Table of Contents


```
clear all;
close all;
clc;
```

Données

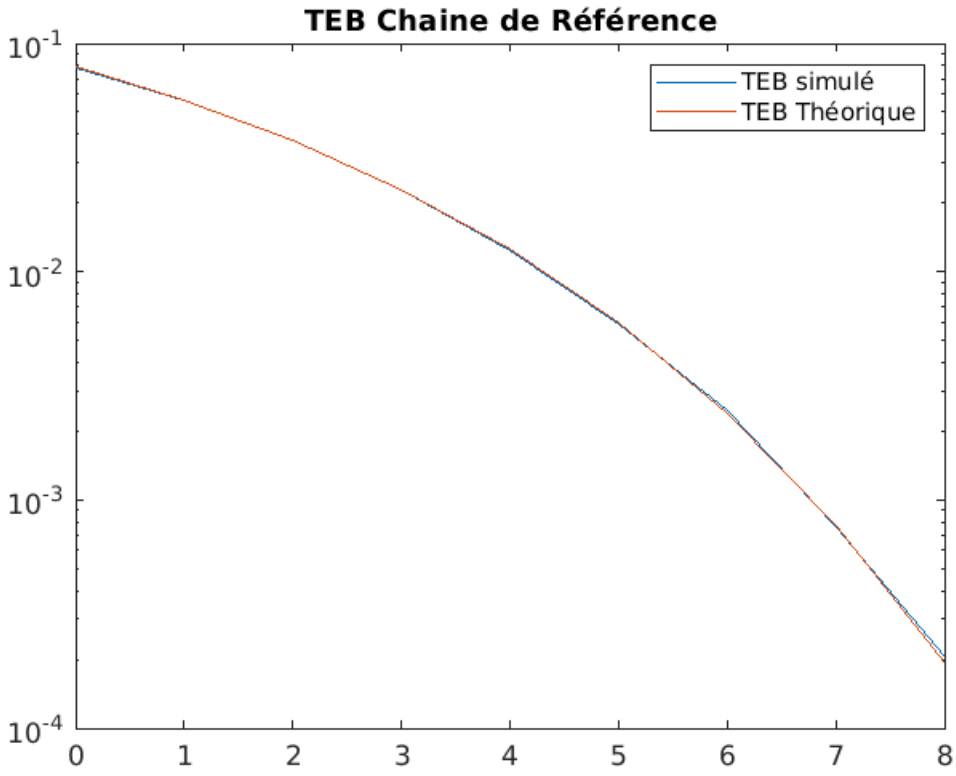
```
Rb = 6000; % Débit Binaire
Fe = 24000; % Fréquence d'échantillonage
Te = 1/Fe;
eb_N0_db = [0:8];
Donnees = randi([0 1], 1, 10000);
% Données Première Chaine
M1 = 2; % M = 2^n (n=1)
Rs1 = Rb/log2(M1); % Débit Symbole
Ns1 = floor(Fe/Rs1);
Ts1 = Ns1*Te;

% Donnée Deuxième Chaine
M2 = 4; % M = 2^n (n=2)
Rs2 = Rb/log2(M2); % Débit symbole
Ns2 = floor(Fe/Rs2);
Ts2 = Ns2*Te;
```

Étude Chaine de Référence

```
mapping1 = Donnees - (Donnees == 0);
h1 = ones(1, Ns1); % Filtre
Kronr1 = kron(mapping1, [1 zeros(1, Ns1-1)]); % Positionnement des
 zéros entre les ak
signalfiltre1 = filter(h1, 1, Kronr1); % Filtrage avec filtre
Px = mean(abs(signalfiltre1).^2);
hrr = fliplr(h1); % Filtre de réception
% Implantation de la chaine sans bruit
z_sbr = filter(hrr, 1, signalfiltre1); % Signal filtré sans bruit
z_ech_sbr = z_sbr(Ns1:Ns1:end); % Signal échantilloné sans bruit
dec_sbr = sign(z_ech_sbr);
dem_sbr = (dec_sbr + 1)/2;
test_sbr = dem_sbr - Donnees;
TEB_sbr = length(find(test_sbr~=0))/length(Donnees); % Taux d'Erreur
 Binaire
% Implantation de la chaine avec bruit
```

