Práctica 3. Simulación de un sistema básico de comunicaciones - I

1. Introducción

El objetivo de la práctica es crear un simulador de un sistema de comunicaciones básico. Para ello, es necesario simular la señal analógica mediante procedimientos digitales, debiendo ajustar adecuadamente los parámetros de simulación y siendo consciente de sus limitaciones.

Inicialmente se supondrá un sistema con modulación MPSK, pero posteriormente podrá extenderse a otros casos. Adicionalmente, se dispondrá de las funciones necesarias para añadir un sistema TCM (Trellis Coded Modulation), de forma que se podrá estudiar la ganancia de prestaciones que proporciona este sistema y comparar con las cotas obtenidas en teoría.

Las simulaciones por ordenador son necesarias e incluso imprescindibles en la industria actual. Sus aplicaciones son muy numerosas y están presentes en multitud de campos. En lo referente al ámbito de la ingeniería de Telecomunicación, cabe mencionar las simulaciones de multitud de sistemas (microondas, comunicaciones, tratamiento de señal, etc.). Las herramientas para dichas simulaciones pueden resultar caras, pero siempre son más económicas y eficientes que las realizaciones físicas de aquellos sistemas que simulan. Por ejemplo, para diseñar redes de comunicaciones móviles existen softwares de planificación que permiten simular dichos sistemas sin necesidad de realizar un despliegue real. Evidentemente, las simulaciones sólo son un primer paso en el desarrollo o diseño de un sistema, pero que ahorra mucho tiempo y costes.

En una simulación es fundamental que el comportamiento de todas las variables del sistema se ajuste lo máximo posible a cómo sería en la realidad para que el sistema pueda considerarse fiable y poder tomar decisiones en función de los resultados obtenidos. Además, debe ser transparente al usuario final, éste debe ver las señales tal y como las vería en un equipo de medida (osciloscopio, analizador, etc.). También es importante que el sistema de simulación permita variar los parámetros de la simulación, para que el sistema sea lo más flexible posible.

En nuestro caso vamos a intentar, en la medida de lo posible, que la implementación sea eficiente aprovechando las posibilidades que ofrece Matlab y que la configuración del sistema sea lo más genérica y versátil posible.

2. Estudio previo *(Realizar cuando se estudie en clases teóricas)*

Proponer y analizar una constelación de 8 símbolos distinta a las vistas en clase (8-PSK y 8-PAM). Puede ser inventada o buscada en la bibliografía. No es necesario encontrar la mejor constelación posible. Lo que se pide es elegir una y analizarla.

Se deberá entregar en la práctica:

- Nombre. La constelación irá acompañada de un nombre en siglas o palabra, inventado o ya existente, pero distinto a PSK y ASK, que nos permitirá identificarla y posteriormente crear alguna función de código en Matlab que tenga que ver con esta constelación.

- Constelación. Un esquema de la disposición de los 8 símbolos de la constelación elegida en el espacio de señal fase/cuadratura. Con las medidas necesarias para poderla implementar o reproducir los cálculos.

- Ganancia sin TCM. El cálculo de la ganancia asintótica (o pérdida) de la constelación con respecto a 8-PSK, es decir, comparando dos sistemas que usen esas modulaciones sin codificación de canal.

3. Trabajo de laboratorio

1. Sistemas a simular

Inicialmente la práctica está dedicada a la simulación de un sistema de comunicaciones con modulación de fase M-PSK, aunque posteriormente puede ampliarse a otras modulaciones en fase y amplitud.

En la figura 1, se observa el esquema básico de un sistema de modulación mediante el uso de dos ramas, fase y cuadratura. Una vez generado el símbolo, se procede al conformado del pulso en banda base de las ramas en fase y en cuadratura para posteriormente realizar el desplazamiento frecuencial ("up-conversion").

