IE-0527-Ingeniería de Comunicaciones. Componente: Modulación banda base

[Líder] Alberto Mata, B74558[1], **[Comunicador]**Rubén Venegas Zúñiga, B78278 [1],
[Experto] Ignacio Mora Calderón, B64657 [1] Marco Chacon Soto, B61868, [1]
[1]Escuela de Ingeniería Eléctrica, UCR.

I. Planteamiento

Se parte de un archivo de texto llamado info.txt, donde este posee en el contenido la letra de la canción "This Charming Man"de la banda *The Smits*. Como se ve:

```
Punctured bicycle on a hillside desolate
Will Nature make a man of me yet?
When in this charming car
This charming man
Why pamper life's complexity
When the leather runs smooth on the passenger seat?
I would go out tonight but I haven't got a stitch to
    wear
This man said, "It's gruesome that someone so
    handsome should care"

Ah, a jumped-up pantry boy
Who never knew his place
He said, "Return the ring"
He knows so much about these things
He knows so much about these things

I would go out tonight but I haven't got a stitch to
    wear
This man said, "It's gruesome that someone so
    handsome should care"
Naaa nana nana nana, this charming man
Naaa nana nana nana, this charming man

A jumped-up pantry boy
Who never knew his place
He said, "Return the ring"
He knows so much about these things
He knows so much about these things
He knows so much about these things
```

I-A. *Modulación banda base*

De la codificación de canal, llegan los bits en paquetes de 6 unidades. Lo primero que se hace en la modulación consiste en empaquetarlos pero ahora en 2 unidades. Esto debido a que la modulación va a depender únicamente de 4 símbolos. Se empaquetan de la manera:

```
bits = []
for block in u0:
    for i in range(int(len(block)/2)):
        bits.append(block[2*i:(2*i)+2])
```

Seguidamente se definen 50 puntos y un tiempo de bit de $2x10^{-6}$. Mediante linspace se dividen en 50 puntos empezando desde 0 hasta terminar en el tiempo de bit. Esto se hace de la forma:

```
pts = 50

T_bit = 2E-6

tp = np.linspace(0, T_bit, pts)
```

Seguidamente se define una función de pulso. Esta función lo que hace es crear un array del mismo tamaño del array ingresado, solo que rellenado únicamente con unos. Por ende se crea la función y se manda el array tp posteriormente creado. Esto se ve como:

```
def pulse(T):
    pulse = np.array([])
    for element in T:
        pulse = np.append(pulse,[1])
    return pulse


pulse = pulse(tp)
```

Se utiliza el linspace nuevamente para crear un array que empiece desde 0 y termine en el tiempo de bit multiplicado por la cantidad de paquetes de 2 bits existentes, y estos se dividen en 50 puntos multiplicados de igual forma con la cantidad de paquetes de 2 bits existentes.Este array se guarda en la variable t.De este array t, únicamente interesa su forma. Por ende se utiliza la función de numpy zero y .shape para tener la forma de t pero rellenada solo con ceros. Este nuevo array se guarda en la variable senal, y se presenta en el código como:

```
t = np.linspace(0, len(bits)*T_bit, len(bits)*pts)

senal = np.zeros(t.shape)
```

La convención de simbolos, respecto a los bits de entrada se ve en la tabla 1.

Tabla I: Convención de símbolos

$b_1...b_{1+k}$	a_n
00	-10
01	-5
10	5
11	10

I-B. *Señal modulada PAM sin ruido:*

Para generar la señal modulada por amplitud de pulsos sin ruido, se analizan los bits que están empaquetados en pares.

De acuerdo al valor de estos bits, se les selecciona su respectivo símbolo. Esto se logra multiplicando el valor del símbolo deseado con el array de pulse realizado anteriormente. Esto ya que este array contiene la forma deseada y está rellena de unos solamente. En el código se realiza de la manera:

```

# Senal modulada PAM
for p,b in enumerate(bits):
    if (b[0]%2  == 0 and b[1]%2 == 0): #00
        senal[p*pts:(p+1)*pts] = -10*pulse
    elif (b[0]%2  == 0 and b[1]%2 != 0): #01
        senal[p*pts:(p+1)*pts] = -5*pulse
    elif (b[0]%2  != 0 and b[1]%2 == 0): #10
        senal[p*pts:(p+1)*pts] = 5*pulse
    elif (b[0]%2  != 0 and b[1] != 0): #11
        senal[p*pts:(p+1)*pts] = 10*pulse
senal = senal[int(pts/2):]
t = t[:-int(pts/2)]
```

