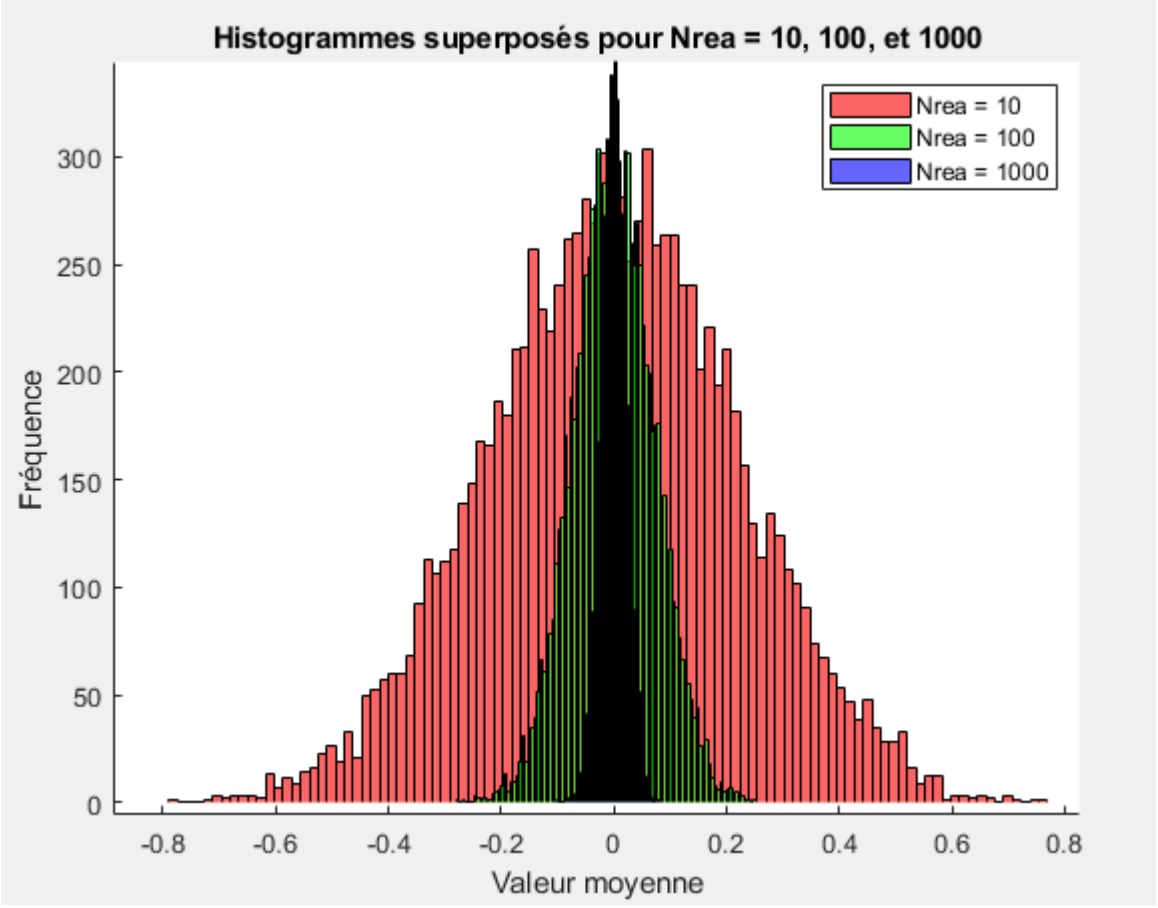
EX1:

```
clear all;
close all;

Nrea_vec=[10,100,1000]; N = 10000; n_vec=1:N; w = 0.05 * pi;
colors = {'r', 'g', 'b'};
hold on;

for i=1:length(Nrea_vec),
    Nrea = Nrea_vec(i);
    Dmat=[];
    for j=1:Nrea,
        phi_vec = 2*pi*rand(1,N);
        d_vec = sin(n_vec * w + phi_vec);
        Dmat = [Dmat; d_vec];
    end
    moy_vec = mean(Dmat);
    histogram(moy_vec,100, 'FaceColor', colors{i});
end;

title('Histogrammes superposés pour Nrea = 10, 100, et 1000');
xlabel('Valeur moyenne');
ylabel('Fréquence');
legend('Nrea = 10', 'Nrea = 100', 'Nrea = 1000');
hold off;
```

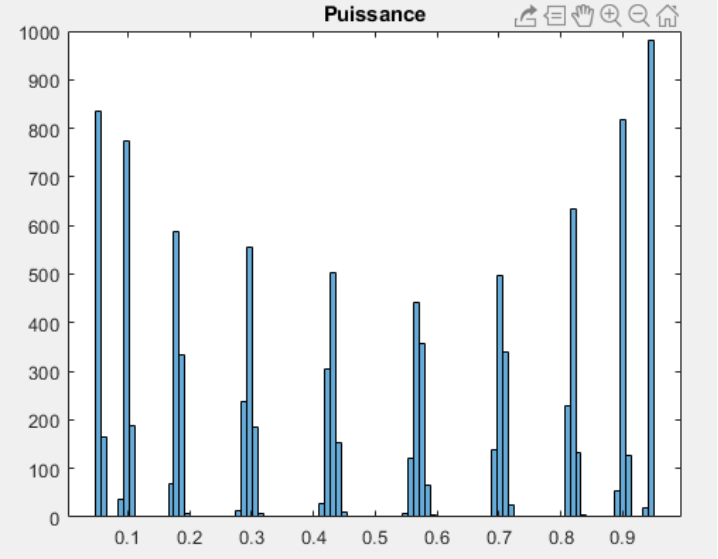


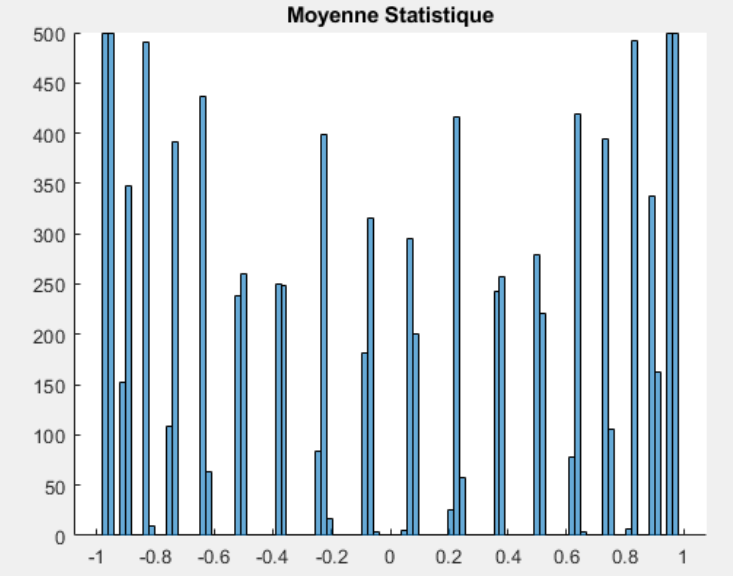
$$m = E[\sin(wn+\phi)] = \frac{1}{2\pi}\int_0^{2\pi} \sin(wn+\phi)\,d\phi = 0$$

On remarque que plus le nombre de réalisation augmente, plus la valeur moyenne devient plus concentrée en 0 c'est a dire plus précise.

EX2:

```
clear all;
close all;

Nrea=1000; N = 10000; n_vec=1:N; w = 0.05 * pi;
hold on;

Dmat=[];
for j=1:Nrea,
    phi_vec = (pi/4)*rand(1,N);
    d_vec = sin(n_vec * w + phi_vec);
    Dmat = [Dmat; d_vec];
end
moy = mean(Dmat);
histogram(moy,100);
title('Moyenne Statistique');
```





$$d(n) = sin(wn+\phi)$$

$$E[sin(wn+\phi)] = \frac{1}{\frac{\pi}{4}-0}\int_0^{\frac{\pi}{4}} sin(wn+\phi)d\phi = \frac{4}{\pi}[-\cos(wn+\phi)]$$

$$E[d(n)] = \frac{4}{\pi}\left(\cos(wn) - \cos\left(wn+\frac{\pi}{4}\right)\right)$$

=> La moyenne est une fonction sinusoidale
l'intervalle [0,pi/4] introduit un **biais positif**, ce qui entraîne une moyenne statistique non nulle.

$$P(n) = E[d(n)2] = E[sin2(wn + \phi)] \; = \; \frac{4}{\pi} \int_0^{\frac{\pi}{4}} sin^2(wn + \phi) d\phi$$

$$P(n) = \frac{4}{\pi}\left(\frac{1}{2}\int_0^{\frac{\pi}{4}} 1\, d\phi - \frac{1}{2}\int_0^{\frac{\pi}{4}} \cos(2wn+\phi)\, d\phi\right) = \frac{1}{2} - \frac{1}{\pi}\cdot\left(\sin\left(2(wn+\frac{\pi}{4})\right) - \sin(2wn)\right)$$

