Université Paul Sabatier
Toulouse

Compte Rendu

Signaux et Systèmes

M1 EEA

- Module KEAX7AH1 : Signaux et Systèmes
- Réalisé par :
  - KABOU Abdeldjalil
  - IAICHE Achour Mehdi Anis

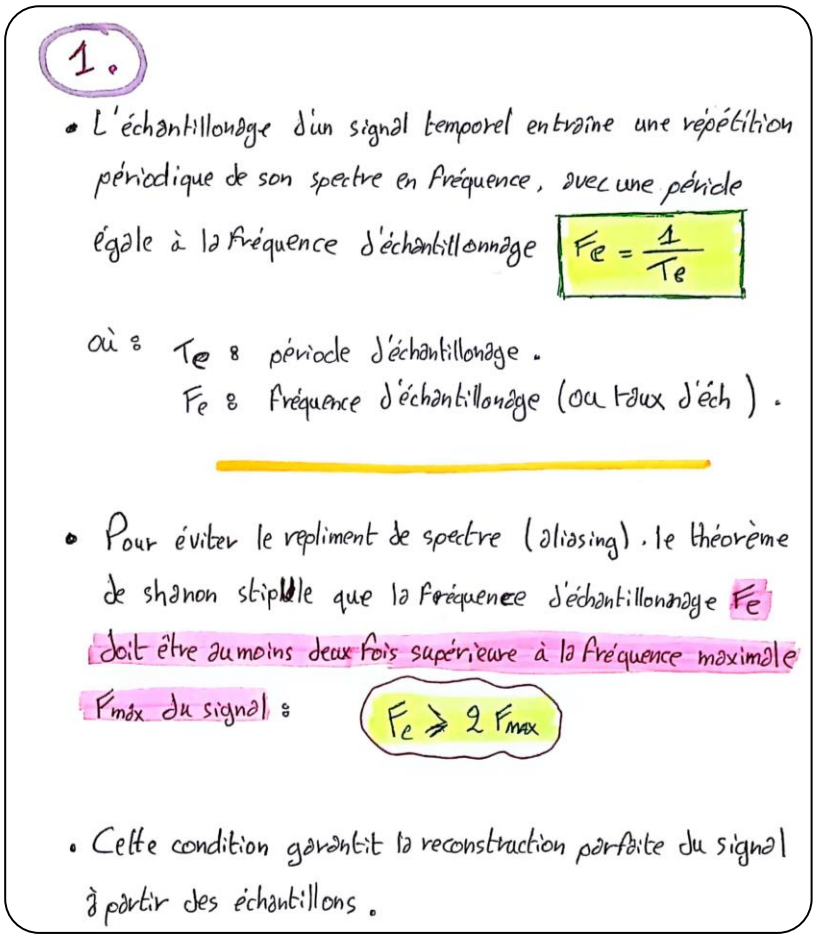
2024 / 2025

TP1 : Numérisation des signaux

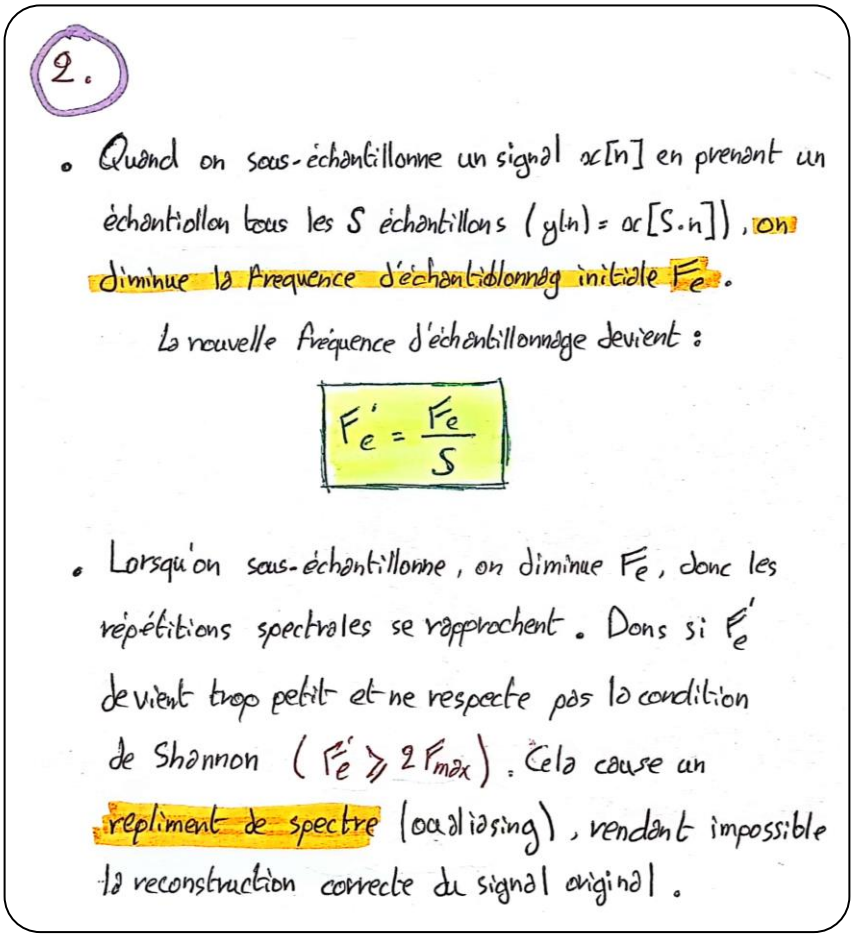
I. *Echantillonnage des Signaux audio*

L'échantillonnage consiste à convertir un signal continu x(t) en une suite discrète x[n] = x(nTe), en prélevant ses valeurs à des intervalles réguliers définis par la période Te.

1) *Effet en fréquence de l'échantillonnage temporel d'un signal et la condition de Shannon pour éviter le repliement de spectre et pouvoir reconstruire le signal original :*



2) Effet en fréquence d'un tel sous-échantillonnage :



3) Fréquence d'échantillonnage de ce signal et sur combien de bits a-t-il été codé :

- Fréquence d'échantillonnage Fe : ***44100 Hz***
- Nombre de bits : ***16 bits***

**Le code est ci-dessous dans les annexes*

4) Pour le signal sous-échantillonné d'un facteur S pour différentes valeurs de S

Observation :

