Université Paris-Sud | **Travaux pratiques**
L3 Mathématiques | **2013-2014**
M315 - Calcul Scientifique

Feuille de TP 4 et 5 - Interpolation

Exercice 1. [Matrices de Vandermonde et polynômes]
L'objectif de cet exercice est de créer certaines fonctions qui seront utiles pour calculer le polynôme d'interpolation d'une fonction f en n points donnés.

1. Écrivez une fonction `vdm` qui a pour argument un vecteur `x` $= (x_1, \ldots, x_n)$ et qui retourne la matrice de Vandermonde associée

$$V(x_1, \ldots, x_n) = \begin{bmatrix} 1 & x_1 & \ldots & x_1^{n-1} \\ 1 & x_2 & \ldots & x_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \ldots & x_n^{n-1} \end{bmatrix}.$$

Étant donnés un polynôme $P = a_0X^n + a_1X^{n-1} + \ldots + a_{n-1}X + a_n \in \mathbb{R}_n[X]$ et un réel x, la méthode d'Horner est une méthode qui permet de calculer la valeur $P(x)$. Cette méthode consiste à calculer la séquence

$$\begin{cases} p_0 = a_0 \\ p_{i+1} = p_i x + a_{i+1}, \text{ pour } 0 \leq i \leq n-1. \end{cases}$$

On a alors $P(x) = p_n$.

2. Écrivez une fonction `y = algohorner(p,x)` qui a pour argument un vecteur `p` $= (a_0, a_1, \ldots, a_n)$ des coefficients d'un polynôme $P = a_0X^n + a_1X^{n-1} + \ldots, a_{n-1}X + a_n$ et un vecteur `x` $= (x_1, \ldots, x_k)$ et qui retourne un vecteur `y` $= (y_1, \ldots, y_k)$ avec $y_j = P(x_j)$, $j = 1, \ldots, k$, par la méthode de Horner.

   On rappelle qu'un polynôme est représenté en `Matlab` par un vecteur ligne contenant la liste des coefficients par ordre de degré décroissant. Par exemple le polynôme $3x^2 + 2x + 1$ s'écrit `[3 2 1]`. Ainsi, la fonction `polyval` de `Matlab` rend le même résultat que notre fonction `algohorner`.

   Vous testerez l'algorithme de Horner en effectuant le tracé sur $[0, 1]$ du polynôme $P(X) = 2X^2 - 3X + 1$, où les valeurs seront calculées par `algohorner`.

Exercice 2. [Interpolation par Vandermonde]
L'objectif de cet exercice est de déterminer et de tracer le polynôme interpolateur de Lagrange en utilisant les matrices de Vandermonde.

Soient $(x_1, y_1), \ldots, (x_N, y_N)$ N couples de réels avec $x_1, \ldots, x_N$ distincts. Le polynôme d'interpolation de Lagrange aux N points (x_i, y_i), $i \in \{1, \ldots, N\}$, est l'unique polynôme P de degré inférieur ou égal à $N - 1$ tel que $P(x_i) = y_i$, $\forall\ i \in \{1, \ldots, N\}$. Si $P(X) = a_0X^{N-1} + a_1X^{N-2} + \cdots + a_{N-2}X + a_{N-1}$, les coefficients a_0, $a_1, \ldots, a_{N-1}$ sont solution d'un système linéaire dont la matrice est une matrice de Vandermonde.

1. Écrivez une fonction `y=InterpVdm(X,Y,x)` dont les arguments sont trois vecteurs `X` $= (X_1, \ldots, X_N)$, `Y` $= (Y_1, \ldots, Y_N)$ et `x` $= (x_1, \ldots, x_k)$. Cette fonction calcule le polynôme interpolateur de Lagrange P aux points (X_i, Y_i), $1 \leq i \leq N$, en résolvant le système linéaire associé (utiliser la fonction `vdm` programmée au TP précédent), et retourne un vecteur `y` $= (y_1, \ldots, y_k)$ tel que $y_j = P(x_j)$, pour $j = 1, \ldots, k$. Le vecteur `y` est calculé en évaluant $P($`x`$)$ par la méthode de Horner (en utilisant la fonction `algohorner`, programmée au TP précédent).

   Pour résoudre le système linéaire $AX = b$, utiliser la commande `Matlab  A\b`.

2. Tester l'algorithme `InterpVdm` pour les fonctions

$$f_1(x) = \exp(-3(x-1.2)^2), \;\; f_2(x) = \frac{x^2-2}{1+2x}, \;\; f_3(x) = \frac{1}{1+(x-1.5)^2}, \;\; f_4(x) = \frac{\sin(2\pi x)}{1.1-\sin(\pi x)},$$

dans l'intervalle $I = [1,3]$ avec $N \in \{4,8,12,16\}$ points $X_i,\ i=1,\dots,N$, uniformément répartis dans I.

Pour chacune de ces fonctions, créer deux graphiques. Sur le premier, tracer la fonction et ses interpolées pour les différentes valeurs de N. Sur le second, tracer l'erreur, différence en valeur absolue entre la fonction et l'interpolée, pour les différentes valeurs de N. Rajouter des titres et des légendes aux graphiques.

