Université Euro Méditerranéenne Fès
Ecole d'Ingénierie Digitale et d'Intelligence Artificielle (EIDIA)

Module: Calcul Formel sous Matlab

Chapitre 5:

Polynômes, Interpolation et Résolution Numérique

Pr : A. SLIMANI

2021/2022

Plan

1- Introduction

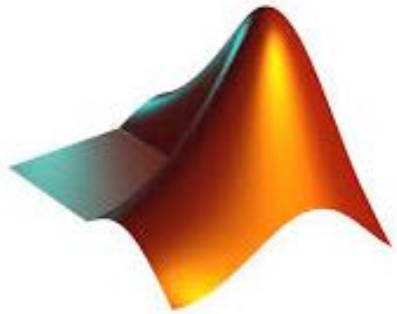
2- Polynômes

3- Interpolation

4- Résolution numérique des équations non linéaires

5- Exercices d’application

1. Introduction



Dans ce chapitre, il sera question de trois choses principales :

- ❑ Les polynômes
- ❑ L'interpolation polynomiale
- ❑ Résolution des équations non linéaires.

Comment ?

- ✓ Premièrement, le savoir de représenter et résoudre des polynômes sous Matlab,
- ✓ Deuxièmement, l'étude de l'interpolation et le traitement de quelques types,
- ✓ Et finalement, la résolution numérique des équations non linéaires.

2. Polynômes

Les polynômes, dans MATLAB, sont représentés sous forme de vecteurs lignes dont les composantes sont données par ordre des puissances décroissantes. Un polynôme de degré n est représenté par un vecteur de taille $(n+1)$.

Exemple: $P(x) = 2x^3 + 5x^2 - 3x + 4$

Dans la zone de commande de Matlab, on peut représenter le polynôme $P(x)$ de degré 3, par ses coefficients sous forme d'un vecteur ligne:

$$\gg P = [2 \quad 5 \quad -3 \quad 4]$$

❑ Produit des polynômes

Le produit de deux polynômes sous Matlab se fait à l'aide de la commande ($\boldsymbol{conv}$) .

$\boldsymbol{conv}$ est une fonction qui fait le produit de convolution.

Exemple:

Soient les deux polynômes $P(x) = 2x^3 - x^2 + 3x + 2$ et $\mathrm{H}(x) = x^2 + 2x + 1$.

Leur produit $\boldsymbol{g(x)}$ est sous forme:

```
≫ g = conv(P, H)
g
     2     3     3     7     7     2
```

Ainsi, le polynôme $\boldsymbol{g(x)}$ obtenu est : $\boldsymbol{g(x) = 2x^5 + 3x^4 + 3x^3 + 7x^2 + 7x + 2}$

❑ Division des polynômes

deconv est une fonction qui fait la division entre deux polynômes, elle affiche le quotient sans donner le reste s'il existe.

Exemple:

On considère les deux polynômes précédents $P(x)$ et $H(x)$.

$\boldsymbol{D(x)}$ est la division de $P(x)$ par $H(x)$:

```
>> D = deconv(P,H)
D =
     2    -5
```

Ainsi, le polynôme $\boldsymbol{D(x)}$ obtenu est :

$$\boldsymbol{D(x) = 2x - 5}$$

❑ Racines des polynômes

Soit le polynôme $\boldsymbol{P(x)}$ d'ordre n.

$$\boldsymbol{P(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \cdots + a_1 x^1 + a_0}$$

Le polynôme $\boldsymbol{P(x)}$ peut s'écrire sous forme factorisée:

$$\boldsymbol{P(x) = a_n (x - r_1)(x - r_2)(x - r_3) \ldots (x - r_n)}$$

Où, $\boldsymbol{r_1}$, $\boldsymbol{r_2}$, …, $\boldsymbol{r_n}$ sont les racines du polynômes $\boldsymbol{P(x)}$.

Un polynôme d'ordre n possède n racines qui peuvent être réelles ou complexes.

Remarque 1:

Pour trouver les racines d'un polynôme en Matlab, on peut utiliser la commande (***roots***).

Exemple: Soit le polynôme $P(x) = x^3 - 7x + 6$

Les racines r de $P(x)$ sous Matlab sont:

```
>> r = roots(P)
 r =
     -3.0000
      2.0000
      1.0000
```

Ainsi, le polynôme $\boldsymbol{P(x)}$ peut s'écrire sous forme factorisée:

$$P(x) = \big(x - (-3)\big)\,(x - 2)\,(x - 1)$$

Remarque 2:

On peut déduire un polynôme à partir de ses racines avec la commande (**poly**).

Exemple: Si on prend l'exemple précèdent, à partir des racines r :

On déduit $P(x)$:

```
>> P = poly(r)
 P =
    1.0000    0.0000   -7.0000    6.0000
```

À partir de ces coefficients, on peut déduire le polynôme $\boldsymbol{P(x)}$:

$$P(x) = x^3 - 7x + 6$$

Remarque 3:

La commande (**poly**) peut accepter une matrice comme argument dont elle retourne le polynôme caractéristique.

Exemple:

À partir de la matrice : $M = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$

On déduit les coefficients de son polynôme $P(x)$:

```
>> M = [1 2; 4 3]
>> P = poly(M)
P =
     1    -4    -5
```

À partir de ces coefficients on déduit le polynôme $\boldsymbol{P(x)}$: $P(x) = x^2 - 4x - 5$

❑ Evaluation des polynômes

La commande (**polyval**) est utilisée pour évaluer un polynôme en un point x donné.

Exemple: Soit le polynôme $P(x) = x^3 - 7x + 6$

On peut évaluer P en $x = 3$ par: $\boldsymbol{polyval(P, 3)}$

Syntaxe:

```
>> P = [1  0  -7  6]
>> polyval(P,3)
ans =
     12
```

$P(3) = 3^3 - (7 \times 3) + 6 = 12$ ...

Exercice d'application:

Soient les deux polynômes suivants:

$$P(x) = x^4 - 5x^3 + 5x^2 + 5x - 6 \qquad G(x) = x^2 + \frac{1}{4}x - \frac{1}{8}$$

1. Calculer le produit des deux polynômes.
2. Calculer le quotient obtenu de la division de P par G.
3. Calculer $P(1)$, $P(2)$ et $P(3)$
4. Calculer les racines de P.
5. Déduire le polynôme qui possède les deux racines suivantes: $-\frac{1}{2}$ et $\frac{1}{4}$

Plan

1- Introduction

2- Polynômes

3- Interpolation

4- Résolution numérique des équations non linéaires

5- Exercices d'application

3. Interpolation

L’interpolation polynomiale est une technique mathématique utilisée pour chercher un polynôme qui vérifie un ensemble de condition.

Par exemple ,

Un nombre de points obtenu d'une expérience (données expérimentales), l’objectif est de chercher un polynôme qui passe par tous ces points.

Les coefficients optimaux de ce polynôme sont ceux qui minimisent la variance de l’erreur d’interpolation.

A. Interpolation au sens des moindres carrés

C'est une méthode utilisée pour minimiser la variance de l'erreur d'interpolation.

- ***polyfit*** est une commande utilisée pour retourner un polynôme P de degré n permettant d'approcher la courbe $y = f(x)$ au sens ***des moindres carrés***.

