Math avancée

Exercice 0.1 *Simplifier les expressions suivantes*
- $log_a(a\, log_b b^2)$,
- $log_a(x^4+3x^2+2)+log_a(x^4+5x^2+6)-4*log_a(\sqrt{x^2+2})$,
- $log_\pi(1-\cos x)+log_\pi(1+\cos x)-2log_\pi \sin x$.

Solutions :

—

$$\begin{aligned} log_a(a\, log_b b^2) &= log_a(a\, 2log_b b) \\ &= log_a(2a) = log_a a + log_a 2 \\ &= 1 + log_a 2 = 1 + \frac{1}{log_2 a}. \end{aligned}$$

—

$$\begin{aligned} log_a(x^4+3x^2+2)+log_a(x^4+5x^2+6)-4*log_a(\sqrt{x^2+2}) &= log_a(x^2+1)(x^2+2)+log_a(x^2+2)(x^2+3) \\ &\quad -2*log_a(x^2+2) \\ &= log_a\left(\frac{(x^2+1)(x^2+2)^2(x^2+3)}{(x^2+2)^2}\right) \\ &= log_a(x^2+1)(x^2+3) \\ &= log_a(x^4+4x^2+3). \end{aligned}$$

—

$$\begin{aligned} log_\pi(1-\cos x)+log_\pi(1+\cos x)-2log_\pi \sin x &= log_\pi(1-\cos x)(1+\cos x) - log_\pi \sin^2 x \\ &= log_\pi(1-\cos^2 x) - log_\pi \sin^2 x \\ &= log_\pi\left(\frac{1-\cos^2 x}{\sin^2 x}\right) = log_\pi\left(\frac{\sin^2 x}{\sin^2 x}\right) \\ &= log_\pi 1 = 0. \end{aligned}$$

Rappels: 0.2
1. $a^x = b, \quad x = log_a b$.
2. $a^{log_a x} = x \quad et \quad log_a a = 1$.
3. $log_a b = \frac{1}{log_b a}$.
4. $log_a c = log_a b * log_b c \quad et \quad log_b c = \frac{log_a c}{log_a b}$.

Exercice 0.3 *Résoudre pour x les expressions suivantes :*
- $log_5 x = 6$.
- $log_x 5 = 6$.
- $x^{\sqrt{2}} = 7$.

— $2^{2x} = 5^{x+1}$.
— $2 * log_3 x + log_9 x = 10$.

Solutions :

— *Noter que* $log_5 x = 6$ *implique que* $x = 5^{log_5 x} = 5^6 = 15625$.
— *Noter que* $log_x 5 = 6$ *implique que* $\frac{log_{10} 5}{log_{10} x} = 6$, *alors* $log_{10} 5 = 6 * log_{10} x$ *et* $log_{10} x = \frac{1}{6} log_{10} 5$. *Finalement* $x = 10^{\frac{1}{6} log_{10} 5} = 10^{0.116495} = 1.30766$.
— *L'équation* $x^{\sqrt{2}} = 7$ *peut être écrite comme* $log_{10} x^{\sqrt{2}} = log_{10} 7$ *ou* $\sqrt{2} log_{10} x = log_{10} 7$ *et* $log_{10} x = \frac{1}{\sqrt{2}} log_{10} 7$ *donne* $x = 10^{\frac{1}{\sqrt{2}} log_{10} 7} = 10^{0.597575} = 3.9589$.
— *L'équation* $2^{2x} = 5^{x+1}$ *peut être écrite comme* $2x log_{10} 2 = (x+1) log_{10} 5$ *ou* $x log_{10} 2^2 - x log_{10} 5 = log_{10} 5$ *et* $x(log_{10} 4 - log_{10} 5) = log_{10} 5$ *produit* $x log_{10} \frac{4}{5} = log_{10} 5$ *résulte en* $x = -7.21257$.
— *Dans* $2 log_3 x + log_9 x = 10$, *le terme* $log_9 x = \frac{log_3 x}{log_3 9} = \frac{log_3 x}{2 log_3 3} = \frac{1}{2} log_3 x$ *d'où on a* $2 log_3 x + log_9 x = 2 log_3 x + \frac{1}{2} log_3 x = 10$ *et* $\frac{5}{2} log_3 x = 10$ *ou* $log_3 x = 10 \frac{2}{5} = 4$. *Cette dernière implique que* $log_3 x = 4$ *ou* $x = 3^4 = 81$.

0.1 Evaluation des sommes d'entiers

Remarquer que la somme S des premiers n entiers positifs est

$$S = \sum_{i=1}^{n} i = 1 + 2 + 3 + \cdots + n = \frac{1}{2} n(n+1).$$

Pour voir cela, écrivons cette somme de deux manières différentes comme suit :

$$\begin{array}{rcl} S & = & 1 + 2 + 3 + \cdots + (n-1) + n \\ S & = & n + (n-1) + (n-2) + \cdots + 2 + 1 \\ -- & -- & - \\ 2S & = & (n+1) + (n+1) + (n+1) + \cdots + (n+1) + (n+1) \\ & = & n(n+1) \end{array}$$

D'où,

$$S = \frac{1}{2} n(n+1).$$

Théorème 0.4 *Formules des sommes d'entiers :*

1. $S = \sum_{i=1}^{n} 1 = \underbrace{1 + 1 + 1 + \cdots + 1}_{n \ termes} = n$
2. $S = \sum_{i=1}^{n} i = 1 + 2 + 3 + \cdots + n = \frac{1}{2} n(n+1)$
3. $S = \sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \cdots + n^2 = \frac{1}{6} n(n+1)(2n+1)$
4. $S = \sum_{i=1}^{n} r^{i-1} = 1 + r + r^2 + \cdots + r^{n-1} = \frac{r^n - 1}{r - 1}$ *pour* $r \neq 1$.

Preuve : *La formule (1) est évidente, la somme de n uns est n. La preuve de la formule (2) est donnée en haut. Pour démontrer la formule (3), rappelons les expressions*

$$(a+b)^3 = a^3 + 3a^2 b + 3ab^2 + b^2$$
$$a^3 - b^3 = (a-b)(a^2 + ab + b^2),$$

et on écrit n copies de l'identité

$$(k+1)^3 - k^3 = 3k^2 + 3k + 1,$$

une pour chaque valeur de k de 1 à n, et les additionner :

$$\begin{aligned}
2^3 - 1^3 &= 3*1^2 + 3*1 + 1 \\
3^3 - 2^3 &= 3*2^2 + 3*2 + 1 \\
4^3 - 3^3 &= 3*3^2 + 3*3 + 1 \\
\vdots &\quad \vdots \\
n^3 - (n-1)^3 &= 3*(n-1)^2 + 3*(n-1) + 1 \\
(n+1)^3 - n^3 &= 3*n^2 + 3*n + 1 \\
- - - - - - - - - - - - - - - &\quad - - - - - - - - - - - - \\
(n+1)^3 - 1^3 &= 3*\sum_{i=1}^{n} i^2 + 3*\sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 \\
n^3 + 3n^2 + 3n &= 3*\sum_{i=1}^{n} i^2 + \frac{3}{2}n(n+1) + n
\end{aligned}$$

alors

$$\begin{aligned}
3*\sum_{i=1}^{n} i^2 &= n^3 + 3n^2 + 3n - \frac{3}{2}n(n+1) - n \\
&= n^3 + 3n^2 + 2n - \frac{3}{2}n(n+1) \\
&= n(n^2 + 3n + 2) - \frac{3}{2}n(n+1) \\
\sum_{i=1}^{n} i^2 &= \frac{1}{3}n(n+1)(n+2) - \frac{1}{2}n(n+1) \\
&= \frac{1}{6}n(n+1)\left[2(n+2) - 3\right] \\
&= \frac{1}{6}n(n+1)(2n+1).
\end{aligned}$$

