

**Instituto Politécnico Nacional**

*Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas*

Teoría de la información

**Practica 3**

Capacidad de canal con AWGN y codificación Huffman

**Profesor**

Jorge Rojas Beltrán

**Grupo**

2TM4

**Alumno**

Alvarado Balbuena Jorge Anselmo

2019/03/25

**Ejercicio 1**

**Código**

w=[1:5:20, 25:20:100, 130:50:300, 400:100:1000, 1250:250:5000, ...

5500:500:10000]; % Ancho de banda

pn0\_db=-20:1:30; % SNR en decibeles

pn0=10.^(pn0\_db/10); % SNR

c=zeros(length(w), length(pn0\_db));

% Se calcula la capacidad de canal

for i=1:45

for j=1:51

c(i,j)=w(i)\*log2(1+pn0(j)/w(i));

end

end

k=[.9 .8 .5 .6]; % Parametros adicionales

s=[-70 35]; % para la funcion surfl

surfl(w, pn0\_db, c', s, k);

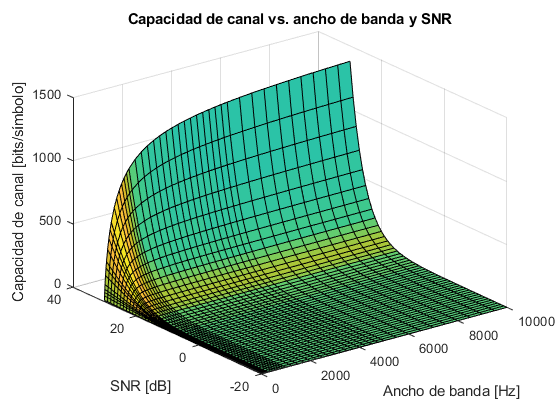
title('Capacidad de canal vs. ancho de banda y SNR');

xlabel('Ancho de banda');

ylabel('SNR');

zlabel('Capacidad de canal');

**Gráfica**

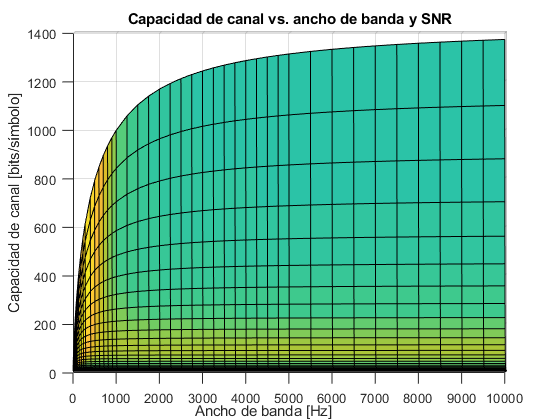


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

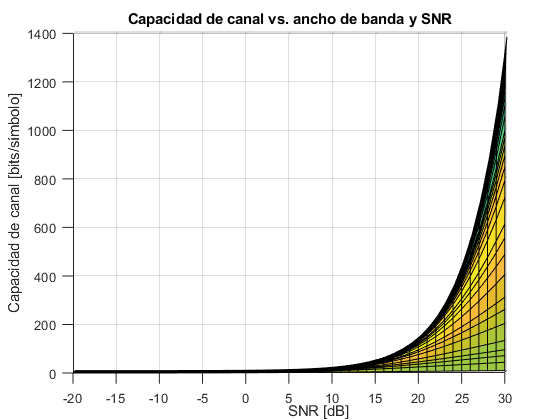


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Ejercicio 2**

**Código**

simb=[1 2 3 4];

p=[.1 .2 .3 .4];

dicc=huffmandict(simb,p);

dicc{1,:};

dicc{2,:};

dicc{3,:};

dicc{4,:};

cadena\_fte=[4 2 1 1 3];

% Codificacion

cadena\_cod=huffmanenco(cadena\_fte, dicc);

% Decodificacion

cadena\_estudio=[1 0 1 0 0 0 0 1 1];

cadena\_codi=huffmandeco(cadena\_estudio,dicc);

**Resultados**

Diccionario resultante.

|  |  |
| --- | --- |
| Símbolo | Código |
| 1 | [0,0,1] |
| 2 | [0,0,0] |
| 3 | [0,1] |
| 4 | [1] |

Cadena fuente: 4 2 1 1 1 3.

Cadena codificada: 1 0 0 0 0 0 1 0 0 1 0 1.

Cadena codificada: 1 0 1 0 0 0 0 1 1.

Cadena decodificada: 4 3 2 3 4.

**Ejercicio 3**

**Código**

secuencia=[4 5 6 5 7 7 7 8 9 8 8 7 6 7 5 4 7 6 5 4 5 6 7 6 7 6 4 5 ...

