Nombre: Eguiarte Morett Luis Andrés.

Proyecto: Simulación de un sistema binario en banda base.

1) Generar una señal (pseudo)-analógica. Un equipo de cómputo no puede producir señales analógicas, en realidad lo que se genera es una señal pseudo-analógica con muchas muestras por segundo. ¿Cómo generar la señal analógica? Como la suma de varias cosenoidales cada una con diferente frecuencia, amplitud y fase (las amplitudes y fases se elijen de forma aleatoria, las frecuencias como múltiplos enteros (n=1, 2, 3, ... 10) de una frecuencia fundamental f0). Para el proyecto:

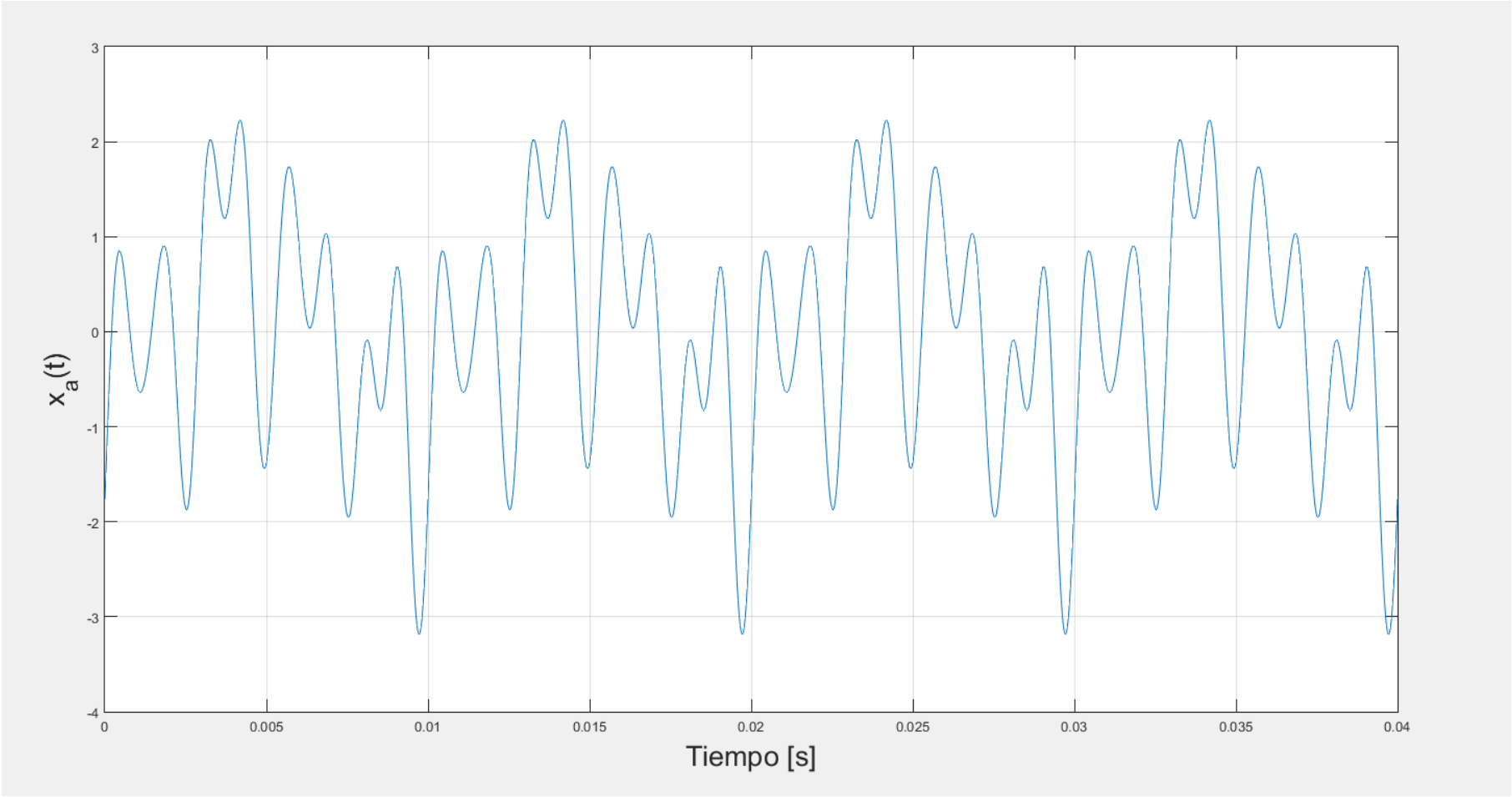
Duración de la señal analógica: 5 segundos,

Número de cosenoidales n:10,

f0 = 100 Hz,

muestras/segundo = 320, 000 (Es decir, Resolución, Res = 32).

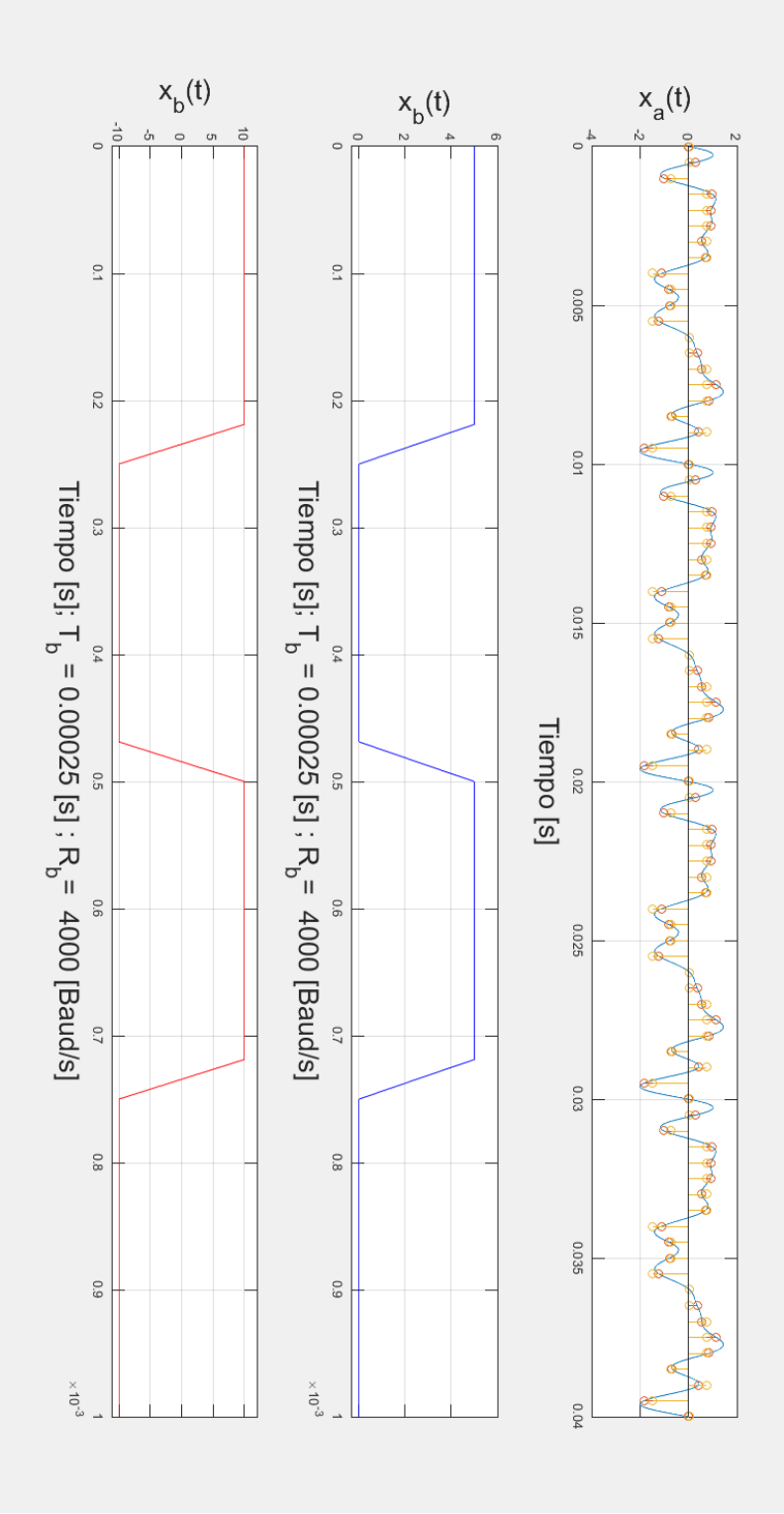
Graficar en el reporte los primeros 4 períodos (=40 milisegundos) de esta señal analógica (indicar bien el eje del tiempo y de la amplitud, con su escala).


3
2
1
0
-1
-2
-3
-4
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
$x_a(t)$
Tiempo [s]

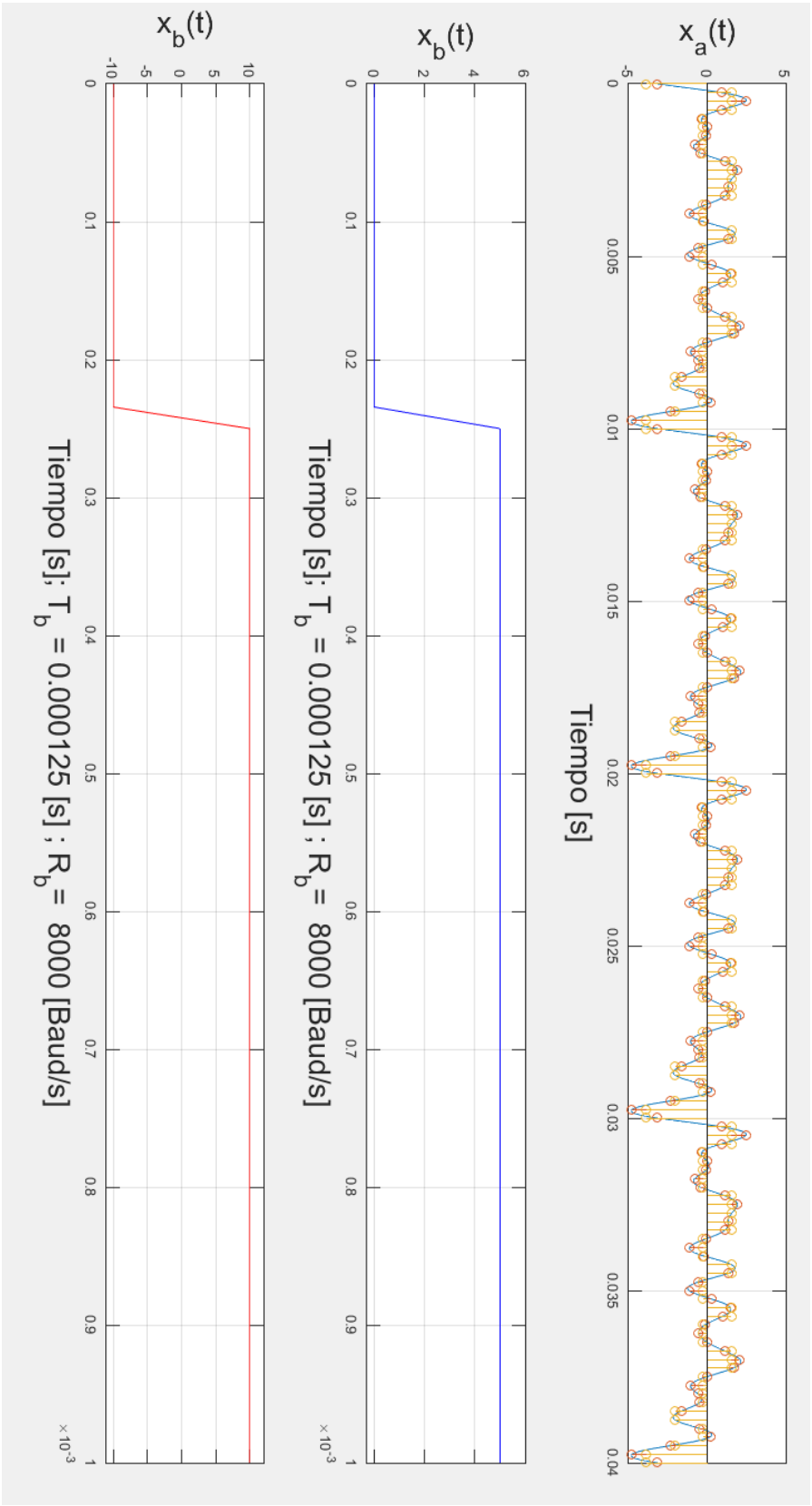
2) Digitalizar toda la señal pseudo-analógica para diferentes casos:

a) m = 4, fs = 2 KHz, 4 KHz y 8 KHz.

