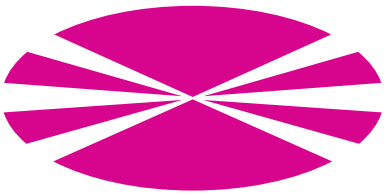

Grado en Ingeniería Informática

Software de Comunicaciones (SC)

Práctica 1

Primera parte

Modulaciones digitales

Curso 2019-20

1. Modelo de canal AWGN

En esta primera parte de la práctica evaluaremos mediante simulación el rendimiento de distintas modulaciones en canales AWGN (*Additive White Gaussian Noise*).

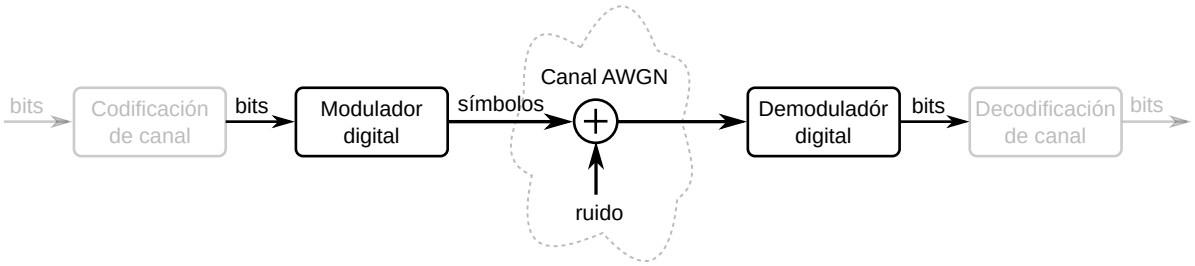

Figura 1: Sistema básico de comunicaciones con canal AWGN.

Nos centraremos en las modulaciones utilizadas por el estándar 802.11n: 2-PSK (BPSK), 4-PSK (QPSK), 16-QAM y 64-QAM. Los bits a transmitir serán generados aleatoriamente. Cada símbolo a transmitir $\mathbf{s}$ será construido a partir de $k = \log_2 M$ bits utilizando una de las cuatro posibles modulaciones (con $M = 2, 4, 16$ o 64).

Utilizaremos el modelo discreto equivalente, por lo que podemos modelar el canal en cada instante de tiempo como

$$\mathbf{r} = \mathbf{s} + \mathbf{n},$$

donde $\mathbf{r}$ es el valor recibido por el demodulador y cada componente del vector $\mathbf{n}$ sigue una distribución Gaussiana de media 0 y varianza $N_0/2$. La calidad del canal vendrá determinada por el valor E_b/N_0 (la relación entre la energía media por bit transmitido y la varianza del ruido), que será un parámetro configurable de la simulación. El valor E_b/N_0 se suele expresar en decibelios:

$$E_b/N_0(\text{dB}) = 10 \log_{10} \frac{E_b}{N_0}.$$

En el receptor, el demodulador convertirá los símbolos recibidos (los transmitidos más el ruido) en bits. Para ello decidirá en función de cada valor recibido $\mathbf{r}$, eligiendo el símbolo de la modulación más cercano y transformando el resultado en bits.

Finalmente calcularemos la probabilidad de error de bit comparando la secuencia de bits original con la demodulada.

2. Desarrollo

Los pasos a seguir en la práctica son los siguientes (se recomienda intentar replicar el ejemplo de la Sección 4 de este enunciado, hecho sobre 4-PAM, antes de proceder a PSK y QAM):

1. Generar N bits de forma aleatoria con probabilidad $p(0) = p(1) = 0.5$, siendo N un valor configurable (para depurar el código se recomienda inicialmente elegir un valor de N bajo). Funciones útiles: `randi, rand`.

2. Modular los bits para obtener los símbolos que serán transmitidos por el canal. Se recomienda hacer un `reshape` del vector de bits a una matriz $k \times N/k$ y convertir el resultado a un vector decimal. El vector decimal se puede utilizar para indexar un array que contenga los M posibles símbolos de la modulación. Funciones útiles: `bi2de`.

   Nota: dado que las modulaciones que se van a usar son de dimensión 2 (PSK y QAM), es aconsejable usar números complejos en lugar de vectores para representar los símbolos, el ruido y los valores recibidos. Es decir, en lugar de trabajar con un símbolo $\mathbf{s} = (1, -3)$, trabajar con el símbolo $s = 1 - 3i$ (un número complejo se puede ver como un vector en un espacio de 2 dimensiones). Esto facilita las operaciones.

3. Para diferentes valores de E_b/N_0 (medido en decibelios) del canal AWGN:

   - Generar el vector de ruido Gaussiano de media 0 y varianza $N_0/2$ por dimensión (que será del mismo tamaño que el vector de símbolos a transmitir) y sumarlo al vector de símbolos. Dado que los valores de E_b/N_0 están en decibelios, si llamamos x al valor de E_b/N_0(dB) en el que estamos interesados podemos hallar el valor de E_b/N_0 correspondiente como:

     $$E_b/N_0 = 10^{x/10}$$

     para, a continuación, hallar N_0 como

     $$N_0 = \frac{E_b}{E_b/N_0}.$$

     siendo E_b la energía promedio por bit (que viene determinada por los símbolos de la modulación).

