Chap. 8 : Interpolation polynomiale de Lagrange

D'après le dictionnaire de l'Académie Française :

Interpolation : *XIVe siècle. Emprunté du latin impérial interpolatio, « action de changer çà et là », puis « altération, erreur ».*

En mathématiques : intercalation, entre certaines valeurs d'une fonction, de valeurs qu'on calcule par approximation et qui permettent d'établir une continuité de la fonction et de sa représentation graphique.

Table des matières

Pour les différents scripts Python, nous importons les bibliothèques suivantes :

```
from sympy import *
import matplotlib.pyplot as plt
import math
init_printing()
x = Symbol('x')
```

Dans tout ce qui suivra :

1. n désigne un entier naturel.
2. I est un intervalle de $\mathbb{R}$.
3. $x_0 < x_1 < \ldots < x_n$ sont $n+1$ points distincts de I.
4. f est une fonction de I vers $\mathbb{R}$.

1 Le problème de l'interpolation

Problème : trouver un polynôme P de degré inférieur ou égal à n tel que :

$$\forall i \in [\![0;n]\!], P(x_i) = f(x_i)$$

Nous allons voir que ce problème admet une unique solution. Le polynôme correspondant est appelé le **polynôme d'interpolation** de f aux points x_i.

Nous allons dans ce cours :

1. Montrer l'existence et l'unicité d'un tel polynôme.
2. Calculer efficacement la valeur du polynôme d'interpolation en un point x.
3. Majorer $|f(x) - P(x)|$ pour $x \in I$.
4. Voir comment rendre ce majorant aussi petit que possible en choisissant astucieusement les x_i.

2 Les polynômes élémentaires de Lagrange

Définition 2.1. *Pour $k = 0, \ldots, n$ le **kième polynôme de Lagrange** est le polynôme :*

$$L_k = \frac{\prod\limits_{j \neq k} X - x_j}{\prod\limits_{j \neq k} x_k - x_j}$$

C'est l'unique polynôme de degré inférieur ou égal à n qui s'annule en tous les x_j, sauf en x_k où il prend la valeur 1.

Proposition 2.2. *$\mathscr{B} = (L_0, L_1, \ldots, L_n)$ est une base de l'espace $\mathbb{R}_n[X]$ des polynômes de degré inférieur ou égal à n.*

Preuve :
Comme $Card\mathscr{B} = \dim \mathbb{R}_n[X]$ il suffit de montrer que $\mathscr{B}$ est libre.
Soient $\lambda_k, k = 0, 1, \ldots, nn + 1$ réels. Supposons que :

$$\sum_{k=0}^{n} \lambda_k L_k = 0$$

On a :

$$\forall (i,k) \in [\![0;n]\!]^2, L_k(x_i) = \delta_{ki}$$

où le symbole de Kronecker δ_{ki} vaut 1 si $k = i$ et 0 sinon. Évaluons l'égalité ci-dessus en x_i. Il vient :

$$0 = \sum_{k=0}^{n} \lambda_k L_k(x_i) = \sum_{k=0}^{n} \lambda_k \delta_{ki} = \lambda_i$$

ceci pour tout i. D'où la liberté.

Tout polynôme P de degré inférieur ou égal à n est donc combinaison linéaire des L_k, d'où leur nom "élémentaires".

Sous Python :
La fonction ***lagrange*** ci-dessous prend en paramètre un entier k, un réel x et une liste $xs = [x_0, \ldots, x_n]$ de réels distincts. Elle renvoie $L_k(x)$.

```
def lagrange(k, x, xs):
    p = 1
    n = len(xs) - 1
    for j in range(n + 1):
        if j != k:
            p *= (x - xs[j]) / (xs[k] - xs[j])
    return p
```

Ayant défini au préalable x comme un symbole, nous pouvons donc obtenir une expression explicite des L_k :

```
In [6]: expand(lagrange(1, x, [-2, -1, 0, 1, 2]))
```

Out[6]:

$$-\frac{x^4}{6}+\frac{x^3}{6}+\frac{2x^2}{3}-\frac{2x}{3}$$

```
In [7]: expand(lagrange(2, x, [-2, -1, 0, 1, 2]))
```

Out[7]:

$$\frac{x^4}{4}-\frac{5x^2}{4}+1$$

3 Subdivisions régulières

Soient $a, b \in \mathbb{R}, a < b$.
Une subdivision du segment $[a, b]$ est une suite :

$$x_0 = a < x_1 < \ldots < x_n = b.$$

Pour tout entier $n \geq 1$, la subdivision régulière à $n+1$ points de $[a, b]$ est définie par :

$$x_k = a + k\frac{b-a}{n}, k = 0, \ldots, n$$

Sous Python :

```
def subdi(a, b, n):
    d = (b - a) / n
    return [a + k * d for k in range(n + 1)]
```