Exemple:

On exécute le programme suivant dans un nouveau script:

```
x = linspace(1,6,4)    % Vecteur de données
y0 = sin(x+1); % Fonction de données à interpoler (y0=f(x))
p = polyfit(x,y0,3)%Interpolation des moindres carrés d'ordre 3
```

Dans la zone de commande, il s'affiche:

$$p = 0.0246 \quad 0.0008 \quad -1.1146 \quad 1.9985$$

Ainsi, le polynôme d'interpolation de y0 est le polynome p (d'ordre 3) :

$$p(x) = 0.0246\,x^3 - 0.0008\,x^2 - 1.1146x + 1.9985$$

- Pour déduire s'il y a une erreur entre les valeurs expérimentales et le modèle obtenu par la fonction **polyfit**, il suffit de remplacer $\boldsymbol{x}$ dans $\boldsymbol{p}$ avec la commande **polyval**.

On ajoute cette commande au programme de script précédent: $\boldsymbol{y1 = polyval(p, x)}$

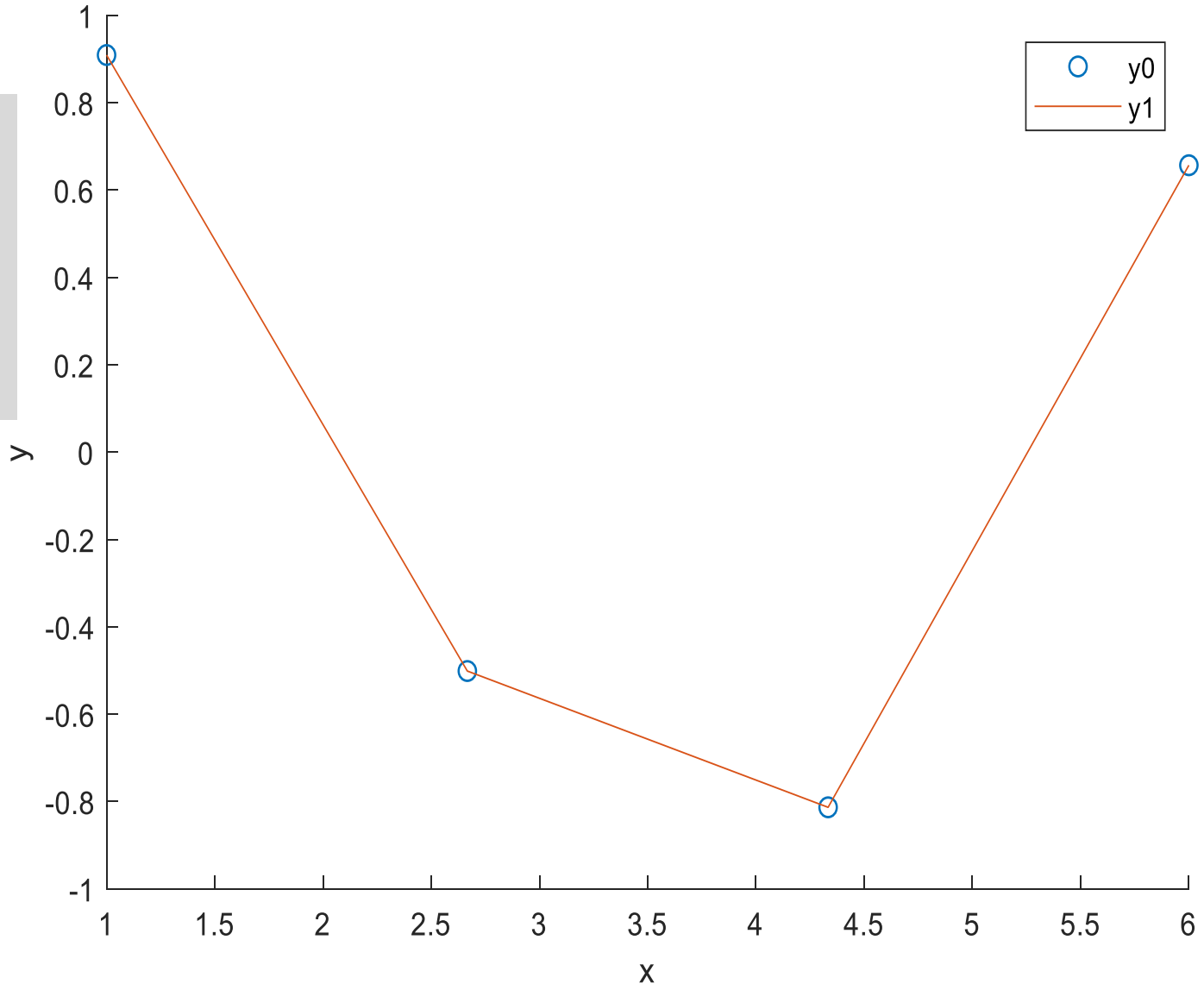
Après exécution, il s'affiche dans la zone de commande:

$$y1 \quad = 0.9093 - 0.5013 - 0.8133 \quad 0.6570$$

$\boldsymbol{L'erreur = |y0 - y1|}$

On peut comparer les deux sur le même graphe en ajoutant les commandes suivantes sur le script précèdent:

```
hold on
plot(x,y0,'o'),plot(x,y1)
xlabel('x'),ylabel('y')
legend('y0','y1')
```



B. Interpolation linéaire et non linéaire

Une interpolation linéaire consiste à relier deux points expérimentaux successifs par une courbe sous forme de segments de droites, par contre des courbes polynomiales pour le cas non linéaire.

Les trois méthodes d'interpolation qu'on va voir sont:

- ' **linear** ' → interpolation linéaire
- ' **spline** ' → interpolation par splines cubiques,
- ' **cubic** ' → interpolation cubique.

' **interp1** ' est la commande utilisée pour réaliser ces interpolations.

Syntaxe: $f = interp1(x, y, z, 'Méthode')$

- x est le vecteur des abscisses des points d'échantillonnage,
- y est le vecteur des valeurs d'échantillonnage, correspondant à x.
- z est le vecteur des abscisses des points dont on souhaite interpoler la valeur,
- $Méthod$ est la méthode d'interpolation (linéaire, cubique,…)
- f est le vecteur des valeurs interpolées.

