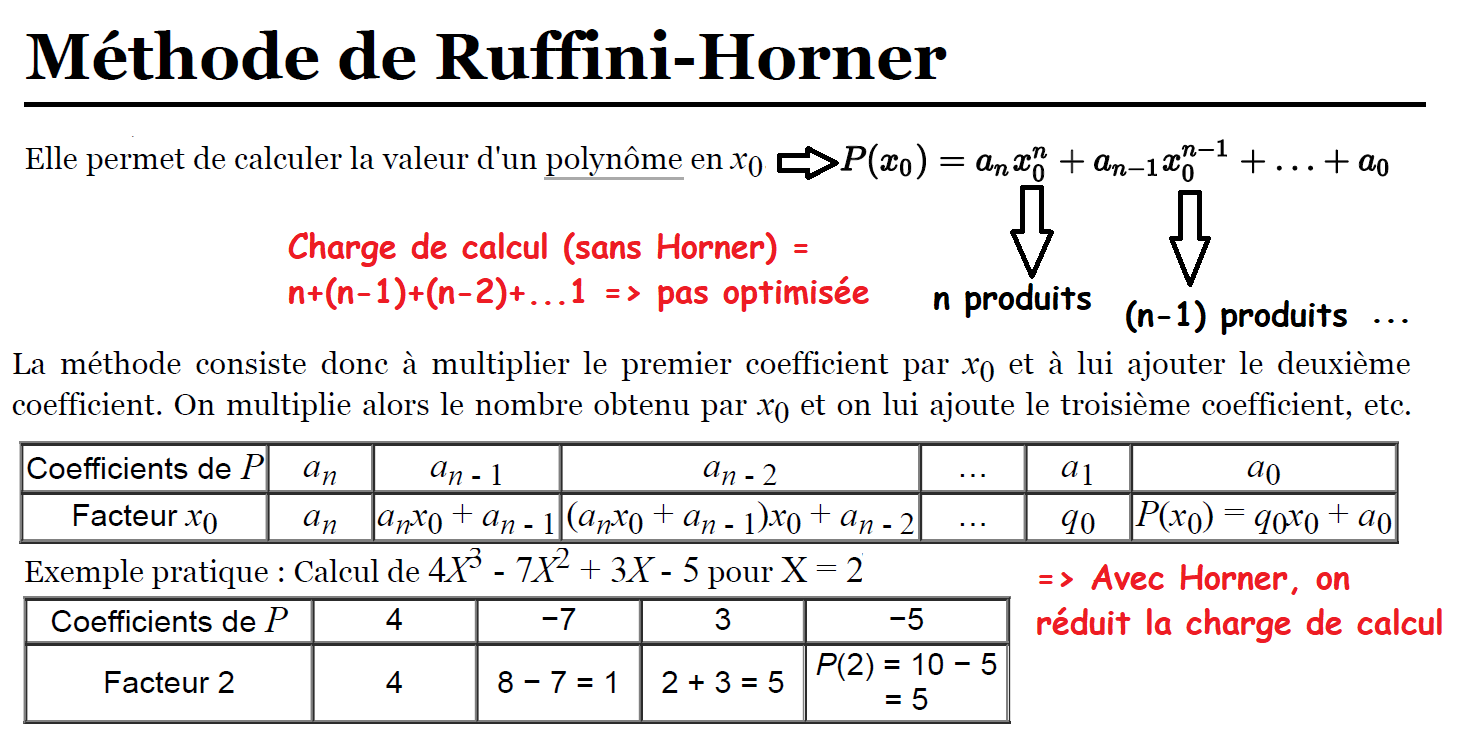
|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | TP6 MATLAB  *calcul scientifique 1* | |  | |