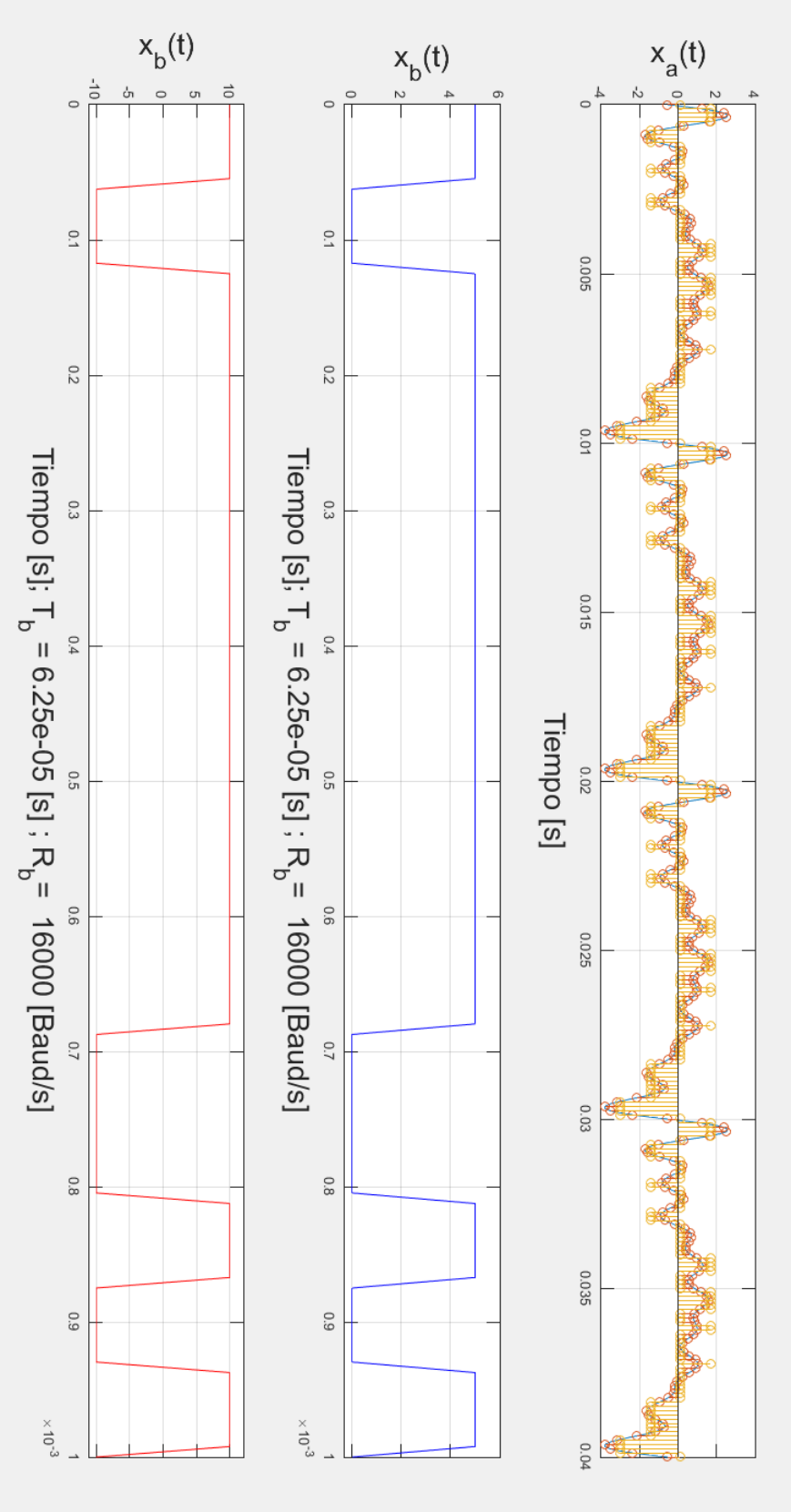
i. fs = 2 KHz



ii. fs= 4 KHz

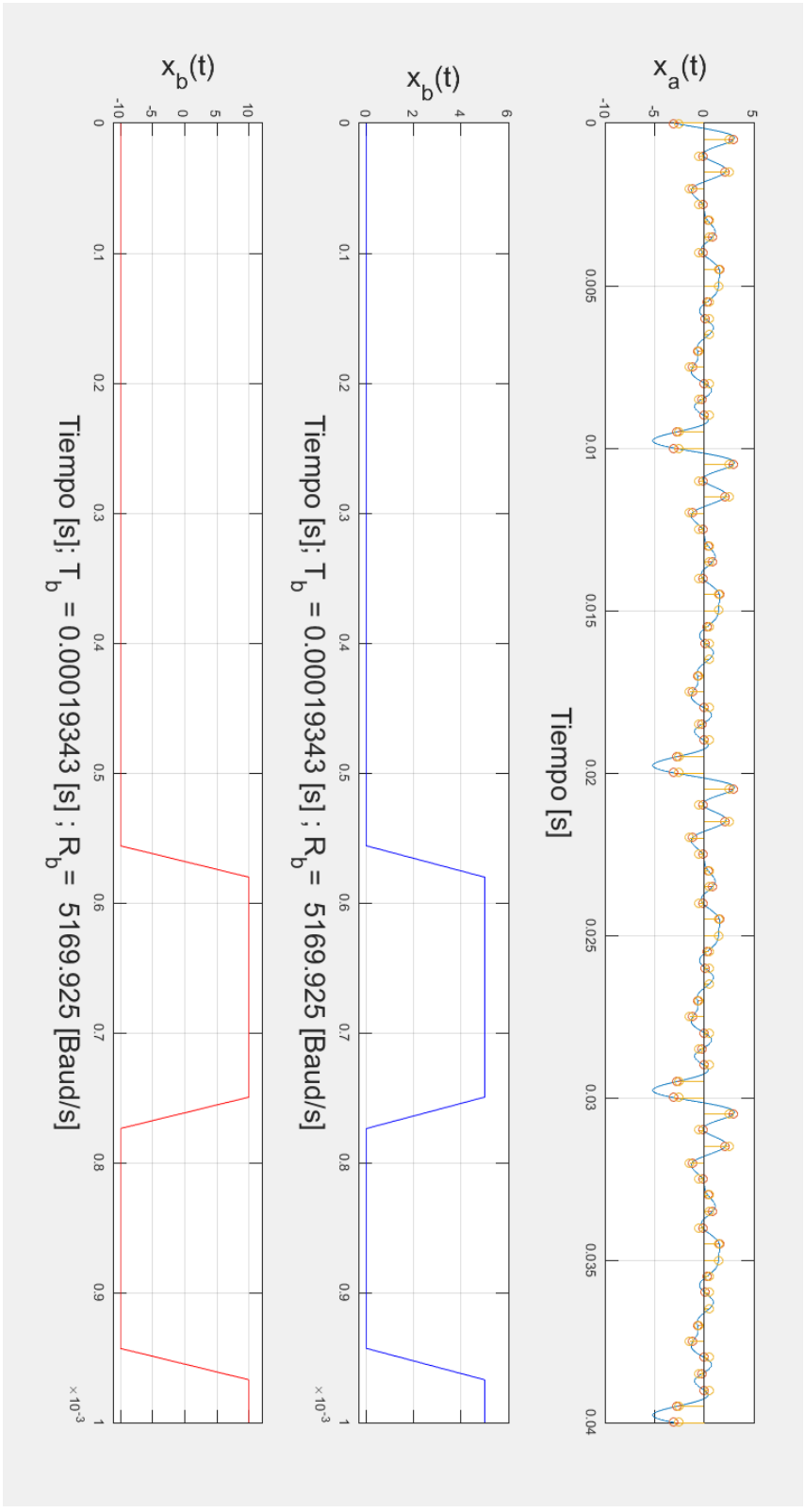

x_a(t)
Tiempo [s]
x_b(t)
Tiempo [s]; T_b = 0.000125 [s] ; R_b = 8000 [Baud/s]
x_b(t)
Tiempo [s]; T_b = 0.000125 [s] ; R_b = 8000 [Baud/s]

iii. fs = 8 KHz

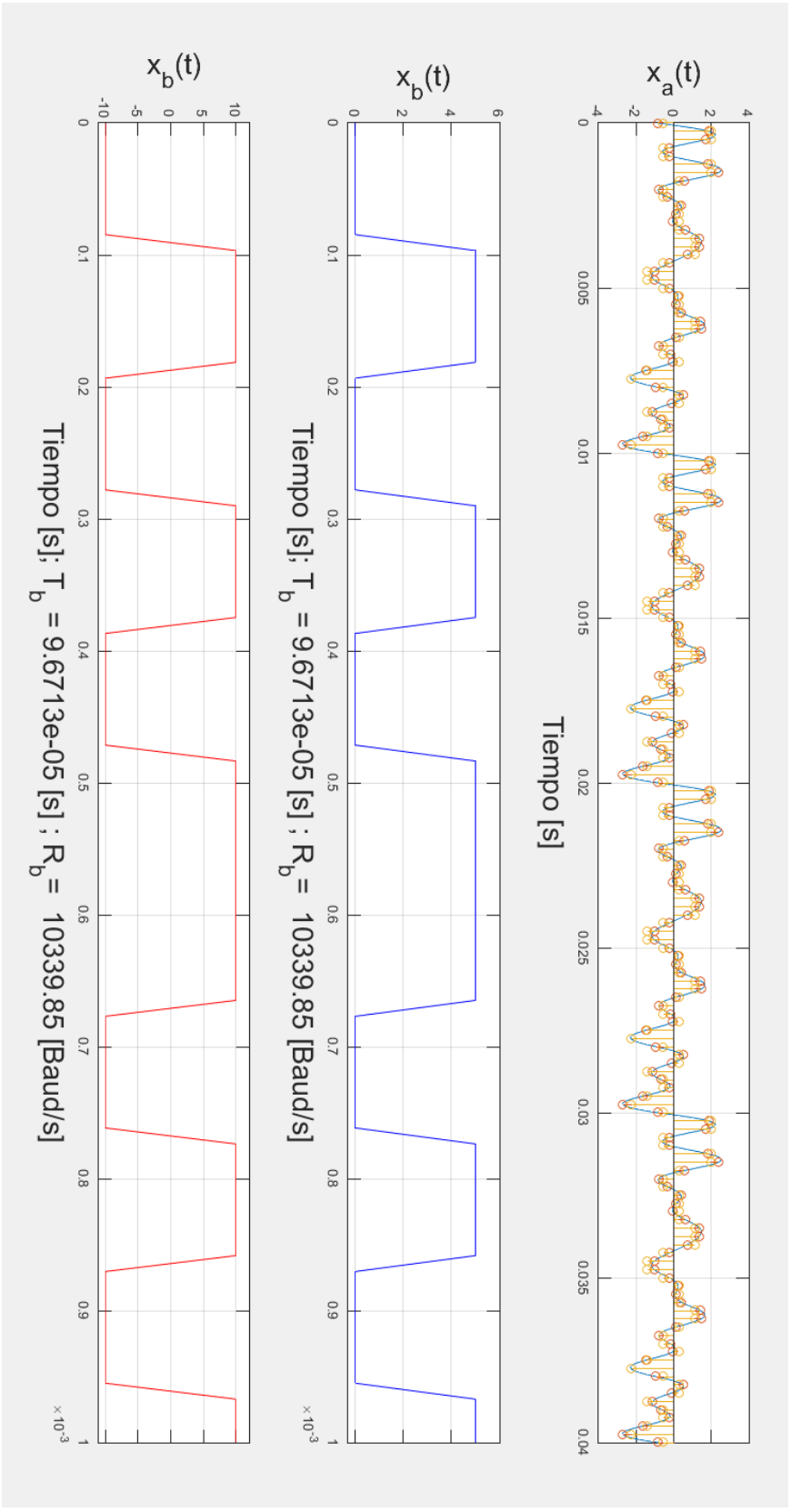

$x_a(t)$
Tiempo [s]
$x_b(t)$
Tiempo [s]; T_b = 6.25e-05 [s] ; R_b = 16000 [Baud/s]
$x_b(t)$
Tiempo [s]; T_b = 6.25e-05 [s] ; R_b = 16000 [Baud/s]
$\times 10^{-3}$

b) m = 6, fs = 2 KHz, 4 KHz y 8 KHz.