Pour démontrer la formule (4), on prend $S = \sum_{i=1}^{n} r^{i-1}$ et on la multiplie par r puis on soustrait du résultat S pour avoir

$$\begin{aligned}
rS - S &= r\sum_{i=1}^{n} r^{i-1} - \sum_{i=1}^{n} r^{i-1} \\
(r-1)S &= \sum_{i=1}^{n} r^{i} - \sum_{i=1}^{n} r^{i-1} \\
&= (r + r^2 + r^3 + r \cdots + r^n) - (1 + r + r^2 + r \cdots + r^{n-1}) \\
&= (r^n - 1)S.
\end{aligned}$$

Ainsi,

$$S = \frac{r^n - 1}{r - 1} \quad \textit{pour} \quad r \neq 1.$$

x	x_0	x_1	x_2	$\cdots$	x_n
y	y_0	y_1	y_2	$\cdots$	y_n

TABLE 1 – Données d'interpolation

Exemple : Evaluer l'expression $\sum_{k=m+1}^{n}(6k^2-4k+3)$, où $1 \leq m < n$.
Solution : Noter que

$$\begin{aligned}\sum_{k=1}^{n}(6k^2-4k+3) &= 6\sum_{k=1}^{n}k^2 - 4\sum_{k=1}^{n}k + 3\sum_{k=1}^{n}1 \\ &= 6\frac{1}{6}n(n+1)(2n+1) - 4\frac{1}{2}n(n+1) + 3n \\ &= 2n^3 + n^2 + 2n.\end{aligned}$$

D'où,

$$\begin{aligned}\sum_{k=m+1}^{n}(6k^2-4k+3) &= \sum_{k=1}^{n}(6k^2-4k+3) - \sum_{k=1}^{m}(6k^2-4k+3), \\ &= (2n^3+n^2+2n) - (2m^3+m^2+2m).\end{aligned}$$

Exercice 0.5 *Démontrer que* $S = \sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$ *utilisant* n *copies de l'identité* $(k+1)^2 - k^2 = 2k+1$, *une pour chaque valeur de* k *de* 1 *à* n, *puis par induction.*

0.2 Interpolation polynomiale

Supposons qu'on a une table de $(n+1)$ points (x_i, y_i) de données, voir la table 1 et on cherche un polynôme p de plus petit dégrée possible pour lequel $p(x_i) = y_i$ pour $0 \leq i \leq n$. Un tel polynôme est appelé le polynôme d'interpolation des données.
Supposons qu'on a un polynôme p_{k-1} de dégrée plus petit ou égale à $k-1$ avec $p_{k-1}(x_i) = y_i$ pour $0 \leq i \leq k-1$. On veut construire un polynôme p_k de la forme

$$p_k(x) = p_{k-1}(x) + c(x-x_0)(x-x_1)\cdots(x-x_{k-1}).$$

Ce dernier est un polynôme au plus de dégrée k. p_k interpole les données que p_{k-1} interpole, car

$$p_k(x_i) = p_{k-1}(x_i) = y_i \text{ pour } 0 \leq i \leq k-1.$$

Maintenant, on détermine les coefficients c inconnus à partir de la condition $p_k(x_k) = y_k$. Ceci mène à l'équation

$$p_{k-1}(x_k) + c(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1}) = y_k.$$

0.2.1 Interpolation polynômiale de Newton

Remarquer que chacun des polynômes $p_0, p_1, \ldots, p_n$ construits précédemment a la propriété que chaque p_k est obtenu en ajoutant un seul terme à p_{k-1}. A la fin du processus, p_n serai une somme de termes, et chacun de $p_0, p_1, \ldots, p_{n-1}$ apparaitra dans l'expression de p_n. Chaque p_k a la forme

$$\begin{aligned} p_k(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_k(x - x_0)\cdots(x - x_{k-1}) \\ &= \sum_{i=0}^{k} c_i \prod_{j=0}^{i-1}(x - x_j) \end{aligned}$$

On adopte la convention que $\prod_{j=0}^{m}(x - x_j) = 1$ chaque fois que $m < 0$. Noter que

$$\begin{aligned} p_0(x) &= c_0 \\ p_1(x) &= c_0 + c_1(x - x_0) = p_0(x) + c_1(x - x_0) \\ p_2(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) = p_1(x) + c_2(x - x_0)(x - x_1) \end{aligned}$$

etc. Ces polynômes sont appelés les polynômes d'interpolation sous la forme de Newton. Pour évaluer $p_k(x)$, on suppose que les coefficients c_0, c_1, ..., c_k sont connus, une méthode efficace appelée l'algorithme de Horner. Il peut être expliqué comme suit :

$$\begin{aligned} u &= \sum_{i=0}^{k} c_i \prod_{j=0}^{i-1} d_j = c_0 + c_1 d_0 + c_2 d_0 d_1 + \cdots + c_k d_0 d_1 \cdots d_{k-1} \\ &= (\cdots(((c_k)d_{k-1} + c_{k-1})d_{k-2} + c_{k-2})d_{k-3} + \cdots + c_1)d_0 + c_0 \end{aligned}$$

L'algorithme pour calculer u peut être développé comme suit :

$$\begin{aligned} u_k &\leftarrow c_k \\ u_{k-1} &\leftarrow u_k d_{k-1} + c_{k-1} \\ u_{k-2} &\leftarrow u_{k-1} d_{k-2} + c_{k-2} \\ &\vdots \\ u_0 &\leftarrow u_1 d_0 + c_0 \end{aligned}$$

Puisqu'on a besoin seulement de u_0, on peut écrire d'une manière algorithmique les étapes suivantes :
$u \leftarrow c_k$
pour $i = k - 1$ à 0 pas -1 faire
$u \leftarrow u * d_i + c_i$
fin faire
Pour une valeur précise t de x, on utilise l'algorithme suivant pour obtenir $u = p_k(t)$
$u \leftarrow c_k$
pour $i = k - 1$ à 0 pas -1 faire
$u \leftarrow u * (t - x_i) + c_i$
fin faire
Les coefficients c_i peut être obtenus utilisant l'équation suivante :

$$c_k = \frac{y_k - p_{k-1}(x_k)}{(x_k - x_0)(x_k - x_1)\cdots(x_k - x_{k-1})}.$$

Un algorithme pour le calcul de c_0, c_1, ..., c_n utilisant les données de la table 1 est :
$c_0 \leftarrow y_0$
pour $k = 0$ à n faire

$d \leftarrow x_k - x_{k-1}$
$u \leftarrow c_{k-1}$
pour $i = k-2$ à 0 pas -1 faire
$u \leftarrow u * (x_k - x_i) + c_i$
$d \leftarrow d * (x_k - x_i)$
fin faire
$c_k \leftarrow (y_k - u)/d$
fin faire
Cette méthode est bonne, mais une procédure plus efficace pour calculer les coefficients c_0, c_1, ..., c_n existe et utilise les différences divisées.