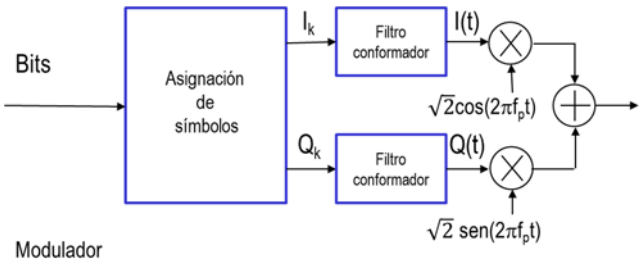

Fig. 1. Modulador del sistema de comunicaciones

En la figura 2 se presenta la etapa demoduladora del sistema sin considerar el sincronismo. Una vez construido todo el sistema, el objetivo es obtener las prestaciones en cuanto a tasa de error, relación señal a ruido y capacidad para las distintas modulaciones.

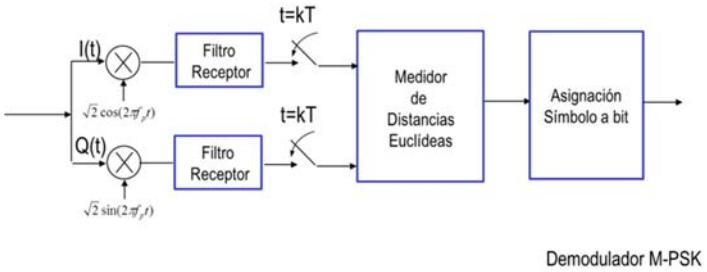

Fig. 2. Demodulador del sistema de comunicaciones

2. Realización práctica

El objetivo de la práctica es simular la transmisión de una señal modulada en fase y amplitud. Debe tenerse en cuenta que la señal simulada es analógica en canal y esa es la impresión que debe tener el usuario del sistema final, aunque Matlab es un sistema que trabaja principalmente con secuencias discretas (agrupadas en vectores).

Habitualmente cuando se realiza una simulación se sigue un esquema de programación análogo al sistema a simular, siendo recomendable implementar cada módulo del esquema en una función diferente, por lo que éste es el procedimiento a seguir.

Se pretende también, consolidar los fundamentos básicos de una simulación y estudiar la dualidad tiempo-frecuencia para la correcta realización de la simulación, a la vez que valorar las limitaciones de la simulación. El sistema de esta práctica servirá para las dos siguientes.

Abrir el fichero ***p1_1.m*** que recoge todas las funciones para simular el sistema. Dichas funciones están contenidas en ficheros con formato ****.p***.

Se pretende ir observando la evolución de la señal a lo largo del sistema, viendo las entradas y salidas de los diferentes bloques simulados mediante funciones.

Fuente discreta sin memoria

1. El primer sistema a simular es la generación de los bits aleatorios, sin memoria, a transmitir. La función `fuente` utiliza la función `rand` y devuelve la secuencia de bits de información aleatorios con el número de bits que se desea simular como argumento.

```
bits = fuente(numero_de_bits)
```

   NOTA: se fija la semilla de los generadores aleatorios de Matlab para poder depurar mejor los errores y hacerlos repetibles. Para fijar las semillas (en este caso a -1, pero puede ser cualquier número) de los dos generadores que se usarán en las prácticas se emplean los comandos:

```
randn('seed',-1);    rand('seed',-1);
```

Asignación de símbolos

2. De forma general se van a generar modulaciones de fase y amplitud, aunque de momento se trabajará, por simplicidad, con el caso concreto 4PSK y 8PSK. En primer lugar, crearemos una función que devuelva la información de las coordenadas de la constelación en un formato genérico que usaremos durante toda la práctica. El formato consiste en una matriz de M filas y dos columnas (Mx2), donde el número de símbolos son las filas y las columnas son las coordenadas en fase (1º columna) y cuadratura (2ª columna) de los símbolos de la constelación. El orden de los símbolos en la tabla es importante, por lo que se dispondrán los símbolos en la tabla en el orden que indica su código binario convertido en número decimal. A continuación, podemos ver un ejemplo del formato elegido para guardar la constelación en Matlab.