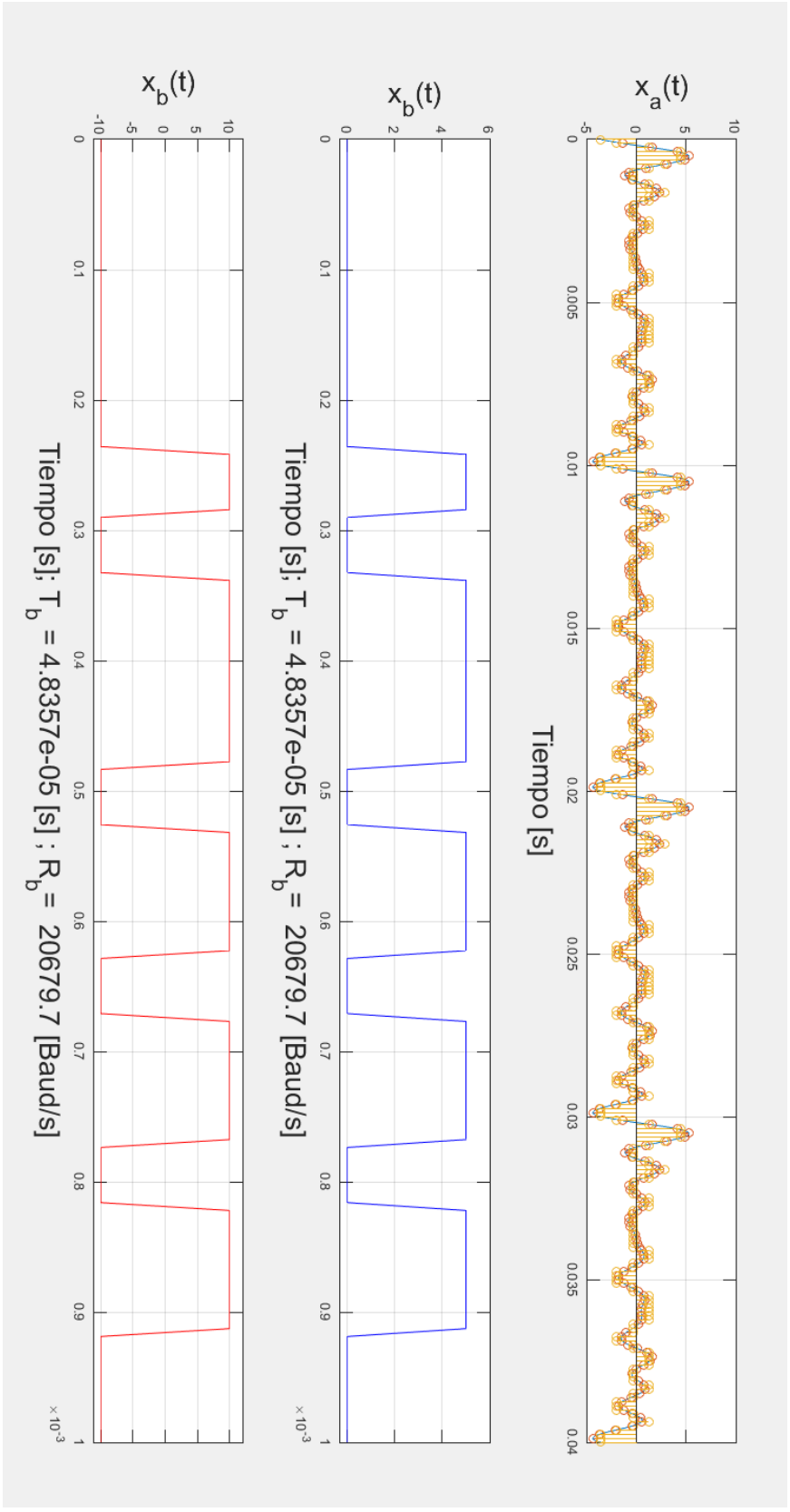
i. fs= 2KHz



ii. fs=4 KHz

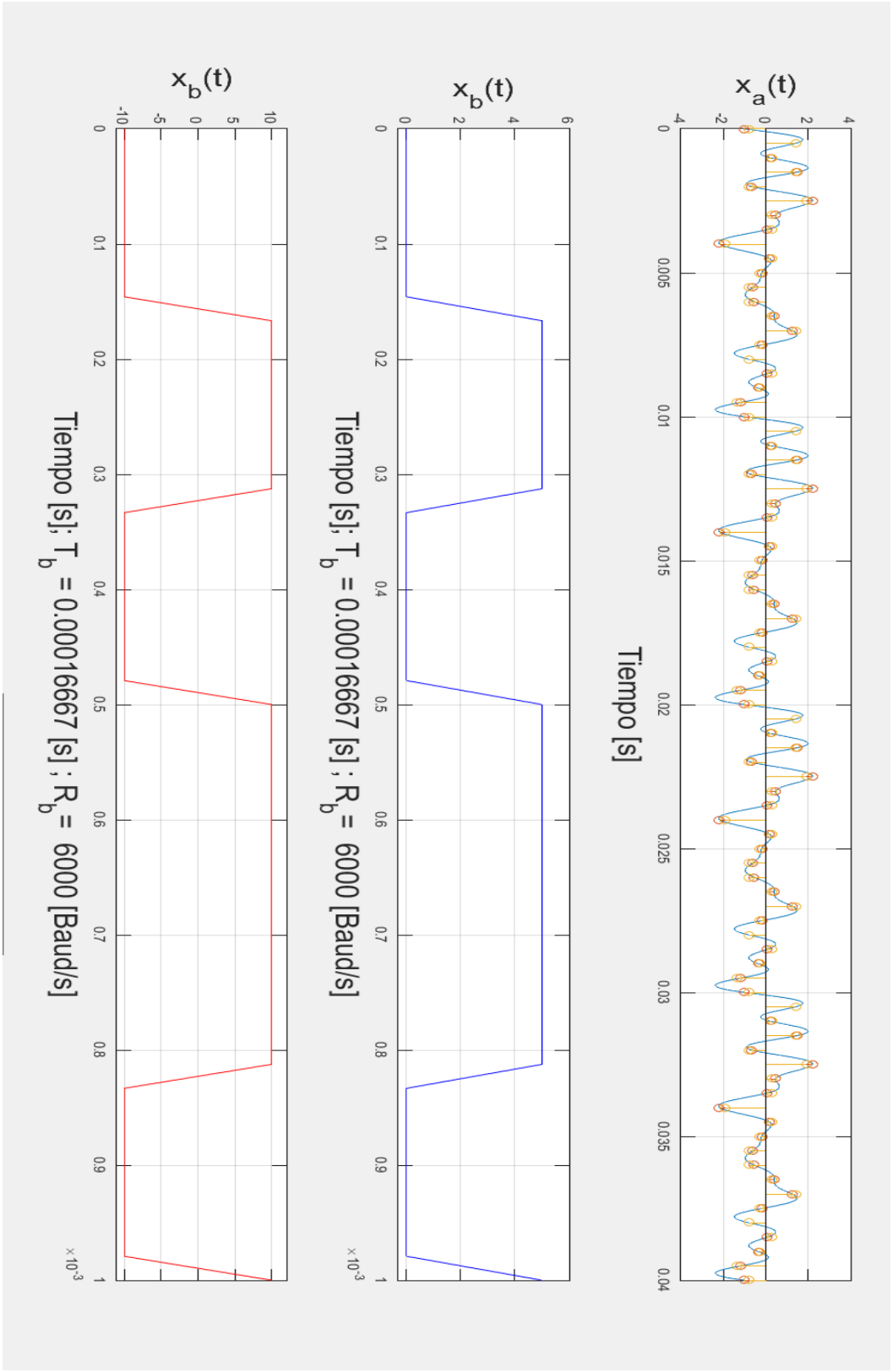

$x_a(t)$
Tiempo [s]
$x_b(t)$
Tiempo [s]; T_b = 9.6713e-05 [s] ; R_b = 10339.85 [Baud/s]
$x_b(t)$
Tiempo [s]; T_b = 9.6713e-05 [s] ; R_b = 10339.85 [Baud/s]

iii. fs=8 KHz

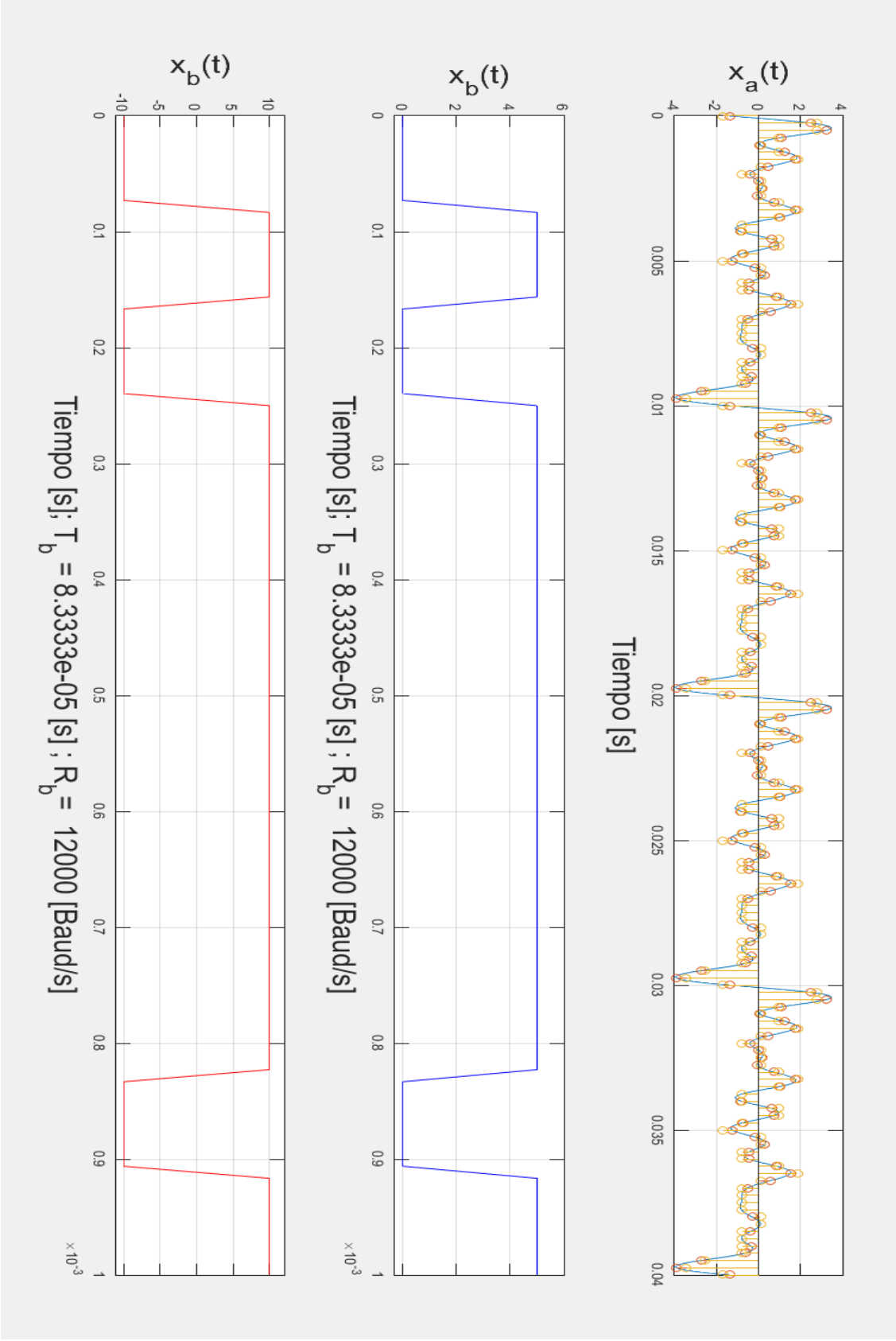

$x_a(t)$
Tiempo [s]
$x_b(t)$
Tiempo [s]; T_b = 4.8357e-05 [s] ; R_b = 20679.7 [Baud/s]
$x_b(t)$
Tiempo [s]; T_b = 4.8357e-05 [s] ; R_b = 20679.7 [Baud/s]

c) m = 8, fs = 2 KHz, 4 KHz y 8 KHz.

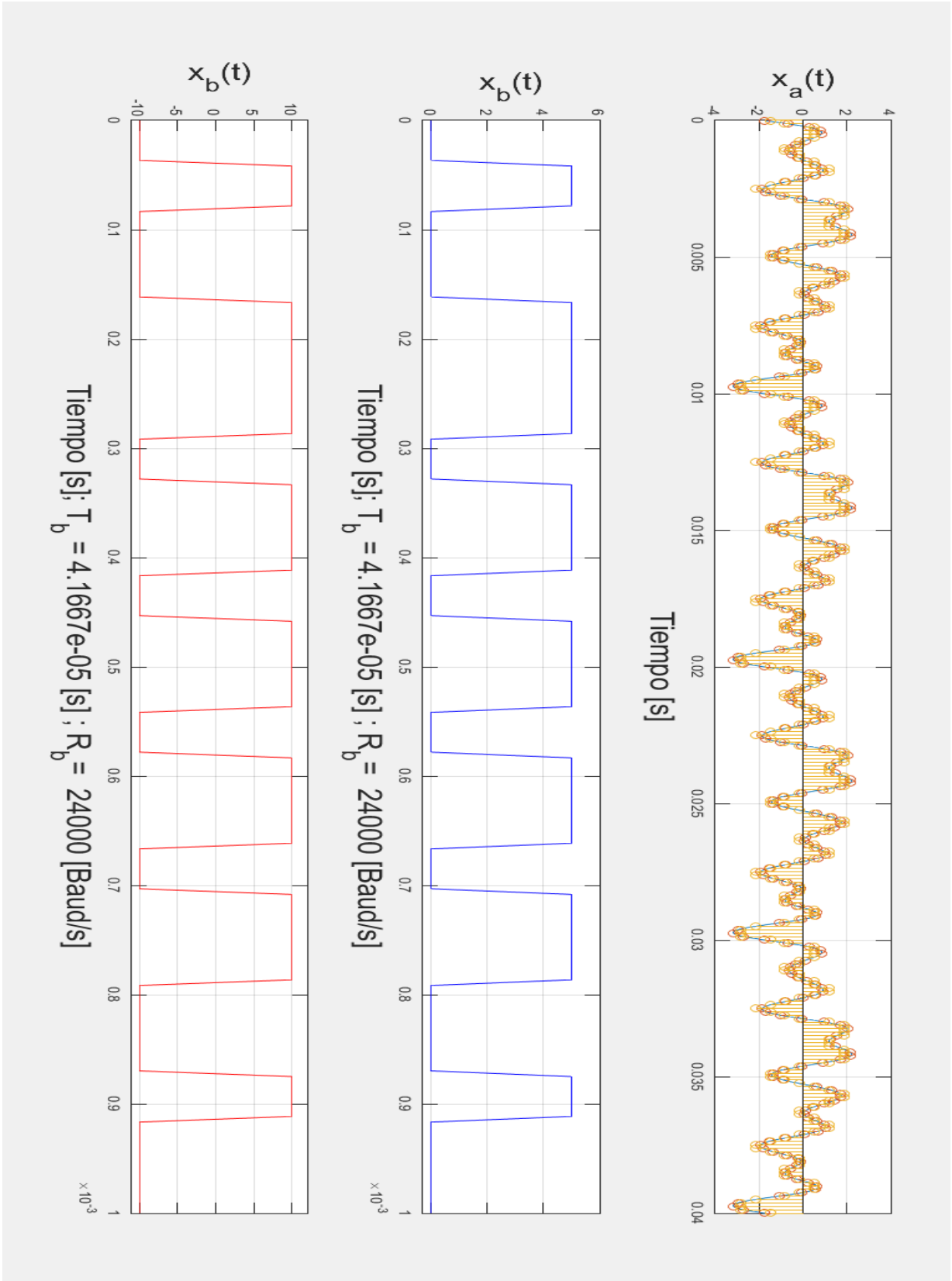
i. fs= 2KHz



ii. fs=4KHz


$x_a(t)$
Tiempo [s]
$x_b(t)$
Tiempo [s]; T_b = 8.3333e-05 [s] ; R_b = 12000 [Baud/s]
$x_b(t)$
Tiempo [s]; T_b = 8.3333e-05 [s] ; R_b = 12000 [Baud/s]
$\times 10^{-3}$