0.2.2 Interpolation polynômiale de Lagrange

Etant donnée la table 1, il est important de comprendre qu'il y'a un et seulement un polynôme d'interpolation de dégrée plus petit ou égale à $n+1$ associé avec les données. On suppose que les $n+1$ abscisses des données sont distinctes. Mais, la possibilité d'exprimer ce polynôme sous différentes formes et par différentes approches existe. Prenons

$$p(x) = y_0 l_0(x) + y_1 l_1(x) + \cdots + y_n l_n(x) = \sum_{k=0}^{n} y_k l_k(x).$$

Où l_0, l_1, ..., l_n sont des polynômes qui dépendent des nœuds x_0, x_1, ..., x_n, mais pas des ordonnées y_0, y_1, ..., y_n. Puisque toutes les ordonnées pourraient être 0 sauf pour un 1 occupant la $i^{ième}$ position, on peut voir ceci

$$\delta_{ij} = p_n(x_j) = \sum_{k=0}^{n} y_k l_k(x_j) = \sum_{k=0}^{n} \delta_{ki} l_k(x_j) = l_i(x_j).$$

Le delta de Kronecker est défini comme

$$\delta_{ki} = \begin{cases} 1 & \text{si} \quad k = i \\ 0 & \text{si} \quad k \neq i. \end{cases}$$

On peut facilement arriver à un ensemble de polynômes ayant cette propriété. Prenons l_0. Ce dernier est un polynôme de dégrée n qui prend la valeur 0 à x_1, x_2, ..., x_n, et la valeur 1 à x_0. Il est clair que l_0 doit avoir la forme

$$l_0(x) = c(x - x_1)(x - x_2) \cdots (x - x_n) = c \prod_{j=1}^{n} (x - x_j).$$

La valeur de c est obtenue en prenant $x = x_0$ pour que

$$1 = c \prod_{j=1}^{n} (x_0 - x_j)$$

et

$$c = \prod_{j=1}^{n} (x_0 - x_j)^{-1}.$$

D'où,

$$l_0(x) = \prod_{j=1}^{n} \frac{x - x_j}{x_0 - x_j}$$

x	5	-7	-6	0
y	1	-23	-54	-954

TABLE 2 – Données d'interpolation de l'exemple 1

Chaque $l_i(x)$ est obtenu d'une manière similaire, et la formule générale est

$$l_i(x) = \prod_{j=1}^{n} \frac{x - x_j}{x_i - x_j} \quad 0 \leq i \leq n.$$

Pour l'ensemble des nœuds x_0, x_1, ..., x_n, ces polynômes sont connus comme les fonctions cardinales. Ainsi, l'expression

$$p(x) = \sum_{k=0}^{n} y_k l_k(x)$$

avec

$$l_i(x) = \prod_{j=1}^{n} \frac{x - x_j}{x_i - x_j} \quad 0 \leq i \leq n$$

constituent l'algorithme d'interpolation polynômiale de Lagrange.
Exemple : Supposons qu'on a la table 2. et on cherche un polynôme p de plus petit dégrée possible pour lequel $p(x_i) = y_i$ pour $0 \leq i \leq 3$. Le polynôme obtenu par la méthode d'interpolation polynômiale de Newton est

$$p_3(x) = 1 + 2(x-5) + 3(x-5)(x+7) + 4(x-5)(x+7)(x+6).$$

Les fonctions cardinales de la méthode d'interpolation polynômiale de Lagrange sont

$$\begin{aligned}
l_0(x) &= \frac{(x+7)(x+6)x}{(5+7)(5+6)5} = \frac{1}{660}x(x+6)(x+7) \\
l_1(x) &= \frac{(x-5)(x+6)x}{(-7-5)(-7+6)(-7)} = -\frac{1}{84}x(x-5)(x+6) \\
l_2(x) &= \frac{(x-5)(x+7)x}{(-6-5)(-6+7)(-6)} = -\frac{1}{66}x(x-5)(x+7) \\
l_3(x) &= \frac{(x-5)(x+7)(x+6)}{(0-5)(0+7)(0+6)} = -\frac{1}{210}(x-5)(x+7)(x+6)
\end{aligned}$$

Le polynôme d'interpolation polynômiale de Lagrange est alors

$$p_3(x) = l_0(x) - 23l_1(x) - 54l_2(x) - 954l_3(x).$$

Si on prend $x = x_i$ pour $0 \leq i \leq n$ dans le polynôme de Lagrange,

$$p(x) = y_0 l_0(x) + y_1 l_1(x) + \cdots + y_n l_n(x) = \sum_{k=0}^{n} y_k l_k(x).$$

on obtient les $n+1$ linéaires équations suivantes :

$$\begin{bmatrix} l_0(x_0) & l_1(x_0) & \cdots & l_n(x_0) \\ l_0(x_1) & l_1(x_1) & \cdots & l_n(x_1) \\ \vdots & \vdots & & \vdots \\ l_0(x_n) & l_1(x_n) & \cdots & l_n(x_n) \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} p(x_0) \\ p(x_1) \\ \vdots \\ p(x_n) \end{bmatrix}.$$

La matrice des coefficients se réduit à une identité I donnant la solution $y_i = p(x_i)$ pour $0 \leq i \leq n$. Puisque seulement un polynôme d'interpolation peut exister pour les $n+1$ points distincts $x_0, x_1, \ldots, x_n$, on a un polynôme de dégrée au plus n :

$$p(x) = \sum_{k=0}^{n} p(x_k) l_k(x).$$

0.2.3 Différences divisées

Prenons f une fonction dont les valeurs sont connues ou calculables à un ensemble de points (nœuds) $x_0, x_1, \ldots, x_n$. On suppose que ces points sont distincts. On sait qu'il existe un unique polynôme de dégrée au plus n qui est un polynôme d'interpolation de f aux $n+1$ noeuds. c'est à dire,

$$p(x_i) = f(x_i) \quad \text{pour} \quad 0 \leq i \leq n.$$

Prenons la méthode d'interpolation polynômiale de Newton avec :

$$\begin{aligned} q_0(x) &= 1 \\ q_1(x) &= (x - x_0) \\ q_2(x) &= (x - x_0)(x - x_1) \\ q_3(x) &= (x - x_0)(x - x_1)(x - x_2) \\ &\vdots \\ q_n(x) &= (x - x_0)(x - x_1)(x - x_2) \cdots (x - x_{n-1}) \end{aligned}$$

Celles mènent à la forme de Newton

$$p(x) = \sum_{k=0}^{n} c_k q_k(x).$$

Le problème alors est de déterminer les coefficients c_i. Mais, on sait que $p(x_i) = f(x_i)$ pour $0 \leq i \leq n$. Alors,

$$p(x_i) = \sum_{k=0}^{n} c_k q_k(x_i) = f(x_i) \quad \text{pour} \quad 0 \leq i \leq n.$$

Sous la forme matricielle, on a

$$\begin{bmatrix} q_0(x_0) & 0 & 0 & \cdots & 0 \\ q_0(x_1) & q_1(x_1) & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ q_0(x_n) & q_1(x_n) & \cdots & q_n(x_n) & \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}.$$

Remarquer que la matrice

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & q_1(x_1) & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & q_1(x_n) & q_2(x_n) & \cdots & q_n(x_n) \end{bmatrix}$$

est une matrice $(n+1) \times (n+1)$ triangulaire inférieure, parce que

$$q_k(x) = \prod_{j=0}^{k-1} (x - x_j)$$
$$q_k(x_i) = \prod_{j=0}^{k-1} (x_i - x_j) = 0 \quad \text{si} \quad i \leq k-1.$$

Prenons $f[x_0, x_1, \ldots, x_n]$ le coefficient de x^n dans le polynôme de dégrée au plus n qui est l'interpolation de f à $x_0, x_1, \ldots, x_n$. Les expressions $f[x_0, x_1, \ldots, x_n]$ sont appelées les différences divisées de f. Premièrement, $f[x_0]$ le coefficient de x^0 dans le polynôme de dégrée 0 qui est l'interpolation de f à x_0. Alors, on a

$$f[x_0] = f(x_0).$$

La quantité $f[x_0, x_1]$ le coefficient de x dans le polynôme de dégrée au plus 1 qui est l'interpolation de f à x_0 et x_1. Puisque le polynôme est

$$p(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0).$$

On voit que le coefficient de $q_1(x)$ est

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

Le polynôme d'interpolation est

$$p(x) = f(x_0) + f[x_0, x_1](x - x_0).$$

L'interpolation polynômiale de Newton prend alors la forme suivante :

$$p(x) = \sum_{k=0}^{n} c_k q_k(x) = \sum_{k=0}^{n} f[x_0, x_1, \ldots, x_k] \prod_{j=0}^{k-1} (x - x_j).$$