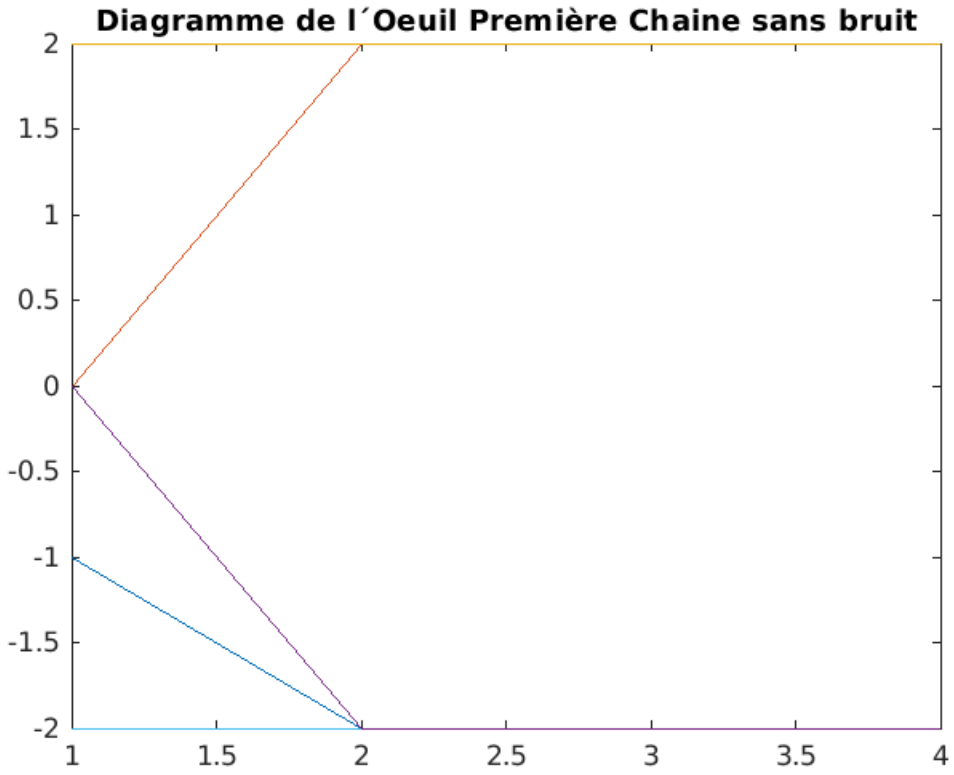
```
TEB_bruit_r = zeros(1, length(eb_N0_db)); % Taux d'Erreur Binaire
tmpr = 0;
Precision = 100;
for k = 1 : Precision % Précision
    Donnees = randi([0 1], 1, 10000);
    for i = 1 : length(eb_N0_db)
        mapping1 = Donnees - (Donnees == 0);
        Kronr1 = kron(mapping1, [1 zeros(1, Ns1-1)]); % Positionnement
 des zéros entre les ak
        signalfiltre1 = filter(h1, 1, Kronr1); % Filtrage avec filtre
        Px = mean(abs(signalfiltre1).^2);
        sigma2 = Px*Ns1/(2*10^(eb_N0_db(i)/10));
        bruit1 = sqrt(sigma2)*randn(1, length(signalfiltre1));
        Zr = signalfiltre1 + bruit1;
        zr = filter(hrr, 1, Zr); % Signal filtré par le filtre de
 réception
        %subplot(3,3,i);
        %plot(reshape(zr,Ns1,length(zr)/Ns1));
        z_echr = zr(Ns1:Ns1:end); % Échantillonage
        decr = sign(z_echr); % Décisions
        demr = (decr + 1)/2; % Démapping
        testr = demr - Donnees;
        TEB_bruit_r(i) = length(find(testr~=0))/length(Donnees);
    end
    tmpr = tmpr + TEB_bruit_r;
end
TEB_bruit_r = tmpr/Precision;
figure();
semilogy(eb_N0_db, TEB_bruit_r, 'DisplayName', 'TEB simulé');
hold on
TEB_th = 1 - normcdf(sqrt(2*10.^(eb_N0_db/10)));
semilogy(eb_N0_db, TEB_th, 'DisplayName', 'TEB Théorique');
legend;
title('TEB Chaine de Référence');
hold off
```

