing.

clear

N = 10^5;    % numero de bits o simbolos

Eb_N0_dB = [0:30];   % multiple Eb/N0 values

nTx = 2;   % numero de antenas transmisoras

nRx = 2; ;   % numero de antenas receptoras

 alpha16qam = [-3 -1 1 3]; % 16-QAM alphabets

 ipHat = zeros(1,N);

 %En esta primera parte inicializamos las variables que vamos a utilizar más adelante en la programación, los símbolos a transmitir son 10^6 almacenados en la variable N y las antenas transmisoras y receptoras son 2 respectivamente.

for ii = 1:length(Eb_N0_dB)

   % Transmisor

  ip = randsrc(1,N,alpha16qam) + j*randsrc(1,N,alpha16qam);

  s = (1/sqrt(10))*ip; % normalization of energy to 1

  sMod = kron(s,ones(nRx,1));    %multiplicamos la matriz s por una matriz (2x1) que
```

```
contiene  unos

 sMod = reshape(sMod,[nRx,nTx,N/nTx]);

 h = 1/sqrt(2)*[randn(nRx,nTx,N/nTx) + j*randn(nRx,nTx,N/nTx)];  % canal Rayleigh

 n = 1/sqrt(2)*[randn(nRx,N/nTx) + j*randn(nRx,N/nTx)]; % ruido blanco gaussiano

  % Adision del canal y ruido blanco gaussiano

 y = squeeze(sum(h.*sMod,2)) + 10^(-Eb_N0_dB(ii)/20)*n;

 % Receptor

  % Forming the Zero Forcing equalization matrix W = inv(H^H*H)*H^H

 hCof = zeros(2,2,N/nTx)  ;

 hCof(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1);  % aqui obtenemos el termino d

 hCof(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1);  % se obtiene el termino a

 hCof(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % el termino c

 hCof(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % y por ultimo se obtiene el termino b

 hDen = ((hCof(1,1,:).*hCof(2,2,:)) - (hCof(1,2,:).*hCof(2,1,:))); % aqui multiplicamos y
restamos como se muestra a continuacion: ad-bc

 hDen = reshape(kron(reshape(hDen,1,N/nTx),ones(2,2)),2,2,N/nTx);  % formatting for
division

 hInv = hCof./hDen; % se obtiene la inversa  inv(H^H*H)para el ZF
```

```
% Forming the MMSE equalization matrix W = inv(H^H*H+sigma^2*I)*H^H
```

H^H*H is of dimension [nTx x nTx]. In this case [2 x 2]. Inverse of a [2x2] matrix [a b; c d] = 1/(ad-bc)[d -b;-c a] obtenemos los terminos a, b, c y d los que nos ayudaran a obtener la matrix inversa

```
hCofu = zeros(2,2,N/nTx) ;

hCofu(1,1,:) = sum(h(:,2,:).*conj(h(:,2,:)),1) + 10^(-Eb_N0_dB(ii)/10); % termino d

hCofu(2,2,:) = sum(h(:,1,:).*conj(h(:,1,:)),1) + 10^(-Eb_N0_dB(ii)/10); % termino a

hCofu(2,1,:) = -sum(h(:,2,:).*conj(h(:,1,:)),1); % termino c

hCofu(1,2,:) = -sum(h(:,1,:).*conj(h(:,2,:)),1); % termino b

hDenu = ((hCofu(1,1,:).*hCofu(2,2,:)) - (hCofu(1,2,:).*hCofu(2,1,:))); % ad-bc term

hDenu = reshape(kron(reshape(hDenu,1,N/nTx),ones(2,2)),2,2,N/nTx); % formatting for division

hInvu = hCofu./hDenu; % inv(H^H*H) del MMSE

hMod = reshape(conj(h),nRx,N); %se realiza la siguiente operacion H^H

yMod = kron(y,ones(1,2)); % formatting the received symbol for equalization

yMod = sum(hMod.*yMod,1); % H^H * y

yMod = kron(reshape(yMod,2,N/nTx),ones(1,2)); % formatting

yHat = sum(reshape(hInv,2,N).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y esto es para el ZF
```

```
yHatu = sum(reshape(hInvu,2,N).*yMod,1); % se realiza la siguiente operacion: inv(H^H*H)*H^H*y para el MMSE

  % Para ZF

y_re = real(yHat); % real

y_im = imag(yHat); % imaginary

  %Para MMSE

y_reu = real(yHatu); % real

y_imu = imag(yHatu); % imaginary

% receiver – se toma la decision del decodificador para el ZF parte real

ipHat_re(find(y_re< -2/sqrt(10)))          = -3;

ipHat_re(find(y_re > 2/sqrt(10)))          =  3;

ipHat_re(find(y_re>-2/sqrt(10) & y_re<=0))  = -1;

ipHat_re(find(y_re>0 & y_re<=2/sqrt(10)))   =  1;