Les différences divisées satisfont l'équation

$$f[x_1, x_2, \ldots, x_n] = \frac{f[x_0, x_1, \ldots, x_n] - f[x_0, x_1, \ldots, x_{n-1}]}{x_n - x_0}.$$

Si une table des valeurs d'une fonction $(x_i, f(x_i))$ est donnée, on peut l'utiliser pour construire une table des différences divisées

$$\begin{array}{cc|cccc} x_0 & f[x_0] & f[x_0,x_1] & f[x_0,x_1,x_2] & f[x_0,x_1,x_2,x_3] & f[x_0,x_1,x_2,x_3,x_4] \\ x_1 & f[x_1] & f[x_1,x_2] & f[x_1,x_2,x_3] & f[x_1,x_2,x_3,x_4] & \\ x_2 & f[x_2] & f[x_2,x_3] & f[x_2,x_3,x_4] & & \\ x_3 & f[x_3] & f[x_3,x_4] & & & \\ x_4 & f[x_4] & & & & \end{array}$$

x	3	1	5	6
$f(x)$	1	-3	2	4

TABLE 3 – Données d'interpolation de l'exemple 2

En générale,

$$f[x_i,\, x_{i+1},\, \ldots,\, x_{i+j}] = \frac{f[x_{i+1},\, x_{i+2},\, \ldots,\, x_{i+j}] - f[x_i,\, x_{i+1},\, \ldots,\, x_{i+j-1}]}{x_{i+j} - x_i}.$$

Exemple : Calculer la table des différences divisées pour les valeurs de la fonction, voir la table 3. Dans ce cas, la table des différences divisées est

$$\begin{array}{cc|cccc} 3 & 1 & f[x_0,x_1] & f[x_0,x_1,x_2] & f[x_0,x_1,x_2,x_3] & f[x_0,x_1,x_2,x_3,x_4] \\ 1 & -3 & f[x_1,x_2] & f[x_1,x_2,x_3] & f[x_1,x_2,x_3,x_4] & \\ 5 & 2 & f[x_2,x_3] & f[x_2,x_3,x_4] & & \\ 6 & 4 & & & & \end{array}$$

Où

$$\begin{aligned} f[x_0,x_1] &= \frac{f(x_1)-f(x_0)}{x_1-x_0} = \frac{-3-1}{1-3} = 2 \\ f[x_1,x_2] &= \frac{f(x_2)-f(x_1)}{x_2-x_1} = \frac{2-(-3)}{5-1} = \frac{5}{4} \\ f[x_2,x_3] &= \frac{f(x_3)-f(x_2)}{x_3-x_2} = \frac{4-2}{6-5} = 2 \end{aligned}$$

alors

$$\begin{aligned} f[x_0,x_1,x_2] &= \frac{f[x_1,x_2]-f[x_0,x_1]}{x_2-x_0} = \frac{\frac{5}{4}-2}{2-1} = -\frac{3}{8} \\ f[x_1,x_2,x_3] &= \frac{f[x_2,x_3]-f[x_1,x_2]}{x_3-x_1} = \frac{2-\frac{5}{4}}{6-1} = \frac{3}{20} \end{aligned}$$

et

$$f[x_0,x_1,x_2,x_3] = \frac{f[x_1,x_2,x_3]-f[x_0,x_1,x_2]}{x_3-x_0} = \frac{\frac{3}{20}-(-\frac{3}{8})}{6-3} = \frac{7}{40}.$$

La table **??** des différences divisées est alors

$$\begin{array}{cc|ccc} 3 & 1 & 2 & -\frac{3}{8} & \frac{7}{40} \\ 1 & -3 & \frac{5}{4} & \frac{3}{20} & \\ 5 & 2 & 2 & & \\ 6 & 4 & & & \end{array}$$

Le polynôme de l'interpolation polynômiale de Newton est donnée comme :

$$p(x) = 1 + 2(x-3) - \frac{3}{8}(x-3)(x-1) + \frac{7}{40}(x-3)(x-1)(x-5).$$

x_0	c_{00}	c_{01}	c_{02}	c_{03}	$\cdots$	$c_{0,n-1}$	$c_{0,n}$
x_1	c_{10}	c_{11}	c_{12}	c_{13}	$\cdots$	$c_{1,n-1}$	
x_2	c_{20}	c_{21}	c_{22}	c_{23}	$\cdots$		
$\vdots$	$\vdots$	$\vdots$					
x_{n-1}	$c_{n-1,0}$	$c_{n-1,1}$					
x_n	c_{n0}						

TABLE 4 – Table de l'algorithme d'interpolation de Newton

Algorithme des différences divisées :
Un algorithme pour calculer les éléments de la table des différences divisées peut être très efficace. Considérons la table 4, où

$$c_{ij} = f[x_i, x_{i+1}, \ldots, x_{i+j}].$$

On peut avoir l'algorithme suivant pour calculer les coefficients c_{ij}
pour $j = 1$ à n faire
pour $i = 0$ à $n - j$ faire
$c_{ij} \leftarrow (c_{i+1,j-1} - c_{i,j-1})/(x_{i+j} - x_i)$
fin faire
fin faire
Dans cet algorithme, les nombres c_{i0} sont les valeurs de la fonction f aux points x_i. Celles-ci sont aussi les valeurs qu'aura le polynôme d'interpolation à ces points. Le polynôme d'interpolation est

$$p(x) = \sum_{i=0}^{n} c_{0i} \prod_{j=0}^{i-1} (x - x_j).$$

Si l'algorithme des différences divisées est utilisé pour calculer seulement les coefficients de l'interpolation polynômiale de Newton, alors un autre algorithme pourrait être conçu qui utiliser moins de mémoire de stockage. L'algorithme est : pour $i = 0$ à n faire
$d_i \leftarrow f(x_i)$
fin faire
pour $j = 1$ à n faire
pour $i = n$ à j pas -1 faire
$d_j \leftarrow (d_i - d_{i-1})/(x_i - x_{i-j})$
fin faire
fin faire
Ainsi, le vecteur d contient les coefficients du polynôme :

$$p(x) = \sum_{i=0}^{n} d_i \prod_{j=0}^{i-1} (x - x_j).$$

0.3 Interpolation spline

Une fonction spline est constituée de morceaux de polynôme sur des sous-intervalles liés ensemble avec certaines conditions de continuité. Formellement, supposons que $n + 1$ points $t_0, t_1, \ldots, t_n$ ont été spécifiés et satisfont la condition $t_0 < t_1 < \ldots < t_n$. Ces points sont appelés nœuds. Supposons qu'un entier $k \geq 0$ a été prescrit. Une fonction de dégrée k ayant les nœuds $t_0, t_1, \ldots, t_n$ est une fonction S telle que :

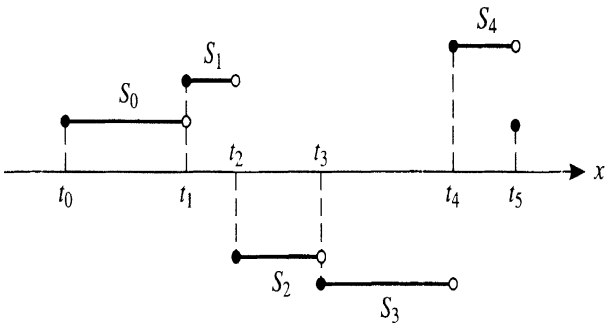

FIGURE 1 – Une spline de dégrée 0

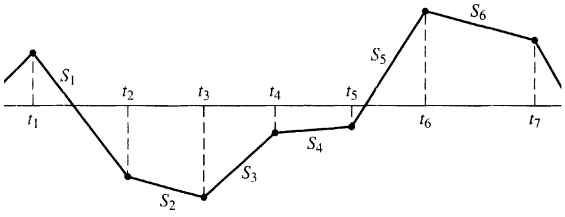

FIGURE 2 – Une spline de dégrée 1

1. Sur chaque intervalle $[t_{i-1}, t_i[$, S est un polynôme de dégrée plus petit ou égale à k.
2. S a une $(k-1)$ième dérivée continue dans $[t_0, t_n]$.

