APENDICES

APENDICE A1. MODULACION WAVELET DE UNA ESCALA FILTRADA (WM1EBB)

```
% ----------MODULACION WAVELET DE 1 ESCALA FILTRADA---------
% -------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Declaracion de variables de entrada

K=100000;              % Numero de bits a transmitir
xc=randsrc(1,K,[-1 1]);% Mensaje digital a transmitir
tipo_wavelet='db8';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);% Filtros prototipo de analisis y
                                     % síntesis.
L=length(ha);          % Longitud de los filtros prototipo
xc=[xc zeros(1,L-2)];  % Relleno con ceros para suavizar forma de onda
N=4;                   % Numero de iteraciones
Rb=1;                  % Tasa de bit en bps
fs=(2^N)*Rb;           % Frecuencia de muestreo en Hz
Eb_No=9;        %Relacion energia de bit a densidad espectral de ruido
SNR=Eb_No-10*log10(2); % Relacion señal a ruido del canal AWGN medida
                       % sobre el primer lóbulo.

%% Generacion de la forma de onda en banda base

s1=filter(gs,1,upsample(xc,2));
s2=filter(hs,1,upsample(s1,2));
s3=filter(hs,1,upsample(s2,2));
s=filter(hs,1,upsample(s3,2));
h=rcosfir(0.125,10,4,1);        % Filtro limitador
h=h/sqrt(sum(h.*h));            % Filtro limitador normalizado
ws=filter(h,1,[s zeros(1,length(h)-1)]);  % Señal filtrada a la salida
                                          % del tx.
wc=awgn(ws,SNR-10*log10((fs/2)/(2*Rb)),'measured','db');  % Señal
     % contaminada por ruido AWGN con SNR medida sobre primer lóbulo.
wz=filter(h,1,[wc zeros(1,length(h)-1)]);% Señal filtrada a la entrada
                                         % del rx.
z=wz(length(h):length(wz)-length(h)+1);% Truncando la señal a su rango
                                       % significativo.

%% Deteccion de los datos transmitidos

e1=filter(ha,1,[z zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
d=filter(ga,1,[e3 zeros(1,L-1)]);d=d(L:length(d));d=downsample(d,2);
dx=sign(d);
```

```
[Nerr,BER]=symerr(xc(1:length(xc)-L+2),dx(1:length(dx)-L+2));

%% Estimacion del numero de errores y la tasa de error
%%--------Visualizacion de resultados----------------

RES=[K Eb_No BER];
disp('Los resultados son:');
disp('      NumBits      Eb/No      BER');
disp(RES);
```

APENDICE A2. MODULACION WAVELET DE UNA ESCALA PASABANDA SSB FILTRADA (WM1ESSB)

```
% -------MODULACION WAVELET DE 1 ESCALA PASABANDA SSB FILTRADA--------
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Declaracion de variables de entrada

K=100000;              % Numero de bits a transmitir
xc=randsrc(1,K,[-1 1]);% Mensaje digital a transmitir
tipo_wavelet='db5';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);% Filtros prototipo de analisis y
                                     % síntesis.
L=length(ha);          % Longitud de los filtros prototipo
xc=[xc zeros(1,L-2)];  % Relleno con ceros para suavizar forma de onda
N=5;                   % Numero de iteraciones
Rb=1;                  % Tasa de bit en bps
fs=(2^N)*Rb;           % Frecuencia de muestreo en Hz
fc=fs/4;
Eb_No=9.8;      %Relacion energia de bit a densidad espectral de ruido
SNR=Eb_No-10*log10(2); % Relacion señal a ruido del canal AWGN medida
                       % sobre el primer lóbulo.
t=(0:length(xc)*(2^N)-1)/fs;    % Base de tiempo.

%% Generacion de la forma de onda en banda base

s1=filter(gs,1,upsample(xc,2));
s2=filter(hs,1,upsample(s1,2));
s3=filter(hs,1,upsample(s2,2));
s4=filter(hs,1,upsample(s3,2));
s=filter(hs,1,upsample(s4,2));
q=rcosfir(0.125,10,(fs/2)/(2*Rb),1);% Filtro de caida senoidal con
                                    % BW=2Rb (Primer lobulo).
b=(length(q)-1)/2;
sd=s.*cos(2*pi*fc*t);  % Señal modulada linealmente en DSB
u=-b:b;
fssb=(cos(2*pi*((fc+Rb)/(fs))*u)).*rcosfir(0.125,5,(fs/2)/(Rb),1);
sdf=filter(fssb,1,[sd zeros(1,length(fssb)-1)]);
sdf=sdf(b+1:length(sdf)-b);      % Truncado a valores significativos
```

```
%% Señal afectada por el canal

zd=awgn(sdf,SNR-10*log10((fs/2)/(2*Rb)),'measured','db');  % Señal DSB
            % contaminada por ruido AWGN medido en su ancho de banda.

%% Deteccion de los datos transmitidos

z=zd.*cos(2*pi*fc*t);% Demodulador balanceado sincronizado idealmente
                    % con el Tx.
zf=filter(q,1,[z zeros(1,length(q)-1)]);
zf=zf(b+1:length(zf)-b);     % Señal truncada a su rango significativo
e1=filter(ha,1,[zf zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
e4=filter(ha,1,[e3 zeros(1,L-1)]);
e4=e4(L:length(e4));e4=downsample(e4,2);
d=filter(ga,1,[e4 zeros(1,L-1)]);d=d(L:length(d));d=downsample(d,2);
dx=sign(d);
[Nerr,BER]=symerr(xc(1:length(dx)-L+2),dx(1:length(dx)-L+2));

%% Estimacion del numero de errores y la tasa de error
%% --------Visualizacion de resultados-----------------

RES=[K Eb_No Nerr BER];
disp('Los resultados son, utilizando la wavelet:'),disp(tipo_wavelet);
disp('      NumBits        Eb/No      NumErr     BER');
disp(RES);
BER_teorica=0.5*erfc(sqrt(0.5*10^(Eb_No/10)));
disp('La BER teorica es:');
disp(BER_teorica);
```

