Interpolation dans la base de Newton

Introduction :

$$\begin{cases} TP_1 : \text{base canonique } (1, x, \dots, x^n),\ V \text{ vandermonde} \\ \quad \hookrightarrow \text{Résolution système} \\ TP_2 : \text{Pas de système : Écrire la base de Lagrange : } L_k(x) = \prod_{i \neq k} \dfrac{(x - x_i)}{x_k - x_i}. \end{cases}$$

Inconvénient Majeur : En cas de données nouvelles, il faut recalculer $\begin{cases} V + \text{Résolution du syst (TP1)} \\ L_k \text{ Lagrange (TP2).} \end{cases}$

Approche adoptée ici (TP3) : Les anciennes données me permettent d'avoir un polynôme de travail, qui sera mis-à-jour au fur & à mesure de l'arrivée de nouvelles données : Il nous faut une base "qui va grandir" avec les données.

Soit

x_0	x_1	…	x_n
y_0	y_1	…	y_n

grille de pts

Rappel : On cherche P de degré $= n$ tq $P(x_i) = y_i$.

Def base de Newton : $N_0, N_1, \dots, N_n$

$$\begin{cases} N_0(x) = 1 \\ N_{k+1}(x) = N_k(x) \times (x - x_k) \end{cases}$$

De manière explicite

$$\begin{cases} N_0(x) = 1 & N_1(x) = x - x_0 \\ N_2(x) = (x-x_0)(x-x_1) \\ \vdots \\ N_n(x) = (x-x_0)(x-x_1) \times \cdots \times (x-x_{n-1}) \end{cases}$$

Remarque : x_n ne figure pas ds l'écriture de la base de Newton

Cherchons $P(x)$ sous la forme :

$$P(x) = \sum_{i=0}^{n} a_i N_i(x) = a_0 N_0(x) + a_1 N_1(x) + \dots + a_n N_n(x)$$

(1) Il suffit de trouver les a_i.

(2) Comment procéder quand j'ajoute un point.

Calcul des a_i :

. Traduire $P(x_i) = y_i$ pour $0 \leq i \leq n$.

. $P(x_0) = y_0$ et $P(x_0) = a_0$ car x_0 est racine de $N_1, N_2, \dots, N_n$.

$$\boxed{a_0 = y_0}$$

. $P(x_1) = y_1$, x_1 est racine de $N_2, N_3 \dots N_n$

$$\boxed{a_0 + a_1(x_1 - x_0) = y_1} \quad \text{donc} \quad \boxed{a_1 = \frac{y_1 - y_0}{x_1 - x_0}}$$

• $P(x_2) = y_2$.

$$a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) + 0 = y_2 .$$

vient du fait que x_2 est racine de $N_3, N_4 \dots N_n$

$a_0 = y_0$

$$a_2 = \frac{y_2 - a_0 - a_1(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)}$$

$$= \frac{y_2 - y_0}{(x_2 - x_0)(x_2 - x_1)} - a_1 \frac{(x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)}$$

$$= \frac{\frac{y_2 - y_0}{x_2 - x_0}}{x_2 - x_1} - \frac{\frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_1}$$

$D(2,0) = \frac{y_2 - y_0}{x_2 - x_0}$

$$= \frac{D(2,0) - D(1,0)}{x_2 - x_1} = DD(2,1)$$

Organisation des différences divisées :

♡ Classique (Partiel 2020) :

Données

x	-1	1	3	7
y	-4	-2	16	1636

Tab. Calcul

	x	y	0	1	2
0	-1	-4			
1	1	-2	$\frac{-2-(-4)}{1-(-1)} = 1$		
2	3	16	$\frac{16-(-4)}{3-(-1)} = 5$	$\frac{5-1}{3-1} = 2$	
3	7	1636	$\frac{1636-(-4)}{7-(-1)} = 205$	$\frac{205-1}{7-1} = 34$	$\frac{34-2}{7-3} = 8$

Les coeff de P de la base canonique sont entourés de rouge : -4 ; 1 ; 2 et 8

$N_0(x) = 1$ $\qquad N_1(x) = (x-(-1)) = x+1$

$N_2(x) = (x+1)(x-1)$ et $N_3(x) = (x+1)(x-1)(x-3)$

D'où : $\boxed{P(x) = -4 + 1(x+1) + 2(x+1)(x-1) + 8(x+1)(x-1)(x-3)}$

Au niveau programmation

X, Y données.

* faire une copie de Y

(Python Coeff = Y Pas bon)

Coeff = deepCopy(Y) ; mettre marq = 1

Marq $\to 0$

y_0 c'est a_0.

$\dfrac{\text{Coeff}(k) - \text{Coeff}(0)}{x(k) - x(0)}$

a_0

a_1

$\dfrac{\text{Coeff}(k) - \text{Coeff}(1)}{x_k - x_1}$

Plus Glt :

$0 \leqslant \text{marq} \leqslant \text{len}(X) - 1$

Pour chaque marq :

$\text{marq}+1 \leqslant k \leqslant \text{len}(X)$ $\quad \dfrac{\text{Coeff}(k) - \text{Coeff}(\text{marq})}{x(k) - x(\text{marq})}$

On vient de récupérer tous les coeff de P dans la base de Newton.

Il n'y a plus qu'à écrire le Poly P

$$P(x) = \sum_{i=0}^{n} a_i N_i(x).$$

---

② Ajouter un pt $A(x_A, y_A)$

reprise du tableau précédent.

x	y				
-1	-4				
1	-2	1			
3	16	5	2		
7	1636	205	34	8	
10	11986	*	. .	. .	β

← 1ière méthode

+ stylée : Soit $\tilde{P}$ le nouveau poly, P l'ancien

$P(x)$

$$\tilde{P}(x) = -4 + 1(x+1) + 2(x+1)(x-1) + 8(x+1)(x-1)(x-3)$$
$$+ \beta(x+1)(x-1)(x-3)(x-7)$$

Il suffit de calculer β :

$$\tilde{P}(10) = 11986.$$

Rq : $\tilde{P}(x) = P(x) + \beta(x+1)(x-1)(x-3)(x-7)$

$$11986 = P(10) + \beta(10+1)(10-1)(10-3)(10-7)$$

$$\beta = \frac{\tilde{P}(x_A) - P(x_A)}{\prod_{k=0}^{n}(x_A - x_k)} = \frac{y_A - P(x_A)}{\prod_{k=0}^{n}(x_A - x_k)}$$

$$\boxed{\beta} = \frac{11986 - P(10)}{(10+1)(10-1)(10-3)(10-7)}$$

ajouter x_A

Mise-à-jour $X = [-1; 1; 3; 7; 10]$

y_A

$Y = [-4; -2; 16; 1636; 11986]$

$Coeff = [-4; 1; 2; 8; \beta]$

On dispose de $\tilde{P}$:

$$\tilde{P}(x) = \sum_{k=0}^{n} coeff(k)\, N_k(x)$$

```
def Newton(Coeff; x):
    N=1
    s=Coef[0]
    for k in range(1, len(X)):
        N*= x - x[k-1]
        S+= Coeff[k] x N
```

---

schéma Gl :

- X, Y : données
- Diff-divisée(X, Y) $\longrightarrow$ Coeff
  (On l'appelle une fois pour toute).
- On appelle Newton (Coeff, X_{att}) $\longrightarrow$ Y_{att}
  retrace.
- Si ajout de point : Ajout(x_A, y_A).
  Créer une procédure

allonger

X ← x_A Y ← y_A Coeff ← β

- Appeler Newton(Coeff, X_{att})
  Je retrace (X_{att}, $Y_{estimé}$).