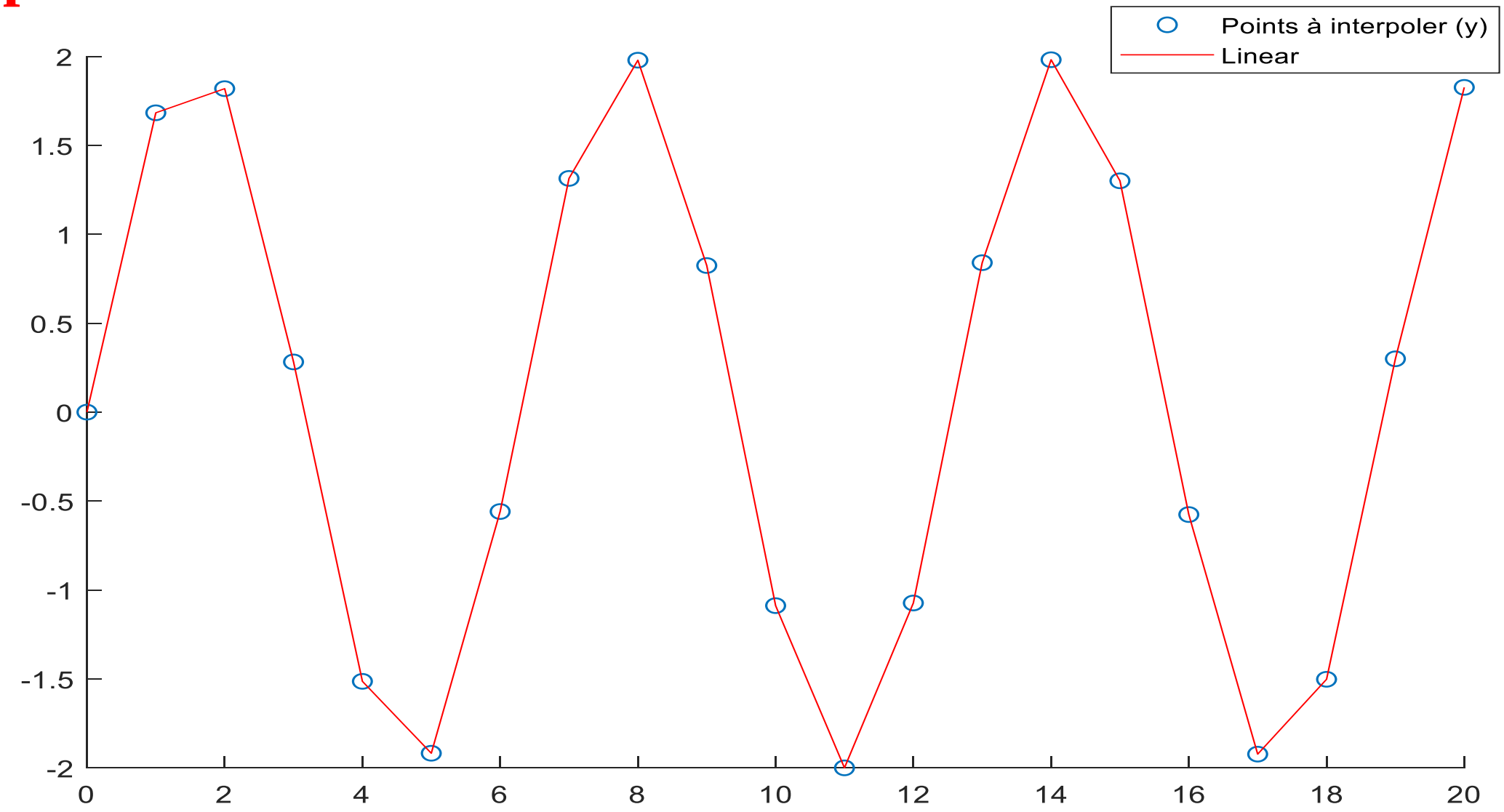
Autrement dit, cette syntaxe retourne la valeur f aux points z de la fonction valant y aux points x, et dont la valeur en dehors de ces points est calculée en utilisant la méthode d'interpolation **Méthode**.

Exemple d'application:

Dans cet exemple, on a étudié l'interpolation linéaire sur un exemple de valeurs discrètes de la fonction **'sinus'.**

```
x = 0:20          % vecteur de données
y = 2*sin(x); % Points à interpoler
z = 0:0.1:20 ; % vect de point pour le polynômes d'Inter
f = interp1(x,y,z,'linear'); % Interpolation linéaire
hold on
plot(x,y,'o') % figure des Points à interpoler
plot(z,f,'b-') % figure d' Interpolation linéaire
legend('Points à interpoler (y) ','Linear')
```

Graphe:


Points à interpoler (y)
Linear
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
2
4
6
8
10
12
14
16
18
20

Exercice d’application:

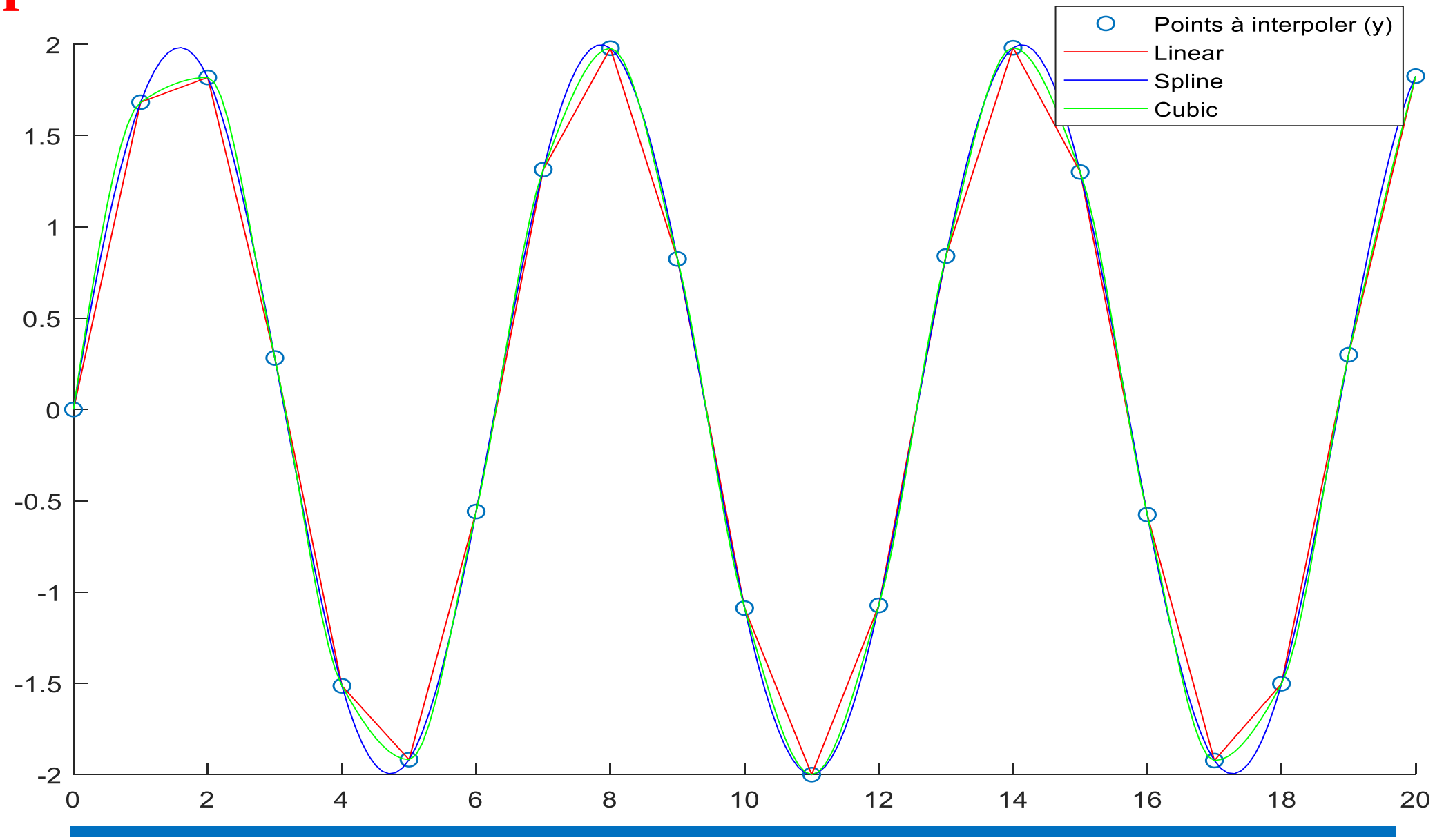
Tracer les différentes courbes d’interpolation:

- ✓ ***linear,***
- ✓ ***spline***
- ✓ ***cubic***

sur la même figure de la fonction précédente **‘sinus’.**

- ❑ Donner une légende pour chaque type.