algorithme horner

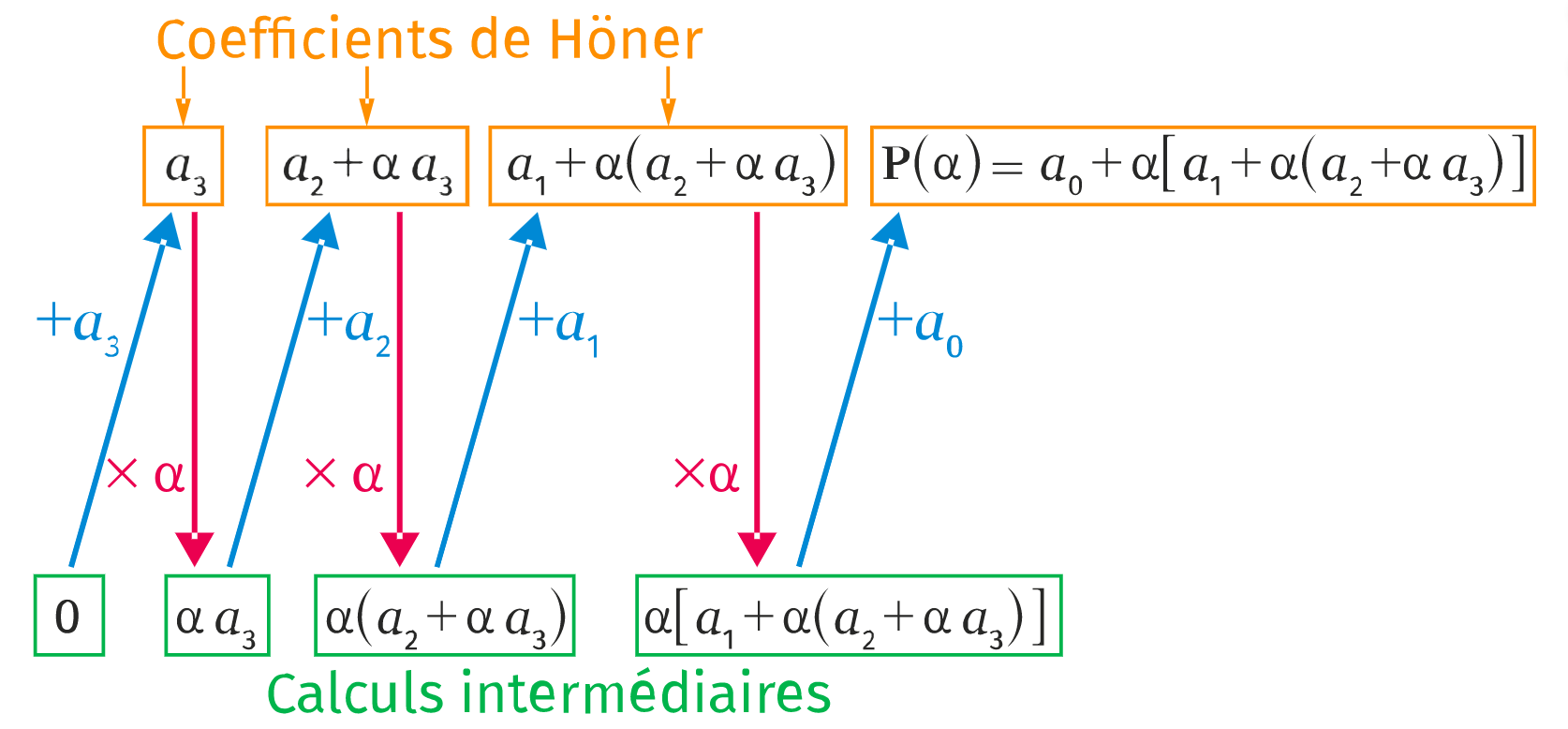
L’algorithme de Horner vise à réduire la charge de calcul (voir illustration ci-dessous) pour déterminer la valeur d’un polynôme en une valeur particulière  : P().

Le principe est le suivant :

* Nous partons du coefficient de plus haut degré :
* Puis on multiplie par  :  🡪  puis on additionne 🡪  +
* Nous réitérons les étapes précédentes comme le montre les figures ci-dessous



**Figure 1 : Principe de l'algorithme de Horner**



**Figure 2 : Exemple avec 3 itérations**

**Exercice 1** : Réaliser une fonction res = Horner() reposant sur l’algorithme de Horner, qui :

1. Demande à l’utilisateur le degré n du polynôme
2. Demande à l’utilisateur les coefficients
3. Demande à l’utilisateur la valeur de et renvoie la valeur dans res

calcul sur les polynomes

Racine d’un polynôme

1. Prenons l’exemple :
2. Créons un vecteur p avec les coefficients du polynôme
3. La commande roots permet d’obtenir les racines du polynôme :

