Chapitre 4

Interpolation et approximation

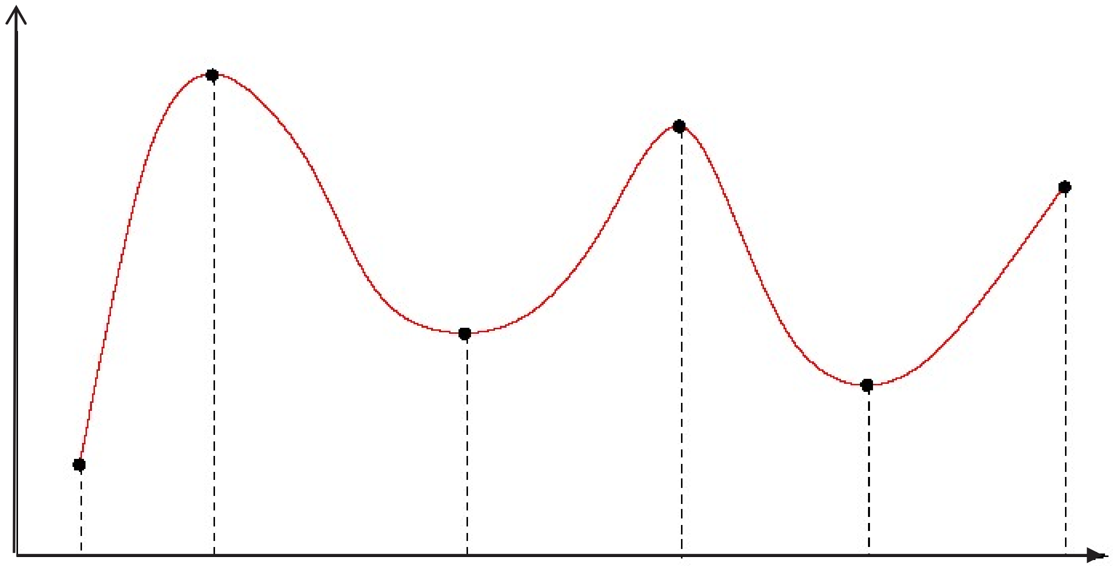
Problème

- Données :
  - un ensemble de points connus (x_i, Y_i) ; ou $Y_i \in R^p$
    - Obtenus par un ensemble de mesures(relevés terrains)
    - ou bien calculé par l'estimation $(x_i, f(x_i))$ d'une fonction f au points x_i
- But : déterminer un "modèle" mathématique pour f
  - réduire f en une expression simple (exemple : polynôme)
  - bonnes propriétés : dérivabilité, etc.
- Dans quels cas ?
  - définir un modèle mathématique à partir d'un nombre discret de mesures
  - analyser un phénomène étudié de manière empirique
  - remplacer une équation de courbe "compliquée" par une fonction polynomiale par exemple.

Interpolation

les x_i, sont des mesures exactes

On veut que la courbe passe par tous les $(x_i, f(x_i))$



On se donne

- une fonction f: R → R inconnue et continue sur un intervalle [a, b].
- un ensemble de points connus (x_i, y_i), *$i \in [0, n]$.*
  - *$\{x_0, x_1, . . . , x_n\}$ est le support de l'interpolation*

On cherche

une fonction φ : R → R telle que

φ (x_i) = f (x_i), $i \in [0, n]$.

- En pratique,φest une somme de fonctions

$$\varphi(x) = \sum_{i=0}^{n} a_i \varphi_i(x)$$

vérifiant

$$f(x_i) = \varphi(x_i) \text{ avec } (x_i) \in R^n \quad (1)$$

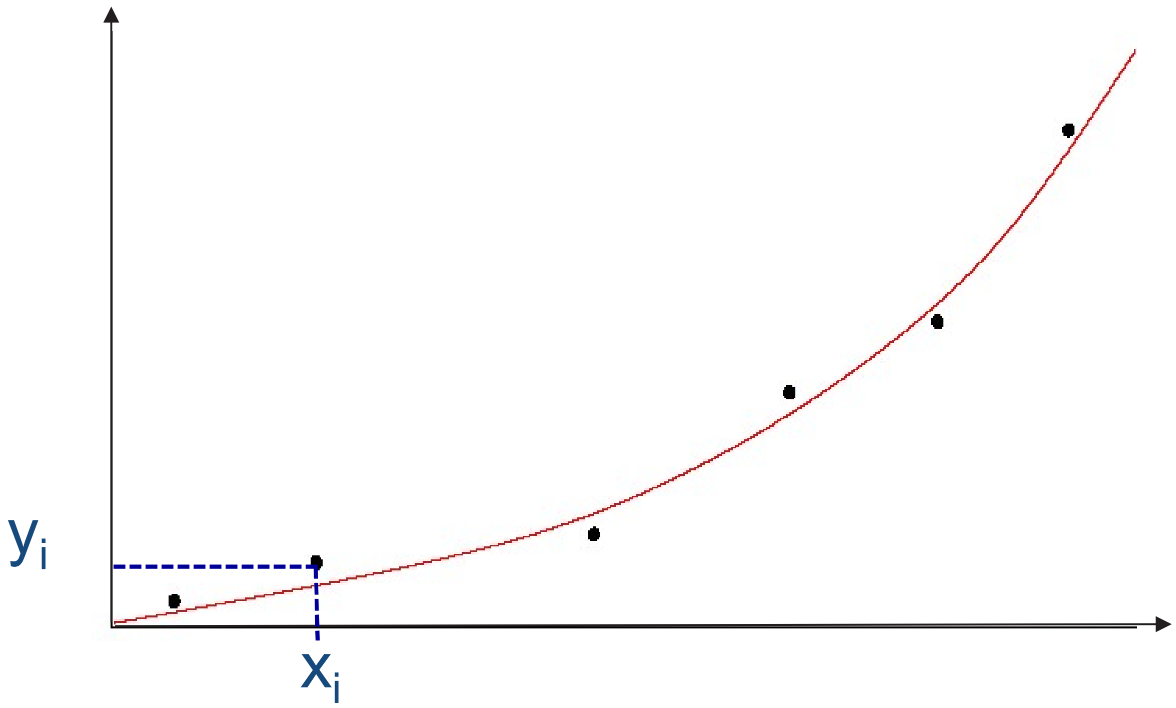
φ_j: fonctions de la base dans laquelle on exprime f ; φ_i doit se prêter aux traitements numériques courants.