Ainsi, S est un polynôme continu par morceaux de dégrée au plus k ayant des dérivées continues de tous les ordres jusqu'à $k-1$. Les fonctions splines ou juste les splines de dégrée 0 sont constantes par morceaux. Une spline de dégrée 0 peut être donnée d'une manière explicite sous la forme :

$$S(x) = \begin{cases} S_0(x) = c_0 & x \in [t_0, t_1[\\ S_1(x) = c_1 & x \in [t_1, t_2[\\ \vdots & \vdots \\ S_{n-1}(x) = c_{n-1} & x \in [t_{n-1}, t_n] \end{cases}$$

Les intervalles $[t_{i-1}, t_i[$ ne se coupent pas. Une spline typique de dégrée 0 est montrée par la figure 1. La figure 2 montre le graphe d'une fonction spline typique de dégrée 1 avec neuf nœuds. Une telle fonction peut être définie comme suit :

$$S(x) = \begin{cases} S_0(x) = a_0x + b_0 & x \in [t_0, t_1[\\ S_1(x) = a_1x + b_1 & x \in [t_1, t_2[\\ \vdots & \vdots \\ S_{n-1}(x) = a_{n-1}x + b_{n-1} & x \in [t_{n-1}, t_n] \end{cases}$$

Si les noeuds t_i et les coefficients a_i et b_i sont tous donnés, alors la valeur de S à x est obtenue en identifiant e premier le sous-intervalle $[t_i, t_{i+1}[$ qui contient x. La spline peut être définie sur toute la ligne réelle. Dans notre cas, on peut utiliser $a_0x + b_0$ sur l'intervalle $]-\infty, t_1[$

x	x_0	x_1	$\cdots$	x_n
y	y_0	y_1	$\cdots$	y_n

TABLE 5 – Données d'interpolation de la spline cubique

et $a_{n-1}x + b_{n-1}$ sur l'intervalle $[t_{n-1}, \infty[$. La fonction S est continue et alors les polynômes par morceaux concordent aux nœuds, c'est à dire, $S_i(t_{i+1}) = S_{i+1}(t_{i+1})$. Un pseudo-code pour évaluer la spline $S(x)$ est :
Entrer $t_i,\ a_i,\ b_i,\ x,\ n$
pour $i = 1$ à $n - 1$ faire
si $x < t_i$ alors
$S(x) = a_{i-1}x + b_{i-1}$
sortir $S(x)$
fin boucle
fin si
fin faire
$S(x) = a_{n-1}x + b_{n-1}$
sortir $S(x)$
On est intéresse par la spline cubique ($k = 3$) car elle est le plus souvent utilisée en pratique. Pour cela, supposons qu'on a la table 5. et qu'une spline cubique S doit être construite pour l'interpolation des données de la table 5. Sur chaque intervalle $[x_0, x_1]$, $[x_1, x_2]$, $\ldots, [x_{n-1}, x_n]$, S est donnée par un polynôme cubique différent. Prenons S_i le polynôme cubique qui représente S sur l'intervalle $[x_i, x_{i+1}]$. Alors,

$$S(x) = \begin{cases} S_0(x) & x \in [x_0, x_1] \\ S_1(x) & x \in [x_1, x_2] \\ \vdots & \vdots \\ S_{n-1}(x) & x \in [x_{n-1}, x_n] \end{cases}$$

Les polynômes S_{i-1} et S_i interpolent la même valeur au point x_i et ainsi

$$S_{i-1}(x_i) = y_i = S_i(x_i), \quad \text{pour} \quad 1 \leq i \leq n-1.$$

0.4 Splines cubiques

Dans cette approche, l'intervalle d'interpolation est divisé en un ensemble de sous-intervalles et un polynôme d'interpolation différent est construit sur chaque sous-intervalle. Celle-ci est appelée l'interpolation polynômiale par morceaux. L'approximation ou l'interpolation polynômiale par morceaux la plus connue est celle qui utilise des polynômes cubiques entre paires de nœuds consécutifs et est appelée interpolation spline cubique. En générale, un polynôme cubique a quatre constantes, alors il y'a suffisamment de flexibilité dans la procédure de la spline cubique pour assurer que le polynôme d'interpolation est non seulement continuellement dérivable, mais a aussi une dérivée du seconde ordre continue.

Définition 0.6 *Etant donnée une fonction f définie sur l'intervalle $[a, b]$ et un ensemble de nœuds $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$, un interpolant spline cubique S pour f est une fonction qui satisfait les conditions suivantes :*

1. *$S(x)$ est un polynôme cubique, dénoté par $S_j(x)$ sur le sous-intervalle $[x_j, x_{j+1}]$ pour chaque $0 \leq j \leq n - 1$;*

2. $S_j(x_j) = f(x_j)$ *et* $S_j(x_{j+1}) = f(x_{j+1})$ *pour chaque* $0 \leq j \leq n-1$;
3. $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ *pour chaque* $0 \leq j \leq n-1$; *(impliqué par (2))*
4. $\dot{S}_{j+1}(x_{j+1}) = \dot{S}_j(x_{j+1})$ *pour chaque* $0 \leq j \leq n-1$;
5. $\ddot{S}_{j+1}(x_{j+1}) = \ddot{S}_j(x_{j+1})$ *pour chaque* $0 \leq j \leq n-1$;
6. *Un des ensembles suivants des conditions limites est satisfait :*
   (a) $\ddot{S}(x_0) = \ddot{S}(x_n) = 0$ *(limites libres ou naturelles) ;*
   (b) $\dot{S}(x_0) = \dot{f}(x_0)$ *et* $\dot{S}(x_n) = \dot{f}(x_n)$ *(limites fixes ou serrées).*

Une spline naturelle n'a pas de conditions imposées pour la direction à ses points extrêmes, alors la courbe prend la forme d'une ligne droite après son passage à travers les points d'interpolation les plus proches des points extrêmes. C'est à dire, quand les conditions limites libres sont utilisées, la spline est appelée spline naturelle.

Construction d'une spline cubique

Une spline définie sur un intervalle qui est divisé en n sous-intervalles exige l'obtention de $4n$ constantes. Pour construire la spline cubique S pour une fonction f donnée, les conditions de la définition sont appliquées aux polynômes cubiques :

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3,$$

pour chaque $0 \leq j \leq n-1$. Puisque $S_j(x_j) = f(x_j) = a_j$, la condition (3) peut être appliquée pour obtenir

$$a_{j+1} = S_{j+1}(x_{j+1}) = S_j(x_{j+1}) = a_j + b_j(x_{j+1} - x_j) + c_j(x_{j+1} - x_j)^2 + d_j(x_{j+1} - x_j)^3,$$

pour chaque $0 \leq j \leq n-2$. Prenons $h_j = x_{j+1} - x_j$ pour chaque $0 \leq j \leq n-1$, et $a_n = f(x_n)$, alors on obtient

$$a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3, \tag{0.1}$$

pour chaque $0 \leq j \leq n-1$. D'une manière similaire, prenons $b_n = \dot{S}(x_n)$ et remarquons que

$$\dot{S}_j(x) = b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2$$

implique que $\dot{S}_j(x_j) = b_j$, pour chaque $0 \leq j \leq n-1$. La condition (4) de la définition donne

$$b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2, \tag{0.2}$$

pour chaque $0 \leq j \leq n-1$. Une autre relation entre les coefficients de S_j est obtenue en définissant $c_n = \ddot{S}(x_n)/2$ et appliquant la condition (5). Alors, pour chaque $0 \leq j \leq n-1$

$$c_{j+1} = c_j + 3d_j h_j. \tag{0.3}$$

La valeur de d_j obtenue de l'équation (0.1) et substituée dans les équations (0.2) et (0.3) pour donner les équations

$$a_{j+1} = a_j + b_j h_j + \frac{1}{3}(2c_j + c_{j+1})h_j^2, \tag{0.4}$$

et

$$b_{j+1} = b_j + (c_j + c_{j+1})h_j, \tag{0.5}$$

pour chaque $0 \leq j \leq n-1$. La dernière équation associant les coefficients est obtenue en résolvant l'équation appropriée comme l'équation (0.4), premièrement pour b_j, pour avoir

$$b_j = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(2c_j + c_{j+1}), \tag{0.6}$$

et puis, avec une réduction de l'indice, pour b_{j-1}. Ceci donne

$$b_{j-1} = \frac{1}{h_{j-1}}(a_j - aj_{-1}) - \frac{h_{j-1}}{3}(2c_{j-1} + c_j),$$