>> roots(p)

Remarque : la commande roots traite aussi bien le corps des réels que le corps des complexes

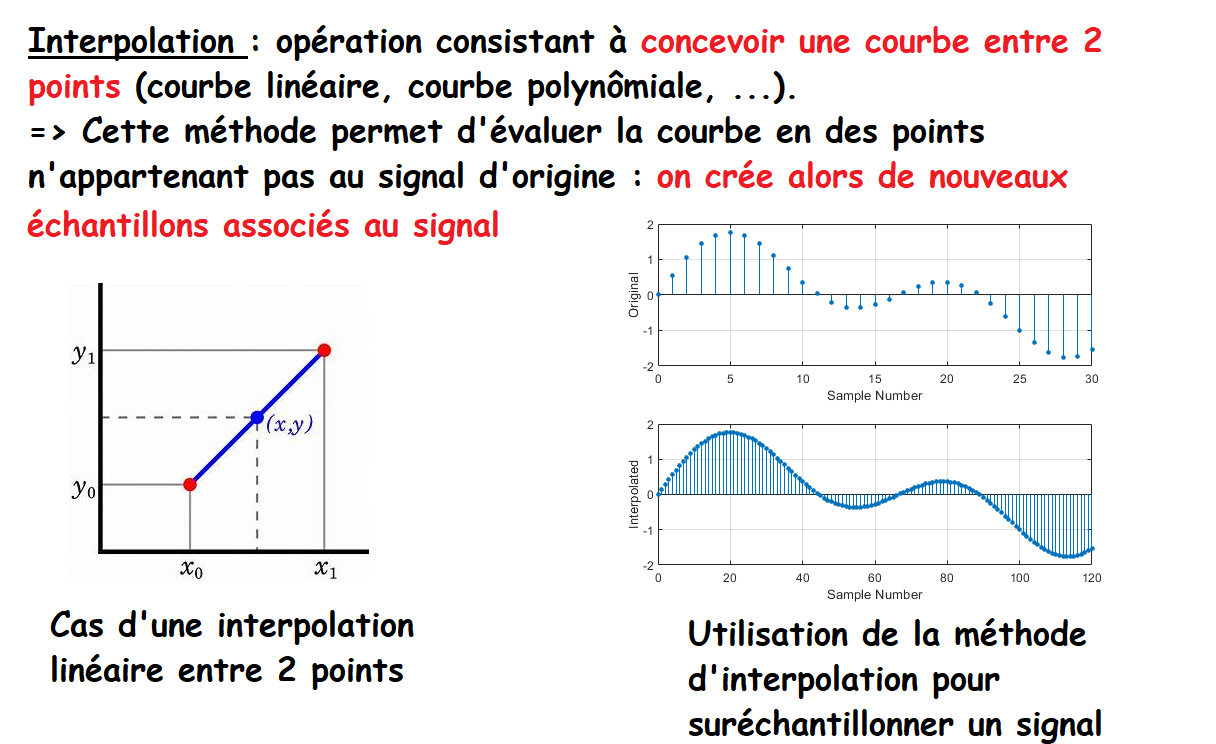
**Exercice 2** : Réaliser une fonction res = racinespolynome() qui :

1. Demande à l’utilisateur le degré n du polynôme
2. Demande à l’utilisateur les coefficients
3. Renvoie les racines du polynôme dans res

notion d’interpolation

L’interpolation consiste à créer une courbe entre 2 échantillons d’un signal pour y insérer un ou plusieurs échantillons (non issus du signal d’origine). L’interpolation peut être utilisée pour, par exemple, sur-échantillonner un signal qui peut permettre :

* d’augmenter le rapport signal sur Bruit
* d’éviter de réaliser un filtrage passe bas