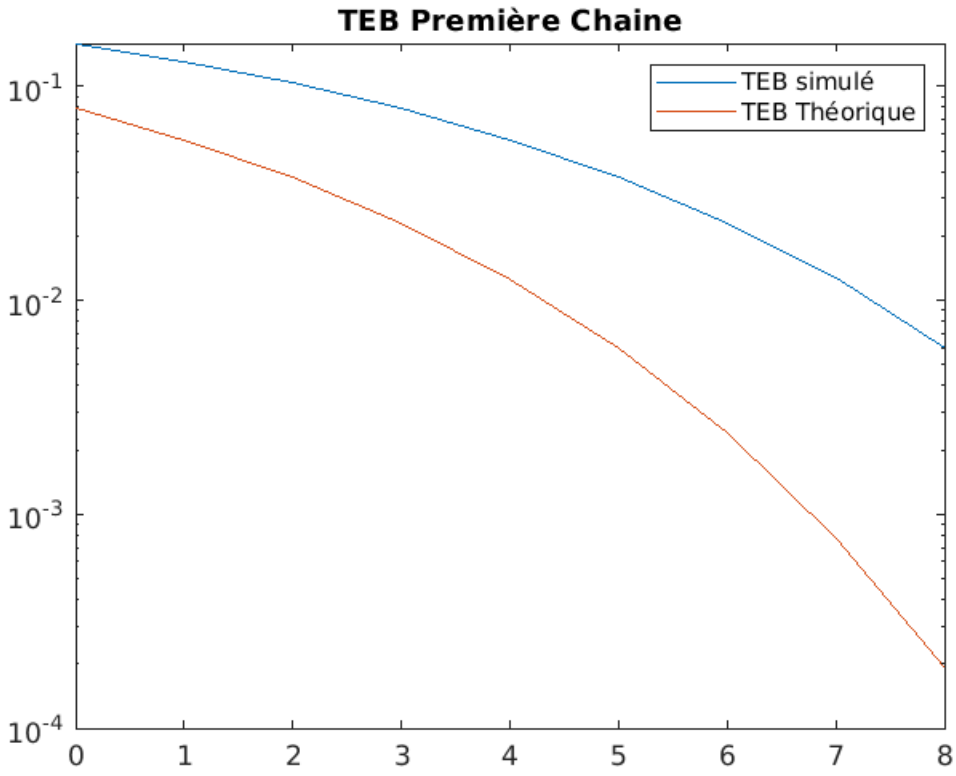


Étude Première Chaine

```
hr1 = [ones(1, Ns1/2), zeros(1,Ns1/2)];
% Implantation de la chaine sans bruit
z_sb1 = filter(hr1, 1, signalfiltre1);
figure,plot(reshape(z_sb1,Ns1,length(z_sb1)/Ns1));
title('Diagramme de l´Oeuil Première Chaine sans bruit');
z_ech_sb1 = z_sb1(Ns1:Ns1:end);
dec_sb1 = sign(z_ech_sb1);
dem_sb1 = (dec_sb1 + 1)/2;
test_sb1 = dem_sb1 - Donnees;
TEB_sb1 = length(find(test_sb1~=0))/length(Donnees);
% Implantation de la chaine avec bruit
tmp1 = 0;
for k = 1 : Precision
    Donnees = randi([0 1], 1, 10000);
    for i = 1 : length(eb_N0_db)
        mapping1 = Donnees - (Donnees == 0);
        Kronr1 = kron(mapping1, [1 zeros(1, Ns1-1)]); % Positionnement
 des zéros entre les ak
        signalfiltre1 = filter(h1, 1, Kronr1); % Filtrage avec filtre
        Px = mean(abs(signalfiltre1).^2);
        sigma2 = Px*Ns1/(2*10^(eb_N0_db(i)/10));
        bruit1 = sqrt(sigma2)*randn(1, length(signalfiltre1));
        Z1 = signalfiltre1 + bruit1;
```

```
        z1 = filter(hr1, 1, Z1); % Signal filtré
        %subplot(3,3,i);
        %plot(reshape(z1,Ns1,length(z1)/Ns1));
        z_ech1 = z1(Ns1:Ns1:end); % Échantillonage
        dec1 = sign(z_ech1); % Décision
        dem1 = (dec1 + 1)/2; % Démapping
        test1 = dem1 - Donnees;
        TEB_bruit_1(i) = length(find(test1~=0))/length(Donnees);
    end
    tmp1 = tmp1 + TEB_bruit_1;
end
TEB_bruit_1 = tmp1/Precision;
figure();
semilogy(eb_N0_db, TEB_bruit_1, 'DisplayName', 'TEB simulé');
hold on
semilogy(eb_N0_db, TEB_th, 'DisplayName', 'TEB Théorique');
legend;
title('TEB Première Chaine');
hold off
```