Substituant ces deux dernières expressions dans l'équation (0.5), avec l'indice réduit par un, donne le système linéaire d'équations

$$h_{j-1}c_{j-1} + 2(h_{j-1} + h_j)c_j + h_j c_{j+1} = \frac{3}{h_j}(a_{j+1} - a_j) - \frac{3}{h_{j-1}}(a_j - aj - 1), \qquad (0.7)$$

pour chaque $1 \leq j \leq n-1$.

Spline naturelle

Dans ce cas, les conditions limites naturelles sont $\ddot{S}(x_0) = 0$ et $\ddot{S}(x_n) = 0$. Ces dernières implique que $c_{_0} = 0$ et $c_{_n} = 0$ respectivement. L'équation (0.7) donne alors le système $Ax = b$ où la matrice A $(n+1) \times (n+1)$ est

$$A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2(h_{n-3}+h_{n-2}) & h_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix},$$

et b et x sont les vecteurs

$$b = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a0) \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - an - 2) \\ 0 \end{bmatrix}, \quad \text{et} \quad x = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix}.$$

Algorithme de la spline cubique naturelle :

Pour construire un interpolant spline cubique S pour la fonction f, définie aux points $x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n$, et satisfaisant $\ddot{S}(x_0) = \ddot{S}(x_n) = 0$, on a les étapes suivantes :

- Entrée : x_0, x_1, x_2, …, x_{n-1}, x_n; $a_0 = f(x_0)$, $a_1 = f(x_1)$, …, $a_n = f(x_n)$.
- Sortie : a_j, b_j, c_j, d_j pour $j = 0, 1, \ldots, n-1$.
  Remarquer que $S(x) = S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$ avec $x_j \leq x \leq x_{_{j+1}}$.
- Etape 1 : Pour $i = 0, 1, \ldots, n-1$, prendre $h_i = x_{i+1} - x_i$.
- Etape 2 : Pour $i = 1, 2, \ldots, n-1$, prendre $\alpha_i = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - ai - 1)$.
- Etape 3 : Prendre $\vartheta_0 = 1$, $\mu_0 = 0$, $\nu_0 = 0$.
- Etape 4 : Pour $i = 1, 2, \ldots, n-1$, prendre $\vartheta_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$, $\mu_i = h_i/\vartheta_i$, $\nu_i = (\alpha_i - h_{i-1}\nu_{i-1})/\vartheta_i$.
- Etape 5 : Prendre $\vartheta_n = 1$, $\nu_n = 0$, $c_n = 0$.
- Etape 6 : Pour $j = n-1, n-2, \ldots, 1, 0$, prendre $c_j = \nu_j - \mu_j c_{j+1}$; $b_j = (a_{j+1} - a_j)/h_j - h_j(2c_j + c_{j+1})/3$; $d_j = (c_{j+1} - c_j)/3h_j$.
- Etape 7 : Afficher a_j, b_j, c_j, d_j pour $j = 0, 1, \ldots, n-1$. Stop.

Spline serrée ou fixe

Si f est définie aux points $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$, et dérivable à a et b,, alors

f a un interpolant spline cubique fixe unique S aux noeuds x_0, x_1, ..., x_{n-1}, x_n, satisfaisant les conditions limites fixes $\dot{S}(a) = \dot{S}(x_0) = \dot{f}(a)$ et $\dot{S}(x_n) = \dot{S}(b) = \dot{f}(b)$.
Puisque $\dot{f}(a) = \dot{S}(a) = \dot{S}(x_0) = b_0$, l'équation (0.7) avec $j = 0$ implique $\dot{f}(a) = \frac{1}{h_0}(a_1 - a_0) - \frac{h_0}{3}(2c_0 + c_1)$. Par conséquent,

$$2h_0c_0 + h_0c_1 = \frac{3}{h_0}(a_1 - a_0) - 3\dot{f}(a).$$

D'une manière similaire,

$$\dot{f}(b) = b_n = b_{n-1} + h_{n-1}(c_{n-1} + c_n),$$

alors équation (0.5) avec $j = n - 1$ implique que

$$\begin{aligned}\dot{f}(b) &= \frac{1}{h_{n-1}}(a_n - a_{n-1}) - \frac{h_{n-1}}{3}(2c_{n-1} + c_n) + h_n(c_{n-1} + c_n) \\ &= \frac{1}{h_{n-1}}(a_n - a_{n-1}) + \frac{h_{n-1}}{3}(c_{n-1} + 2c_n),\end{aligned}$$

et

$$h_{n-1}c_{n-1} + 2h_{n-1}c_n = 3\dot{f}(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}).$$

Les équations (0.7) avec

$$2h_0c_0 + h_0c_1 = \frac{3}{h_0}(a_1 - a_0) - 3\dot{f}(a),$$

et

$$h_{n-1}c_{n-1} + 2h_{n-1}c_n = 3\dot{f}(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1})$$

déterminent le système linéaire $Ax = b$, où la matrice A $(n+1) \times (n+1)$ est

$$A = \begin{bmatrix} 2h_0 & h_0 & 0 & \cdots & 0 & 0 & 0 \\ h_0 & 2(h_0 + h_1) & h_1 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1 + h_2) & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2(h_{n-3} + h_{n-2}) & h_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{bmatrix},$$

et b et x sont les vecteurs

$$b = \begin{bmatrix} \frac{3}{h_0}(a_1 - a_0) - 3\dot{f}(a) \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a0) \\ \vdots \\ \frac{3}{h_{n-1}}(a_n - a_{n-1}) - \frac{3}{h_{n-2}}(a_{n-1} - an - 2) \\ 3\dot{f}(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{bmatrix}, \quad \text{et} \quad x = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix}.$$

Algorithme de la spline cubique fixe :
Pour construire un interpolant spline cubique S pour la fonction f, définie aux points $x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n$, et satisfaisant $\dot{S}(x_0) = \dot{f}(x_0)$ et $\dot{S}(x_n) = \dot{f}(x_n)$. , on a les étapes suivantes :

