Problème de Mathématiques

Référence pp2008 — Version du 15 octobre 2025

Soit $f : [a, b] \to \mathbb{R}$, une fonction continue. Étant données $(n+1)$ abscisses deux à deux distinctes :

$$a \leqslant x_0 < x_1 < \cdots < x_n \leqslant b$$

on appelle **polynôme interpolateur de** f **aux points** x_i tout polynôme $P \in \mathbb{R}_n[X]$ qui coïncide avec la fonction f aux points x_i, c'est-à-dire tel que

$$\forall\, 0 \leqslant i \leqslant n, \quad P(x_i) = f(x_i).$$

Partie A. Existence du polynôme interpolateur

Pour tout entier $0 \leqslant i \leqslant n$, on définit le polynôme

$$\ell_i \in \mathbb{R}_n[X]$$

de la façon suivante :

$$\ell_i = \prod_{\substack{0 \leqslant k \leqslant n \\ k \neq i}} \frac{X - x_k}{x_i - x_k}$$

et on pose ensuite

$$L_n(f) = \sum_{i=0}^{n} f(x_i)\ell_i.$$

Il est manifeste que $L_n(f)$ est un polynôme. La fonction polynomiale associée à ce polynôme est définie par

$$\forall\, t \in \mathbb{R}, \quad L_n(f)(t) = \sum_{i=0}^{n} f(x_i)\ell_i(t).$$

1. Démontrer que $L_n(f)$ est bien un polynôme interpolateur de f aux points x_i, puis démontrer l'unicité d'un tel polynôme.

Partie B. Calcul effectif

Plus généralement, quels que soient les nombres $y_0, \ldots, y_n$, le polynôme

$$L_y = \sum_{i=0}^{n} y_i \ell_i$$

est l'unique polynôme $P \in \mathbb{R}_n[X]$ tel que

$$\forall\, 0 \leqslant i \leqslant n, \quad P(x_i) = y_i.$$

2. On suppose ici que

$$(x_0, x_1, x_2) = (-1, 0, 1) \quad \text{et} \quad (y_0, y_1, y_2) = (4, 0, 4).$$

Démontrer que le polynôme interpolateur est $4X^2$.

3. Écrire une fonction `lagrange(x, y, a)` en langage Python dont les arguments sont :
— la liste $x = (x_i)_{0 \leqslant i \leqslant n}$ des abscisses ;
— la liste $y = (y_i)_{0 \leqslant i \leqslant n}$ des ordonnées ;
— une abscisse $a \in \mathbb{R}$.
Cette fonction doit renvoyer la valeur du polynôme interpolateur au point a.
Si, par exemple, $x = (-1, 0, 1)$ et $y = (4, 0, 4)$, alors l'instruction `lagrange(x, y, 3)` doit renvoyer 36.

4. Chercher le polynôme interpolateur de f aux points x_i sous la forme

$$P = a_0 + a_1 X + \cdots + a_n X^n$$

revient à résoudre le système linéaire

$$\forall\, 0 \leqslant i \leqslant n, \quad P(x_i) = f(x_i)$$

qui peut s'écrire matriciellement sous la forme suivante :

$$V \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}.$$

4. a. Quelle est la taille de la matrice V ? Quels sont ses coefficients ?
4. b. On résout ce système par l'algorithme du pivot de Gauss. Indiquer la complexité de ce calcul en fonction de n. Comparer avec la complexité de l'algorithme que vous avez proposé à la question précédente.

Partie C. Erreur d'interpolation

Dans cette partie, on suppose que la fonction f est de classe $\mathscr{C}^{n+1}$ sur le segment $[a, b]$. On note

$$\sigma = \{x_0, x_1, \ldots, x_n\}$$

l'ensemble des abscisses d'interpolation et on considère le polynôme

$$\pi_\sigma = \prod_{i=0}^{n} (X - x_i) \in \mathbb{R}_{n+1}[X].$$

On veut démontrer, pour tout $x \in [a, b]$, la propriété $\mathscr{P}_x$ suivante :

$$\exists\, a < c_x < b, \quad f(x) - L_n(f)(x) = \frac{f^{(n+1)}(c_x)}{(n+1)!} \cdot \pi_\sigma(x).$$

