INSA Toulouse
Département STPI
2ème année POIC
UF Analyse Numérique
2014-2015

Analyse Numérique

Pascal Noble
Jean-Paul Vila

Département GMM (Génie Mathématique et Modélisation)
Adresses email et bureaux :
noble@insa-toulouse.fr, bureau 127
vila@insa-toulouse.fr, bureau 117

Table des matières

Chapitre 1

Résolution de systèmes linéaires

1.1 Motivation du problème

Considérons l'équation

$$-u''(x) = s(x), \quad \forall x \in [0, L]$$

où $L > 0$. On suppose en plus que $u(0) = u(L) = 0$. Ce problème peut modéliser le chauffage d'une barre métallique : S représente une source de chaleur alors que les conditions aux limites $u(0) = u(L) = 0$ représente le fait qu'on impose une température nulle aux extrémités. On ne peut en général calculer explicitement la solution d'un tel problème et on le remplace par un système discrétisé. Posons $h = L/(N+1)$ avec $N \in \mathbb{N}^*$ et notons $x_k = k\,h, k = 0, \ldots, N+1$. En pratique N est destiné à être grand. On cherche à approcher $(u(x_1), \ldots, u(x_N))$ par un vecteur $U = (U_1, \ldots, U_N)^T$. A partir de ce vecteur, on peut calculer une valeur approchée de $u''(x_k), k = 1, \ldots, N$:

$$u''(x_k) \approx \frac{U_{k+1} - 2\,U_k + U_{k-1}}{h^2}, \quad k = 1, \ldots, N.$$

Dans ce cas, U doit vérifier le sysème linéaire $AU = S$ où $S = (s(x_1), \ldots, s(x_N))^T$ où A et S désignent

$$A = h^{-2} \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix}, \quad S = \begin{pmatrix} s(x_1) \\ \vdots \\ s(x_N) \end{pmatrix}.$$

Outre le calcul effectif de la solution, on doit se demander l'influence d'une "erreur" (numérique, de mesure) dans la matrice A ou dans S sur la solution U. En effet, rien que sur la représentation du problème en machine, on commet une erreur numérique.

Dans ce chapitre, on commence par introduire des fonctions permettant de "mesurer" des vecteurs et des matrices. Ceci permettra de quantifier les erreurs numériques. On passera aux méthodes effectives, directes et indirectes, de résolution d'un système linéaire.

1.2 Normes et conditionnement

1.2.1 Normes sur $\mathbb{R}^d$

Définition 1.1. *On appelle norme sur $\mathbb{R}^d$ toute application $N : \mathbb{R}^d \to \mathbb{R}^+$ telle que*

1. $N(x) = 0 \quad \Leftrightarrow \quad x = 0$,
2. $N(\lambda\, x) = |\lambda|\, N(x), \quad \forall x \in \mathbb{R}^d, \quad \forall \lambda \in \mathbb{R}$,
3. $N(x + y) \leq N(x) + N(y), \quad \forall x \in \mathbb{R}^d, \quad \forall y \in \mathbb{R}^d$.

Les exemples suivants sont les plus élémentaires.

Exemple 1.2.
1. *La fonction valeur absolue de $\mathbb{R}$ dans $\mathbb{R}^+$ est une norme sur $\mathbb{R}$.*
2. *La fonction module de $\mathbb{C}$ dans $\mathbb{R}^+$ est une norme sur $\mathbb{C}$.*
3. *La fonction $|.|_2 : \mathbb{R}^2 \to \mathbb{R}^+$ définit par $|(x, y)|_2 = \sqrt{x^2 + y^2}$ est une norme.*

La propriété suivante permet de construire de nouvelles normes.

Proposition 1.3. *Soit N_1 une norme sur $\mathbb{R}^{d_1}$ et N_2 une norme sur $\mathbb{R}^{d_2}$. Notons $d = d_1 + d_2$. Alors les fonctions suivantes sont des normes sur R^d :*

1. $N^{(1)}\left((x, y)\right) = N_1(x) + N_2(y), \quad \forall x \in \mathbb{R}^{d_1}, \quad \forall y \in \mathbb{R}^{d_2}$.
2. $N^{\infty}\left((x, y)\right) = \max(N_1(x),\, N_2(y)), \quad \forall x \in \mathbb{R}^{d_1}, \quad \forall y \in \mathbb{R}^{d_2}$

Exemple 1.4. *Voici deux exemples de normes très importantes construites grâce à la proposition précédente et la valeur absolue en tant que norme sur $\mathbb{R}$.*

1. *$x \mapsto |x|_1 = \sum_{j=1}^{d} |x_j|$ est une norme sur $\mathbb{R}^d$*
2. *$x \mapsto |x|_\infty = \max_{j=1,\ldots,d}(|x_j|)$ est une norme sur $\mathbb{R}^d$*

Nous allons maintenant une dernière norme, dite norme euclidienne. Tout d'abord, commençons par introduire le produit scalaire sur $\mathbb{R}^d$. soit x et y deux vecteurs de $\mathbb{R}^d$, on dfinit le produit scalaire de x et y par

$$\langle x, y\rangle = \sum_{j=1}^{d} x_j\, y_j.$$

On définit alors la norme euclidienne par

$$|x|_2^2 = \langle x, x\rangle = \sum_{j=1}^{d} |x_j|^2.$$

Proposition 1.5. *La fonction $|.|_2 : \mathbb{R}^d \to \mathbb{R}^+$ définit bien une norme qui vérifie en plus les propriétés suivantes.*

1. $|x + y|_2^2 = |x|_2^2 + |y|_2^2 + 2\langle x, y\rangle, \quad \forall x \in \mathbb{R}^d,\ \forall y \in \mathbb{R}^d$
2. **(Inégalité de Cauchy Schwarz)** $|\langle x,\, y\rangle| \leq |x|_2\ |y|_2$
3. **(Égalité du parallèlogramme)** $|x|_2^2 + |y|_2^2 = \dfrac{1}{2}\left(|x + y|_2^2 + |x - y|_2^2\right)$

4. $x = 0 \quad \Leftrightarrow \quad \langle x, y\rangle = 0, \quad \forall y \in \mathbb{R}^d.$

De l'inégalité de Cauchy Schwarz, on déduit qu'il existe $\theta \in [0, 2\pi[$ tel que

$$\langle x,\, y\rangle = |x|_2\, |y|_2\, \cos(\theta)$$

L'angle θ représente l'angle entre les vecteurs x et y (dans le plan engendreé par x et y). A ce titre, on retrouve bien la propriété de Pythagore connue dans $\mathbb{R}^2$:

$$|x + y|_2^2 = |x|_2^2 + |y|_2^2 \quad \Leftrightarrow \quad \langle x, y\rangle = 0.$$

1.2.2 Normes sur $M_d(\mathbb{R})$

Passons à la définition de normes sur $M_d(\mathbb{R})$. En utilisant l'isomorphisme $\Psi : M_d(\mathbb{R}) \to \mathbb{R}^{d\times d}$ défini par

$$\Psi(M)_{i+(j-1)d} = M_{i,j}, \quad \forall i = 1, \dots, d, \quad \forall j = 1, \dots, d,$$

on peut construire naturellement des normes sur $M_d(\mathbb{R})$:

Exemple 1.6.

$$N_1(M) = \sum_{i=1}^{d}\sum_{j=1}^{d} |M_{i,j}|, \qquad N_\infty(M) = \max_{i,j=1,\dots,d} \left(|M_{i,j}|\right).$$

Une autre norme sur les matrices est inspirée de la norme euclidenne. En effet, on peut construire un produit scalaire entre matrice :

$$\langle A, B\rangle = \mathrm{Tr}(A\, B^T), \quad \forall A \in M_d(\mathbb{R}),\ \forall B \in M_d(\mathbb{R}).$$

La norme associée est appelée *norme de Frobenius*. Ces normes sont cependant peu utilisées car elles ne sont pas adaptées à la multiplication entre matrices. A la palce, on introduit des normes *induites*. Etant donnée une norme vectorielle, notée N, sur $\mathbb{R}^d$, on appelle norme subordonnée à N la fonction définie sur $M_d(\mathbb{R})$ par

$$\|A\|_N = \sup_{x\in\mathbb{R}^d\setminus\{0\}} \frac{N(Ax)}{N(x)} = \sup_{N(x)=1} N(Ax).$$

La norme subordonnée à N d'une matrice A est également caractérisée par

$$\|A\|_N = \inf\{M > 0 \ \mid\ N(Ax) \le M\, N(x), \quad \forall x \in \mathbb{R}^d\}$$

L'un des intérêts de telles normes est leur comportement vis à vis de la multiplication entre matrices.

Proposition 1.7. *Soit N une norme vectorielle sur $\mathbb{R}^d$ alors*

$$\|A\, B\|_N \le \|A\|_N\, \|B\|_N, \quad \forall A \in M_d(\mathbb{R}),\ \forall B \in M_d(\mathbb{R})$$

Voici deux exemples de normes subordonneés pour lesquelles on a une formule explicite.

Exemple 1.8. *Soit $A \in M_d(\mathbb{R})$ alors la norme de A subordonnée à la norme $|.|_\infty$ est donnée par*

$$\|A\|_\infty = \max_{1\leq i\leq d} \sum_{j=1}^{d} |A_{ij}|$$

La norme de A subordonnée à la norme $|.|_1$ est donnée par

$$\|A\|_1 = \max_{1\leq j\leq d} \sum_{i=1}^{d} |A_{ij}|.$$

1.2.3 Conditionnement d'une matrice

Soient $A \in M_d(\mathbb{R})$ et $b \in \mathbb{R}^d$. On considère le système d'équations linéaires à résoudre $Ax = b$. En pratique, la matrice A et le vecteur b sont entachés d'erreurs (numériques, d'approximation, de mesure,...). De fait, on est donc amené à résoudre le système linŕaire

$$(A + \delta A)(x + \delta x) = b + \delta b.$$

On peut s'interroger sur l'influence des erreurs δA et δb sur le résultat final. Considérons le problème, plus simple, d'une erreur sur le second membre $A(x + \delta x) = b + \delta b$. On a alors

$$Ax = b, \quad A\,\delta x = \delta b.$$

Soit N une norme vectorielle sur $\mathbb{R}^d$. On souhaite majorer la norme de δx. Comme $\delta x = A^{-1}\delta b$, on a

$$N(\delta x) = N(A^{-1}\delta b) \leq \|A^{-1}\|_N \; N(\delta b).$$

Bien entendu, la norme de l'erreur absolue a peu de sens et on souhaite une information sur l'erreur relative $N(\delta x)/N(x)$. Comme $b = Ax$, on obtient $N(b) \leq \|A\|_N \, N(x)$. En multipliant les deux inégalités, on obtient l'estimation voulue

$$\frac{N(\delta x)}{N(x)} \leq \|A\|_N \|A^{-1}\|_N \, \frac{N(\delta b)}{N(b)}.$$

Ainsi, l'erreur relative sur b est multipliée par un facteur $\|A\|_N \, \|A^{-1}\|_N$ appelé conditionnement de la matrice A relative à la norme N. On note $\mathrm{cond}_N(A) = \|A\|_N \, \|A^{-1}\|_N$. Le conditionnement d'une matrice est toujours supérieur à 1. En effet, puisqu'on manipule des normes subordonnées, on a

$$1 = \|\mathrm{Id}_{\mathbb{R}^d}\|_N = \|A \; A^{-1}\|_N \leq \|A\|_N \, \|A^{-1}\|_N = \mathrm{cond}_N(A), \quad \forall A \in M_d(\mathbb{R}).$$

Le conditionnement intervient également pour évaluer une l'influence d'une perturbation de la matrice A. Pour le problème perturbé complet, $(A + \delta A)(x + \delta x) = b + \delta b$, on peut montrer l'estimation suivante :

$$\frac{N(\delta x)}{N(x)} \leq \frac{\mathrm{cond}_N(A)}{1 - \mathrm{cond}_N(A)\dfrac{\|\delta A\|_N}{\|A\|_N}} \left(\frac{N(\delta b)}{N(b)} + \frac{\|\delta A\|_N}{\|A\|_N} \right).$$

Le conditionnement d'une matrice mesure donc l'amplification de l'erreur relative lorsqu'on résout un système linéaire. Plus le conditionnement est grand, pus la résolution du système est sensible aux erreurs. Voici un exemple simple de problème mal conditionné.

