Analyse Numérique

TP 1 : Interpolation polynomiale

On s'intéresse dans ce TP aux diverses méthodes vues en cours et/ou en TD pour le calcul de l'interpolation de Lagrange et d'Hermite.

Interpolation de Lagrange. On rappelle que le polynôme d'interpolation de Lagrange de degré n d'une fonction f sur l'ensemble des $n+1$ points $\{(x_i, f(x_i))\}_{i=0}^n$ s'écrit

$$P_n(x) = \sum_{i=0}^{n} f(x_i) L_i(x) \quad \text{avec} \quad L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} .$$

On fixe $n \in \mathbb{N}^*$, $x = (x_0, \cdots, x_n) \in \mathbb{R}^{n+1}$ avec $x_i \neq x_j$ $\forall i \neq j$ et une fonction f.

1. Implémenter une fonction Matlab appelée `base_lagrange.m` prenant en entrée un réel scalaire z, le support $\{x_0, \ldots, x_n\}$ ainsi qu'un entier k et qui renvoie $L_k(z)$ la valeur de la $k$ième fonction de Lagrange calculée au point z.
2. Afficher la fonction L_0 pour le support $\{0, 1, 2\}$ dans l'intervalle $[-1, 3]$.
3. Modifier la fonction précédente pour qu'elle devienne vectorielle, c'est à dire qu'elle accepte en entrée un vecteur z et qu'elle revoie un vecteur de même taille que $z = (z_i)_{i=1}^n$ dont la composante i vaut $L_k(z_i)$.
4. Afficher la fonction L_1 en rouge et L_3 en vert pour le support $\{0, 1, 2\}$ dans l'intervalle $[-1, 3]$.
5. Implémenter une fonction Matlab appelée `interp_lagrange.m` permettant de calculer P_n, le polynôme d'interpolation de Lagrange de f.
6. Soit $f : \mathbb{R} \to \mathbb{R}$ la fonction définie par $f(x) = \sin(x)$.
   a) Calculer numériquement les polynômes P_n d'interpolation de Lagrange de degré $n = 2$ et $n = 5$ sur l'intervalle $[0; \pi]$ de f avec des nœuds équirépartis.
   b) Dessiner ensuite dans une même figure les graphes de f et de P_n pour $x \in [0; \pi]$.
7. Soit f la fonction de Runge $f : \mathbb{R} \to \mathbb{R}$ définie par $f(x) = \frac{1}{1+x^2}$.
   a) Calculer numériquement les polynômes P_n d'interpolation de Lagrange de degré $n = 3, 4, 5$ et 10 sur l'intervalle $[-5; 5]$ de f avec des nœuds équirépartis.
   b) Dessiner ensuite dans un même plan les graphes de f et des P_n sur l'intervalle $[-5; 5]$.
   c) Répéter avec les nœuds de Tchebychev définis par $x_i = \frac{1}{2}\left(a + b + (b - a)\cos\left(\frac{2i+1}{n+1}\frac{\pi}{2}\right)\right)$, $i = 0, \ldots, n$.

Base de Newton. On rappelle que la base de Newton associée à l'ensemble des $n+1$ points $\{x_i\}_{i=0}^n \subset [a, b]$ est

$$\mathscr{B} := \{H^0(x) := 1\} \cup \left\{H^k(x) := \prod_{i=0}^{k-1}(x - x_i) \mid k \in \{1, \ldots, n\}\right\} .$$

La décomposition du polynôme $P_n \in \mathscr{P}_n$ interpolant une fonction f aux points $(x_i)_{i=0}^n$ s'écrit :

$$P_n(x) = \sum_{i=0}^{n} f[x_0, \ldots, x_i] H_k(x)$$

où $f[x_0, \ldots, x_i]$ désignent les différences divisées de f.

1. Ecrire un script matlab qui construit $n+1$ point équirépartis dans un intervalle $[a,b]$
2. Ecrire une fonction matlab appelée *base_newton* prenant en entrée
   ▷ le support $(x_i)_{i=0}^n$
   ▷ un entier $k \in \{0,\dots,n\}$
   ▷ un scalaire x
   et qui renvoie :
   ▷ le $k^{\text{ième}}$ élément de la base $\mathscr{B}$ (calculé en x).
3. Ecrire une fonction matlab appelée *diff_divise* prenant en entrée
   ▷ le support $(x_i)_{i=0}^n$
   ▷ la fonction f
   et qui renvoie :
   ▷ les coefficients $(f[x_0,\dots,x_i])_{i=0}^n$.
4. Ecrire une fonction matlab appelée *interp_newton* prenant en entrée
   ▷ le support $(x_i)_{i=0}^n$
   ▷ la fonction f
   ▷ un scalaire x
   et qui renvoie :
   ▷ le polynome (P_n calculé en x).
5. Ecrire un scrip matlab qui affiche les fonctions de la base $\mathscr{B}$ sur une même figure.
6. Ecrire un scrip matlab qui affiche la fonction f ainsi que le polynôme P_n sur une même figure.

Interpolation de Hermite. On rappelle que le polynôme d'interpolation de Hermite de degré $N = 2n+1$ d'une fonction f sur l'ensemble des $n+1$ points $\{(x_i, f(x_i))\}_{i=0}^n$ s'écrit

$$Q_N(x) = \sum_{i=0}^{n} \left(f(x_i)A_i(x) + f'(x_i)B_i(x) \right), \qquad c_i = \sum_{\substack{j=0 \\ j\neq i}}^{n} \frac{1}{x_i - x_j}$$

où

$$A_i(x) = (1 - 2(x - x_i)c_i)(L_i(x))^2, \qquad B_i(x) = (x - x_i)(L_i(x))^2.$$

1. Implémenter une fonction Matlab appelée `base_hermiteA.m` (resp. `base_hermiteB.m`) prenant en entrée un réel scalaire z, le support $\{x_0,\dots,x_n\}$ ainsi qu'un entier k et qui renvoie $A_k(z)$ (resp. $B_k(z)$). Les rendre vectorielles.
2. Afficher les fonctions A_0, A_1, B_0 et B_1 pour le support $\{0,1,2\}$ dans l'intervalle $[-1,3]$.
3. Implémenter une fonction Matlab appelée `interp_hermite.m` permettant de calculer P_n, le polynôme d'interpolation de Hermite de f.
4. Soit $f : \mathbb{R} \to \mathbb{R}$ la fonction définie par $f(x) = \sin(4\pi x)$.
   a) Calculer numériquement les polynômes P_n d'interpolation de Lagrange de degré $n = 3$ et Q_n d'interpolation de Hermite sur l'intervalle $[0;1]$ de f avec des nœuds équirépartis.
   b) Afficher les graphes de f, de P_n et de Q_n pour $x \in [0;1]$.
   c) Répéter avec $n = 4$.