Étude Deuxième Chaine

Mapping

```
mapping2 = (2*bi2de(reshape(Donnees, 2, length(Donnees)/2).') -
 3).'; % Du binaire au decimal
% Suréchantillonage
h2 = ones(1,Ns2); % Filtre
Kronr2 = kron(mapping2, [1 zeros(1, Ns2-1)]); % Positionnement des
 zéros entre les ak
signalfiltre2 = filter(h2,1,Kronr2); % Filtrage
% Filtrage de réception
hr2 = fliplr(h2); % filtre rectangulaire

% Implantation de la chaine sans bruit
z_sb2 = filter(hr2, 1, signalfiltre2); % signal filtré par le filtre
 réctangulaire
% Echantillonage
z_ech_sb2 = z_sb2(Ns2:Ns2:end); % échantillonage du rectangulaire
% Decision
cond_sb1 = find(z_ech_sb2 > 2*Ns2);
cond_sb2 = find(z_ech_sb2 >= 0 & z_ech_sb2 <= 2*Ns2);
cond_sb3 = find(z_ech_sb2 < 0 & z_ech_sb2 > -2*Ns2);
cond_sb4 = find(z_ech_sb2 <= -2*Ns2);
dec_sb2 = zeros(1, length(z_ech_sb2));
dec_sb2(cond_sb1) = 3;
```

```
dec_sb2(cond_sb2) = 1;
dec_sb2(cond_sb3) = -1;
dec_sb2(cond_sb4) = -3;
% Demapping
dem_sb2 = reshape(de2bi((dec_sb2 + 3)/2).', 1, length(Donnees));
figure,plot(reshape(z_sb2,Ns2,length(z_sb2)/Ns2));
title('Diagramme de l´Oeuil Deuxième Chaine sans bruit');
% Taux d'erreur binaire
test_sb2 = dem_sb2 - Donnees;
TEB_sb2 = length(find(test_sb2~=0))/length(Donnees); % Taux d'Erreur
 Binaire

% Implantation de la chaine avec bruit
tmp2 = 0;
for k = 1 : Precision
    Donnees = randi([0 1], 1, 10000);
    for i = 1 : length(eb_N0_db)
        mapping2 = (2*bi2de(reshape(Donnees, 2, length(Donnees)/2).')
 - 3).'; % Du binaire au decimal
        Kronr2 = kron(mapping2, [1 zeros(1, Ns2-1)]); % Positionnement
 des zéros entre les ak
        signalfiltre2 = filter(h2,1,Kronr2); % Filtrage
        Px2 = mean(abs(signalfiltre2).^2);
        sigma2 = Px2*Ns1/(2*10^(eb_N0_db(i)/10));
        bruit2 = sqrt(sigma2)*randn(1, length(signalfiltre2));
        Z2 = signalfiltre2 + bruit2;
        z2 = filter(hr2, 1, Z2); % Signal filtré
        %subplot(3,3,i);
        %plot(reshape(z2,Ns1,length(z2)/Ns1));
        z_ech2 = z2(Ns2:Ns2:end); % Échantillonage
        % Decision
        cond1 = find(z_ech2 > 2*Ns2);
        cond2 = find(z_ech2 >= 0 & z_ech2 <= 2*Ns2);
        cond3 = find(z_ech2 < 0 & z_ech2 > -2*Ns2);
        cond4 = find(z_ech2 <= -2*Ns2);
        dec2(cond1) = 3;
        dec2(cond2) = 1;
        dec2(cond3) = -1;
        dec2(cond4) = -3;
        test2 = dec2 - mapping2;
        TES(i) = length(find(test2~=0))/length(mapping2);
        TEB_bruit_2(i) = TES(i)/log2(M2);
    end
    tmp2 = tmp2 + TES;
end
TES = tmp2/Precision;
TEB_bruit_2 = TES/log2(M2);
TES_th = (3/2) * qfunc(sqrt(4/5*10.^(eb_N0_db/10)));
TEB_th2 = TES_th/2;
figure();
semilogy(eb_N0_db, TES_th, 'DisplayName', 'TES théorique');
hold on
semilogy(eb_N0_db, TES, 'DisplayName', 'TES simulé');
legend;
```

```
title('TES Deuxième Chaine');
hold off

figure();
semilogy(eb_N0_db, TEB_th2, 'DisplayName', 'TEB théorique');
hold on
semilogy(eb_N0_db, TEB_bruit_2, 'DisplayName', 'TEB simulé');
legend;
title('TEB Deuxième Chaine avec bruit');
hold off

figure();
semilogy(eb_N0_db, TEB_bruit_r, 'DisplayName', 'TEB Chaine de
 Référence');
hold on
semilogy(eb_N0_db, TEB_bruit_1, 'DisplayName', 'TEB Première Chaine');
legend;
title('Comparaison TEB 1');
hold off

figure();
semilogy(eb_N0_db, TEB_bruit_r, 'DisplayName', 'TEB Chaine de
 Référence');
hold on
semilogy(eb_N0_db, TEB_bruit_2, 'DisplayName', 'TEB Deuxième Chaine');
legend;
title('Comparaison TEB 2');
hold off

figure();
semilogy(eb_N0_db, TEB_bruit_1, 'DisplayName', 'TEB Première Chaine');
hold on
semilogy(eb_N0_db, TEB_bruit_2, 'DisplayName', 'TEB Deuxième Chaine');
legend;
title('Comparaison TEB 3');
hold off
```