Problème : déterminer les a_i pour vérifier (1) et assurer l'unicité de la solution donc de a_i

Approximation

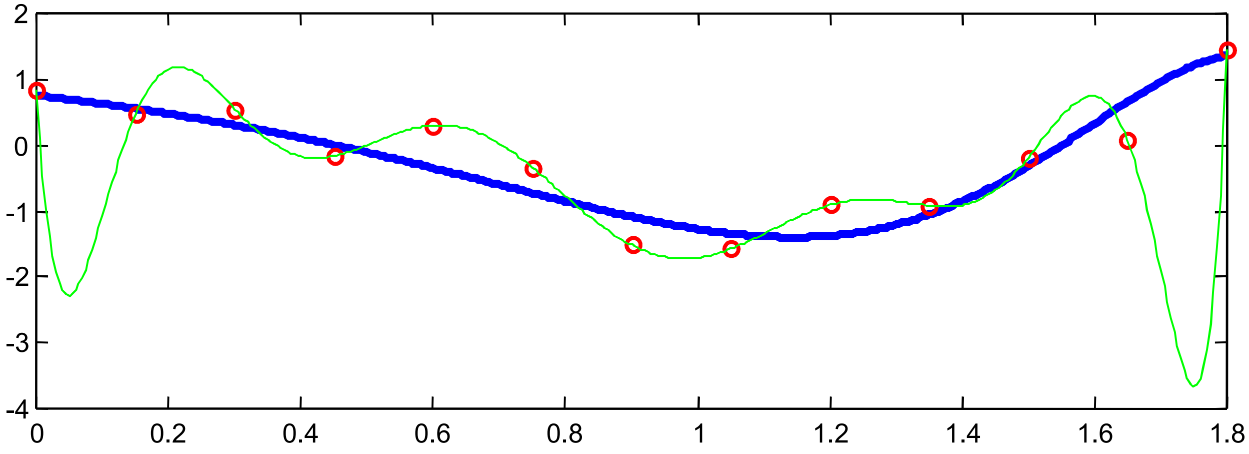
les (x_i, y_i) sont des mesures données

Objet de l'étude : déterminer la courbe s'approchant au mieux des points $(x_i, f(x_i))$



Approximation

- En général, on se restreint à une famille de fonctions connues
  - polynômes,
  - exponentielles, logarithme
  - fonctions trigonométriques…



Quelques méthodes d'interpolation

- Interpolation polynomiale
  - polynômes de degré au plus *n*
    - polynômes de Lagrange
    - différences finies de Newton

- Interpolation par splines
  - polynômes par morceaux

- Interpolation d'Hermite (ce chapitre ne sera pas traité)
  - informations sur les dérivées de la fonction à approcher

Théorème de Weierstrass

soit f fct continue sur $[a, b]$

Alors, $\forall \varepsilon > 0$, il existe un polynôme $P(x)$, défini sur $[a, b]$
tel que :

$$|f(x) - P(x)| < \varepsilon \qquad \forall\, x \in [a, b]$$

plus ε, est petit,

plus l'ordre du polynôme est grand

Interpolation :

$n + 1$ points, $n + 1$ contraintes, $n + 1$ équations,
$n + 1$ inconnues: ordre du polynôme n

Interpolation polynomiale

- Le problème : Solution recherchée
- Données --> $(x_0, y_0 = f(x_0)), \ldots, (x_i, y_i = f(x_i)), \ldots, (x_i, y_i = f(x_i))$
- Solution --> $P(x)$ tel que $P(x_i) = f(x_i), i = 0, n$

- mauvaise solution : résoudre le système linéaire

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

- la combinaison linéaire de polynômes est un polynôme

Interpolation polynomiale

la combinaison linéaire de polynômes est un polynôme

$$(x_0, y_0 = f(x_0)), \ldots, (x_i, y_i = f(x_i)), \ldots, (x_i, y_i = f(x_i))$$

$$P(x) \text{ tel que } P(x_i) = f(x_i), \qquad i = 0, n$$

→ Idée de Lagrange

$$P(x) \; = y_0 P_0(x) + \ldots + y_i P_i(x) + \; y_n P_n(x)$$

tel que $P_i(x_i) = 1$ et $P_i(x_j) = 0 \quad j \neq i$

ainsi $P(x_i) = y_0 P_0(x_i) + \ldots + y_i P_i(x_i) + \; y_n P_n(x_i)$

$$\downarrow \qquad\qquad \downarrow \qquad\qquad \downarrow$$

$$0 \qquad\qquad 1 \qquad\qquad 0$$

Méthode de Lagrange pour l'interpolation polynômiale

➔ Idée changer de base pour les polynômes

$$L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n+1} \frac{(x - x_j)}{(x_i - x_j)}$$

$$L(x) = \sum_{i=0}^{n} y_i L_i$$

L est un polynôme d'ordre n

- Théorème
  - Soient *n+1* points distincts de coordonnée $(x_i\,,\,y_i)$

avec x_i , y_i réels

il existe un unique polynôme $p \in P_n$ tel que $p(x_i) = y_i$ pour $i = 0$ à n

Théorème

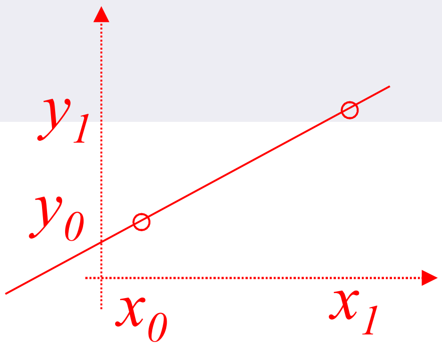
- Soient $n+1$ points distincts x_i réels et $n+1$ réels y_i, il existe un unique polynôme $p \in P_n$ tel que $p(x_i) = y_i$ pour $i = 0$ à n