Exemple 1.9. *Soit $\varepsilon > 0$ un réel. Considérons la matrice A donnée par*

$$A = \begin{pmatrix} 1 & 1+\varepsilon \\ 1 & 1 \end{pmatrix}$$

Considérons ensuite les deux systèmes linéaires

$$Ax = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \qquad A(x+\delta x) = \begin{pmatrix} 1+\varepsilon \\ 1 \end{pmatrix}.$$

L'erreur relative sur le second membre est clairement d'ordre ε. On a $x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ et $\delta x = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$. Si on choisit la norme infinie, l'erreur relative commise est égale à 1 ! (soit 100% d'erreur !). L'amplification de l'erreur est dûe au mauvais conditionnement de la matrice, ici d'ordre ε^{-1}.

1.3 Résolution directe de systèmes linéaires

1.3.1 Méthode du pivot de Gauss

Résolution de systèmes triangulaires

Considérons un système linéaire de la forme $Ax = b$ où $A \in M_d(\mathbb{R})$ est telle que $A_{ij} = 0$ pour tout $i > j$. On dit que le système est *triangulaire*. Dans ce cas, la résolution se fait par *remontée*. Voici un exemple de résolution par remontée

Exemple 1.10. *Considérons le système*

$$\begin{aligned} 4\,x_1 + \; 5\,x_2 + \; x_3 &= \; 4, \\ 3\,x_2 + \; 4\,x_3 &= \; 1 \\ 2\,x_3 &= \; 2 \end{aligned}$$

De la dernière équation, on déduit que $x_3 = 1$. Ensuite, on injecte cette valeur de x_3 dans la deuxième équation et on déduit $x_2 = -1$. Enfin, à l'aide de la première équation, on obtient $x_1 = 2$.

Résolution de systèmes quelconques

La stratégie du pivot de Gauss consiste à transformer le système de départ en un système triangulaire qu'on résout ensuite par remontée. Voici un exemple de mise en oeuvre d'une telle stratégie.

Exemple 1.11. *Considérons le système*

$$\begin{aligned} 2\,x_1 \quad + x_2 \quad + x_3 \quad &= 1 \\ 6\,x_1 \quad + 2\,x_2 \quad + x_3 \quad &= -1 \\ -2\,x_1 \quad + 2\,x_2 \quad + x_3 \quad &= 7 \end{aligned}$$

Le premier coefficient de la première ligne est non nul : on va s'en servir comme "pivot". On va conserver la première ligne et éliminer la variable x_1 des lignes 2 et 3. On remplace la ligne L_2 par $\tilde{L}_2 = L_2 - \frac{6}{2}L_1$ et L_3 par $\tilde{L}_3 = L_3 - \frac{-2}{2}L_1$. Notons qu'à chaque étape, on a divisé par le "pivot".

$$\begin{array}{rrrr} 2\,x_1 & +x_2 & +x_3 & = 1 \\ & -\,x_2 & -2\,x_3 & = -4 \\ & 3\,x_2 & +2\,x_3 & = 8 \end{array}$$

Enfin, on élimine la variable x_2 de la dernière ligne : on remplace la ligne L_3 par $L_3 - \frac{3}{-1}L_2$. On obtient ainsi le système

$$\begin{array}{rrrr} 2\,x_1 & +x_2 & +x_3 & = 1 \\ & -\,x_2 & -2\,x_3 & = -4 \\ & & -4\,x_3 & = -4 \end{array}$$

On est donc ramené à résoudre un système triangulaire par remontée.

Voici un autre exemple, pour lequel on doit permuter des lignes au cours des calcul.

Exemple 1.12. *Considérons le système*

$$\begin{array}{rrrrr} x_1 & +x_2 & +x_3 & +x_4 & = 1 \\ x_1 & +x_2 & +3\,x_3 & +3\,x_4 & = 3 \\ x_1 & +x_2 & +2\,x_3 & +3\,x_4 & = 3 \\ x_1 & +3\,x_2 & +x_3 & +3\,x_4 & = 4 \end{array}$$

On remplace les lignes $L_i, i = 2, 3, 4$ par $\tilde{L}_i = L_i - L_1$. On obtient

$$\begin{array}{rrrrr} x_1 & +x_2 & +x_3 & +x_4 & = 1 \\ & & 2\,x_3 & +2\,x_4 & = 2 \\ & & x_3 & +2\,x_4 & = 2 \\ & 2\,x_2 & +2\,x_3 & +2\,x_4 & = 3 \end{array}$$

On observe que l'inconnue x_2 a été éliminée de la deuxième ligne : on ne peut donc appliquer directement la méthode du pivot de Gauss. On permute les lignes L_2 et L_4.

$$\begin{array}{rrrrr} x_1 & +x_2 & +x_3 & +x_4 & = 1 \\ & 2\,x_2 & +2\,x_3 & +2\,x_4 & = 3 \\ & & x_3 & +2\,x_4 & = 2 \\ & & 2\,x_3 & +2\,x_4 & = 2 \end{array}$$

On est ramené à one situation standard : on termine en remplaçant la ligne L_4 par $\tilde{L}_4 = L_4 - 2L_3$:

$$\begin{array}{rrrrr} x_1 & +x_2 & +x_3 & +x_4 & = 1 \\ & 2\,x_2 & +2\,x_3 & +2\,x_4 & = 3 \\ & & x_3 & +2\,x_4 & = 2 \\ & & & -2\,x_4 & = -2 \end{array}$$

1.3.2 Interprétation matricielle et factorisation LU

On a vu dans la section précédente qu'on faisait des manipulations entre lignes pour transformer un système linéaire en un système triangulaire. Ces manipulations ont une interprétation matricielle. Notons $\mathcal{L}_{ij}(\lambda) = \mathrm{Id}_{\mathbb{R}^d} + \lambda E_{ij}$ pour tout $i > j$. Ce sont des matrices *triangulaires inférieures*. Soit $A \in M_d(\mathbb{R})$ alors la matrice $A^{(1)} = \mathcal{L}_{ij}(\lambda)A$ est la matrice A dans laquelle on a remplacé la ligne L_i de A par $\tilde{L}_i = L_i + \lambda L_j$. Il est très facile d'inverser la matrice $\mathcal{L}_{ij}(\lambda)$ puisque $\mathcal{L}_{ij}(\lambda)^{-1} = \mathcal{L}_{ij}(-\lambda)$.

Revenons à l'exemple précédent. Dans ce cas, la matrice A est donnée par

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 6 & 2 & 1 \\ -2 & 2 & 1 \end{pmatrix}$$

La première étape revient matriciellement à considérer

$$A^{(1)} = \mathcal{L}_{31}(-\frac{-2}{2})\mathcal{L}_{21}(-\frac{6}{2})A = \begin{pmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 3 & 2 \end{pmatrix}$$

La deuxième étape revient matriciellement à

$$A^{(2)} = \mathcal{L}_{32}(-\frac{3}{-1})A^{(1)} = \begin{pmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & -4 \end{pmatrix} := U.$$

La matrice U est triangulaire supérieure (U pour "upper"). On revient à la matrice A de la manière suivante :

$$A = \mathcal{L}_{32}(\frac{3}{-1})\mathcal{L}_{31}(\frac{-2}{2})\mathcal{L}_{21}(\frac{6}{2})U.$$

Posons $L = \mathcal{L}_{32}(\frac{3}{-1})\mathcal{L}_{31}(\frac{-2}{2})\mathcal{L}_{21}(\frac{6}{2})$, on a alors factorisé la matrice A sous la forme $A = LU$. La matrice L est triangulaire inférieure (L pour "lower") avec des 1 sur la diagonale et U est triangulaire supérieure. Une fois cette factorisation effectuée, la résolution du système linéaire $Ax = b$ se réduit à la résolution de deux systèmes triangulaires

$$Ly = b, \qquad Ux = y.$$

Une autre conséquence intéressante de la factorisation LU d'une matrice est le calcul effectif du déterminant d'une matrice puisque si on a $A = LU$, U triangulaire supérieure et L triangulaire inférieure avec des 1 sur la diagonale alors

$$\det(A) = \det(L)\det(U) = \Pi_{j=1}^{d} U_{jj}$$

Voici un cadre général pour lequel on peut toujours faire une factorisation LU

Proposition 1.13. *Soit $A = (a_{ij})_{1\leq i,j\leq d} \in M_d(\mathbb{R})$ si pour tout k, $1 \leq k \leq d$ la sous-matrice principale $A_k = (a_{ij})_{1\leq i,j\leq k}$ de A telle que $\det(A_k) \neq 0$, il existe une matrice triangulaire inférieure L dont les éléments diagonaux sont égaux à 1 et une matrice triangulaire supérieure inversible U telle que $A = LU$. De plus cette factorisation est unique.*

Evaluons le coût de la factorisation LU. A l'étape k, on doit faire $d-k$ divisions pour le calcul de la k-ième colonne de L puis, pour la mise à jour de la matrice A, on doit faire $(d-k)^2$ multiplications et $(d-k)^2$ soustractions. Le coût de calcul de la factorisation s'obtient en sommant pour k variant de 1 à $d-1$:

$$C(d) = \frac{1}{2}d(d-1) + \frac{2}{3}d(d-\frac{1}{2})(d-1) = d(d-1)\frac{4d-1}{6}.$$

On a donc $C(d) \sim \dfrac{2\,d^3}{3}$ opérations élémentaires pour la factorisation. Ce coût de calcul est bien en deça du coût de calcul à l'aide des formules de Cramer qui nécessite le calcul de $d+1$ déterminants de matrice de taille d soit un coût de l'ordre de $(d+1)\,d!$.

Lorsqu'en cours de calcul, on est obligé de permuter des lignes, on a un résultat plus faible. Pour l'énoncer, on a besoin d'introduire la notion de *matrice de permutation.*

Définition 1.14. *Une matrice $P \in M_d(\mathbb{R})$ est dite matrice de permutation s'il existe une bijection σ de $[1,d]$ dans $[1,d]$ telle que pour tout $i \in [1,d]$, on ait $P\,e_i = e_{\sigma(i)}$.*

Proposition 1.15. *Soit $A \in M_d(\mathbb{R})$ une matrice inversible, il existe une matrice de permutation P, une matrice triangulaire inférieure L dont les éléments diagonaux sont égaux à 1 et une matrice triangulaire supérieure inversible U telle que $PA = LU$.*

1.3.3 Le cas particulier des matrices symétriques définies positives

Définition 1.16. *Soit $A \in M_d(\mathbb{R})$. La matrice A est dite symétrique et définie positive si et seulement si*

1. $A = A^T$
2. $\forall x \in \mathbb{R}^d, \quad x \neq 0 \;\Rightarrow x^T A x > 0$

Dans ce cas, on a une factorisation de A de type factorisation LU. On a le théorème suivant

Proposition 1.17. *Soit A une matrice symétrique définie positive : il existe une matrice triangulaire inférieure L telle que $A = LL^T$. La factorisation est unique si on impose les coefficients diagonaux de L strictement positifs.*

Démonstration. On obtient la factorisation directement par identification

$$A_{ij} = \sum_{k=1}^{\min(i,j)} L_{ik}L_{jk}.$$

La matrice A étant symétrique, on peut supposer $i \leq j$. Si $i = j = 1$, on a $L_{11}^2 = A_{11}$. On pose donc $L_{11} = \sqrt{A_{11}}$. Ensuite, pour $i = 1$ et $j = 2, \ldots, d$, on a

$$A_{1j} = L_{11}L_{j1} \quad \Rightarrow \quad L_{j1} = \frac{A_{1j}}{L_{11}}.$$

Ceci achève la détermination de la première colonne de L en au plus d opérations. En considérant la colonne i, on a

$$L_{ii}^2 = A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2, \quad L_{ji} = \frac{1}{L_{ii}} \left(A_{ij} - \sum_{k=1}^{i-1} L_{ik} L_{jk} \right).$$

Le coût total de la factorisation est de l'ordre de $\frac{d^3}{3}$ soit la moitié du coût de la factorisation LU d'une matrice.