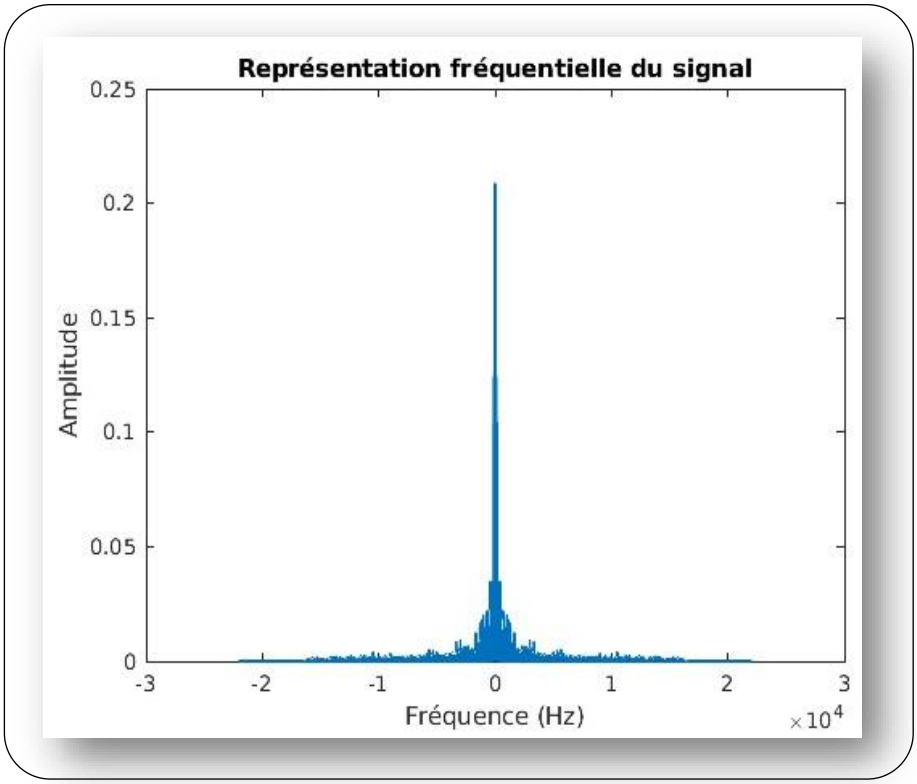
- Pour de petites valeurs de S, le signal est restitué fidèlement.
- À partir d'une certaine valeur de S, le signal commence à perdre des détails, devient moins clair, et des artefacts d'aliasing peuvent apparaître.
- Ces modifications apparaissent lorsque $Fe' = Fe / S < 2f\text{max}$, violant ainsi la condition de Shannon.

A partir de quelle valeur de S entend-on que le signal a subi des modifications :

- S=1 : Le signal est restitué parfaitement, aucune perte n'est perceptible.
- S=2 : Le signal reste le même , mais le son est plus long
- S=4 : Une légère perte de clarté peut être perçue pour des sons complexes.
- S=8 : Des dégradations significatives commencent à apparaître, notamment pour des hautes fréquences.
- S=16 : Le signal est méconnaissable, avec des artefacts d'aliasing. Cela est dû à la violation de la condition de Shannon.

**Le code est ci-dessous dans les annexes*

5) *Traçage la représentation fréquentielle du signal en utilisant la fonction tfsc2.*



L'objectif : est de tracer la représentation fréquentielle et d'identifier la fréquence maximale fmax dans le signal. Ensuite, nous vérifierons si la condition de Shannon est respectée.

- On identifie fmax à partir de la représentation fréquentielle (dans notre cas fmax = ***13387.081 Hz***), donc d'après le théorème de Shannon, le signal peut être échantillonné sans perte d'information *à une fréquence :* *Fe,min = 2fmax*
  Fe ≥ Fe,min

  Avec fmax = ***13387.081Hz,*** *On obtient :* | *Fe, min =* ***27374.162 Hz*** |

  Donc, *Fe > Fe, min.* Il est possible d'échantillonner le signal sans perte d'information.

- *Cohérence avec vos réponses à la question précédente :*

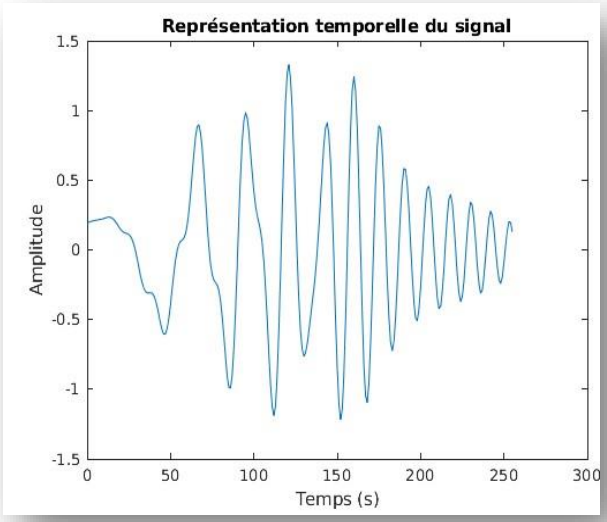
Effectivement, car la condition de Shannon est respectée *Fe > Fe,min*.

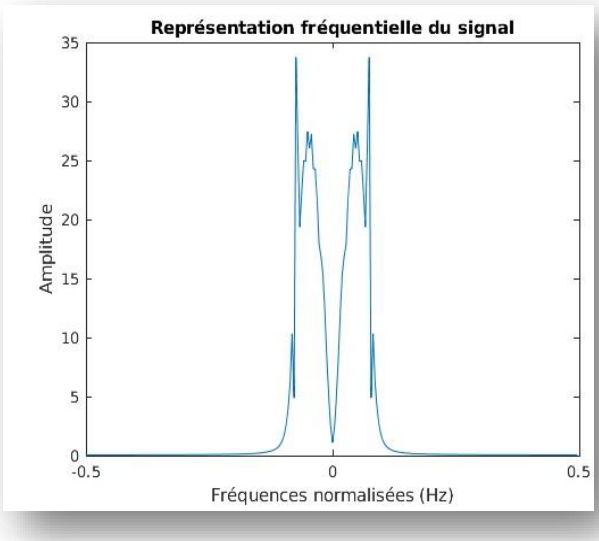
Fe,min = ***27374.162 Hz***
Fe = ***44100 Hz***
Avec : ***Fe > Fe,min***
Donc : elle est respectée