Utilizando de pyplot, se logra gráficar los primeros bits modulados.La gráfica se aprecia en la figura 1. Esto se realiza con el código:

```
# Visualizacion de los primeros bits modulados
pb = 10
plt.plot(t[0:pb*pts],senal[0:pb*pts])
plt.title('Visualizacion de los primeros 10 bloques de bits modulados sin ruido')
plt.ylabel('Amplitud')
plt.xlabel('tiempo(s)')
plt.savefig('senal2.png')
plt.close()
```

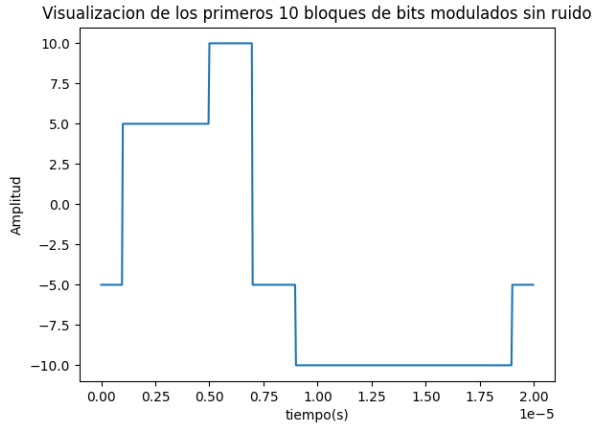

Figura 1: Gráfica de primeros bits modulados sin ruido

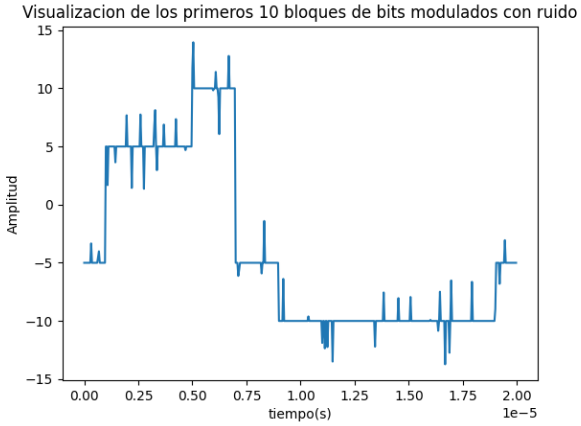

Figura 2: Gráfica de primeros bits modulados con ruido

I-C. Señal modulada PAM con ruido:

Para generar la señal modulada por amplitud de pulsos con ruido, se genera el mismo procedimiento, pero con la adición del error. Se genera un error del 10 %, donde la señal es capaz de variar en un 40 %. La gráfica para los primeros bits modulados con ruido se muestra en la figura 2. El código para esta parte corresponde ha:

```
# Senal modulada PAM
for p,b in enumerate(bits):
    p_e = 0.10  # porcentaje de error
    if (b[0]%2  == 0 and b[1]%2 == 0): #00
        senal[p*pts:(p+1)*pts] = -10*pulse
    elif (b[0]%2  == 0 and b[1]%2 != 0): #01
        senal[p*pts:(p+1)*pts] = -5*pulse
    elif (b[0]%2  != 0 and b[1]%2 == 0): #10
        senal[p*pts:(p+1)*pts] = 5*pulse
    elif (b[0]%2  != 0 and b[1] != 0): #11
        senal[p*pts:(p+1)*pts] = 10*pulse
    #agregar el ruido
    for i in range(p*pts,((p+1)*pts)):
        prob = np.random.rand(1)
        if (prob <= p_e):
            decimal=np.random.rand(1)
            if(b[0]%2  == 0 and b[1]%2 == 0):
                entero=np.random.randint(6,14)
                numeroerror=-entero-decimal
                senal[i]=numeroerror
            if(b[0]%2  == 0 and b[1]%2 != 0):
                entero=np.random.randint(1,9)
                numeroerror=-entero-decimal
                senal[i]=numeroerror
            if(b[0]%2  != 0 and b[1]%2 == 0):
                entero=np.random.randint(1,9)
                numeroerror=entero+decimal
                senal[i]=numeroerror
            if(b[0]%2  != 0 and b[1] != 0):
                entero=np.random.randint(6,14)
                numeroerror=entero+decimal
                senal[i]=numeroerror


senal = senal[int(pts/2):]
t = t[:-int(pts/2)]
```

I-D. Demodulación:

Se empieza la demodulación tomando el array senal y reduciendo la cantidad de muestras con las que se contaba, a 1 y colocándolo en el array muestreo. Esto mediante:

```
#Muestreo
muestreo = []
for i in range(len(bits)):
    muestreo.append(senal[i*len(tp)])
```

