Compte rendu SAE22

Partie 1 : Prise en main de Matlab

II / Génération d'un signal sinusoïdal x(t)

1) Que représente la variable t ? Quelle est sa nature ?

t est un tableau et sa nature est une variable.

2) Que fait la ligne 10 ? Quelle est la nature de x ?

La ligne 10 signifie la génération de x(t) ; x aussi est une variable.

3) Combien de période du signal $x(t)$ seront calculées ?

Le nombre de période du signal est :

4) Quelle petite critique ou amélioration peut être apportée à ce programme ?

Voici la saisi de ce programme :

```
clc;
close;
f = 1000;
A = 5;
fe = 40000;
Te = 1/fe;
Ns = 200;
t=0:Te:(Ns-1)*Te;
x = A*cos(2*pi*f*t);
```

III / Affichage du signal x(t)

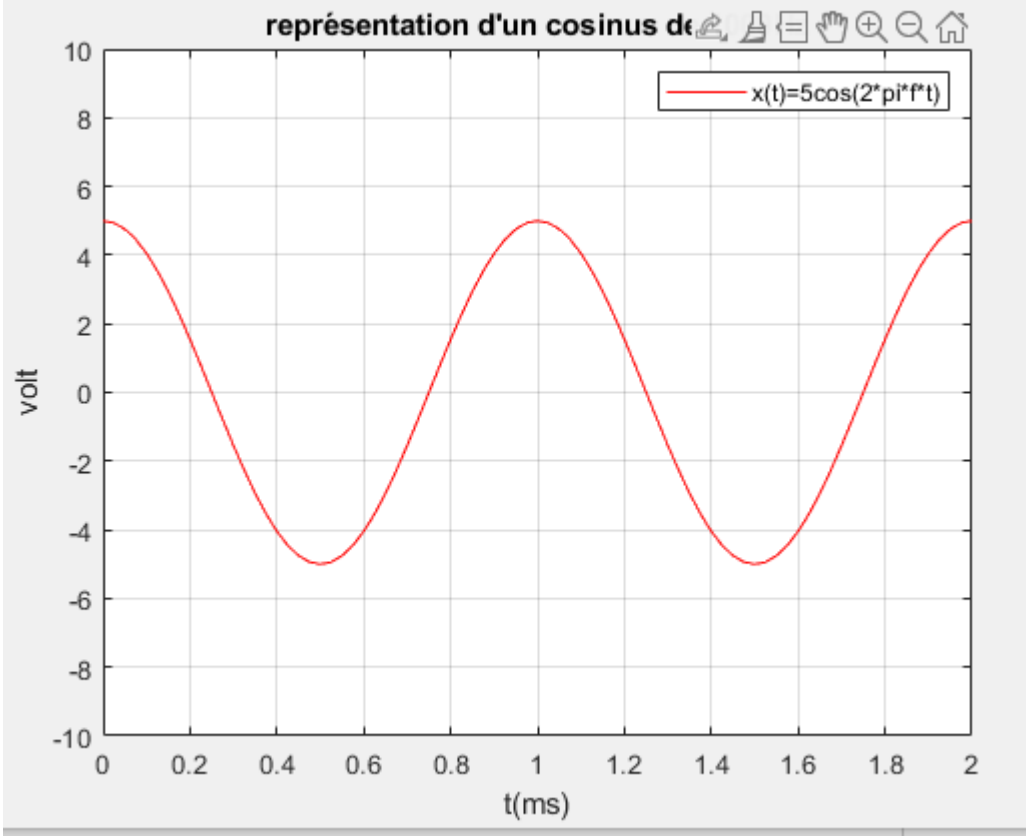
Chercher de l'aide sur la fonction « plot » puis compléter le programme suivant afin d'afficher

x(t) avec les informations suivantes :

- Couleur : rouge
- Mettre un titre : title
- Mettre une légende : legend
- Mettre le nom des axes sur la figure : xlabel et ylabel
- Fixer les axes suivants : de 0 à 2ms et de -10V à +10V : axis

Voici le code :

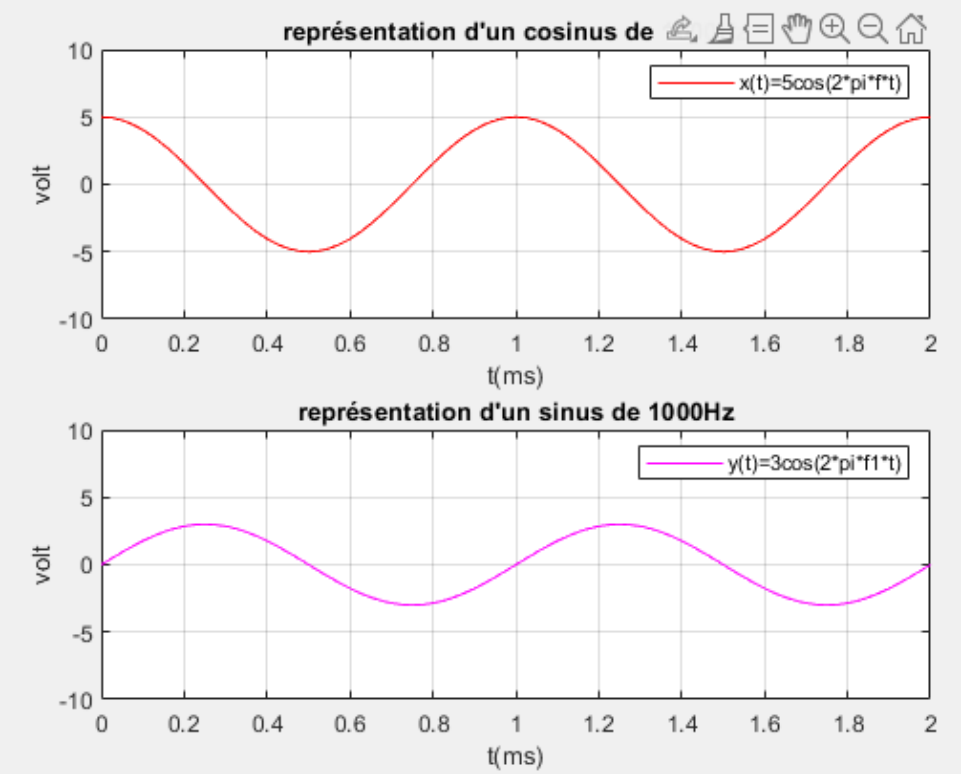
```
clc;
close;
clear;
f = 1000;
A = 5;
fe = 40000;
Te = 1/fe;
Ns = 200;
t=0:Te:(Ns-1)*Te;
x = A*cos(2*pi*f*t);
plot(t*1000,x,"r")
title("représentation d'un cosinus de 1000Hz")
xlabel("t(ms)")
ylabel("volt")
legend("x(t)=5cos(2*pi*f*t)")
grid on
axis ([0 2 -10 10])
```