Tabla constelación:

Índice símbolo	Símbolo binario	I	Q
1	00	+0.7	+0.7
2	01	-0.7	+0.7
3	10	-0.7	-0.7
4	11	+0.7	-0.7

Variable constelación en Matlab:

```
constelacion = [+0.7 +0.7;
                -0.7 +0.7;
                -0.7 -0.7;
                +0.7 -0.7];
```

Para que posteriormente el ajuste del nivel de ruido sea más sencillo, se procurará que todas las funciones de generación de constelaciones devuelvan los símbolos **normalizados a energía promedio unidad**. Para comprobar este aspecto, se utiliza una función que devuelve la energía promedio de los símbolos de una constelación genérica:

```
Eav = energia_promodio_constelacion(constelación)
```

También es de utilidad una función que permita visualizar la constelación en pantalla

```
ver_constelacion(constelacion)
```

Como muestra la tabla anterior, el orden en el que introducimos las coordenadas en la matriz de Matlab es importante, ya que un símbolo estará asociado a sus bits según su posición en la tabla. Con el ejemplo de la tabla vista obtendríamos la siguiente constelación.

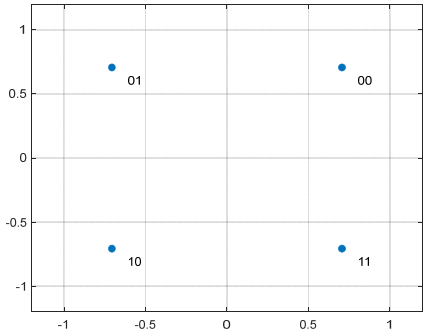

Fig. 3. Constelación QPSK

Esta constelación puede funcionar, pero la asignación no es Gray (el símbolo '00' es adyacente al '11') y eso puede crear más errores en promedio.

La constelación 4PSK con codificación Gray viene dada con la función:

```
constelacion = constelacion_4psk()
```

El siguiente paso es tener una función que tome los bits cada $\log_2 M$ y que proporcione a su salida las coordenadas en fase y cuadratura de los símbolos. Con el fin de mantener un mismo criterio, en la conversión de bit a símbolo y de símbolo a bit, se emplea el criterio de bit más significativo a la izquierda, tal y como se presenta en la tabla anterior.

```
[Ik,Qk] = asignacion_simbolos(bits, constelacion)
```

Portadora

Para la realización del sistema es necesaria una frecuencia portadora, que es la que vamos a abordar a continuación, generándola y representándola en tiempo y frecuencia.

Matlab trabaja con vectores y cada elemento de los mismos representa una muestra de la señal simulada. Si sólo contamos con el vector, no conocemos cada cuanto tiempo se han tomado esas muestras. Es por ello que, para cualquier simulación de una señal variante con el tiempo, hay que definir el vector de tiempo con sus instantes de muestreo.

Para un muestreo correcto se debe aplicar el criterio de Nyquist: $\mathbf{f_m >= 2f_{max}}$, donde f_m es la frecuencia de muestreo y f_{max} la frecuencia máxima.

Considerando únicamente el espectro de la portadora, una delta situada en la frecuencia f_p, a frecuencia máxima es f_p por lo que la frecuencia de muestreo podría ser $f_m=2f_p$.

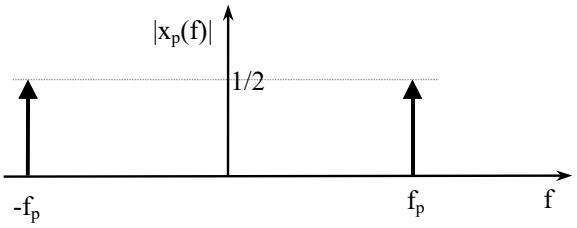

Fig. 4. Espectro de la portadora f_p

En el caso de información que module la portadora, f_m y f_{max} varían. Por ejemplo, para una modulación BPSK en la que la mayor parte de la energía de la señal moduladora se encuentra en el lóbulo principal y, en consecuencia, el ancho de banda de ésta viene dado por el ancho de banda entre ceros, la frecuencia de muestreo sería $f_m=2(f_p+f_s)$, siendo f_s es la frecuencia de la señal moduladora, es decir, la tasa de símbolo R_s.