5. Démontrer que la propriété $\mathscr{P}_x$ est vraie pour tout $x \in \sigma$.
6. Soit $p \in \mathbb{N}^*$. Démontrer que : si

$$\varphi \,:\, [a, b] \to \mathbb{R}$$

est une fonction p fois dérivable qui s'annule $(p+1)$ fois sur $[a, b]$, alors il existe $a < c_p < b$ tel que

$$\varphi^{(p)}(c_p) = 0.$$

7. On considère maintenant une abscisse $x \notin \sigma$. On définit la fonction F par

$$\forall\, t \in [a, b], \quad F(t) = f(t) - L_n(f)(t) - \lambda\, \pi_\sigma(t)$$

où λ est une constante réelle.
7. a. Expliquer pourquoi on peut choisir λ de telle sorte que

$$F(x) = 0.$$

7. b. Démontrer que, dans ce cas, la fonction F s'annule $(n+2)$ fois sur $[a, b]$. En déduire que la propriété $\mathscr{P}_x$ est vraie.
8. Démontrer que la fonction $f^{(n+1)}$ est bornée sur $[a, b]$. En déduire un réel positif K indépendant de n tel que

$$\|f - L_n(f)\|_\infty \leqslant \frac{K^{n+1}}{(n+1)!}\, \|f^{(n+1)}\|_\infty.$$

9. On considère ici $[a, b] = [-1, 1]$ et

$$\forall\, x \in [-1, 1], \quad f(x) = \frac{1}{1 + x^2}.$$

9. a. Démontrer, par exemple à l'aide d'une formule de Taylor, que

$$\forall\, k \in \mathbb{N}, \quad \|f^{(2k)}\|_\infty \geqslant (2k)!.$$

9. b. Que dire de $\|f - L_n(f)\|_\infty$ lorsque n devient grand ?

Solution ❀ Interpolation de Lagrange

Partie A. Existence du polynôme interpolateur

1. Il est clair les ℓ_i sont tous des polynômes de degré n et que $\ell_i(t) = 1$ pour $t = x_i$ et $\ell_i(t) = 0$ pour $t = x_j$ avec $j \neq i$. Par conséquent, $L_n(f)$ est un polynôme de degré *inférieur à* n et que

$$\forall\, 0 \leqslant j \leqslant n, \quad L_n(f)(x_j) = \sum_{i=0}^{n} f(x_i)\delta_{i,j} = f(x_j).$$

Donc $L_n(f)$ est bien un polynôme interpolateur de f aux points x_i.

❧ Si P est un polynôme interpolateur de f aux points x_i, alors

$$\deg[P - L_n(f)] \leqslant \max\{\deg P, \deg L_n(f)\} \leqslant n$$

et d'autre part,

$$\forall\, 0 \leqslant j \leqslant n, \quad P(x_j) = L_n(f)(x_j).$$

Comme les x_j sont deux à deux distincts, le polynôme $P - L_n(f)$ admet au moins $(n+1)$ racinces distinctes, alors que son degré est inférieur à n : c'est donc le polynôme nul et $P = L_n(f)$. L'unicité du polynôme interpolateur est ainsi démontrée.

Partie B. Calcul effectif

2. Il est clair que $4x_i^2 = y_i$ pour tout $i \in \{0, 1, 2\}$ et que $\deg(4X^2) = 2$. Comme il existe un, et un seul, polynôme interpolateur, le polynôme $4X^2$ est bien cet unique polynôme interpolateur.

3. On définit une fonction auxiliaire `interp(i, x, a)` qui évalue au point a le polynôme ℓ_i.

```
def interp(i, x, a):
  n, li = len(x), 1
  for k in range(n):
    if (k!=i):
      li *= (a-x[k])/(x[i]-x[k])
  return li
```

Le code de la fonction `lagrange(x, y, a)` s'en déduit alors sans peine.

```
def lagrange(x, y, a):
  n, s = len(x), 0.
  for i in range(n):
    s += y[i]*interp(i, x, a)
  return s
```

REMARQUE.— Comme disait (à peu près) mon vénéré collègue René D., un bon code ne fait qu'*une seule chose à la fois*.