**Le code est ci-dessous dans les annexes*

II. Echantillonnage et reconstruction de signal simple

1) *Traçage les représentations temporelle et fréquentielle du signal x :*



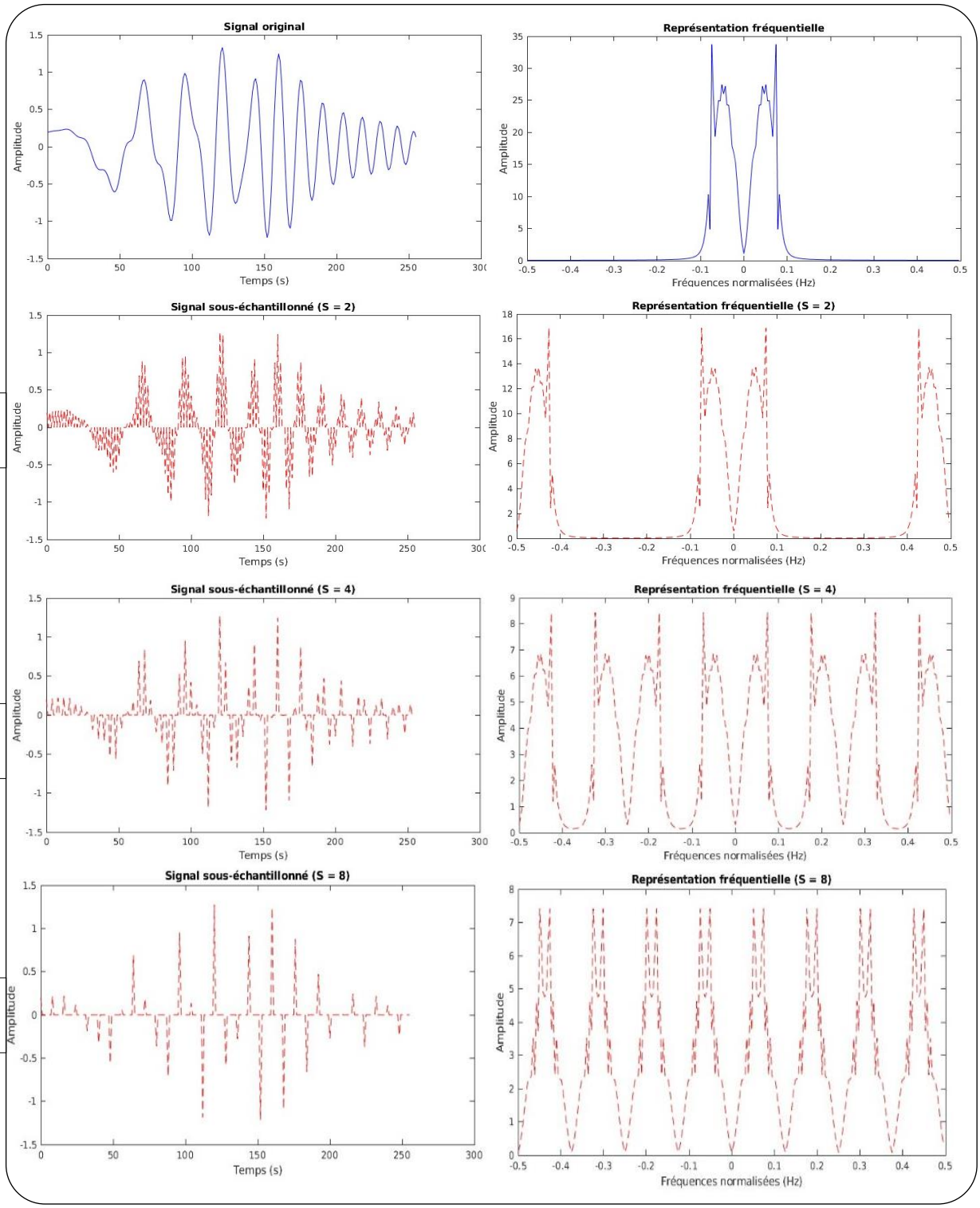


Graphique temporel :
Pour visualiser le signal dans le domaine temporel.

Graphique fréquentiel :
Pour obtenir la représentation fréquentielle en utilisant FTD.

**Le code est ci-dessous dans les annexes*

2) *Traçage les représentations temporelle et fréquentielle des signaux résultants (en ne conservant qu'un échantillon sur S de x (sous-échantillonnage d'un facteur S) et en mettant les autres échantillons à zéro) :*



Facteur ***S=2***

Facteur ***S=4***

Facteur ***S=8***

Résultats attendus :

- Représentation temporelle : À mesure que S augmente, le signal devient discontinu et perd ses détails.
- Représentation fréquentielle : Les composantes hautes fréquences se replient (aliasing), provoquant une distorsion dans le signal reconstruit.

Commentaire :

Pour *petit S,* le sous-échantillonnage reste acceptable car ***Fe' = Fe/S ≥ 2fmax*** (conformément au théorème de Shannon). Et pour des grandes valeurs de S, ***Fe' < 2fmax*** les hautes fréquences du signal se replient, provoquant des pertes irréversibles et rendant le signal non reconstructible.

**Le code est ci-dessous dans les annexes*

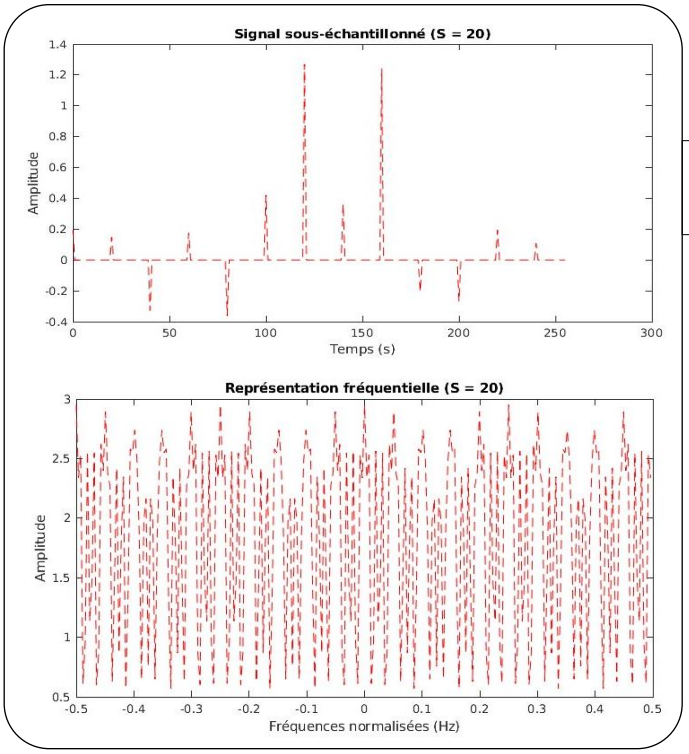
3) *La fréquence minimale avec laquelle on peut échantillonner le signal :*

- Selon le **théorème de Shannon**, pour échantillonner un signal sans perte d'information, la fréquence d'échantillonnage minimale doit être au moins deux fois la fréquence maximale présente dans le signal.

  Si fmax est la fréquence maximale dans le spectre, alors : $Fe,min = 2fmax$

- Le facteur de sous-échantillonnage S est donné par : $S = Fe / Fe,min$

- Observation pour un facteur de sous-échantillonnage trop grand (exemple S=20)



Facteur ***S=20***

S est trop grand (Fe'=Fe/S<2fmax) alors :

- Le *repliement de spectre (aliasing)* se produit.
- Les hautes fréquences du signal "se replient" dans la bande passante inférieure, provoquant des distorsions et rendant le signal non reconstructible.
- Le signal sous-échantillonné perd ses détails.

- *Solution pratique pour éviter le phénomène :*

Si la fréquence d'échantillonnage Fe est fixe et ne satisfait pas la condition de Shannon (Fe<2fmax), il faut réduire la bande passante du signal avant l'échantillonnage en appliquant *un filtre passe-bas* avec une fréquence de coupure fc telle que : $fc \leq Fe / 2$

Ce filtre élimine les composantes fréquentielles au-dessus de Fe/2, empêchant le repliement.