APENDICE A3. MODULACION WAVELET DE UNA ESCALA PASO BANDA SSB M-ARIA FILTRADA (M_WM1ESSB)

```
% ---MODULACION WAVELET DE 1 ESCALA PASO BANDA SSB M-ARIA
FILTRADA----
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Declaracion de variables de entrada

M=8;                  % Tamaño del alfabeto fuente
n=log2(M);            % Numero de bits mapeados en cada simbolo
K=100000;             % Numero de bits a transmitir
x=randsrc(1,K,[-(M-1):2:(M-1)]);   % Mensaje discreto a transmitir
tipo_wavelet='Haar';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);% Filtros prototipo de analisis y
```

```
                                % síntesis.
L=length(ha);          % Longitud de los filtros prototipo
x=[x zeros(1,L-2)];    % Relleno con ceros para suavizar forma de onda
N=5;                   % Numero de iteraciones
Rs=1;                  % Tasa de simbolos en bauds/sg
fs=(2^N)*Rs;           % Frecuencia de muestreo en Hz
fc=fs/4;
Eb_No=18.28;   % Relacion energia de bit a densidad espectral de ruido
Es_No=Eb_No+10*log10(n);    % Relacion energia promedio de simbolo a
                        % densidad espectral de ruido.
SNR=Es_No-10*log10(2);% Relacion señal a ruido del canal AWGN medida
                    % sobre el primer lóbulo.
t=(0:length(x)*(2^N)-1)/fs;      % Base de tiempo


%% Generacion de la forma de onda en banda base

s1=filter(gs,1,upsample(x,2));
s2=filter(hs,1,upsample(s1,2));
s3=filter(hs,1,upsample(s2,2));
s4=filter(hs,1,upsample(s3,2));
s=filter(hs,1,upsample(s4,2));
c=sqrt(2)*cos(2*pi*fc*t);    % Onda portadora
sd=s.*c;                     % Señal modulada linealmente en DSB
q=rcosfir(0.125,20,(fs/2)/(2*Rs),1);    % Filtro de demodulacion
b=(length(q)-1)/2;
u=-b:b;
qm=rcosfir(0.125,10,(fs/2)/(Rs),1);
fssb=(cos(2*pi*((fc+Rs)/(fs))*u)).*qm;
sdf=filter(fssb,1,[sd zeros(1,length(fssb)-1)]);
sdf=sdf(b+1:length(sdf)-b);    % Truncado a valores significativos

%% Señal afectada por el canal

z=awgn(sdf,SNR-10*log10((fs/2)/(2*Rs)),'measured','db');% Señal M-WM
          % contaminada por ruido AWGN medido en su ancho de banda

%% Etapa de prefiltraje en Rx

zk=filter(fssb,1,[z zeros(1,length(fssb)-1)]);
zk=zk(b+1:length(zk)-b);     % Truncado a valores significativos

%% Extraccion de la onda en banda base

zb=zk.*c;                    % Demodulador balanceado
zbb=filter(q,1,[zb zeros(1,length(q)-1)]);
zbb=zbb(b+1:length(zbb)-b);  % Señal truncada a su rango significativo
zbb=(sqrt(sum(s.*s)/sum(zbb.*zbb)))*zbb;

%% Deteccion de los datos transmitidos

e1=filter(ha,1,[zbb zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
```

```
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
e4=filter(ha,1,[e3 zeros(1,L-1)]);
e4=e4(L:length(e4));e4=downsample(e4,2);
d=filter(ga,1,[e4 zeros(1,L-1)]);d=d(L:length(d));d=downsample(d,2);
dx=zeros(1,length(d));

for (m=-0.5*M+1:0.5*M-1)
   dx=dx+sign(d-2*m);
end

[Nerr_sym,SER]=symerr(x(1:length(dx)-L+2),dx(1:length(dx)-L+2));

%% Estimacion del numero de errores y la tasa de error de símbolo

BER=SER/n;
u=-M-1+0.1:0.1:M+1;
y=zeros(1,length(u));

for (m=-0.5*M+1:0.5*M-1)
   y=y+sign(u-2*m);
end

figure(3)

stairs(u,y,'r'),grid,title('Esquema de decision'),xlabel('Entrada'),ylabel('Salida'),axis([-M-1
M+1 -M M])

%% --------Visualizacion de resultados-----------------

RES=[K Eb_No M Nerr_sym SER BER];
disp('Los resultados son, utilizando la wavelet:'),disp(tipo_wavelet)
disp('    NumSym     Eb/No     M     NumErrSym     SER     BER');
disp(RES);

%% Resultados teoricos

k=0:n-1;
a=sum(2.^(2*k));% Factor de reduccion de relacion Es/No para M niveles
p=(10^(0.1*Es_No))/a;
SER_teorica=(1-(1/M))*erfc(sqrt(p));
BER_teorica=SER_teorica/n;
disp('y los resultados teoricos son:');
disp('   SER_teorica    BER_teorica');
disp([SER_teorica BER_teorica]);
```