Graphe obtenu:



C. Interpolation de Lagrange

Rappel de la définition:

Soit $\boldsymbol{n}$ un entier naturel. $(x_0, \ldots, x_n)$ sont $(n+1)$ réels deux à deux distincts d'un segment $[a, b]$ et f une fonction de $[a, b]$ dans R. Il existe un et un seul polynôme de degré inférieur ou égale à n vérifiant $\forall\, i \in [0, n]$, $P(x_i) = f(x_i)$ à savoir :

$$P_n(x) = \sum_{i=0}^{n} f(x_i)\, L_i(x)$$

avec,

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right)$$

Exemple d'application:

Le tableau ci-dessous représente les valeurs d'une variable $\boldsymbol{Y}$ en fonction de sa position $\boldsymbol{x}$:

$\boldsymbol{x}$	**Y**
2	5
3	7
5	8

1. Écrire la formule d'interpolation de Lagrange qui permet d'interpoler les différents points de données précédentes.
2. Créer un algorithme qui permet de :
   - calculer $\boldsymbol{Y}$ à $\boldsymbol{x = 4}$, en utilisant l'interpolation de Lagrange.
   - Déterminer le polynôme d'interpolation $P_n(x)$.

Solution:

1. La formule d'interpolation de Lagrange:

Le degré du polynôme de Lagrange sera de degré 2 car le nombre de points total est égal à 3.
Ce polynôme s'écrit :

$$P_2(x) = \sum_{i=1}^{3} Y(x_i)\, L_i(x) = \sum_{i=1}^{3} Y(x_i) \prod_{\substack{j=1 \\ j \neq i}}^{3} \left(\frac{x - x_j}{x_i - x_j} \right)$$

D'où :

$$P_2(x) = 5\left(\frac{x-3}{2-3}\right)\left(\frac{x-5}{2-5}\right) + 7\left(\frac{x-2}{3-2}\right)\left(\frac{x-5}{3-5}\right) + 8\left(\frac{x-3}{5-3}\right)\left(\frac{x-2}{5-2}\right)$$

2. **Algorithme :** calculer Y à $x = 4$

```
X=[2 3 5]; Y=[5 7 8];
Xc=4; % On cherche Y à Xc
P0=0; % Initialisation de P0 pour la somme
n=length(Y); % Nombre de points
for i=1:n
L0=1; % Initialisation de L0 pour le produit
    for j=1:n
        if i~=j
            L0=L0.*(Xc-X(j))/(X(i)-X(j));
        end
    end
     P0=P0+(Y(i)*L0);
end
disp(' la valeur de Y est: ')
Y=P0
```

Après exécution du programme, on trouve:

```
Y=8
```

2. **Algorithme :** le polynôme d'interpolation $P_n(x)$

```
X=[2 3 5];
Y=[5 7 8];
P=0; % Initialisation de P pour la somme
n=length(Y); % Nombre de points
for i=1:n
    L=1; % Initialisation de L pour le produit
    for j=1:n
        if i~=j
            L=conv(L,poly(X(j)))/(X(i)-X(j));
        end
    end
      P=P+(Y(i)*L);
end
disp('Le polynôme Pn est : ')
P
```

Après exécution du programme, on trouve:

P= -0.5000 4.5000 -2.0000

Exercice d'application:

Le tableau ci-dessous représente les valeurs de viscosité d'huile à différents degrés de Température T :

T	Visc
110	10,8
130	8,1
160	5,5
190	4,8

1. Écrire la formule d'interpolation de Lagrange qui permet d'interpoler les différents points de données précédentes.
2. Créer un algorithme qui permet de :
   - Calculer $\boldsymbol{Visc}$ à $\boldsymbol{T = 140°}$, en utilisant l'interpolation de Lagrange.
   - Déterminer le polynôme d'interpolation $P_n(x)$.

Plan

1- Introduction

2- Polynômes

3- Interpolation

4- Résolution numérique des équations non linéaires

5- Exercices d'application

4. Résolution numérique des équations non linéaires

Rappel:

Soit une fonction réelle f définie et continue sur un intervalle $[a, b]$, avec $a < b$. On suppose que f admet une unique racine sur $I =]a, b[$, c'est-`a-dire qu'il existe un unique $\alpha \in I$ tel que $f(\alpha) = 0$.

Les méthodes de résolution qu'on va traiter à cette partie sont:

- **Méthode de la bissection (dichotomie)**
- **Méthode du point fixe**
- **Méthode de Newton-Raphson**

❑ Méthode de la bissection

On suppose que f est continue dans $[a, b]$ et que $f(a).f(b) < 0$

- On pose $c = \frac{a+b}{2}$,
  - si $f(c) = 0$ alors $\alpha = c$
  - si $f(a) * f(c) < 0$ on remplace b par c
  - sinon , on remplace a par c
- On continue cette opération jusqu'à ce qu'on trouve α avec la précision demandée.

❑ Algorithme de la bissection

```
a= ... ; b= ... ; % définir les bornes a et b
pre= ... ; % définir la précision
ite=0;  % itération
 while abs(b-a) > pre
    c=(a+b)/2;
     if  f(a)*f(c)<0
         b=c;
     else
         a=c;
     end
    ite=ite+1;
 end
c,  ite
```

Exemple d'application

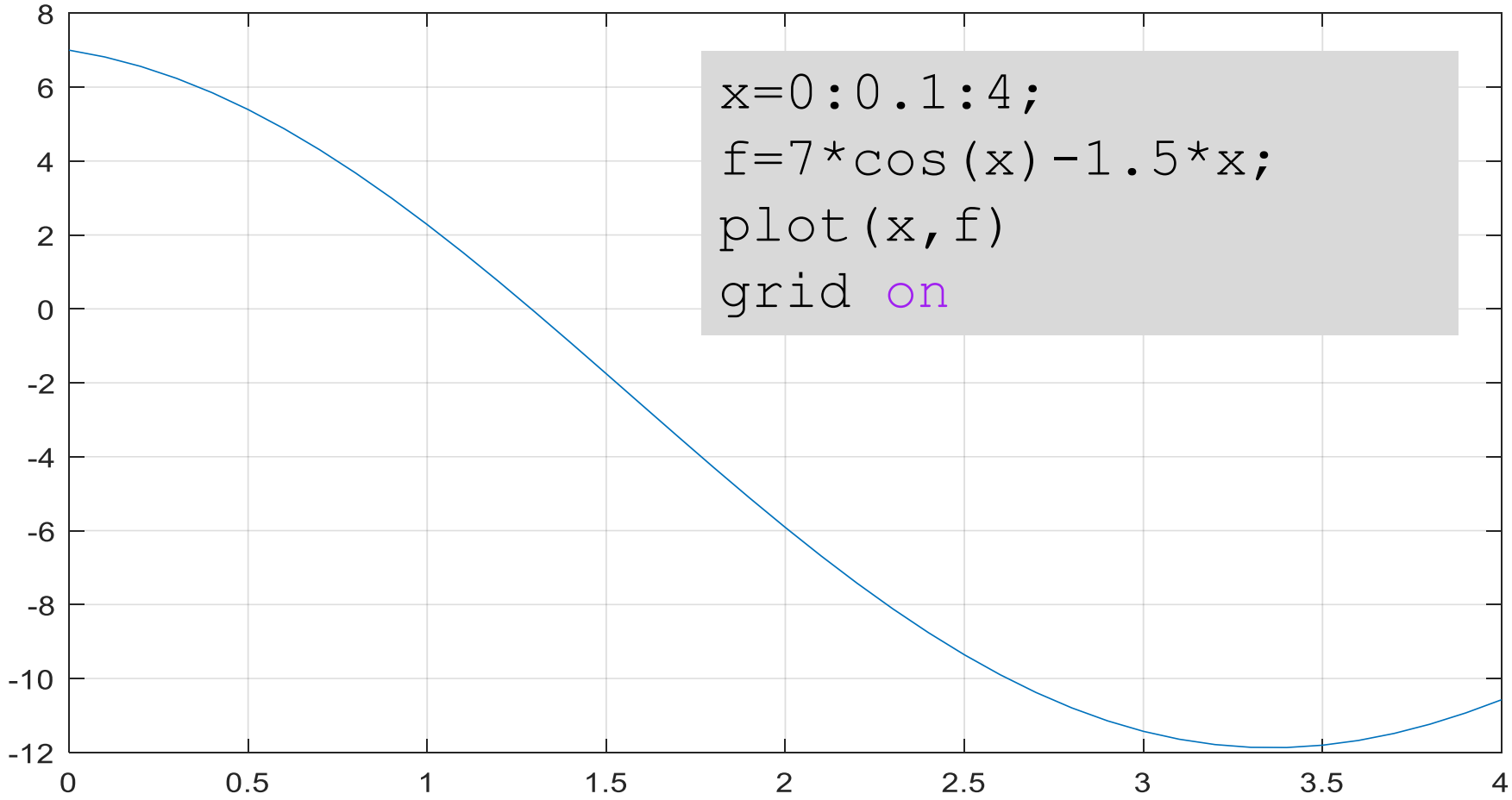
Soit la fonction :