4) *La réponse en fréquence du filtre idéal permettant de reconstruire le signal continu :*

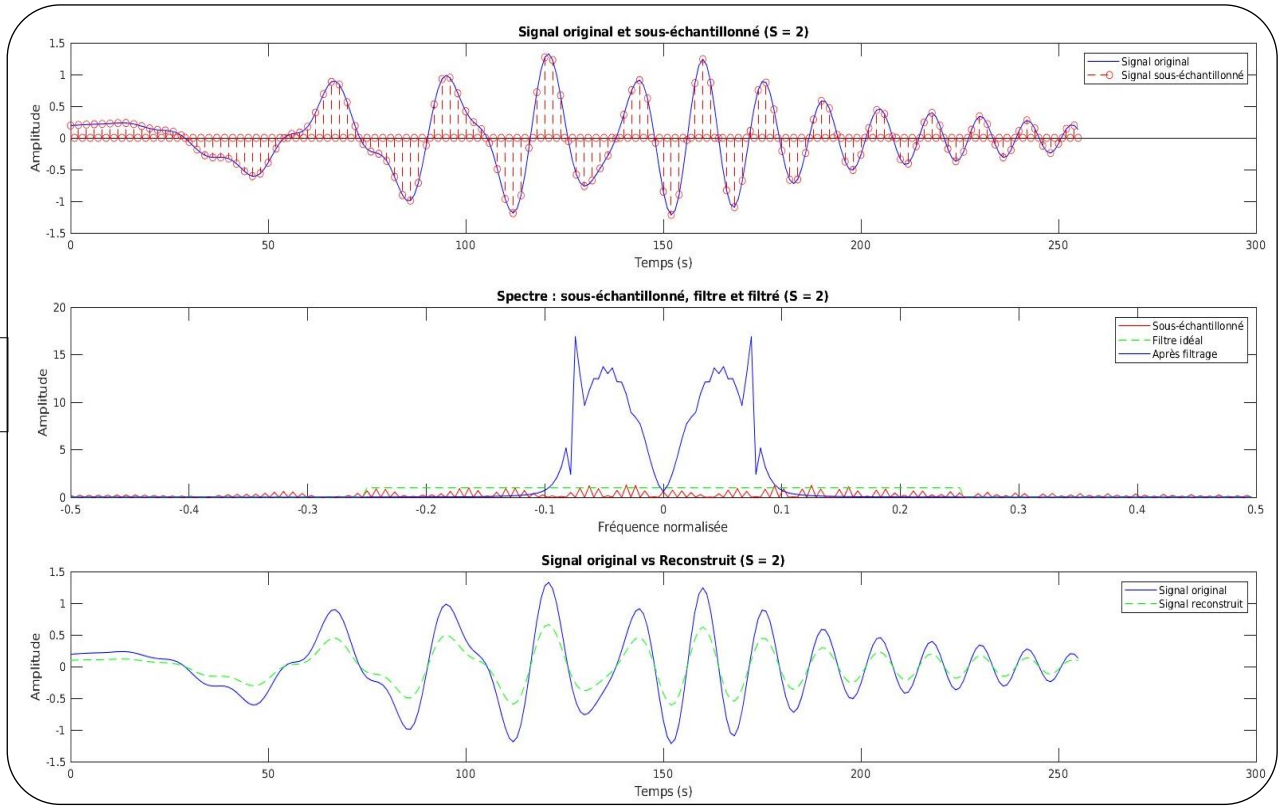
Lorsqu'un signal est sous-échantillonné d'un facteur S, la reconstruction parfaite nécessite un filtrage idéal. Ce filtre doit passer uniquement les fréquences du signal original qui sont en dessous de la fréquence maximale *f* max, tout en supprimant les fréquences hors de cette bande (*Bande passante* : [−fmax, fmax]].

Pour un sous-échantillonnage d'un facteur S, la fréquence d'échantillonnage effective devient Fe'= Fe / S.

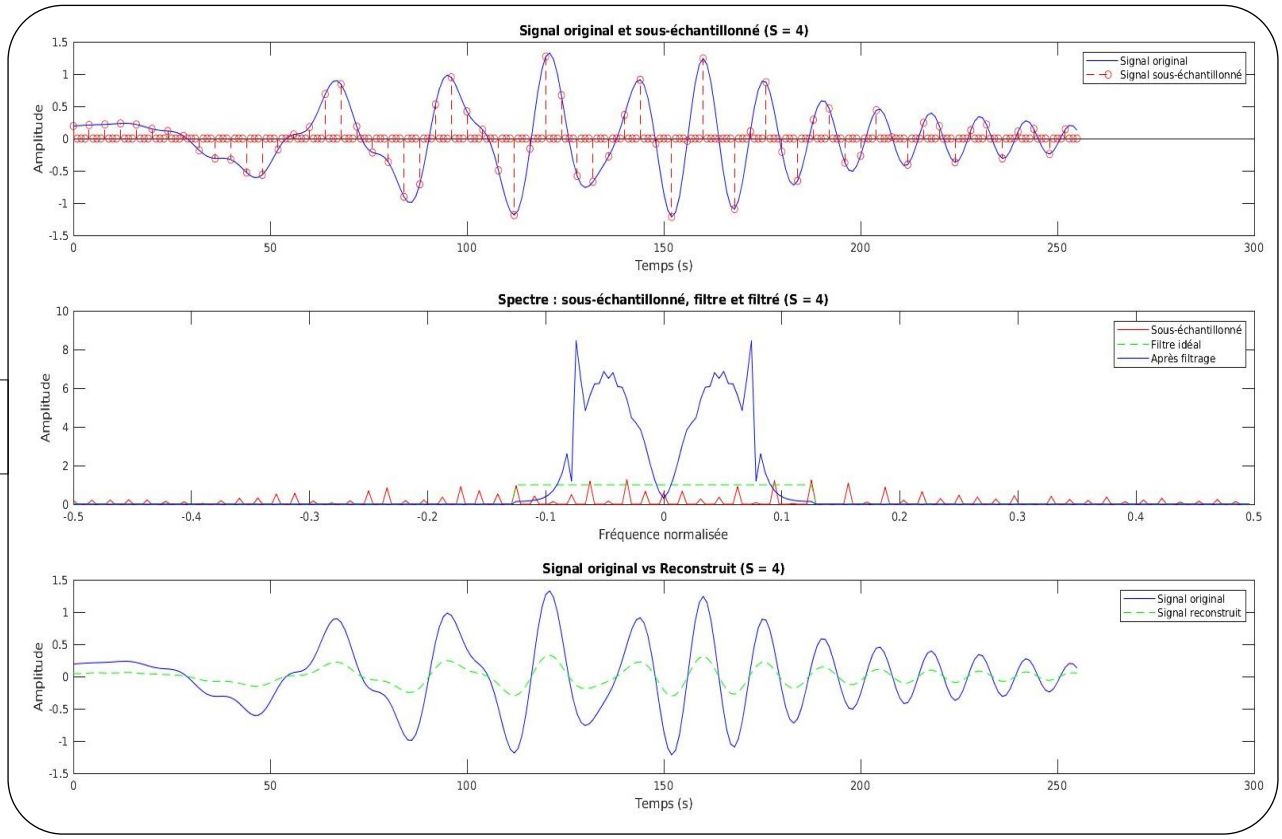
- Le filtre doit avoir une fréquence de coupure fc telle que : $fc = Fe'/2 = Fe/2S$.
- Cela garantit que seules les fréquences du signal original sont conservées.