Remarque 1.18. *Un grand nombre de problèmes se ramène à la résolution d'un système linéaire de la forme $Ax = b$ avec A symétrique et définie positive d'où l'intérêt de méthodes de factorisations adaptées. Le problème présenté dans l'introduction est de ce type.*

1.4 Méthodes indirectes de résolution

Les méthodes indirectes de résolution de systèmes linéaires reposent sur la construction d'une suite $(x_k)_{k\in\mathbb{N}}$ de $\mathbb{R}^d$ qui converge vers la solution $\bar{x}$ de $Ax = b$ avec $A \in M_d(\mathbb{R})$ et $b \in \mathbb{R}^d$. La construction et la convergence de telles suites reposent sur le théorème du point fixe

Proposition 1.19. *Soit $f : \mathbb{R}^d \to \mathbb{R}^d$ une fonction k-lipschitzienne avec $k \in]0, 1[$. Alors pour tout $x_0 \in \mathbb{R}^d$, la suite $(x_n)_{n\in\mathbb{N}}$ définie par récurrence $x_{n+1} = f(x_n)$ converge vers $\bar{x}$, l'unique solution de $f(\bar{x}) = \bar{x}$. De plus, on a l'estimation*

$$\|x_n - \bar{x}\| \leq \frac{k^n}{1-k} \|x_1 - x_0\|, \quad \forall\, n \in \mathbb{N}$$

Démonstration. Montrons que la suite $(x_n)_{n\in\mathbb{N}}$ est de Cauchy. D'abord, par récurrence, on montre que

$$\|x_{n+1} - x_n\| = \|f(x_n) - f(x_{n-1})\| \leq k\|x_n - x_{n-1}\| \leq \cdots \leq k^n \|x_1 - x_0\|.$$

Soit $p, q \in \mathbb{N}$ tels que $p > q$. On a

$$\|x_p - x_q\| = \|\sum_{n=q}^{p-1} x_{n+1} - x_n\| \leq \sum_{n=q}^{p-1} \|x_{n+1} - x_n\| \leq \sum_{n=q}^{p-1} k^n \|x_1 - x_0\| \leq \frac{k^q}{1-k} \|x_1 - x_0\|.$$

Lorsque $q \to \infty$, on a $k^q \to 0$, on en déduit que la suite $(x_n)_{n\in\mathbb{N}}$ est de Cauchy. L'espace $\mathbb{R}^d$ est complet donc la suite $(x_n)_{n\in\mathbb{N}}$ converge. On note $\bar{x}$ sa limite. Dans l'inégalité précédente, si on passe à la limite $p \to \infty$, on obtient l'estimation voulue

$$\|\bar{x} - x_q\| \leq \frac{k^q}{1-k} \|x_1 - x_0\|, \quad \forall q \in \mathbb{N}.$$

1.4.1 Méthode du Gradient

La méthode du gradient permet de résoudre des systèmes linéaires de la forme $Ax = b$ où A est une matrice réelle, symétrique, définie positive. L'objectif est de trouver l'unique minimum de la fonction $J : x \in \mathbb{R}^d \mapsto \frac{1}{2}\langle Ax,\, x\rangle - \langle b,\, x\rangle$. En effet, on a la propriété suivante

Proposition 1.20. *Soit $A \in M_d(\mathbb{R})$ une matrice symétrique et définie positive alors*

$$A\bar{x} = b \qquad \Longleftrightarrow \qquad J(y) \geq J(\bar{x}),\ \ \forall y \in \mathbb{R}^d$$

Démonstration. Supposons que $\bar{x}$ réalise le minimum de J alors pour tout $h \in \mathbb{R}^d$ et $t \in \mathbb{R}$, on a $J(\bar{x} + t\,h) \geq J(\bar{x})$. On obtient ainsi

$$J(\bar{x} + t\,h) = J(\bar{x}) + t\langle A\bar{x} - b,\ h\rangle + \frac{t^2}{2}\langle Ah,\, h\rangle \geq J(\bar{x}).$$

On en déduit alors que pour tout $t \in \mathbb{R}$ et $h \in \mathbb{R}^d$, on a

$$t\langle A\bar{x} - b,\ h\rangle + \frac{t^2}{2}\langle Ah,\, h\rangle \geq 0.$$

En passant à la limite $t \to 0^+$ et $t \to 0^-$, on trouve

$$\langle A\bar{x} - b,\, h\rangle = 0, \quad \forall h \in \mathbb{R}^d.$$

Donc $A\bar{x} = b$. Réciproquement, supposons que $A\bar{x} = b$. Soit $y \in \mathbb{R}^d$, on a

$$J(y) = J(\bar{x} + y - \bar{x}) = J(\bar{x}) + \langle A\bar{x} - b,\, y - \bar{x}\rangle + \frac{1}{2}\langle A(y - \bar{x},\, y - \bar{x}\rangle = J(\bar{x}) + \frac{1}{2}\langle A(y - \bar{x}),\, y - \bar{x}\rangle.$$

Donc $\bar{x}$ est un minimum global de la fonction J et il est unique car la matrice A est gymŐtriquŐ et définie positive. Ceci achève la démonstration de la proposition.
On a remplacé un problème de rśolution de système linŕaire par celui consistant à chercher le minimum d'une fonction. La méthode suivante est appelée *méthode du gradient*. Soit $x_0 \in \mathbb{R}^d$, on cherche à déterminer x_1 tel que $J(x_1) < J(x_0)$. On a

$$J(x_1) = J(x_0 + x_1 - x_0) = J(x_0) + \langle Ax_0 - b,\, x_1 - x_0\rangle + \frac{1}{2}\langle A(x_1 - x_0),\, x_1 - x_0\rangle$$

Le dernier terme de l'égalité est toujours positif. Pour être sur que $J(x_1) < J(x_0)$, il faut rendre le second terme négatif. Pour cela, on fait le choix

$$x_1 = x_0 - \mu_1(Ax_0 - b),$$

où μ_1 est un réel strictement positif dont on doit fixer la valeur. Avec ce choix, on obtient

$$\begin{aligned} J(x_1) &= J(x_0) - \mu_1\|Ax_0 - b\|^2 + \frac{\mu_1^2}{2}\langle A(Ax_0 - b),\, Ax_0 - b\rangle, \\ J(x_1) &= J(x_0) - \mu_1\|Ax_0 - b\|^2\left(1 - \frac{\mu_1}{2}\frac{\langle A(Ax_0 - b),\, Ax_0 - b\rangle}{\|Ax_0 - b\|^2}\right). \end{aligned}$$

Si $\rho_1 < 2/\rho(A)$ où $\rho(A)$ désigne la plus grande valeur propre de A alors, on est sur que $J(x_1) < J(x_0)$. On remarque que $x_1 - x_0$ est colinéaire à $Ax_0 - b$ qui est exactement le gradient de la fonction J au point x_0, d'où le nom de la méthode. L'algorithme s'écrit ainsi :

$$x_0 \in \mathbb{R}^d \text{ arbitraire}, \quad x_{n+1} = x_n - \mu_n (Ax_n - b), \quad \mu_n \in]0, 2/\rho(A)[.$$

Examinons la convergence de l'algorithme et notons $\bar{x}$ la solution de $A\bar{x} = b$. Alors, on a

$$x_{n+1} - \bar{x} = x_n - \bar{x} - \mu_n(Ax_n - A\bar{x}).$$

On en déduit alors que

$$\begin{aligned} &\|x_{n+1} - \bar{x}\|^2 = \|x_n - \bar{x}\|^2 - 2\mu_n\langle A(x_n - \bar{x}), x_n - \bar{x}\rangle + \mu_n^2\|A(x_n - \bar{x})\|^2, \\ &\|x_{n+1} - \bar{x}\|^2 \leq \|x_n - \bar{x}\|^2 \left(1 - 2\mu_n\sigma(A) + \mu_n^2\rho(A)^2\right), \end{aligned}$$

où $\sigma(A)$ désigne la plus petite valeur propre de A. On montre alors facilement la propriété suivante :

Proposition 1.21. *Si $\mu_n = \mu_0 \in]0, 2\dfrac{\sigma(A)}{\rho(A)^2}[$ pour tout $n \in \mathbb{N}$, alors l'algorithme du gradient converge. Pour $\mu_0 = \sigma(A)/\rho(A)^2$, alors on a la majoration*

$$\|x_{n+1} - \bar{x}\|^2 \leq \left(\frac{\kappa(A)^2 - 1}{\kappa(A)^2}\right) \|x_n - \bar{x}\|^2, \quad \kappa(A) = \frac{\rho(A)}{\sigma(A)}.$$

On peut montrer que $\kappa(A)$ est exactement le conditionnement de la matrice A pour la norme $\|.\|_2$. Ainsi l'algorithme du gradient converge d'autant plus lentement que le conditionnement de la matrice A est grand.

Notons qu'on peut améliorer la convergence de la méthode du gradient en adoptant la méthode du gradient à pas optimal : dans ce cas, une fois la direction de descente $d_n = Ax_n - b$ donnée, il faut choisir μ_n optimal de telle sorte que

$$J(x_{n+1}) = \min_{\mu \in \mathbb{R}} J(x_n + \mu(Ax_n - b)).$$

1.4.2 Problème des Moindres carrés

Voici un exemple pour lequel la méthode du gradient peut s'appliquer, celui de l'approximation au sens des moindres carrés. Prenons un exemple modèle. Supposons qu'on souhaite reconstituer un signal dont on connait la forme :

$$f(x) = a_1 \sin(\frac{\pi x}{2}) + a_2 \sin(\frac{2\pi x}{3}) + a_3 \sin(\pi x),$$

mais pas les coefficients $a_i, i = 1, 2, 3$. On dispose d'une série de mesures, données dans le tableau ci-dessous

x_i	0.5	1	1.5	2	2.5
y_i	1.6914	0.4523	-0.8934	-0.3332	0.2332

Le système d'équations $f(x_i) = y_i, i = 1, \cdots, 5$ s'écrit $A\mathbf{a} = y$ avec

$$A = \begin{pmatrix} 0.7071 & 0.5000 & 1.0000 \\ 1.0000 & -0.5000 & 0.0000 \\ 0.7071 & -1.0000 & -1.0000 \\ 0.0000 & -0.5000 & -0.0000 \\ -0.7071 & 0.5000 & 1.0000 \end{pmatrix}, \quad y = \begin{pmatrix} 1.6914 \\ 0.4523 \\ -0.8934 \\ -0.3332 \\ 0.2332 \end{pmatrix}, \quad \mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Bien évidemment, on ne peut résoudre exactement ce problème car il y'a plus d'équations ($f(x_i) = y_i, i = 1, \ldots, 5$) que d'inconnues ($a_j, j = 1, 2, 3$). A la place, on cherche à minimiser la fonction $J(\mathbf{a}) = \|A\mathbf{a} - y\|^2$. Supposons que $\bar{\mathbf{a}}$ réalise le minimum de la fonction J. Pour tout $t \in \mathbb{R}$ et pour tout $h \in \mathbb{R}^3$, on a $J(\bar{\mathbf{a}} + t\,h) \geq J(\bar{\mathbf{a}})$:

$$J(\bar{\mathbf{a}} + t\,h) = \|A\bar{\mathbf{a}} - y + t\,Ah\|^2 = J(\bar{\mathbf{a}}) + 2t\langle A\bar{\mathbf{a}} - y,\, Ah\rangle + t^2\|Ah\|^2.$$