$$f(x) = 7\cos(x) - \frac{3}{2}x$$

1. Tracer la fonction f avec un quadrillage sur l'intervalle $[0,4]$. (Prenez un pas de 0.1)
2. Localiser la racine $f(x) = 0$ à partir du graphe.
3. Trouver la valeur approchée α de la racine de f par la méthode de la bissection avec une précision de 10^{-5}.
4. Combien d'itération est nécessaire pour atteindre cette précision ?

Solution:

1. Graphe de la fonction f:



2. À partir du graphe on peut déduire que $f(x) = 0$ sur l'intervalle $[1\ ,1.5]$

L'algorithme de la bissection :

```
a=1 ; b=1.5 ;
pre=1e-5; % précision
ite=0;  % iteration
 while abs(b-a) > pre
    c=(a+b)/2;
     if (7*cos(a)-1.5*a)*(7*cos(c)-1.5*c) <0   %f(a)*f(c)<0
         b=c;
     else
         a=c;
     end
    ite=ite+1;
 end
c,  ite
```

- La valeur approchée α de la racine est : $\alpha = 1.2906$
- Le nombre d'itération nécessaire est 12 itérations

❑ Méthode du point fixe

Le principe de cette méthode consiste à transformer l'équation $f(x) = 0$ à une équation équivalente $g(x) = x$, ou g est une fonction auxiliaire.

❑ Technique :

- On commence par choisir le point initial x_0 $(n = 0)$
- On calcule $x_{n+1} = g(x_n)$
- Si $|\boldsymbol{x_{n+1} - x_n}| < \boldsymbol{pr\acute{e}sicion,}$ alors la méthode a convergé, et on s'arrête.
- Sinon, on passe à l'étape 2 pour une nouvelle itération $n + 1$ (n devient $n + 1$).

❑ Algorithme du point fixe

```
g=@(x)  f(x); ;    % la fonction
pre = ... ;                % la précision
x0= ... ;
x1=g(x0);
iter= 0   ;              %la 1ère itération
while abs(x0-x1)>pre
    x0=x1;
    x1=g(x0);
    iter=iter+1;
end
x1,  iter
```

Exemple d'application

Soit la fonction :

$$g(x) = 2\sin(x) - \frac{x}{3}$$

1. Trouver la valeur approchée α de la racine de g par la méthode du point fixe avec une précision de 10^{-3} sachant que $x_0 = 0.5$.
2. Combien d'itération est nécessaire pour atteindre cette précision ?

Solution:

```
g=@(x) 2*sin(x)-x/3;    % la fonction
pre =1e-3 ;               % la précision
x0= 0.5;
x1=g(x0);
iter= 0   ;              %la 1ère itération
while abs(x0-x1)>pre
    x0=x1;
    x1=g(x0);
    iter=iter+1;
end
x1,
iter
```

Après exécution du programme :

- La valeur approchée α de la racine est : $x_1 = 1.4957$
- Le nombre d'itérations nécessaire est 6 itérations

❑ Méthode de Newton-Raphson

La méthode de **Newton-Raphson** permet d'approcher par itérations la valeur x^* au moyen de la relation suivante :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Pour éviter une divergence de la solution, il est nécessaire de bien choisir la valeur initiale x_1. Cette valeur peut être obtenue graphiquement.

Programme:

```
x(1)= ... ; précision= ... ; n= ... ;
for i=2:n
    f= ...; % Définir la fonction f
    div= ...;   % dérivée de f
    x(i)=x(i-1)-f/div; % méthode de Newton-Raphson
    if abs(x(i)-x(i-1))<=précision
        x_précision=x(i);
        break;
    end
end
```

Exemple d'application:

On se propose d'appliquer la méthode de ***Newton-Raphson*** pour chercher des racines dans l'intervalle $[-1, 1]$ de la fonction non linéaire suivante pour une précision égale à 10^{-9} :

$$f(x) = e^{x-1} - \cos(x)$$

1. tracer la courbe représentative de cette fonction pour bien choisir une valeur initiale x_1 proche de zéro pour avoir une convergence rapide.
2. Appliquer la méthode de ***Newton-Raphson*** pour chercher les racines de f
3. Tracer l'évolution de la convergence en fonction du nombre d'itération.

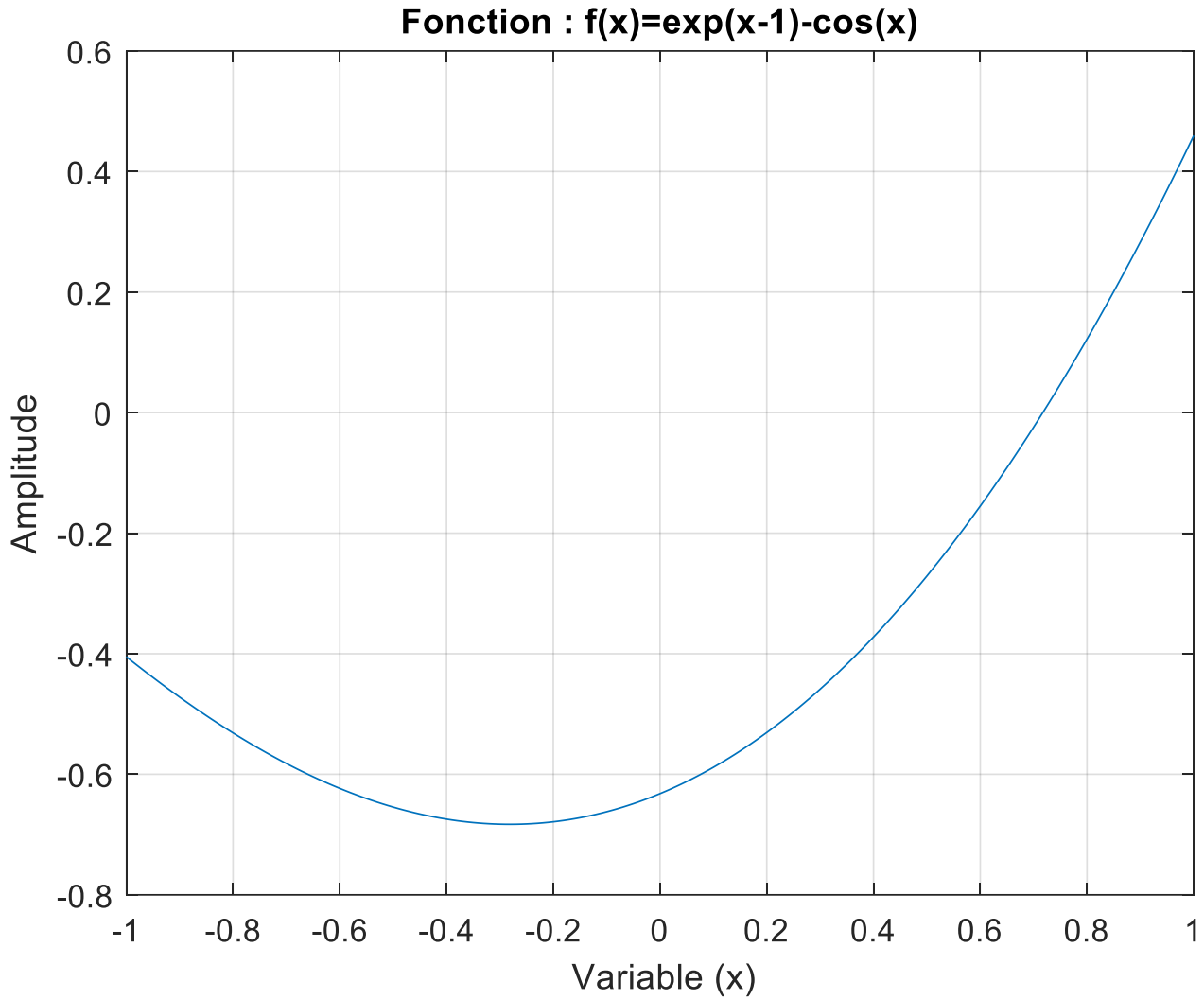
Solution:

Pour avoir une convergence rapide, il faut choisir une valeur de $f(x_1)$ proche de zéro.

- D'après la courbe de la fonction $f(x)$, on peut observer qu'il est judicieux de choisir $x_1 = 0.7$.

- La fonction dérivée $f'(x)$ a pour expression :

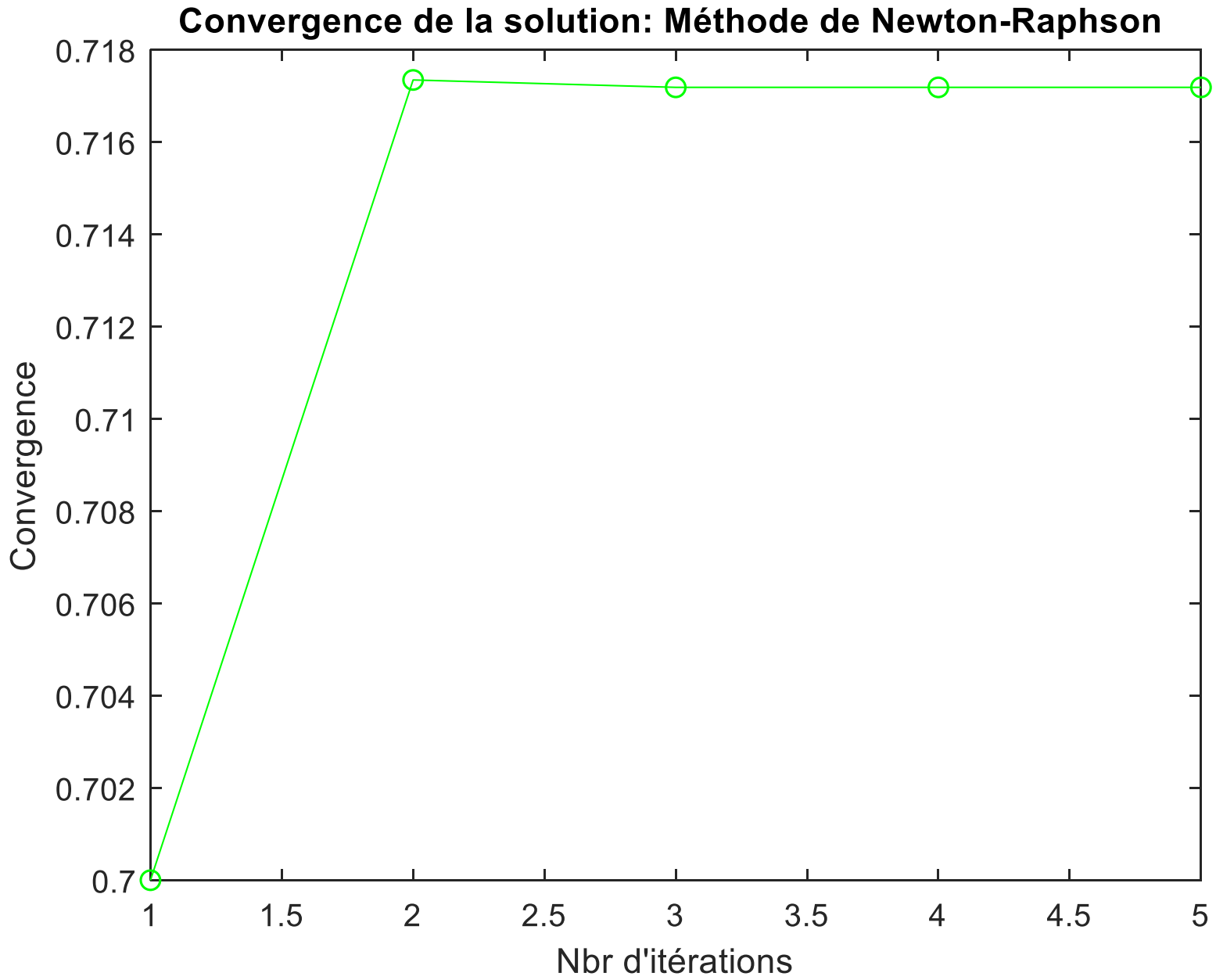
$$f'(x) = e^{x-1} + \sin(x)$$



Programme:

```
x(1)=0.7; precision=10^-8; n=100;
for i=2:n
    f=exp(x(i-1)-1)-cos(x(i-1));
    div=exp(x(i-1)-1)+sin(x(i-1));    % div: dérivée de f
    x(i)=x(i-1)-f/div;
    if abs(x(i)-x(i-1))<=precision
        x_precision=x(i); fprintf(' x_precision=%f\n',x(i))
        break;
    end
end
j=1:i;
plot(j,x(j),'go-');  xlabel('Nbr d''itérations');
ylabel('Convergence')
title('Convergence de la solution: Méthode de Newton-Raphson ')
fprintf('\n Le nombre d''itérations est: %d ',i)
fprintf('\n\n Les valeurs de x(i) sont')
x=x(j)
```

Graphe:


Convergence de la solution: Méthode de Newton-Raphson
Convergence
Nbr d'itérations
0.718
0.716
0.714
0.712
0.71
0.708
0.706
0.704
0.702
0.7
1
1.5
2
2.5
3
3.5
4
4.5
5

Plan

1- Introduction

2- Polynômes

3- Interpolation

4- Résolution numérique des équations non linéaires

5- Exercices d’application

Exercice 1:

Soient les deux polynômes suivants g_1 et g_2 :

$$\begin{cases} g_1(x) = 3x^2 + 12x - 2 \\ g_2(x) = x^3 - 4x^2 + 2 \end{cases}$$

1. Calculer les racines de chaque polynôme.
2. Calculer le polynôme S, la somme des deux polynômes.
3. Calculer le polynôme P, le produit des deux polynômes.
4. Calculer $g_1(1/2)$, $g_2(1)$, $S(2)$ et $P(3/2)$.
5. Tracer l'évolution de P et S sur l'intervalle $[-5,5]$.