5) *La réponse en fréquence du filtre idéal permettant de reconstruire le signal continu :*

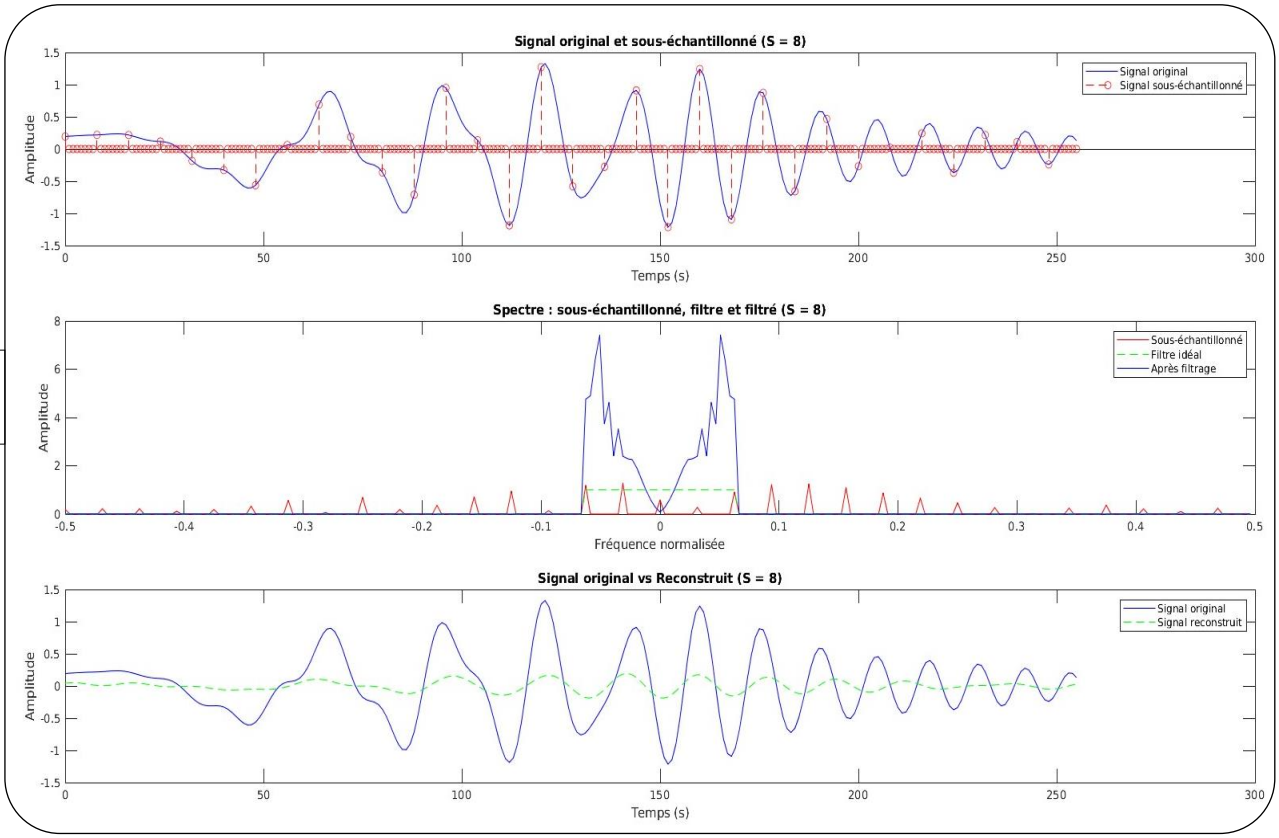
Facteur ***S=2***



Facteur ***S=4***



Facteur
S=8



Observations et commentaires :

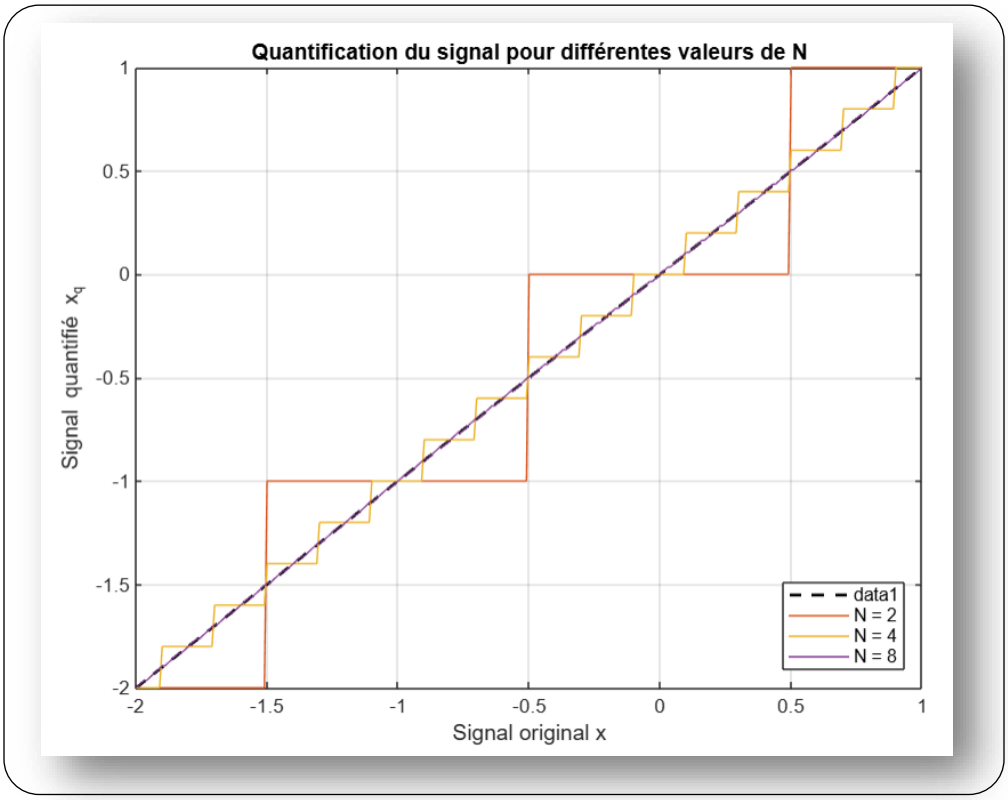
- Pour des valeurs faibles de S (ex. S=2) : La reconstruction est fidèle au signal original car la condition de Shannon est respectée (Fe'≥2fmax).
- Pour des valeurs élevées de S (ex. S=8) : La reconstruction échoue car le repliement de spectre (aliasing) est irréversible lorsque Fe'<2fmax. Les hautes fréquences du signal original sont déjà perdues dans le signal sous-échantillonné.