On obtient par exemple une subdivision régulière à 11 points de l'intervalle $[-1; 1]$:

```
In [8]: def subdi(a, b, n):
            d = (b - a) / n
            return [a + k * d for k in range(n + 1)]

In [10]: xs = subdi(-1, 1, S(10))
         xs
```

Out[10]:

$$\left[-1,\ -\frac{4}{5},\ -\frac{3}{5},\ -\frac{2}{5},\ -\frac{1}{5},\ 0,\ \frac{1}{5},\ \frac{2}{5},\ \frac{3}{5},\ \frac{4}{5},\ 1\right]$$

4 Graphes des polynômes élémentaires

Nous pouvons maintenant effectuer quelques petits affichages. Commençons par le graphe des polynômes élémentaires. Pour :

```
xs = subdi(-1,1,S(3))
ps = [expand(lagrange(k, x, xs)) for k in range(len(xs))]
```

On obtient :

```
In [8]: xs
Out[8]:
```

$$\left[-1,\ -\frac{1}{3},\ \frac{1}{3},\ 1\right]$$

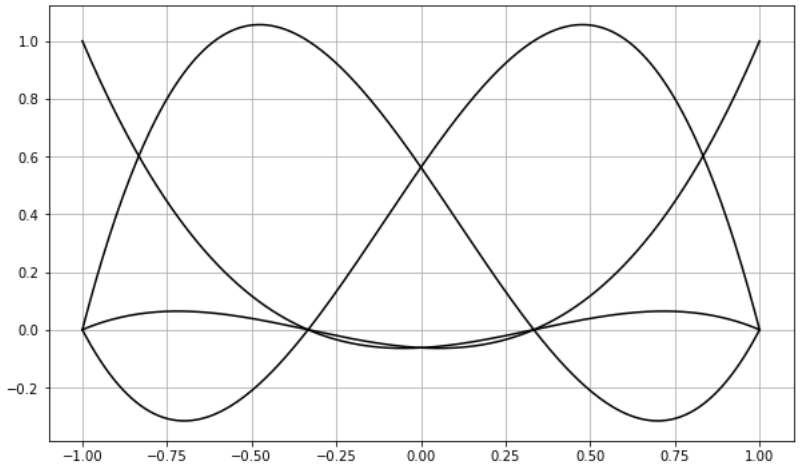
```
In [9]: ps
Out[9]:
```

$$\left[-\frac{9x^3}{16}+\frac{9x^2}{16}+\frac{x}{16}-\frac{1}{16},\ \frac{27x^3}{16}-\frac{9x^2}{16}-\frac{27x}{16}+\frac{9}{16},\ -\frac{27x^3}{16}-\frac{9x^2}{16}+\frac{27x}{16}+\frac{9}{16},\ \frac{9x^3}{16}+\frac{9x^2}{16}-\frac{x}{16}-\frac{1}{16}\right]$$

La fonction ***tracer_poly*** qui suit trace la courbe de p pour x entre xmin et xmax.
Les paramètres optionnels de la fonction sont passés à *plt.plot*.

```
def tracer_poly(p, xmin, xmax, *opt1, **opt2):
    xs = subdi(xmin, xmax, 300)
    ys = [p.subs(x, t) for t in xs]
    plt.plot(xs, ys, *opt1, **opt2)
```

Voici le graphique obtenu :



Exercice 4.1. *Parmi les quatre graphiques ci-dessus, retrouver les polynômes d'interpolation de Lagrange correspondants.*

5 Le polynôme d'interpolation

Proposition 5.1. *Il existe un unique polynôme P de degré inférieur ou égal à n tel que $P(x_k) = f(x_k)$ pour $k = 0, \ldots, n$. Le polynôme P est donné par*

$$P = \sum_{k=0}^{n} f(x_k) L_k$$

où les L_k sont les polynômes élémentaires de Lagrange.

Preuve :
Soit P le polynôme défini ci-dessus. Tout d'abord, P est de degré inférieur ou égal à n.
De plus, pour $i = 0, 1, \ldots, n$:

$$P(x_i) = \sum_{k=0}^{n} f(x_k) L_k(x_i) = \sum_{k=0}^{n} f(x_k) \delta_{ki} = f(x_i)$$

Ce polynôme est donc solution du problème. Par ailleurs, supposons que P et Q sont solutions.
Le polynôme $P - Q$ est de degré inférieur ou égal à n et s'annule en les $n+1$ points $x_0, x_1, \ldots, x_n$. C'est donc le polynôme nul.
D'où l'unicité.