iii. fs= 8KHz

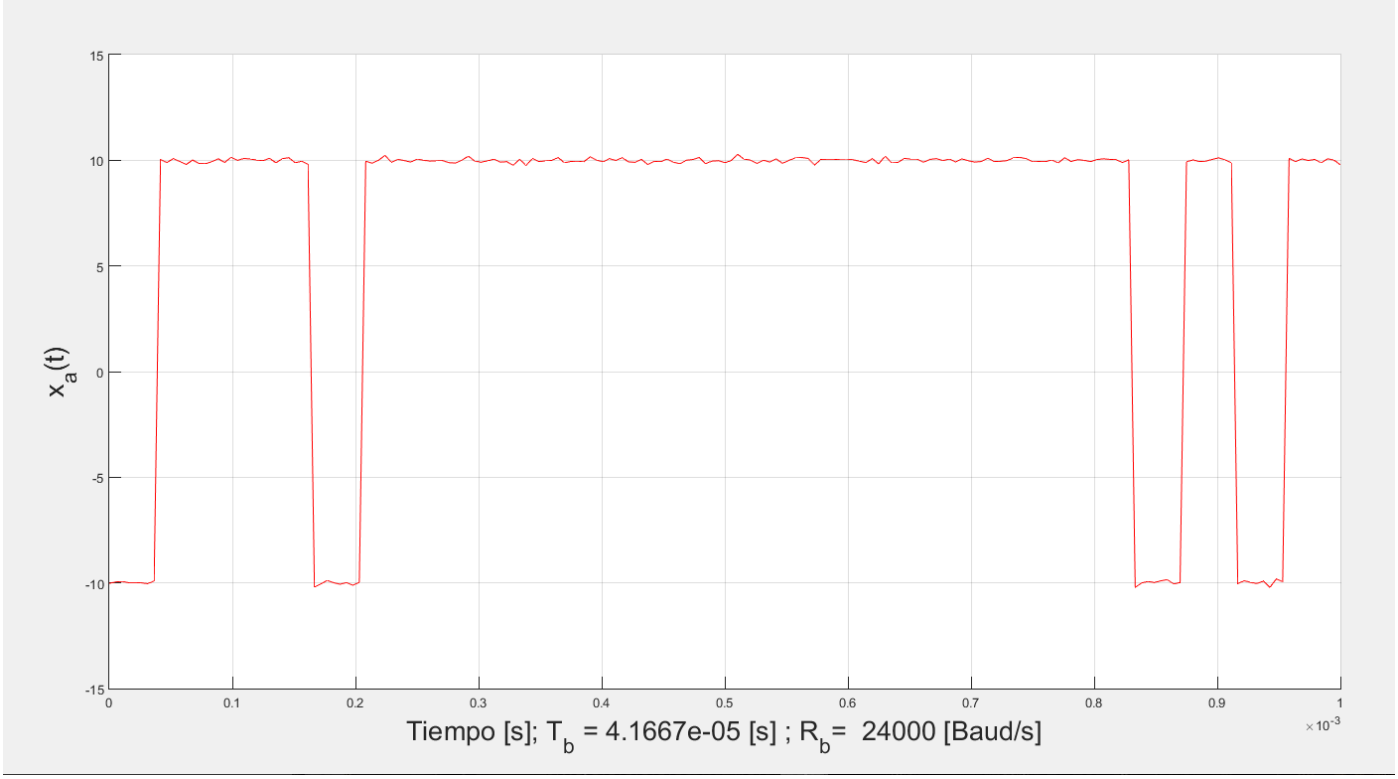

$x_a(t)$
Tiempo [s]
$x_b(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b = 24000 [Baud/s]
$x_b(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b = 24000 [Baud/s]

Graficar en el reporte la señal analógica (continua con plot), la señal muestreada (solamente círculos con stem) y la señal cuantizada (solamente círculos con stem) (las tres dos superpuestas, cada una con un color diferente) para cada uno de los 9 casos, desde 0 a 10 milisegundos.

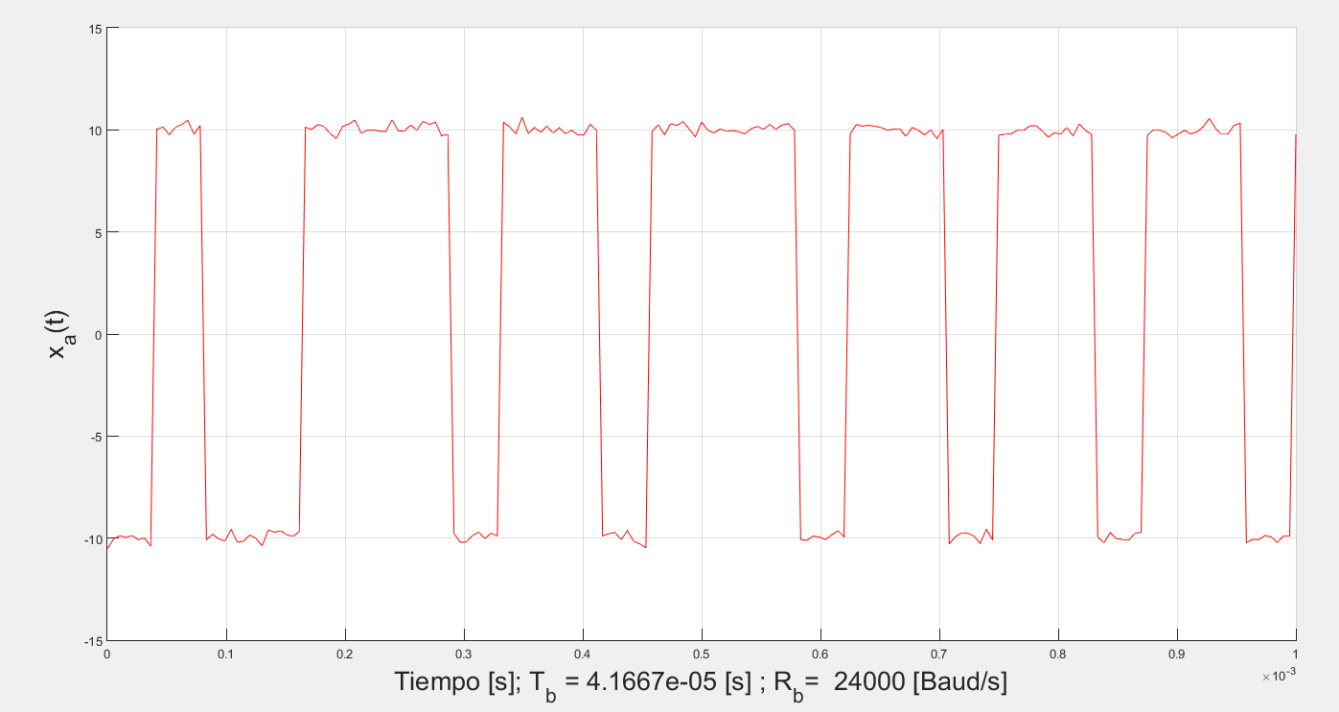
Graficar la señal binaria (cada bit debe tener 8 muestras de resolución) con un código de línea NRZ-bipolar [A = 5 V] para los 9 casos, de 0 a 1 ms, y de -10 a 10 Volts. Para los 9 casos, calcular y anotar la tasa binaria (Rb) y la duración del bit (Tb). Verificar que la duración del bit calculada coincida con la medida en las gráficas.

3) Para el caso en que m = 8 y fs = 8 KHz, sumar a la señal binaria completa (debe durar 5 segundos, igual que la señal pseudo-analógica completa) ruido blanco gaussiano (con el mismo número de muestras que la señal binaria completa), con una desviación estándar de:

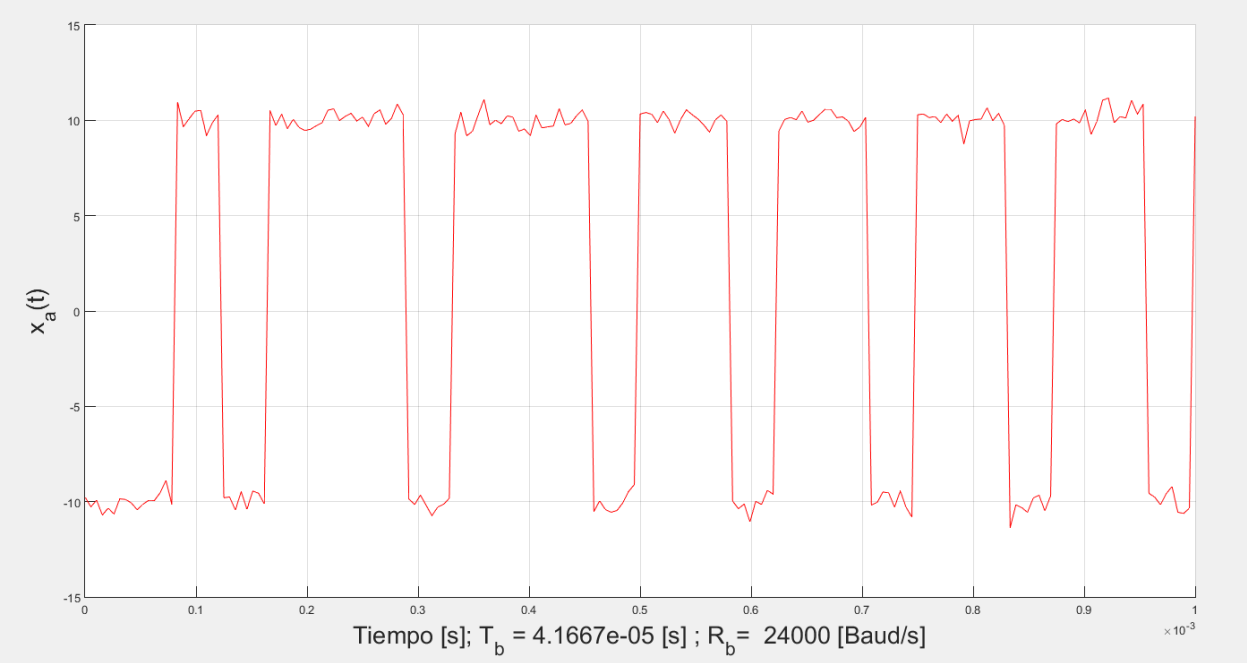
a) 100 mV



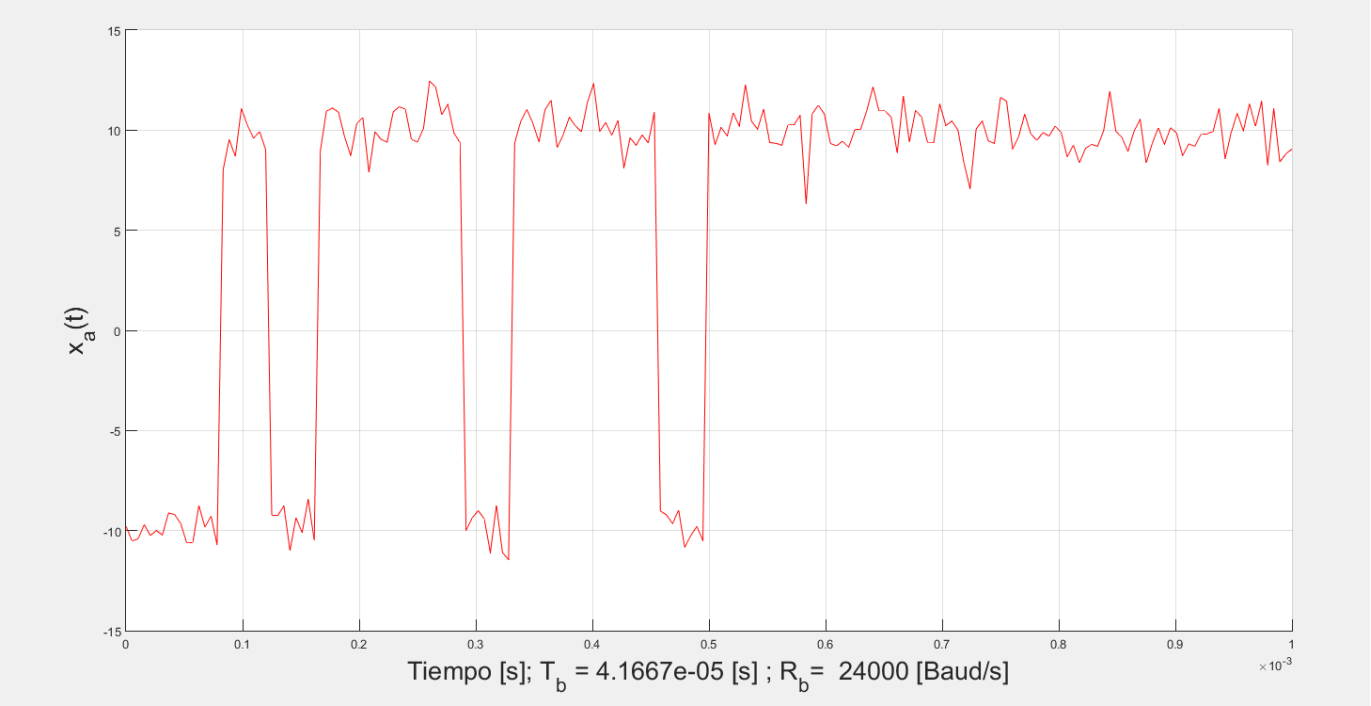
b) 250 mV



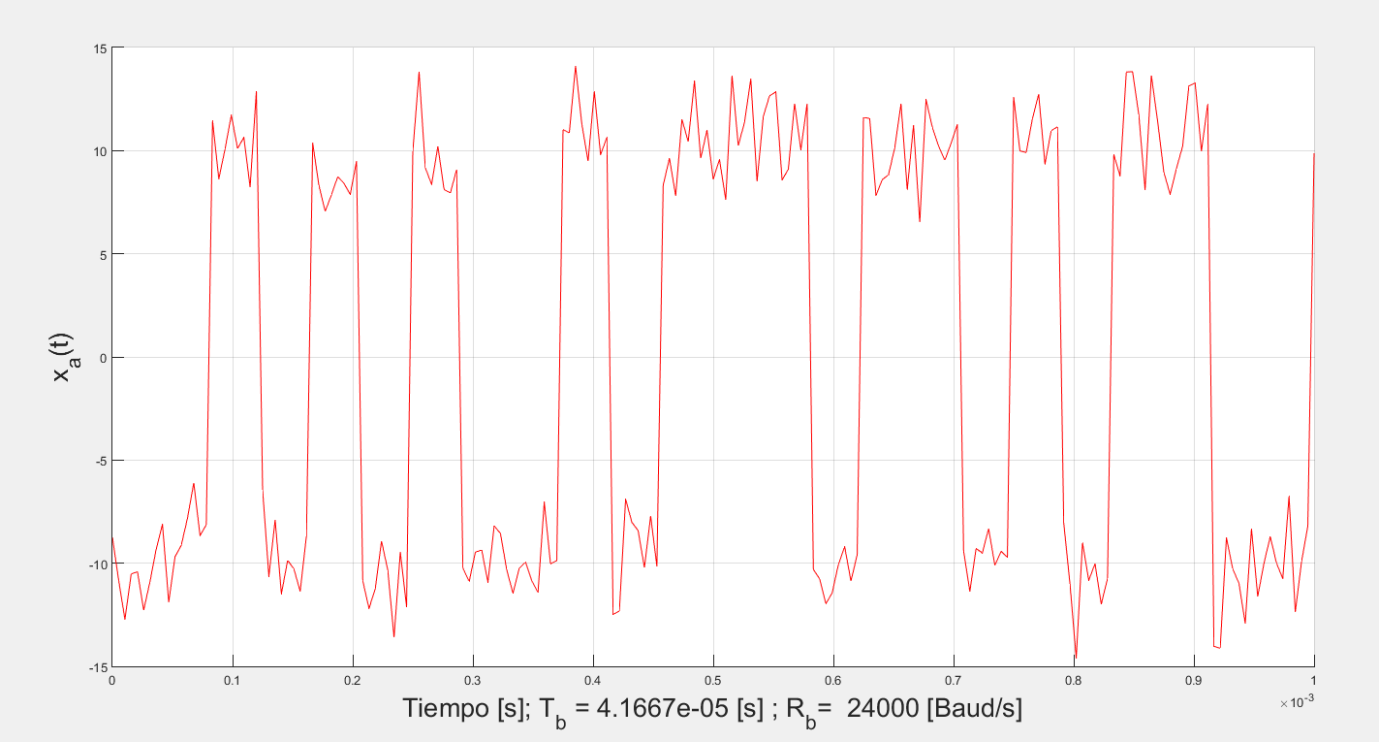
c) 500 mV



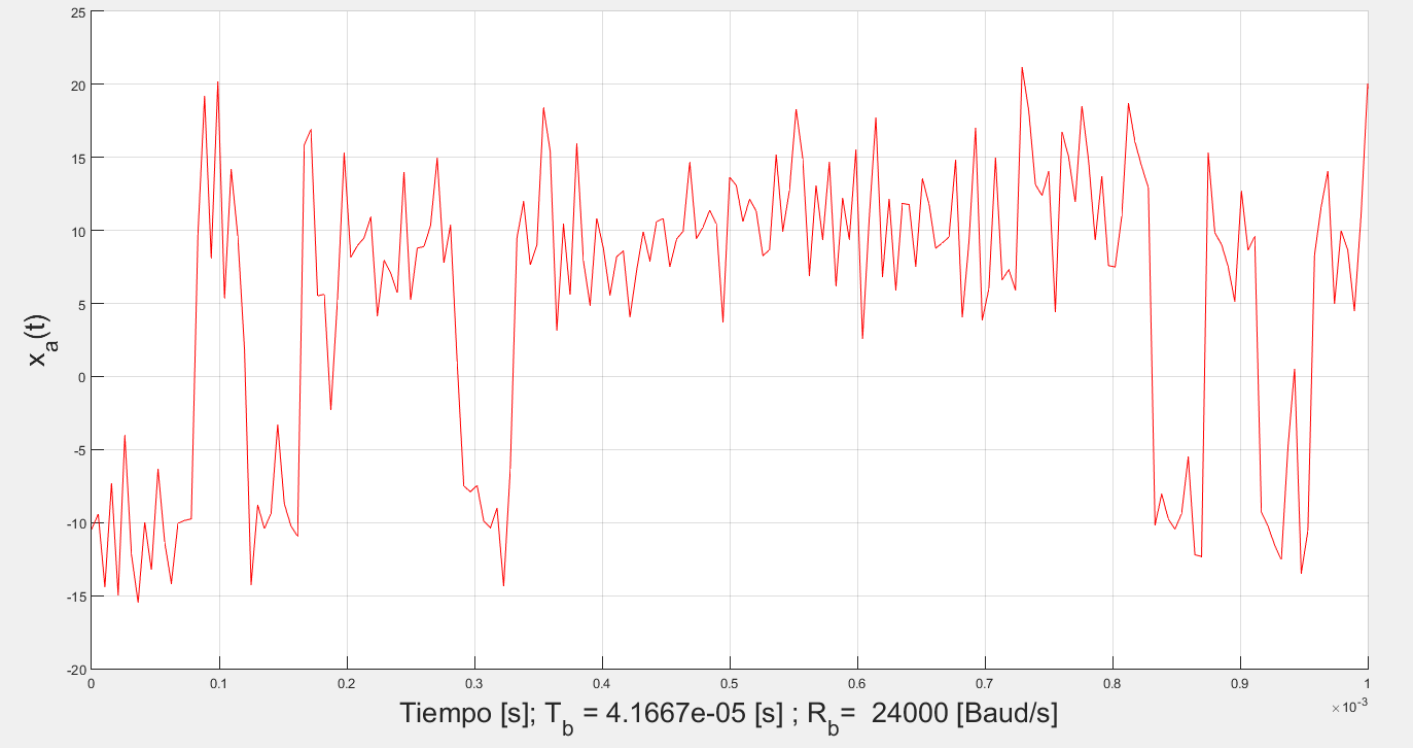
d) 1 V


15
10
5
0
-5
-10
-15
$x_a(t)$
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
$\times 10^{-3}$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]

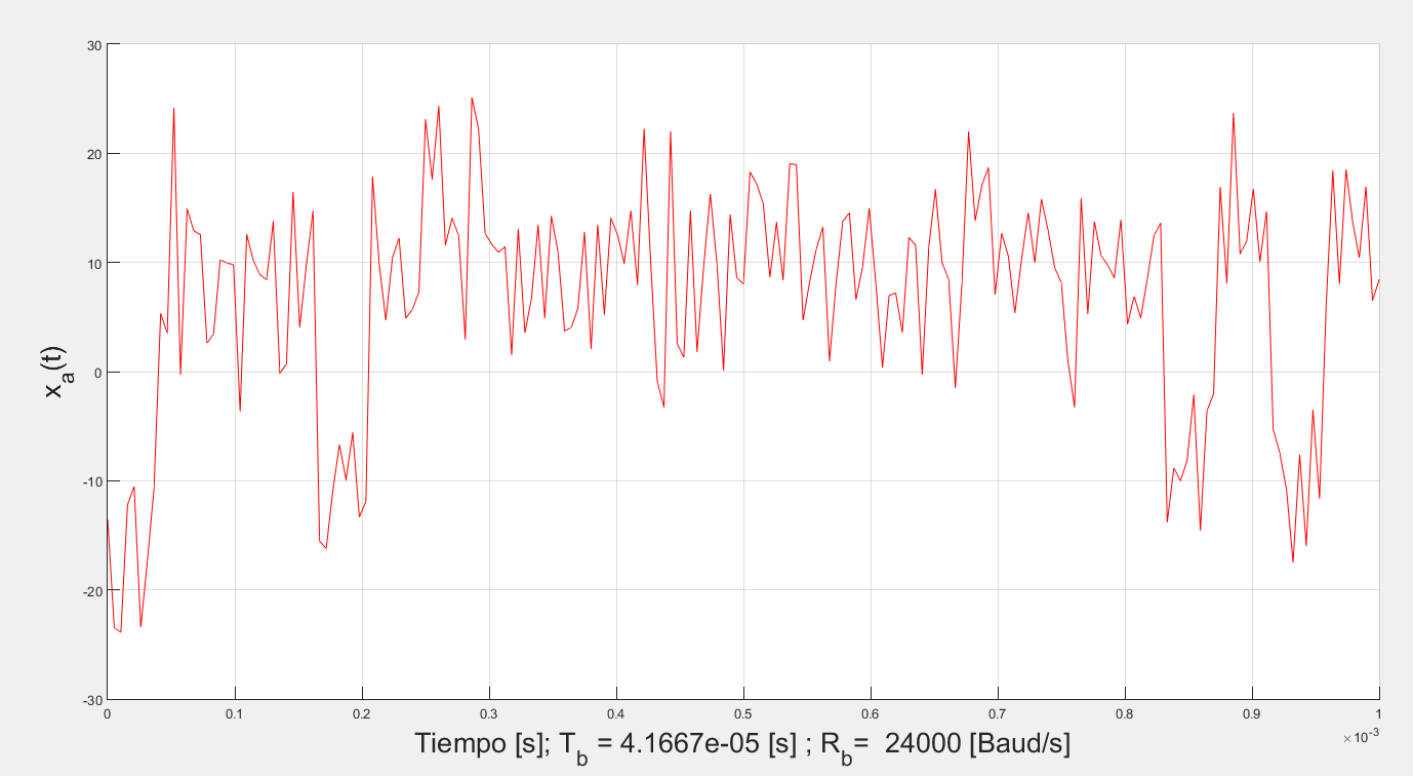
e) 2 V


15
10
5
0
-5
-10
-15
$x_a(t)$
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
$\times 10^{-3}$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]

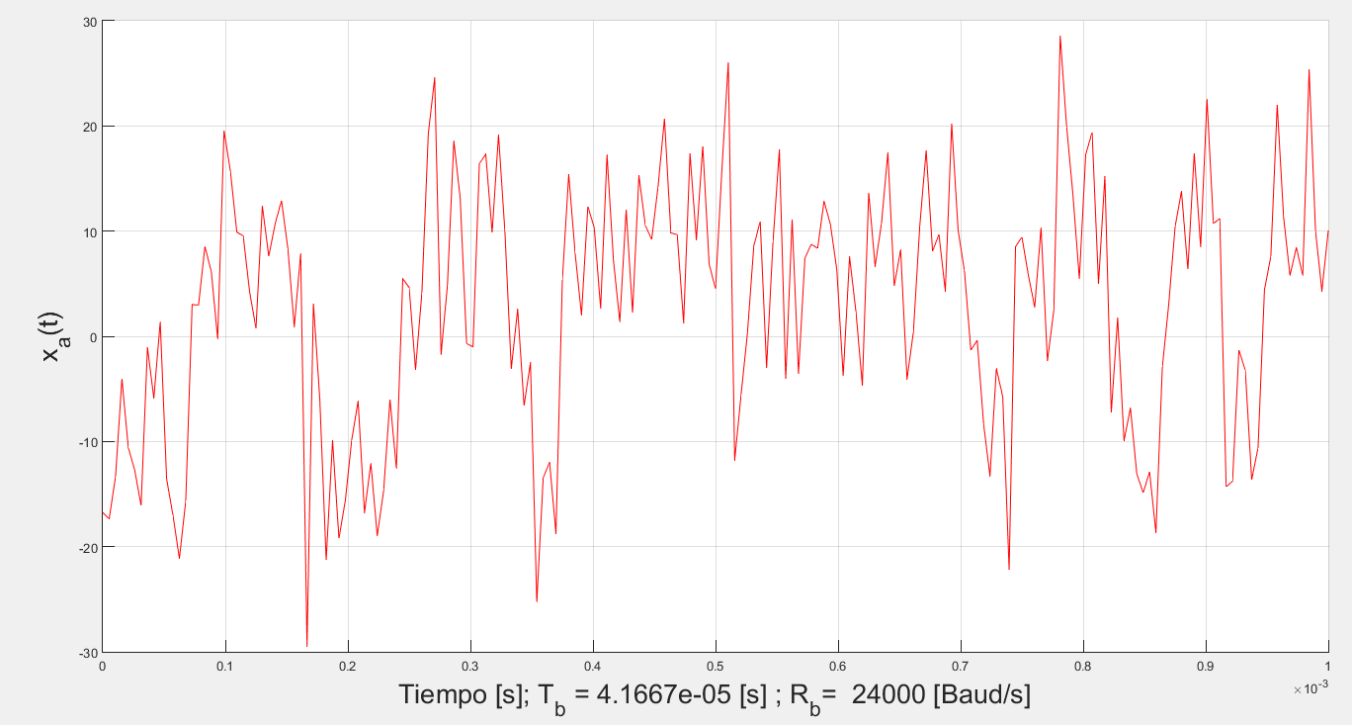
f) 4 V


25
20
15
10
5
0
-5
-10
-15
-20
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
$\times 10^{-3}$
$x_a(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]

g) 6 V


30
20
10
0
-10
-20
-30
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
$\times 10^{-3}$
$x_a(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]

h) 8 V

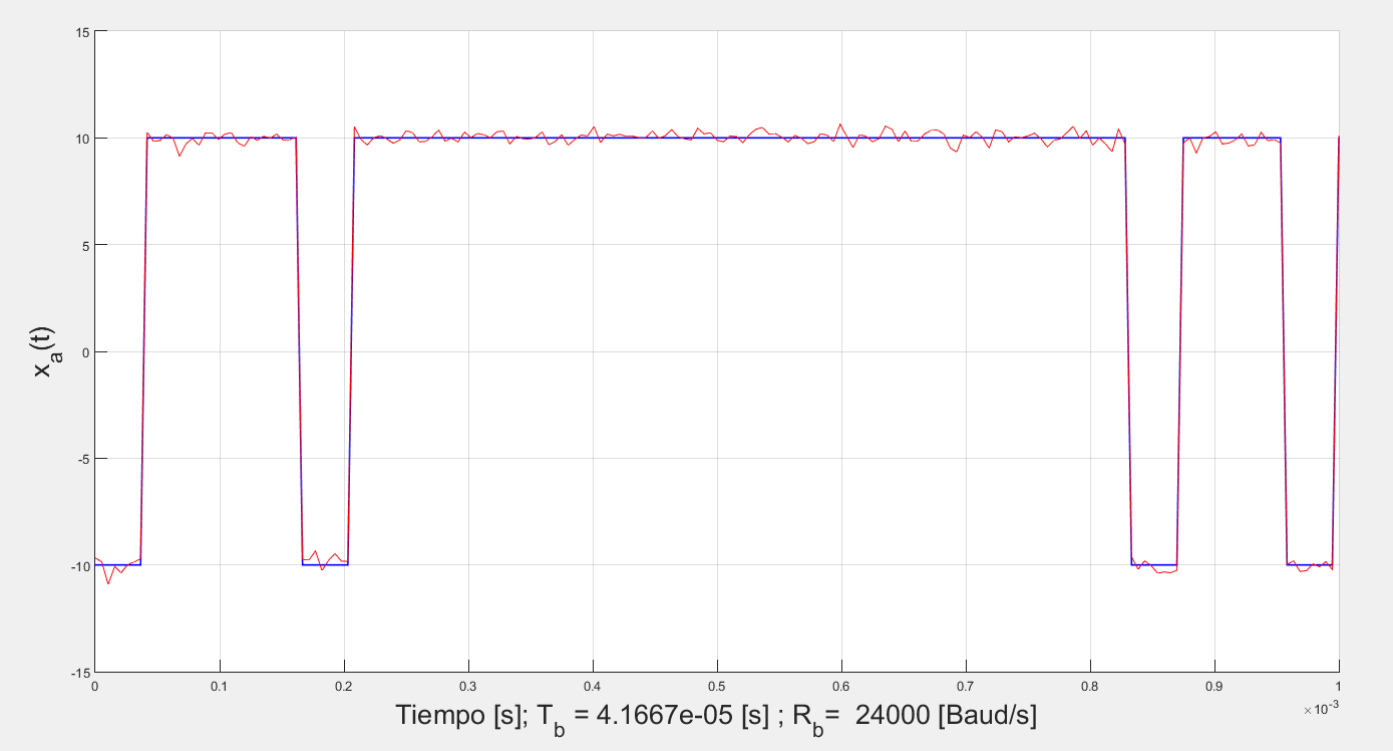


Graficar la señal binaria ruidosa para todos los casos, de 0 a1 ms, y de -10 a 10 Volts.