Se prosigue a transformar de símbolos, a los bits correspondientes en paquetes pares. Por ende si el símbolo es menor

a -7.5 se considera que posee al simbolo -10 y se transcribe a sus bits respectivos de 00. Si está entre el rango de mayor a -7.5 pero menor a 0 se considera -5 y sus respectivos bits equivalen a 01. Si está entre el rango de mayor a 0 pero menor a 7.5 se toma como 5 y sus respectivos bits corresponden a 10. Por último si es mayor a 7.5 se considera como 10 y sus bits equivalen a 11. Esta cadena de bits en serie se guarda en el array des. Como se puede observar en:

```
#Decision y simbolo a bits
des = []
for valor in muestreo:
    if(valor < -7.5):
        zz = [0,0]
        des.append(zz)
    elif(valor > -7.5 and valor < 0):
        zo = [0,1]
        des.append(zo)
    elif(valor > 0 and valor < 7.5):
        oz = [1,0]
        des.append(oz)
    elif(valor > 7.5):
        oo = [1,1]
        des.append(oo)
```

Por último se envían al decodificador de canal esos bits del array de des, pero empaquetados en grupos de 6, no de pares. Esto se realiza mediante el código:

```
#bloques de 6 bits nuevamente
u0_d = []
for i in range(len(u0)):
    aux = des[i*3:(i*3)+3]
    eight = np.array([])
    for bits in aux:
        for i in range(len(bits)):
            eight = np.append(eight,int(bits[i]))
    u0_d.append(eight)
```

II. Resultados

Al ejecutar el comando:

```
python3 p4sinruido.py
```

El código sin ruido se ejecuta y se imprimirá el mensaje que se observa en la figura 3.

```
-master/Proyecto$ python3 codificacion.py
Transmitiendo ...
Hecho!
```

Figura 3: Resultado de la simulación

Para el caso sin ruido se tiene que el archivo transmision.txt, cuenta en su contenido con:

```
Punctured bicycle on a hillside desolate
Will Nature make a man of me yet?
When in this charming car
This charming man
Why pamper life's complexity
When the leather runs smooth on the passenger seat?
I would go out tonight but I haven't got a stitch to
     wear
This man said, "It's gruesome that someone so
    handsome should care"
Ah, a jumped-up pantry boy
Who never knew his place
He said, "Return the ring"
He knows so much about these things
He knows so much about these things

I would go out tonight but I haven't got a stitch to
     wear
This man said, "It's gruesome that someone so
    handsome should care"
Naaa nana nana nana, this charming man
Naaa nana nana nana, this charming man

A jumped-up pantry boy
Who never knew his place
He said, "Return the ring"
He knows so much about these things
He knows so much about these things
He knows so much about these things
```

Al ejecutar el comando:

```
python3 p5connruido.py
```

El código con ruido se ejecuta y se imprimirá nuevamente el mensaje que se observa en la figura 3.Para el caso con ruido se tiene que el archivo transmision.txt, cuenta en su contenido con:

```
Punctured bic{cle on a lillside desolate
Will NatuRe make a man of me yet?When in this
    blarming$car
This charming man
Why pamper life's complexitq
When the l e ther runs smooth on the passener
    seat?I would go out tonight but I haren'd got a
     statch to wear
This man scidl It's gr5esome that someone 3o
    handsome shoulf cAre"

Ah, a jemped- p  pantRy boy
 W o n%ver knew his xlace
He said, "Ret5rn t h e ring "
He knows so }5ch about thmse things
Ie knows so much about thmSe thing{
I would0go /ut to~ight but I haven't goT a stitch to
     wear
This0man sa) d  "It's gruesom that skm%one so
    handsoe qhould care"
 N aa nana n!na nana, this charming man
Jaaa nana na.c naoa, 4jis$#harming man

A jumped-up paftry rky
Who nevar$kndw his place
Hm said, "Return the ring"
He know{ so'mukh about"these things
He knows s/ much about the3e things
He knows co mubh afout these things
```

III. Comentarios

Se muestra una transmisión de datos con modulación y demodulación banda base en la cuál los datos transmitidos se muestran en el archivo transmisión.txt. Para el caso donde no se posee ruido, se cuenta con toda la información correcta transmitida desde el archivo info.txt, lo que comprueba el óptimo funcionamiento de la modulación y demodulación banda base.Sin embargo en el caso con ruido se cuenta con algunos datos incorrectos los cuáles son producto del error generado por la modulación, donde este simula así el ruido

generado por el medio de transmisión. Dichos errores no fueron arreglados en su corrección de errores respectiva, esto provoca que algunos símbolos recibidos no sean los mismos que fueron enviados. Sin embargo se observa que la cantidad de errores con respecto a la codificación de canal disminuye una vez que se utiliza la modulación.