On en déduit que pour tout $t \in \mathbb{R}$ et pour tout $h \in \mathbb{R}^3$,

$$2t\langle A\bar{\mathbf{a}} - y,\, Ah\rangle + t^2\|Ah\|^2 \geq 0.$$

En passant à la limite $t \to 0^+$ et $t \to 0^-$, on obtient

$$\langle A\bar{\mathbf{a}} - y,\, Ah\rangle,\ \forall h \in \mathbb{R}^d \quad \Longleftrightarrow \quad A^T A\bar{\mathbf{a}} = A^T y.$$

Le problème des moindres carrés se réduit donc à la résolution d'un système linéaire dont la matrice $A^T A$ est symétrique et, en général, définie positive (sinon cela signifie que certaines équations du problème sont redondantes). On peut donc appliquer la méthode du gradient. Dans l'exemple traité, le système linéaire à résoudre s'écrit :

$$\begin{pmatrix} 2.5000 & -1.2071 & -0.7071 \\ -1.2071 & 2.0000 & 2.0000 \\ -0.7071 & 2.0000 & 3.0000 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 0.8517 \\ 1.7961 \\ 2.8180 \end{pmatrix}$$

Chapitre 2

Résolution d'équations non linéaires

Dans ce chapitre, on s'intéresse à la résolution *approchée* d'équations de la forme $f(x) = 0$ où f est une fonction définie sur un intervalle $I \subset \mathbb{R}$ et à valeur dans $\mathbb{R}$. Sauf cas particuliers (par exemple lorsque f est un polynôme du second degré), on ne sait pas résoudre explicitement ces équations : la stratégie consiste alors à construire une suite de nombres (réels ou complexes) qui vont converger vers une solution de l'équation.

2.1 Localisation de racines et méthode de dichotomie

Avant de calculer la valeur approchée d'une solution, il faut commencer par *localiser* la solution en question. L'outil principal pour montrer l'existence d'une solution à l'équation $f(x) = 0$ où $f : I \subset \mathbb{R} \to \mathbb{R}$ est le théorème des valeurs intermédiaires dont voici un énoncé adapté à notre probème.

Proposition 2.1. *Soit $a, b \in \mathbb{R}$ tels que $a < b$ et $f : [a, b] \to \mathbb{R}$ une fonction continue sur $[a, b]$. On suppose que $f(a)f(b) < 0$: alors il existe au moins un $x_0 \in]a, b[$ tel que $f(x_0) = 0$.*

Cette proposition ne permet pas de parler a priori de "la" solution de léquation $f(x) = 0$ sur $[a, b]$ puisqu'on n'a pas nécessairement unicité.

Exemple 2.2. *Soit $f : [-2, 2] \to \mathbb{R}$ telle que $f(x) = x(x^2-1)$, $\forall x \in [-2, 2]$. On a $f(-2)f(2) = -36 < 0$ donc f admet au moins une racine sur $[-2, 2]$. En fait elle en admet trois : $0, 1$ et -1.*

Pour avoir unicité, il faut rajouter une hypothèse de monotonie sur la fonction f.

Proposition 2.3. *Soit $a, b \in \mathbb{R}$ tels que $a < b$ et $f : [a, b] \to \mathbb{R}$ une fonction continue et strictement monotone sur $[a, b]$. On suppose que $f(a)f(b) < 0$: alors il existe un unique $x_0 \in]a, b[$ tel que $f(x_0) = 0$.*

Pour localiser toutes les solutions de l'équation $f(x) = 0$, il faut donc commencer par dresser le tableau de variation de la fonction f. Une fois ce travail effectué, on est ramené au cadre de la proposition précédente. On suppose dorénavant que $f : [a, b] \to \mathbb{R}$ est strictement croissante et continue sur $[a, b]$ et est telle que $f(a)f(b) < 0$. La proposition précédente est à la base de la méthode de dichotomie.

Notons $a_0 = a$ et $b_0 = b$. Introduisons le point milieu $c = \dfrac{a_0 + b_0}{2}$. Si $f(a_0)f(c) < 0$ alors nécessairement la solution $x_0 \in [a, b]$ telle que $f(x_0) = 0$ vérifie nécessairement $x_0 \in [a_0, c]$. On pose alors $a_1 = a_0$ et $b_1 = c$. Sinon, on a nécessairement $x_0 \in [c, b_0]$: dans ce cas, on pose $a_1 = c$ et $b_1 = b_0$. Dans les deux cas, on a $x_0 \in [a_1, b_1]$. On répète ce procédé par récurrence : supposons avoir contruit a_n et b_n tels que $x_0 \in [a_n, b_n]$. On pose $c = \dfrac{a_n + b_n}{2}$. Si $f(a_n)f(c) < 0$, on pose $a_{n+1} = a_n$ et $b_{n+1} = c$. Sinon, on pose $a_{n+1} = c$ et $b_{n+1} = b_n$.

Par récurrence, on montre que

- La suite $(a_n)_{n\in\mathbb{N}}$ est croissante, la suite $(b_n)_{n\in\mathbb{N}}$ est décroissante
- $|b_n - a_n| = \dfrac{b-a}{2^n}, \quad \forall n \in \mathbb{N}$
- $x_0 \in [a_n, b_n], \quad \forall n \in \mathbb{N}$

Les suites $(a_n)_{n\in\mathbb{N}}$ et $(b_n)_{n\in\mathbb{N}}$ sont adjacentes donc convergent vers une limite $l = x_0$. De plus, si on se sert de $(a_n)_{n\in\mathbb{N}}$ comme de la suite approchant x_0, on a la majoration

$$|a_n - x_0| \leq \frac{b-a}{2^n}, \quad \forall n \in \mathbb{N}.$$

Cet algorithme converge assez lentement. En voici une illustration sur un exemple.

Exemple 2.4. *Soit $f : [1, 2] \to \mathbb{R}$ définie par $f(x) = x^2 - 2$. L'équation $f(x) = 0$ possède une solution unique, $x_0 = \sqrt{2} \in [1, 2]$. En appliquant la méthode de dichotomie, on construit $(a_n)_{n\in\mathbb{N}}$ telle que*

$$|a_n - \sqrt{2}| \leq \frac{1}{2^n}, \quad \forall n \in \mathbb{N}.$$

Si on souhaite une valeur approchée à 10^{-16} près, il faut choisir n tel que $2^{-n} \leq 10^{-16}$ soit

$$n \geq \frac{16\ln(10)}{\ln(2)} \approx 53.15$$

On verra dans la suite un algorithme permettant d'avoir la même précision en 5 ou 6 itérations.

2.2 Méthode du point fixe

Une méthode pour résoudre une équation de la forme $f(x) = 0$ où $f : I \subset \mathbb{R} \to \mathbb{R}$ et de la remplacer par un problème de point fixe. On rappelle ici le théorème du point fixe, la démonstration étant donnée dans le chapitre 1.

Proposition 2.5. *Soit $I = \mathbb{R}$ ou $I = [a, b]$ avec $a < b$ deux réels. Soit f une application de I dans $\mathbb{R}$ telle que*

- *$f(I) \subset I$,*
- *il existe une constante $k \in]0, 1[$ telle que f est k-lipschitzienne*

$$\forall (x, y) \in I^2, \quad |f(x) - f(y)| \leq k|x - y|.$$

Alors pour tout $x_0 \in I$, la suite $(x_n)_{n\in\mathbb{N}}$ définie par $x_{n+1} = f(x_n),\ \forall n \in \mathbb{N}$ converge vers $l \in I$ l'unique point fixe de f (i.e. tel que $f(l) = l$). De plus, on a la majoration

$$|x_n - l| \leq \frac{k^n}{1-k}|x_1 - x_0|, \quad \forall n \in \mathbb{N}.$$

Nous passons maintenant à la formulation du problème $f(x) = 0$ sous la forme d'un problème de point fixe. Commençons par remarquer que pour tout $\lambda \in \mathbb{R}^*$,

$$f(x) = 0 \quad \Longleftrightarrow \quad x = x + \lambda\, f(x) = g(x, \lambda).$$

Dans le cas où on peut appliquer le théorème du point fixe à $g(., \lambda)$, on obtient ainsi un algorithme pour obtenir une solution de $f(x) = 0$:

$$a_0 \ \text{ arbitraire}, \quad a_{n+1} = a_n + \lambda\, f(a_n), \quad \forall n \in \mathbb{N}.$$

En utilisant l'inégalité des accroissements finis, on peut montrer le résultat de convergence de la suite $(a_n)_{n\in\mathbb{N}}$ vers x_0 solution de $f(x_0) = 0$.

Proposition 2.6. *Soit $f : [a, b] \to \mathbb{R}$ une fonction de classe $\mathcal{C}^1$. On suppose qu'il existe $k(\lambda) \in]0, 1[$ tel que pour tout $x \in [a, b], |1 + \lambda\, f'(x)| \leq k(\lambda)$ alors la suite $(a_n)_{n\in\mathbb{N}}$ converge vers $x_0 \in [a, b]$ tel que $f(x_0) = 0$. De plus, on a l'estimation*

$$|a_n - x_0| \leq \frac{k(\lambda)^n}{1 - k(\lambda)}|a_1 - a_0|, \quad \forall n \in \mathbb{N}.$$

2.3 Méthode de Newton et de la sécante

Notons qu'on a un degré de liberté dans le choix du λ dans la méthode du point fixe et qu'on pourrait tout aussi bien considérer un algorithme du point fixe de la forme

$$a_0 \ \text{ arbitraire}, \quad a_{n+1} = a_n + \lambda_n\, f(a_n), \quad \forall n \in \mathbb{N}.$$

Cet algorithme converge dès lors que les λ_n sont choisis de telle sorte que

$$|1 + \lambda_n\, f'(x_n)| \leq k < 1, \quad \forall n \in \mathbb{N}, \quad \forall x \in [a, b].$$

La convergence est d'autant plus rapide que k est petit. Un choix "optimal" serait donc de prendre $\lambda_n = -\dfrac{1}{f'(a_n)}$. On obtient ainsi la méthode de Newton :

$$a_0 \ \text{ arbitraire}, \qquad a_{n+1} = a_n - \frac{f(a_n)}{f'((a_n)}, \quad \forall n \in \mathbb{N}.$$

L'algorithme de Newton converge *localement* : cela nécessite que a_0 soit suffisamment proche de la solution x_0 de $f(x) = 0$. Voici un résultat de convergence de la méthode de Newton.