4. a. La matrice V appartient à $\mathfrak{M}_{n+1}(\mathbb{R})$ (elle envoie un vecteur de $\mathbb{R}^{n+1}$ sur un vecteur de $\mathbb{R}^{n+1}$).

Sur la colonne i, on traduit l'équation

$$f(x_i) = P(x_i) = \sum_{j=0}^{n} a_j x_i^j.$$

Quels que soient $0 \leqslant i, j \leqslant n$, le coefficient de la matrice situé à l'intersection de la i-ème ligne et de la j-ème colonne est donc

$$x_i^j.$$

REMARQUE.— Comme disent (à peu près) Alan Moore et David Lloyd, *V for Vandermonde*.

4. b. Comme il s'agit d'un système de $(n+1)$ équations en $(n+1)$ inconnues, la complexité de la résolution par l'algorithme du pivot est en $\mathcal{O}(n^3)$.

REMARQUE.— Soyons sérieux : $\mathcal{O}\big((n+1)^3\big)$ et $\mathcal{O}(n^3)$, c'est pareil !

❧ Dans tous les calculs à effectuer (pour chacun des deux algorithmes), l'opération élémentaire la plus coûteuse est la multiplication. Nous ne compterons donc que les multiplications.

Pour bien comparer avec l'algorithme proposé plus haut, il faut préciser ce qu'on cherche à calculer : on notera e, le nombre d'évaluations du polynôme interpolateur à effectuer.

Lagrange Chacune de ces évaluations effectue $\mathcal{O}(n)$ multiplications et $\mathcal{O}(n)$ appels à la fonction auxiliaire `interp`. Chaque appel à `interp` effectue $\mathcal{O}(n)$ multiplications. Par conséquent, chaque évaluation du polynôme interpolateur effectue $\mathcal{O}(n^2)$ multiplications. Le coût global *en temps* des évaluations est donc en $\mathcal{O}(e \cdot n^2)$. La complexité *spatiale* est négligeable (on n'enregistre aucune donnée).

Gauss L'algorithme du pivot s'exécute, on l'a dit, en $\mathcal{O}(n^3)$ multiplications. Un fois que cet algorithme est appliqué, on connaît les coefficients du polynôme interpolateur (complexité *spatiale* en $\mathcal{O}(n)$) et les évaluations de ce polynômes ont un coût unitaire en $\mathcal{O}(n)$ (algorithme de Horner ou assimilé). Le coût global *en temps* est donc en

$$\mathcal{O}(n^3 + en).$$

L'algorithme de Lagrange l'emporte donc facilement pour $e = o(n)$; les deux algorithmes se valent pour $e = \Theta(n)$; l'algorithme du pivot l'emporte pour $n = o(e)$.

❧ On pourrait chercher à améliorer le calcul du polynôme interpolateur (on a proposé le code le plus simple qui soit). Il est également possible d'inverser une matrice de Vandermonde avec une complexité en $\mathcal{O}(n^2)$... Mais ça n'a rien d'immédiat.

❧ Par ailleurs, on compare ici deux algorithmes numériques sans se soucier de la stabilité des calculs : est-on bien sûr qu'on obtient des résultats aussi précis avec chacun des deux algorithmes ?

Partie C. Erreur d'interpolation

5. Si $x \in \sigma$, alors $f(x) = L_n(f)(x)$ par construction du polynôme interpolateur $L_n(f)$. Par ailleurs, par définition du polynôme π_σ, on a aussi $\pi_\sigma(x) = 0$. La propriété $\mathscr{P}_x$ est donc vérifiée

$$0 - 0 = \frac{f^{(n+1)}(c_x)}{(n+1)!} \cdot 0$$

en prenant c_x n'importe où dans l'intervalle $]a, b[$.