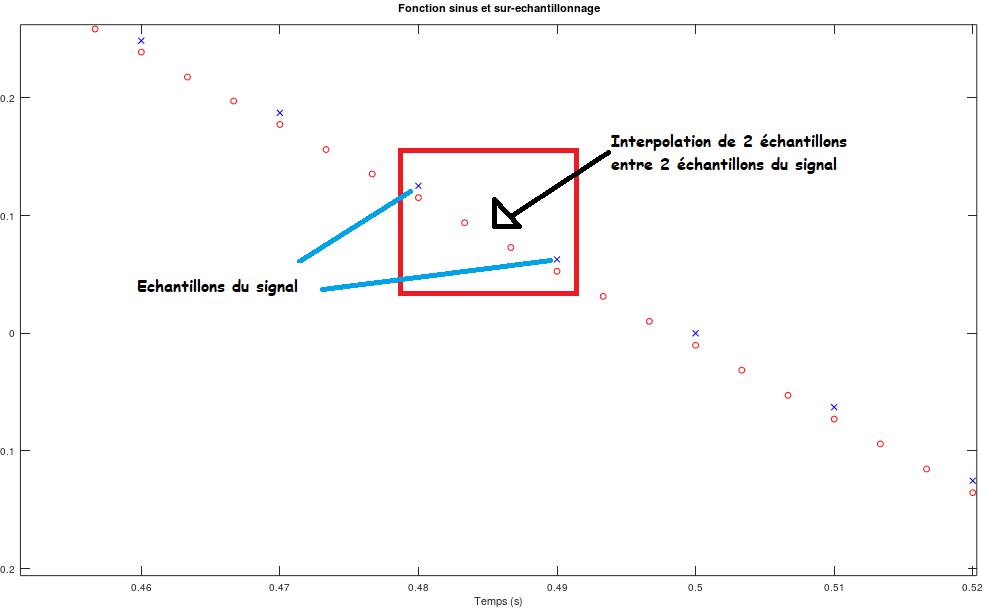


Si nous souhaitons interpoler 2 échantillons entre 2 échantillons du signal, il faut appliquer un facteur d’interpolation = 3. En effet :

* pour un échantillon signal, 3 échantillons sont générés après interpolation d’un facteur 3

La commande Matlab est la suivante :

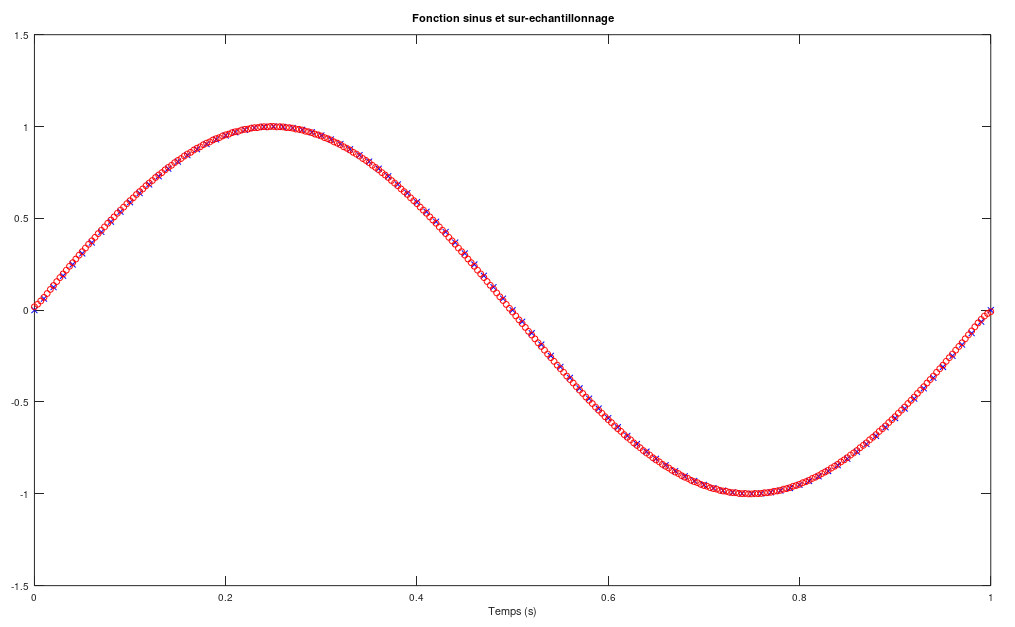
* factsample = 3; % on veut interpoler des points entre 2 points du signal d'origine
* y2 = interp(y,factsample); % création du signal sur-échantillonné y2, le signal y étant le signal d’origine avant interpolation