Proposition 2.7. *Soit $f : [a, b] \to \mathbb{R}$ une fonction de classe $\mathcal{C}^2$. On suppose que x_0 est une racine simple de $f(x) = 0$ (ce qui signifie que $f(x_0) = 0$ et $f'(x_0) \neq 0$). Alors il existe $\varepsilon > 0$ telle que la suite définie par*

$$a_{n+1} = a_n - \frac{f(a_n)}{f'(a_n)}, \quad \forall n \in \mathbb{N}, \quad |a_0 - x_0| \leq \varepsilon$$

converge vers x_0. De plus, il existe $C > 0$ tel que

$$|a_{n+1} - x_0| \leq C|a_n - x_0|^2, \quad \forall n \in \mathbb{N}.$$

Démonstration. On a

$$a_{n+1} - x_0 = a_n - x_0 - \frac{f(a_n)}{f'(a_n)} = \frac{(a_n - x_0)f'(a_n) - f(a_n)}{f'(a_n)}$$

En faisant un dénveloppement deTaylor de f en a_n à l'ordre 2, on obtient

$$0 = f(x_0) = f(a_n) + (x_0 - a_n)f'(a_n) + (x_0 - a_n)^2 \frac{f''(\theta_n)}{2},$$

pour un θ_n compris entre x_0 et a_n. On en déduit que

$$|a_{n+1} - x_0| \leq \frac{|f''(\theta_n)|}{2|f'(a_n)|}|a_n - x_0|^2, \quad \forall n \in \mathbb{N}.$$

La fonction f est de classe $\mathcal{C}^2$ donc il existe M tel que $|f''(x)| \leq M$, $\forall x \in [a, b]$. Quitte à restreindre l'intervalle $[a, b]$, on peut supposer qu'il existe m tel que $|f'(x)| > 1/m$, $\forall x \in [a, b]$. On obtient ainsi

$$|a_{n+1} - x_0| \leq |a_n - x_0|^2 \frac{mM}{2}.$$

En choisissant $\varepsilon < 2/(Mm)$, on montre par récurrence que si $|a_0 - x_0| \leq \varepsilon$ alors $|a_n - x_0| \leq \varepsilon$, $\forall n \in \mathbb{N}$. De plus, on a

$$|a_{n+1} - x_0| \leq \frac{\varepsilon\, mM}{2}|a_n - x_0|, \quad \forall n \in \mathbb{N}$$

Donc la suite $(a_n)_{n\in\mathbb{N}}$ converge vers x_0 et la convergence est *quadratique* grâce à l'estimation $|a_{n+1} - x_0| \leq C|a_n - x_0|^2$, $\forall n \in \mathbb{N}$ (avec $C = mM/2$).

Exemple 2.8. *Considérons l'équation $x^2 - 2 = 0$. On cherche à calculer une valeur approchée de $x_0 = \sqrt{2}$. On montre sans difficulté que $x_0 \in [1.4, 1.5]$. La méthode de Newton s'écrit*

$$a_{n+1} = a_n - \frac{a_n^2 - 2}{2a_n} = \frac{1}{2}\left(a_n + \frac{2}{a_n}\right).$$

On a

$$a_{n+1} - \sqrt{2} = a_n - \sqrt{2} - \frac{a_n^2 - 2}{2a_n} = \frac{(a_n - \sqrt{2})^2}{2a_n}.$$

Si on choisit $a_0 = 3/2$ alors la suite $(a_n)_{n\in\mathbb{N}}$ est décroissante et minorée par $\sqrt{2}$ donc converge vers $\sqrt{2}$. On en déduit l'estimation

$$a_{n+1} - \sqrt{2} \leq (a_n - \sqrt{2})^2, \quad \forall n \in \mathbb{N}.$$

La convergence est très rapide : on a $a_0 - \sqrt{2} \leq 10^{-1}$ donc $a_1 - \sqrt{2} \leq 10^{-2}$, $a_2 - \sqrt{2} \leq 10^{-4}$ et ainsi de suite. A la quatrième itération, on atteint la précision de 10^{-16} !

L'hypothèse $f'(x_0) \neq 0$ dans la proposition précédente est cruciale pour obtenir la convergence quadratique de la méthode de Newton. Voici un exemple où la méthode de Newton converge mais pas quadratiquement.

Exemple 2.9. *Si on souhaite résoudre l'équation $x^2 = 0$ à l'aide de la méthode de Newton, l'algorithme s'écrit*

$$x_0 \text{ arbitraire}, \quad x_{n+1} = x_n - \frac{x_n^2}{2x_n} = \frac{x_n}{2}.$$

On a donc $x_n = \dfrac{x_0}{2^n}$ donc la suite converge bien vers la racine double $x_0 = 0$ mais la convergence est linéaire.

La principale limitation de la méthode de Newton est le caractère local de la convergence : il faut en général commencer par une méthode de dichotomie pour réduire la taille de l'intervalle de recherche de la racine puis passer à la méthode de Newton lorsque la convergence est garantie. On peut cependant donner un résultat de convergence globale.

Proposition 2.10. *Si $f(x_0) = 0, f'(x_0) > 0$ et si $f'' \geq 0$ sur $[x_0, b]$ alors pour tout $a_0 \in [x_0, b]$ la suite de la méthode de Newton est définie, décroissante, minorée par x_0 et converge vers x_0.*

Interprétation graphique. On peut réécrire l'algorithme de Newton sous la forme

$$f(a_n) + f'(a_n)(a_{n+1} - a_n) = 0.$$

Donc a_{n+1} est le zéro de la fonction affine $x \mapsto f(a_n) + f'(a_n)(x - a_n)$ qui est exactement l'équation de la tangente à la courbe représentant f au point a_n. Ainsi la méthode de Newton consiste, à chaque étape, à remplacer la fonction f par sa tangente et à trouver la racine de la fonction affine.

La dérivée de la fonction f peut être difficile à calculer : dans ce cas, on peut remplacer la méthode de Newton par la méthode de la sécante :

$$a_0, a_1 \text{ arbitraires}, \quad a_{n+1} = a_n - \frac{a_n - a_{n-1}}{f(a_n) - f(a_{n-1})} f(a_n), \quad \forall n \in \mathbb{N}.$$

Comme pour la méthode de Newton, la convergence est seulement locale :

Proposition 2.11. *Soit $f : [a, b] \to \mathbb{R}$ une fonction de classe $\mathcal{C}^2$ et $x_0 \in]a, b[$ tel que $f(x_0) = 0$ et $f'(x_0) \neq 0$. Alors il existe $\varepsilon > 0$ tel que pour tout $a_{,0}, a_1 \in [x_0 - \varepsilon, x_0 + \varepsilon]$, la suite $(a_n)_{n\in\mathbb{N}}$ définie par*

$$a_{n+1} = a_n - \frac{a_n - a_{n-1}}{f(a_n) - f(a_{n-1})} f(a_n), \quad \forall n \in \mathbb{N}$$

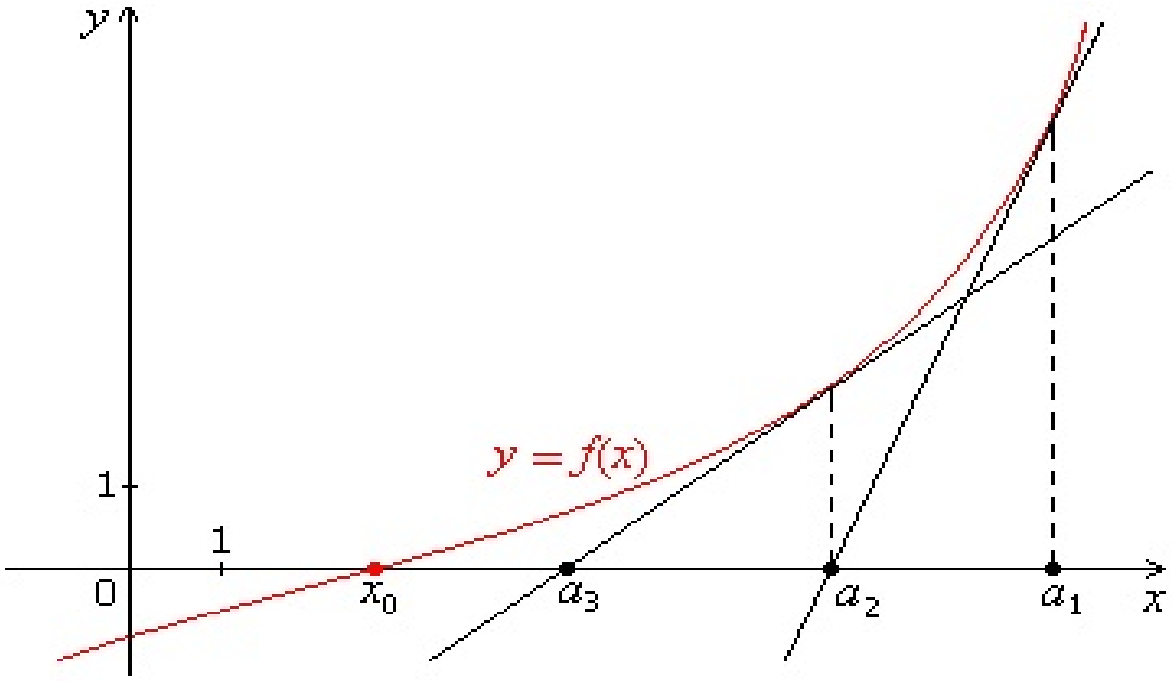

FIGURE 2.1 – Illustration graphique de la méthode de Newton

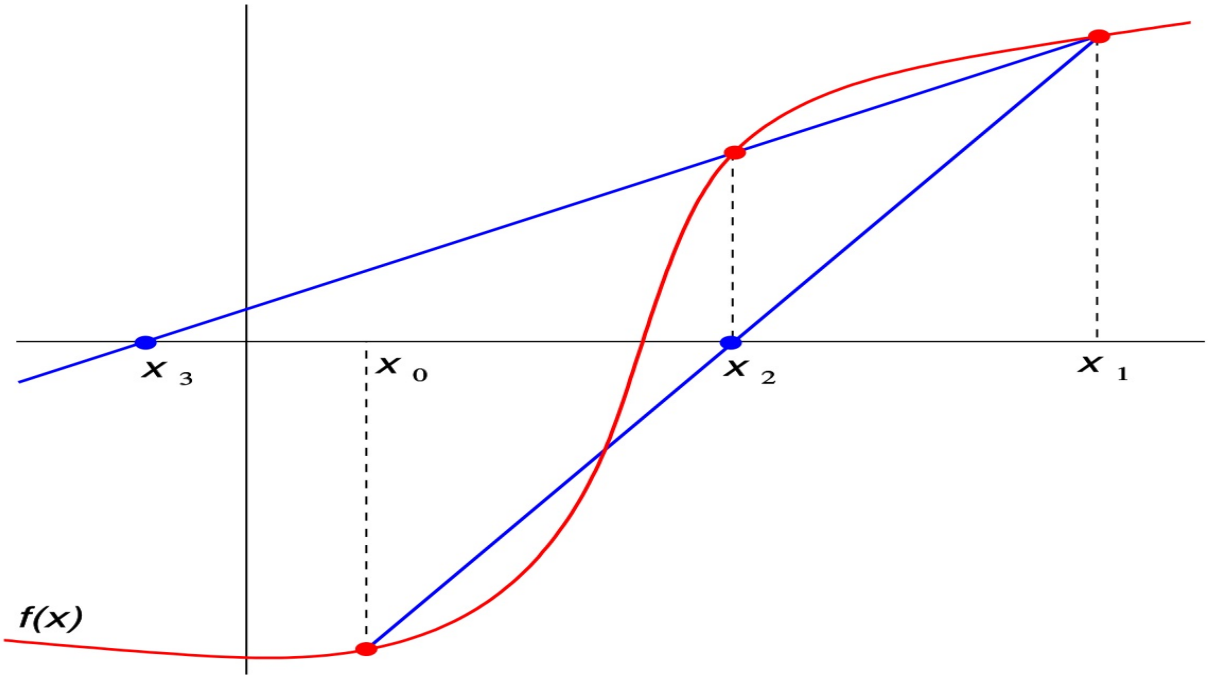

FIGURE 2.2 – Illustration graphique de la méthode de la sécante

converge vers x_0*.De plus, il existe* $C > 0$ *tel que*

$$|a_{n+1} - x_0| \leq C|a_n - x_0|^r, \quad r = \frac{1+\sqrt{5}}{2}.$$

L'ordre de convergence de la méthode de la sécante est égale au nombre d'or : la convergence est donc moins bonne que pour la méthode de Newton mais reste plus rapide qu'une méthode de point fixe.