Remarque :

Si Fe est insuffisante pour éviter le repliement, il faut filtrer le signal original avant l'échantillonnage pour éliminer les hautes fréquences non représentables.

III. Quantification

1. *Traçage x et xq sur une même figure pour différentes valeurs de N (nous avons choisi N= [2,4,8]) :*



**Le code est ci-dessous dans les annexes*

2. *Ecouter du résultat de la quantification du signal contenu dans le fichier SignalQuantif.mat:*

- ***N=2*** : Le signal est fortement dégradé. On entendra une distorsion marquée avec un son rugueux ou métallique.
- **N=4** : La qualité sonore s'améliore, mais des distorsions sont encore perceptibles.
- **N=8** : Le signal est proche de l'original.
- **N=16** : La qualité est quasiment identique à celle du signal original, car les erreurs dues à la quantification deviennent imperceptibles.

Remarque :

Pour une bonne qualité sonore, *N ≥ 8* est recommandé. Une valeur trop basse de N provoque une distorsion perceptible, illustrant l'importance d'un nombre de bits suffisant pour la quantification.

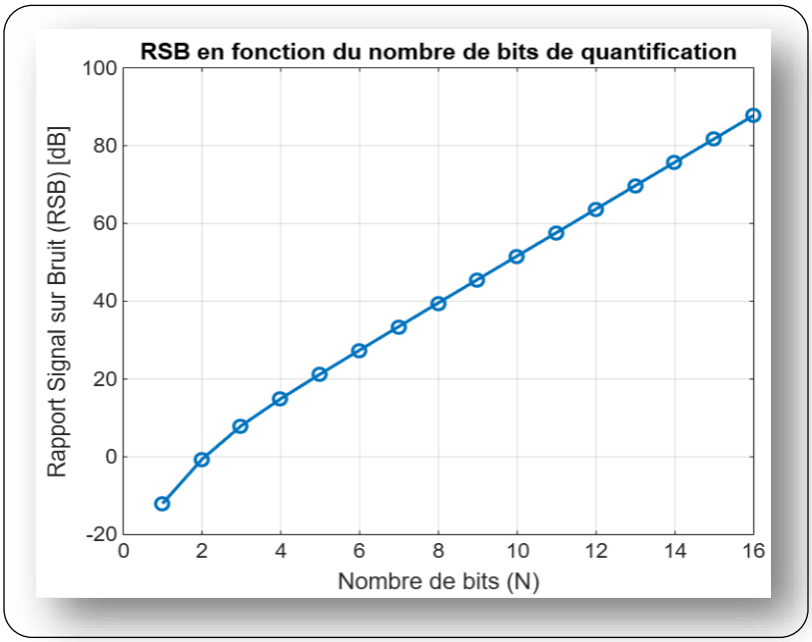
**Le code est ci-dessous dans les annexes*

3. *Rapport signal sur bruit de quantification en dB :* $RSB = 10\log_{10}\frac{P_S}{P_B}$

- La puissance du signal *Ps* est calculée avec *mean(x.^2)*.
- La puissance de l'erreur de quantification *Pb* est obtenue à partir de l'écart *x−xq*.

Traçage RSB en fonction de N :

Le graphe montre comment le RSB augmente avec le nombre de bits de quantification N



Commentaire :

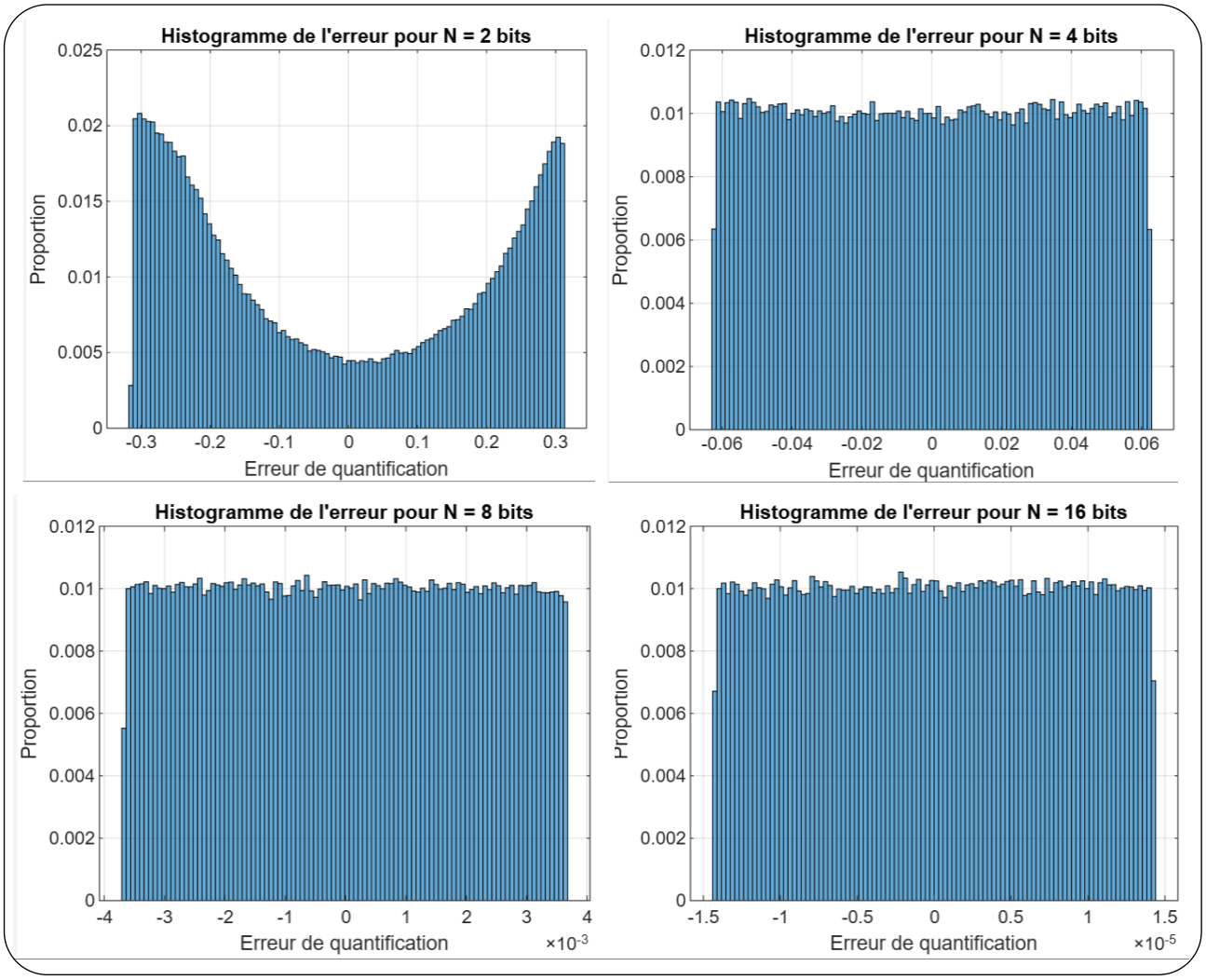
Pour des valeurs faibles de N, le RSB est bas, indiquant une perte de qualité notable. Avec N ≥ 8, le RSB est suffisamment haut pour garantir une qualité sonore acceptable.

