TP 3 : Interpolation de Lagrange

On importera le module math en tapant `from math import *` en début de TP et on utilisera `abs(x) < prec` pour tester `x == 0`, avec `prec = 1e-15` (précision machine).
Toute fonction mathématique sera définie comme une procédure Python.

Exercice 1. Soit $X = (x_0, \ldots, x_n)$ une liste de nœuds deux-à-deux distincts et f une fonction définie au moins en ces nœuds. On écrit le polynôme d'interpolation de Lagrange de f aux nœuds X sous la forme

$$P(x) = \sum_{i=0}^{n} \frac{f(x_i)}{\tilde{L}_i(x_i)} \times \tilde{L}_i(x), \quad \text{où} \quad \tilde{L}_i(x) = \prod_{j \neq i} (x - x_j).$$

a. Écrire une procédure `CoeffLagrange(f,X)` qui retourne la liste `C` des coefficients apparaissant devant les polynômes $\tilde{L}_i$ dans l'expression ci-dessus.

b. Écrire une procédure `Lagrange(C,X,x)` qui calcule la valeur au point `x` du polynôme d'interpolation de Lagrange de f aux nœuds `X`, à l'aide de la liste `C` calculée par `CoeffLagrange`. On itérera un compteur global `mult` pour compter le nombre de multiplications.

c. Écrire une procédure `Noeuds(a,b,n)` qui retourne la liste X des $n+1$ réels $a = x_0 < x_1 < \cdots < x_n = b$ régulièrement espacés dans l'intervalle $[a, b]$.

d. On teste ces procédures sur la fonction $f : x \mapsto \sin(x)$.

   (i) Calculer la liste C correspondant à `X = Noeuds(-pi,pi,5)` et `f=sin`. Vérifier que `Lagrange` retourne les valeurs correctes aux nœuds `x = X[i]`.

   (ii) Calculer le polynôme d'interpolation P de la fonction sinus correspondant à `X = Noeuds(-pi,pi,3)`.
   Tracer les graphes de f et P entre 0 et 2π.
   On utilisera les valeurs des deux fonctions aux points de `Y=Noeuds(-pi,pi,99)`.

   (iii) Recommencer en remplaçant $n = 3$ par $n = 5$, 10, 20.
   Noter le nombre total de multiplications effectué à chaque fois.

e. Déterminer le nombre de multiplications effectuées par `Lagrange` en fonction de n.

f. Calculer $P(15)$ pour chacun des polynômes précédents ($n = 3$, 5, 10, 20). Que constate-t-on ?

Exercice 2. Soit $X = (x_0, \ldots, x_n)$ une liste de nœuds deux-à-deux distincts et f une fonction définie au moins en ces nœuds. Le polynôme d'interpolation de Lagrange de f aux nœuds X se décompose dans la *base de Newton* sous la forme

$$P = \sum_{i=0}^{n} f[x_0, \ldots, x_i] N_i, \quad \text{où} \quad N_i = \prod_{j<i} (X - x_j)$$

et les *différences divisées* $f[x_0, \ldots, x_j]$ sont définies par récurrence : $f[x_0] = f(x_0)$ et

$$f[x_0, x_1, \ldots, x_{k-1}, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}.$$

a. Écrire une procédure `PuissancesDivisées(f,X)` qui retourne la liste des puissances divisées ($f[x_0]$, $f[x_0, x_1]$, …, $f[x_0, x_1, \ldots, x_n]$). *Indication. On pourra effectuer plusieurs passages sur une liste de longueur $n+1$. Au premier passage la liste contient $f[x_0]$, $f[x_1]$, …, $f[x_n]$. Au deuxième passage elle contient $f[x_0]$, $f[x_0, x_1]$, …, $f[x_{n-1}, x_n]$. Après le dernier passage elle contient la liste recherchée.*

b. Écrire une procédure `Newton(D,X,x)` qui retourne la valeur au point `x` du polynôme d'interpolation de Lagrange de f aux nœuds `X`, à l'aide de la liste `D` des puissances divisées. On itérera un compteur global `mult` pour compter le nombre de multiplications.

c. Vérifier que la procédure retourne les valeurs correctes aux nœuds `x = X[i]`, comme à l'exercice 1.
Reprendre les tracés de graphes de l'exercice 1 et noter les nouveaux nombres de multiplications effectuées.
Quel est le nombre de multiplications effectuées par `Newton` en fonction de n ?

On rappelle la méthode de Horner-Ruffini pour calculer la valeur $P(x)$ d'un polynôme $P = \sum_{i=0}^{n} a_i X^i$ avec le nombre minimal de multiplications :

$$P(x) = a_0 + x(a_1 + x(\cdots + x(a_{n-1} + xa_n)\cdots)).$$

d. Écrire une procédure `Horner(D,X,x)` analogue à `Newton` qui minimise le nombre de multiplications effectuées, en utilisant une variante de la méthode de Horner-Ruffini. Vérifier le nombre de multiplications effectuées sur les tracés de graphes de l'exercice 1.

e. Soient P_1 et P_2 les polynômes d'interpolation de la fonction sinus sur la liste de noeuds `X = Noeuds(-pi,pi,50)` obtenus respectivement par la méthode de Lagrange et la méthode de Newton. On a en théorie $P_1 = P_2$. Cependant, on peut se demander quelle méthode est la plus stable numériquement, c'est à dire la moins sensible aux erreurs d'arrondi.

   (i) Déterminer la plus grande différence entre les valeurs $P_1(x)$ et $P_2(x)$, pour x dans la liste `Y1 = Noeuds(-pi, pi, 100)`. Les polynômes P_1, P_2 semblent-ils coïncider ?

   (ii) Recommencer avec `Y2 = Noeuds(20,30,100)`, qui est cette fois une liste de points loin des points d'interpolation. Que constate-t-on ? Tracer les graphes de P_1 et P_2 à l'aide des points de `Y2`. Quelle méthode semble la plus stable numériquement ?