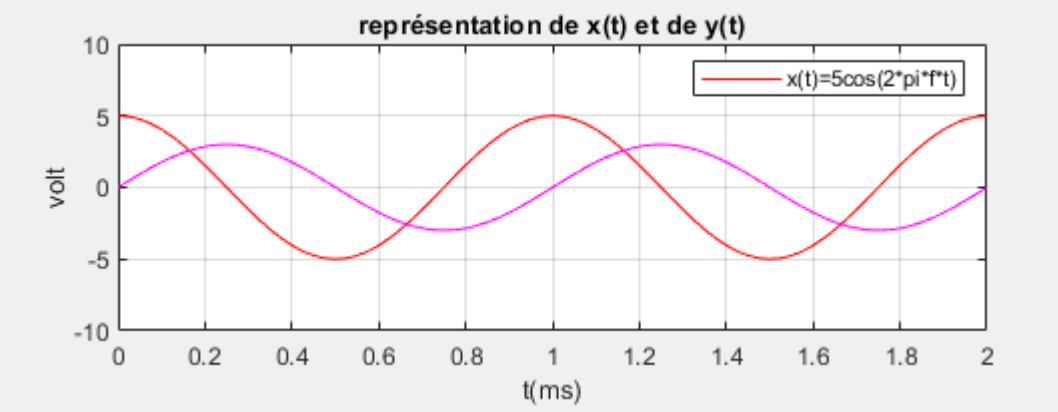
Compléter le programme afin de créer et d'afficher le signal $y(t)$ en bleu.

```
clc;
close;
clear;
f = 1000;
A = 5;
fe = 40000;
Te = 1/fe;
Ns = 200;
t=0:Te:(Ns-1)*Te;
x = A*cos(2*pi*f*t);
f1 = 1000;
A1 = 3;
y = A1*sin(2*pi*f1*t);
subplot(2,1,1)
plot(t*1000,x,"r")
title("représentation d'un cosinus de 1000Hz")
xlabel("t(ms)")
ylabel("volt")
legend("x(t)=5cos(2*pi*f*t)")
grid on
axis ([0 2 -10 10])
subplot(2,1,2)
plot(t*1000,y,"m")
title("représentation d'un sinus de 1000Hz")
```

```
xlabel("t(ms)")
ylabel("volt")
legend("y(t)=3cos(2*pi*f1*t)")
grid on
axis ([0 2 -10 10])
```



Modifier le programme afin d'afficher les deux courbes sur deux graphes sur la même figure et Obtenir la représentation suivante :



V / Affichage du spectre

1) Etudier la ligne 28 ainsi que la fonction « spectre.m » à laquelle elle fait appel

```
27      %calcul de X(f): spectre en amplitude de x(t)
28 -    [X f]=spectre(x,fe,Ns);  % appel à la fonction  spectre.
```

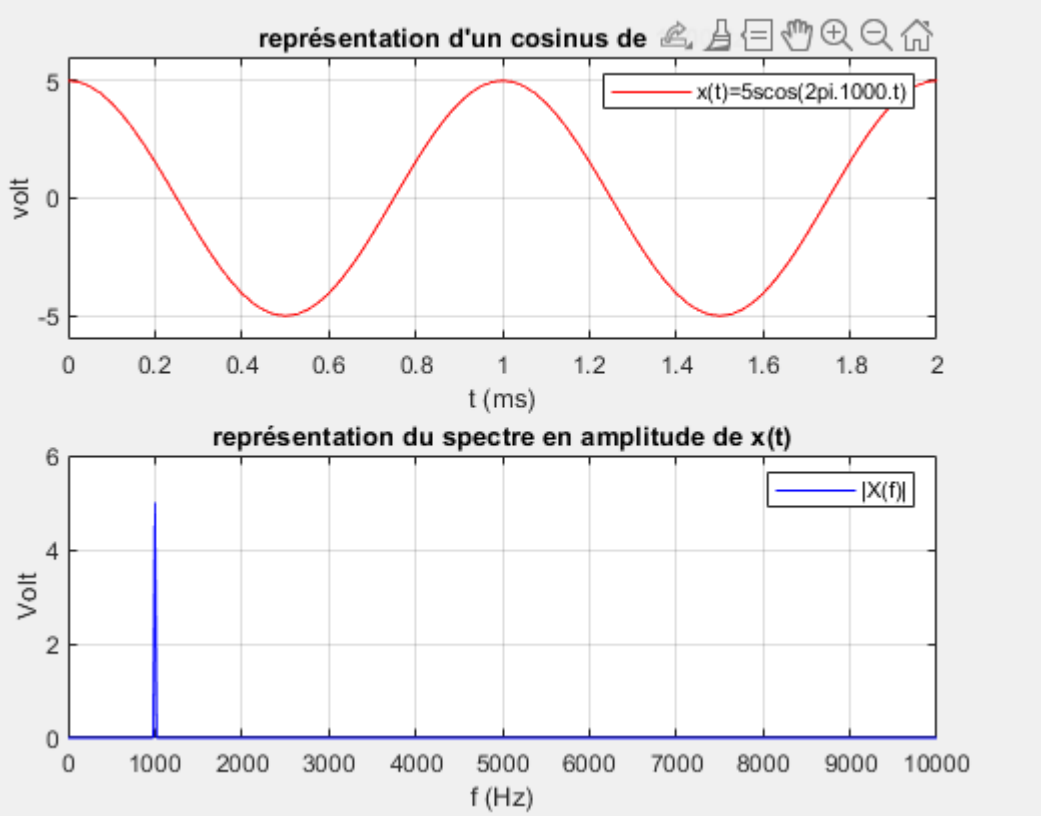
Cet extrait de code MATLAB semble être une fonction qui calcule le spectre en amplitude $X(f)$ d'un signal $x(t)$. Voici une analyse de ce script :