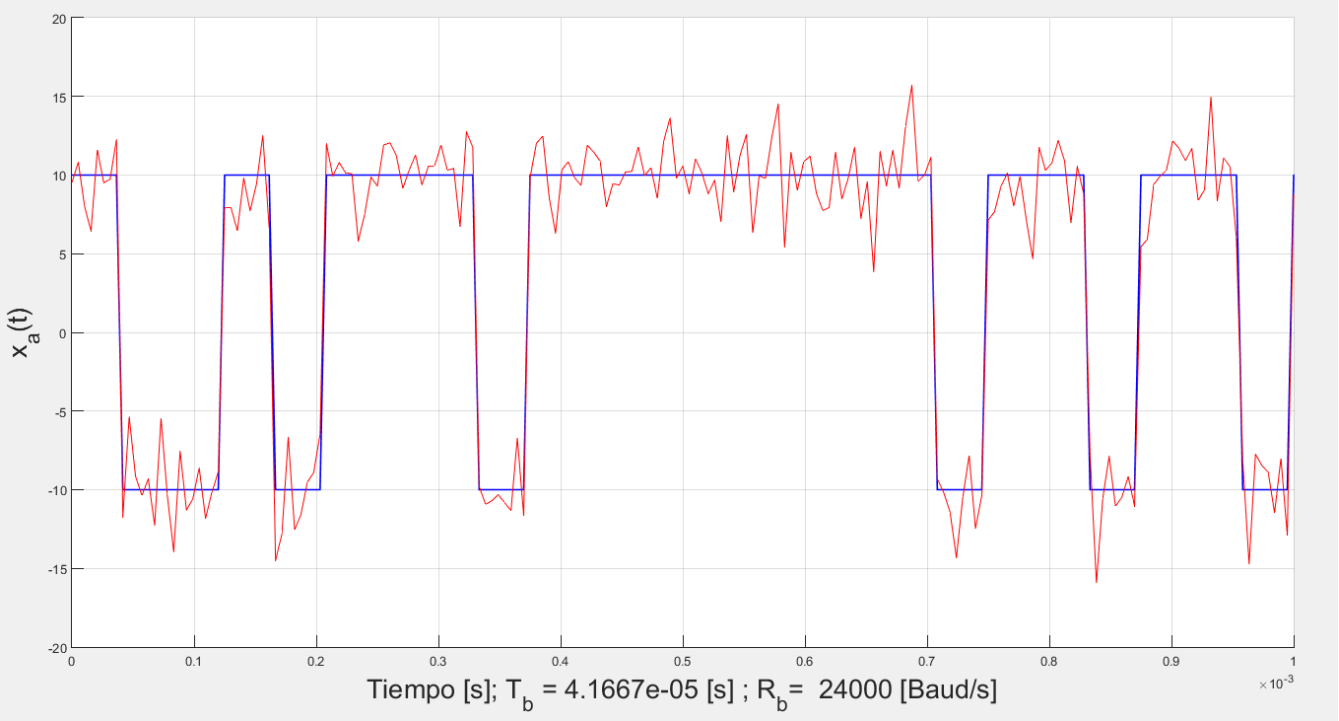
4) Recibir la señal binaria ruidosa con el receptor visto en clase. El receptor debe calcular la Tb con base en la Rb de la señal recibida.

Graficar la señal binaria después del receptor para los casos b), e) y h) de 0 a 1 ms, de -10 a 10 Volts. y compararla con la señal transmitida (superponer la Rx y la Tx con colores diferentes).

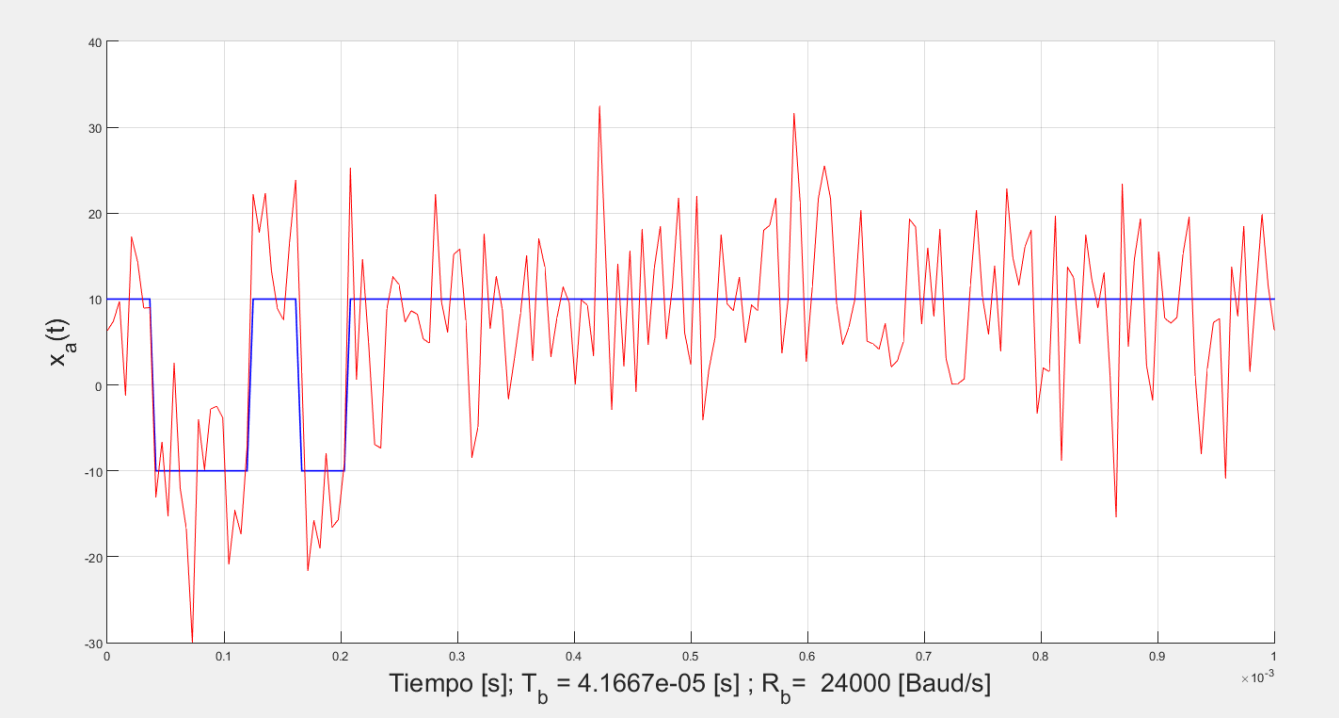
b)



e)

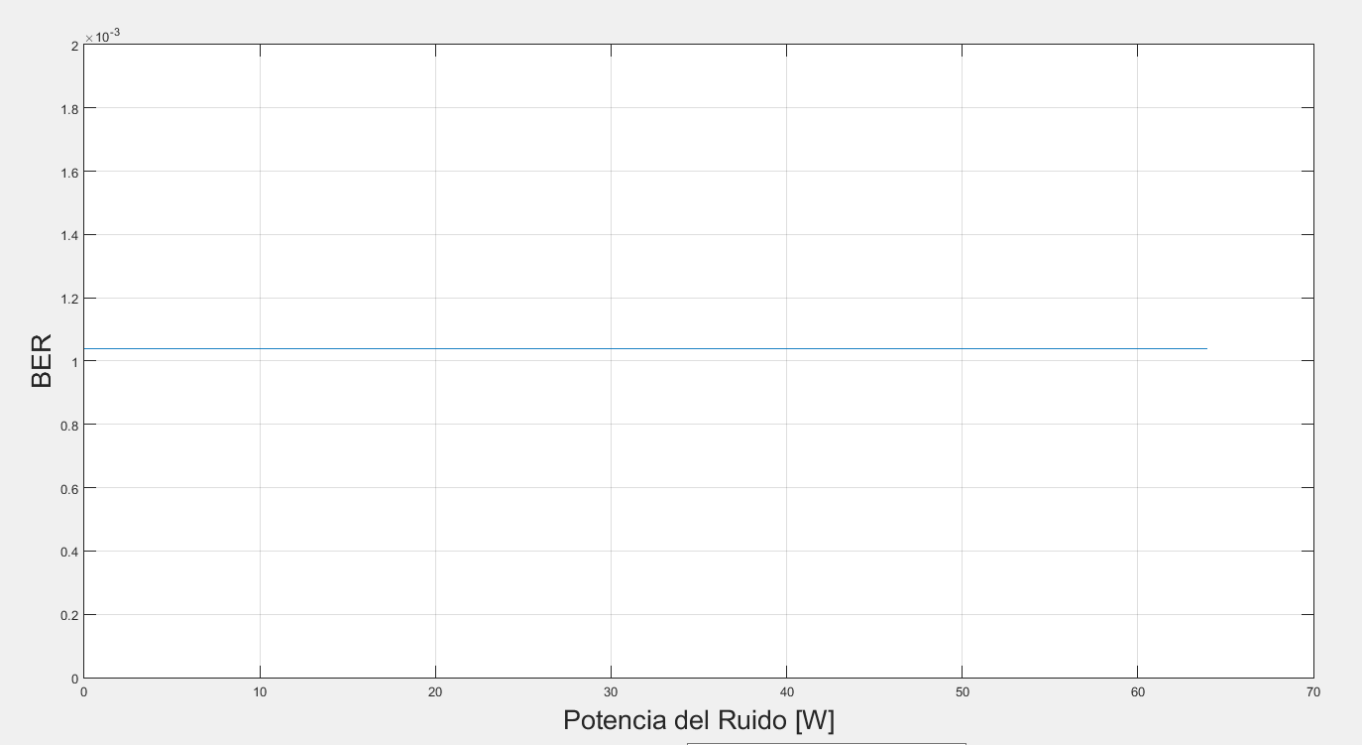

$x_a(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]
$\times 10^{-3}$

h)

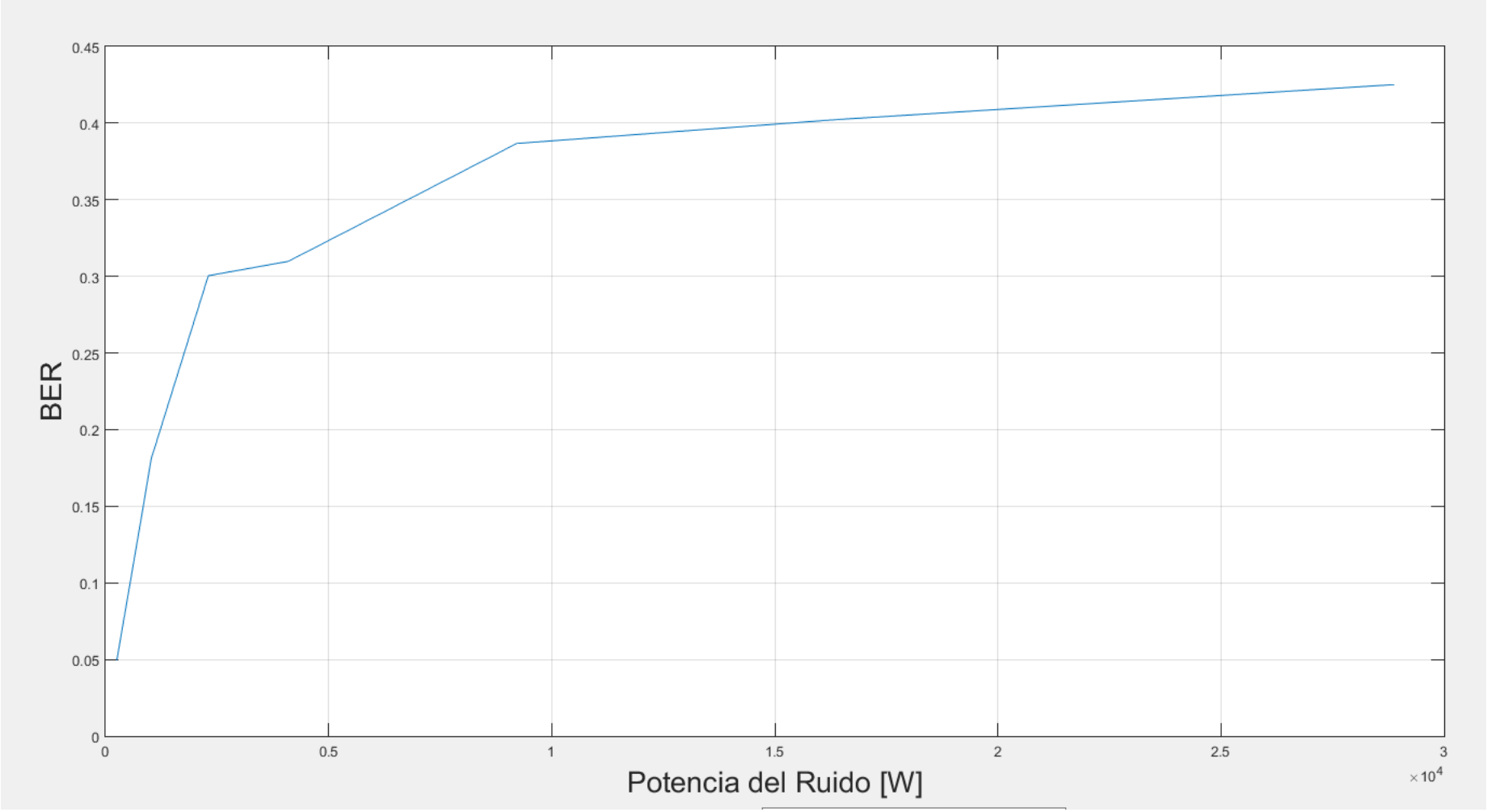

$x_a(t)$
Tiempo [s]; T_b = 4.1667e-05 [s] ; R_b= 24000 [Baud/s]
$\times 10^{-3}$

5) Para los casos a) al h) del punto (3), comparar la señal binaria transmitida con la señal recibida, con el fin de contar el número de bits que llegaron erróneos. Con base en el número de bits erróneos y el número de bits totales transmitidos calcular la BER para todos los casos y graficar BER vs varianza del ruido (en Watts).