Idée de démonstration

- **Construction de p :**
  avec L_i polynôme de Lagrange $$p(x) = \sum_{i=0}^{n} y_i L_i(x)$$

- **Propriétés de L_i**
  - ▶ $L_i(x_i) = 1$
  - ▶ $L_i(x_j) = 0 \quad (j \neq i)$

Lagrange : degré 1



- Exemple avec *n=1*
  - on connaît 2 points (x_0,y_0) et (x_1,y_1)
  - on cherche la droite $y=ax+b$ (polynôme de degré 1) qui passe par les 2 points :
    - $y_0 = a\,x_0 + b$
    - $y_1 = a\,x_1 + b$

en résolvant le système

$a = (y_0 - y_1) / (x_0 - x_1)$

$b = (x_0\,y_1 - x_1\,y_0) / (x_0 - x_1)$

en passant par l'expression de Lagrange

$$y = \frac{y_0 - y_1}{x_0 - x_1}x + \frac{x_0 y_1 - x_1 y_0}{x_0 - x_1}$$

$$y = y_0\frac{x - x_1}{x_0 - x_1} - y_1\frac{x - x_0}{x_0 - x_1} = y_0 \underbrace{\boxed{\frac{x - x_1}{x_0 - x_1}}}_{L_0(x)} + y_1 \underbrace{\boxed{\frac{x - x_0}{x_1 - x_0}}}_{L_1(x)}$$

Lagrange : degré 2

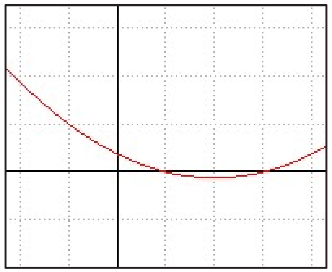
- Exemple avec *n=2*
  - on connaît 3 points (-*1*,*1*), (*1*,*4*) et (*3*,*16*)
  - polynômes de Lagrange associés :

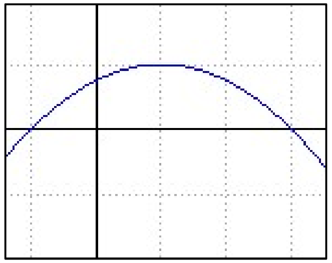
    → Espace vectoriel : avec $\{L_i\}$ base de l'interpolation

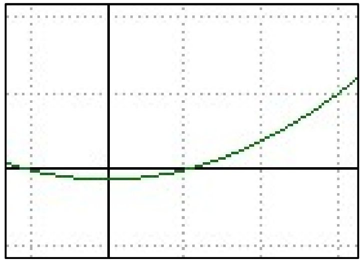
$$L_0(x) = \frac{(x-1)(x-3)}{8}$$

$$L_1(x) = \frac{(x+1)(x-3)}{-4}$$

$$L_2(x) = \frac{(x+1)(x-1)}{8}$$







Lagrange : degré 2

- calcul du polynôme d'interpolation

points : (-1,1), (1,4) et (3,16)

- ▶ $p(x) = l_0(x) + 4l_1(x) + 16l_2(x)$
- ▶ $p(x) = \frac{(x-1)(x-3)}{(-1-1)(-1-3)} + 4\,\frac{(x+1)(x-3)}{(1-(-1))(1-3)} + 16\,\frac{(x+1)(x-1)}{(3+1)(3-1)}$
- ▶ en développant, on trouve $\mathrm{p(x)} = \frac{9}{8}x^2 + \frac{3}{2}x + \frac{11}{8}$

Lagrange : Algorithme

Fonction y = Lagrange (x, x_i, y_i)