1. **%calcul de X(f): spectre en amplitude de x(t)**: Cette ligne est un commentaire qui explique brièvement ce que fait la fonction qui suit.
2. **[X f]=spectre(x,fe,Ns);**: Cette ligne appelle une fonction appelée **spectre** avec trois arguments :
    - **x**: Ce devrait être le signal $x(t)$ dont vous voulez calculer le spectre en amplitude.
    - **fe**: C'est la fréquence d'échantillonnage du signal $x(t)$.
    - **Ns**: C'est le nombre d'échantillons à utiliser pour calculer le spectre.
3. Le résultat de l'appel de la fonction **spectre** est assigné à deux variables :
    - **X**: C'est le spectre en amplitude du signal $x(t)$.
    - **f**: C'est le vecteur de fréquence correspondant aux composantes spectrales dans **X**.

Cela ressemble à un script qui encapsule le calcul du spectre en amplitude d'un signal pour faciliter son utilisation. Pour avoir une compréhension complète de ce script, il faudrait examiner le contenu de la fonction **`spectre`**.

2) Exécuter le script et valider le résultat.

On exécute le script et ça donne :



- Créer une fonction « spectredBV.m » afin d'afficher maintenant le spectre en dBV ou dBµV. On utilisera pour cela la fonction $log10(X)$. On ne calculera pas ici la valeur efficace pour le moment.

Pour cela, on crée la fonction spectredBV.m :

```
function [XdBV f]=spectredBV(signal,fe,Nech)
% L'appel de la fonction spectredBV
% cette fonction génére par FFT le spectre unilatéral en amplitude, X(f),
% du signal x(t). On crée aussi le vecteur fréquence, f.

df=fe/Nech;       % les échantillons fréquentiels sont espacés de fe/Nech
f=0:df:fe/2-df;  %création d'un vecteur fréquence constitué de Nech/2 points
            %répartis entre 0 et fe/2
```

%calcul et affichage du spectre du signal

X=fft(signal)/Nech;

X=[X(1) 2*X(2:Nech/2)]; %passage du spectre bilatéral au spectre unilatéral...

X=abs(X); % calcul du module pour afficher le spectre en amplitude.

XdBV=20*log10(X); % On converit X en dBV

end

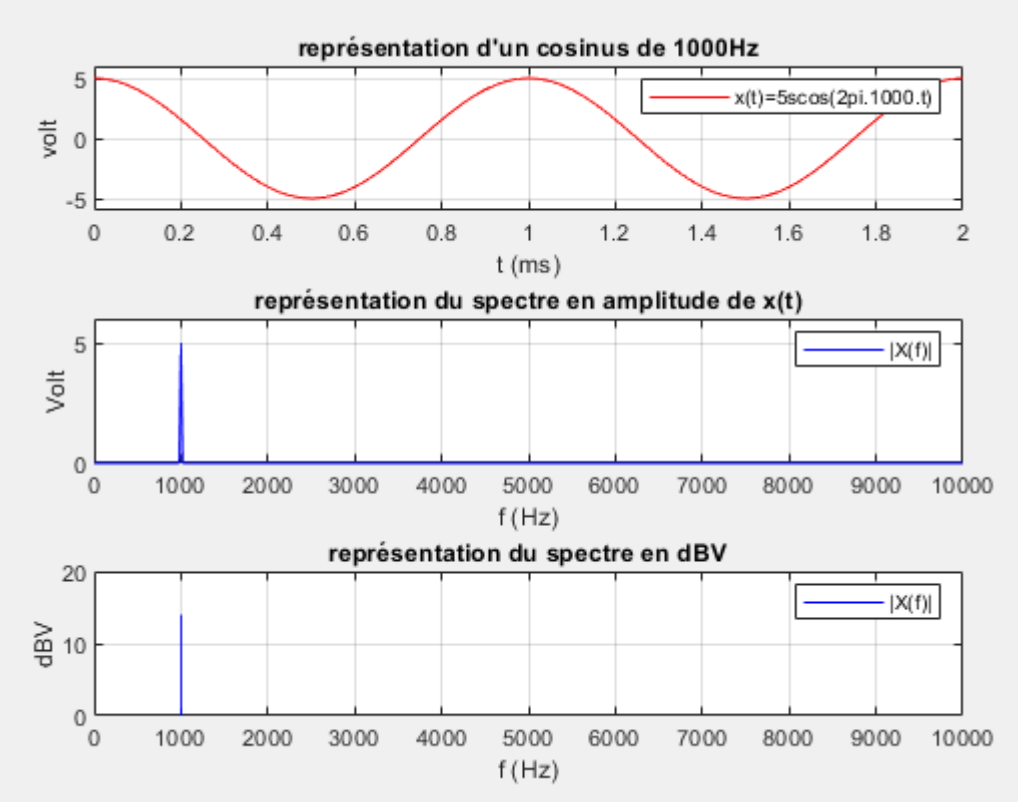
Ensuite sur le script affichage-spectre-sinus.m, on modifie les parties suivantes :

```
%calcul de X(f): spectre en amplitude de x(t)
[XdBV f]=spectredBV(x,fe,Ns);  % appel à la fonction  spectredBV.
```

Et :

```
% affichage du spectre en dBV
subplot(3,1,3)
```

Résultat :