Conclusion :

Le RSB augmente avec N, ce qui signifie que le nombre de bits améliore significativement la qualité du signal (Une augmentation de N améliore la fidélité).

**Le code est ci-dessous dans les annexes*

4. *Histogramme de l'erreur de quantification sur 100 niveaux :*



Commentaire :

Pour faible N (N=2) : Les erreurs de quantification sont plus importantes, avec une distribution étendue et uniforme. Cela reflète une précision limitée dans la quantification. Et pour grand N (N=16) : Les erreurs deviennent très petites, centrées autour de zéro. La distribution se contracte et la précision est nettement améliorée.

Donc on obtient une variable aléatoire uniforme centré entre [-1/2^N,1/2^N]

Annexes

Echantillonnage des Signaux audio

3. *Fréquence d'échantillonnage de ce signal et sur combien de bits a-t-il été codé :*

```
% Charger le signal audio avec wavread (ancienne méthode) ou audioread (nouvelle
méthode)
% y : données audio, Fs : fréquence d'échantillonnage, nbits : nombre de bits
[y, Fs, nbits] = wavread('signal0.wav');
% Afficher les informations extraites
disp(['Fréquence d'échantillonnage : ', num2str(Fs), ' Hz']);
disp(['Nombre de bits : ', num2str(nbits), ' bits']);
% Écouter le signal
PlaySignal(y, Fs); % Lecture avec notre fonction
```

4. *Pour le signal sous-échantillonné d'un facteur S pour différentes valeurs de S*

```
% Charger le signal audio avec wavread (ancienne méthode) ou audioread (nouvelle
méthode)
% Charger le signal
[signal, Fe] = audioread ('signal0.wav'); % signal : données audio, Fe : fréquence
d'échantillonnage
% Facteurs de sous-échantillonnage
S = 1;
%S = 2;
%S = 4;
%S = 8;
%S = 16;
disp(['Lecture du signal sous-échantillonné avec S = ', num2str(S)]);
PlaySignal(signal, Fe/S); % Lecture sous-échantillonnée
```

5. *Traçage la représentation fréquentielle du signal en utilisant la fonction tfsc2 et déterminer la fréquence maximale.*

```
% Charger le signal audio
 [y, Fs] = audioread('signal0.wav'); % y : données audio, Fs : fréquence
 d'échantillonnage
 % Calculer la transformée de Fourier discrète avec tfsc2
 [Xhat, f] = tfsc2(y, Fs);  % tfsc2 retourne les fréquences et amplitudes
 figure;
 plot(f, abs(Xhat)); xlabel('Fréquence (Hz)');
 ylabel('Amplitude'); title('Représentation fréquentielle du signal');
 % Identifier la fréquence maximale
 f_max = max(f);
 disp(['Fréquence maximale dans le signal : ', num2str(f_max), ' Hz']);
 % Appliquer le théorème de Shannon
 Fe_min = 2 * f_max; % Fréquence minimale d'échantillonnage selon Shannon
 disp(['Fréquence minimale d'échantillonnage selon Shannon : ', num2str(Fe_min),
 'Hz']);
 % Vérification de cohérence
 if Fe >= Fe_min
     disp('La fréquence d'échantillonnage initiale est suffisante pour éviter les
 pertes.');
 else
     disp('La fréquence d'échantillonnage initiale est insuffisante.');
 end
```

Echantillonnage et reconstruction de signal simple

1. *Traçage les représentations temporelle et fréquentielle du signal x :*

```
% Charger les données
load SignalReconst.mat; % Chargement des variables x (signal) et t (temps)
% Représentation temporelle
figure;
plot(t, x);
xlabel('Temps (s)'); ylabel('Amplitude');
title('Représentation temporelle du signal');
% Représentation fréquentielle
% Calculer la transformée de Fourier discrète avec tfsc2
Fe = 1;
[Xhat, f] = tfsc2(x, Fe);  % tfsc2 retourne les fréquences et amplitudes
figure;
plot(f, abs(Xhat));
xlabel('Fréquences normalisées (Hz)'); % (F_e = 1)
ylabel('Amplitude');
title('Représentation fréquentielle du signal');
```

2. *Traçage les représentations temporelle et fréquentielle des signaux résultants (en ne conservant qu'un échantillon sur S de x (sous-échantillonnage d'un facteur S) et en mettant les autres échantillons à zéro)*

```
% Charger les données
load SignalReconst.mat; % Chargement des variables x (signal) et t (temps)
% Calculer la transformée de Fourier discrète avec tfsc2
Fe = 1;
[Xhat, f] = tfsc2(x, Fe);  % tfsc2 retourne les fréquences et amplitudes
% Définir les valeurs de S à tester
S_values = [2, 4, 8]; % Facteurs de sous-échantillonnage
% Boucle sur différentes valeurs de S
for S = S_values
    % Sous-échantillonnage : ne garder qu'un échantillon sur S
    x_Sous_echantillon = zeros(size(x));
    x_Sous_echantillon(1:S:end) = x(1:S:end); % Échantillons à zéro sauf 1/S
    [Xhat_sous, f_sous] = tfsc2(x_Sous_echantillon, Fe);  % tfsc2 retourne les
fréquences et amplitudes
    % Représentation temporelle
    figure;
    sabplot(2,2,1);
    plot(t, x, 'b');
    xlabel('Temps (s)');
    ylabel('Amplitude');
    title('Signal original');
    sabplot(2,2,3);
    plot(t, x_Sous_echantillon, 'r--');
    xlabel('Temps (s)');
    ylabel('Amplitude');
    title(['Signal sous-échantillonné (S = ', num2str(S), ')']);
    % Représentation fréquentielle
    sabplot(2,2,2);
    plot(f, abs(Xhat),'b');
    xlabel('Fréquences normalisées (Hz)');
    ylabel('Amplitude');
    title('Représentation fréquentielle');
    sabplot(2,2,4);
    plot(f_sous,abs(Xhat_sous),'r--');
    xlabel('Fréquences normalisées (Hz)');
    ylabel('Amplitude');
    title(['Représentation fréquentielle (S = ', num2str(S), ')']);
end
```