pour $i = 1$ à n

pour $j = 1$ à $n, j \neq i$

$$l \leftarrow l * \frac{x - x_i(j)}{x_i(i) - x_i(j)}$$

fin pour

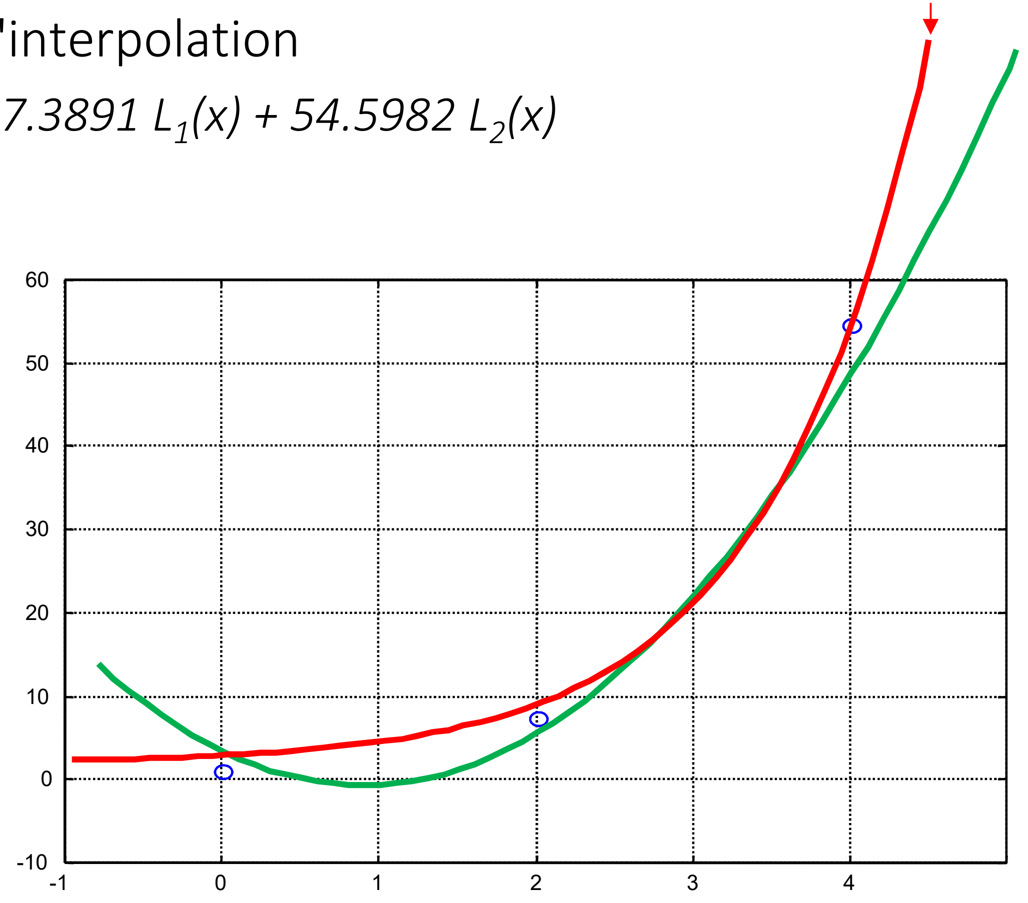
$y \leftarrow y + y_i * l$

fin pour

Donner la complexité de l'algorithme !

Lagrange : exemple n°3

- Exemple avec $n=2$ (fonction à approcher $y=e^x$)
  - on connaît 3 points $(0,1)$, $(2,7.3891)$ et $(4,54.5982)$
  - Polynôme d'interpolation
    - $p(x) = L_0(x) + 7.3891\, L_1(x) + 54.5982\, L_2(x)$



Lagrange : estimation de l'erreur d'interpolation

- Erreur d'interpolation $e(x) = \|f(x) - p(x)\|$

❑ Théorème :

- si *f* est *n+1* dérivable sur [*a*,*b*], ∀ *x* ∈ [*a*,*b*], notons :
  - *I* le plus petit intervalle fermé contenant *x* et les x_i
  - $\phi(x)=(x-x_0)(x-x_1)\dots(x-x_n)$
- alors, il existe ξ ∈ I tel que

$$\boxed{e(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\varphi(x)}$$

- NB : ξ est dans le voisinage de x
- Utilité = on contrôle l'erreur d'interpolation donc la qualité de l'interpolation (voir exercice fait en TD)

Lagrange : choix de n

- Supposons que l'on possède un nb élevé de points pour approcher *f* ... faut-il tous les utiliser ?
  - (calculs lourds)

- Méthode de Neville :
  - on augmente progressivement n
  - on calcule des L_i de manière récursive
  - on arrête dès que l'erreur est inférieure à un seuil

    (d'où l'utilité du calcul de l'erreur)

La méthode de Neville

Méthode récursive du calcul de la valeur du polynôme d'interpolation en un point donné, il est aisé d'ajouter des points d'interpolation au fur et à mesure.

$$p_{i,0}(x) = y_i, \qquad 0 \leq i \leq n,$$

$$p_{i,j+1}(x) = \frac{(x_i - x)p_{i+1,j}(x) + (x - x_{i+j+1})p_{i,j}(x)}{x_i - x_{i+j+1}}, \quad 0 \leq j < n \text{ et } 0 \leq i < n-j.$$

Algorithme de Neville-Aitken

- Application

$$
\begin{array}{ccccc}
p_{0,0}(x) = y_0 & & & & \\
 & p_{0,1}(x) & & & \\
p_{1,0}(x) = y_1 & & p_{0,2}(x) & & \\
 & p_{1,1}(x) & & p_{0,3}(x) & \\
p_{2,0}(x) = y_2 & & p_{1,2}(x) & & \boxed{p_{0,4}(x)} \\
 & p_{2,1}(x) & & p_{1,3}(x) & \\
p_{3,0}(x) = y_3 & & p_{2,2}(x) & & \\
 & p_{3,1}(x) & & & \\
p_{4,0}(x) = y_4 & & & &
\end{array}
$$

x_i	1	2	3	5
y_i	1	4	2	5

$i \backslash j$	0	1	2	3
0	1	$3x-2$	$-\frac{5}{2}x^2+\frac{21}{2}x-7$	$\frac{11}{12}x^3-8x^2+\frac{247}{12}x-\frac{25}{2}$
1	4	$-2x+8$	$\frac{7}{6}x^2-\frac{47}{6}x+15$	
2	2	$\frac{3}{2}x-\frac{5}{2}$		
3	5			

L'algorithme de Neville

Fonction y = Neville(x,x_i,y_i)

pour $i = 1$ à n

$Q(i,0) \leftarrow y_i(i)$

fin pour

pour $i = 1$ à n

pour $j = 1$ à i

$$Q(i,j) \leftarrow \frac{\left(x - x_i(i-j)\right)Q(i,j-1) - \left(x - x_i(i)\right)Q(i-1,j-1)}{x_i(i) - x_i(i-j)}$$

fin pour

$y \leftarrow Q(n,n)$

fin pour

Vérifier : complexité du calcul : n^2

Méthode de Newton pour l'interpolation polynomiale :

Polynômes de Newton :

- base = $\{1, (x-x_0), (x-x_0)(x-x_1), \ldots, (x-x_0)(x-x_1)\ldots(x-x_{n-1})\}$

- on peut ré-écrire $p(x)$:

$$p(x)=a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1)+\ldots+ a_n(x-x_0)(x-x_1)\ldots(x-x_{n-1})$$

- calcul des a_k : méthode des différences divisées

Newton : différences divisées

- **Définition :**

Soit une fonction f dont on connaît les valeurs en des points distincts a, b, c, …