APENDICE A4. MODULACION WAVELET DE UNA ESCALA DSB M-ARIA EN CUADRATURA (MQ_WM1EDSB)

```
% ------MODULACION WAVELET DE 1 ESCALA DSB M-ARIA EN CUADRATURA-----
--
% ----------------------------------------------------------------

%% Acondicionamiento inicial
```

```
clear all
format short g

%% Declaracion de variables de entrada

M=64;                 % Tamaño del alfabeto fuente
n=log2(M);            % Numero de bits mapeados en cada simbolo
N=5;                  % Numero de iteraciones
Rs=1;                 % Tasa de simbolos en bauds/sg
fs=(2^N)*Rs;          % Frecuencia de muestreo en Hz
fc=fs/4;              % Frecuencia portadora
Eb_No=16.68;   % Relacion energia de bit a densidad espectral de ruido
Es_No=Eb_No+10*log10(n);      % Relacion energia promedio de simbolo a
                              % densidad espectral de ruido.
SNR=Es_No-10*log10(4);  % Relacion señal a ruido del canal AWGN medida
                        % sobre el primer lóbulo.
tipo_wavelet='coif5';
[ha,ga,hs,gs]=wfilters(tipo_wavelet); % Filtros prototipo de analisis
                                      % y síntesis.
L=length(ha);          % Longitud de los filtros prototipo
K=100000;              % Numero de bits a transmitir
a=randsrc(1,K,[-(sqrt(M)-1):2:(sqrt(M)-1)]);        %Simbolos canal I
b=randsrc(1,K,[-(sqrt(M)-1):2:(sqrt(M)-1)]);        %Simbolos canal Q
c=a+b*i;               %Simbolo complejo transmitido

%% Transmisor

a=[a zeros(1,L-2)];    % Relleno con ceros para suavizar forma de onda
b=[b zeros(1,L-2)];    % Relleno con ceros para suavizar forma de onda
t=(0:length(a)*(2^N)-1)/fs;            % Base de tiempo

%% Generacion de señales en banda base

a1=filter(gs,1,upsample(a,2));
a2=filter(hs,1,upsample(a1,2));
a3=filter(hs,1,upsample(a2,2));
a4=filter(hs,1,upsample(a3,2));
at=filter(hs,1,upsample(a4,2));        % Señal banda base canal I
b1=filter(gs,1,upsample(b,2));
b2=filter(hs,1,upsample(b1,2));
b3=filter(hs,1,upsample(b2,2));
b4=filter(hs,1,upsample(b3,2));
bt=filter(hs,1,upsample(b4,2));        % Señal banda base canal Q
s=at.*cos(2*pi*fc*t)+bt.*sin(2*pi*fc*t);% Señal modulada wavelet en
                                        % cuadratura.

%% Canal AWGN

z=awgn(s,SNR-10*log10((fs/2)/(4*Rs)),'measured');    % Señal MQ-WM
            % contaminada por ruido AWGN medido en su ancho de banda

%% Receptor

q=rcosfir(0.125,10,(fs/2)/(2*Rs),1);    % Filtro de demodulacion
```

```
T=(length(q)-1)/2;

%% Mezclador

za=z.*cos(2*pi*fc*t);
zb=z.*sin(2*pi*fc*t);
ar=filter(q,1,[za zeros(1,length(q)-1)]);
ar=ar(T+1:length(ar)-T);     % Señal truncada a su rango significativo
br=filter(q,1,[zb zeros(1,length(q)-1)]);
br=br(T+1:length(br)-T);% Señal truncada a su rango significativo

%% Transformada wavelet discreta

ea1=filter(ha,1,[ar zeros(1,L-1)]);
ea1=ea1(L:length(ea1));ea1=downsample(ea1,2);
ea2=filter(ha,1,[ea1 zeros(1,L-1)]);
ea2=ea2(L:length(ea2));ea2=downsample(ea2,2);
ea3=filter(ha,1,[ea2 zeros(1,L-1)]);
ea3=ea3(L:length(ea3));ea3=downsample(ea3,2);
ea4=filter(ha,1,[ea3 zeros(1,L-1)]);
ea4=ea4(L:length(ea4));ea4=downsample(ea4,2);
da=filter(ga,1,[ea4 zeros(1,L-1)]);
da=da(L:length(da));da=downsample(da,2);      % Canal I
da=sqrt(sum(a.*a)/sum(da.*da))*da;
da=da(1:K);
eb1=filter(ha,1,[br zeros(1,L-1)]);
eb1=eb1(L:length(eb1));eb1=downsample(eb1,2);
eb2=filter(ha,1,[eb1 zeros(1,L-1)]);
eb2=eb2(L:length(eb2));eb2=downsample(eb2,2);
eb3=filter(ha,1,[eb2 zeros(1,L-1)]);
eb3=eb3(L:length(eb3));eb3=downsample(eb3,2);
eb4=filter(ha,1,[eb3 zeros(1,L-1)]);
eb4=eb4(L:length(eb4));eb4=downsample(eb4,2);
db=filter(ga,1,[eb4 zeros(1,L-1)]);
db=db(L:length(db));db=downsample(db,2);       % Canal Q
db=sqrt(sum(b.*b)/sum(db.*db))*db;
db=db(1:K);
d=da+db*i;

%% Decisor

aD=zeros(1,length(da));
bD=zeros(1,length(db));

for (m=-0.5*sqrt(M)+1:0.5*sqrt(M)-1)
   aD=aD+sign(da-2*m);
   bD=bD+sign(db-2*m);
end

cD=aD+bD*i;            %Simbolos detectados

%% Estimacion del comportamiento

[Nerr_sym,SER]=symerr(c,cD);
```

```
%% Estimacion del numero de errores y la tasa de error de símbolo

BER=SER/n;

%% Dibujos (4)

u=-sqrt(M)-1+0.1:0.1:sqrt(M)+1;
y=zeros(1,length(u));
for (m=-0.5*sqrt(M)+1:0.5*sqrt(M)-1)
   y=y+sign(u-2*m);
end

figure(8)

subplot(221),stairs(u,y,'r'),grid,title('Esquema de decision canal
I'),xlabel('Entrada'),ylabel('Salida'),axis([-sqrt(M)-1 sqrt(M)+1 -sqrt(M) sqrt(M)])

subplot(222),stairs(u,y,'r'),grid,title('Esquema de decision canal
Q'),xlabel('Entrada'),ylabel('Salida'),axis([-sqrt(M)-1 sqrt(M)+1 -sqrt(M) sqrt(M)])

%% --------Visualizacion de resultados-----------------

RES=[K Nerr_sym SER BER];
P=[Eb_No Es_No M];

disp('Parametros del sistema MQ_WM1EDSB con la wavelet:'),disp(tipo_wavelet)

disp('       Eb/No      Es/No            M')
disp(P)
disp('Resultados de la simulacion')
disp('      NumSym      NumErrSym       SER       BER');
disp(RES);

%% Resultados teoricos

p=(10^(0.1*Es_No));
SER_teorica=2*(1-sqrt(1/M))*erfc(sqrt(1.5*p/(M-1)));
BER_teorica=SER_teorica/n;
disp('y los resultados teoricos son:');
disp('    SER_teorica    BER_teorica');
disp([SER_teorica BER_teorica]);
```