La fonction ***interpoler(f,xs)*** prend en paramètres une fonction f et une liste xs de points.
Elle renvoie le polynôme d'interpolation de f aux points de la liste xs .

```
def interpoler(f,xs):
    ps=[lagrange(k, x, xs) for k in range(len(xs))]
    p=0
    n = len(xs)
    for k in range(n):
        p = p + f(xs[k]) * ps[k]
    return p
```

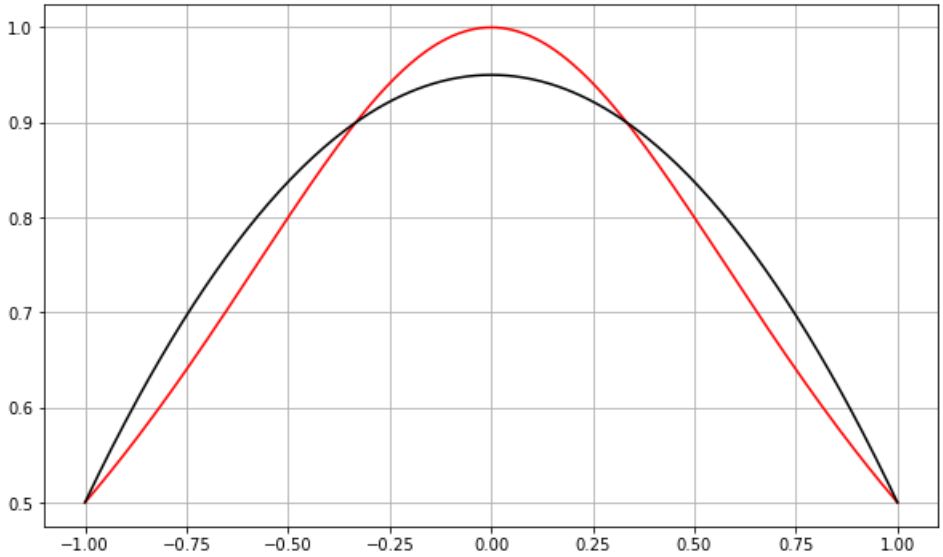
Testons sur un petit exemple : prenons $f : x \mapsto \frac{1}{x^2+1}$ et interpolons sur une subdivision de $[-1, 1]$.
Nous allons fréquemment utiliser cette fonction dans toute la suite, alors appelons-la *exemple*.

```
def exemple(x):
    return 1 / (x ** 2 + 1)

p = expand(interpoler(exemple, subdi(-1, 1, S(3))))
xs = subdi(-1, 1, 300)
ys = [exemple(t) for t in xs]
plt.plot(xs, ys, 'r')
tracer_poly(p, -1, 1, 'k')
```

```
plt.grid()
plt.show()
```

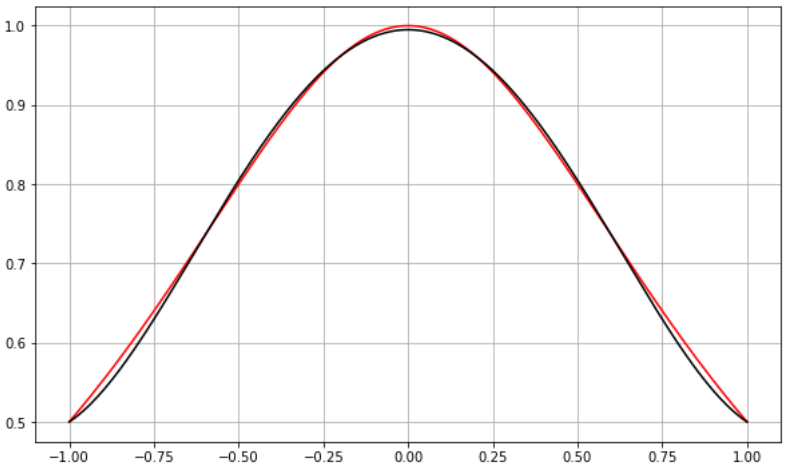
On obtient :



Que se passe-t-il si on augmente le nombre de points de la subdivision ?

```
p = expand(interpoler(exemple, subdi(-1, 1, S(5))))
xs = subdi(-1, 1, 300)
ys = [exemple(t) for t in xs]
plt.plot(xs, ys, 'r')
tracer_poly(p, -1, 1, 'k')
plt.grid()
plt.show()
```

On obtient :



6 Estimation de l'erreur dans l'interpolation de Lagrange

Avant de donner une estimation de l'erreur, nous allons démontrer la propriété suivant.