On appelle différence divisée d'ordre $0, 1, 2, \dots, n$ les expressions définies par récurrence sur l'ordre k :

- ✓ $k=0 \quad f[a] = f(a)$
- ✓ $k=1 \quad f[a,b] = (f[b] - f[a]) / (b - a)$
- ✓ $k=2 \quad f[a,b,c] = (f[a,c] - f[a,b]) / (c - b)$

…

- ✓ $f[X,a,b] = (f[X,b] - f[X,a]) / (b - a)$
  $a \notin X, \; b \notin X, \; a \neq b$

Newton : différences divisées

- Détermination des coefficients de *p(x)* dans la base de Newton :

Théorèmes

calcul des coeficients de newton

$$a_k = f\,[x_0, x_1, \ldots, x_k] \qquad \text{avec } k = 0 \ldots n$$

Calcul de l'erreur d'interpolation

$$e(x) = f\,[x_0, x_1, \ldots, x_n, x]\ \phi(x)$$

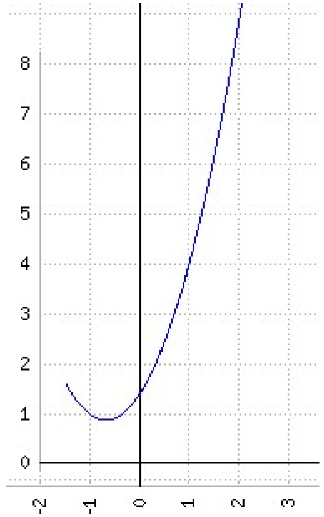
- Calcul pratique des coefficients :

$$a_1 = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$$

	$\boldsymbol{a_0}$			
x_0	$f[x_0]$	$\boldsymbol{a_1}$		
x_1	$f[x_1]$	$f[x_0, x_1]$	$\boldsymbol{a_2}$	
x_2	$f[x_2]$	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$	
…	…	…	…	
…	…	…	$f[x_{n-3}, x_{n-2}, x_{n-1}]$	$\boldsymbol{a_n}$
x_n	$f[x_n]$	$f[x_{n-1}, x_n]$ ….	$f[x_{n-2}, x_{n-1}, x_n]$	$f[x_0, \ldots, x_n]$

Newton : exemple

- Retour sur l'exercice : *n=2* avec (*-1,1*), (*1,4*) et (*3,16*)



	a_0		
0	$f\,[x_0]=1$		
2	$f\,[x_1]=5$	$f\,[x_0, x_1]$ a_1 $=(1-4)/(1+1)=3/2$	
4	$f\,[x_2]=17$	$f\,[x_1, x_2]$ $=(16-4)/(3-1)=6$	$f\,[x_0, x_1, x_2]$ a_2 $= (6-3/2)/(3+1)=9/8$

$(\mathrm{p}(\mathrm{x}) = 1 + \frac{3}{2}(x+1) + \frac{9}{8}(x+1)(x-1)$

et on retombe sur $\mathrm{p}(\mathrm{x}) = \frac{9}{8}x^2 + \frac{3}{2}x + \frac{11}{8}$