Exercice 2:

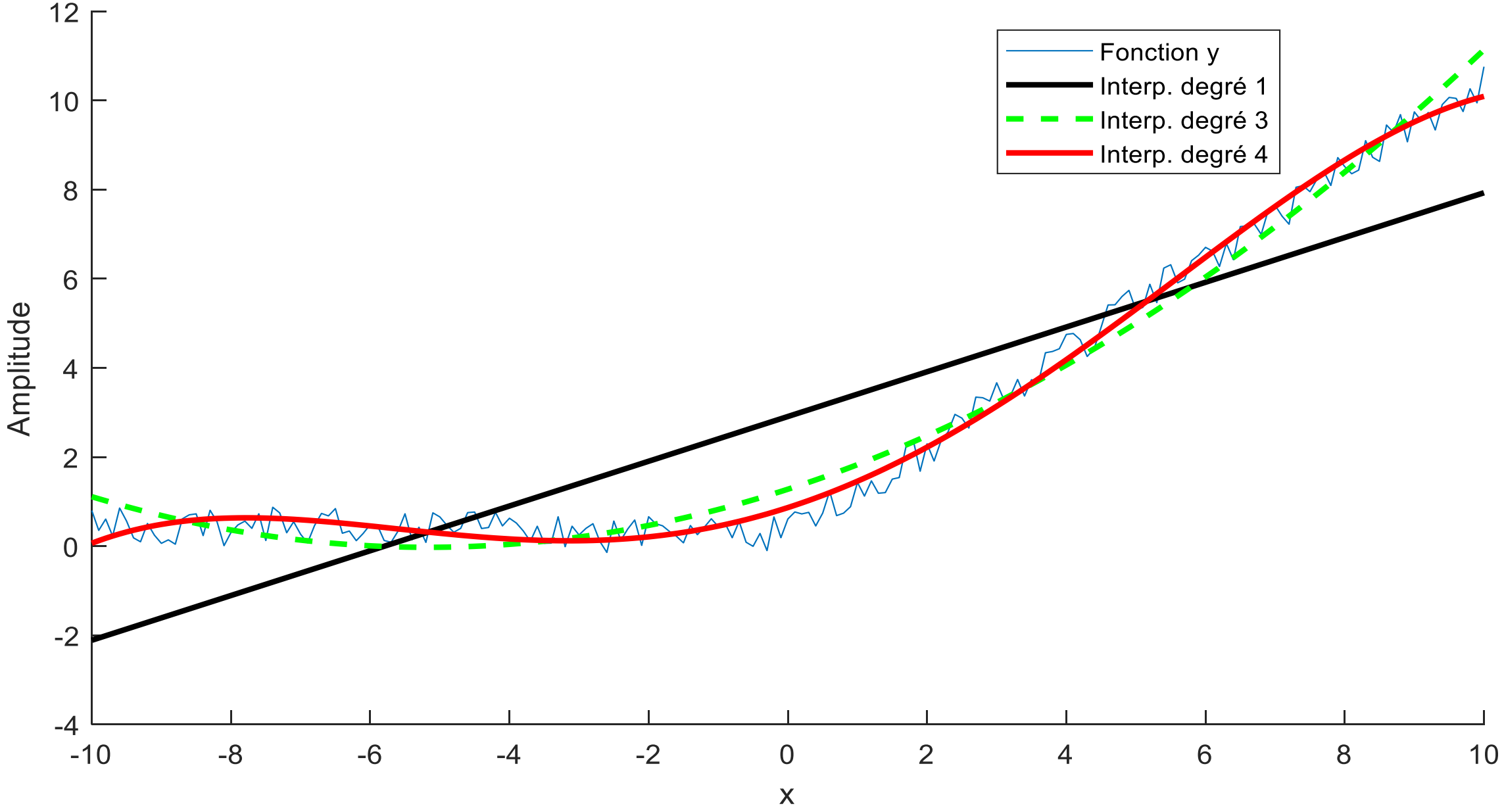
La courbe de la fonction suivante représente le résultat expérimental obtenu pour des données x allant de -10 à 10 avec un pas 0.1 :

$$y(x) = \frac{x}{1 + e^{-x}} + 0.9 \times rand\big(1, length(x)\big)$$

Dans un nouveau script:

1. Tracer la courbe de la fonction y en fonction des données x.
2. Donner le polynôme d'interpolation $\boldsymbol{P}$ de degré 1 permettant d'approcher la courbe de y au sens des moindres carrés.
3. Tracer sur le même graphe la courbe de la fonction y et du polynôme d'interpolation $\boldsymbol{P}(\boldsymbol{x})$.
4. Essayer de faire la question 3 pour des degrés du polynôme $\boldsymbol{P}$ égalent à 3 et 4. Qu'est ce que vous remarquez ?

Graphe obtenu:



Exercice 3:

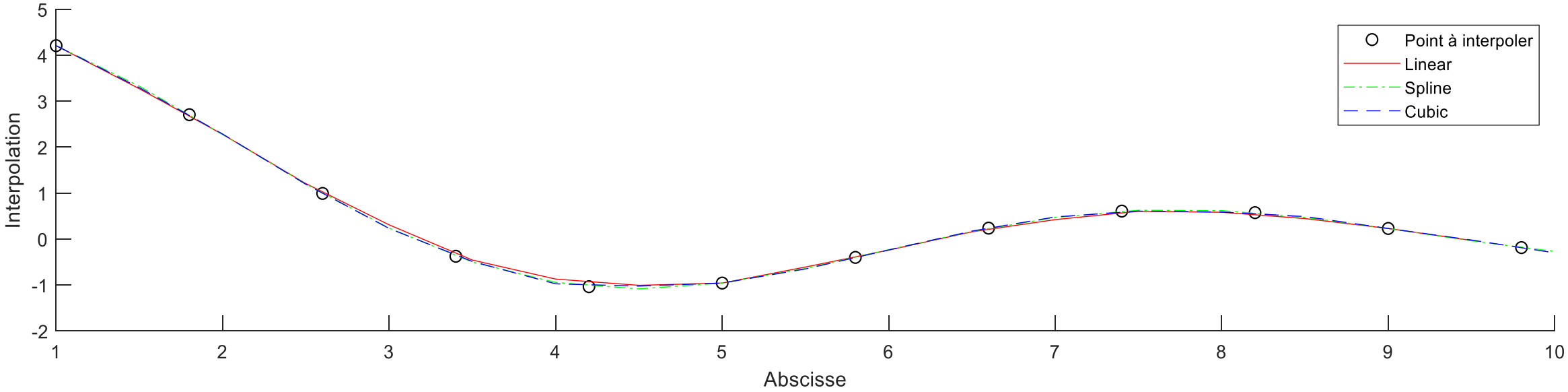
On considère la fonction sinus cardinal suivante:

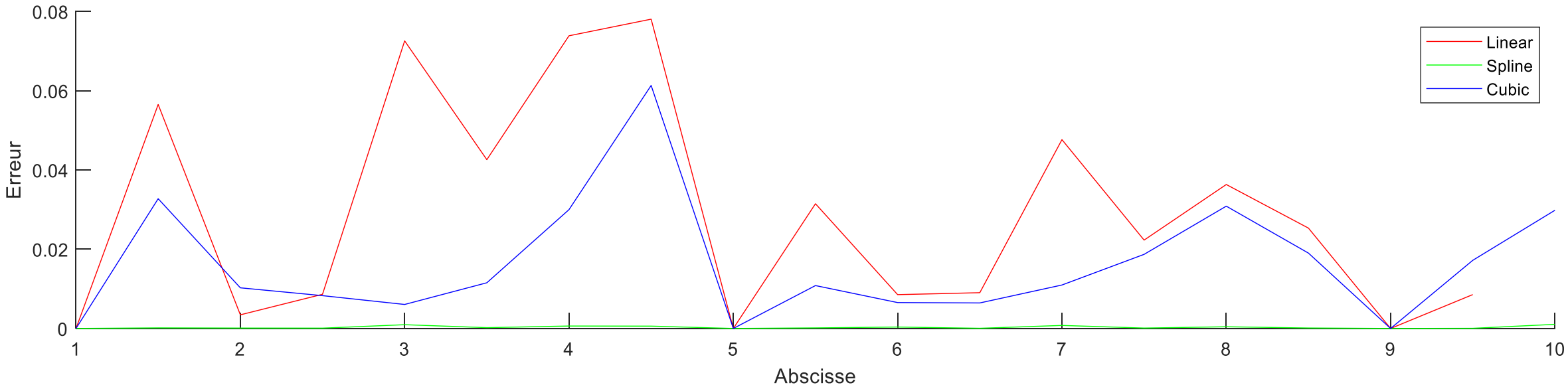
$$f(x) = \frac{5\sin(x)}{x}$$ avec x allant de 0 à 10 avec un pas 0.8.