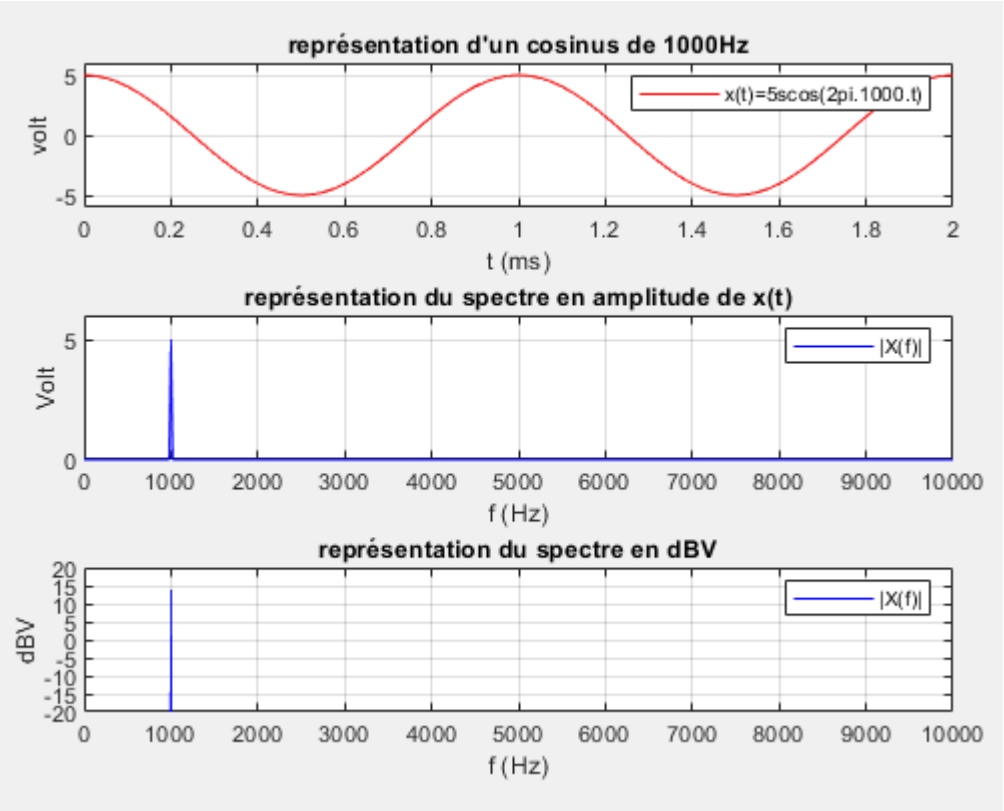


- Réaliser l'affichage comme montré ci-dessous, en mieux (penser aux noms des axes et autres …)

Pour cela, il faut rajouter yticks

```
yticks((-20:5:20))
axis([0 10000 -20 20])  %affichage entre -20 et 20kHz
```

Ainsi le résultat est :



IV Multiplication de deux signaux sinusoïdaux

- Générer les deux signaux x1(t) et x2(t).

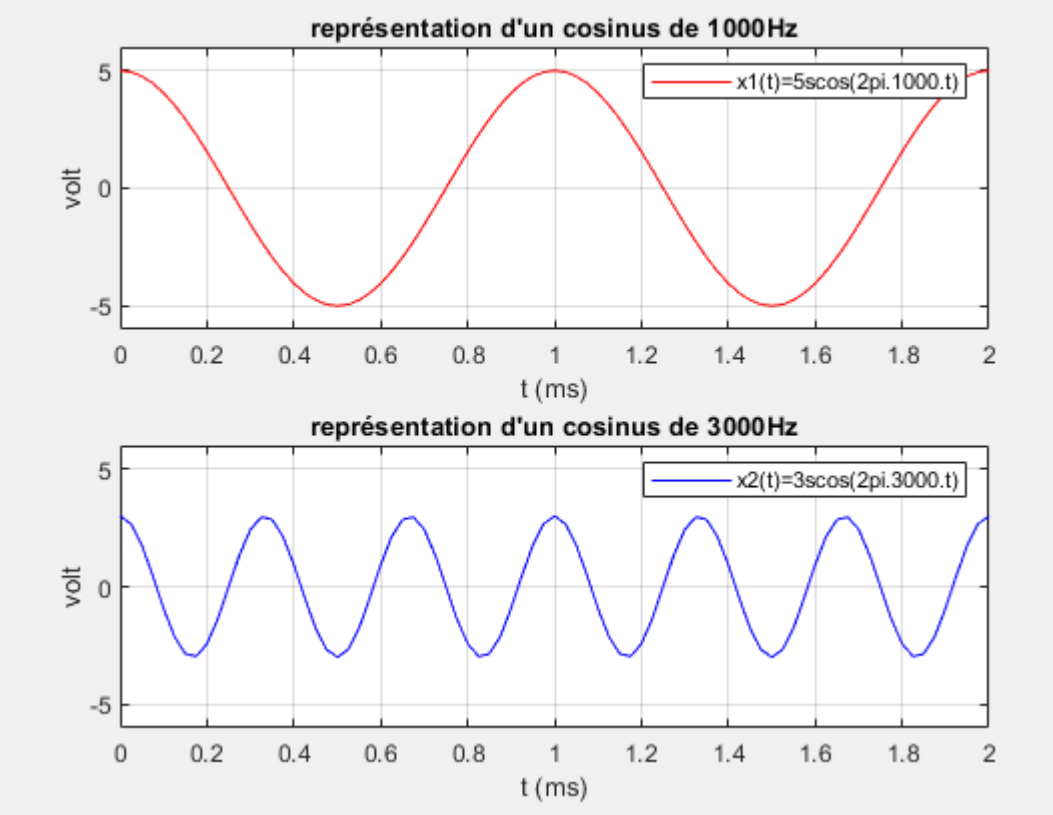
C:\Users\absow5\Downloads\affichage_spectre_cosinus.m