2.4 Le cas des équations polynomiales

Dans cette section, on s'intéresse à la résolution d'équations polynomiales de la forme

$$p(x) = 0, \quad \text{avec} \qquad p(x) = a_0\, x^n + a_1\, x^{n-1} + \cdots + a_n, \quad a_0, \cdots, a_n \in \mathbb{C}.$$

On peut localiser les racines du polynôme p à l'aide du théorème suivant

Proposition 2.12. *Toutes les racines $(\xi_i)_{i=1,\cdots,n} \in \mathbb{C}$ du polynôme p vérifient*

$$|\xi_i| \leq \max\left(\left|\frac{a_n}{a_0}\right|, 1 + \left|\frac{a_{n-1}}{a_0}\right|, \ldots, 1 + \left|\frac{a_1}{a_0}\right|\right), \quad \forall i = 1, \ldots, n$$
$$|\xi_i| \leq \max\left(1, \sum_{j=1}^{n} \left|\frac{a_j}{a_0}\right|\right), \quad \forall i = 1, \ldots, n$$

Pour les polynômes réels, un tableau de variation permet de localiser plus précisëment les racines *réelles* et on peut alors appliquer la méthode de Newton. Pour le calcul de la plus grand racine réelle, on a le théorème suivant :

Proposition 2.13. *Soit p un polynôme à coefficients réels de degré $n \geq 2$. On suppose que toutes les racines de p, notées $\xi_i, i = 1, \ldots, n$ sont réelles et telles que $\xi_i \geq \xi_{i+1}, \forall i = 1, \ldots, n-1$ alors la méthode de Newton :*

$$x_0 > \xi_1, \quad x_{k+1} = x_k - \frac{p(x_k)}{p'(x_k)}, \quad \forall k \in \mathbb{N}$$

converge vers ξ_1 et la suite $(x_k)_{k\in\mathbb{N}}$ est décroissante.

Pour évaluer les polynômes p et p', on utilise l'algorithme de Horner. Pour évaluer p en $x = \xi$, on factorise $p(\xi)$ sous la forme

$$\begin{aligned} p(\xi) &= a_0\xi^n + a_1\xi^{n-1} + \cdots + a_n = (a_0\xi + a_1)\xi^{n-1} + a_2\xi^{n-2} + \cdots + a_n, \\ &= ((a_0\xi + a_1)\xi + a_2)\,\xi^{n-2} + a_3\xi^{n-3} + \cdots + a_n. \end{aligned}$$

L'algorithme de Horner s'écrit $b_0 = a_0, \quad b_i = b_{i-1}\xi + a_i, \quad \forall i = 1, \ldots n$ et $p(\xi) = b_n$. Les valeurs b_i sont aussi les coefficients d'un polynôme p_1 tel que

$$p(x) = (x - \xi)p_1(x) + b_n, \quad p_1(x) = b_0x^{n-1} + b_1x^{n-2} + \cdots + b_{n-1}.$$

On montre sans difficulté que $p'(\xi) = p_1(\xi)$: l'algorithme de Horner permet donc d'évaluer rapidement $p(\xi)$ et $p'(\xi)$ simultanément et donc implémenter efficacement la méthode de Newton.

Si la plus grande racine du polynôme p est très grande (en valeur absolue), la convergence de la méthode de Newton peut être lente. En effet, on a

$$x_{k+1} = x_k - \frac{p(x_k)}{p'(x_k)} \approx x_k - \frac{x_k^n}{n\, x_k^{n-1}} = (1 - \frac{1}{n})x_k.$$

Pour accélerer la convergence, on utilise une méthode à pas double. L'algorithme de Newton modifié s'écrit ainsi

$$x_0 \ \text{ arbitraire}, \quad x_{k+1} = x_k - 2\frac{p(x_k)}{p'(x_k)}, \ \forall k \in \mathbb{N}.$$

Contrairement à la méthode de Newton, la suite n'est plus nécessairement minorée pas x_0 car il y'a un risuqe de "dépasser" la plus grande racine. On peut cependent détecter ce moment (et à partir de là mettre en place une méthode de Newon simple) grâ ce à la proposition suivante (dont on admettra la démonstration).

Proposition 2.14. *Soit p un polynome à coefficients réels dont les racines $(\xi_i)_{i=,\dots,n}$ sont réelles et telles que*

$$\xi_1 \geq \xi_2, \cdots, \geq \xi_n.$$

On note $a_1 \in [\xi_1, \xi_2]$ la plus grande racine de p'. Pour tout $z > \xi_1$, les nombres suivants

$$z' = z - \frac{p(z)}{p'(z)}, \quad y = z - 2\frac{p(z)}{p'(z)}, \quad y' = y - \frac{p(y)}{p'(y)}$$

sont bien définis et vérifient $a_1 < y$ et $\xi_1 \leq y' \leq z'$.

Donc soit la suite $(x_k)_{k\in\mathbb{N}}$ converge vers ξ_1 en décroissant et plus rapidement que la méthode de Newton classique soit il existe k_0 tel que $y = x_{k_0} < \xi_1$. D'après le théorème précédent, on a cependant $y > a_1$ et $x_{k_0+1} = x_{k_0} - \dfrac{p(x_{k_0})}{p'(x_{k_0})} > \xi_1$. Ainsi à partir de $k = k_0$, on utilise la méthode de Newton simple.

Une fois calculée la plus grande des racines du polynôme p, on souhaite obtenir la suivante. Notons $\tilde{\xi_1} \approx \xi_1$ la valeur approchée de ξ_1. Théoriquement, il suffirait de considérer la polynôme p_1 défini par

$$p_1(x) = \frac{p(x)}{x - \xi_1}, \quad \forall x \in \mathbb{R},$$

la valeur $x = \xi_2$ étant alors la plus grande racine de p_1. Cependant, on a accès qu'à un polynôme approché $\tilde{p}_1(x) = p(x)/(x - \tilde{\xi_1})$ et la plus grande racine devient $\tilde{\xi_2} \approx \xi_2$. Si on souhaite poursuivre pour calculer toutes les racines de p, il y'a un risque important de cumuler les erreurs d'arrondis. Pour éviter cela, on introduit une méthode alternative : on calcule explicitement la dérivée de p_1 :

$$p_1'(x) = \frac{p'(x)}{x - \xi_1} - \frac{p(x)}{(x - \xi_1)^2}$$

La méthode de Newton pour p_1 s'écrit alors

$$x_{k+1} = x_k - \frac{p_1(x_k)}{p_1'(x_k)} = x_k - \frac{p(x_k)}{p'(x_k) - \dfrac{p(x_k)}{x_k - \xi_1}} = \Psi_1(\xi_k).$$

L'avantage d'un tel algorithme est qu'il converge (localement) vers ξ_2 et de manière quadratique même si ξ_1 n'est pas une racine de p ! Pour calculer les autres racines, on introduit l'algorithme

$$x_{k+1} = x_k - \frac{p(x_k)}{p'(x_k) - \sum_{i=1}^{j} \frac{p(x_k)}{x_k - \xi_i}} = \Psi_j(x_k).$$

Cette suite converge (localement) et quadratiquement vers ξ_{j+1}.

Exemple 2.15. *Mettre en oeuvre les méthodes exposées sur le polynôme* $p(x) = \Pi_{j=0}^{7}(x - 2^{-j})$.

2.5 Recherche de valeurs propres

2.5.1 Méthode de la puissance

Soit $A \in M_d(\mathbb{C})$: on cherche à calculer numériquement les valeurs propres de la matrice A. Ce sont les valeurs $\lambda \in \mathbb{C}$ telles que $A - \lambda \mathrm{Id}_{\mathbb{C}^d}$ ne soit pas injective : il existe $X \in \mathbb{C}^d$ non nul tel que

$$AX = \lambda\, X$$

On peut relier ce problème à la recherche des racines du *polynôme caractéristique*

$$P(\lambda) = \det(A - \lambda\, \mathrm{Id}_{\mathbb{C}^d}) = 0$$

Pour la localisation des valeurs propres d'une matrice, on a la proposition suivante.

Proposition 2.16. *Soit* $A \in M_d(\mathbb{C})$ *L'ensemble des valeurs propres de* A *est inclus dans* $\bigcup_{j=1}^{d} D_j$ *avec*

$$D_j = \left\{ z \in \mathbb{C} \ / \ |z - A_{jj}| \leq \sum_{i \neq j} |A_{ji}| \right\}.$$

Comme pour les polynômes, on a une méthode efficace, différente de la méthode de Newton, pour calculer la plus grande valeur propre en module de A ainsi qu'un vecteur propre associé : c'est la *méthode de la puissance*.

Soit $v \in \mathbb{C}^d$: on introduit la suite

$$v_0 = \frac{v}{\|v\|}, \qquad v_{n+1} = \frac{A\, v_n}{\|A v_n\|}, \quad \forall n \in \mathbb{N}.$$

On montre sans difficulté que $v_n = \dfrac{A^n\, v}{\|A^n\, v\|}$, $\quad \forall n \in \mathbb{N}$. Sous certaines conditions, la suite $(v_n)_{n\in\mathbb{N}}$ converge vers un vecteur propre associé à la plus grande valeur propre de A :

Proposition 2.17. *Soit $A \in M_d(\mathbb{C})$ telles que ses valeurs propres $\lambda_1, \ldots, \lambda_d$ vérifient*

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_d|.$$

Alors si v_0 possède une composante non nulle sur l'espace propre associé à λ_1 alors la suite $(v_n \wedge w_1)_{n\in\mathbb{N}}$ où w_1 est un vecteur propre associé à λ_1 converge vers 0 (v_n est de plus en plus colinéaire à w_1). Si λ_1 est réel, la suite $(v_n)_{n\in\mathbb{N}}$ converge. Dans tous les cas, on a

$$\lambda_1 = \lim_{n\to\infty} \frac{(Av_n)_1}{(v_n)_1}.$$

Démonstration On note $w_1, \ldots, w_d$ une base de vecteurs propres associées aux λ_i et on décompose v dans cette base :

$$v = \alpha_1 w_1 + \alpha_2 w_2 + \cdots + \alpha_d w_d.$$

En utilisant le fait que $A^n w_i = \lambda_i^n w_i, \quad \forall i = 1, \ldots, d$, on obtient

$$\begin{aligned} v_n &= \frac{\alpha_1\lambda_1^n w_1 + \alpha_2\lambda_2 + \cdots + \alpha_d\lambda_d^n w_d}{\|\alpha_1\lambda_1^n w_1 + \alpha_2\lambda_2 + \cdots + \alpha_d\lambda_d^n w_d\|} \\ &= \left(\frac{\lambda_1}{|\lambda_1|}\right)^n \frac{\alpha_1 w_1 + \alpha_2\left(\frac{\lambda_2}{\lambda_1}\right)^n w_2 + \cdots + \alpha_d\left(\frac{\lambda_d}{\lambda_1}\right)^n w_d}{\|\alpha_1 w_1 + \alpha_2\left(\frac{\lambda_2}{\lambda_1}\right)^n w_2 + \cdots + \alpha_d\left(\frac{\lambda_d}{\lambda_1}\right)^n w_d\|} \\ &= \left(\frac{\lambda_1}{|\lambda_1|}\right)^n \frac{\alpha_1 w_1}{\|\alpha_1 w_1\|} + \mathcal{O}\left((\frac{\lambda_2}{\lambda_1})^n\right). \end{aligned} \tag{2.1}$$

On en déduit alors aisément que $v_n \wedge w_1 = \mathcal{O}\left((\frac{\lambda_2}{\lambda_1})^n\right)$ tend vers 0 lorsque n tend vers $+\infty$. Si $\lambda_1 \in \mathbb{R}$, la suite $(v_n)_{n\in\mathbb{N}}$ converge au signe près et on a bien dans tous les cas

$$\lambda_1 = \lim_{n\to\infty} \frac{(Av_n)_1}{(v_n)_1}.$$

On remarque que la convergence est d'autant plus rapide que le rapport λ_2/λ_1 est petit. Les hypothèses sur le spectre de A du théorème précédent sont un peu restrictives. En fait il suffit qu'il y'ait une seule valeur propre de plus grand module et qu'elle soit simple mais la preuve de convergence est plus compliquée.