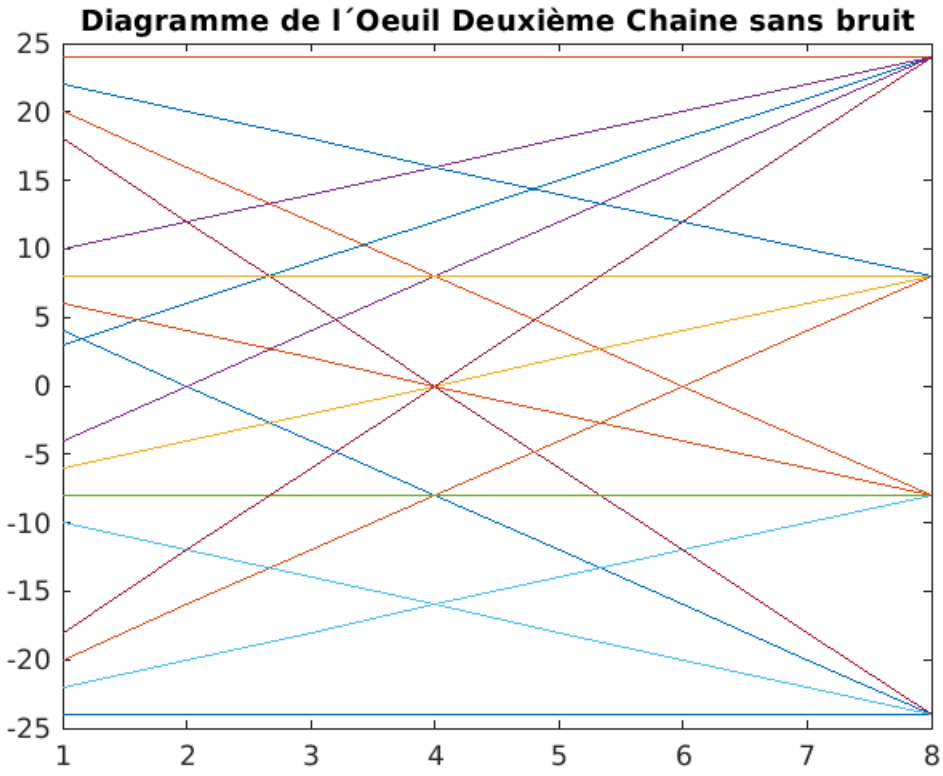
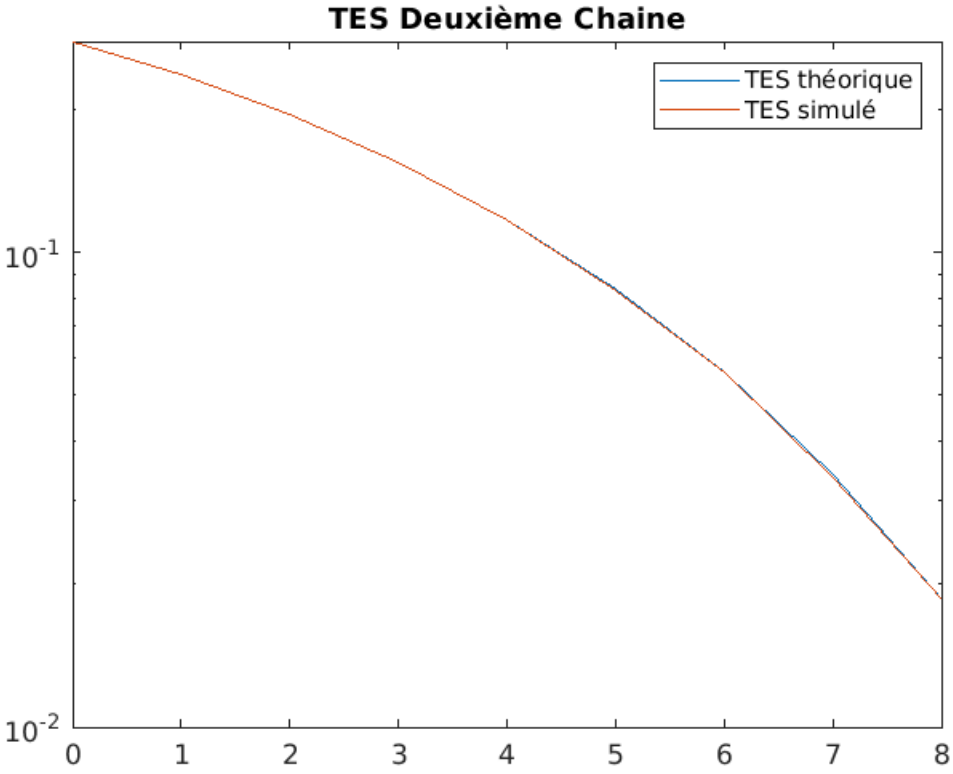
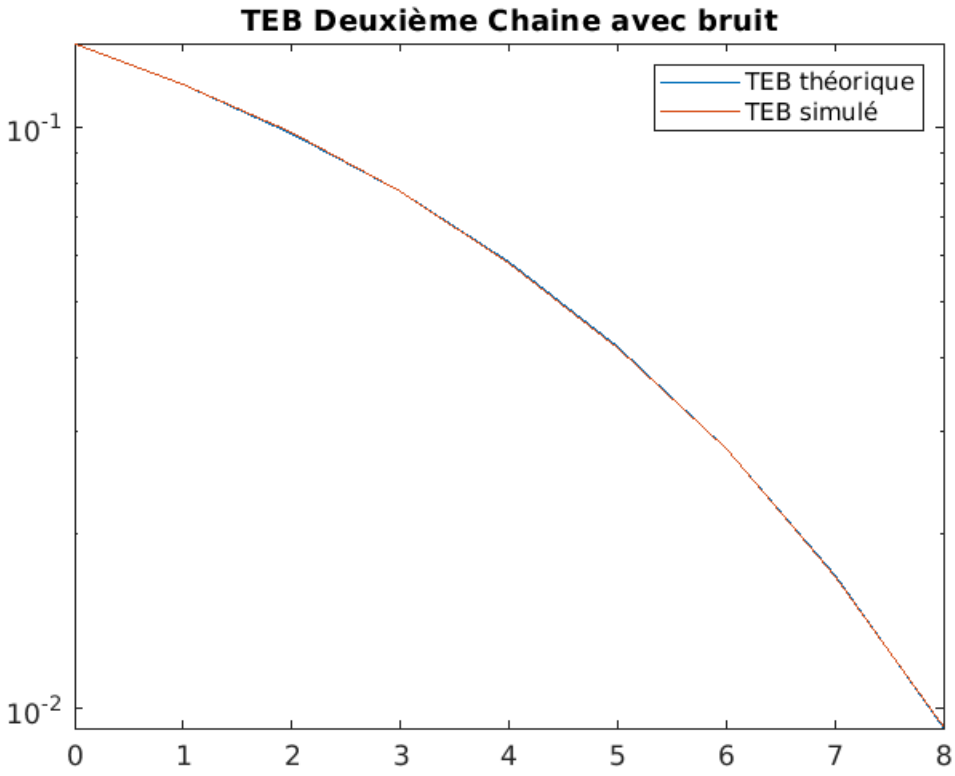