6. Considérons les points où la fonction φ s'annule :

$$a \leqslant x_0^0 < x_1^0 < \cdots < x_p^0 \leqslant b.$$

Supposons que, pour un entier $0 \leqslant i < p$, il existe $(p - i + 1)$ abscisses

$$a \leqslant x_0^i < x_1^i < \cdots < x_{p-i}^i \leqslant b$$

où la fonction $\varphi^{(i)}$ s'annule. Comme $i < p$, la fonction $\varphi^{(i)}$ est dérivable sur chaque segment $[x_k^i, x_{k+1}^i]$ (donc continue sur le segment et dérivable sur l'intervalle ouvert) et en appliquant le Théorème de Rolle sur chacun de ces intervalles, il existe $(p - i)$ réels x_k^{i+1} tels que

$$\begin{aligned} a \leqslant x_0^i < x_0^{i+1} < x_1^i < \cdots \\ < x_k^i < x_k^{i+1} < x_{k+1}^i < \cdots \\ < x_{p-i-1}^i < x_{p-i-2}^{i+1} < x_{p-i}^i \leqslant b \end{aligned}$$

et que

$$\forall\, 0 \leqslant k < p, \quad \varphi^{(i+1)}(x_k^{i+1}) = 0.$$

On a ainsi démontré par récurrence (finie !) que, pour tout entier $0 \leqslant i \leqslant p$, il existe $(p - i + 1)$ abscisses

$$a \leqslant x_0^i < x_1^i < \cdots < x_{p-i}^i \leqslant b$$

où la fonction $\varphi^{(i)}$ s'annule. En particulier, pour $i = p$, il existe une abscisse $c_p = x_0^p$ telle que $a < c_p < b$ et $\varphi^{(p)}(c_p) = 0$.

REMARQUE.— Raisonnement classique. La principale difficulté réside dans l'hypothèse de récurrence : on fait une hypothèse pour $0 \leqslant i < p$ et la conclusion vaut pour $0 \leqslant i \leqslant p$. (La propriété n'est pas héréditaire pour $i = p$!)

7. a. Comme $x \notin \sigma$, alors $\pi_\sigma(x) \neq 0$ et l'équation

$$F(x) = 0$$

d'inconnue λ admet alors pour unique solution

$$\lambda = \frac{F(x) - f(x) + L_n(f)(x)}{\pi_\sigma(x)}.$$

REMARQUE.— Pour $x \in \sigma$, le dénominateur serait nul !

7. b. Comme on l'a vu au **[5.]**, la fonction F s'annule en chaque point de σ (indépendamment du choix de λ). D'autre part, on a choisi λ au **[7.a.]** de telle sorte que F s'annule aussi au point $x \notin \sigma$.

La fonction F est donc de classe $\mathscr{C}^{n+1}$ (combinaison linéaire de f, de classe $\mathscr{C}^{n+1}$, et de fonctions polynomiales) sur $[a, b]$ et s'annule en $(n+2)$ points distincts. On peut appliquer le résultat du **[6.]** avec $p = n+1$. Il existe donc un réel $a < c_x < b$ tel que $F^{(n+1)}(c_x) = 0$.

Comme $L_n(f)$ est une fonction polynomiale dont le degré est inférieur à n par construction, sa dérivée $(n+1)$-ième est identiquement nulle. Comme π_σ est une fonction polynomiale de degré $(n+1)$, sa dérivée $(n+1)$-ième est constante, égale à $(n+1)!$. Par conséquent,

$$\forall\, t \in [a, b], \quad F^{(n+1)}(t) = f^{(n+1)}(t) - \lambda \cdot (n+1)!$$

et en particulier pour $t = c_x$:

$$0 = f^{(n+1)}(c_x) - \lambda \cdot (n+1)!$$

ce qui nous donne

$$\lambda = \frac{f^{(n+1)}(c_x)}{(n+1)!}$$

et la propriété $\mathscr{P}_x$ est bien démontrée.

Bref : la propriété $\mathscr{P}_x$ est vraie pour chaque $x \in [a, b]$, qu'il appartienne à σ ou non.

8. Comme f est de classe $\mathscr{C}^{n+1}$, sa dérivée $f^{(n+1)}$ est continue sur le segment $[a, b]$ et donc bornée sur ce segment.

La borne supérieure $\|f^{(n+1)}\|_\infty$ est donc bien définie dans $\mathbb{R}$.