— Entrée : x_0, x_1, x_2, …, x_{n-1}, x_n; $a_0 = f(x_0)$, $a_1 = f(x_1)$, …, $a_n = f(x_n)$; $\lambda_0 = \dot{f}(x_0)$, $\lambda_n = \dot{f}(x_n)$.
— Sortie : a_j, b_j, c_j, d_j pour $j = 0, 1, \ldots, n-1$.
Remarquer que $S(x) = S_j(x) = a_j + b_j(x-x_j) + c_j(x-x_j)^2 + d_j(x-x_j)^3$ avec $x_j \leq x \leq x_{j+1}$.
— Etape 1 : Pour $i = 0, 1, \ldots, n-1$, prendre $h_i = x_{i+1} - x_i$.
— Etape 2 : Prendre $\alpha_0 = \frac{3}{h_0}(a_1 - a_0) - 3\lambda_0$ et $\alpha_n = 3\lambda_n - \frac{3}{h_{n-1}}(a_n - a_{n-1})$.
— Etape 3 : Pour $i = 1, 2, \ldots, n-1$, prendre $\alpha_i = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - ai - 1)$.
— Etape 4 : Prendre $\vartheta_0 = 2h_0$, $\mu_0 = 0.5$, $\nu_0 = \alpha_0/\vartheta_0$.
— Etape 5 : Pour $i = 1, 2, \ldots, n-1$, prendre $\vartheta_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$, $\mu_i = h_i/\vartheta_i$, $\nu_i = (\alpha_i - h_{i-1}\nu_{i-1})/\vartheta_i$.
— Etape 6 : Prendre $\vartheta_n = h_{n-1}(2 - \mu_{n-1})$, $\nu_n = (\alpha_n - h_{n-1}\nu_{n-1})/\vartheta_n$, $c_n = \nu_n$.
— Etape 7 : Pour $j = n-1, n-2, \ldots, 1, 0$, prendre $c_j = \nu_j - \mu_j c_{j+1}$; $b_j = (a_{j+1} - a_j)/h_j - h_j(2c_j + c_{j+1})/3$; $d_j = (c_{j+1} - c_j)/3h_j$.
— Etape 8 : Afficher a_j, b_j, c_j, d_j pour $j = 0, 1, \ldots, n-1$. Stop.

0.5 Intégration et dérivée numériques

0.5.1 Intégration numérique

Supposons qu'on veut déterminer ou développer une règle d'approximation pour l'intégrale $\int_a^b f(x)dx$, pour cela, prenons $x_0 = a$, $x_1 = b$, et $h = x_1 - x_0 = b - a$.

1. Règle trapézoïdale : Prenons $x_0 = a$, $x_1 = b$, et $h = x_1 - x_0 = b - a$, et le polynôme linéaire de Lagrange $p_1(x) = \frac{(x-x_1)}{x_0-x_1}f(x_0) + \frac{(x-x_0)}{x_1-x_0}f(x_1)$. On a

$$\int_a^b f(x)dx = \frac{h}{2}(f(x_0) + f(x_1)) - \frac{h^3}{2}\ddot{f}(\xi) \quad \text{où} \quad \xi \in]x_0, x_1[.$$

2. Règle de Simpson : est le résultat de l'intégration de Lagrange seconde polynôme

$$p_2(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_{00}-x_2)}f(x_1) + \frac{(x-x_0)(x-x_1)}{2(x_2-x_0)(x_2-x_1)}f(x_2)$$

sur $\lfloor a, b \rfloor$ avec des nœuds également espacés $x_0 = a$, $x_2 = b$, et $x_1 = a + h$ où $h = (b-a)/2$. Voir la figure 3. Ainsi,

$$\int_a^b f(x)dx = \int_{x_0}^{x_2} \left[\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_{00}-x_2)}f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}f(x_2) \right] dx + \int_{x_0}^{x_2} frac(x-x_0)(x-x_1)(x-x_2)6f^{(3)}(\xi(x))dx.$$

Pour inclure des termes d'ordres élevés f est développée en polynôme de Taylor de troisième ordre autour de par exemple x_1. Alors, pour chaque $x \in \lfloor x_0, x_2 \rfloor$, un nombre $\xi(x) \in]x_0, x_2[$ existe avec

$$f(x) = f(x_1) + \dot{f}(x_1)(x-x_1) + \frac{1}{2}\ddot{f}(x_1)(x-x_1)^2 + \frac{1}{6}\dddot{f}(x_1)(x-x_1)^3 + \frac{1}{24}f^{(4)}(\xi(x))(x-x_1)^4.$$

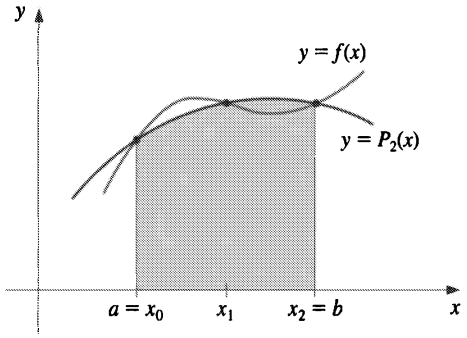

FIGURE 3 – Règle de Simpson.

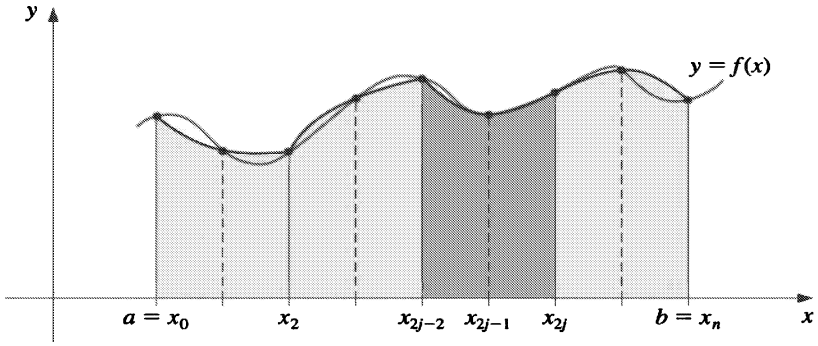

FIGURE 4 – Règle composée de Simpson.

Remarquer que $\frac{1}{24}\int_{x_0}^{x_2} f^{(4)}(\xi(x))(x-x_1)^4 dx = \frac{1}{120} f^{(4)}(\xi_1)(x-x_1)^5|_{x_0}^{x_2}$, pour un certain nombre $\xi_1 \in]x_0, x_2[$. Mais, $h = x_2 - x_1 = x_1 - x_0$, et

$$\begin{aligned}\int_{x_0}^{x_2} f(x)dx &= 2hf(x_1) + \frac{h^3}{3}\ddot{f}(x_1) + \frac{h^5}{60}f^{(4)}(\xi_1) \\ &= 2hf(x_1) + \frac{h^3}{3}\left\{\frac{1}{h^2}\left[f(x_0) - 2f(x_1) + f(x_2)\right] - \frac{h^2}{12}f^{(4)}(\xi_2)\right\} + \frac{h^5}{60}f^{(4)}(\xi_1)\end{aligned}$$

D'où

$$\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] - \frac{h^5}{90}f^{(4)}(\xi_1).$$

Règle composée de Simpson :
Si l'intervalle d'intégration $[a, b]$ est très large, on choisi un entier pair n. On divise l'intervalle $[a, b]$ en sous-intervalles et applique la règle de Simpson sur chaque consécutive paire de sous-intervalles. Voir la figure 4. Avec $h = (b-a)/n$ et $x_j = a + jh$, pour chaque $0 \leq j \leq n$, on

a

$$\int_a^b f(x)dx = \sum_{j=1}^{n/2} \int_{x_{2j}}^{x_{2j-2}} f(x)dx \tag{0.8}$$

$$= \sum_{j=1}^{n/2} \left\{ \frac{h}{3} \left[f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_{2j}) \right] - \frac{h^5}{90} f^{(4)}(\xi_j) \right\}, \tag{0.9}$$

avec $x_{2j-2} < \xi_j < x_{2j}$. Utilisant le fait que pour chaque $j = 1, 2, \ldots, (n/2) - 1$ on a $f(x_{2j})$ apparaissant dans le terme correspondant à l'intervalle $[x_{2j-2}, x_{2j}]$ et aussi dans le terme correspondant à l'intervalle $[x_{2j}, x_{2j+2}]$, on peut réduire cette somme à

$$\int_a^b f(x)dx = \frac{h}{3} \left[f(x_0) + 2 \sum_{j=1}^{(n/2)-1} f(x_{2j}) + 4 \sum_{j=1}^{(n/2)} f(x_{2j-1}) + f(x_n) \right] - \frac{h^5}{90} \sum_{j=1}^{(n/2)} f^{(4)}(\xi_j).$$

Algorithme de la règle composée de Simpson :
On veut déterminer l'approximation de l'intégrale $I = \int_a^b f(x)dx$:

— Entrée : les points extrêmes a, b; un entier positif pair n.
— Sortie : l'approximation XI à I.
— Etape 1 : Prendre $h = (b - a)/n$.
— Etape 2 : Prendre

$$XI0 = f(a) + f(b);$$
$$XI1 = 0 \quad (\text{somme de } f(x_{2j-1})$$
$$XI2 = 0 \quad (\text{somme de } f(x_{2j}).$$

— Etape 3 : Pour $i = 1, \ldots, n - 1$ faire
  1. Prendre $X = a + ih$
  2. Si i est pair, prendre $XI2 = XI2 + f(X)$,
     autrement prendre $XI1 = XI1 + f(X)$.
— Etape 4 : Prendre $XI = h * (XI0 + 2 * XI2 + 4 * XI1)/3$.
— Etape 5 : Sortie XI; Stop.