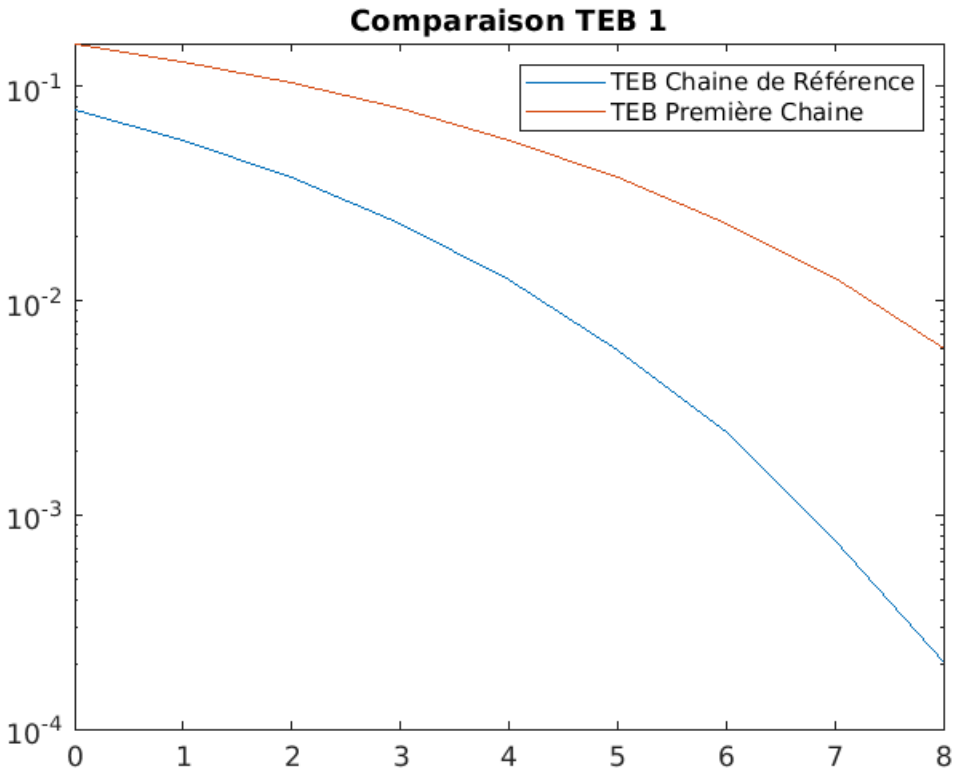
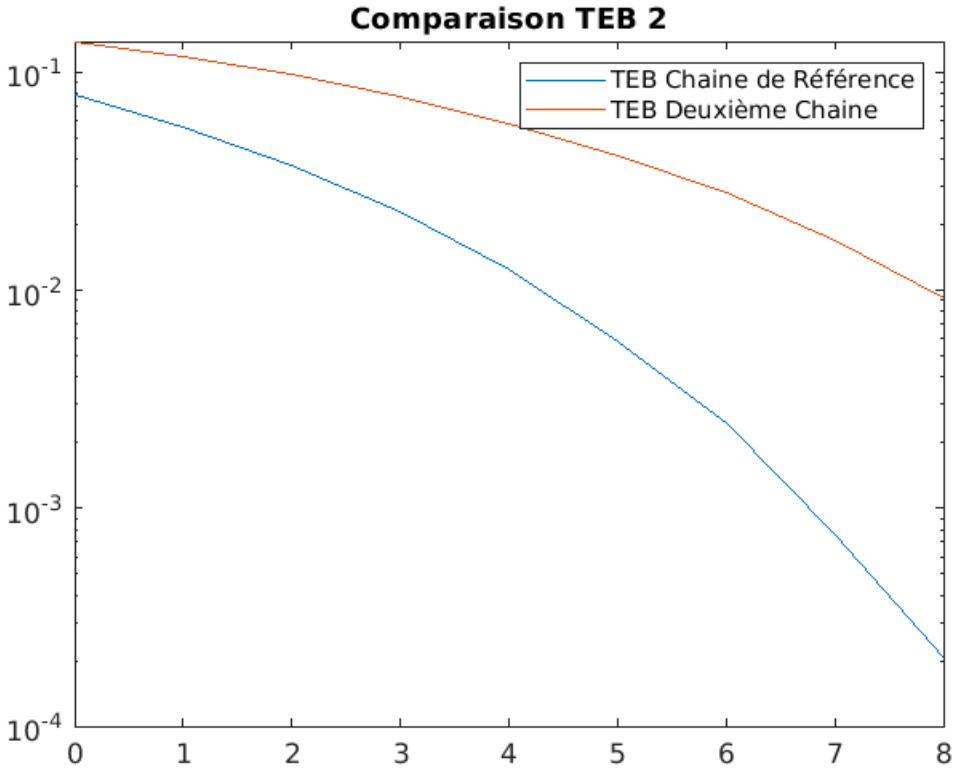