3. Calculer pour $N = 16$, le conditionnement de la matrice de Vandermonde V associée aux 16 points uniformément distribués dans l'intervalle $[1,3]$. Calculer, pour la fonction f_4, la solution a de $Va = Y$ donnée par la commande `Matlab` $a = V\backslash Y$, où $Y = (f_4(X_1),\dots,f_4(X_N))^T$, et afficher la valeur de $\|Va-Y\|$. Refaire ces calculs pour $N = 20$. Commenter les résultats obtenus. La méthode que l'on vient d'implémenter pour calculer le polynôme d'interpolation de Lagrange d'une fonction f quelconque semble-t-elle être très précise ?

4. Afin de tester la stabilité de la méthode, tester l'algorithme en modifiant légèrement la valeur de la fonction, c'est-à-dire qu'au lieu de prendre $Y_i = f(X_i)$, pour $1 \le i \le N$, où f est une des fonctions de la question 2.), on pourra prendre $Y_i = f(X_i) + (-1)^{r_i}10^{-3}$, où r_i vaut de façon aléatoire 0 ou 1. Un moyen de créer ce vecteur $r = (r_1,\dots,r_N)$ est d'utiliser la commande `r=rand(1,length(X))<0.5;`

5. Pour la fonction f_3, calculer pour les mêmes valeurs de N que dans la question 2.), le polynôme d'interpolation de Lagrange aux points de Tchebychev, définis par

$$x_k = \frac{a+b}{2} + \frac{b-a}{2}\cos\left(\frac{(2k-1)\pi}{2N}\right),\ k = 1,\dots,N.$$

Représenter dans une première figure la fonction et les polynômes d'interpolation pour les différentes valeurs de N. Créer ensuite 4 graphiques, pour les 4 valeurs de N considérées. Dans chacun de ces graphiques, représenter l'erreur entre f et son interpolée de Lagrange aux points de Tchebychev et l'erreur entre f et son interpolée de Lagrange aux points uniformément distribués dans I. Rajouter des titres et des légendes aux graphiques. Remarquer les différences entre les deux interpolations.

Exercice 3.
Dans cet exercice, nous proposons un algorithme de calcul du polynôme interpolateur de Lagrange qui est plus rapide et plus stable que celui utilisant la matrice de Vandermonde.

Étant donné n points $X_1,\dots,X_n$ deux à deux distincts, on choisit d'écrire le polynôme interpolateur de Lagrange aux points $(X_i)_{1\le i\le n}$ sous la forme

$$P(t) = a_1 + a_2(t-X_1) + a_3(t-X_1)(t-X_2) + \ldots + a_n(t-X_1)\ldots(t-X_{n-1}).$$

Remarque : $a_1,\dots,a_n$ sont ainsi les coefficients de P dans la base $\big(1,\ (t-X_1),\dots,\ (t-X_1)\dots(t-X_{n-1})\big)$ de $\mathbb{R}_{n-1}[X]$.

1. Soit $P(t) = a_1$. Montrez que P est le polynôme interpolateur de Lagrange au point (X_1, a_1). Si

$$P(t) = a_1 + a_2(t-X_1) + a_3(t-X_1)(t-X_2) + \ldots + a_n(t-X_1)\ldots(t-X_{n-1})$$

est le polynôme interpolateur aux points $(X_1,Y_1),\dots,(X_n,Y_n)$, montrer que le polynôme

$$Q(t) = a_2 + a_3(t-X_2) + \ldots + a_n(t-X_2)\ldots(t-X_{n-1}),$$

de degré $n-2$, est le polynôme interpolateur de Lagrange aux points $\left(X_i, \frac{Y_i-Y_1}{X_i-X_1}\right)$, $i = 2,\dots,n$.

Comprendre que, de ces relations, nous pouvons déduire l'algorithme récursif suivant pour calculer les coefficients $a_1,\dots,a_n$:

```
function a = coeff(X,Y)
si taille(X)=1
      a=Y
sinon
      a = [Y(1), coeff(XX,YY)]
      avec XX = [X(2),..., X(end)]
      et   YY = [(Y(2)-Y(1))/(X(2)-X(1)),..., (Y(end)-Y(1))/(X(end)-X(1))]
```

2. Programmer cet algorithme dans une fonction `coeff.m`.
3. À l'image de l'algorithme programmé dans la fonction `algohorner` du TP précédent, programmer un algorithme de type Horner pour évaluer un polynôme P, qui s'écrit sous la forme

$$P(t) = a_1 + a_2(t - X_1) + a_3(t - X_1)(t - X_2) + \ldots + a_n(t - X_1)\ldots(t - X_{n-1}),$$

   en k points $x_1, \ldots, x_k \in \mathbb{R}$.
4. Tester les algorithmes programmés pour évaluer les polynômes d'interpolation de Lagrange de la fonction

$$f_4(x) = \frac{\sin(2\pi x)}{1.1 - \sin(\pi x)},$$

   aux N points uniformément répartis dans l'intervalle $[1, 3]$ et aux N points de Tchebychev, pour $N = 12$, puis pour $N = 20$, puis pour $N = 24$. Pour chaque valeur de N, tracer dans une figure la fonction f_4, les deux polynômes d'interpolation et les deux ensembles de points d'interpolation. Tracer dans une autre figure la différence en valeur absolue entre f_4 et le polynôme d'interpolation, pour les deux choix de points. Comparer les résultats obtenus dans cet exercice et dans l'exercice **1**.