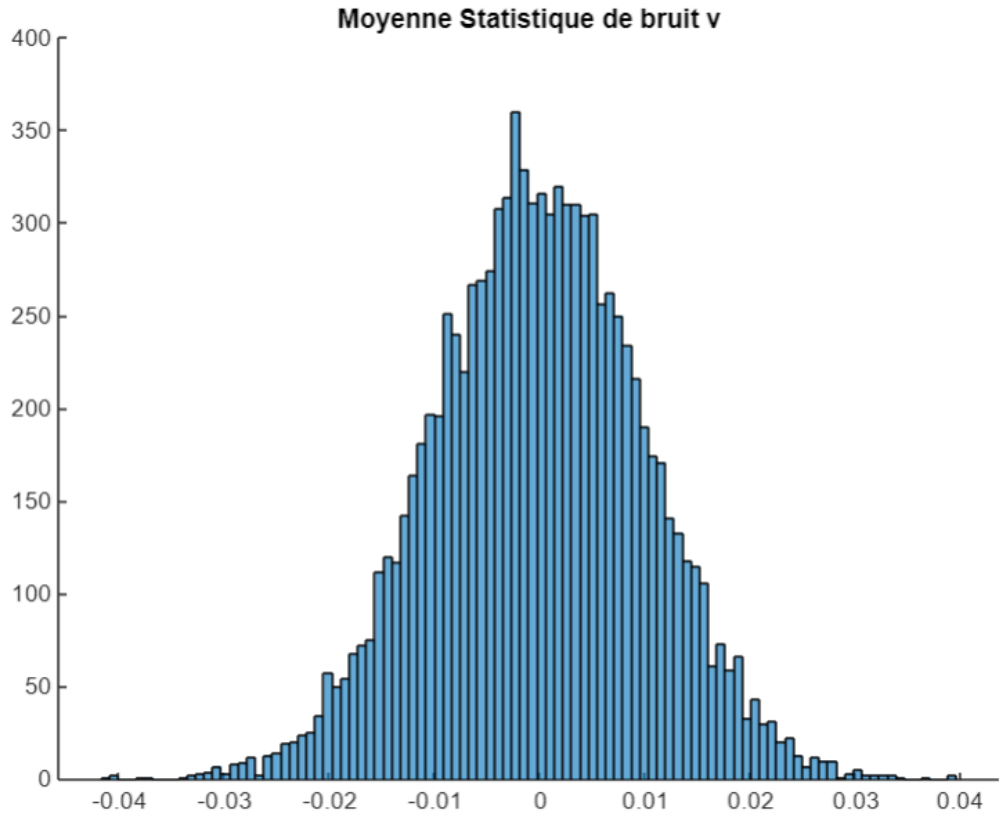
Des oscillations dues à la distribution restreinte de la phase phi. Ces oscillations elles montrent que la puissance n'est pas uniformément répartie lorsque la phase est limitée à un intervalle restreint comme [0,pi/4]

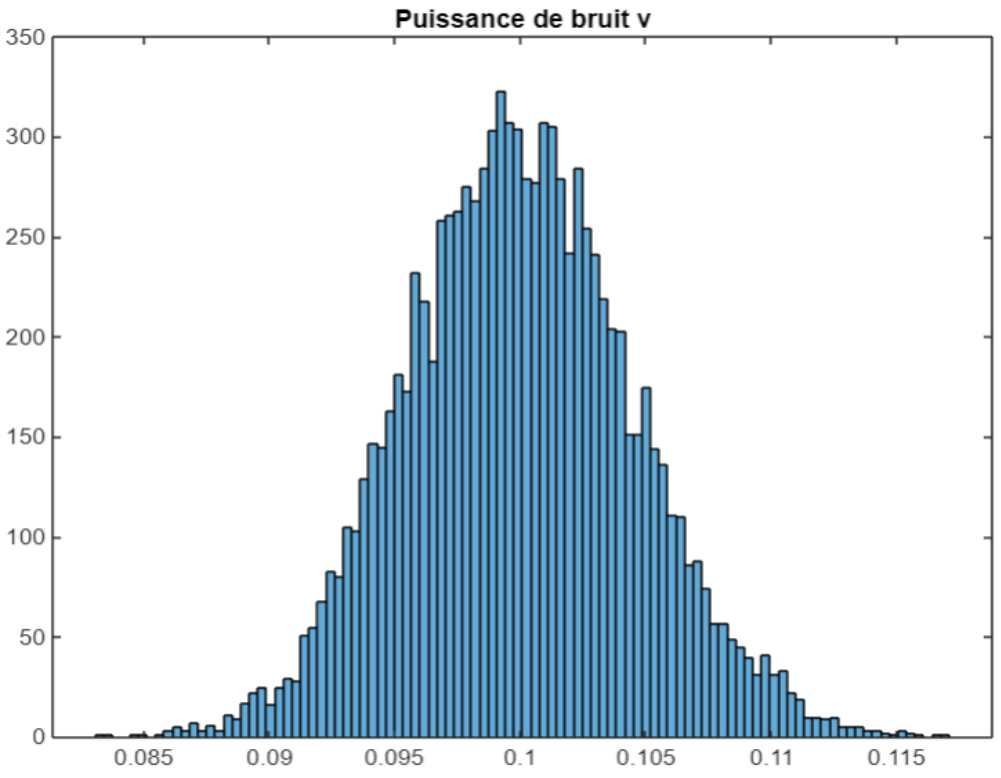
EX3:

```
clear all;
close all;
Nrea=1000; N = 10000; n_vec=1:N ; sigma_v=sqrt(0.1); mv=0;
hold on;
Vmat=[];
for j=1:Nrea,

   v_vec=sigma_v * randn(1,N)+mv;
   Vmat = [Vmat; v_vec];
end
moy = mean(Vmat);
histogram(moy,100);
title('Moyenne Statistique ');
figure;
puissance = mean(Vmat.^2);
histogram(puissance,100);
title('Puissance');
```

---

sigma_v=sqrt(0.1); mv=0:



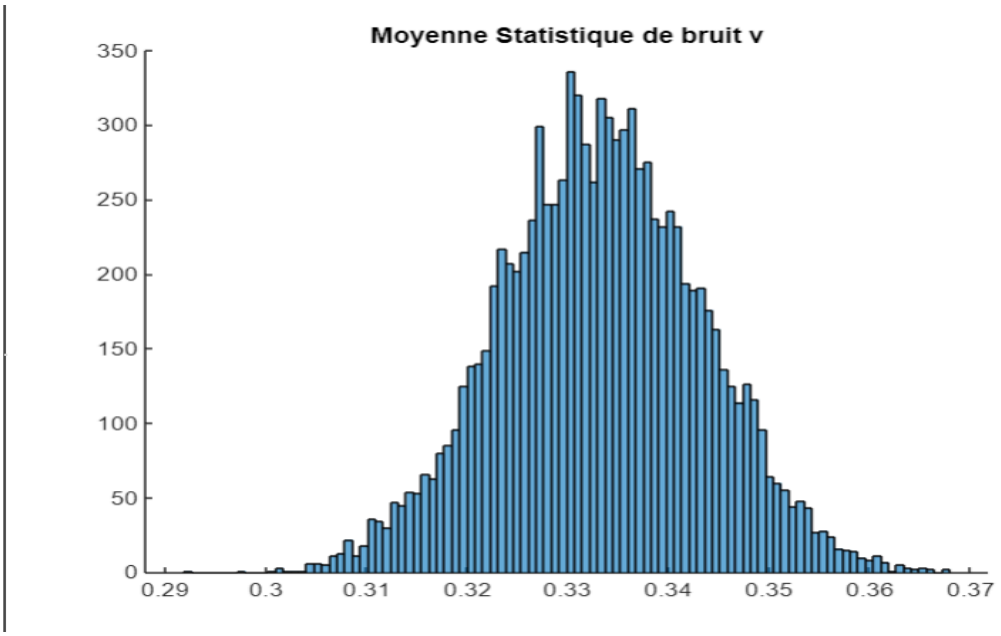


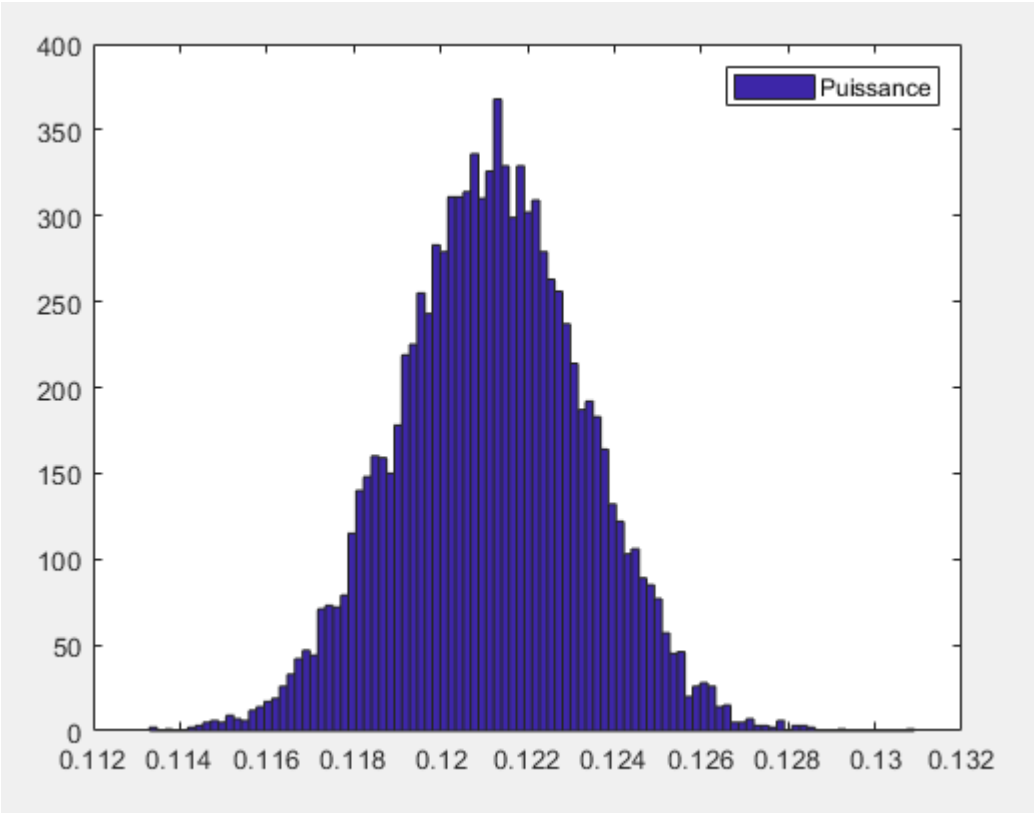
$$E[v] = 0$$

car bruit est centre

$$Pv = \sigma^2 + m^2 = 0,1$$

sigma_v=sqrt(0.01); mv=1/3:



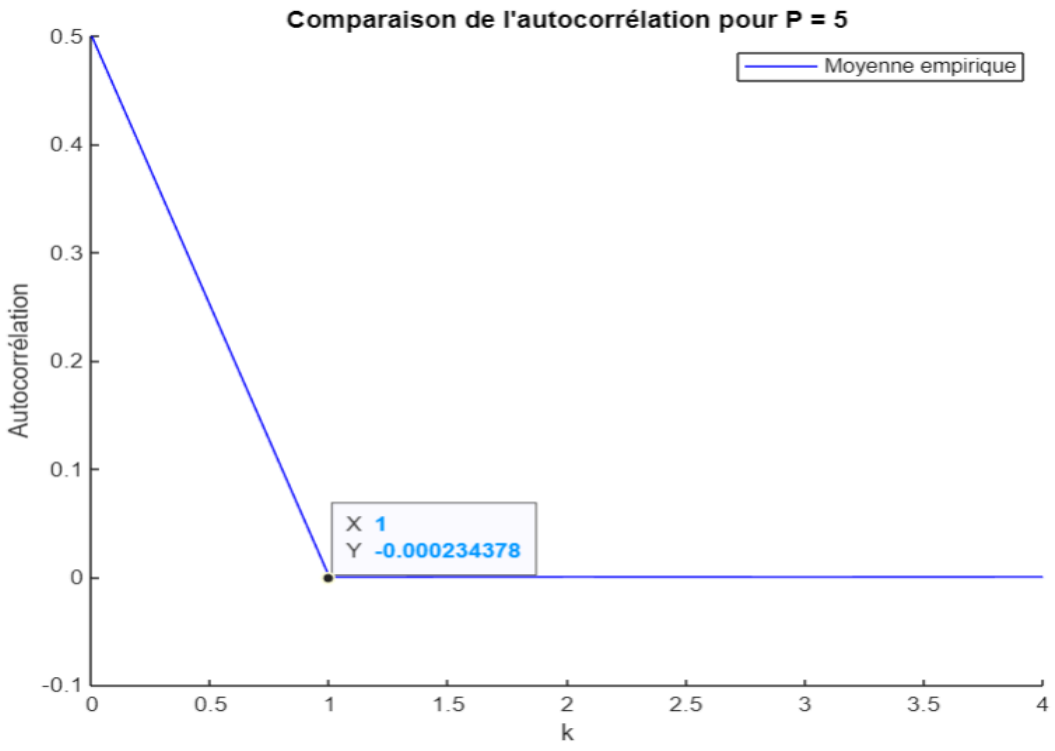


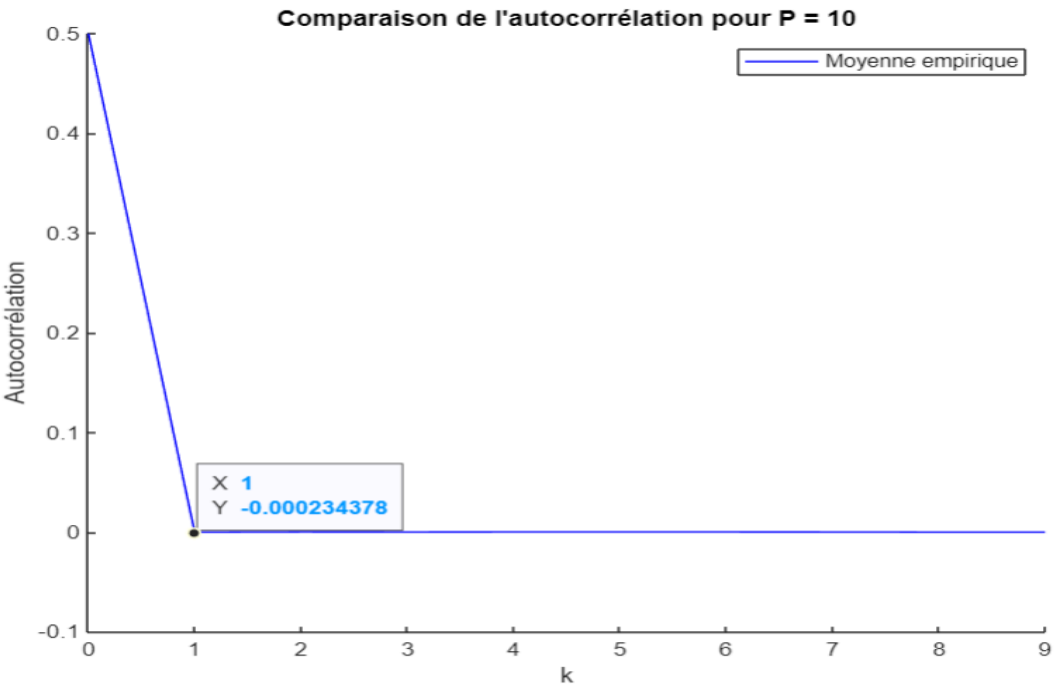
$$Pv = \sigma^2 + m^2 = 0,01 + \left(\frac{1}{3}\right)^2 = 0,121$$

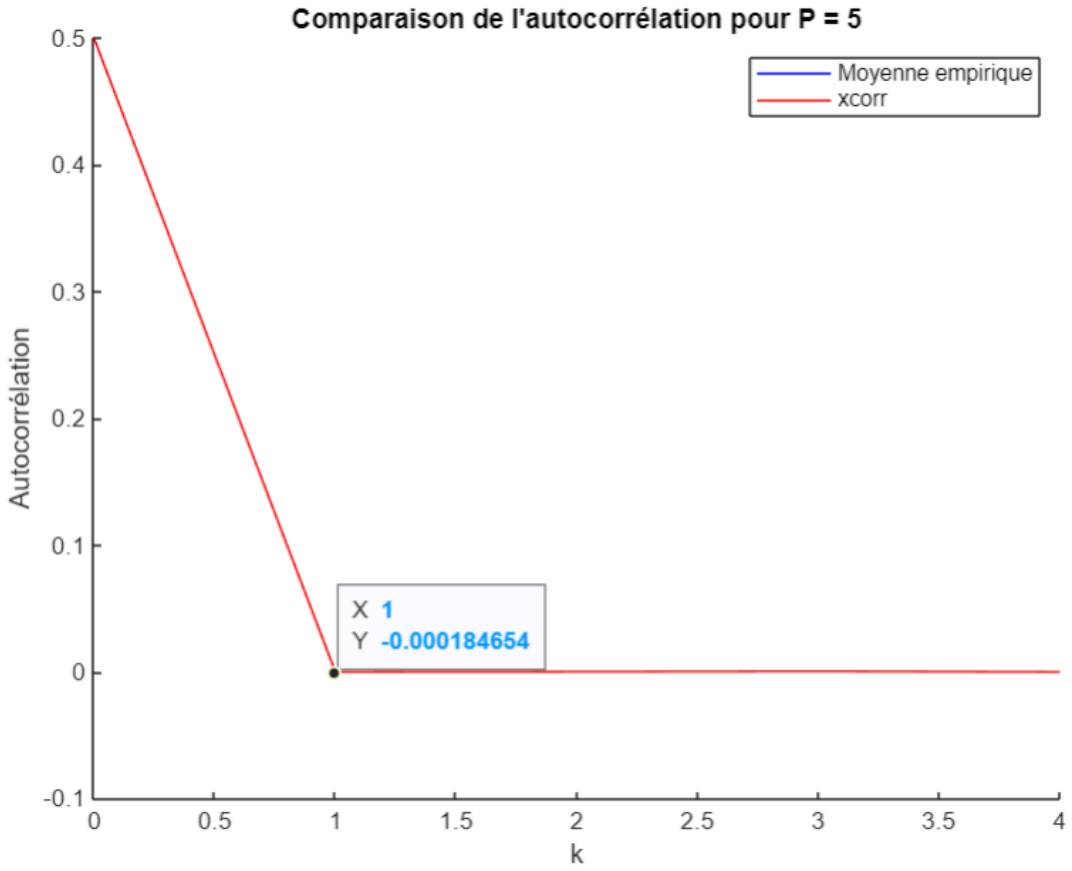
EX4:

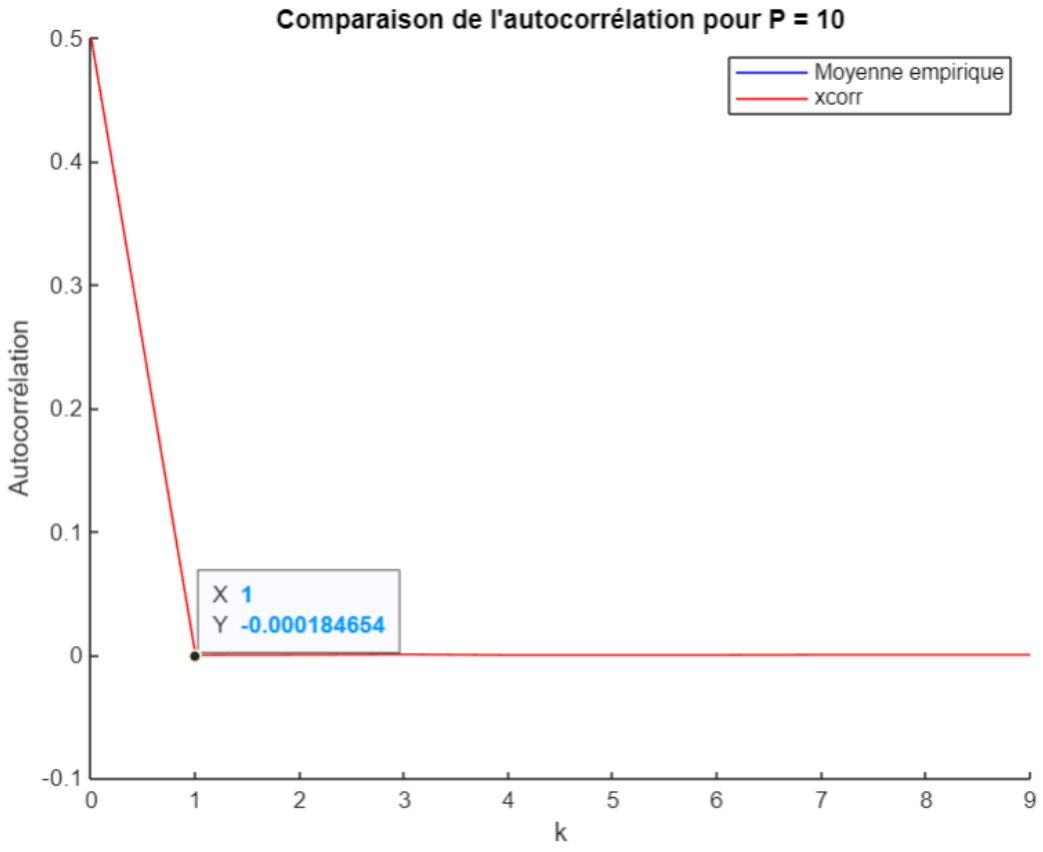
```
clear all;
close all;
% Paramètres
N = 10000;        % Nombre d'échantillons
P1 = 5;           % Valeur de P = 5
P2 = 10;          % Valeur de P = 10
Nrea = 100;      % Nombre de réalisations
w = 0.05 * pi;    % Fréquence
n_vec = 1:N;      % Vecteur des échantillons
% Génération du signal avec phi ~ U([0, 2*pi])
Dmat = [];
for j = 1:Nrea
   phi_vec = 2 * pi * rand(1, N);  % phi ~ U([0, 2*pi])
   d_vec = sin(n_vec * w + phi_vec); % Signal sinusoidal avec phase aléatoire
   Dmat = [Dmat; d_vec];            % Stocker chaque réalisation dans Dmat
end
% Autocorrélation empirique
P_values = [P1, P2];
for i = 1:length(P_values)
   P = P_values(i);
   r_empirique = zeros(1, P); % Initialisation de r_empirique
   for k = 0:P-1
       % Pour chaque k, calculez l'autocorrélation empirique
       % Assurez-vous que les indices sont valides pour chaque k
       if k == 0
           r_empirique(k+1) = mean(mean(Dmat.^2)); % Cas k=0, moyenne des
carrés
       else
           r_empirique(k+1) = mean(mean(Dmat(:, 1:N-k) .* Dmat(:, k+1:N))); %
Moyenne sur toutes les réalisations
       end
   end

   % Utilisation de la fonction xcorr pour chaque réalisation séparément
   r_xcorr = zeros(1, P);
   for j = 1:Nrea
       r_xcorr_temp = xcorr(Dmat(j, :), P-1, 'biased'); % Autocorrélation
pour une réalisation
```

```
        r_xcorr = r_xcorr + r_xcorr_temp(P:end); % Accumuler les valeurs de
xcorr
    end
    r_xcorr = r_xcorr / Nrea;  % Moyenne sur toutes les réalisations
    % Comparaison graphique dans une seule figure
    figure;
    hold on;
    plot(0:P-1, r_empirique, 'b', 'DisplayName', 'Moyenne empirique');
    title(['Comparaison de l''autocorrélation pour P = ', num2str(P)]);
    xlabel('k');
    ylabel('Autocorrélation');
    legend;
    hold off;
end
```




Comparaison de l'autocorrélation pour P = 10
Moyenne empirique
Autocorrélation
k
X 1
Y -0.000234378


Comparaison de l'autocorrélation pour P = 5
Moyenne empirique
xcorr
Autocorrélation
k
X 1
Y -0.000184654



→on observe que les deux méthodes (moyenne empirique et xcorr) donnent des résultats très similaires, mais avec de légères différences dues aux biais de l'estimation.

théorique:

Rd(k)=E{ d(n) * d(n-k)}

Rd(k)=E{[sin(nw0+phi) * sin((n-k)w0+phi)}

Rd(k)=E{½ *[cos(kw0)−cos(2nw0-kw0+2phi)]}

Rd(k)=E{½ *cos(kw0)}−E{cos(2nw0-kw0+2phi)}

on a E{cos(2nw0-kw0+2phi)}=0

→Cela est dû au fait que la fonction cosinus, lorsqu'on l'intègre sur une période complète de phi, a une espérance de zéro.

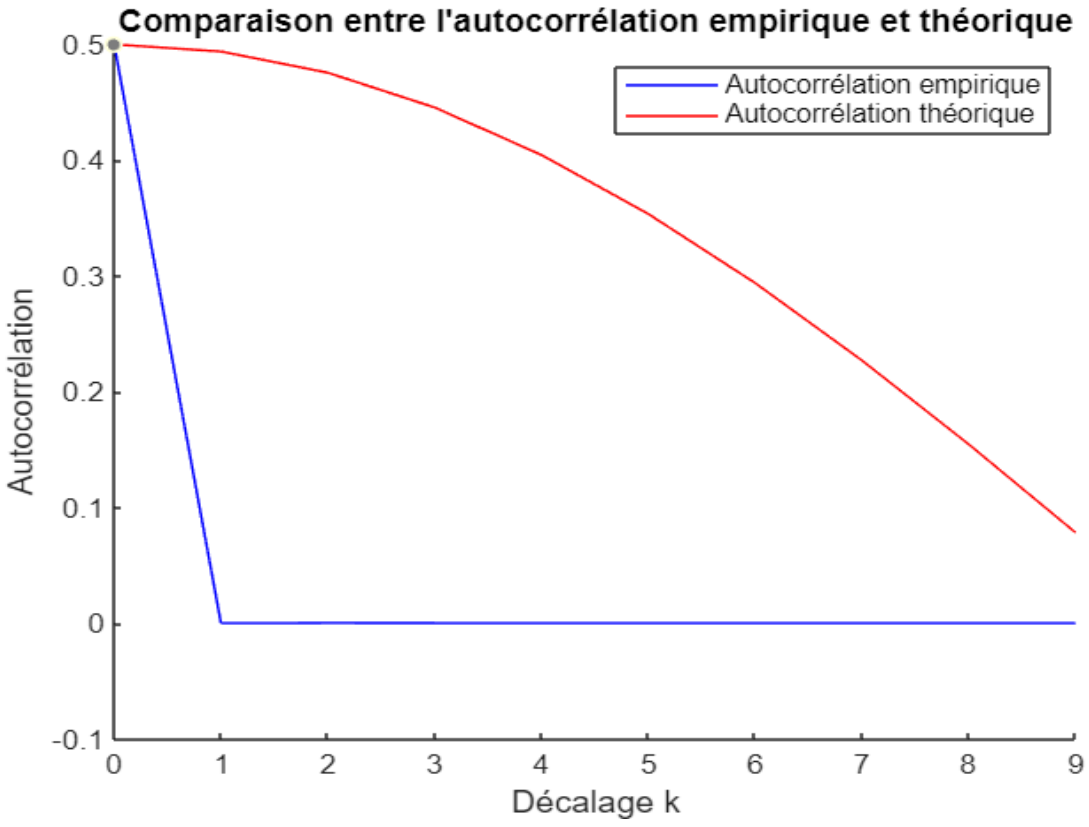
donc

Rd(k)=E{½ *cos(kw0)} = ½ *cos(kw0)