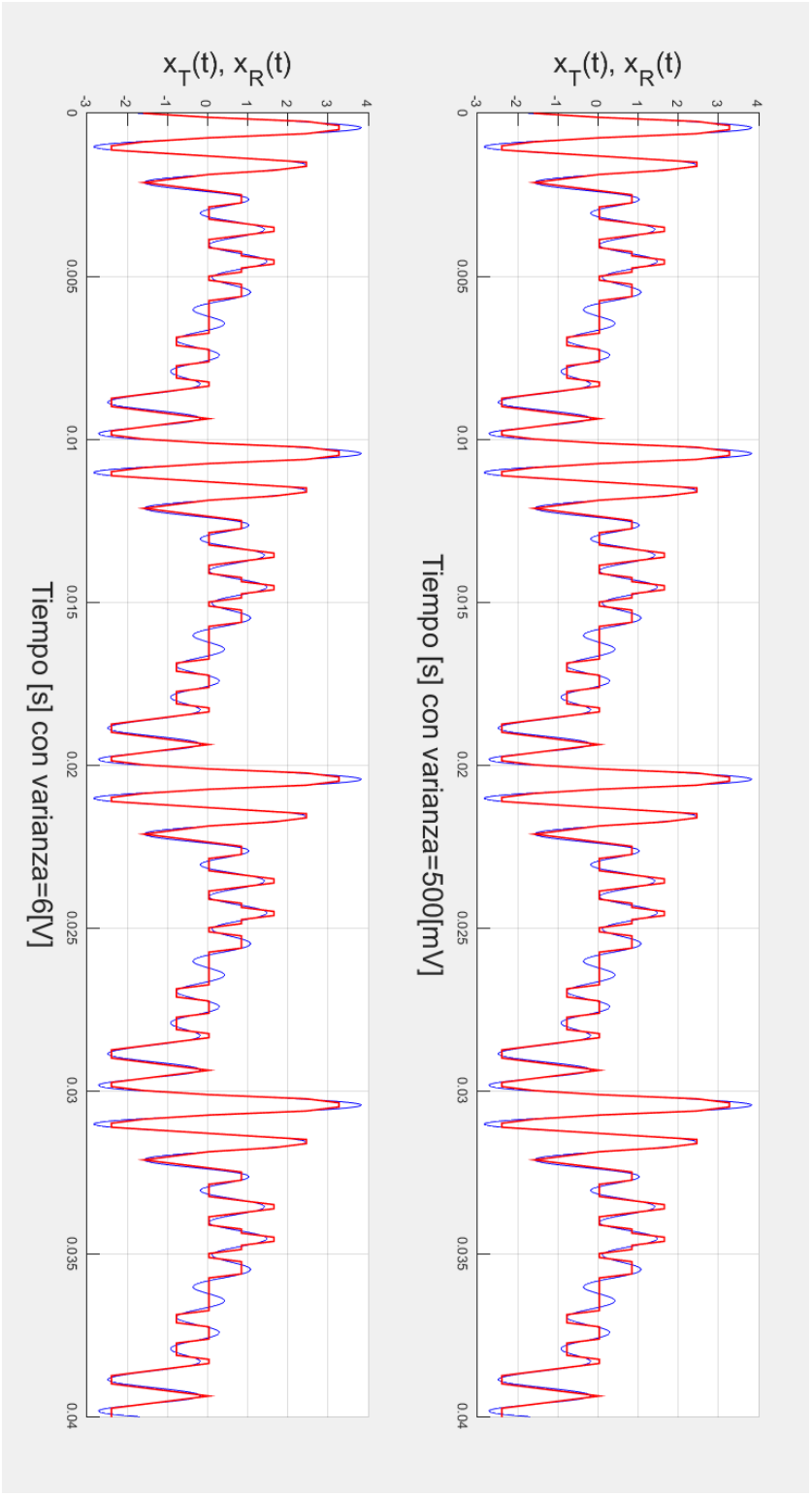
Con los valores dados de la varianza del ruido se obtiene una BER constante en casi todos los casos de un bit erróneo por los 962 enviados, BER= 0.001, como se muestra en la gráfica:



Sin embargo, al modificar los valores de la varianza del ruido a los siguientes: `[16 28 32 48 64 96 128 170]`, se obtiene una BER mucho más variable.


BER
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.5
1
1.5
2
2.5
3
Potencia del Ruido [W]
×10^4

6) Reconstruir la señal pseudo-analógica para los casos c) y g) del punto (3) y compararla con la señal pseudo-analógica original (superponer la Rx y la Tx con colores diferentes). Anotar un párrafo de observaciones.



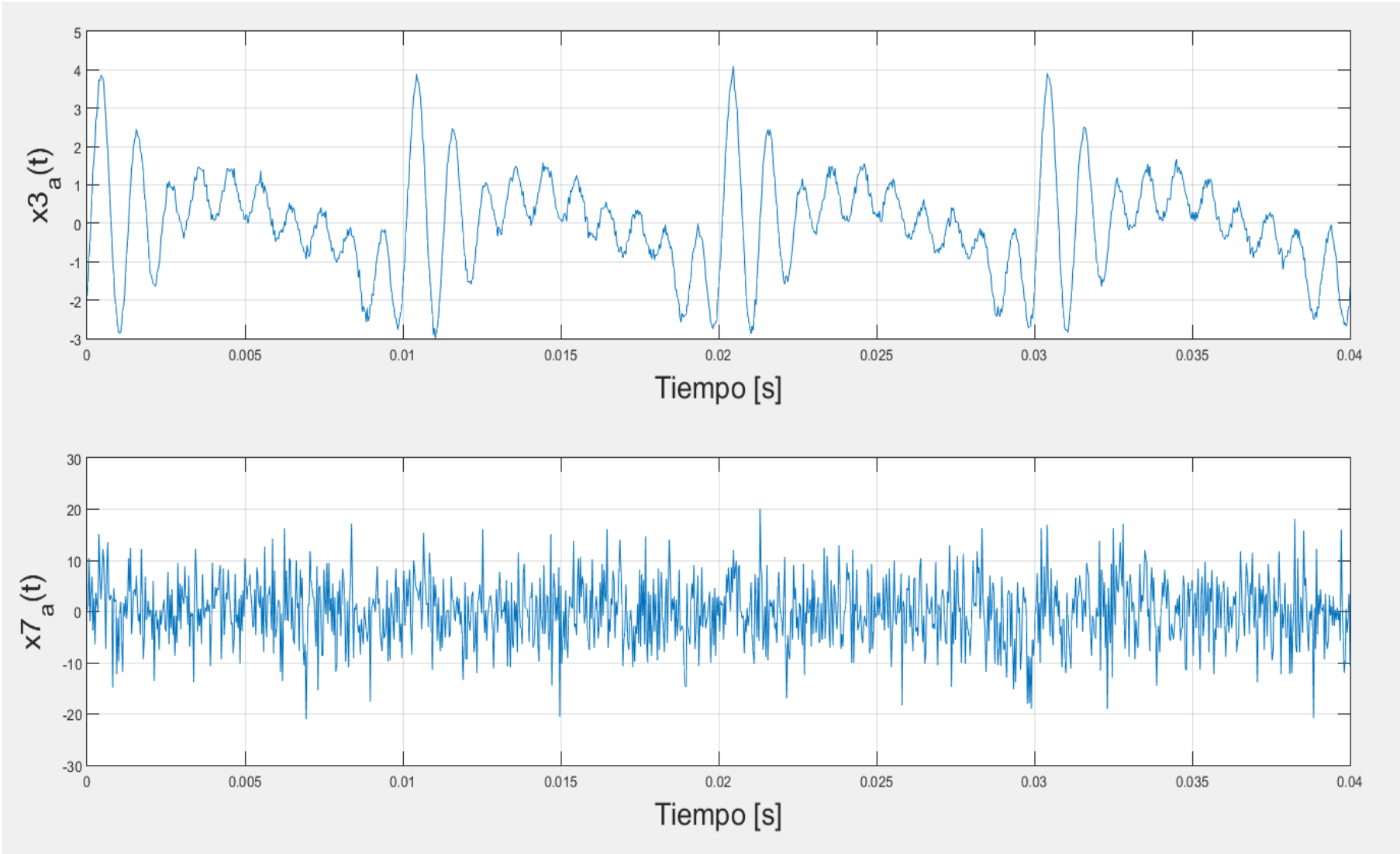
$X_T(t)$: Azul $X_R(t)$: Rojo

Como se puede observar ambas señales reconstruidas no presentan mayor diferencia puesto que el receptor se ve poco afectado por los niveles tan bajos de varianza del ruido, como ya se pudo observar anteriormente al calcular la BER. Así mismo se puede observar la existencia de un ruido de cuantización al comparar la señal analógica original con la señal reconstruida, ya que esta ultima, al estar forzada por los niveles a caer en uno de los 8, pierde algo de información y fidelidad, derivando en una deformación con respecto a la señal original, un ruido de cuantización.

7) Agregar a la señal pseudo-analógica ruido con la misma varianza de los casos c) y g) del punto (3), directamente, sin convertirla a señal digital. Comparar las señales pseudo-analógicas ruidosas, con las respectivas señales pseudo-analógicas reconstruidas en el punto 6. Anotar un párrafo de observaciones.

Como se puede observar al añadir dos señales de ruido con las mismas varianzas que en el punto anterior, pero esta vez directamente a la señal pseudo-analógica, representaría mandar directamente la señal por el canal ruidoso, sin ningún tipo de discretización, cuantización y posterior codificación , y por ello se observa que mientras que la primera señal ya se puede ver afectada por el ruido con varianza de 500 mV sin embargo aún se puede distinguir la forma principal de la señal y probablemente se pueda recibir, pero para la varianza de 6 V la señal queda prácticamente irreconocible y básicamente toda la información de la señal ya se perdió.

A diferencia del punto anterior donde se puede observar que la codificación y la utilización del receptor digital binario en banda base, provocan que sea mucho menos susceptible al ruido la información, con estos dos últimos puntos se puede observar muy claramente que el sistema provoca una transmisión de datos mucho mejor ante el ruido que simplemente pasar la señal a través del medio ruidoso.


x3 a (t)
5
4
3
2
1
0
-1
-2
-3
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
Tiempo [s]
x7 a (t)
30
20
10
0
-10
-20
-30
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
Tiempo [s]