0.5.2 Equations d'ordres élevés et système d'équations différentielles

Supposons que le système d'ordre m du problème de valeur initiale a la forme :

$$\begin{aligned} \frac{dy_1}{dt} &= f_1(t, y_1, y_2, \ldots, y_m), \\ \frac{dy_2}{dt} &= f_2(t, y_1, y_2, \ldots, y_m), \\ &\vdots \\ \frac{dy_m}{dt} &= f_m(t, y_1, y_2, \ldots, y_m), \end{aligned} \tag{0.10}$$

pour $a \leq t \leq b$, avec les conditions initiales

$$y_1 = \alpha_1, y_2 = \alpha_2, \ldots, y_m = \alpha_m. \tag{0.11}$$

L'objectif est de trouver m fonctions $y_1, y_2, \ldots, y_m$ qui satisfont chacune des équations différentielles avec toutes les conditions initiales.

Définition 0.7 *La fonction* $f(t, y_1, y_2, \ldots, y_m)$ *définie sur l'ensemble*

$$D = \{(t, y_1, y_2, \ldots, y_m) | a \leq t \leq b, \text{ et } -\infty < y_i < \infty, \text{ pour } i = 1, 2, \ldots, m\},$$

satisfait une condition de Lipschitz sur D *avec les variables* $y_1, y_2, \ldots, y_m$, *si une constante* $L > 0$ *existe avec*

$$|f(t, y_1, y_2, \ldots, y_m) - f(t, z_1, z_2, \ldots, z_m)| \leq L \sum_{j=1}^{m} |y_j - z_j|, \tag{0.12}$$

pour tout $(t, y_1, y_2, \ldots, y_m)$ *et* $(t, z_1, z_2, \ldots, z_m)$ *dans* D.

Théorème 0.8 *Supposons que*

$$D = \{(t, y_1, y_2, \ldots, y_m) | a \leq t \leq b, \text{ et } -\infty < y_i < \infty, \text{ pour } i = 1, 2, \ldots, m\},$$

et prenons $f_i(t, y_1, y_2, \ldots, y_m)$, *pour* $i = 1, 2, \ldots, m$ *des fonctions continues satisfaisant une condition de Lipschitz sur* D *avec une constant de Lipschitz* L. *Le système du premier ordre différentielles équations (0.10) sujette aux conditions initiales (0.11), a une solution unique* $y_1, y_2, \ldots, y_m$, *pour* $a \leq t \leq b$.

Prenons un entier $N > 0$ $h = (b-a)/N$ tel que $t_j = a + jh$. La notation w_{ij} pour $j = 0, 1, \ldots, N$ et $i = 1, 2, \ldots, m$ l'approximation de $y_i(t_j)$. C'est à dire, w_{ij} est l'approximation de la $i^{\text{ième}}$ solution $y(t)$ de l'équation (0.10) au $j^{\text{ième}}$ point t_j. Pour les conditions initiales,

$$w_{1,0} = \alpha_1, w_{2,0} = \alpha_2, \ldots, w_{m,0} = \alpha_m. \tag{0.13}$$

Méthode du quatrième ordre de runge-Kutta pour les systèmes d'équations différentielles
Pour déterminer la solution du système d'ordre m du problème du premier ordre initiale valeur

$$\dot{y}_j = f_j(t, y_1, y_2, \ldots, y_m), \quad a \leq t \leq b, \quad \text{avec} \quad y_j = \alpha_j,$$

pour $j = 1, 2, \ldots, m$ aux $N + 1$ points également espacés sur l'intervalle $[a, b]$:

- Entrée : les points extrêmes ; le nombre d'équations m; l'entier N; les conditions initiales $\alpha_1, \alpha_2, \ldots, \alpha_m$.
- Sortie : Les approximations w_j à $y_j(t)$ aux $N + 1$ valeurs de t.
- Etape 1 : Prendre $h = (b - a)/N$; et $t = a$.
- Etape 2 : Pour $j = 1, 2, \ldots, m$ prendre $w_j = \alpha_j$.
- Etape 3 : Afficher $(t, w_1, w_2, \ldots, w_m)$.
- Etape 4 : Pour $i = 1, 2, \ldots, N$ faire (étapes 5-11.)
  - Etape 5 : Pour $j = 1, 2, \ldots, m$ prendre $k_{1,j} = hf_j(t, w_1, w_2, \ldots, w_m)$.
  - Etape 6 : Pour $j = 1, 2, \ldots, m$ prendre $k_{2,j} = hf_j(t+\frac{h}{2}, w_1+\frac{1}{2}k_{1,1}, w_2+\frac{1}{2}k_{1,2}, \ldots, w_m+\frac{1}{2}k_{1,m})$.
  - Etape 7 : Pour $j = 1, 2, \ldots, m$ prendre $k_{3,j} = hf_j(t+\frac{h}{2}, w_1+\frac{1}{2}k_{2,1}, w_2+\frac{1}{2}k_{2,2}, \ldots, w_m+\frac{1}{2}k_{2,m})$.
  - Etape 8 : Pour $j = 1, 2, \ldots, m$ prendre $k_{4,j} = hf_j(t+h, w_1+k_{3,1}, w_2+k_{3,2}, \ldots, w_m+k_{3,m})$.
  - Etape 9 : Pour $j = 1, 2, \ldots, m$ prendre $w_j = w_j + (k_{1,j} + 2k_{2,j} + 2k_{3,j} + k_{4,j})/6$.
  - Etape 10 : Prendre $t = a + ih$.
  - Etape 11 : Afficher $(t, w_1, w_2, \ldots, w_m)$. fin faire
- Etape 12 : Stop.

0.5.3 Zéros de polynômes et la méthode de Muller

Un polynôme de dégrée n a la forme

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0.$$

Où les a_is sont appelés les coefficients de $p(x)$, sont constants et $a_n \neq 0$.

Théorème 0.9 *(Théorème fondamental d'algèbre :) Si $p(x)$ est un polynôme de dégrée $n \geq 1$ avec des coefficients réels ou complexes, alors $p(x) = 0$ a au moins une racine (il est possible qu'elle soit complexe).*

Corollaire 0.10 *Si $p(x)$ est un polynôme de dégrée $n \geq 1$ avec des coefficients réels ou complexes, alors il existe des constants uniques $x_1, x_2, \ldots, x_k$, il est possible qu'ils soient complexe et des entiers positifs uniques $m_1, m_2, \ldots, m_k$, telle que $\sum_{i=1}^{k} m_i = n$ et*

$$p(x) = a_n (x - x_1)^{m_1} (x - x_2)^{m_2} \cdots (x - x_k)^{m_k}.$$

Corollaire 0.11 *Prenons $p(x)$ et $q(x)$ deux polynômes de dégrée au plus n. Si $x_1, x_2, \ldots, x_k$, avec $k > n$, sont des nombres distincts avec $p(x_i) = q(x_i)$ pour $i = 1, 2, \ldots, k$, alors $p(x) = q(x)$ pour toutes les valeurs de x.*