Newton : l'algorithme

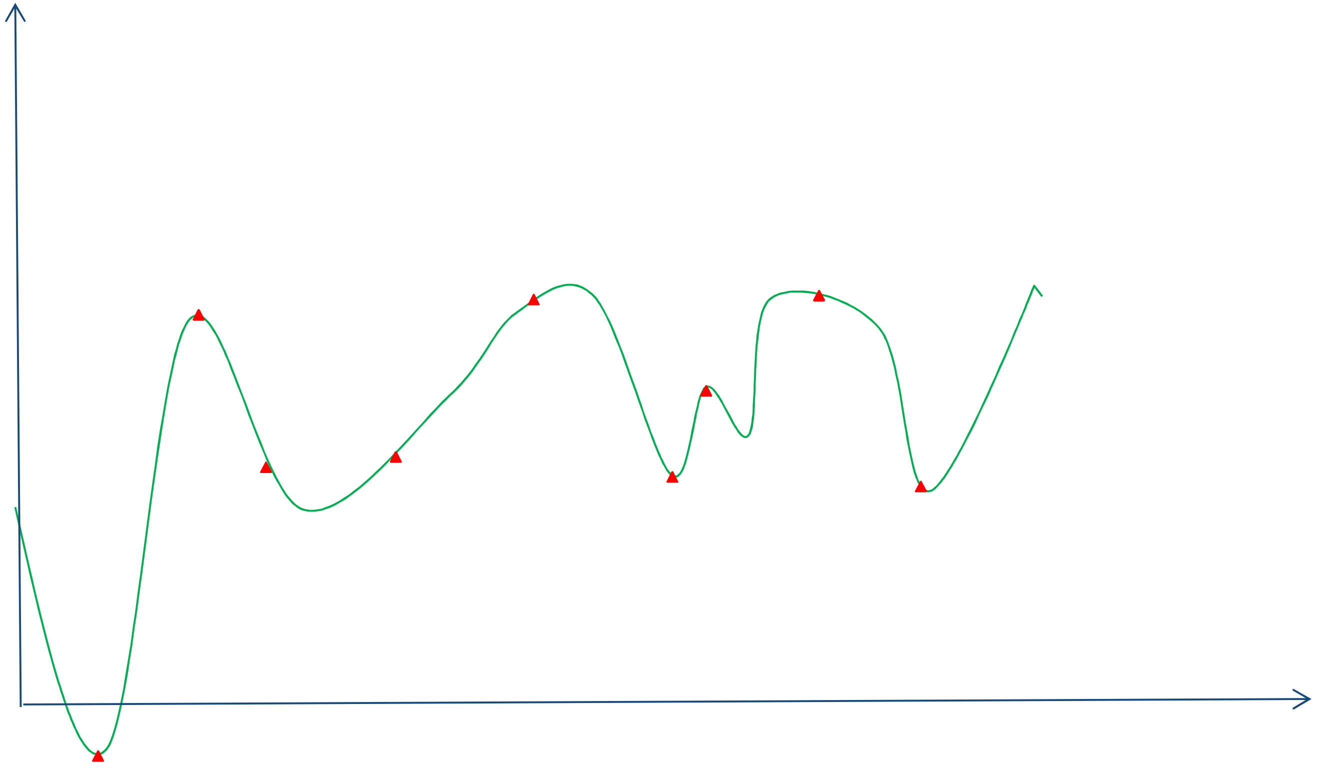
Fonction $a = \text{Newton}(x_i, y_i)$

$$
\begin{array}{l}
\text{pour } i = 1 \text{ jusqu'à } n \\
\qquad F(i,0) \leftarrow y_i(i) \\
\text{fait} \\
\text{pour } i = 1 \text{ jusqu'à } n \\
\qquad \text{pour } j = 1 \text{ jusqu'à } i \\
\qquad\qquad F(i,j) \leftarrow \dfrac{F(i,j-1) - F(i-1,j-1)}{x_i(i) - x_i(i-j)} \\
\qquad \text{fait} \\
\text{fait} \\
\text{pour } i = 1 \text{ jusqu'à } n \\
\qquad a(\text{i}) \leftarrow F(n,i) \\
\text{fait}
\end{array}
$$

Vérifier que la complexité est de : n^2

Si le nombre de points est élevé

- entre les points, le polynôme fait ce qu'il veut !!!
  et plus son degré est élevé plus il est susceptible d'osciller !
- en dehors de l'intervalle des points d'interpolation la fonction tend vers ($\pm\infty$)



Interpolation par splines cubiques

Principe :

- on approche la courbe par morceaux (localement)
- on prend des polynômes de degré faible (3) pour éviter les oscillations

Comment

– on décompose l’espace de définition (des points) en un ensemble contigu d’intervalles sur lesquels on applique des interpolations polynômiales de degré 3

➔ Résultat un ensemble de polynômes définis de façon continue et « lisse »

Splines cubiques : définition

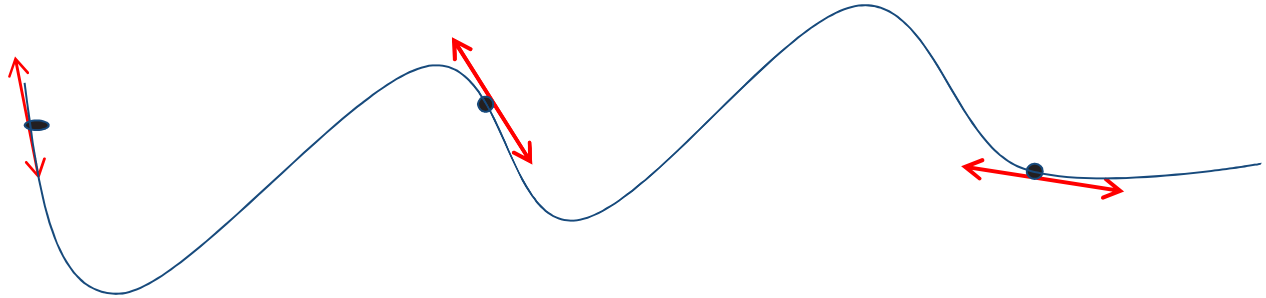
- Définition :
  - On appelle spline cubique (d'interpolation) une fonction notée g, qui vérifie les propriétés suivantes :
    - $g \in C^2[a; b]$ (g est deux fois continûment dérivable),
    - g coïncide sur chaque intervalle $[x_i;\ x_{i\ +\ 1}]$ avec un polynôme de degré inférieur ou égal à 3,
    - $g(x_i)\ =\ y_i\ pour\ i\ =\ 0\ \ldots\ n$

Splines cubiques : définition

- Remarque (voir plus loin):
  - Il faut des conditions supplémentaires pour définir la spline d'interpolation de façon unique
  - Ex. de conditions supplémentaires : conditions aux limites
    - ► $g''(a) = g''(b) = 0$ spline naturelle.

- Remarques :
  - Sur le plan pratique, ces conditions permettent d'avoir une courbe continue et d'aspect lisse ;
  - Forme ≡ forme d'une barre souple soumise à des contraintes physiques
  - Sur le plan technique, cela permet de poser les équations qui permettent d'obtenir l'expression mathématique de la fonction.

Splines : illustration



Fonction formée par 2 morceaux de polynôme

$$P_1(x)=\alpha_1 x^3+\beta_1 x^2+\chi_1 x+\delta_1$$

$$= a_1\,(x\text{-}x_1)^3+b_1\,(x\text{-}x_1)^2+c_1\,(x\text{-}x_1)+d_1$$

$$P_2(x)=a_2\,(x\text{-}x_2)^3+b_2\,(x\text{-}x_2)^2+c_2\,(x\text{-}x_2)+d_2$$

Splines cubiques : équations

- Déterminer la spline d'interpolation
  - g coïncide sur chaque intervalle $[x_i; x_{i+1}]$ avec un polynôme de degré inférieur ou égal à 3
  - *g'' est de degré 1 et est déterminé par 2 valeurs:*
    - *mi = g"(xi) et mi+1 = g"(xi+1) (moment au noeud n°i)*
  - *Notations :*
    - $h_i = x_{i+1} - x_i$ pour $i = 0 \ldots n-1$
    - $\delta_i = [x_i; x_{i+1}]$
    - $g_i(x)$ le polynôme de degré 3 qui coïncide avec g sur l'intervalle δ_i