Código:

```
close all;
clear all;
clc;

Amp=1; N=10;                          % Parámetros de la señal x_a(t), N =
Número de coeficientes de la Serie de Fourier
FLAG_PLOT=1;                          % Indica si graficar (1) o no (0)

f0=100;                               % Frecuencia fundamental de la señal
x_a(t)
T0=1/f0;                              % Período fundamental de la señal x_a(t)
Res=32;                               % Muestras por cada período fundamental
de la señal pseudo-analógica, les había dicho 8, pueden dejarla como 32
fs_mat=Res*N*f0;                      % Frecuencia de muestreo para generar la
señal pseudo-analógica (NO confundir con frecuencia de muestreo para
discretizarla)
Ts_mat=1/fs_mat;                      % Intervalos de muestreo para generar la
señal pseudo-analógica
t=0:Ts_mat:500*T0;                      % Vector de tiempo, de 0 a 2 períodos
(para graficar)
%valores de voltaje
v = [0.1 0.25 0.5 1 2 4 6 8];
%v = [16 28 32 48 64 96 128 170];
p = v.^2;


%Cálculo de los coeficientes de la serie de Fourier y reconstrucción de la
señal x(t)
a0=0;                                 % Componente de DC
x_a=a0;
A=rand(1,N);
Theta=pi*A;
for n=1:1:N
    x_temp=A(n)*cos(2*pi*f0*n*t-Theta(n));
    x_a=x_a+x_temp;
end

%% Discretizar la señal pseudo-analógica

S=8;
fs=S*N*f0;      % Frecuencia de muestreo = S veces la frecuencia máxima
(N*f0) de la señal pseudo-analógica
Ts=1/fs;        % Período de muestreo [s]
t_dis=0:Ts:4*T0;
for i=1 : 1 : size(t_dis, 2)
   x_dis(i)=x_a(1+(i-1)*Res/S);
end

%% Añadir ruido a la señal pseudo-analógica de 500mV y 6V
x3_a = x_a + wgn(1,length(x_a),p(1),'linear');
x7_a = x_a + wgn(1, length(x_a), p(7), 'linear');
```

```
%% Cuantizar la señal muestreada
x_dis_max = max(x_dis);
x_dis_min =  min(x_dis);
vpp=x_dis_max-x_dis_min;
N2=8; %Numero de niveles de cuantizacion
lvl=vpp/N2;
aux=0;
n_bits=log2(N2);
Tb = Ts/n_bits;
Rb = 1/Tb;
for i = 1 : 1 : size(t_dis,2)
    for n = 0 : 1 : N2
        if x_dis(i) >= x_dis_min+lvl*n
            if n==N2
                x_cuant(i) = x_dis_min-lvl/2+lvl*(n);
                aux=n-1;
                break;
            else
                x_cuant(i) = x_dis_min+lvl/2+lvl*n;
            end
        else
            aux=n-1;
            break;
        end
    end
    aux=de2bi(aux, 'left-msb');
    while length(aux)<n_bits
        aux=[0,aux];
    end
    if i==1
        x_bits = aux;
    else
        x_bits = [x_bits, aux];
    end
end


%% Codificar la señal cuantizada
t_bits=0:Tb:(length(x_bits)-1)*Tb;% Definiendo vector de duracion de bit
t_bits_res=0:Tb/8:(length(x_bits)-1)*Tb;%Definiendo vector de resolucion de
bit (8 muestras)
t_bits_res(end)= [];%Removiendo cadena vacia al final
%t_bits = Ts:Ts/n_bits:(length(x_bits)-1)*Ts/n_bits;
t_bits = [t_bits;t_bits]; %Concatenando por columna
aux = x_bits(end);
x_bits(end) = []; %Eliminando cadena vacia al final
x_bits_res=[x_bits ; x_bits]; % vector de  bits de resolucion; Concatenando
por columna 3 veces (8 muestras de resolucion)
x_bits_res=[x_bits_res ; x_bits_res];
x_bits_res=[x_bits_res ; x_bits_res];
x_bits_res = x_bits_res(:); % transformando en vector columna vector de bits
de resolucion
x_bits(end)= aux;
% amp
x_bits_nrzu = 5.*x_bits;%amplificando unos y ceros por 5 volts
%NRZ-b
```

```
for i = 1 : 1 : length(x_bits)
    if x_bits(i) == 1
        x_bits_nrzb(i)=10;
    else
        x_bits_nrzb(i)=-10;
    end
end
x_bits_res_nrzu= 5.*x_bits_res;% NRZ unipolar con resolucion para graficar
%NRZ-b con resolucion
for i = 1 : 1 : length(x_bits_res)
    if x_bits_res(i) == 1
        x_bits_res_nrzb(i)=10;
    else
        x_bits_res_nrzb(i)=-10;
    end
end
x_bits_res_nrzb=x_bits_res_nrzb(:);
t_bits = t_bits(:);
x_bits = x_bits(:);

%% Introduciendo ruido NRZ-Unipolar y NRZ-Bipolar

y2 = wgn(1,length(x_bits_res_nrzu),p(2),'linear');y2 = y2(:);
sruidosa = x_bits_res_nrzu + y2;
sruidosa2 = x_bits_res_nrzb + y2;

y1 = wgn(1,length(x_bits_res_nrzu),p(1),'linear');y1 = y1(:);sruidosa1=
x_bits_res_nrzb + y1;
y3 = wgn(1,length(x_bits_res_nrzu),p(3),'linear');y3 = y3(:);sruidosa3=
x_bits_res_nrzb + y3;
y4 = wgn(1,length(x_bits_res_nrzu),p(4),'linear');y4 = y4(:);sruidosa4=
x_bits_res_nrzb + y4;
y5 = wgn(1,length(x_bits_res_nrzu),p(5),'linear');y5 = y5(:);sruidosa5=
x_bits_res_nrzb + y5;
y6 = wgn(1,length(x_bits_res_nrzu),p(6),'linear');y6 = y6(:);sruidosa6=
x_bits_res_nrzb + y6;
y7 = wgn(1,length(x_bits_res_nrzu),p(7),'linear');y7 = y7(:);sruidosa7=
x_bits_res_nrzb + y7;
y8 = wgn(1,length(x_bits_res_nrzu),p(8),'linear');y8 = y8(:);sruidosa8=
x_bits_res_nrzb + y8;

%% Receptor
% Promediar
ac2 = 0;acz2 =  ac2;ac = ac2;acz = ac2;
ac1 = 0; acz1= 0;ac3 = 0; acz3= 0;ac4 = 0; acz4= 0;ac5 = 0; acz5= 0;ac6 = 0;
acz6= 0; ac7 = 0; acz7= 0; ac8 = 0; acz8= 0;
Tb = 1/Rb;
ap = 0;
gamma = (5^2)*Tb/2;
for i = 1 : 1 : length(x_bits)
    for n = 1 : 1 : 8
        ac = sruidosa(n+ap) + ac;
        acz = sruidosa(n+ap)*-1 + acz;
        ac1 = sruidosa1(n+ap) + ac1;
        acz1 = sruidosa1(n+ap)*-1 + acz1;
        ac2 = sruidosa2(n+ap) + ac2;
```

```
        acz2 = sruidosa2(n+ap)*-1 + acz2;
        ac3 = sruidosa3(n+ap) + ac3;
        acz3 = sruidosa3(n+ap)*-1 + acz3;
        ac4 = sruidosa4(n+ap) + ac4;
        acz4 = sruidosa4(n+ap)*-1 + acz4;
        ac5 = sruidosa5(n+ap) + ac5;
        acz5 = sruidosa5(n+ap)*-1 + acz5;
        ac6 = sruidosa6(n+ap) + ac6;
        acz6 = sruidosa6(n+ap)*-1 + acz6;
        ac7 = sruidosa7(n+ap) + ac7;
        acz7 = sruidosa7(n+ap)*-1 + acz7;
        ac8 = sruidosa8(n+ap) + ac8;
        acz8 = sruidosa8(n+ap)*-1 + acz8;
    end
    ap=ap+8;
    prom = ac/8;promz = acz/8;
    prom1 = ac1/8;promz1 =acz1/8;
    prom2 = ac2/8;promz2 = acz2/8;
    prom3 = ac3/8;promz3 =acz3/8;
    prom4 = ac4/8;promz4 =acz4/8;
    prom5 = ac5/8;promz5 =acz5/8;
    prom6 = ac6/8;promz6 =acz6/8;
    prom7 = ac7/8;promz7 =acz7/8;
    prom8 = ac8/8;promz8 =acz8/8;
    ac=0;acz=0;ac1 = 0; acz1= 0;ac2=0;acz2=0;ac3 = 0; acz3= 0;ac4 = 0; acz4=
0;ac5 = 0; acz5= 0;
    ac6 = 0; acz6= 0; ac7 = 0; acz7= 0; ac8 = 0; acz8= 0;
    z(i)=prom-promz;
    z1(i)= prom1-promz1;
    z2(i)= prom2-promz2;
    z3(i)= prom3-promz3;
    z4(i)= prom4-promz4;
    z5(i)= prom5-promz5;
    z6(i)= prom6-promz6;
    z7(i)= prom7-promz7;
    z8(i)= prom8-promz8;
end

%% Comparando con Gamma y reconstruyendo
for n =1 : 1 : length(z)
    if z(n) > gamma
        z(n)=5;
    else
        z(n)=0;
    end

    if z1(n) > 0
        z1(n) = 10;
    else
        z1(n) = -10;
    end

    if z2(n) > 0
        z2(n) = 10;
    else
        z2(n) = -10;
```

```
    end

    if z3(n) > 0
        z3(n) = 10;
    else
        z3(n) = -10;
    end

    if z4(n) > 0
        z4(n) = 10;
    else
        z4(n) = -10;
    end

    if z5(n) > 0
        z5(n) = 10;
    else
        z5(n) = -10;
    end

    if z6(n) > 0
        z6(n) = 10;
    else
        z6(n) = -10;
    end

    if z7(n) > 0
        z7(n) = 10;
    else
        z7(n) = -10;
    end

    if z8(n) > 0
        z8(n) = 10;
    else
        z8(n) = -10;
    end
end

%% Obtener la BER
errores=0;errores1=0;errores2=0;errores3=0;errores4=0;errores5=0;errores6=0;e
rrores7=0;errores8=0;
for n = 1: 1 : length(x_bits)
    if x_bits_nrzu(n)~= z(n)
        errores = errores + 1;
    end
    if x_bits_nrzb(n)~= z1(n)
        errores1 = errores1 + 1;
    end
    if x_bits_nrzb(n)~= z2(n)
        errores2 = errores2 + 1;
    end
    if x_bits_nrzb(n)~= z3(n)
        errores3 = errores3 + 1;
    end
```

```
    if x_bits_nrzb(n)~= z4(n)
        errores4 = errores4 + 1;
    end
    if x_bits_nrzb(n)~= z5(n)
        errores5 = errores5 + 1;
    end
    if x_bits_nrzb(n)~= z6(n)
        errores6 = errores6 + 1;
    end
    if x_bits_nrzb(n)~= z7(n)
        errores7 = errores7 + 1;
    end
    if x_bits_nrzb(n)~= z8(n)
        errores8 = errores8 + 1;
    end
end
BER1 = errores/length(x_bits);
BER =
[errores1/length(x_bits),errores2/length(x_bits),errores3/length(x_bits),erro
res4/length(x_bits),errores5/length(x_bits),
errores6/length(x_bits),errores7/length(x_bits), errores8/length(x_bits)];

%%Reconstruir la señal pseudoanalógica para 500mV y 6V (z3) y (z7)
ap=0;
for i = 1 : 1 : length(z3)
    if z3(i)==10
        z3(i)=1;
    else
        z3(i)=0;
    end

    if z7(i)==10
        z7(i)=1;
    else
        z7(i)=0;
    end
end

ap=0;
for i = 1 : 1 : length(x_cuant)
    for n = 1 : 1 : n_bits
        if ap+n<=length(x_bits)
            aux(n)=z3(ap+n);
            aux2(n)=z7(ap+n);
        end
    end
    ap=ap+3;
    x3_cuant(i)=x_dis_min + lvl/2 + lvl*bi2de(aux,'left-msb');
    x7_cuant(i)=  x_dis_min + lvl/2 + lvl*bi2de(aux2,'left-msb');
end

%% Graficas
for i = 1: 1: 3
    z= [z;z];
    z2= [z2;z2];
end
```

```
z = z(:);
z2 = z2(:);
if FLAG_PLOT == 1

    figure
    subplot(2,1,1);
    plot(t, x3_a)
    grid on; xlabel('Tiempo [s]','fontsize',20);
ylabel('x3_a(t)','fontsize',20);
    xlim([0 4*T0]);

    subplot(2,1,2);
    plot(t, x7_a);
    grid on; xlabel('Tiempo [s]','fontsize',20);
ylabel('x7_a(t)','fontsize',20);
    xlim([0 4*T0]);

    figure
    plot(t,x_a);
    grid on; xlabel('Tiempo [s]','fontsize',20);
ylabel('x_a(t)','fontsize',20);
    xlim([0 4*T0]);

    figure
    subplot(3,1,1);
    plot(t,x_a);                % Gráfica de la señal x_a(t)
    grid on; xlabel('Tiempo [s]','fontsize',20);
ylabel('x_a(t)','fontsize',20);
    xlim([0 4*T0]);
    hold on                     % Sirve para graficar dos curvas en la misma
gráfica. Las señales cuantizadas de tarea las quiero así traslapadas con la
señal original
    stem(t_dis,x_dis);          % Gráfica de la señal muestreada
    stem(t_dis, x_cuant);
    hold off
    subplot(3,1,2);
    plot(t_bits_res, x_bits_res_nrzu, 'b');
    grid on; xlabel(['Tiempo [s]; T_b = ' (num2str(Tb)) ' [s] ; R_b=  '
num2str(Rb) ' [Baud/s]'], 'fontsize', 20); ylabel('x_b(t)','fontsize',20);
    xlim([0 T0/10]); ylim([-0.3, 6]);
    subplot(3,1,3);
    plot(t_bits_res, x_bits_res_nrzb, 'r');
    grid on; xlabel(['Tiempo [s]; T_b = ' (num2str(Tb)) ' [s] ; R_b=  '
num2str(Rb) ' [Baud/s]'], 'fontsize', 20); ylabel('x_b(t)','fontsize',20);
    xlim([0 T0/10]); ylim([-11 12]);

    figure
    hold on
    %plot(t_bits_res, z2, 'b');
    plot(t_bits_res, sruidosa2, 'r');
    grid on; xlabel(['Tiempo [s]; T_b = ' (num2str(Tb)) ' [s] ; R_b=  '
num2str(Rb) ' [Baud/s]'], 'fontsize', 20); ylabel('x_a(t)','fontsize',20);
    xlim([0 T0/10]);
    hold off

    figure
```

```
    hold on
    plot(t_bits_res, z2, 'b', 'Linewidth', 1.2);
    plot(t_bits_res, sruidosa2, 'r');
    grid on; xlabel(['Tiempo [s]; T_b = ' (num2str(Tb)) ' [s] ; R_b=  '
num2str(Rb) ' [Baud/s]'], 'fontsize', 20); ylabel('x_a(t)','fontsize',20);
    xlim([0 T0/10]);
    hold off

    figure
    plot(p,BER);
    grid on; xlabel('Potencia del Ruido [W]','fontsize',20);
ylabel('BER','fontsize',20);

    figure
    subplot(2,1,1);
    grid on; xlabel('Tiempo [s] con varianza=500[mV]','fontsize',20);
ylabel('x_T(t), x_R(t)','fontsize',20);
    xlim([0 4*T0]);
    hold on
    plot(t, x_a, 'b');
    plot(t_dis, x3_cuant, 'r', 'LineWidth', 1.2);
    hold off

    subplot(2,1,2);
    grid on; xlabel('Tiempo [s] con varianza=6[V]','fontsize',20);
ylabel('x_T(t), x_R(t)','fontsize',20);
    xlim([0 4*T0]);
    hold on
    plot(t, x_a, 'b');
    plot(t_dis, x7_cuant, 'r', 'LineWidth', 1.2);
    hold off
end
```