8 7 7 6 7 6 7 7 6 5 9 8 4 7 6 7 7 6 7 8 6 5];

simbolos=[4 5 6 7 8 9];

propbabilidades=zeros(1, length(simbolos)); % Vector de probabilidades

for i=4:9

prob=histc(secuencia, i)/50; % Probabilidad por simbolo

propbabilidades(i-3)=prob; % Se agrega la probabilidad al vector

end

% Diccionario

dicc=huffmandict(simbolos, propbabilidades);

% Codificacion

cadena\_estudio=[4 5 5 7 6 8 7 7 9 6];

cadena\_cod=huffmanenco(cadena\_estudio,dicc);

% Decodificacion

cad=[1 0 1 1 0 0 1 0 0 0 0 1 0 1];

cadena\_deco=huffmandeco(cad, dicc);

**Resultados**

Diccionario resultante.

|  |  |  |
| --- | --- | --- |
| Símbolo | Código | Probabilidad |
| 4 | [0,1,0,0] | 0.1000 |
| 5 | [1,1] | 0.1600 |
| 6 | [1,0] | 0.2400 |
| 7 | [0,0] | 0.3400 |
| 8 | [0,1,1] | 0.1200 |
| 9 | [0,1,0,1] | 0.0400 |

Cadena por codificar: 4 5 5 7 6 8 7 7 9 6.

Cadena codificada: 0 1 0 0 1 1 1 1 0 0 1 0 0 1 1 0 0 0 0 0 1 0 1 1 0.

Cadena por decodificar: 1 0 1 1 0 0 1 0 0 0 0 1 0 1.

Cadena decodificada: 6 5 7 6 7 9.

**Ejercicio 4**

**Código**

cadena=['j' 'o' 'r' 'g' 'e' ' ' ...

'a' 'n' 's' 'e' 'l' 'm' 'o' ' ' ...

'a' 'l' 'v' 'a' 'r' 'a' 'd' 'o' ' ' ...

'b' 'a' 'l' 'b' 'u' 'e' 'n' 'a' ' ' ...

'p' 'a' 't' 'i' 'l' 'l' 'a'];

simbolos=unique(cadena); % Obtenemos los símbolos fuente de la cadena

% Calculamos la probabilidades y agregamos al vector

propbabilidades=zeros(1, length(simbolos));

j=1;

for i=simbolos

prob=histc(cadena, i)/length(cadena);

propbabilidades(j)=prob;

j=j+1;

end

% Mapeamos los símbolos al código ascii

mapSimbolos=double(simbolos);

% Obtenemos el diccionario

dicc=huffmandict(mapSimbolos,propbabilidades);

% Codificacio de la palabra

palabra=['e' 's' 'p' 'o' 'n' 't' 'a' 'n' 'e' 'i' 'd' 'a' 'd'];

mapPalabra=double(palabra);

cadena\_cod=huffmanenco(mapPalabra,dicc);

**Resultados**

Diccionario resultante.

|  |  |  |  |
| --- | --- | --- | --- |
| Símbolo | Mapeo | Código | Probabilidad |
| esp | 32 | [0,1,1] | 0.1026 |
| a | 97 | [1,0] | 0.2051 |
| b | 98 | [1,1,0,0] | 0.0513 |
| d | 100 | [1,1,0,1,1] | 0.0256 |
| e | 101 | [0,0,1,1] | 0.0769 |
| g | 103 | [1,1,0,1,0] | 0.0256 |
| i | 105 | [0,0,0,1,0,1] | 0.0256 |
| j | 106 | [0,0,0,1,0,0] | 0.0256 |
| l | 108 | [0,1,0] | 0.1282 |
| m | 109 | [0,0,0,1,1,1] | 0.0256 |
| n | 110 | [1,1,1,1] | 0.0513 |
| o | 111 | [0,0,1,0] | 0.0769 |
| p | 112 | [0,0,0,1,1,0] | 0.0256 |
| r | 114 | [1,1,1,0] | 0.0513 |
| s | 115 | [0,0,0,0,0,1] | 0.0256 |
| t | 116 | [0,0,0,0,0,0] | 0.0256 |
| u | 117 | [0,0,0,0,1,1] | 0.0256 |
| v | 118 | [0,0,0,0,1,0] | 0.0256 |

Palabra por codificar: 'e' 's' 'p' 'o' 'n' 't' 'a' 'n' 'e' 'i' 'd' 'a' 'd'.

Palabra codificada: 0011000001000110001011110000001011110011000101110111011011.

**Conclusiones**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_