age_spectre_cosinus.m | spectredBV.m | spectre.m | +

```
clc;
close;    %ferme les anciennes figures

f1=1000;  %fréquence du signal x1(t)
f2=3000;  %fréquence du signal x2(t)
A1=5;  %amplitude de x1(t)
A2=3;  %amplitude de x2(t)

fe=40000; %fréquence d'échantillonnage
Te=1/fe;  %durée d'un échantillon
Ns=2000; %nombre d'échantillons

t=0:Te:(Ns-1)*Te;

x1=A1*cos(2*pi*f1*t); % génération de x1(t)

% Affichage de x1(t)
subplot(2,1,1)
plot(t*1000,x1,"r")
title('représentation d''un cosinus de 1000Hz')
xlabel('t (ms)')
ylabel('volt')
legend('x1(t)=5scos(2pi.1000.t)')
grid on
axis([0 2 -6 6])

x2=A2*cos(2*pi*f2*t); % génération de x2(t)
```

```
% Affichage de x2(t)
subplot(2,1,2)
plot(t*1000,x2,"b")
title('représentation d''un cosinus de 3000Hz')
xlabel('t (ms)')
ylabel('volt')
legend('x2(t)=3scos(2pi.3000.t)')
grid on
axis([0 2 -6 6])
```

L'affichage :



- Vérifier la taille des vecteurs x1 et x2 créés. Justifier.

X1 est un tableau d'une ligne et de 2000 colonnes.

X2 est un tableau d'une ligne et de 2000 colonnes aussi.

- Quelle est la différence entre les opérations suivantes sous Matlab : « * » et «.* » ?

En résumé, "*" est pour la multiplication matricielle conforme aux règles algébriques, tandis que ". *" est pour la multiplication élément de matrices de même taille.

Par la suite, on va utiliser <<. *>>

- Quelle est l’opération réalisée avec l’opérateur « .* » ? C’est-à-dire, en fonction des indices, quels sont les termes multipliés entre eux ? (z(1)=x1(??)*x2(??) ….)

L'opérateur « .* » en MATLAB ou en Octave représente la multiplication élément par élément (ou produit Hadamard) de deux matrices ou vecteurs de même dimension. Cela signifie que chaque élément du résultat est le produit des éléments correspondants des deux matrices ou vecteurs d'entrée. Par exemple, si \(A \) et \(B \) sont deux matrices ou vecteurs de même dimension, alors l'opération \(C = A .* B \) donnera une matrice \(C \) où chaque élément \(C(i, j) \) est le produit de \(A(i, j) \) et \(B(i, j) \). Prenons un exemple concret avec deux vecteurs : ```matlab x1 = [a1, a2, a3]; x2 = [b1, b2, b3]; z = x1 .* x2; ``` Le résultat \(z \) sera : ```matlab z = [a1*b1, a2*b2, a3*b3]; ``` Donc, les termes multipliés entre eux sont les éléments correspondants des vecteurs \(x1 \) et \(x2 \). En termes de notation générale pour une composante donnée \(i \) du vecteur \(z \) (que nous pouvons noter \(z(i) \)), cela peut s'écrire comme : \[z(i) = x1(i) \cdot x2(i) \] En résumé, avec l'opérateur « .* » : - Pour chaque indice \(i \), \(z(i) = x1(i) \cdot x2(i) \) - C'est une multiplication élément par élément entre les vecteurs ou matrices \(x1 \) et \(x2 \).

- Calculer le produit de ces deux signaux :

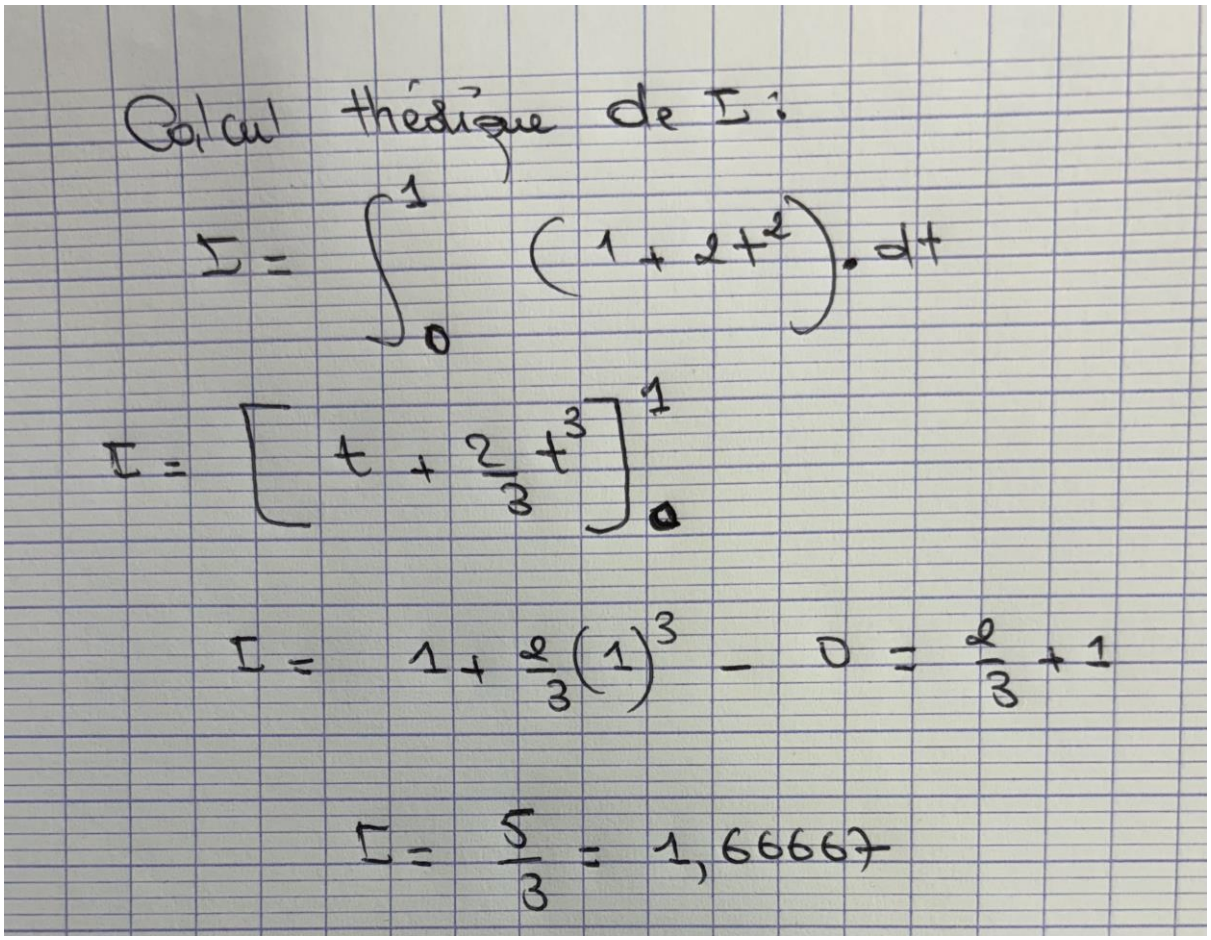
$$z = (A1 \times A2)/ 2 \times [\cos (2\pi (f1 + f2) \times t) + \cos(2\pi(f1 - f2) \times t)]$$