Splines cubiques : équations

$g''_i(x)$ est linéaire : on peut l'estimer par la méthode de Lagrange

- $\forall\, x \in \delta_i$

$$g_i''(x) = m_{i+1}\frac{x - x_i}{h_i} + m_i\frac{x_{i+1} - x}{h_i}$$

- Pour $g_i'(x)$ on intègre : $g_i''(x)$

$$g_i'(x) = m_{i+1}\frac{(x - x_i)^2}{2h_i} - m_i\frac{(x_{i+1} - x)^2}{2h_i} + a_i$$

(a_i constante d'intégration)

Splines cubiques : équations

- Pour calculer $g_i(x)$, on intègre $g'_i(x)$

$$g_i(x) = m_{i+1}\frac{(x-x_i)^3}{6h_i} + m_i\frac{(x_{i+1}-x)^3}{6h_i} + a_i(x-x_i) + b_i$$

(b_i constante d'intégration)

- $g_i(x_i) = y_i \longrightarrow y_i = \frac{m_i {h_i}^2}{6} + b_i$ 1

- $gi(xi+1) = yi+1 \longrightarrow y_{i+1} = \frac{m_{i+1} {h_i}^2}{6} + a_i h_i + b_i$ 2

Splines cubiques : équations

- g'(x) est continue *:* $g'_i(x_i) = -m_i \frac{h_i}{2} + a_i = m_i \frac{h_{i-1}}{2} + a_{i-1} = g'_{i-1}(x_i)$ 3
- 1 *et* 2 $a_i = \frac{1}{h_i}(y_{i+1} - y_i) - \frac{h_i}{6}(m_{i+1} - m_i)$
- *on remplace les ai dans :* 3

$$h_{i-1}m_{i-1} + 2(h_i + h_{i-1})m_i + h_i m_{i+1} = 6\left(\frac{1}{h_i}(y_{i+1} - y_i) - \frac{1}{h_{i-1}}(y_i - y_{i-1})\right)$$ 4

Splines cubiques : équations

- Rappel : on cherche les m_i (n+1 inconnues)
  - on a seulement (n-1) équations grâce aux données

  - Pour obtenir une solution unique il manques 2 équations :
  - il faut rajouter 2 conditions → par exemple condition aux limites
    - $m_0 = m_n = 0$ *(appelée : spline naturelle)*

Splines cubiques : calcul des coefficients

4

$$h_{i-1}m_{i-1} + 2(h_i + h_{i-1})m_i + h_i m_{i+1} = 6\left(\frac{1}{h_i}(y_{i+1} - y_i) - \frac{1}{h_{i-1}}(y_i - y_{i-1})\right)$$

- Ex de résolution avec $h_i = x_{i+1} - x_i$ (h_i constant) :
  - $$m_{i-1} + 4m_i + m_{i+1} = \frac{1}{h^2}(y_{i-1} - 2y_i + y_{i+1}) = f_i$$
  - Forme matricielle

$$T * M = f = \begin{pmatrix} 4 & 1 & & & 0 \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{pmatrix} \begin{pmatrix} m_1 \\ \cdots \\ m_{n-1} \end{pmatrix} = \begin{pmatrix} f_1 \\ \cdots \\ f_{n-1} \end{pmatrix}$$

  - T matrice inversible, tridiagonale, à diagonale strictement dominante, système facile à résoudre.

Splines cubiques : algorithme

pour $i = 2; n-1$

$T(i,i) \leftarrow 2(h_i + h_{i-1})$

$T(i,i-1) \leftarrow h_{i-1}$

$T(i,i+1) \leftarrow 2h_i$

$$f(i-1) \leftarrow 6\left(\frac{y_{i+1}-y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}\right)$$

fin pour

$T \leftarrow T(2:n-1, 2:n-1)$

$m \leftarrow T/f$

$m \leftarrow [0, m, 0]$

pour $i = 1; n-1$

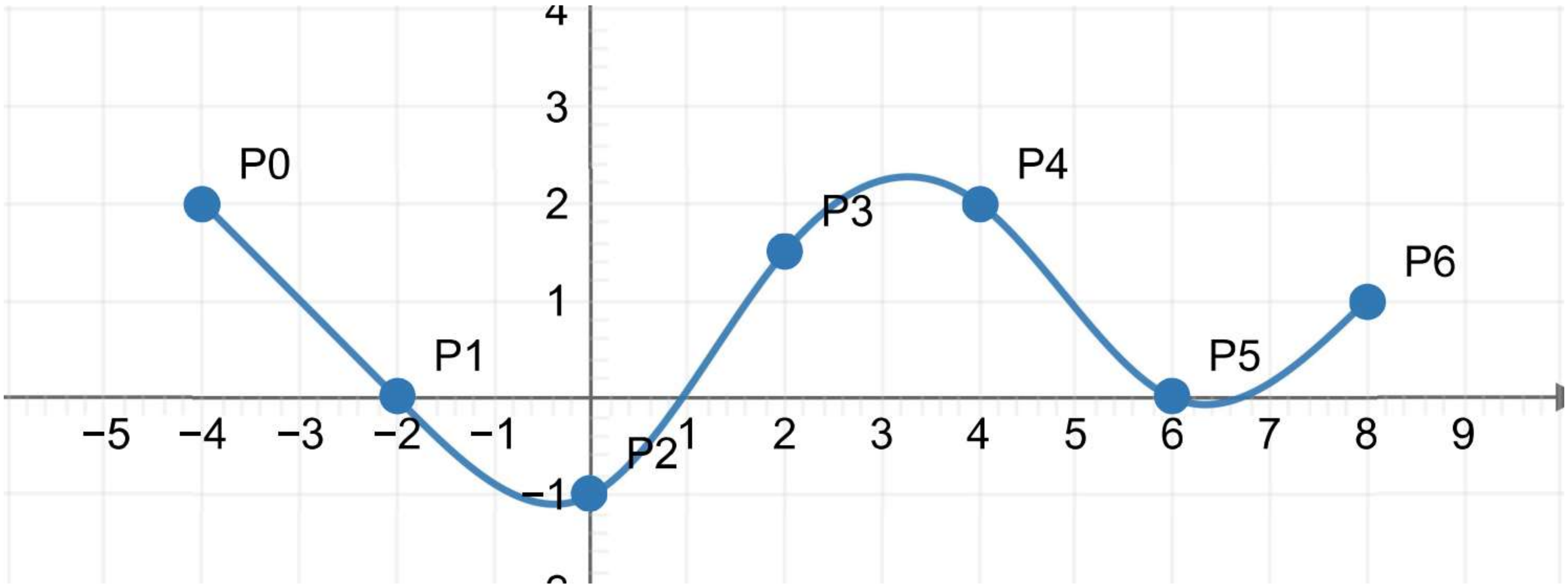
$$a(i) \leftarrow \frac{1}{h_i}(y_{i+1} - y_i) - \frac{h_i}{6}(m_{i+1} - m_i)$$