Definida la frecuencia de muestreo (f_m), queda definido el intervalo de muestreo ($T_m=1/f_m$) y podemos definir la variable temporal t como:

```
t=0:1/fm:tiempo_de_simulacion-1/fm
```

`tiempo_de_simulacion` puede ajustarse como un número de muestras de la señal o como un tiempo absoluto. Esto es de vital importancia, pues la secuencia debe ser finita pero tener suficientes muestras para que los resultados parezcan analógicos en su representación. En nuestras simulaciones su valor normalmente vendrá dado por el número de bits que queramos simular.

Vamos ahora a analizar la portadora.

3. Abrir *portadora.m* que simula la generación de la portadora (sin modular).
   Tomar los siguientes valores: `numero_de_bits=1, fp=1000 y fb=10`.
   Visualizar la señal temporal, observar si aparece alguna anomalía y, en tal caso, proponer una posible solución.

4. Realice una representación del espectro del tono con el eje de frecuencias apropiado en Hz: `f=-fm/2:fm/N:fm/2-fm/N` (si se usa fftshift para centrar en 0 la fft). Compare la visualización con la frecuencia mínima de Nyquist, utilizando un número mayor de puntos y sobremuestreando la señal en tiempo.

Modulador

Volvemos a ***p1_1.m***, donde ahora ya se conocen los valores adecuados de *fp, fb* y *fm* para realizarse la modulación de la señal PSK. La salida de la fuente de información nos suministra un bit cada $T_b=1/R_b$. El módulo de conversión de símbolos toma los bits cada $\log_2 M$ para dar lugar a la secuencia de símbolos en coordenadas fase y cuadratura (I_k, Q_k). Esta secuencia de símbolos es la entrada al bloque encargado de realizar la conformación del pulso en banda base. Para ello, es necesario realizar un proceso de expansión o conformado sobre la señal de entrada, en el que por cada símbolo de partida, deben generarse f_m/R_s muestras, donde R_s es la velocidad de transmisión en baudios o tasa de símbolo, ($R_s=R_b/\log_2 M$).

En primer lugar, se va a considerar un pulso conformador rectangular. Dicho proceso sobre la rama en fase del modulador puede observarse en la siguiente figura:

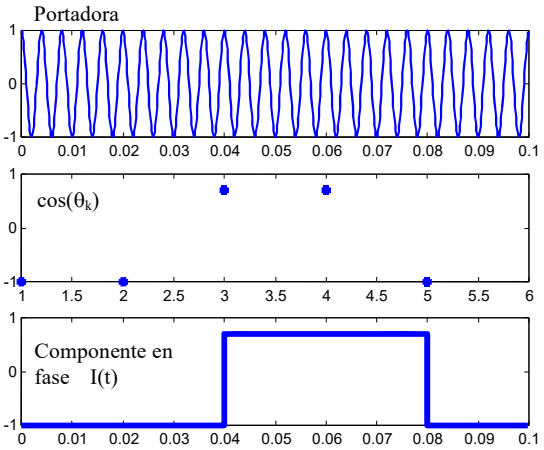

Fig. 5. Representación de la portadora, I_k e I(t)

En la figura se representan los primeros 100 ms de la transmisión, los 5 primeros valores de amplitud de la componente en fase y la expansión realizada al vector de valores de la componente en fase para dar lugar a I(t).

5. Para conformar las formas de onda de las ramas fase y cuadratura a partir de los valores de los símbolos vamos a construir una función que realice el proceso de conformado de manera que por cada valor de I_k ó Q_k deben salir f_m/f_s muestras. Para encontrar una f_m adecuada que cumpla con el criterio de Nyquist, que permita representar funciones con comodidad y facilite

el conformado, en lugar de ajustarnos al mínimo valor teórico, vamos a elegir un valor de f_m para el que la relación **f_m/f_s sea un número entero y que f_m cumpla con Nyquist**.