% Parte imaginaria

ipHat_im(find(y_im< -2/sqrt(10)))          = -3;

ipHat_im(find(y_im > 2/sqrt(10)))          =  3;

ipHat_im(find(y_im>-2/sqrt(10) & y_im<=0))  = -1;

ipHat_im(find(y_im>0 & y_im<=2/sqrt(10)))   =  1;
```

```
 ipHat = ipHat_re + j*ipHat_im;

 % receiver - se toma la decision del decodificador para el MMSE

 ipHat_reu(find(y_reu< -2/sqrt(10)))          = -3;

 ipHat_reu(find(y_reu > 2/sqrt(10)))          =  3;

 ipHat_reu(find(y_reu>-2/sqrt(10) & y_reu<=0))  = -1;

 ipHat_reu(find(y_reu>0 & y_reu<=2/sqrt(10)))   =  1;

% Parte imaginaria

 ipHat_imu(find(y_imu< -2/sqrt(10)))          = -3;

 ipHat_imu(find(y_imu > 2/sqrt(10)))          =  3;

 ipHat_imu(find(y_imu>-2/sqrt(10) & y_imu<=0))  = -1;

 ipHat_imu(find(y_imu>0 & y_imu<=2/sqrt(10)))   =  1;

 ipHatu = ipHat_reu + j*ipHat_imu;

    % contando los errores del Zero Forcing

   nErr(ii) = size(find([ip- ipHat]),2);

   % contando los errores del MMSE

   nErru(ii) = size(find([ip- ipHatu]),2);

end

simBer = nErr/N; % se simula el  BER del Zero Forcing
```

```
 simBeru = nErru/N; % se simula el BER para el MMSE

EbN0Lin = 10.^(Eb_N0_dB/10);

 close all

 figure

semilogy(Eb_N0_dB,simBer,'mo-','LineWidth',2); % grafico del ZF

hold on

 semilogy(Eb_N0_dB,simBeru,'bp-','LineWidth',2);% grafico del MMSE

 axis([0 30 10^-3 1])

grid on

legend( 'sim (nTx=2, nRx=2, ZF)', 'sim (nTx=2, nRx=2, MMSE)');

xlabel('Average Eb/No,dB');

ylabel('Bit Error Rate');

title('BER for 16QAM modulation with 2x2 MIMO,  ZF and MMSE equalizer (Rayleigh channel)');
```

BIBLIOGRAFIA

[1] Beamforming; http://www.comm.utoronto.ca/~rsadve/Notes/BeamForming.pdf; March 2007.

[2] Jraifi Abdelouahed1† and El Hassan Saidi2††, “Optimization of MIMO Systems in a Correlated Channel”, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.2, February 2008

[3] YANG Liang, QIN Jiayin, “Cochannel Interference and Its Effect on the Capacity of Multi-Antenna Systems in Ricean-Fading Channels”, Journal of Communication and Computer, ISSN1548-7709, Volume 2, No.1 (Serial No.2); Jan.2005.

[4] Cangahuamín Jácome César Jermánico, Ing. Darío Duque M.Sc, Ing. Gonzalo Olmedo M.Sc; “ESTUDIO, ANÁLISIS Y SIMULACIÓN DEL BEAMFORMING EN ANTENAS INTELIGENTES PARA UN ENTORNO DE TELEFONÍA CELULAR CDMA”; Departamento de Eléctrica y Electrónica de la ESPE Campus Sangolquí, Ecuador; 2006.

[5] Taesang Yoo and Andrea Goldsmith” Optimality of Zero-Forcing Beamforming with Multiuser Diversity”; Dept. of Electrical Engineering, Stanford University, Stanford; 2005

[6] Taesang Yoo, Student Member, IEEE, and Andrea Goldsmith, Fellow, IEEE; ”On the Optimality of Multiantenna Broadcast Scheduling Using Zero-Forcing Beamforming”; IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 24, NO. 3, MARCH 2006.

[7] Young-il Shin, Tae-Sung Kang, Hyung-Myung Kim; “AN EFFICIENT RESOURCE ALLOCATION FOR MULTIUSER MIMO-OFDM SYSTEMS WITH ZERO-FORCING BEAMFORMER”; the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications; 2007

[8] TUAN NGUYEN; “NULL DEPTH TRADE OFF FOR OUTPUT POWER REDUCTION IN A DOWNLINK ADAPTIVE ANTENNA ARRAY”; VICTORIA UNIVERSITY OF TECHNOLOGY MELBOURNE, AUSTRALIA; 2006.