APENDICE A5. WAVELET MULTIESCALA BINARIA (N=3) N.R (WMMSBB)

```
% -------  MODULACION WAVELET MULTIESCALA BINARIA (N=3) N.R  ---------
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Parametros del sistema

tipo_wavelet='sym8';
```

```
[ha,ga,hs,gs]=wfilters(tipo_wavelet); % Filtros prototipo de analisis
                                      % y síntesis.
L=length(ha);          % Longitud de los filtros prototipo
E=3;                   % Numero de escalas de modulacion
N=7;                   % Numero de iteraciones
Rb=1;                  % Tasa de bits en bits/sg
fs=(2^N)*Rb/((2^E)-1);  % Frecuencia de muestreo en Hz
Bw=(2^E)*Rb/((2^E)-1);  % Ancho de banda del primer lobulo
Eb_No=9;       % Relacion energia de bit a densidad espectral de ruido
SNR=Eb_No+10*log10(Rb/Bw);          % Relacion señal a ruido
K=100002;              % Numero de bits a transmitir
x=randsrc(1,K,[-1 1]);  % Mensaje digital a transmitir

%% Transmisor

% S/P

b1=[x(1:K/7) zeros(1,L-2)];
b2=[x(K/7+1:3*K/7) zeros(1,2*(L-2))];
b3=[x(3*K/7+1:K) zeros(1,4*(L-2))];

% IDWT

s1=filter(gs,1,upsample(b1,2));
s2=filter(gs,1,upsample(b2,2))+filter(hs,1,upsample(s1,2));
s3=filter(gs,1,upsample(b3,2))+filter(hs,1,upsample(s2,2));
s4=filter(hs,1,upsample(s3,2));
s5=filter(hs,1,upsample(s4,2));
s6=filter(hs,1,upsample(s5,2));
s=filter(hs,1,upsample(s6,2));%Señal modulada en banda base
t=(0:length(s)-1)*length(x)/(Rb*length(s));% Base de tiempo
%% Canal AWGN

z=awgn(s,SNR-10*log10((fs/2)/Bw),'measured');

%% Receptor

% DWT

e1=filter(ha,1,[z zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
e4=filter(ha,1,[e3 zeros(1,L-1)]);
e4=e4(L:length(e4));e4=downsample(e4,2);
e5=filter(ha,1,[e4 zeros(1,L-1)]);
e5=e5(L:length(e5));e5=downsample(e5,2);
d5=filter(ga,1,[e4 zeros(1,L-1)]);
d5=d5(L:length(d5));d5=downsample(d5,2);
e6=filter(ha,1,[e5 zeros(1,L-1)]);
e6=e6(L:length(e6));e6=downsample(e6,2);
d6=filter(ga,1,[e5 zeros(1,L-1)]);
d6=d6(L:length(d6));d6=downsample(d6,2);
```

```
d7=filter(ga,1,[e6 zeros(1,L-1)]);
d7=d7(L:length(d7));d7=downsample(d7,2);

% P/S

d=[d7(1:K/7) d6(1:2*K/7) d5(1:4*K/7)];

% Decisor

dx=sign(d);

% Estimacion del comportamiento

[Nerr_bit,BER]=symerr(x,dx);

%% --------Visualizacion de resultados-----------------

RES=[K Eb_No E Nerr_bit BER];
disp('Los resultados son, utilizando la wavelet:'),disp(tipo_wavelet)
disp('    NumBit      Eb/No     NumEscalas      NumErrBit       BER');
disp(RES);
BER_teorica=0.5*erfc(sqrt(10^(0.1*Eb_No)));
disp('Resultados teoricos:');
disp(BER_teorica);
```

APENDICE A6. MODULACION WAVELET MULTIESCALA BINARIA (N=3) N.R SSB (WMMSSSB)

```
% ------ MODULACION WAVELET MULTIESCALA BINARIA (N=3) N.R SSB  -------
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g
%% Parametros del sistema

tipo_wavelet='db8';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);  % Filtros prototipo de analisis
                                       % y síntesis.
L=length(ha);          % Longitud de los filtros prototipo
E=3;                   % Numero de escalas de modulacion
N=7;                   % Numero de iteraciones
Rb=1;                  % Tasa de bits en bits/sg
fs=(2^N)*Rb/((2^E)-1);  % Frecuencia de muestreo en Hz
Bw=(2^E)*Rb/((2^E)-1);  % Ancho de banda del primer lobulo
Eb_No=9.46;    % Relacion energia de bit a densidad espectral de ruido
fc=2*Rb;               % Frecuencia portadora
SNR=Eb_No+10*log10(Rb/Bw);  % Relacion señal a ruido
K=100002;              % Numero de bits a transmitir
x=randsrc(1,K,[-1 1]);  % Mensaje digital a transmitir

%% Transmisor
```

```
% S/P

b1=[x(1:K/7) zeros(1,L-2)];
b2=[x(K/7+1:3*K/7) zeros(1,2*(L-2))];
b3=[x(3*K/7+1:K) zeros(1,4*(L-2))];

% IDWT

s1=filter(gs,1,upsample(b1,2));
s2=filter(gs,1,upsample(b2,2))+filter(hs,1,upsample(s1,2));
s3=filter(gs,1,upsample(b3,2))+filter(hs,1,upsample(s2,2));
s4=filter(hs,1,upsample(s3,2));
s5=filter(hs,1,upsample(s4,2));
s6=filter(hs,1,upsample(s5,2));
st=filter(hs,1,upsample(s6,2));     %Señal modulada en banda base
t=(0:length(st)-1)*(length(x)+N*(L-2))/(Rb*length(st)); % Base de
                                                        % tiempo

% Modulador balanceado

sd=st.*cos(2*pi*fc*t);

% Filtro eliminador de banda

q=rcosfir(0.125,20,(fs/2)/Bw,1);  % Filtro de caida senoidal con
                                  % BW=2Rb (Primer lobulo)
b=(length(q)-1)/2;
u=-b:b;
fssb=(cos(2*pi*((fc+(0.5*Bw))/(fs))*u)).*rcosfir(0.125,10,(fs/2)/(Bw/2),1);
s=filter(fssb,1,[sd zeros(1,length(fssb)-1)]);
s=s(b+1:length(s)-b);             % Truncado a valores significativos

%% Canal AWGN

z=awgn(s,SNR-10*log10((fs/2)/Bw),'measured');

%% Receptor

% Prefiltraje pasa banda

zp=filter(fssb,1,[z zeros(1,length(fssb)-1)]);
zp=zp(b+1:length(zp)-b);         % Truncado a valores significativos

% Mezclador

w=zp.*cos(2*pi*fc*t);            % Demodulador balanceado
zw=filter(q,1,[w zeros(1,length(q)-1)]);
zw=zw(b+1:length(zw)-b);

% DWT

e1=filter(ha,1,[zw zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
```

```
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
e4=filter(ha,1,[e3 zeros(1,L-1)]);
e4=e4(L:length(e4));e4=downsample(e4,2);
e5=filter(ha,1,[e4 zeros(1,L-1)]);
e5=e5(L:length(e5));e5=downsample(e5,2);
d5=filter(ga,1,[e4 zeros(1,L-1)]);
d5=d5(L:length(d5));d5=downsample(d5,2);
e6=filter(ha,1,[e5 zeros(1,L-1)]);
e6=e6(L:length(e6));e6=downsample(e6,2);
d6=filter(ga,1,[e5 zeros(1,L-1)]);
d6=d6(L:length(d6));d6=downsample(d6,2);
d7=filter(ga,1,[e6 zeros(1,L-1)]);
d7=d7(L:length(d7));d7=downsample(d7,2);

% P/S

d=[d7(1:K/7) d6(1:2*K/7) d5(1:4*K/7)];

% Decisor

dx=sign(d);

% Estimacion del comportamiento

[Nerr_bit,BER]=symerr(x,dx);

%% --------Visualizacion de resultados-----------------

RES=[K Eb_No E Nerr_bit BER];
disp('Los resultados son, utilizando la wavelet:'),disp(tipo_wavelet)
disp(' NumBit Eb/No NumEscalas NumErrBit BER');
disp(RES);
BER_teorica=0.5*erfc(sqrt(10^(0.1*Eb_No)));
disp('Resultados teoricos:');
disp(BER_teorica);
```