Dans un nouveau script:

1. À l'aide de la commande subplot:
   - Tracer sur la première fenêtre, la courbe des points à interpoler et de différents polynômes d'interpolation **linear, spline** et **cubic** de la fonction f avec un pas 0.5.
   - Tracer sur la deuxième fenêtre, la courbe d'erreur entre les points à interpoler et chaque méthode d'interpolation.
2. Donner les titres et les légendes pour chaque axe et fonction pour chaque fenêtre.

2. Graphes obtenus:





Exercice 4:

Le tableau ci-dessous représente la vitesse d'un mobile en fonction de la distance parcourue :

$\boldsymbol{d(m)}$	0	200	400	600	800	1000
$\boldsymbol{V(m/s)}$	0	48,5	62	71,5	75	76,5

❖ À l'aide des commandes de Matlab:

❑ En se basant sur l'interpolation de **Lagrange** trouver la valeur de la vitesse à: $d = 240m, 655m$ et $820m$.

Exercice 5:

Le tableau ci-dessous représente les masses volumiques $\boldsymbol{R}$ d'un matériau pour quelques valeurs de température:

i	Température T en (°C)	Masse volumique $\boldsymbol{R}$ en $(\boldsymbol{kg/m^3})$
1	94	929
2	205	902
3	371	860

1. Écrire la formule d'interpolation de Lagrange qui permet d'interpoler les différents points de données précédentes.
2. Créer un algorithme qui permet de calculer la masse volumique $\boldsymbol{Ri}$ pour $T = 251\ °C$, $T = 300\ °C$ et $T = 820\ °C$, , en utilisant l'interpolation de Lagrange.
3. Déterminer le polynôme d'interpolation de **R**.

Exercice 6:

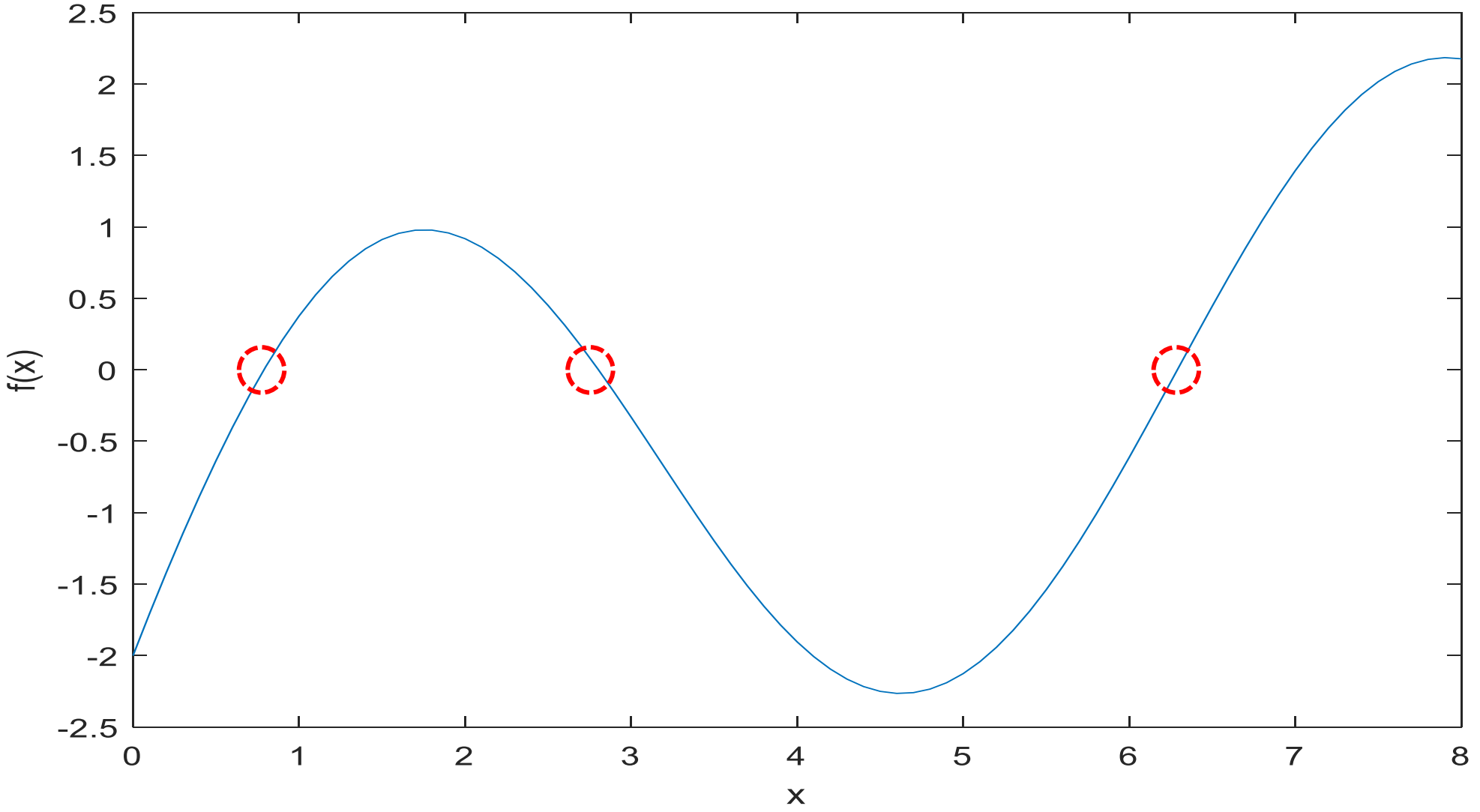
Soit la fonction : $$f(x) = \log(x+1)+2\sin(x) - 2$$

Sur l'intervalle $[0\ ,8]$:

1. Déterminer le nombre des racines de f (graphiquement)
2. Déterminer la position approximative de ces racines.
3. En se basant sur la méthode de bissection, créer **une fonction** qui permet de:
   - Déterminer les racines de f, avec une précision de 10^{-9} .
   - Calculer le nombre d'itérations nécessaire pour trouver cette racine.

Dans chaque intervalle approximatif.

1. Graphe obtenu pour déterminer le nombre de racines de f (3 racines):



Exercice 7:

Soit la fonction :

$$f(x) = x^3 - 2x^2 + 2$$

1. Démontrer mathématiquement et graphiquement que f admet une racine sur l'intervalle $[-1\ ,\ 1]$. (Prenez un pas de 0.1)

2. En se basant sur la méthode de **Newton-Raphson**, déterminer la valeur de cette racine pour une précision égale à 10^{-6} , sachant que le point initial $x_1 = -0.8$.

3. Calculer le nombre d'itérations nécessaire pour atteindre la racine.

FIN