Proposition 6.1. *Soit $f : [a,b] \longmapsto \mathbb{R}$ dérivable sur $[a,b]$ alors, si f possède au moins $n+2$ zéros distincts sur $[a,b]$, f' possède au moins $n+1$ zéros distincts sur $[a,b]$.*

Preuve :
il suffit d'appliquer le théorème de Rolle entre deux zéros consécutifs de f.

Corollaire 6.2. *Soit $f \in \mathscr{C}^{n+1}([a,b])$. Si f possède au moins $n+2$ zéros distincts sur $[a,b]$, alors $f^{(n+1)}$ a au moins un zéro sur $[a,b]$.*

Preuve : il suffit de faire une récurrence en appliquant la propriété précédente.

Soit f une fonction réelle définie sur un intervalle $[a,b]$ et soient :

$$a \leq x_0 < \ldots < x_n \leq b,$$

$n+1$ points de $[a,b]$.
On note P le polynôme d'interpolation de Lagrange de f aux points $x_0, \ldots, x_n$.

Théorème 6.3. *On suppose $f \in \mathscr{C}^{n+1}([a,b])$, alors :*

$$\forall x \in [a,b], \exists \xi \in [a,b], f(x) - P(x) = \frac{(x-x_0)(x-x_1)\ldots(x-x_n)}{(n+1)!} f^{(n+1)}(\xi)$$

Preuve :
si $x = x_i$, alors la relation est vérifiée.

Soit $x \in [a,b]$ fixé, x différent de tous les x_i.
Posons $q(x) = (x-x_0)(x-x_1)\ldots(x-x_n)$ et :

$$W(t) = f(t) - P(t) - \frac{q(t)}{q(x)}(f(x) - P(x)).$$

La fonction W est de classe $\mathscr{C}^{n+1}$ comme f et s'annule pour $t = x, x_0, x_1, \ldots, x_n$; elle admet donc au moins $n+2$ zéros.
D'après le corollaire 8 :

$$\exists \xi \in [a,b], W^{(n+1)}(\xi) = 0.$$

On en déduit la relation.

Le point ξ étant inconnu, on cherche une majoration et on a le corollaire immédiat :

Corollaire 6.4. *Si $f^{(n+1)}$ est continue sur $[a,b]$, alors :*

$$\forall x \in [a,b], |f(x) - P(x)| \leq \frac{|(x-x_0)(x-x_1)\dots(x-x_n)|}{(n+1)!} \sup_{x\in[a,b]} \left|f^{(n+1)}(x)\right|$$

Autrement dit, pour tout $x \in [a,b]$:

$$|f(x) - P(x)| \leq \frac{M_{n+1}}{(n+1)!}|Q(x)|$$

si :

- M_{n+1} est un majorant de $f^{(n+1)}$ sur $[a;b]$
- $Q(x) = (x-x_0)(x-x_1)\dots(x-x_n)$

Remarque 6.5. *La majoration que nous venons d'obtenir est intéressante :*

- *le facteur $(n+1)$! au dénominateur est prometteur, puisqu'il tend très vite vers $+\infty$ lorsque n augmente.*
- *le facteur M_{n+1} ne dépend que de f et pas des x_i.*
- *le facteur $|Q(x)|$ ne dépend que des x_i et de x, et pas de f. De plus :*

$$\forall x \in [a,b], |Q(x)| \leq (b-a)^{n+1}$$

Une autre majoration de l'erreur d'approximation devient :

$$\forall x \in [a,b], |f(x) - P(x)| \leq \frac{(b-a)^{n+1}}{(n+1)!} M_{n+1}$$

Exemple 6.6. *Si $f(x) = cos(x)$ ou $f(x) = sin(x)$ alors $M_{n+1} \leq 1$ et :*

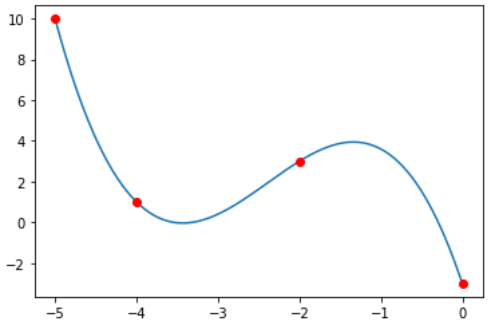
$$\forall x \in [a,b], |f(x) - P(x)| \leq \frac{(b-a)^{n+1}}{(n+1)!}$$

7 Application

Tracer la courbe d'une fonction polynomiale passant par les points de coordonnées :

$$(-5;10), (-2,3), (-4,1) \text{ et } (0,-3).$$

On doit obtenir :


10
8
6
4
2
0
−2
−5
−4
−3
−2
−1
0