Rd(k)= ½ *cos(kw0)

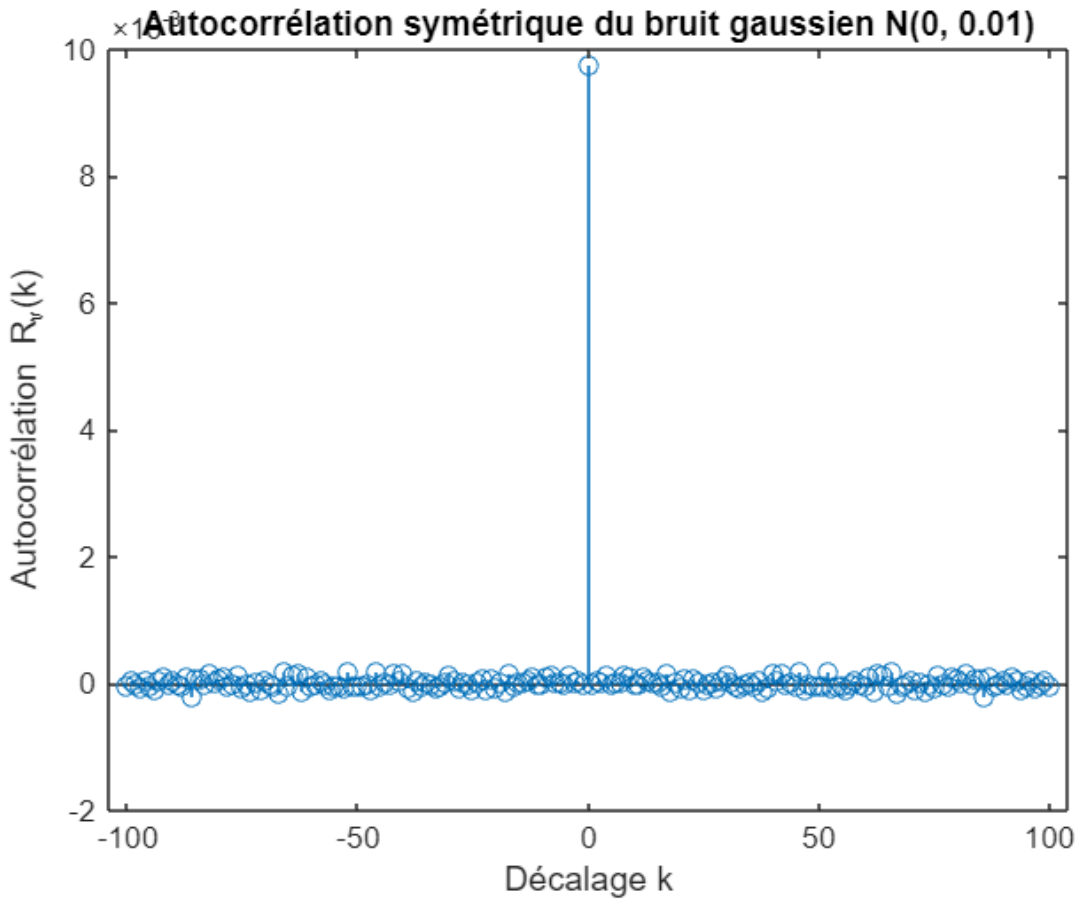
→l'autocorrélation théorique décroît de manière sinusoidale

→Avec un grand nombre de réalisations et une longueur de signal suffisante, l'autocorrélation empirique(pratique) converge vers l'autocorrélation théorique.



EX5:

```
% Paramètres
N = 10000;           % Nombre d'échantillons
P = 100;             % Valeur de P pour l'autocorrélation
sigma_v = sqrt(0.01);  % Écart-type du bruit
mu_v = 0;            % Moyenne du bruit
% Génération du bruit gaussien N(0, 0.01)
v = sigma_v * randn(1, N) + mu_v;
% Calcul de l'autocorrélation symétrique avec xcorr
r_v = xcorr(v, P, 'biased');
% Affichage de l'autocorrélation
figure;
stem(-P:P, r_v);
xlabel('Décalage k');
ylabel('Autocorrélation R_v(k)');
title('Autocorrélation symétrique du bruit gaussien N(0, 0.01)');
```



observation:

covariance = autocorrélation car mv=0(v(n) centré)

Valeur à k=0 : L'autocorrélation est égale à la variance du bruit, sigma^2=0.01.

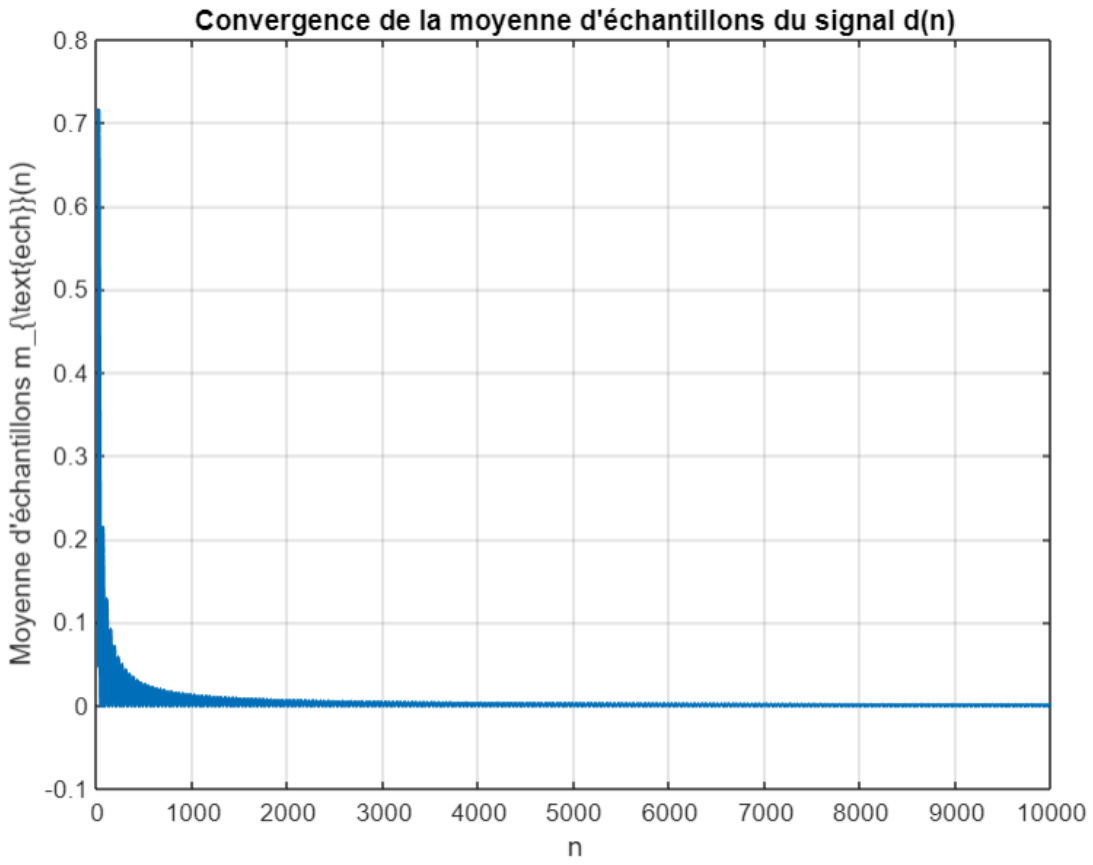
Valeurs pour k≠0k : L'autocorrélation est proche de 0 pour tous les autres décalages.

→ l'autocorrélation de v(n) présente une impulsion à k=0 et est proche de 0 (nulle) pour k≠0k donc le signal peut être considéré comme un **bruit blanc**.