$$Z = (3 * 5)/2 * [\cos (2 * \pi * 4000 * t) + \cos(2 * \pi * 2000 * t)]$$

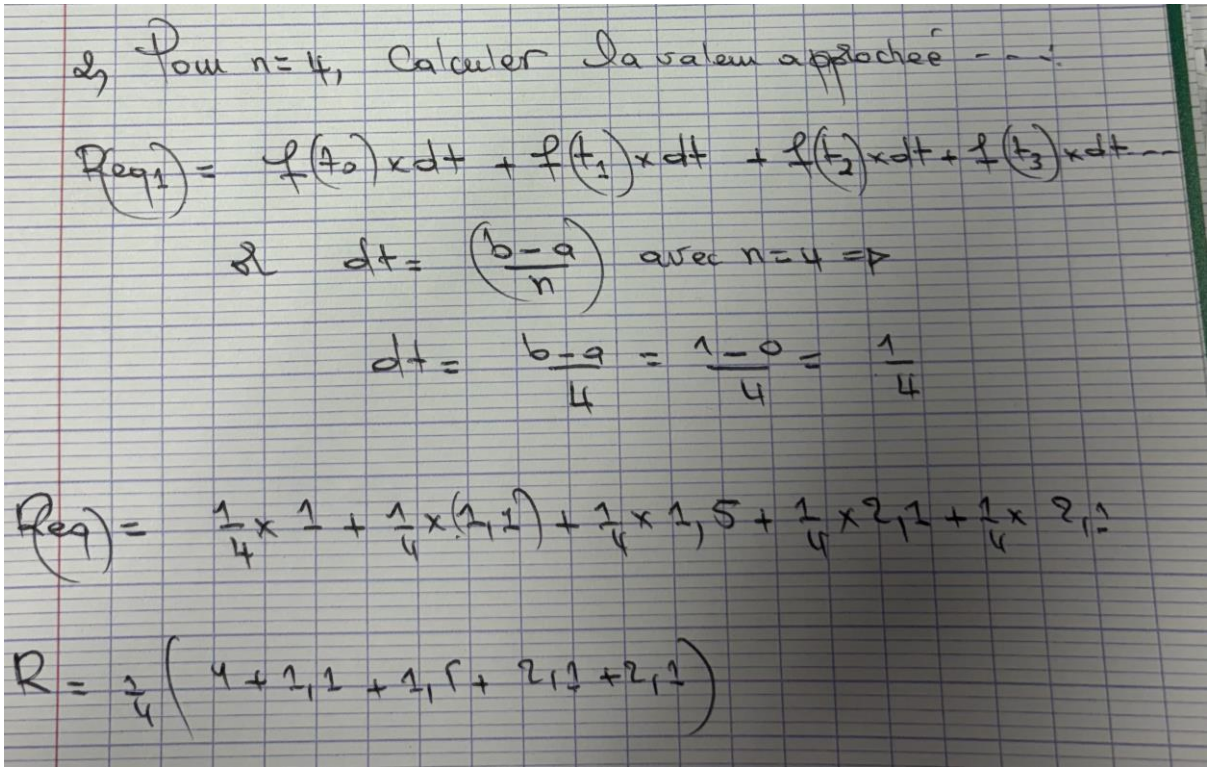
Partie 2 : Calcul numérique d'une intégrale

II Méthode des rectangles

1) Calculer théoriquement la valeur de I. Faire une application numérique à 10^{-5} près.



2) Pour **n=4**, calculer à l'aide d'un tableur la valeur approchée de I la méthode des rectangles, c'est-à-dire **R** (eq 1).