5. *La réponse en fréquence du filtre idéal permettant de reconstruire le signal continu :*

```
% Charger le signal original et ses paramètres
load SignalReconst.mat; % Charge x (signal) et t (temps)
% Définir les valeurs de S
S_values = [2, 4, 8];
Fe = 1;
% Boucle sur différentes valeurs de S
for S = S_values
    % Sous-échantillonnage : ne conserver qu'un échantillon sur S
    x_Sous_echantillon = zeros(size(x));
    x_Sous_echantillon(1:S:end) = x(1:S:end); % Échantillons à zéro sauf 1/S
    [Xhat_sous, f_sous] = tfsc2(x_Sous_echantillon, Fe);  % tfsc2 retourne les
fréquences et amplitudes
    % Construction de la réponse en fréquence du filtre idéal
    f_coupure = 1 / (2 * S); % Fréquence de coupure
    H = double(abs(f_sous) <= f_coupure); % Réponse idéale (1 dans la bande utile,
0 ailleurs)
    % Filtrage dans le domaine fréquentiel
    X_filtre = Xhat_sous .* H;
    % Transformée inverse pour retrouver le signal temporel
    x_reconstruit = tfsc2_inv(X_filtre, 1);
    % Comparaison visuelle des signaux
    figure;
    subplot(3, 1, 1);
    plot(t, x, 'b'); hold on;
    stem(t, x_Sous_echantillon, 'r--');
    xlabel('Temps (s)');
    ylabel('Amplitude');
    title(['Signal original et sous-échantillonné (S = ', num2str(S), ')']);
    legend('Signal original', 'Signal sous-échantillonné');
    subplot(3, 1, 2);
    plot(f, abs(x_Sous_echantillon), 'r', f, abs(H), 'g--', f, abs(X_filtre),
'b');
    xlabel('Fréquence normalisée');
    ylabel('Amplitude');
    title(['Spectre : sous-échantillonné, filtre et filtré (S = ', num2str(S),
')']);
    legend('Sous-échantillonné', 'Filtre idéal', 'Après filtrage');
    subplot(3, 1, 3);
    plot(t, x, 'b', t, x_reconstruit, 'g--');
    xlabel('Temps (s)');
    ylabel('Amplitude');
    title(['Signal original vs Reconstruit (S = ', num2str(S), ')']);
    legend('Signal original', 'Signal reconstruit');
end
```

Quantification

1. *Traçage x et xq sur une même figure pour différentes valeurs de N :*

```
function xq = quantif(x, N,a,b)
    b = min(x); %Minimum du Signal
    a = max(x) - b; %Amplitude total
    y = (x - b) / a; % Normalisation
    L = 2^N; % Nombre de niveaux de quantification
    yq = round(y * (L - 1)) / (L - 1); % Quantification et renormalisation
    % Dénormalisation pour revenir à l'échelle originale
    xq = yq * a + b;
end
x = -2 : 0.01 : 1; % test
% Différentes valeurs de N
N_values = [2, 4, 8];
% Tracé des signaux
figure;
plot(x, x, 'k--', 'LineWidth', 1.5); % Signal original
hold on;
for N = N_values
    xq = quantif(x, N); % Quantification
    plot(x, xq, 'DisplayName', ['N = ', num2str(N)]); % Signal quantifié
end
hold off;
xlabel('Signal original x');
ylabel('Signal quantifié x_q');
title('Quantification du signal pour différentes valeurs de N');
legend('Location', 'best');
grid on;
```

2. *Ecoute du résultat de la quantification du signal contenu dans le fichier SignalQuantif.mat :*

```
function xq = quantif(x, N)
    b = min(x); % Minimum du signal
    a = max(x) - b; % Amplitude totale
    y = (x - b) / a; % Normalisation
    L = 2^N; % Nombre de niveaux de quantification
    yq = round(y * (L - 1)) / (L - 1); % Quantification et renormalisation
    % Dénormalisation pour revenir à l'échelle originale
    xq = yq * a + b;
end
load SignalQuantif.mat; % Variables x (signal) et Fe (fréquence d'échantillonnage)
% Paramètres de quantification
N_values = [2, 4, 8, 16]; % Différentes valeurs de N (nombre de bits)
% Boucle pour quantifier et écouter le signal
for N = N_values
    % Quantification du signal
    xq = quantif(x, N);
    % Écouter le signal quantifié
    fprintf('Lecture du signal quantifié avec N = %d bits\n', N);
    PlaySignal(xq, Fe);
    %sound(xq, Fe); % Fonction MATLAB pour jouer un signal sonore
    % Pause pour écouter avant de passer à la prochaine valeur de N
    pause(5); % Ajuster selon la durée du signal
end
```

3. *Rapport signal sur bruit de quantification en dB :*

```
function xq = quantif(x, N)
    b = min(x); % Minimum du signal
    a = max(x) - b; % Amplitude totale
    y = (x - b) / a; % Normalisation
    L = 2^N; % Nombre de niveaux de quantification
    yq = round(y * (L - 1)) / (L - 1); % Quantification et renormalisation
    % Dénormalisation pour revenir à l'échelle originale
    xq = yq * a + b;
end
load SignalQuantif.mat;
N_values = 1:16; % Plage de valeurs pour N (de 1 à 16 bits)
% Initialisation du vecteur RSB
RSB = zeros(size(N_values));
% Calcul pour chaque N
for i = 1:length(N_values)
    N = N_values(i);
    % Quantification du signal
    xq = quantif(x, N);
    % Puissance du signal
    Ps = mean(x.^2);
    % Puissance de l'erreur de quantification
    erreur = x - xq; % Différence entre le signal original et le signal quantifié
    Pb = mean(erreur.^2);
    % Calcul du RSB en dB
    RSB(i) = 10 * log10(Ps / Pb);
end
% Tracé du RSB en fonction de N
figure;
plot(N_values, RSB, '-o', 'LineWidth', 1.5);
xlabel('Nombre de bits (N)');
ylabel('Rapport Signal sur Bruit (RSB) [dB]');
title('RSB en fonction du nombre de bits de quantification');
grid on;
```

4. *Histogramme de l'erreur de quantification sur 100 niveaux :*

```
load SignalQuantif.mat;
N_values = [2, 4, 8, 16];
% Nombre de niveaux pour l'histogramme
n_bins = 100;
% Boucle sur chaque valeur de N
for i = 1:length(N_values)
    N = N_values(i);
    % Quantification du signal
    xq = quantif(x, N);
    % Calcul de l'erreur de quantification
    erreur = x - xq;
    % Tracer l'histogramme
    figure;
    histogram(erreur, n_bins, 'Normalization', 'probability'); % Normalisation
pour avoir des proportions
    xlabel('Erreur de quantification');
    ylabel('Proportion');
    title(['Histogramme de l''erreur pour N = ', num2str(N), ' bits']);
    grid on;
end
```