EX6:

```
% Paramètres
N = 10000;        % Nombre d'échantillons
w0 = 0.05 * pi;   % Fréquence
phi = 2 * pi * rand();  % Phase aléatoire phi ~ U([0, 2*pi])
n_vec = 1:N;      % Vecteur des indices d'échantillons
% Génération du signal d(n) = sin(nw0 + phi)
d = sin(n_vec * w0 + phi);
% Calcul de la moyenne des échantillons
m_ech = zeros(1, N);
for n = 1:N
   m_ech(n) = mean(d(1:n));  % Moyenne des échantillons pour n = 1, ..., N
end
% Affichage de la moyenne d'échantillons
figure;
plot(n_vec, m_ech, 'LineWidth', 1.5);
xlabel('n');
ylabel('Moyenne d''échantillons m_{\text{ech}}(n)');
```

```
title('Convergence de la moyenne d''échantillons du signal d(n)');
grid on;
```



moyenne théorique:

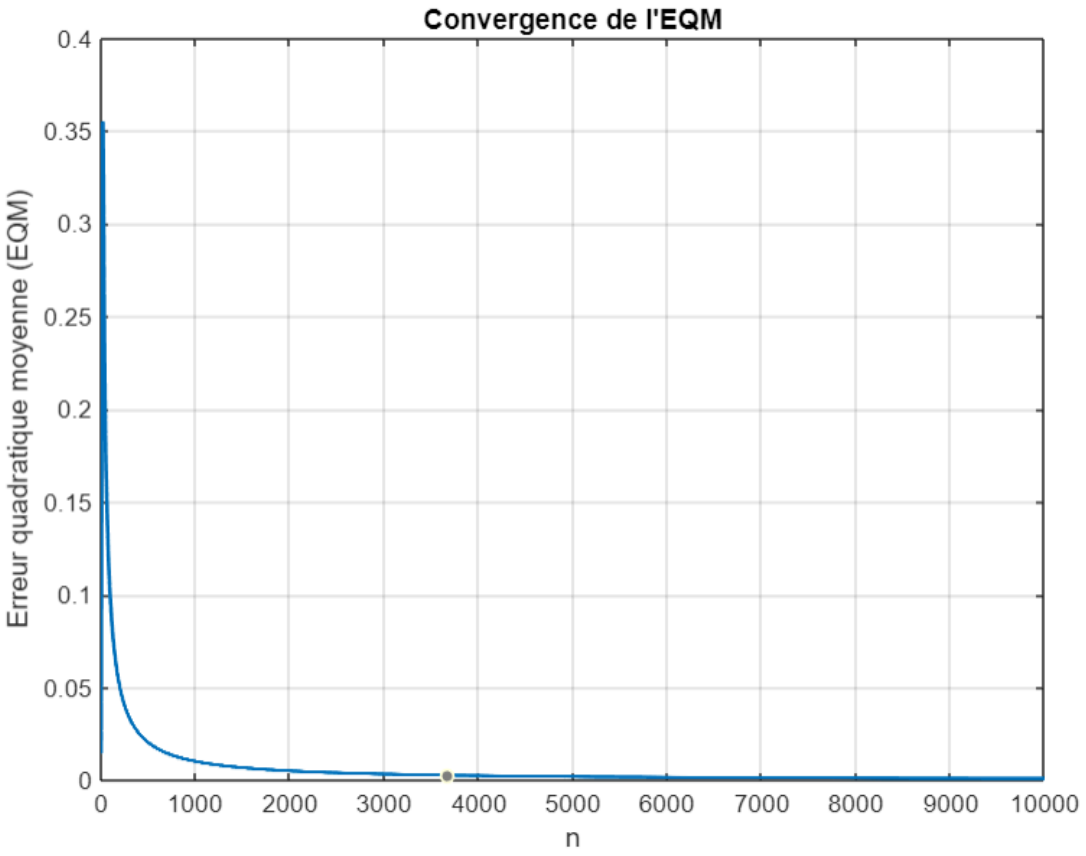
E[sin(nw0+phi)]=0

car phi est une phase aléatoire uniforme sur [0,2π], rendant la moyenne de sin nulle

Description de la courbe : Au début, la moyenne fluctue beaucoup, car il y a peu d'échantillons. En augmentant n, elle converge lentement vers zéro.

→donc le signal est ergodique, car la moyenne empirique rejoint la moyenne théorique.

```
% Étape 2 : Calcul et affichage de l'EQM
eqm = zeros(1, N);  % Erreur quadratique moyenne (EQM)
for n = 1:N
   eqm(n) = mean((m_ech(1:n) - 0).^2);  % Approximation de l'EQM
end
% Affichage de l'EQM
figure;
plot(n_vec, eqm, 'LineWidth', 1.5);
xlabel('n');
ylabel('Erreur quadratique moyenne (EQM)');
title('Convergence de l''EQM');
grid on;
```



- L'EQM eqm(n) mesure la précision avec laquelle la moyenne empirique converge vers la moyenne théorique.

→Elle diminue au fur et à mesure que n augmente, ce qui montre que la moyenne d'échantillons devient de plus en plus précise (proche de moyenne théorique 0).

un signal audio:

EX7:

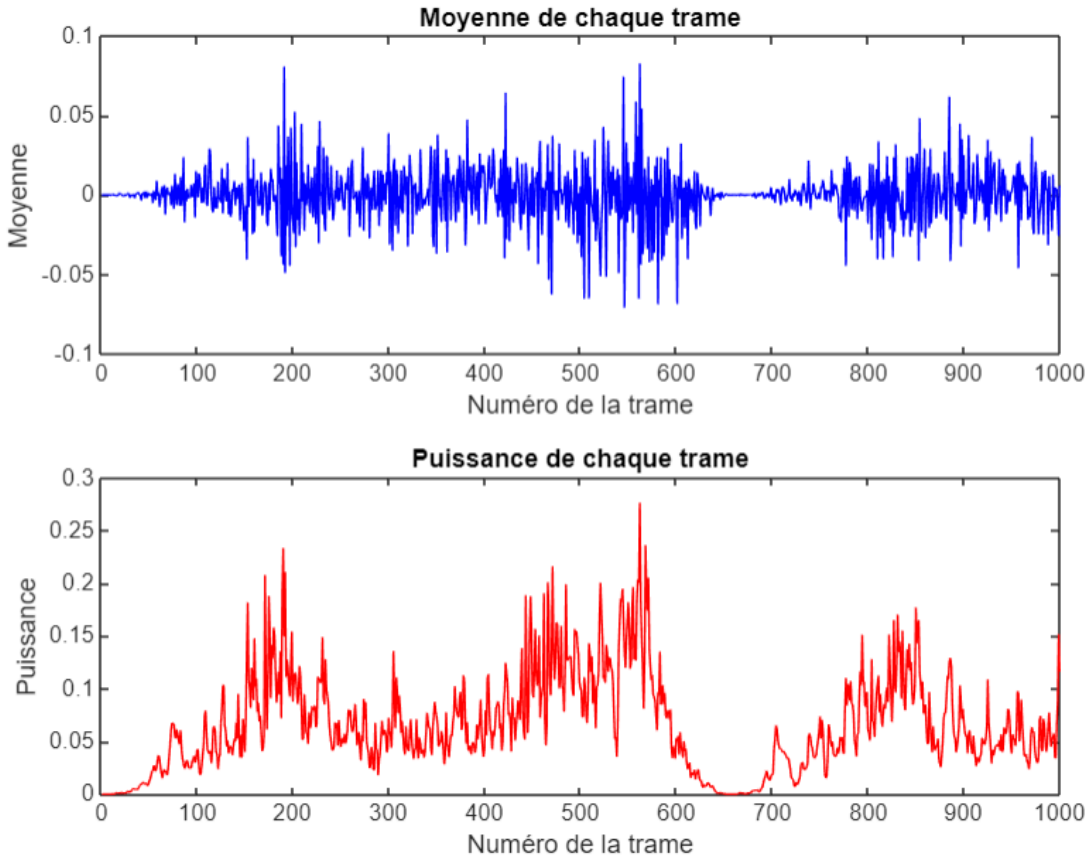
```
% Ouvrir le fichier audio
[x, Fs] = audioread('BREAKING NEWS.mp3');  % Remplacez par votre fichier
```

```
% Paramètres de segmentation
Lt = 1024;      % Longueur de chaque trame
Nt = 1000;      % Nombre de trames

% Extraire la partie du signal qui contient les Nt trames
vec_sig = x(1:Nt*Lt);  % Prendre Nt*Lt premiers échantillons du signal

% Segmenter le signal en trames et stocker dans une matrice
Xmat = reshape(vec_sig, Lt, Nt);
```

EX8:

```
% Ouvrir le fichier audio
[x, Fs] = audioread('BREAKING NEWS.mp3');  % Remplacez par votre fichier
% Paramètres de segmentation
Lt = 1024;      % Longueur de chaque trame
Nt = 1000;      % Nombre de trames
% Extraire la partie du signal qui contient les Nt trames
vec_sig = x(1:Nt*Lt);  % Prendre Nt*Lt premiers échantillons du signal
% Segmenter le signal en trames et stocker dans une matrice
Xmat = reshape(vec_sig, Lt, Nt);
moyenne = zeros(1, Nt);   % Vecteur de moyenne pour chaque trame
puissance = zeros(1, Nt); % Vecteur de puissance pour chaque trame
% Calculer la moyenne et la puissance pour chaque trame
for n = 1:Nt
   moyenne(n) = mean(Xmat(:,n));  % Moyenne de la trame n
   puissance(n) = mean(Xmat(:,n).^2);  % Puissance de la trame n
end
% Afficher la moyenne de chaque trame
figure;
subplot(2,1,1);  % Première sous-figure
plot(1:Nt, moyenne, 'b');
title('Moyenne de chaque trame');
xlabel('Numéro de la trame');
ylabel('Moyenne');
% Afficher la puissance de chaque trame
subplot(2,1,2);  % Deuxième sous-figure
```

```
plot(1:Nt, puissance, 'r');
title('Puissance de chaque trame');
xlabel('Numéro de la trame');
ylabel('Puissance');
```



interprétation:

- **Moyenne** : Le signal semble fluctuer autour de zéro, mais avec quelques variations plus importantes à certains endroits, ce qui suggère une certaine non-stationnarité.
- **Puissance** : Le signal présente des variations significatives dans sa puissance, ce qui indique que des événements acoustiques ou des changements d'amplitude se produisent dans le signal.

