UNIVERSIDAD POLITÉCNICA DE VALENCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

Máster Universitario en Ingeniería de Telecomunicación

Práctica 1
Control de errores

PSCA

Autor:
Andrés Ruz Nieto

VALENCIA, 2021

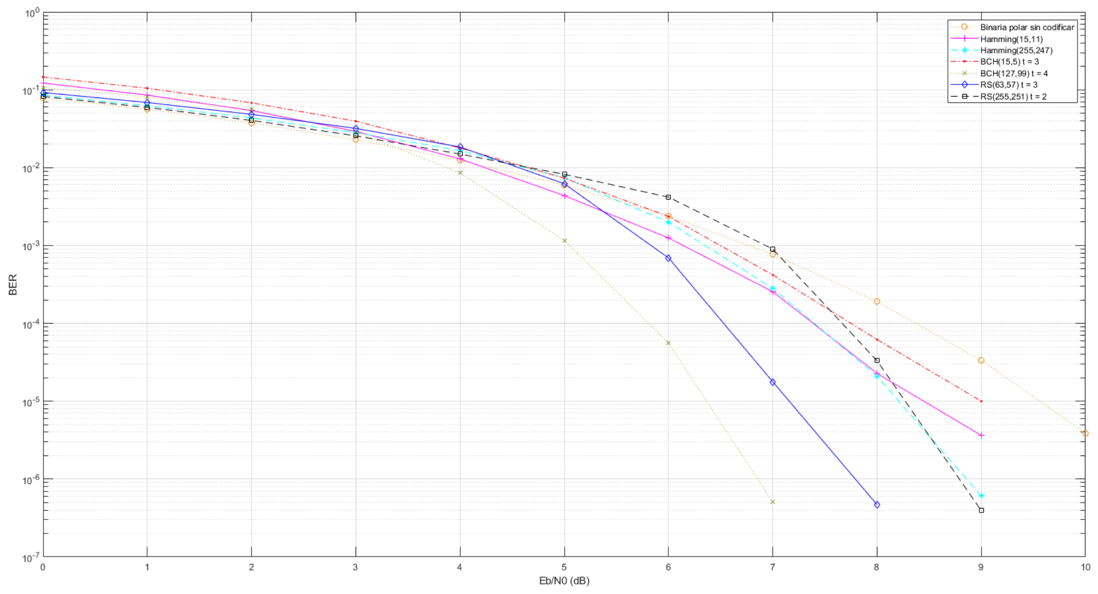

Figura 1: Probabilidad de error de bit

Código	m	n	k	R	t	Gc
Hamming(15,11)	4	15	11	0,73333333	-	$8{,}372 - 7{,}386 = 0{,}986$
Hamming(255,247)	8	255	247	0,96862745	-	$8{,}372 - 7{,}398 = 0{,}974$
BCH(15,5)	4	15	5	0,33333333	3	$8{,}372 - 7{,}749 = 0{,}623$
BCH(127,99)	7	127	99	0,77952756	4	$8{,}372 - 5{,}808 = 2{,}564$
RS(63,57)	6	63	57	0,9047619	3	$8{,}372 - 6{,}528 = 1{,}844$
RS(255,251)	8	255	251	0,98431373	2	$8{,}372 - 7{,}665 = 0{,}707$

