TP noté du 14 novembre 2022

Les questions 1 et (au moins) 2a doivent être traitées avant les autres, avec une implémentation qui fonctionne. Ensuite, les questions 3, 4 et 5 sont indépendantes et peuvent être traitées dans l'ordre souhaité.

On importera les modules `math` et `matplotlib`. On utilisera `abs(x) < prec` pour tester `x == 0`, avec `prec = 1e-15` (précision machine). Toute fonction mathématique sera définie comme une procédure Python.

Soit $X = (x_0, \dots, x_n)$ une liste de nœuds deux-à-deux distincts. On pose

$$\tilde{L}_i(x) = \prod_{j \neq i} (x - x_j) \quad \text{et} \quad M_i(x) = \frac{\tilde{L}_i'(x)}{\tilde{L}_i(x)} = \sum_{j \neq i} \frac{1}{x - x_j}.$$

Soit f une fonction dérivable sur un intervalle contenant les nœuds x_i. Le polynôme d'interpolation de Hermite de f aux points $x_0, \dots, x_n$ est l'unique polynôme H_n de degré inférieur ou égal à $2n+1$ tel que $H_n(x_i) = f(x_i)$ et $H_n'(x_i) = f'(x_i)$ pour tout i. On admet qu'il est donné par la formule suivante :

$$H_n(x) = \sum_{i=0}^{n} \frac{\tilde{L}_i(x)^2}{\tilde{L}_i(x_i)^2} \times \left[f(x_i) + (x - x_i)(f'(x_i) - 2f(x_i)M_i(x_i)) \right].$$

On dispose d'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 $[a, b]$. Dans la suite H_n est le polynôme d'interpolation de Hermite de f en ces points.

1. (a) Écrire une procédure `CoeffHermite(f,g,X)` qui retourne les listes `C`, `D` des coefficients `C[i] =` $f(x_i)/\tilde{L}_i(x_i)^2$ et `D[i] =` $(g(x_i) - 2f(x_i)M_i(x_i))/\tilde{L}_i(x_i)^2$.
   (b) Écrire une procédure `Hermite(C,D,X,x)` qui calcule la valeur au point `x` du polynôme d'interpolation de Hermite de f aux nœuds `X`, à l'aide des listes `C`, `D` calculées par `CoeffHermite`.
2. On teste ces procédures sur la fonction $f_1 : x \mapsto \sin(x+2)$ et les nœuds `X = Noeuds(-pi,pi,5)`.
   (a) Calculer les listes `C`, `D` correspondant à f_1 et `X` puis vérifier que `Hermite` retourne les valeurs correctes aux nœuds `X[i]` pour le polynôme d'interpolation de Hermite de f.
   (b) Vérifier que la dérivée du polynôme de Hermite H_n calculé par `Hermite` prend les valeurs correctes aux nœuds `X[i]`. On utilisera la valeur approchée suivante de $H_n'(x)$:

$$H_n'(x) \approx \frac{H_n(x + \text{eps}) - H_n(x)}{\text{eps}}$$

   avec `eps = 1e-10` et on testera les valeurs de $H_n'(x)$ à `prec/eps` prêt (au lieu de `prec`).
3. On souhaite comparer l'interpolation de Hermite et l'interpolation de Lagrange pour la même fonction $f_1(x) = \sin(x+2)$. On dispose d'une procédure `Lagrange(C,X,x)` qui calcule la valeur en `x` du polynôme d'interpolation de Lagrange L_n de `f` aux nœuds `X`, à l'aide de la liste des coefficients `C` retournée par la procédure `CoeffLagrange(f,X)`, comme au TP 3.
   (a) Tracer sur un même graphique les graphes de f_1 et de ses polynômes d'interpolation de Lagrange et de Hermite L_3 et H_3 aux nœuds `X = Noeuds(-pi,pi,3)`. Commenter (dans le fichier python).
   On tracera les graphes sur l'intervalle $[-2\pi, 2\pi]$ en plaçant 1000 valeurs de chaque fonction. On pourra limiter les ordonnées à l'intervalle $[-2.5, 2.5]$ et on légendera le graphique.
   (b) Il serait plus cohérent de comparer H_3 à un autre polynôme L_n. Lequel ? Ajouter le graphe de ce polynôme au graphique et commenter.
4. On souhaite étudier dans quelle mesure les polynômes H_n approximent la fonction interpolée f. On considère le cas des fonctions $f_1 : x \mapsto \sin(x+2)$ et $f_2 : x \mapsto 1/(1+5x^2)$ sur l'intervalle $[-1, 1]$.
   (a) Calculer $\|f_1 - H_n\|_{\infty,Y} = \max_{y \in Y} |f_1(y) - H_n(y)|$ pour `Y = Noeuds(-1,1,1000)` et $n = 3, 4, 5, 6, 7$. Cela aurait-il du sens d'essayer des valeurs plus grandes de n ? *Répondre en commentaire dans le fichier python.*
   (b) Recommencer avec f_2. Le fait d'utiliser les polynômes de Hermite à la place des polynômes de Lagrange résout-il le phénomène de Runge ? *Répondre en commentaire dans le fichier python.*

Suite au verso

On note maintenant K_i le polynôme d'interpolation de Hermite de f aux deux seuls points x_i et x_{i+1}. Ainsi K_i est de degré au plus trois et vérifie $K_i(x_i) = f(x_i)$, $K_i'(x_i) = f'(x_i)$, $K_i(x_{i+1}) = f(x_{i+1})$, $K_i'(x_{i+1}) = f'(x_{i+1})$.
On peut alors interpoler f par la fonction K polynômiale par morceaux sur $[x_0, x_n]$ définie comme suit : $K(x) = K_i(x)$ si $x \in [x_i, x_{i+1}]$ avec $i \in \{0, \ldots, n-1\}$.

5. (a) Écrire une procédure `H3(f,g,a,b,x)` qui calcule, lorsque `g =` f', la valeur en `x` du polynôme d'interpolation de Hermite de `f` aux points `a`, `b`.

   (b) Écrire une procédure `Morceaux(f,g,X,x)` qui calcule la valeur en `x` du polynôme K défini ci-dessus.

   (c) On considère le cas de la fonction $f_2 : x \mapsto 1/(1+5x^2)$ sur l'intervalle $[-1, 1]$. Tracer sur un même graphique le graphe de f_2, de son polynôme de Lagrange aux nœuds `Noeuds(-1,1,41)`, de son polynôme de Hermite aux nœuds `Noeuds(-1,1,20)` et du polynôme K pour les nœuds `Noeuds(-1,1,41)`.
   On tracera les graphes sur l'intervalle $[-1, 1]$ *en plaçant 1000 valeurs de chaque fonction.*