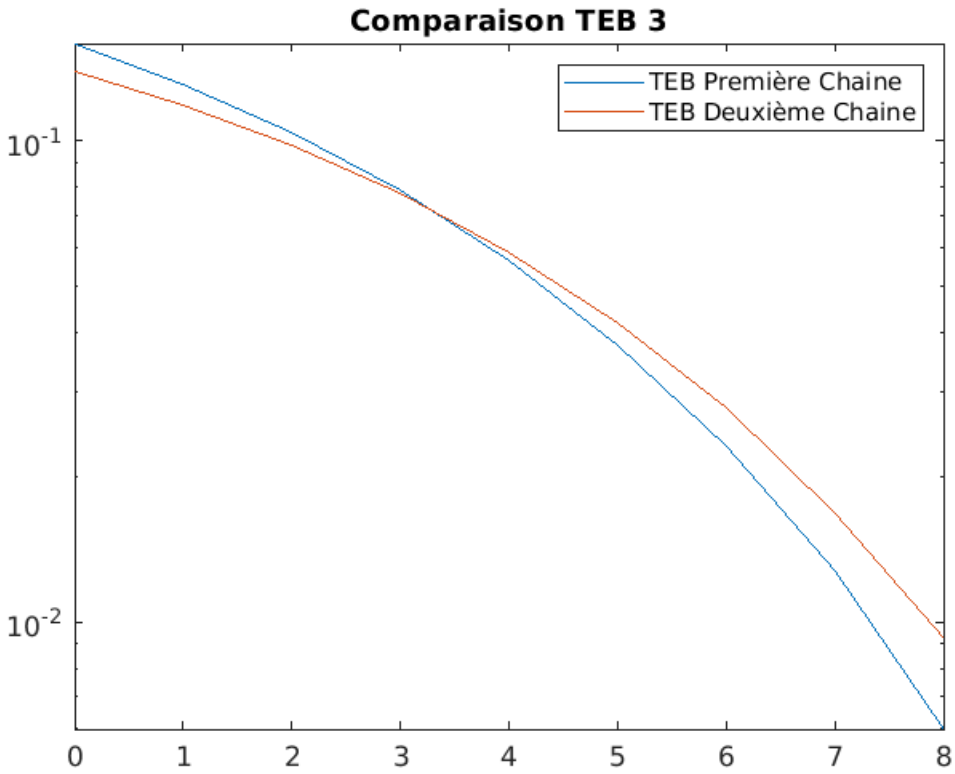
Diagramme de l´Oeuil Deuxième Chaine sans bruit
25
20
15
10
5
0
-5
-10
-15
-20
-25
1
2
3
4
5
6
7
8