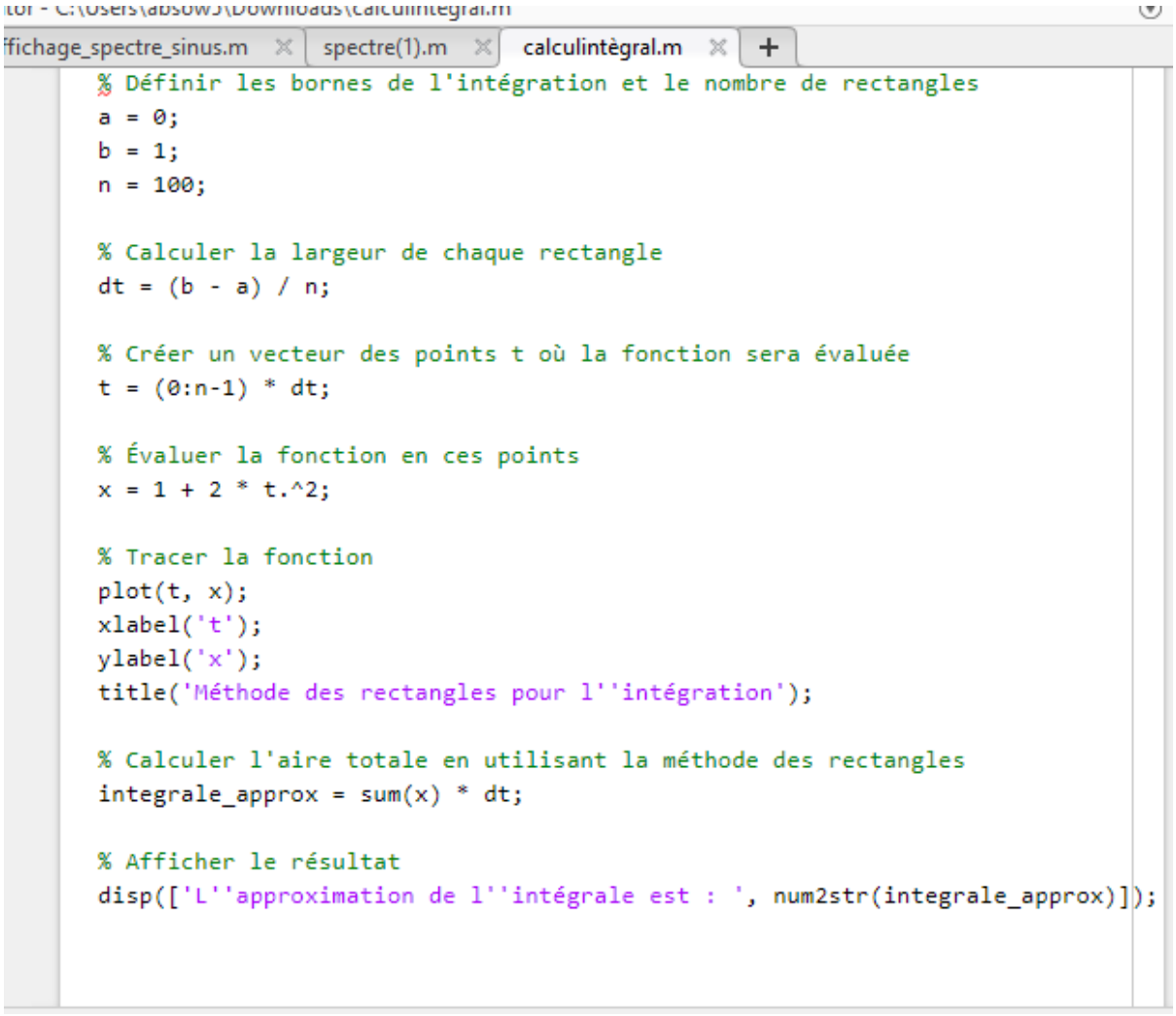
3) Montrer que de manière générale la formule du calcul approché de l'intégrale I par la méthode des rectangles sur n intervalles est :

Tout d'abord, calculer l'intégrale d'une fonction revient à calculer l'aire totale de cette fonction or l'aire d'une fonction est égale à la somme des aires repartionnées d'où la méthode des intégrales.

Réalisation :

On se propose maintenant d'implanter le calcul de cette intégrale en utilisant Matlab.

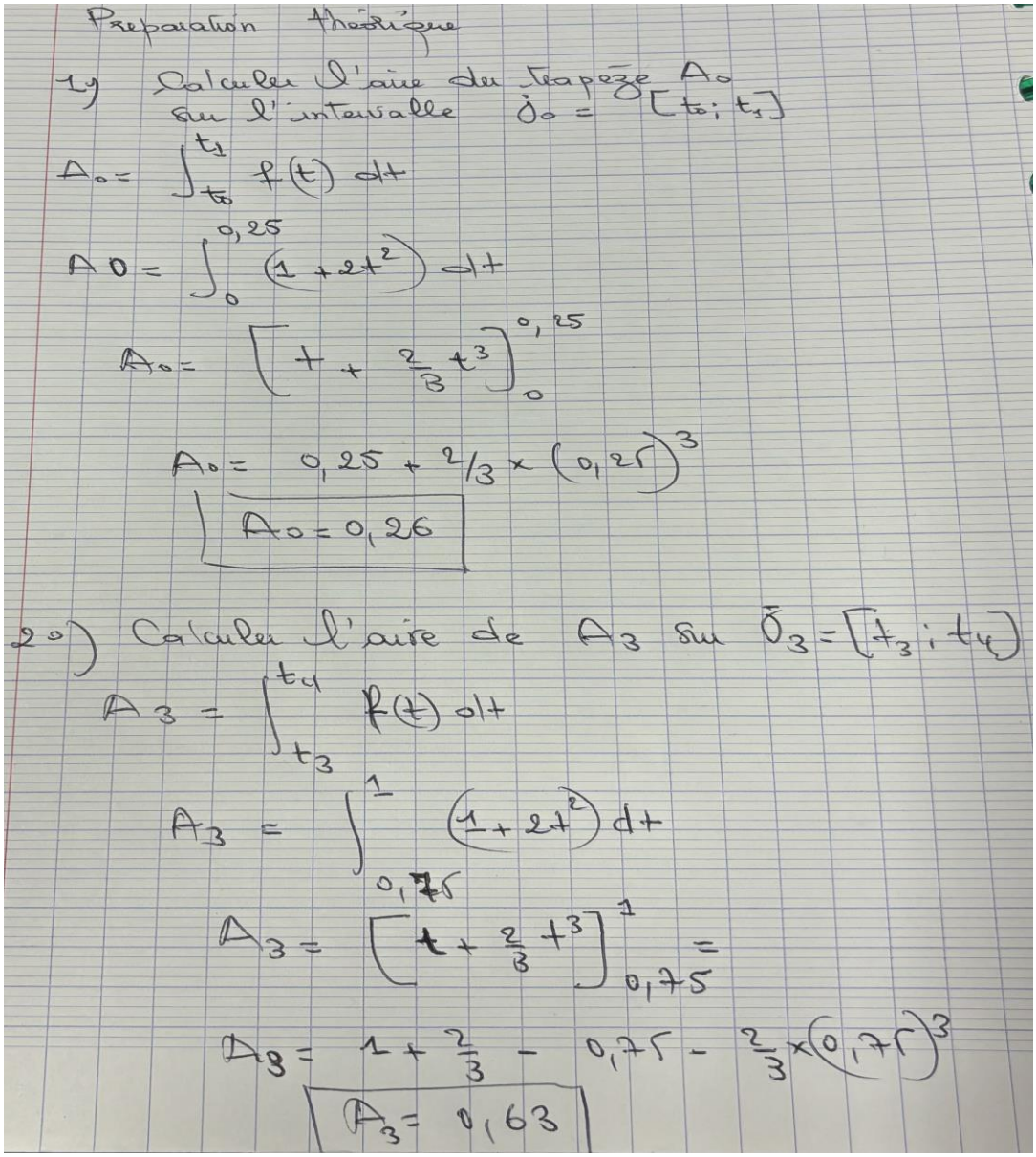
I **Compléter le script suivant** afin d'implanter l'approximation de la méthode des rectangles sur Matlab.