APENDICE A7. MODULACION WAVELET MULTIESCALA M-ARIA (N=3) N.R SSB (M_WMMSSSB)

```
% ------ MODULACION WAVELET MULTIESCALA M-ARIA (N=3) N.R SSB ---------
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Parametros del sistema

tipo_wavelet='db8';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);  % Filtros prototipo de analisis
                                       % y síntesis.
```

```
L=length(ha);           % Longitud de los filtros prototipo
E=3;                    % Numero de escalas de modulacion
N=7;                    % Numero de iteraciones
M=8;                    % Tamaño del alfabeto fuente
n=log2(M);              % Numero de bits mapeados en cada simbolo
Rs=2;                   % Tasa de simbolos en bauds/sg
fs=(2^N)*Rs/((2^E)-1);  % Frecuencia de muestreo en Hz
Bw=(2^E)*Rs/((2^E)-1);  % Ancho de banda del primer lobulo
Eb_No=18.28;   % Relacion energia de bit a densidad espectral de ruido
Es_No=Eb_No+10*log10(n);      % Relacion energia promedio de simbolo a
                              % densidad espectral de ruido.
fc=2*Rs;                % Frecuencia portadora
SNR=Es_No+10*log10(Rs/Bw);    % Relacion señal a ruido
K=100002;               % Numero de simbolos a transmitir
x=randsrc(1,K,[-(M-1):2:(M-1)]);      % Mensaje discreto a transmitir

%% Transmisor

% S/P

b1=[x(1:K/7) zeros(1,L-2)];
b2=[x(K/7+1:3*K/7) zeros(1,2*(L-2))];
b3=[x(3*K/7+1:K) zeros(1,4*(L-2))];

% IDWT

s1=filter(gs,1,upsample(b1,2));
s2=filter(gs,1,upsample(b2,2))+filter(hs,1,upsample(s1,2));
s3=filter(gs,1,upsample(b3,2))+filter(hs,1,upsample(s2,2));
s4=filter(hs,1,upsample(s3,2));
s5=filter(hs,1,upsample(s4,2));
s6=filter(hs,1,upsample(s5,2));
st=filter(hs,1,upsample(s6,2));       %Señal modulada en banda base
t=(0:length(st)-1)*(length(x)+N*(L-2))/(Rs*length(st)); % Base de
                                                        % tiempo.

% Modulador balanceado

sd=st.*cos(2*pi*fc*t);

% Filtro eliminador de banda

q=rcosfir(0.125,20,(fs/2)/Bw,1);  % Filtro de caida senoidal con
                                  % BW=2Rb (Primer lobulo).
b=(length(q)-1)/2;
u=-b:b;
fssb=(cos(2*pi*((fc+(0.5*Bw))/(fs))*u)).*rcosfir(0.125,10,(fs/2)/(Bw/2),1);
s=filter(fssb,1,[sd zeros(1,length(fssb)-1)]);
s=s(b+1:length(s)-b);             % Truncado a valores significativos

%% Canal AWGN

z=awgn(s,SNR-10*log10((fs/2)/Bw),'measured');

%% Receptor
```

```
% Prefiltraje pasa banda

zp=filter(fssb,1,[z zeros(1,length(fssb)-1)]);
zp=zp(b+1:length(zp)-b);        % Truncado a valores significativos

% Mezclador

w=zp.*cos(2*pi*fc*t);           % Demodulador balanceado
zw=filter(q,1,[w zeros(1,length(q)-1)]);
zw=zw(b+1:length(zw)-b);

% DWT

e1=filter(ha,1,[zw zeros(1,L-1)]);
e1=e1(L:length(e1));e1=downsample(e1,2);
e2=filter(ha,1,[e1 zeros(1,L-1)]);
e2=e2(L:length(e2));e2=downsample(e2,2);
e3=filter(ha,1,[e2 zeros(1,L-1)]);
e3=e3(L:length(e3));e3=downsample(e3,2);
e4=filter(ha,1,[e3 zeros(1,L-1)]);
e4=e4(L:length(e4));e4=downsample(e4,2);
e5=filter(ha,1,[e4 zeros(1,L-1)]);
e5=e5(L:length(e5));e5=downsample(e5,2);
d5=filter(ga,1,[e4 zeros(1,L-1)]);
d5=d5(L:length(d5));d5=downsample(d5,2);
e6=filter(ha,1,[e5 zeros(1,L-1)]);
e6=e6(L:length(e6));e6=downsample(e6,2);
d6=filter(ga,1,[e5 zeros(1,L-1)]);
d6=d6(L:length(d6));d6=downsample(d6,2);
d7=filter(ga,1,[e6 zeros(1,L-1)]);
d7=d7(L:length(d7));d7=downsample(d7,2);

% P/S

d=[d7(1:K/7) d6(1:2*K/7) d5(1:4*K/7)];
d=sqrt(sum(x.*x)/sum(d.*d))*d;

% Decisor

dx=zeros(1,length(d));

for (m=-0.5*M+1:0.5*M-1)
   dx=dx+sign(d-2*m);
end

% Estimacion del comportamiento

[Nerr_bit,SER]=symerr(x,dx);
BER=SER/log2(M);

% Dibujos(3)

u=-M-1+0.1:0.1:M+1;
y=zeros(1,length(u));
```

```
for (m=-0.5*M+1:0.5*M-1)
   y=y+sign(u-2*m);
end

figure(6)

stairs(u,y,'r'),grid,title('Esquema de decision'),xlabel('Entrada'),ylabel('Salida'),axis([-M-1
M+1 -M M])

%%--------Visualizacion de resultados-----------------

disp('Parametros del sistema, utilizando la wavelet'),disp(tipo_wavelet)
disp('      NumEscalas          M            Rs           Bw')
disp([E M Rs Bw])
RES=[K Eb_No Nerr_bit SER  BER];
disp('Los resultados son:')
disp('     NumBit      Eb/No       NumErrBit         SER        BER');
disp(RES);

%% Resultados teoricos

p=(10^(0.1*Es_No));
SER_teorica=(1-(1/M))*erfc(sqrt(3*p/((4^n)-1)));
BER_teorica=SER_teorica/n;
disp('Resultados teoricos:');
disp('    SER_teorica    BER_teorica');
disp([SER_teorica BER_teorica]);
```