Tabla 1: Códigos y sus parámetros de prueba

```
clc,clear

Eb_div_N0_dB = 0 : 1 : 10;
Eb_div_N0 = 10.^( Eb_div_N0_dB / 10 );
p_sin_cod = Q( sqrt( 2 * Eb_div_N0 ) );
nMensajes = 100000;

BER_hamming1511 = zeros(size(Eb_div_N0_dB));
BER_hamming255247 = zeros(size(Eb_div_N0_dB));
BER_bch155 = zeros(size(Eb_div_N0_dB));
BER_bch12799 = zeros(size(Eb_div_N0_dB));
BER_rs6357 = zeros(size(Eb_div_N0_dB));
BER_rs255251 = zeros(size(Eb_div_N0_dB));


%% Hamming(15,11)
disp('Simulación Hamming(15,11)')
m = 4; n = 15; k = 11; R = k/n; t = 1;
mm = randi( [ 0 1 ], k * nMensajes, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    cc = encode( mm, n, k, 'hamming/binary');
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = decode( rr, n, k, 'hamming/binary' );
    errores_hamming1511 = sum( mod( mm+mmDec ,2));
    BER_hamming1511(i) = errores_hamming1511/(k*nMensajes);
end

%% Hamming(255,247)
disp('Simulación Hamming(255,247)')
m = 8; n = 255; k = 247; R = k/n; t = 1;
mm = randi( [ 0 1 ], k * nMensajes, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    cc = encode( mm, n, k, 'hamming/binary');
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = decode( rr, n, k, 'hamming/binary' );
    errores_hamming255247 = sum( mod( mm+mmDec ,2));
    BER_hamming255247(i) = errores_hamming255247/(k*nMensajes);
end

%% BCH(15,5)
disp('Simulación BCH(15,5) t = 3')
m = 4; n = 15; k = 5; R = k/n; t = 3;
mm = randi( [ 0 1 ], k * nMensajes, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    hEnc = comm.BCHEncoder( 'CodewordLength', n, 'MessageLength', k );
    hDec = comm.BCHDecoder( 'CodewordLength', n, 'MessageLength', k );
    cc = step( hEnc, mm );
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = step( hDec, rr );
    errores_bch155 = sum( mod( mm+mmDec ,2));
    BER_bch155(i) = errores_bch155/(k*nMensajes);
end
```

```
%% BCH(127,99)
disp('Simulación BCH(127,99) t = 4')
m = 7; n = 127; k = 99; R = k/n; t = 4;
mm = randi( [ 0 1 ], k * nMensajes, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    hEnc = comm.BCHEncoder( 'CodewordLength', n, 'MessageLength', k );
    hDec = comm.BCHDecoder( 'CodewordLength', n, 'MessageLength', k );
    cc = step( hEnc, mm );
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = step( hDec, rr );
    errores_bch12799 = sum( mod( mm+mmDec ,2));
    BER_bch12799(i) = errores_bch12799/(k*nMensajes);
end

%% RS(63,57)
disp('Simulación RS(63,57) t = 3')
m = 6; n = 63; k = 57; R = k/n; t = 3;
mm = randi( [ 0 1 ], k * nMensajes * m, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    hEnc = comm.RSEncoder( 'BitInput',true,'CodewordLength', n, 'MessageLength', k );
    hDec = comm.RSDecoder( 'BitInput',true,'CodewordLength', n, 'MessageLength', k );
    cc = step( hEnc, mm );
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = step( hDec, rr );
    errores_rs6357 = sum( mod( mm+mmDec ,2));
    BER_rs6357(i) = errores_rs6357/(k*nMensajes*m);
end

%% RS(255,251)
disp('Simulación RS(255,251) t = 2')
m = 8; n = 255; k = 251; R = k/n; t = 2;
mm = randi( [ 0 1 ], k * nMensajes * m, 1 );
for i = 1:length(Eb_div_N0_dB)
    p = Q( sqrt( 2 * R * Eb_div_N0(i) ) );
    hEnc = comm.RSEncoder( 'BitInput',true,'CodewordLength', n, 'MessageLength', k );
    hDec = comm.RSDecoder( 'BitInput',true,'CodewordLength', n, 'MessageLength', k );
    cc = step( hEnc, mm );
    ee = rand( size( cc ) ) < p;
    rr = mod( cc + ee, 2 );
    mmDec = step( hDec, rr );
    errores_rs255251 = sum( mod( mm+mmDec ,2));
    BER_rs255251(i) = errores_rs255251/(k*nMensajes*m);
end

%%
semilogy( Eb_div_N0_dB, p_sin_cod, ':o', 'Color', [0.9,0.5,0]);
hold on
grid on
semilogy( Eb_div_N0_dB, BER_hamming1511, '-+m');
semilogy( Eb_div_N0_dB, BER_hamming255247, '--*c');
semilogy( Eb_div_N0_dB, BER_bch155, '-..r');
semilogy( Eb_div_N0_dB, BER_bch12799, ':x', 'Color', [0.5,0.6,0.1] );
semilogy( Eb_div_N0_dB, BER_rs6357, '-db');
semilogy( Eb_div_N0_dB, BER_rs255251, '--sk');
xlabel( 'Eb/N0 (dB)' );
```

```
ylabel( 'BER' );
legend({'Binaria polar sin codificar','Hamming(15,11)', 'Hamming(255,247)', 'BCH(15,5) t
     = 3', 'BCH(127,99) t = 4', 'RS(63,57) t = 3', 'RS(255,251) t = 2'},'Location','
    northeast','Orientation','vertical')
hold off
```

Script 1: Script de Matlab