IV / Méthodes des trapèzes

Préparation théorique :

Calculer l'aire du trapèze A0, sur l'intervalle $J_0 = [t_0 \; ; \; t_1]$ et calculer l'aire du trapèze A3, sur l'intervalle $J_3 = [t_3 \; ; \; t_4]$.



De façon plus générale, que vaut l'aire du trapèze Ai dans l'intervalle $J_i = [t_i \; ; \; t_{i+1}]$ pour $i \in \langle 0 \; ; \; n-1 \rangle$? Exprimer le résultat en fonction de $dt = \frac{b-a}{n}$, $f(t_i)$ et $f(t_{i+1})$

Soit *f(t)* une fonction continue définie sur l'intervalle [a,b]. Nous divisons cet intervalle en n sous-intervalles de largeur $dt=b-a$. Les points de division sont t0 ,t1 ,...,tn , où ti =a+i·dt pour i=0, 1,...,n.

L'aire Ai du trapèze sur l'intervalle [ti ,ti+1] est donnée par :

Ai =2f(ti)+f(ti+1) ·dt

Ainsi, l'expression de Ai en fonction de f(ti), f(ti+1) et dt est :

Ai =2f(ti)+f(ti+1) ·dt

En résumé, la formule générale de l'aire Ai d'un trapèze dans l'intervalle [ti ,ti+1] est :

Ai =(f(ti)+f(ti+1)) /2·dt

Montrer que la formule du calcul approché de l'intégrale I par la méthode des trapèzes est :

$$T = \frac{b-a}{n} \times \sum_{i=0}^{n-1} \frac{f(t_i)+f(t_{i+1})}{2} \quad \text{(eq 2)}$$

Pour montrer que la formule du calcul approché de l'intégrale I par la méthode des trapèzes est :

I≈2dt [f(t0)+2∑i=1n−1 f(ti)+f(tn)],

nous allons partir de la définition de l'intégrale par la méthode des trapèzes et exprimer le résultat de manière compacte.

Considérons une fonction continue f(t) sur l'intervalle [a,b]. Nous divisons cet intervalle en n sous-intervalles égaux de largeur dt=nb−a . Les points de division sont

t0 ,t1 ,...,tn , où ti =a+i·dt pour i=0,1,...,n.

L'intégrale de f(t) sur [a,b] peut être approximée par la somme des aires des trapèzes formés par les segments [ti ,ti+1] pour i=0,1,...,n−1. L'aire de chaque trapèze est donnée par :

Ai =(f(ti)+f(ti+1))/2 ·dt

La somme totale des aires des trapèzes est donc :

I≈∑i=0n−1 Ai =∑i=0n−1 2f(ti)+f(ti+1) ·dt

Développons cette somme :

I≈2dt ∑i=0n−1 [f(ti)+f(ti+1)]

Si nous développons cette somme, nous remarquons que chaque f(ti) apparaît deux fois, sauf les termes aux extrémités *f*(t0) et f(tn) qui apparaissent une seule fois :

I≈2dt [f(t0)+2∑i=1n−1 f(ti)+f(tn)]

Ainsi, nous avons montré que la formule du calcul approché de l'intégrale I par la méthode des trapèzes est :

$$T = \frac{b-a}{n} \times \sum_{i=0}^{n-1} \frac{f(t_i)+f(t_{i+1})}{2}$$

Cela conclut la démonstration.

4°) (*) Montrer que cette formule peut se mettre sous la forme suivante :

$$T = \frac{b-a}{n} \times \left(\frac{f(t_0)+f(t_n)}{2} + \sum_{i=1}^{n-1} f(t_i) \right)$$

Implanter la méthode des trapèzes sur Matlab.

Voici le code sur Matlab :

Editor - C:\Users\absow5\Downloads\methodedestrapezes.m

affichage_spectre_sinus.m | spectre(1).m | calculintègral.m | methodedestrapezes.m

```
% Définir les bornes de l'intégration et le nombre de sous-intervalles
a = 0; % borne inférieure
b = 1; % borne supérieure
n = 100; % nombre de sous-intervalles

% Calculer la largeur de chaque sous-intervalle
dt = (b - a) / n;

% Créer un vecteur des points de division
t = linspace(a, b, n+1);

% Définir et évaluer la fonction en ces points
f = @(t) 1 + 2 * t.^2; % définir la fonction
y = f(t); % évaluer la fonction aux points t

% Calculer l'aire totale en utilisant la méthode des trapèzes
integrale_approx = (dt / 2) * (y(1) + 2 * sum(y(2:end-1)) + y(end));

% Afficher le résultat
disp(['L''approximation de l''intégrale est : ', num2str(integrale_approx)])

% Tracer la fonction et les trapèzes
figure;
plot(t, y, '-o'); % tracer la fonction
hold on;

% Tracer les trapèzes
```

Suite :

```
% Tracer la fonction et les trapèzes
figure;
plot(t, y, '-o'); % tracer la fonction
hold on;

% Tracer les trapèzes
for i = 1:n
    fill([t(i), t(i+1), t(i+1), t(i)], [0, 0, y(i+1), y(i)], 'b', 'FaceAlph
end

xlabel('t');
ylabel('f(t)');
title('Méthode des trapèzes pour l''intégration');
hold off;
```

V Méthode de Simpson