APENDICE A8. MODULACION WAVELET MULTIESCALA M-ARIA EN CUADRATURA (N=3) N.R DSB (MQ_WMMSDSB)

```
% - MODULACION WAVELET MULTIESCALA M-ARIA EN CUADRATURA (N=3) N.R
DSB-
% ------------------------------------------------------------------

%% Acondicionamiento inicial

clear all
format short g

%% Parametros del sistema

tipo_wavelet='db5';
[ha,ga,hs,gs]=wfilters(tipo_wavelet);  % Filtros prototipo de analisis
                                       % y síntesis.
L=length(ha);          % Longitud de los filtros prototipo
E=3;                   % Numero de escalas de modulacion
N=7;                   % Numero de iteraciones
M=64;                  % Tamaño del alfabeto fuente
n=log2(M);             % Numero de bits mapeados en cada simbolo
Rs=2;                  % Tasa de simbolos en bauds/sg
fs=(2^N)*Rs/((2^E)-1);  % Frecuencia de muestreo en Hz
Bw=2*(2^E)*Rs/((2^E)-1);% Ancho de banda del primer lobulo
Eb_No=18.38;   % Relacion energia de bit a densidad espectral de ruido
```

```
Es_No=Eb_No+10*log10(n);% Relacion energia promedio de simbolo a
                        % densidad espectral de ruido
fc=2*Rs;                % Frecuencia portadora
SNR=Es_No+10*log10(Rs/Bw);        % Relacion señal a ruido
K=100002;                         % Numero de simbolos a transmitir
a=randsrc(1,K,[-(sqrt(M)-1):2:(sqrt(M)-1)]);      %Simbolos canal I
b=randsrc(1,K,[-(sqrt(M)-1):2:(sqrt(M)-1)]);      %Simbolos canal Q
c=a+b*i;                          %Simbolo complejo transmitido

%% Transmisor

% S/P

w1=[a(1:K/7) zeros(1,L-2)];
w2=[a(K/7+1:3*K/7) zeros(1,2*(L-2))];
w3=[a(3*K/7+1:K) zeros(1,4*(L-2))];
v1=[b(1:K/7) zeros(1,L-2)];
v2=[b(K/7+1:3*K/7) zeros(1,2*(L-2))];
v3=[b(3*K/7+1:K) zeros(1,4*(L-2))];

% IDWT

a1=filter(gs,1,upsample(w1,2));
a2=filter(gs,1,upsample(w2,2))+filter(hs,1,upsample(a1,2));
a3=filter(gs,1,upsample(w3,2))+filter(hs,1,upsample(a2,2));
a4=filter(hs,1,upsample(a3,2));
a5=filter(hs,1,upsample(a4,2));
a6=filter(hs,1,upsample(a5,2));
at=filter(hs,1,upsample(a6,2));   %Señal banda base canal I
b1=filter(gs,1,upsample(v1,2));
b2=filter(gs,1,upsample(v2,2))+filter(hs,1,upsample(b1,2));
b3=filter(gs,1,upsample(v3,2))+filter(hs,1,upsample(b2,2));
b4=filter(hs,1,upsample(b3,2));
b5=filter(hs,1,upsample(b4,2));
b6=filter(hs,1,upsample(b5,2));
bt=filter(hs,1,upsample(b6,2));   %Señal banda base canal Q
t=(0:length(at)-1)*(length(a)+N*(L-2))/(Rs*length(at));%Base de tiempo
s=at.*cos(2*pi*fc*t)+bt.*sin(2*pi*fc*t);   % Señal modulada wavelet en
                                  % cuadratura

%% Canal AWGN

z=awgn(s,SNR-10*log10((fs/2)/Bw),'measured');

%% Receptor

% Mezclador

q=rcosfir(0.125,10,(fs/2)/(Bw/2),1);        % Filtro de demodulacion
T=(length(q)-1)/2;
za=z.*cos(2*pi*fc*t);
zb=z.*sin(2*pi*fc*t);
ar=filter(q,1,[za zeros(1,length(q)-1)]);
ar=ar(T+1:length(ar)-T);     % Señal truncada a su rango significativo
br=filter(q,1,[zb zeros(1,length(q)-1)]);
```

```
br=br(T+1:length(br)-T);     % Señal truncada a su rango significativo

% DWT

ea1=filter(ha,1,[ar zeros(1,L-1)]);
ea1=ea1(L:length(ea1));ea1=downsample(ea1,2);
ea2=filter(ha,1,[ea1 zeros(1,L-1)]);
ea2=ea2(L:length(ea2));ea2=downsample(ea2,2);
ea3=filter(ha,1,[ea2 zeros(1,L-1)]);
ea3=ea3(L:length(ea3));ea3=downsample(ea3,2);
ea4=filter(ha,1,[ea3 zeros(1,L-1)]);
ea4=ea4(L:length(ea4));ea4=downsample(ea4,2);
ea5=filter(ha,1,[ea4 zeros(1,L-1)]);
ea5=ea5(L:length(ea5));ea5=downsample(ea5,2);
da5=filter(ga,1,[ea4 zeros(1,L-1)]);
da5=da5(L:length(da5));da5=downsample(da5,2);
ea6=filter(ha,1,[ea5 zeros(1,L-1)]);
ea6=ea6(L:length(ea6));ea6=downsample(ea6,2);
da6=filter(ga,1,[ea5 zeros(1,L-1)]);
da6=da6(L:length(da6));da6=downsample(da6,2);
da7=filter(ga,1,[ea6 zeros(1,L-1)]);
da7=da7(L:length(da7));da7=downsample(da7,2);    %Canal I
eb1=filter(ha,1,[br zeros(1,L-1)]);
eb1=eb1(L:length(eb1));eb1=downsample(eb1,2);
eb2=filter(ha,1,[eb1 zeros(1,L-1)]);
eb2=eb2(L:length(eb2));eb2=downsample(eb2,2);
eb3=filter(ha,1,[eb2 zeros(1,L-1)]);
eb3=eb3(L:length(eb3));eb3=downsample(eb3,2);
eb4=filter(ha,1,[eb3 zeros(1,L-1)]);
eb4=eb4(L:length(eb4));eb4=downsample(eb4,2);
eb5=filter(ha,1,[eb4 zeros(1,L-1)]);
eb5=eb5(L:length(eb5));eb5=downsample(eb5,2);
db5=filter(ga,1,[eb4 zeros(1,L-1)]);
db5=db5(L:length(db5));db5=downsample(db5,2);
eb6=filter(ha,1,[eb5 zeros(1,L-1)]);
eb6=eb6(L:length(eb6));eb6=downsample(eb6,2);
db6=filter(ga,1,[eb5 zeros(1,L-1)]);
db6=db6(L:length(db6));db6=downsample(db6,2);
db7=filter(ga,1,[eb6 zeros(1,L-1)]);
db7=db7(L:length(db7));db7=downsample(db7,2);    %Canal I

% P/S

da=[da7(1:K/7) da6(1:2*K/7) da5(1:4*K/7)];
da=sqrt(sum(a.*a)/sum(da.*da))*da;
db=[db7(1:K/7) db6(1:2*K/7) db5(1:4*K/7)];
db=sqrt(sum(b.*b)/sum(db.*db))*db;
d=da+db*i;

%Decisor

aD=zeros(1,length(da));
bD=zeros(1,length(db));

for (m=-0.5*sqrt(M)+1:0.5*sqrt(M)-1)
```

```
    aD=aD+sign(da-2*m);
    bD=bD+sign(db-2*m);
end

cD=aD+bD*i;             %Simbolos detectados

%Estimacion del comportamiento

[Nerr_sym,SER]=symerr(c,cD);   % Estimacion del numero de errores y la
                               % tasa de error de símbolo.
BER=SER/n;

% Dibujos (4)

u=-sqrt(M)-1+0.1:0.1:sqrt(M)+1;
y=zeros(1,length(u));

for (m=-0.5*sqrt(M)+1:0.5*sqrt(M)-1)
    y=y+sign(u-2*m);
end

figure (9)

subplot(221),stairs(u,y,'r'),grid,title('Esquema de decision canal
I'),xlabel('Entrada'),ylabel('Salida'),axis([-sqrt(M)-1 sqrt(M)+1 -sqrt(M) sqrt(M)])

subplot(222),stairs(u,y,'m'),grid,title('Esquema de decision canal
Q'),xlabel('Entrada'),ylabel('Salida'),axis([-sqrt(M)-1 sqrt(M)+1 -sqrt(M) sqrt(M)])

%%--------Visualizacion de resultados-----------------

disp('Parametros del sistema, utilizando la wavelet'),disp(tipo_wavelet)
disp('      NumEscalas          M            Rs           Bw')
disp([E M Rs Bw])
RES=[K Eb_No Nerr_sym SER BER];
disp('Los resultados son:')
disp('     NumBit       Eb/No       NumErrBit        SER        BER');
disp(RES);

%% Resultados teoricos

p=(10^(0.1*Es_No));
SER_teorica=2*(1-sqrt(1/M))*erfc(sqrt(1.5*p/(M-1)));
BER_teorica=SER_teorica/n;
disp('Resultados teoricos:');
disp('    SER_teorica    BER_teorica');
disp([SER_teorica BER_teorica]);
```