Para conformar la secuencia con mayor facilidad generaremos un tren de impulsos cuyo valor será I_k en t=kT y cero en el resto para la rama fase y Q_k para la de cuadratura. A continuación, para conformar la señal, sólo queda filtrar con el pulso.

Así pues, la función tendrá como entradas las secuencias I_k y Q_k, el pulso, la frecuencia de muestreo y la frecuencia de símbolo. La salida de esta función debe ser la componente en fase I(t) y la componente en cuadratura Q(t) debidamente muestreadas a f_m muestras por segundo.

```
[I,Q]= filtro_tx(Ik,Qk,fm,fs,pulso)
```

Para poder probar esta función utilizaremos inicialmente un **pulso rectangular** de tamaño $N=f_m/f_s$, que podemos generar con la función `ones` y lo escalaremos dividiendo por el factor $\sqrt{N}$ (para tener E_s=1).

Para ello lo más conveniente es crear una función:

```
[pulso,retardo] = rect(fm,fs)
```

cuya salida es el pulso rectangular y el retardo que introduce el filtro, necesario para muestrear más adelante.

Finalmente es necesario una función que realice la modulación, es decir, que tome como entrada las componentes de fase y cuadratura, las module y devuelva la señal modulada x(t)

```
x = modulador(I,Q,fm,fp)
```

Utilizando todas las funciones vistas module la señal generada tomando los siguientes valores: *f_p=1000 Hz, f_b=100 Hz, M=4 y 10 bits transmitidos*. Observe los símbolos en las diferentes etapas del modulador.

Demodulador

Una vez realizado todo el proceso de modulación hay que proceder a realizar el proceso inverso, la demodulación. Inicialmente se considerará un canal ideal, es decir, sin ruido y sin ningún tipo de distorsión, con lo que la señal recibida (r) será la misma que la transmitida (x).

La señal recibida se descompone en dos ramas para obtener sus componentes en fase y en cuadratura tras pasar por el filtro receptor. Posteriormente se ha de realizar el muestreo, la obtención de las distancias entre el símbolo recibido y los símbolos de la constelación y finalmente, eligiendo la menor de dichas distancias, decidir el símbolo decodificado. Por simplicidad, en todo el proceso del demodulador se obvia el problema del sincronismo de portadora y de símbolo.

6. La función que realiza el proceso de demodulación toma la señal recibida (r) como entrada y como salida proporciona las componentes en fase y en cuadratura demoduladas:

```
[xI,xQ] = demodulador(r,fm,fp)
```

7. La función que realiza el filtro receptor que hemos considerado (rectangular). La función será genérica recibiendo el pulso como argumento:

```
[I,Q] = filtro_rx(xI,xQ,pulso)
```

8. La función que realiza el muestreo de las coordenadas fase y cuadratura de los símbolos teniendo en cuenta el retardo que introduce el filtro que se está empleando:

```
[Ik,Qk] = muestreo(I,Q,fm,fs,retardo)
```

   En este punto, se tienen los símbolos recibidos en fase (I_k) y cuadratura (Q_k), por lo que lo siguiente es calcular las distancias euclídeas a cada uno de los símbolos de la constelación.

9. La función que calcula las distancias de los símbolos recibidos a los símbolos de la constelación y dé como salida una matriz de distancias de tamaño M x número_de_simbolos. El orden de los símbolos debe ser el mismo que el usado para codificarlos.

```
D = distancias(Ik,Qk,constelacion)
```

10. La función que decide el símbolo más cercano y convierte el resultado de la función anterior en bits de salida, siguiendo de nuevo el criterio del bit más significativo a la izquierda.

```
bits = decodificador_map(D)
```

11. Utilizando todas las funciones vistas, compruebe el correcto funcionamiento del sistema para un canal sin ruido utilizando: f_p=1000 Hz, f_b=100 Hz, M=4 y 10 bits transmitidos

Canal AWGN

Una vez que se tienen los procesos de modulación y demodulación funcionando correctamente en un canal ideal, se puede proceder a añadir el canal de comunicaciones, que en nuestro caso va a ser AWGN (ruido aditivo gaussiano blanco). En el simulador hasta ahora desarrollado, tendremos que añadir este ruido gaussiano blanco a la señal recibida a la entrada al demodulador:

```
r = x + n
```

a partir de E_s, se obtiene E_b y para una E_b/N_0 fijada se generan las muestras de ruido AGWN que se añaden a la señal *x*.