2.5.2 Méthode de la puissance inverse

Pour simplifier la discussion, on se place encore dans les hypothèses du théorème précédent, à savoir que les valeurs propres $\lambda_i \in \mathbb{C}$ de A vérifient

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_d|$$

On suppose en plus que A est inversible et donc $\lambda_d \neq 0$. Si on souhaite calculer λ_d, on commence par remarquer que les valeurs propres de A^{-1} sont λ_i^{-1}, $i = 1, \ldots, d$ et vérifient

$$\frac{1}{|\lambda_d|} > \frac{1}{|\lambda_{d-1}|} > \cdots > \frac{1}{|\lambda_1|}$$

On peut donc appliquer la méthode de la puissance à A^{-1} pour calculer λ_d^{-1}. L'algorithme s'écrit formellement pour tout $v \in \mathbb{C}^d$ arbitraire

$$v_0 = \frac{v}{\|v\|}, \qquad v_{n+1} = \frac{A^{-1}v_n}{\|A^{-1}v_n\|}.$$

En pratique, on n'inverse jamais la matrice A et la méthode de puissance inverse s'Őcrit

- v est donné et on pose $v_0 = \dfrac{v}{\|v\|}$
- On résout le système linéaire (par exemple avec une méthode de Gauss ou en faisant la décomposition LU de A au départ) $Ax_n = v_n$
- On pose $v_{n+1} = \dfrac{x_n}{\|x_n\|}$

Sous les hypothèses énoncées dans le théorème sur la convergence de la méthode de la puissance, on montre la convergence de la méthode de la puissance inverse.

La méthode de la puissance inverse a une extension intéressante : supposons qu'on connaisse une valeur approchée, notée τ, d'une valeur propre λ_j de A. On peut se servir de la méthode de la puissance inverse pour calculer une approximation plus précise que τ de λ_j. En effet, si on considère la matrice $A - \tau \operatorname{Id}_{\mathbb{C}^d}$, la plus petite valeur propre en valeur absolue est $\lambda_j - \tau$. L'algorithme pour trouver une valeur approchée de λ_j et un vecteur propre associé est donné par

- v est donné et on pose $v_0 = \dfrac{v}{\|v\|}$
- On résout le système linéaire (par exemple avec une méthode de Gauss) $Ax_n - \tau\, x_n = v_n$
- On pose $v_{n+1} = \dfrac{x_n}{\|x_n\|}$

Une autre manière d'interpréter cet algorithme est de dire qu'on recherche la valeur propre de A la plus proche du nombre (réel ou complexe) τ.

Chapitre 3

Interpolation et intégration numérique

3.1 Interpolation de Lagrange

3.1.1 Position du problème et première résolution

Soient $n \in \mathbb{N}$ et $x_0 = a < x_1 < \cdots < x_n = b$ une subdivision (pas nécessairement équirépartie) d'un intervalle $[a, b] \subset \mathbb{R}$. Etant donnés $y_0, y_1, \ldots, y_n$ des réels, on cherche un polynôme P tel que

$$P(x_i) = y_i, \quad \forall i = 1, \ldots, n$$

Si un tel polynome existe et est unique, on l'appelle le polynôme d'interoplation de Lagrange. Si $y_i = f(x_i)$, $i = 1, \ldots, n$ où f est une fonction de $[a, b] \to \mathbb{R}$ alors P est destiné à être une approximation de la fonction f sur l'intervalle $[a, b]$. Ceci est particulièrement utile puisque on n'a besoin de connaître que les coefficients du polynôme pour l'évaluer alors que ce n'est pas le cas pour une fonction quelconque. Cette approche sera également utilisée dans le calcul approché d'intégrales puisqu'on sait toujours calculer l'intégrale d'une fonction polynomiale : pour calculer la valeur approchée de l'intégrale d'une fonction quelconque, on approchera cette fonction par une fonction polynomiale par morceaux.

Comme il y'a $n + 1$ équations à satisfaire, il faut que P soit au moins de degré n Une approche naive consiste à rechercher le polynome sous la forme

$$P(x) = a_0 + a_1\, x + \cdots + a_n\, x^n.$$

La condition $P(x_i) = y_i$, $i = 1, \ldots, n$ donne lieu à la résolution du système linéaire

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

On peut se poser les questions suivantes : existe-t-il une solution à ce problème ? si oui, est-elle unique ? Autrement dit la matrice $V(x_0, \ldots, x_n)$ (aussi appelée matrice de Vandermonde)

donnée par

$$V(x_0,\dots,x_n)=\begin{pmatrix}1 & x_0 & x_0^2 & \cdots & x_0^n\\ 1 & x_1 & x_1^2 & \cdots & x_1^n\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & x_n & x_n^2 & \cdots & x_n^n\end{pmatrix}$$

est-elle inversible ? Pour les matrices, ceci est équivalent à montrer que le noyau de A est réduit à 0. On a la proppriété suivante

Proposition 3.1. *Si $x_0, x_1, \dots, x_n$ sont distincts alors la matrice $V(x_0,\dots,x_n)$ est inversible.*

Démonstration. On pourrait faire un calcul explicite du déterminant de $V(x_0, x_1, \dots, x_n)$ mais on peut obtenir le résultat plus rapidement. Montrons que le noyau de $V(x_0,\dots,x_n)$ est réduit à 0. S'il existe un vecteur $(a_0,\dots,a_n)^T$ dans le noyau de A alors on a

$$\begin{pmatrix}1 & x_0 & x_0^2 & \cdots & x_0^n\\ 1 & x_1 & x_1^2 & \cdots & x_1^n\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & x_n & x_n^2 & \cdots & x_n^n\end{pmatrix}\begin{pmatrix}a_0\\ a_1\\ \vdots\\ a_n\end{pmatrix}=\begin{pmatrix}0\\ 0\\ \vdots\\ 0\end{pmatrix}$$

Donc la fonction polynomiale $P : x \mapsto a_0 + a_1\, x + \cdots + a_n\, x^n$. est un polynome de degré n qui s'annule en $n+1$ points distincts, c'est donc le polynome nul. On en déduit que ses coefficients, $(a_i)_{i=0,n}$ sont tous nuls et donc le noyau de A est réduit à 0, ce qui donne l'inversibilité de la matrice $V(x_0,\dots,x_n)$.

3.1.2 Méthode de Newton de calcul du polynôme d'interpolation

Résoudre le système linéaire précédent pour calculer les coefficients du polynome d'interpolation est cependant coûteux : ceci vient du fait qu'on a exprimé le polynome d'interpolation dans la base $1, x, x^2, \dots, x^n$ qui n'est pas adapt'ee au problème. A la place, on va contruire une base des polynômes de degré inférieur ou égal à n adaptée au problème d'interpolation.

Notons $\mathbb{R}_n[x]$ l'ensemble des fonctions polynômiales de degré inférieur ou Ógal à n. C'est un espace vectoriel de dimension $n+1$. Considérons l'application linéaire $\mathcal{E} : \mathbb{R}_n[x] \to \mathbb{R}^{n+1}$ définie par

$$\mathcal{E}(P) = \left(P(x_0),\, P(x_1), \cdots, P(x_n)\right)^T.$$

On a déjà montré que cette application était injective et donc bijective car $\mathbb{R}_n[x]$ et $\mathbb{R}^{n+1}$ ont même dimension. L'espace $\mathbb{R}^{n+1}$ possède une base naturelle, la base canonique $(e_i)_{i=0,n}$ tel que $(e_i)_j = \delta_{ij}$ (où δ_{ij} désigne le symbole de Kronecker). L'application $\mathcal{E}$ étant un isomorphisme, si on pose $L_i = \mathcal{E}^{-1}(e_i)$, on obtient ainsi une base de $\mathbb{R}_n[x]$ autre que la base canonique. Déterminons explicitement les polynômes L_i. Ces derniers vérifient

$$L_i(x_j) = \delta_{ij}, \quad \forall i = 0, 1, \dots, n \quad \forall j = 0, 1, \dots, n$$

Si on fixe i, alors $(x - x_j)$, $j \neq i$ divise L_i donc il existe $\alpha_i \in \mathbb{R}$ tel que

$$L_i(x) = \alpha_i \prod_{j \neq i} (x - x_j), \quad \forall x \in \mathbb{R}.$$

De plus, comme on a $L_i(x_i) = 1$, on en déduit que

$$L_i(x) = \frac{\prod_{j \neq i}(x - x_j)}{\prod_{j \neq i}(x_i - x_j)}, \quad \forall x \in \mathbb{R}.$$

Etant donnés $y_0, y_1, \ldots, y_n$, il devient aisé d'obtenir le polynôme d'interpolation de Lagrange P tel que $P(x_i) = y_i$, $i = 0, 1, \ldots, n$: il suffit de remarquer que

$$P(x) = \sum_{i=0}^{n} y_i \, L_i(x), \quad \forall x \in \mathbb{R}$$

convient et, par unicité, c'est le seul possible. Cette solution, plus élégante que la méthode directe, ne permet cependant pas une évaluation rapide du polynôme d'interpolation de Lagrange : il faut pour cela introduire une autre base de polynômes adaptée au problème d'interpolation et à l'évaluation poncutelle. On introduit donc la famille de polynômes N_i définis par

$$N_0(x) = 1, \quad N_1(x) = x - x_0, \quad N_2(x) = (x - x_0)(x - x_1), \quad \ldots, \quad N_{i+1}(x) = \prod_{j=0}^{i} (x - x_j).$$

On cherche le polynôme d'interpolation P dans cette base : $P(x) = \sum_{k=0}^{n} \alpha_k \, N_k(x)$. Les conditions $P(x_i) = y_i$ donnent le système triangulaire

$$\begin{array}{llll}
\alpha_0 & & & = y_0 \\
\alpha_0 & +\alpha_1(x_1 - x_0) & & = y_1 \\
\alpha_0 & +\alpha_1(x_2 - x_0) & +\alpha_2(x_2 - x_1)(x_2 - x_0) & = y_2 \\
\vdots & \vdots & \vdots & = \vdots \\
\alpha_0 & +\alpha_1(x_n - x_0) & +\alpha_2(x_2 - x_1)(x_2 - x_0) \quad \cdots \quad +\alpha_n \prod_{j=0}^{n-1}(x_n - x_j) & = y_n
\end{array}$$

On résout ce système par remontée :

- La première ligne donne $\alpha_0 = y_0$.
- La deuxième donne $\alpha_1 = \dfrac{y_1 - y_0}{x_1 - x_0} := [y_0, y_1]$
- La troisième donne

$$\alpha_2 = \frac{y_2 - y_0 - \dfrac{x_2 - x_0}{x_1 - x_0}(y_1 - y_0)}{(x_2 - x_1)(x_2 - x_0)} = \frac{[y_1, y_2] - [y_0, y_1]}{x_2 - x_0} := [y_0, y_1, y_2]$$

- et ainsi de suite

$$\alpha_k = \frac{[y_1, y_2, \ldots, y_k] - [y_0, y_1, \ldots, y_{k-1}]}{x_k - x_0} := [y_0, y_1, y_2, \ldots, y_k].$$

Le polynôme d'interpolation s'écrit alors

$$P(x) = \sum_{k=0}^{n} [y_0, y_1, \ldots, y_k] \prod_{j=0}^{k-1} (x - x_j).$$

Cette forme est bien adaptée à une évaluation "à la Horner". L'évaluation des coefficients est très rapide et on peut montrer que les coûts de calcul associés aux trois méthodes présentées sont les suivantes :

- Méthode avec la matrice de Vandermonde : $C(n) \sim 2n^3/3$ opérations.
- Méthode avec les polynômes de Lagrange : $C(n) \sim n^2$ opérations.
- Méthode de Newton : $C(n) \sim n^2/2$ opérations.