$$b(i) \leftarrow y(i) - \frac{m_i h_i}{6}$$

fin pour

Splines cubiques : exemple

- Ex : avec 7 points → spline cubique d'interpolation



Splines cubiques : exemple

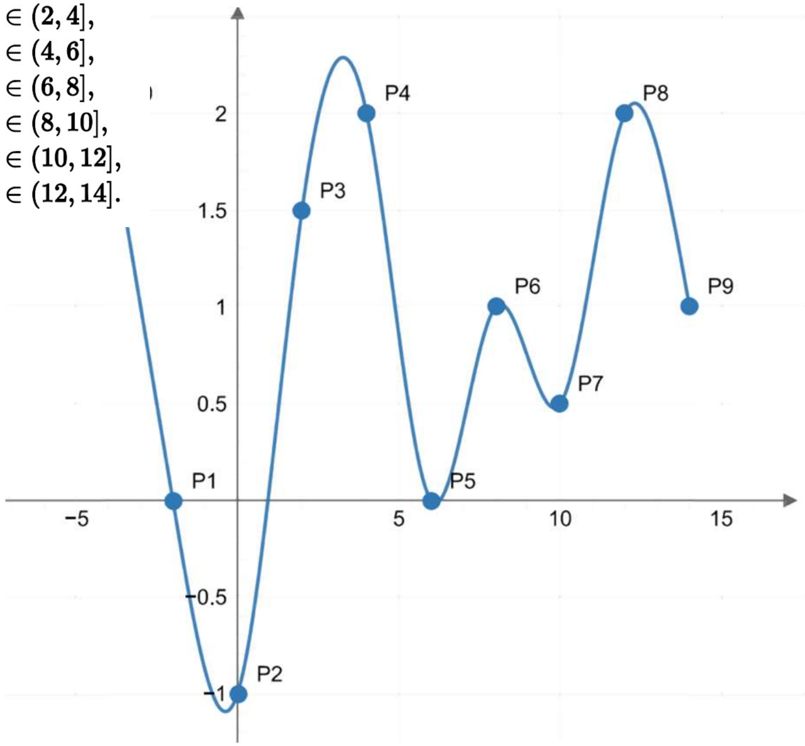
- Ex : avec 9 points → spline cubique d'interpolation

Points

$P_0(-4|2); P_1(-2|0); P_2(0|-1); P_3(2|1.5); P_4(4|2); P_5(6|0); P_6(8|1); P_7(10|0.5); P_8(12|2); P_9(14|1)$

Equation

$$f(x) = \begin{cases} -4.9636\cdot10^{-4}\cdot x^3 + -5.9563\cdot10^{-3}\cdot x^2 + -1.0218\cdot x + -2.0238, & \text{if } x \in [-4,-2],\\ 1.2748\cdot10^{-1}\cdot x^3 + 7.6191\cdot10^{-1}\cdot x^2 + 5.1390\cdot10^{-1}\cdot x + -1.0000, & \text{if } x \in (-2,0],\\ -1.9693\cdot10^{-1}\cdot x^3 + 7.6191\cdot10^{-1}\cdot x^2 + 5.1390\cdot10^{-1}\cdot x + -1.0000, & \text{if } x \in (0,2],\\ -2.7258\cdot10^{-2}\cdot x^3 + -2.5612\cdot10^{-1}\cdot x^2 + 2.5500\cdot x + -2.3574, & \text{if } x \in (2,4],\\ 2.4346\cdot10^{-1}\cdot x^3 + -3.5048\cdot x^2 + 1.5545\cdot10^{1}\cdot x + -1.9684\cdot10^{1}, & \text{if } x \in (4,6],\\ -2.5910\cdot10^{-1}\cdot x^3 + 5.5413\cdot x^2 + -3.8732\cdot10^{1}\cdot x + 8.8870\cdot10^{1}, & \text{if } x \in (6,8],\\ 2.3043\cdot10^{-1}\cdot x^3 + -6.2073\cdot x^2 + 5.5257\cdot10^{1}\cdot x + -1.6177\cdot10^{2}, & \text{if } x \in (8,10],\\ -2.2511\cdot10^{-1}\cdot x^3 + 7.4589\cdot x^2 + -8.1405\cdot10^{1}\cdot x + 2.9377\cdot10^{2}, & \text{if } x \in (10,12],\\ 1.0752\cdot10^{-1}\cdot x^3 + -4.5159\cdot x^2 + 6.2293\cdot10^{1}\cdot x + -2.8102\cdot10^{2}, & \text{if } x \in (12,14]. \end{cases}$$



Conclusion

- Interpolation polynomiale
  - évaluer la fonction en un point : Polynôme de Lagrange -> méthode de Neville
  - *compiler* la fonction : Polynôme de Newton

- Interpolation polynomiale par morceau : splines
  - spline cubique d'interpolation : passage par les nœuds (points d'interpolation), mais on limite les oscillations.
  - spline cubique d'approximation : on régule mieux la fonction, mais minimise la distance aux nœuds (les points de passage)