❧ Comme la propriété $\mathscr{P}_x$ est vraie pour tout $x \in [a, b]$, on a donc

$$\forall\, x \in [a, b], \quad \big|f(x) - L_n(f)(x)\big| \leqslant \frac{\|f^{(n+1)}\|_\infty}{(n+1)!} \cdot \big|\pi_\sigma(x)\big|.$$

Il reste à trouver un réel positif K indépendant de n et de x tel que

$$\forall\, x \in [a, b], \quad \big|\pi_\sigma(x)\big| \leqslant K^{n+1}.$$

Comme x et les x_i appartiennent tous au segment $[a, b]$, la distance $|x - x_i|$ est inférieure à la longueur totale $(b-a)$ du segment et donc

$$\forall\, x \in [a, b], \quad \big|\pi_\sigma(x)\big| = \prod_{i=0}^{n} |x - x_i| \leqslant (b-a)^{n+1}.$$

Avec $K = b - a$, on a donc

$$\forall\, x \in [a, b], \quad \big|f(x) - L_n(f)(x)\big| \leqslant \frac{K^{n+1}}{(n+1)!}\, \|f^{(n+1)}\|_\infty.$$

Comme le majorant trouvé est indépendant de $x \in [a, b]$, on peut passer au sup et en déduire que

$$\big\|f - L_n(f)\big\|_\infty \leqslant \frac{(b-a)^{n+1}}{(n+1)!}\, \|f^{(n+1)}\|_\infty.$$

9. a. La fonction f est de classe $\mathscr{C}^\infty$. Elle admet donc un développement limité à tout ordre au voisinage de 0. On sait que

$$\frac{1}{1+x^2} = \sum_{k=0}^{n} (-1)^k x^{2k} + o(x^{2n})$$

et, d'après la Formule de Taylor-Young, que

$$f(x) = \sum_{j=0}^{2n} \frac{f^{(j)}(0)}{j!} x^j + o(x^{2n}).$$

Par unicité du développement limité à l'ordre n, on en déduit que

$$\forall\, k \in \mathbb{N}, \quad \frac{f^{(2k)}(0)}{(2k)!} = (-1)^k.$$

Comme la borne supérieure est un majorant, on en déduit que

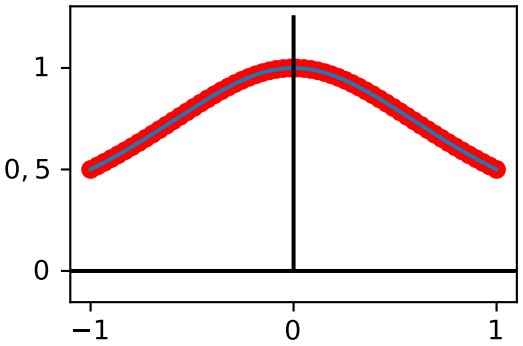
$$\forall\, k \in \mathbb{N}, \quad \|f^{(2k)}\|_\infty \geqslant \left|f^{(2k)}(0)\right| = (2k)!.$$

9. b. Dans ce cas particulier, $a = -1$, $b = 1$ et, pour $n = 2k-1$, le majorant de $\|f - L_n(f)\|_\infty$ trouvé au **[8.]** est supérieur à

$$\frac{2^{2k}(2k)!}{(2k)!} = 4^k.$$

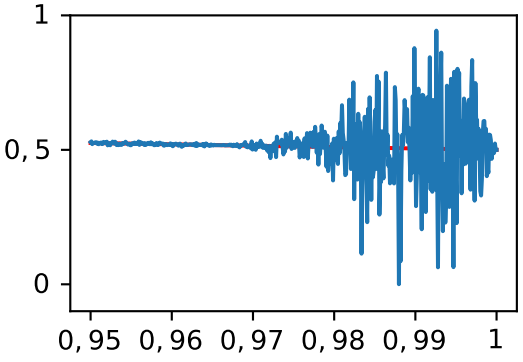
Comme ce majorant tend vers $+\infty$, il est probable que $\|f - L_n(f)\|_\infty$ ne tende pas vers 0...

REMARQUE.— Rien de plus logique : un polynôme interpolateur de f est construit pour interpoler la fonction f aux points de σ, pas pour fournir une approximation de f en dehors des points de σ!



La fonction f et 60 points d'interpolation

Lorsque les points d'interpolation sont mal choisis, on peut mettre en évidence le phénomène de Runge : le polynôme $L_n(f)$ est une mauvaise approximation de la fonction f, même lorsque l'entier n est grand.



Phénomène de Runge