APÉNDICE A9. CONSIDERACIONES PARA TRANSMISIÓN EN TIEMPO REAL

Cuando se quiere transmitir un flujo de información en tiempo real, se necesita agrupar los datos en bloques iguales de determinada longitud, los cuales van siendo modulados sucesivamente según su orden de llegada a un dispositivo de almacenamiento temporal conocido como *buffer*. La modulación de cada bloque independiente requiere un tiempo de procesamiento, el cual, junto con los retardos de transmisión, representa el retardo total del enlace. El desafío más importante en la transmisión en tiempo real es lograr que el usuario final perciba continuidad en el flujo de información que fue enviada desde el transmisor. Es decir, que el tiempo transcurrido entre el inicio y el final del bloque total de información sea el mismo tanto en el transmisor como en el receptor. Este objetivo se cumple dividiendo el flujo total de información en bloques iguales, con lo que se garantiza igual retardo para cada bloque.

Por otro lado, se sabe que un bloque de datos modulado en *wavelet* ocupa un ancho temporal más grande que el tiempo utilizado para generar los datos. En ese sentido pueden presentarse dos grandes problemas que pueden degradar notablemente la calidad del sistema de comunicación, como se observa en la figura A9.1. El primero es el fenómeno conocido como *Interferencia Inter-Trama* (IFI: "Inter-frame interferente"), el cual se produce al transmitir secuencialmente los bloques, es decir sin un tiempo de espera entre bloques consecutivos, lo que ocasiona que la parte final de la forma de onda modulada de cada bloque se traslape con la parte inicial de la siguiente, produciendo errores en la detección de los datos ubicados en estos sectores de las tramas. El segundo problema, conocido como *retardo acumulativo*, se produce cuando hay un tiempo de espera entre la transmisión de cada bloque consecutivo con el propósito de garantizar que la forma de onda correspondiente al símbolo anterior se desvanezca en el tiempo, para así poder enviar sobre el canal la siguiente forma de onda perteneciente a la trama actual. Esto provoca que el flujo de información original no llegue en forma continua al receptor y se pierda el carácter de tiempo real, pues cada bloque llegará a su destino aisladamente, separados entre si un tiempo igual al alargamiento temporal de la forma de onda asociada al bloque.

La solución para evitar cualquiera de estos inconvenientes, es hacer que la duración de la señal modulada sea exactamente igual a la duración de la secuencia de datos que representa. Con lo que se anula el *retardo acumulativo* entre tramas y la *interferencia entre tramas*. El precio a pagar es un leve incremento en el ancho de banda de transmisión. Para calcular dicho incremento se tiene lo siguiente:

Sea $s(t)$ la señal modulada en *wavelet* que representa a la secuencia generadora $x[n]$ cuya duración es T segundos, la duración de $s(t)$ será entonces $T + p$ segundos, donde p corresponde al exceso de duración en la forma de onda. Para modificar la duración de $s(t)$ e igualarla a T segundos es necesario escalonar a $s(t)$ por un factor constante a mayor que la unidad (es decir, comprimir a $s(t)$ en el tiempo), y transmitir dicha versión escalonada $s(a\,t)$ en lugar de $s(t)$. El factor a estará dado por:

$$a = \frac{T+p}{T} = 1 + \frac{p}{T}, \qquad \text{(A9.1)}$$

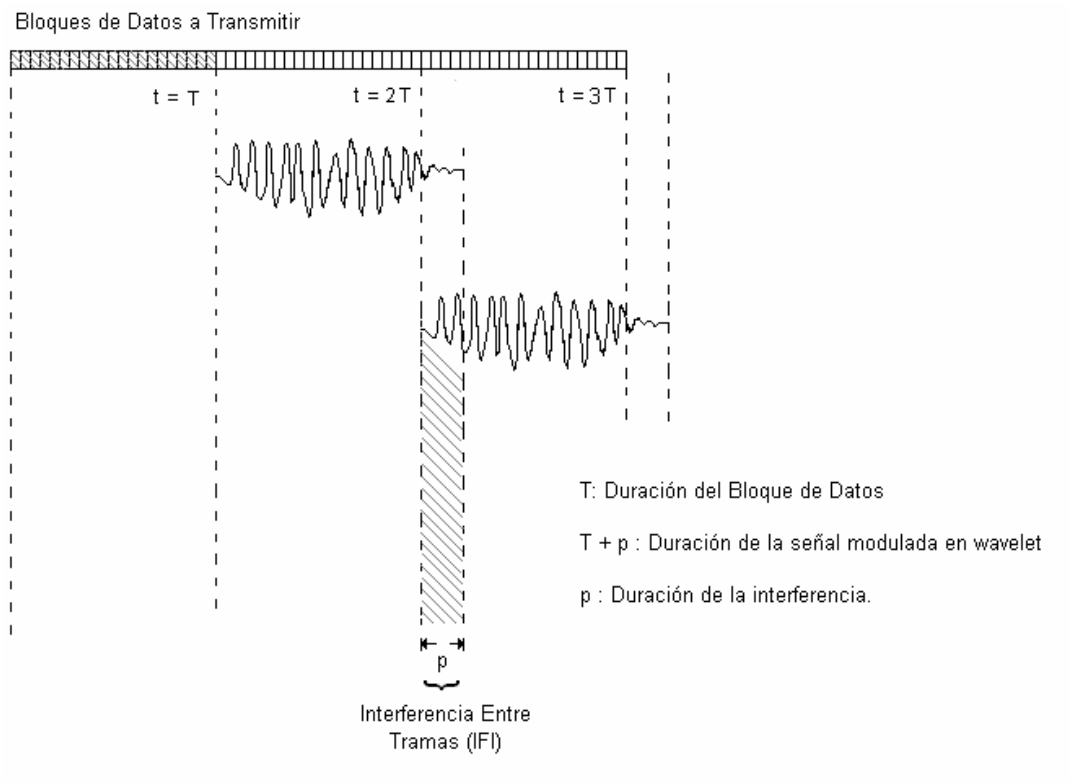

Figura A9.1. Gráfico ilustrativo de la interferencia inter-trama.

pero $p = \frac{T}{K}(L-1)$, donde *K* es la longitud en símbolos de la secuencia generadora y *L* es la longitud de los filtros de la wavelet utilizada. Entonces, el factor de escalonamiento a viene dado por:

$$a = 1 + \frac{L-1}{k} \tag{A9.2}$$

El espectro de la señal escalonada está dado por:

$$CFT\{s(at)\} = \frac{1}{|a|} S\left(\frac{f}{a}\right) \tag{A9.3}$$

Esto quiere decir que el ancho de banda se incrementa por un factor a como era de esperarse, debido a la contracción temporal de la señal. En ese sentido, el nuevo ancho de banda está dado por:

$$Bw = 2\,a\,R_s = 2\left[1 + \frac{L-1}{k}\right] R_s \tag{A9.4}$$

En la ecuación (A9.4) se observa que el ancho de banda no se modifica cuando se usa la *wavelet* de Haar (*L*=1). Por otro lado, cuando se utiliza una *wavelet* Daubechies de orden *N*, con *L*= 2*N* – 1, el ancho de banda será:

$$Bw = 2\left[1 + \frac{2N-2}{k}\right]R_s \qquad \text{(A9.5)}$$

Aquí se nota claramente que el incremento de ancho de banda es más pequeño mientras más grande sea la longitud del bloque de datos, por tanto, sería deseable diseñar un sistema con bloques grandes para hacer que el incremento sea imperceptible. Sin embargo los bloques muy largos generan un retardo total mayor y exigen mayor capacidad en los dispositivos de almacenamiento temporal, con lo cual, la elección del tamaño de los bloques es un compromiso entre efectividad en el uso del espectro, complejidad del sistema y calidad percibida por el usuario final.

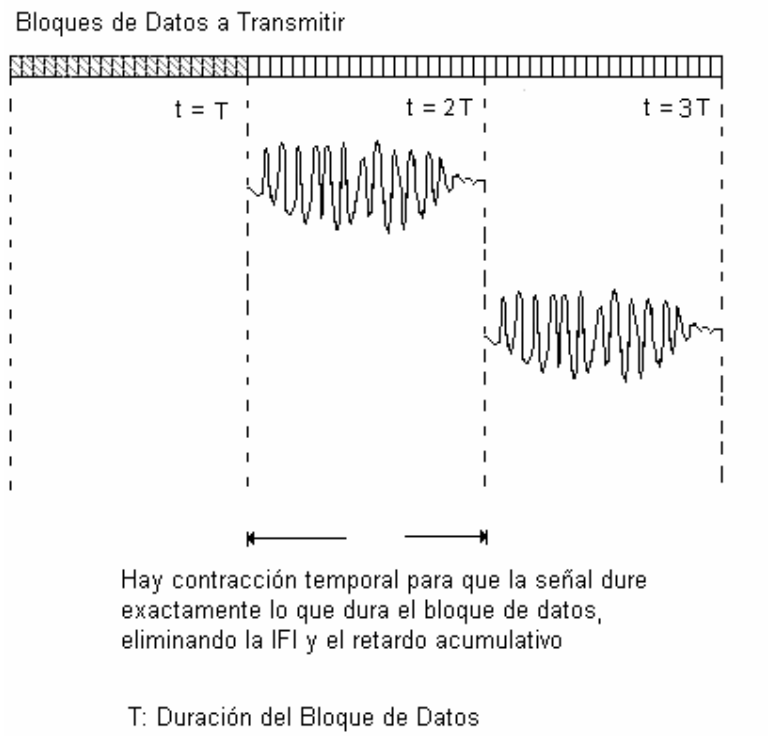

Figura A9.2. Transmisión en tiempo real aplicando contracción temporal.

Por otro lado, se sabe que a medida que aumenta el orden una *wavelet*, mayor es su duración, pero más amplio es el *gap* de energía generado entre los extremos del lóbulo principal. Luego, se puede asumir en forma muy general que el ancho de banda sigue siendo aproximadamente el mismo para la señal comprimida, esto para una longitud suficientemente grande del bloque de datos. Dada esta característica de las *wavelets*, puede decirse que el incremento de ancho de banda no es un aspecto preocupante, pues la distribución espectral de las *wavelets* de orden *N* compensa su exceso de duración a fin de mantener fijo el ancho de banda utilizado. En esos términos, el filtro pasa bajo del transmisor puede seguir teniendo un ancho de banda $Bw = 2R_s$ sin que ésto represente distorsión en la información.