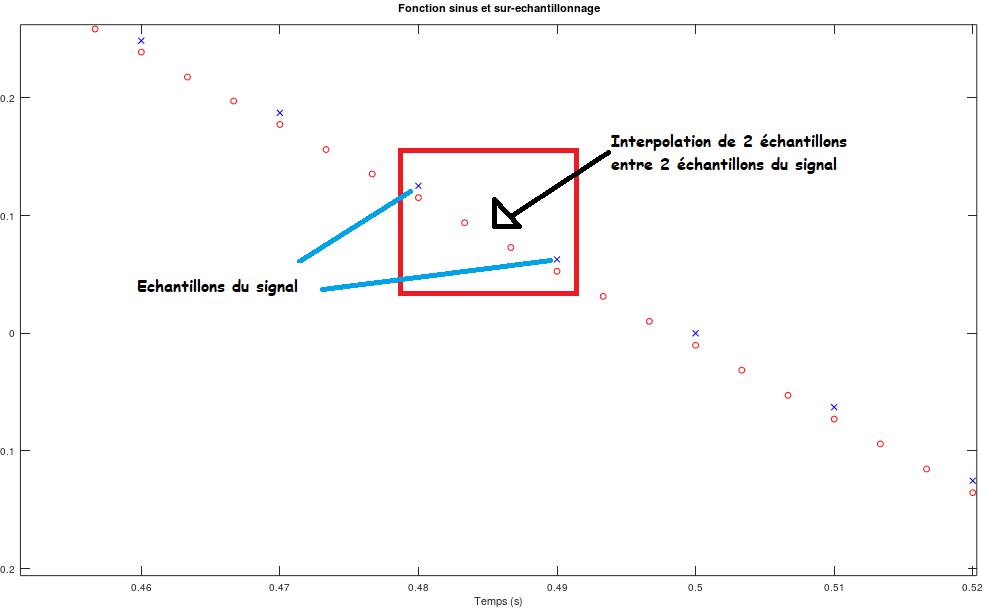
**Figure 3 : interpolation d’un facteur 3 : 2 échantillons sont crées entre 2 échantillons du signal d’origine**

**Exercice 3** : Réaliser un script qui respecte les étapes suivantes :

1. lancer le package signal contenant la fonction d’interpolation :
   * >> pkg load signal
2. Génération d’une sinusoide de 1 Hz et de fréquence d’échantillonnage Fe = 100Hz avec une durée du signal de 1s.
3. Réaliser un signal sur-échantillonné d’un facteur 3
4. Mettre à jour le vecteur temps (qui sera de fait également sur-échantillonné d’un facteur 3) pour ce signal sur-échantillonné
5. Superposer les deux signaux (origine et interpolé) sur une même courbe
6. Zoomer sur une portion du signal pour constater que 2 points ont bien été interpolés entre 2 échantillons du signal d’origine



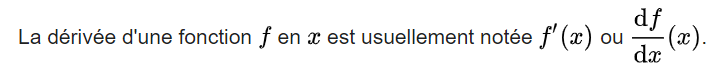
**Figure 4 : Superposition d’un signal sinusoïdale et sa version sur-échantillonnée**



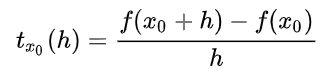
**Figure 5 : Superposition d’un signal sinusoïdale et sa version sur-échantillonnée (zoom)**

integration numérique et derivees successives

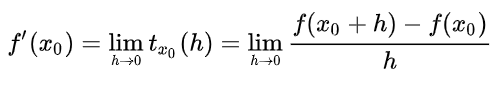
Dérivée et Dérivées successives



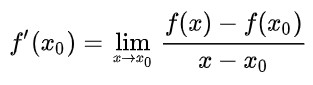
On appelle le taux d’accroissement de de f en xo, avec un pas h, la quantité :



Si le taux d’accroissement admet une limite en 0, alors la dérivée f’(xo) existe et est définie par :

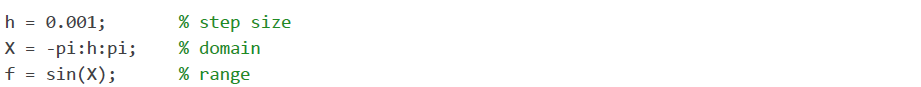


Ou de manière équivalente :



**Exercice 4 :**

On vous donne le début d’un programme matlab générant la fonction sinus en fonction de la variable X