TES Deuxième Chaine
TES théorique
TES simulé
10^{-1}
10^{-2}
0
1
2
3
4
5
6
7
8


TEB Deuxième Chaine avec bruit
TEB théorique
TEB simulé
10^{-1}
10^{-2}
0
1
2
3
4
5
6
7
8

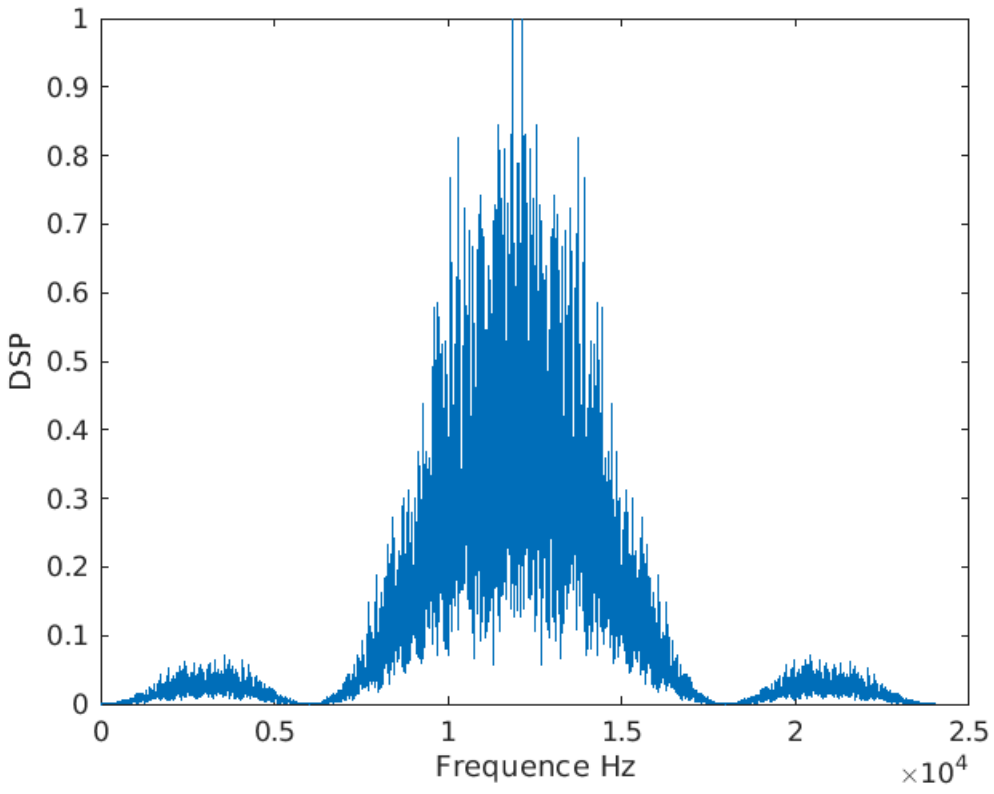

Comparaison TEB 1
TEB Chaine de Référence
TEB Première Chaine
10^{-1}
10^{-2}
10^{-3}
10^{-4}
0
1
2
3
4
5
6
7
8

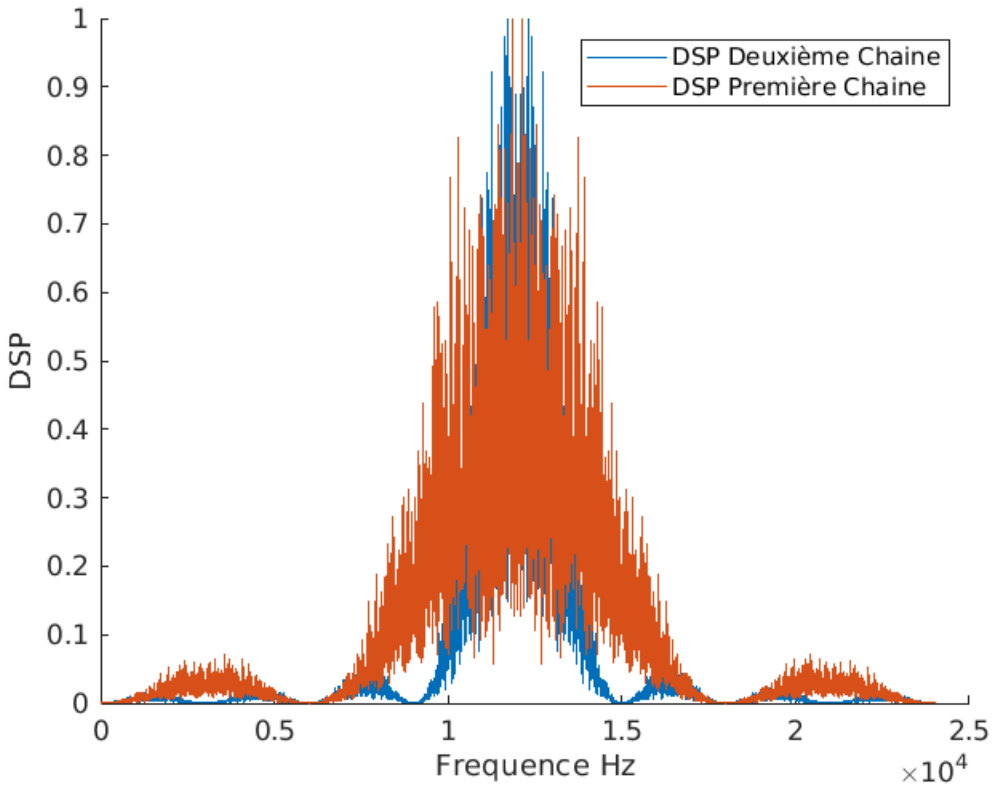

Comparaison TEB 2
TEB Chaine de Référence
TEB Deuxième Chaine
10^{-1}
10^{-2}
10^{-3}
10^{-4}
0
1
2
3
4
5
6
7
8


Comparaison TEB 3
TEB Première Chaine
TEB Deuxième Chaine
10^{-1}
10^{-2}
0
1
2
3
4
5
6
7
8

Comparaisons

```
DSP_x1_centered = pwelch(signalfiltre1, [], [], Fe, "centered"); %
 Densité spectrale de puissance centrée
DSP_x1_centered = DSP_x1_centered./max(DSP_x1_centered); % Normaliser
 la DSP centrée
figure,plot(DSP_x1_centered);
xlabel("Frequence Hz");
ylabel("DSP");
DSP_x2_centered = pwelch(signalfiltre2, [], [], Fe, "centered"); %
 Densité spectrale de puissance centrée
DSP_x2_centered = DSP_x2_centered./max(DSP_x2_centered); % Normaliser
 la DSP centrée
figure,hold on
plot(DSP_x2_centered, 'DisplayName', 'DSP Deuxième Chaine');
plot(DSP_x1_centered, 'DisplayName', 'DSP Première Chaine');
xlabel("Frequence Hz");
ylabel("DSP");
legend;
hold off
```





Published with MATLAB® R2020a