La non-stationnarité est confirmée par ces variations importantes dans la puissance.

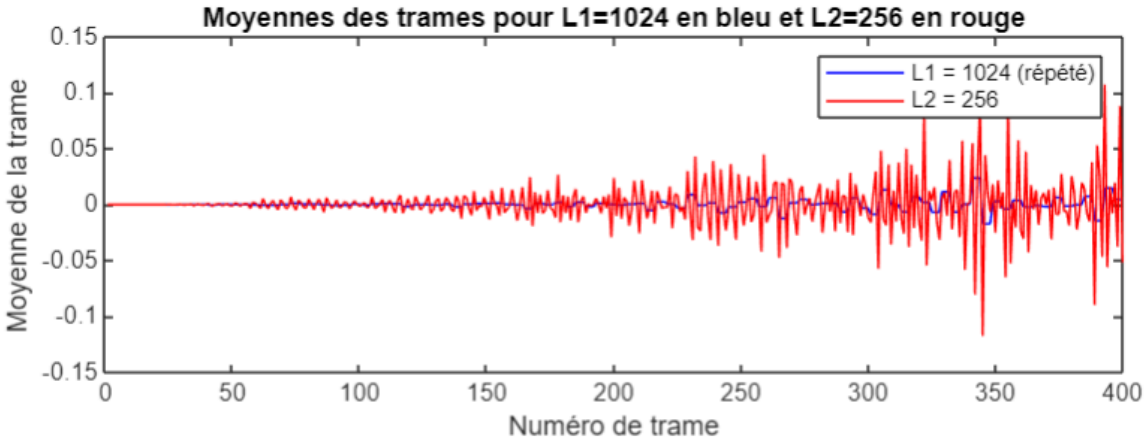
En fait le signal analysé est un enregistrement audio,qu'il contienne des sons ou des événements acoustiques variables en intensité, ce qui se traduit par les variations observées dans la puissance.

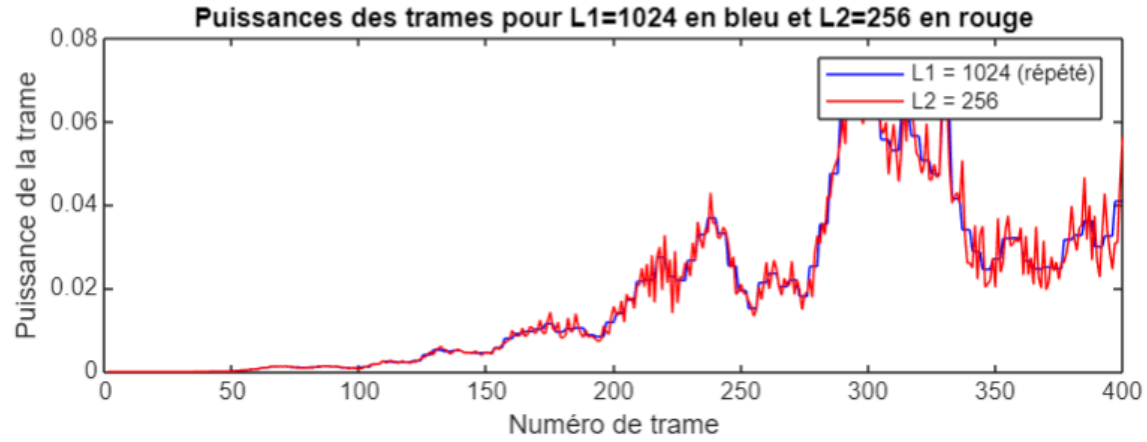
→Cela montre que le signal n'est pas purement stationnaire, et qu'il est donc important de tenir compte de ses variations temporelles pour bien comprendre sa nature.

EX9:

```
clear all;
close all;
% Paramètres
N = 100;% Nombre de trames à récupérer
L1 = 1024; % Longueur de trame L1
L2 = 256;  % Longueur de trame L2 (4 fois plus de trames pour L2)
% Chargement du fichier audio
[x, Fs] = audioread('BREAKING NEWS.mp3');  % Remplacez par votre fichier
% Vérifier si le signal est assez long pour L1 et L2
if length(x) < N * L1
   error('Le signal est trop court pour L1 = 1024.');
end
if length(x) < N * L2 * 4 % Prendre en compte qu'il faut 4 fois plus de
trames pour L2
   error('Le signal est trop court pour L2 = 256.');
end
% Extraction du signal segmenté pour les trames L1 = 1024 et L2 = 256
vec_sig1 = x(1:N * L1); % Signal pour L1
vec_sig2 = x(1:4 * N * L2); % Signal pour L2
% Segmenter le signal pour L1 = 1024
Xmat1 = reshape(vec_sig1, L1, N);
% Segmenter le signal pour L2 = 256 (4 fois plus de trames)
```

```
Xmat2 = reshape(vec_sig2, L2, 4 * N);
% Calcul des moyennes des trames
moyennes_trames1 = mean(Xmat1);
moyennes_trames2 = mean(Xmat2);
% Calcul des puissances des trames
puissances_trames1 = mean(Xmat1.^2);
puissances_trames2 = mean(Xmat2.^2);
% Répéter les résultats pour L1 = 1024, 4 fois pour une même trame
moyennes_trames1_repeated = kron(moyennes_trames1, ones(1, 4));
puissances_trames1_repeated = kron(puissances_trames1, ones(1, 4));
% Numéros de trames pour affichage (L2 aura 4 fois plus de trames)
num_trame = 1:N*4;
% Affichage des résultats
% Moyenne des trames
figure;
subplot(2, 1, 1);
plot(num_trame, moyennes_trames1_repeated, 'b');
hold on;
plot(num_trame, moyennes_trames2, 'r');
xlabel('Numéro de trame');
ylabel('Moyenne de la trame');
title('Moyennes des trames pour L1=1024 en bleu et L2=256 en rouge');
legend('L1 = 1024 (répété)', 'L2 = 256');
% Puissance des trames
subplot(2, 1, 2);
plot(num_trame, puissances_trames1_repeated, 'b');
hold on;
plot(num_trame, puissances_trames2, 'r');
xlabel('Numéro de trame');
ylabel('Puissance de la trame');
title('Puissances des trames pour L1=1024 en bleu et L2=256 en rouge');
legend('L1 = 1024 (répété)', 'L2 = 256');
```





interprétation:

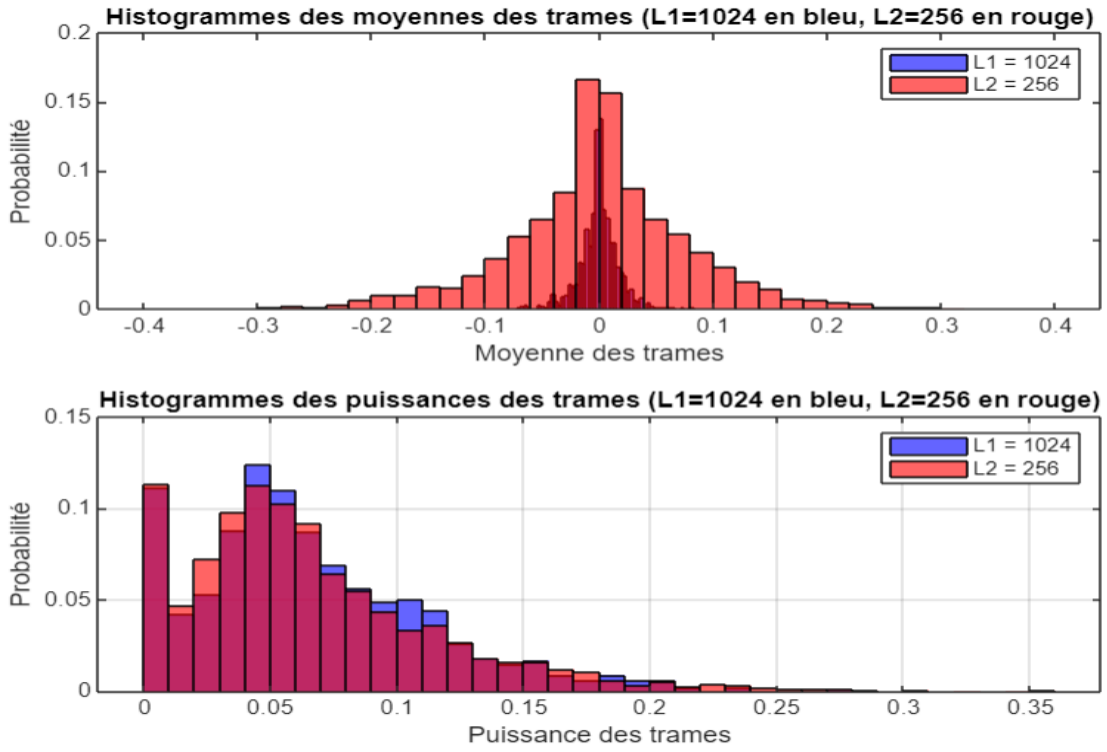
- **Moyenne des trames** : on observe que les moyennes des trames de `Lt1 = 1024` sont répétées pour correspondre à la segmentation de `Lt2 = 256`. En général, la moyenne des trames ne change pas significativement entre les deux tailles de trame, mais `L2 = 256` permet d'observer plus de fluctuations sur des intervalles plus courts.
- **Puissance des trames** : La puissance des trames montre également des variations similaires pour les deux tailles de trame, mais avec une meilleure détection des variations locales du signal pour `L2 = 256`.