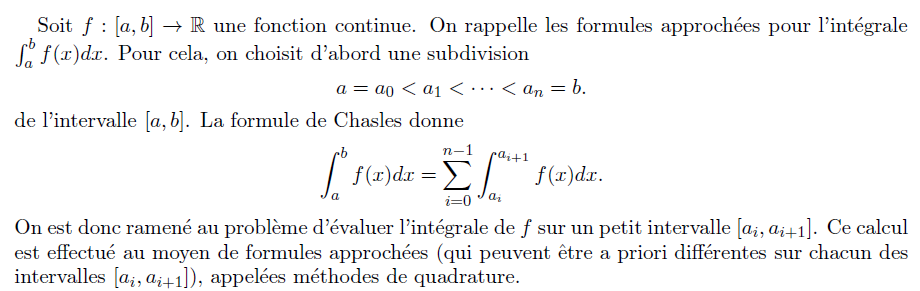


Q1. En vous appuyant de la fonction **diff** de matlab (vous aider de l’aide et documentation Matlab si nécessaire), générer la fonction dérivée première de la fonction sin(X) et visualiser le résultat en superposant les 2 courbes.

Q2. En vous appuyant à nouveau de la fonction **diff** de matlab, générer la fonction dérivée seconde de la fonction sin(X) et visualiser le résultat en superposant la courbe à celle de sin(X).

Q3. Calculer et afficher la courbe représentative de la fonction dérivée de la fonction logarithme népérien ln(x) [identifiée par log(x) en matlab]. Comparer cette courbe avec celle de la fonction f(x) = 1/x. On prendra la variable x comprise entre 1 et 10 par pas de 0.001.

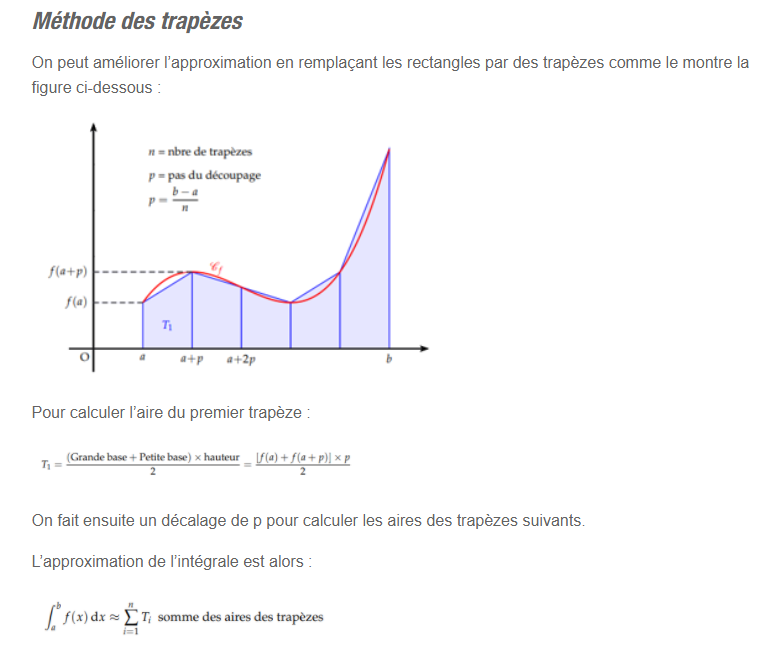
Intégration numérique



Une des méthodes très connues et implémentée dans Matlab est la méthode dite « des trapèzes » qui est présentée ci-dessous :

c’est-à-dire que chaque portion d’intégrale (si on adopte la notation précédente) :

(formule d’aire d’un trapèze : [Grande base + Petite base]\*hauteur/2)



Q1. En vous appuyant de la fonction **cumtrapz(x,f)** de matlab (la fonction **trapz** donne le résultat de l’intégrale ), générer la fonction primitive F(x) de la fonction f(x) = cos(x) : vérifier que la fonction obtenue est sin(x). On prendra la variable x comprise entre -pi et pi par pas de 0.001.

Q2. Vérifier que la fonction primitive de la fonction f(x) = 1/x est F(x) = log(x) On prendra la variable x comprise entre 1 et 10 par pas de 0.001.