     Funciones útiles: `randn`. Es importante recordar que

     $$X \sim \mathcal{N}(0,1) \Rightarrow \sqrt{\frac{N_0}{2}} X \sim \mathcal{N}(0, N_0/2).$$

   - Demodular los valores recibidos. Esto implica elegir el símbolo de la modulación más cercano en distancia euclídea al valor recibido para cada instante de tiempo. Funciones útiles: `repmat` (para calcular las distancias a todos los vectores de la modulación sin necesidad de iterar).
   - Calcular los errores cometidos en la demodulación comparando la secuencia de bits recibida con la original.
   - Calcular la BER (*bit error rate*).

   Se deben elegir valores de E_b/N_0 que permitan dibujar una curva hasta una probabilidad de error de 10^{-5}.

4. Representar las curvas de BER (eje Y, en logarítmico) frente a E_b/N_0(dB) de las cuatro modulaciones en la misma gráfica. Funciones útiles: `semilogy`. La figura resultante debe ser similar a la mostrada en la Figura 2 (en la figura se representa sólo el rendimiento de BPSK).

5. Comparar los resultados experimentales de BER con los valores de probabilidad de error de símbolo vistos en clase de teoría. ¿Coincide la probabilidad de error de bit con la probabilidad de error de símbolo? ¿Por qué?

6. Repite la prueba para 16-QAM utilizando Gray Mapping (se puede implementar simplemente reordenando el vector `modulacion`). ¿Mejoran los resultados? ¿Por qué?

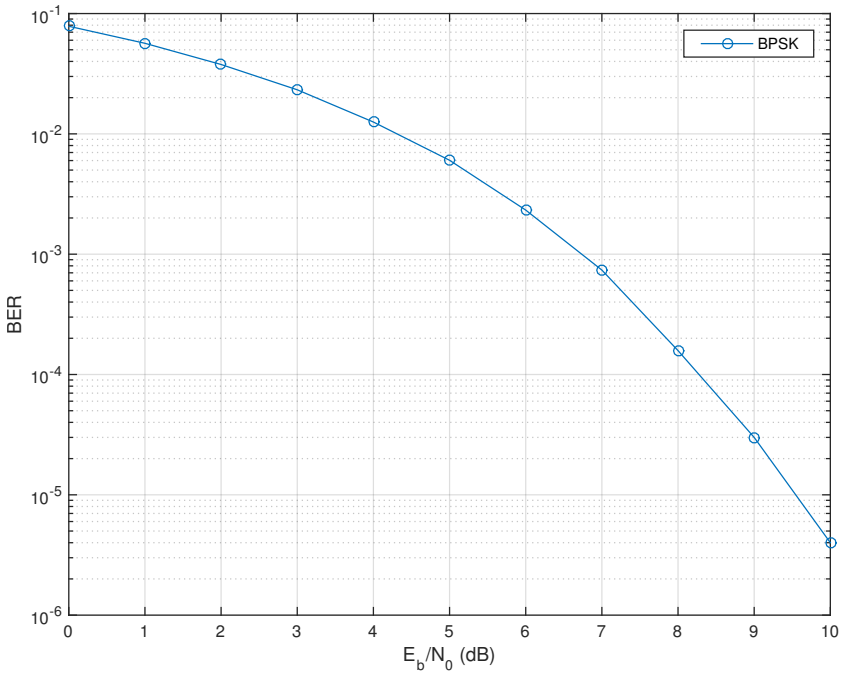

Figura 2: BER vs E_b/N_0(dB) para un sistema de transmisión BPSK en un canal AWGN.

3. Defensa de la práctica

El plazo estimado para la realización de la práctica será hasta el 13 de Marzo. En la sesión de prácticas de esa semana, se le pedirá a cada alumno que muestre el código desarrollado, lo ejecute y muestre las curvas de BER obtenidas (tanto con Gray-Mapping como sin él). Finalmente, se le harán una serie de preguntas para verificar que realmente ha realizado y entendido la práctica.

4. Ejemplo de transmisión 4-PAM

Ejemplo de transmisión de 6 bits generados aleatoriamente utilizando una 4-PAM para una $E_b/N_0 = 0$ dB. Se elige $\sqrt{E_p} = 1$ para que los símbolos de la modulación sean valores enteros. (Solo se muestran los valores intermedios; las instrucciones necesarias para obtenerlos están omitidas.)

```
modulacion =

  -3  -1   1   3

energia_media_simbolo =

   5.0000

energia_media_bit =

   2.5000

bits =

   0   1   0   0   1   1

bits_reshaped =

   0   0   1
   1   0   1

indice_simbolos =

   1   0   3

simbolos =

  -1  -3   3

ebn0db =

   0

ebn0 =

   1

n0 =
```

```
   2.5000

ruido =

   1.0696  -1.0941  -0.5468

recibido =

   0.069642  -4.094060   2.453155

demodulado =

   1  -3   3

indice_demodulado =

   2   0   3

bits_recibidos =

   1   0   0   0   1   1

errores =

   2

ber =

   0.3333
```