Calcul scientifique

TD/TP N°2 : Interpolation numérique

Exercice 1 : *Interpolation de Lagrange*

Soit n un entier ≥ 1. On considère la subdivision régulière suivante de l'intervalle $[x_{\min}, x_{\max}]$:

$$x_i = x_{\min} + i\frac{x_{\max} - x_{\min}}{n}, \qquad i = 0, \dots, n.$$

On considère les fonctions suivantes définies par

$$\begin{array}{rcl} f_1 \colon [0,2] & \to & \mathbb{R} \\ x & \mapsto & 2^x - 1 \end{array} \qquad \begin{array}{rcl} f_2 \colon [-1,1] & \to & \mathbb{R} \\ x & \mapsto & |x| \end{array} \qquad \begin{array}{rcl} f_3 \colon [-1,1] & \to & \mathbb{R} \\ x & \mapsto & \begin{cases} -1 \text{ si } x < 0 \\ +1 \text{ si } x \geq 0 \end{cases} \end{array}$$

1. On suppose que $n = 2$. Calculer les polynômes de Lagrange $(L_i)_{0 \leq i \leq n}$ associés à ces intervalles. Calculer les polynômes d'interpolation P_k de f_k pour $k = 1, \dots, 3$ associés à la subdivision de la question précédente.
2. On note
$$\varepsilon_k(x) = |f_k(x) - P_k(x)|$$
l'erreur d'interpolation. Lorsque cela est possible, énoncer le théorème qui donne l'erreur d'interpolation en fonction de $f^{(n+1)}$. En déduire une majoration de cette erreur.

Exercice 2 : *Implémentation de l'interpolation de Lagrange*

On cherche à interpoler par des polynôme de Lagrange, la fonction $f_1(x) = \sin(x)$ sur l'intervalle $[-3\pi, 3\pi]$.

1. Définir une fonction `f1` qui renvoie $f_1(x) = \sin(x)$.
2. Définir une fonction `subdivision_reguliere` qui prend en argument deux réels `xmin` et `xmax` et un entier `N` ; et qui renvoie une liste (ou un tableau numpy) `x_inter` de `N` points équirépartis sur l'intervalle [`xmin`, `xmax`]. Pour ce faire vous pouvez utiliser la commande `np.linspace` ou entrer les points manuellement.
3. En utilisant la fonction précédente, créer une liste (ou un tableau numpy) `x_inter_1` qui définit une subdivision régulière $[-3\pi, 3\pi]$ avec $N = 5$ points d'interpolations.
4. Créer une liste (ou un tableau numpy) `y_inter_1` contenant les valeurs des ordonnées des points d'interpolation par la fonction f_1 tel que $f(x_i) = \sin(x_i)$ avec $x_i \in$ `x_inter_1`.
5. Définir une fonction `poly_lagrange(k, x, x_inter)` qui renvoie la valeur en un point x du k-ème polynôme de Lagrange L_k construit à partir des nœuds $(x_i)_{i=1,\dots,n}$.
6. Définir une fonction `poly_interpolation(x, x_inter, y_inter)` qui renvoie la valeur en un point x du polynôme d'interpolation de Lagrange P, construit à partir des points d'interpolation $(x_i, y_i)_{i=1,\dots,n}$.

On veut maintenant représenter sur un même graphique la fonction, son interpolée et les points d'interpolations.

7. Définir un entier `N_plot` assez grand (par exemple 200) qu'on utilisera pour tracer les fonctions.

8. Utiliser la fonction `np.linspace` pour générer un tableau `X_1` de `N_plot` abscisses, équiréparties sur le segment [`xmin`, `xmax`].
9. À partir de `X_1` générer un tableau NumPy `Y_1` d'ordonnées associées à la fonction f_1.
10. À partir de `X_1` générer un tableau NumPy `P_1` des valeurs du polynôme d'interpolation de Lagrange P_1 de la fonction f_1 : `P_1[i]` $= P_1$(`X[i]`).
11. Représenter sur un même graphique :
    (a) les points d'interpolation `x_inter_1` sous forme de points rouges,
    (b) la fonction $f_1(x) = \sin(x)$ par une ligne verte continue,
    (c) Le polynôme d'interpolation de Lagrange P_1 par une ligne bleue continue.

    On pensera à ajouter une légende à la figure.
12. Implémenter une fonction `interpolation_lagrange(f, xmin, xmax, Ninter, Xplot)` qui prend en argument une fonction `f`, deux réels `xmin` et `xmax`, un entier `Ninter` et un tableau de valeurs `Xplot` ; et qui, à l'aide des fonctions précédemment implémentées, renvoie le triplet `(x_inter, y_inter, P)` où : `x_inter` et `y_inter` sont les points d'interpolation de `f` sur [`xmin`, `xmax`], et leurs images par `f` ; et `P` est le polynôme d'interpolation de Lagrange de `f` sur [`xmin`, `xmax`] évalué en les points de `Xplot`.

On s'interesse maintenant au comportement de l'interpolée en fonction du nombre de points d'interpolation N.

13. Faites des tracés pour $N = 3, 4$ et 7 points d'interpolations sur plusieurs sous-figures, à l'aide de la fonction `interpolation_lagrange` et des subplots de matplotlib. Que constatez-vous ?
14. Essayer ensuite avec des multiples de 5 jusqu'à 20. Que constatez-vous ?
15. À partir de quel nombre de points d'interpolation considérez-vous que l'interpolée colle visuellement de façon satisfaisante à la fonction originale ?

Pour calculer l'erreur d'interpolation, on définit un ensemble de points `Xtest = np.linspace(xmin, xmax, 2000)`, et on calcule l'erreur en norme ℓ^∞ entre la fonction et son interpolée sur ces points :

$$E_n = \|f(\texttt{Xtest}) - P_n(\texttt{Xtest})\|_\infty = \max_{x \in \texttt{Xtest}} |f(x) - P_n(x)|$$

où f est la fonction à interpoler, P_n son interpolée avec n points d'interpolation.

16. Écrire une procédure `error_inf` qui évalue l'erreur en norme ℓ^∞ entre le polynôme d'interpolation P_n et la fonction f.
17. Pour la fonction f_1 donnée, tracer en fonction de n, l'évolution de l'erreur en norme ℓ^∞. On pourra représenter cette erreur en échelle logarithmique.
18. Reprendre la question précédente pour la fonction f_2 définie par :

$$f_2(x) = \frac{1}{1 + 25x^2}$$

    sur l'intervalle d'interpolation $[-1, 1]$.
19. Sur un même graphe, tracer la fonction f_2 et son polynôme d'interpolation P_n. Que constatez-vous ? Comment est appelé le phénomène observé ?