Ainsi, on peut observer que la segmentation en petites trames permet de mieux capturer les variations rapides du signal.

ex10:

```
clear all;
close all;
% Paramètres
N = 1000;  % Nombre de trames
L1 = 1024;  % Longueur de trame 1
L2 = 256;   % Longueur de trame 2
[x, Fs] = audioread('BREAKING NEWS.mp3');  % Remplacez par votre fichier
% Segmenter le signal pour L1 = 1024 et L2 = 256
vec_sig = x(1:N * L1);
Xmat1 = reshape(vec_sig, L1, N);  % Segmenter pour L1
Xmat2 = reshape(vec_sig, L2, 4 * N);  % Segmenter pour L2 (4 fois plus de
trames)
% Calculez les moyennes et puissances des trames
moyennes_trames1 = mean(Xmat1);
moyennes_trames2 = mean(Xmat2);
puissances_trames1 = mean(Xmat1.^2);
puissances_trames2 = mean(Xmat2.^2);
% Répéter les moyennes et puissances pour L1 pour correspondre au nombre de
trames L2
moyennes_trames1_repeated = kron(moyennes_trames1, [1, 1, 1, 1]);
puissances_trames1_repeated = kron(puissances_trames1, [1, 1, 1, 1]);
% Affichage des histogrammes normalisés
% Histogramme des moyennes
figure;
subplot(2, 1, 1);
histogram(moyennes_trames1_repeated, 'Normalization', 'probability',
'FaceColor', 'b');
hold on;
histogram(moyennes_trames2, 'Normalization', 'probability', 'FaceColor',
'r');
xlabel('Moyenne des trames');
```

```
ylabel('Probabilité');
title('Histogrammes des moyennes des trames (L1=1024 en bleu, L2=256 en rouge)');
legend('L1 = 1024', 'L2 = 256');
% Histogramme des puissances
subplot(2, 1, 2);
histogram(puissances_trames1_repeated, 'Normalization', 'probability', 'FaceColor', 'b');
hold on;
histogram(puissances_trames2, 'Normalization', 'probability', 'FaceColor', 'r');
xlabel('Puissance des trames');
ylabel('Probabilité');
title('Histogrammes des puissances des trames (L1=1024 en bleu, L2=256 en rouge)');
legend('L1 = 1024', 'L2 = 256');
grid on;
```



interprétation:

- **Histogramme des moyennes** :

  →pour des trames plus courtes (Lt′=256), il y a plus de variabilité dans les moyennes car elles capturent moins d'informations. Avec des trames plus longues (Lt=1024L), les moyennes sont plus stables car elles couvrent une plus grande portion du signal, réduisant ainsi la variabilité.

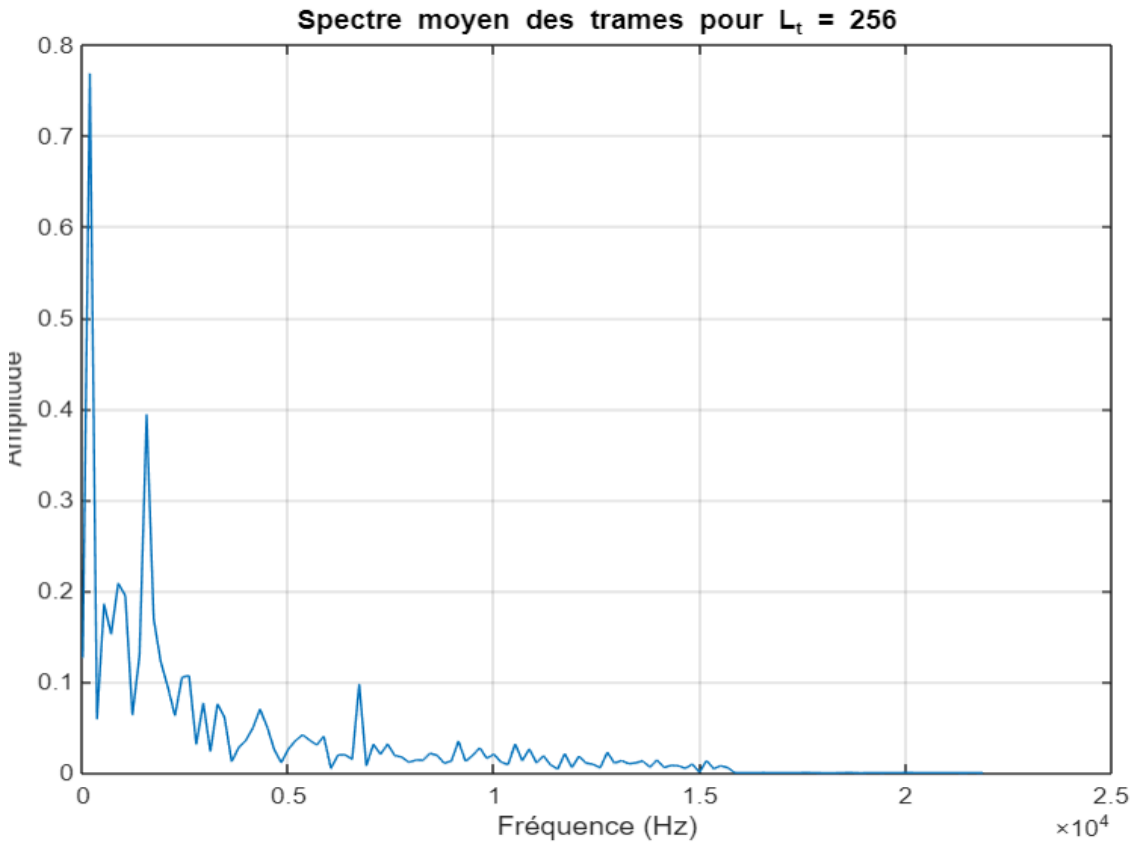
- **Histogramme des puissances** :

  →Les trames plus longues (L1 = 1024) fournissent des résultats plus lissés, car elles englobent plus d'échantillons, tandis que les trames plus courtes (L2 = 256) permettent de capturer les variations plus fines du signal, ce qui entraîne une plus grande dispersion des valeurs de puissance.

  10:

```
% Initialiser la matrice des spectres
spec_mat = zeros(Lt2, N);
% Calculer la FFT de chaque trame et stocker les résultats dans spec_mat
for i = 1:N
   spec_mat(:, i) = fft(Xmat2(:, i));  % FFT sur chaque trame
end
% Prendre seulement la moitié des fréquences (symétrie de la FFT)
spec_mat = spec_mat(1:Lt2/2, :);
% Fréquences associées (axe fréquentiel)
frequencies = (0:Lt2/2-1) * Fs / Lt2;
% Afficher le spectre moyen des trames
figure;
plot(frequencies, abs(mean(spec_mat, 2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre moyen des trames pour L_t = 256');
```

```
grid on;
```



---

interprétation:

Les fréquences significatives ici sont majoritairement comprises dans la plage de basses fréquences (<1Hz), indiquant que le signal contient essentiellement des variations lentes ou des composants dominants de basse fréquence. Les composantes haute fréquence, bien qu'elles existent, semblent avoir une amplitude beaucoup plus faible.

Ex11:

on a Fs=44600

la FFT sur le signal de longueur Lt=256, donne Lt coefficients complexes. Cependant, la deuxième moitié des coefficients est une image miroir de la première moitié, donc seuls les Lt/2=128 premiers coefficients contiennent des informations de fréquence uniques.

→Cela signifie que les 128 premières composantes de la FFT donnent des informations sur les fréquences basses jusqu'à Fs/2 Hz.

ces composantes couvrent les fréquences allant de 0 à Fs/2F_s/2Fs/2 Hz, correspondant à la moitié de la bande de fréquences.