Para ello abrir el fichero ***p1_2.m***, que contiene lo mismo que el anterior, pero añadiendo el canal AWGN.

12. Compruebe la tasa de error estimada en la simulación con la teórica para E_b/N_0 = 3 dB, generando un número de bits suficientemente grande como para que el resultado sea fiable. Observe a su vez la constelación en recepción.

Canal limitado en banda

Hasta ahora hemos utilizado pulso conformador rectangular. Como bien es sabido, este sistema genera una señal de ancho de banda infinito. En la práctica, todo canal de comunicaciones presenta un ancho de banda limitado, por lo que debemos adaptar el ancho de banda de nuestra señal al ancho de banda disponible en el canal. Esto puede dar lugar a la aparición de Interferencia Intersimbólica (ISI) que debe evitarse o minimizarse con el adecuado diseño de los filtros terminales. Ahora se van a considerar como filtros terminales del sistema un pulso conformador raíz cuadrada del coseno realzado, consiguiendo así que la limitación en banda requerida en todo sistema de comunicaciones digitales real no afecte a las prestaciones.

Abra el fichero ***p1_3.m***, que contiene lo mismo que *p1_2.m*, pero sustituyendo los filtros rectangulares por filtros raíz cuadrada del coseno realzado.

La función `rcosfir` de MatLab permite diseñar un filtro coseno realzado de manera inmediata mediante la técnica del enventanado de la respuesta impulsional, obteniendo así un filtro FIR:

```
rcosfir(R, N_T, RATE, T, FILTER_TYPE)
```

R: el factor de roll-off
N_T: número de lóbulos del tamaño de la duración del filtro a izquierda y derecha
RATE: número de muestras por símbolo
T: periodo de símbolo
FILTER_TYPE: - *coseno realzado si no ponemos nada o* `'normal'`
- *raíz cuadrada del coseno realzado si ponemos* `'sqrt'`.

Emplearemos los parámetros de frecuencia de muestreo y frecuencia de símbolo del apartado anterior y una longitud del filtro de 5 periodos de símbolos a la izquierda y derecha del máximo y factor de roll-off 0.5.

13. Para finalizar la simulación del modulador con pulso conformador raíz cuadrada del coseno realzado sólo queda programar la función alternativa a la anterior rectangular, que devuelva el retardo producido por el filtro para que funcione perfectamente el muestreador:

```
[pulso,retardo]=rcos(fm,fs,beta,taps)
```

14. Obtenga la representación del filtro coseno realzado y el retardo del mismo.

15. Represente las gráficas P_b en función de E_b/N_0 para:
    a. Simulación de 4PSK
    b. 4PSK teórica usando la función Q(): $P_b = Q\left(\sqrt{2 \cdot \frac{E_b}{N_0}}\right)$
    c. Cota exponencial: $Q(x) \leq \frac{1}{2} e^{-x^2/2}$

16. Lo mismo para:

a. Simulación de 8PSK

b. 8PSK teórica según la expresión: $$P_b^{MPSK} \simeq \frac{2}{\log_2 M} Q\left(\sqrt{2\frac{E_b}{N_0}\log_2 M}\ sen\frac{\pi}{M}\right)$$

c. Simulación de la constelación propuesta en el estudio previo (programar la función correspondiente)

Observaciones:

- Las gráficas deben tener leyendas de cada una de las curvas.
- Debe darse una breve explicación del comportamiento de dichas curvas.
- Realizarlo para el caso de canal limitado en banda.