3.1.3 Erreur d'interpolation

Soit $f : [a, b] \to \mathbb{R}$ une fonction "régulière" (c'est à dire de classe $\mathcal{C}^k$ avec $k \in \mathbb{N}$ assez grand). On souhaite estimer l'erreur commise en remplaçant la fonction f par son polynôme d'interpolation aux point $(x_i)_{i=0,1,\ldots,n}$. On note $E_n(x) = f(x) - P(x)$ l'erreur commise au point $x \in [a, b]$. On a la proposition suivante

Proposition 3.2. *Soit $f : [a, b] \to \mathbb{R}$ une fonction de classe $\mathcal{C}^{n+1}$ et $x_0 = a < x_1 < \cdots < x_{n-1} < x_n = b$ une subdivision de l'intervalle $[a, b]$ alors pour tout $x \in [a, b]$, il existe $\xi \in]a, b[$ tel que*

$$E_n(x) = f^{(n+1)}(\xi) \frac{\pi(x)}{(n+1)!}, \quad \pi(x) = \prod_{i=0}^{n} (x - x_i).$$

Démonstration. Soit $x \in]a, b[$. Si $x = x_i, i = 0, n$, n'importe quel $\xi \in]a, b[$ convient. Dans la suite, on suppose que $x \neq x_i, \;\; i = 0, 1, \ldots, n$. On introduit la fonction auxiliaire $F_x : [a, b] \to \mathbb{R}$ définie par

$$F_x(t) = E_n(t) - \frac{\pi(t)}{\pi(x)} E_n(x), \quad \forall t \in [a, b].$$

La fonction F_x s'annule aux points $t = x_i, i = 0, 1, \ldots, n$ et est de classe $\mathcal{C}^{n+1}$ donc en appliquant n fois le théorème de Rolle, on montre qu'il existe $\xi \in]a, b[$ tel $F_x^{(n+1)}(\xi) = 0$. En remplaçant par l'expression de F_x, on obtient le résultat voulu.

On déduit de ce théorème l'estimation globale :

$$\sup_{x \in [a,b]} |f(x) - P(x)| \leq \frac{\sup\limits_{y \in [a,b]} |f^{(n+1)}(y)|}{(n+1)!} \sup_{y \in [a,b]} |\pi(y)|.$$

La question est ensuite de savoir si le polynôme P approche bien la fonction f sur l'intervalle $[a, b]$ lorsqu'on prend de plus en plus de points $n \to \infty$. La résponse est délicate et dépend en particulier du choix des points d'interpolation.

Si on fait le choix, apparemment naturel, d'une subdivision équirépartie, on peut être amené à observer le phénomène d'oscillations de Runge aux bornes du domaines. Voici un exemple très

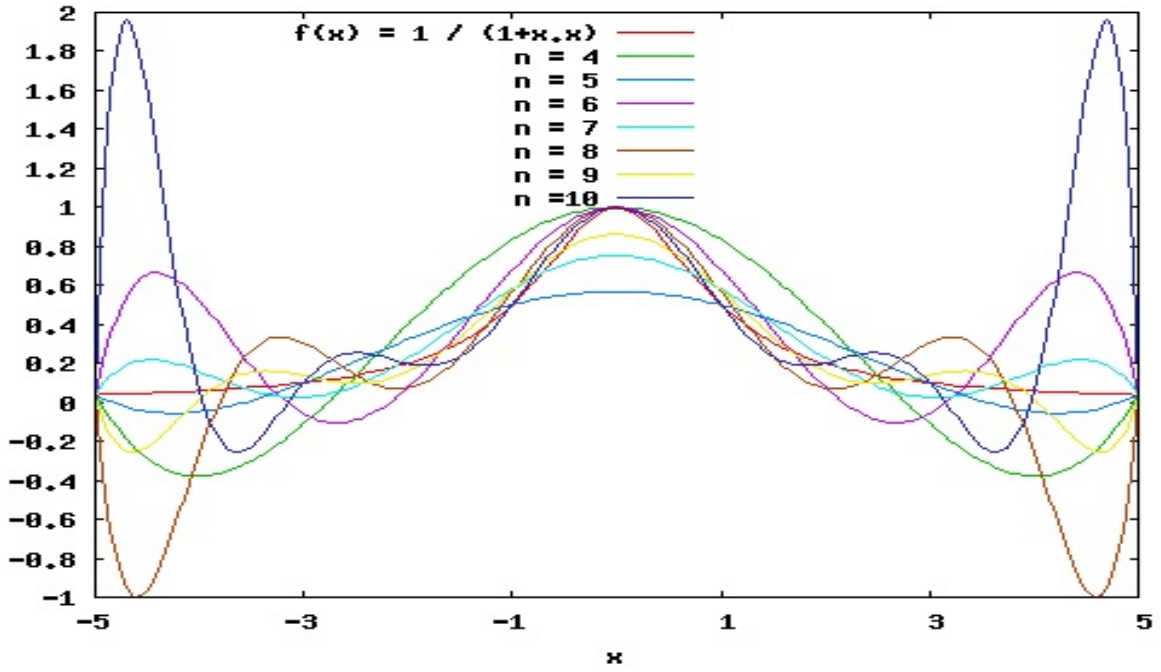

FIGURE 3.1 – Illustration du phénomène de Runge : lorsqu'on augmente le nombre de points, on a des oscillations de plus en plus fortes aux bords du domaine

connu : choisissons $f : [-5, 5] \to \mathbb{R}$ tel que $f(x) = \dfrac{1}{1+x^2}$. C'est une fonction très régulière, et pourtant, en choisissant une subdivision $x_j = -5 + j\frac{10}{n}$, $j = 0, 1, \ldots, n$, on obtient :
On peut résoudre ce problème en choisissant mieux les points d'interpolation. Le choix est guidé par le fait de minimiser $\sup_{y\in[a,b]} |\prod_{j=0}^{n}(y - x_j)|$. Le choix optimal est celui des points de Gauss, ici relatif à l'intervalle $[-5, 5]$, donnés par

$$x_k = -5\cos\left(\frac{\pi(2k+1)}{2(n+1)}\right), \quad k = 0, 1, \ldots, n.$$

Ces points sont en fait les racines des polynômes de Tchebychev de degré inférieur ou égal à n+1.

Une autre stratégie possible est de faire de l'interpolation avec des fonctions polynômiales par morceaux et en limitant le degré du polynôme dans chaque sous intervalle où la fonction est polynômiale. C'est la méthode mise en oeuvre en CAO où l'interpolation par des "splines cubiques" est utilisée.

3.2 Intégration numérique

L'objectif de cette section est de présenter les méthodes classiques de calcul approché d'intégrales de fonctions $f : [a, b] \to \mathbb{R}$. Selon le contexte, on supposera la fonction f de classe $\mathcal{C}^k$ avec $k \in \mathbb{N}$.

3.2.1 Sommes de Riemann et méthode des rectangles

Proposition 3.3. *Soit $f : [a,b] \to \mathbb{R}$ une fonction continue et $x_k = a + k\frac{b-a}{n}$, $k = 0, 1, \ldots, n$ alors*

$$\lim_{n\to\infty} \frac{b-a}{n} \sum_{k=1}^{n} f(x_k) = \int_a^b f(t)dt.$$

Démonstration. Soit $\varepsilon > 0$. La fonction $f : [a,b] \to \mathbb{R}$ est continue donc uniformément continue : il existe $\eta > 0$ tel que

$$\forall x, y \in [a,b],\ |x-y| \leq \eta \quad \Longrightarrow \quad |f(x) - f(y)| \leq \varepsilon.$$

En appliquant la relation de Chasles, on a

$$\begin{aligned}
\int_a^b f(t)dt &= \sum_{k=1}^{n} \int_{x_{k-1}}^{x_{k-1}} f(t)dt \\
&= \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} f(x_{k-1}) + (f(t) - f(x_{k-1}))\, dt \\
&= \frac{1}{n} \sum_{k=1}^{n} f(x_{k-1}) + \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} (f(t) - f(x_{k-1}))\, dt.
\end{aligned}$$

Ensuite en appliquant la formule de Taylor reste intégral à $F : t \mapsto \int_{x_{k-1}}^{t} f(s)ds$, on obtient

$$\begin{aligned}
\int_{x_{k-1}}^{x_k} f(t) - f(x_{k-1})dt &= F(x_k) - F(x_{k-1}) \\
&= F'(x_{k-1})(x_k - x_{k-1}) + (x_k - x_{k-1})^2 \int_0^1 (1-t)F''(x_{k-1} + t(x_k - x_{k-1}))dt \\
&= \frac{b-a}{n} f(x_{k-1}) + (x_k - x_{k-1})^2 \int_0^1 (1-t)f'(x_{k-1} + t(x_k - x_{k-1}))dt. \quad (3.1)
\end{aligned}$$

On en déduit que

$$\begin{aligned}
\left| \int_a^b f(t)dt - \frac{b-a}{n} \sum_{k=1}^{n} f(x_{k-1}) \right| &\leq \sum_{k=1}^{n} (x_k - x_{k-1})^2 \int_0^1 (1-t)|f'(x_{k-1} + t(x_k - x_{k-1}))|dt \\
&\leq (\frac{b-a}{n})^2 \sum_{k=1}^{n} \int_0^1 (1-t) \sup_{z\in[a,b]} |f'(z)|dt = \left(\frac{(b-a)}{2} \sup_{z\in[a,b]} |f'(z)| \right) \frac{b-a}{n}.
\end{aligned}$$

On conclut en faisant tendre n vers $+\infty$.

Ce résultat fournit une manière de calcul la valeur approchée d'une intégrale :

Principe du calcul approché d'une intégrale

– On coupe l'intervalle $[a, b]$ en n sous intervalle $[x_{k-1}, x_k]$, $k = 1, 2, \ldots, n$.
– Sur chaque sous intervalle, on remplace $\int_{x_{k-1}}^{x_k} f(t)dt$ par une valeur approchée, notée $I_k(f)$.
– On somme : $\int_a^b f(t)dt \approx \sum_{k=1}^n I_k(f)$.

Définition 3.4. *On dit qu'une méthode d'intégration numérique est d'ordre $p \in \mathbb{N}$ si pour tout fonction $f[a, b] \to \mathbb{R}$ assez régulière, il existe $C(f, a, b) > 0$ tel que*

$$\left| \int_a^b f(t)dt - \sum_{k=1}^n I_k(f) \right| \leq C(f, a, b) \left(\frac{b-a}{n} \right)^p .$$

Dans la proposition présentée au début de la section, on voit qu'on a approché $\int_{x_{k-1}}^{x_k} f(t)dt$ par $\int_{x_{k-1}}^{x_k} f(x_{k-1})dt = (x_k - x_{k-1})f(x_{k-1})$. De fait, sur l'intervalle $[x_{k-1}, x_k]$ la fonction f par la fonction constante égale à $f(x_{k-1})$: c'est la *méthode des rectangles à gauche.*

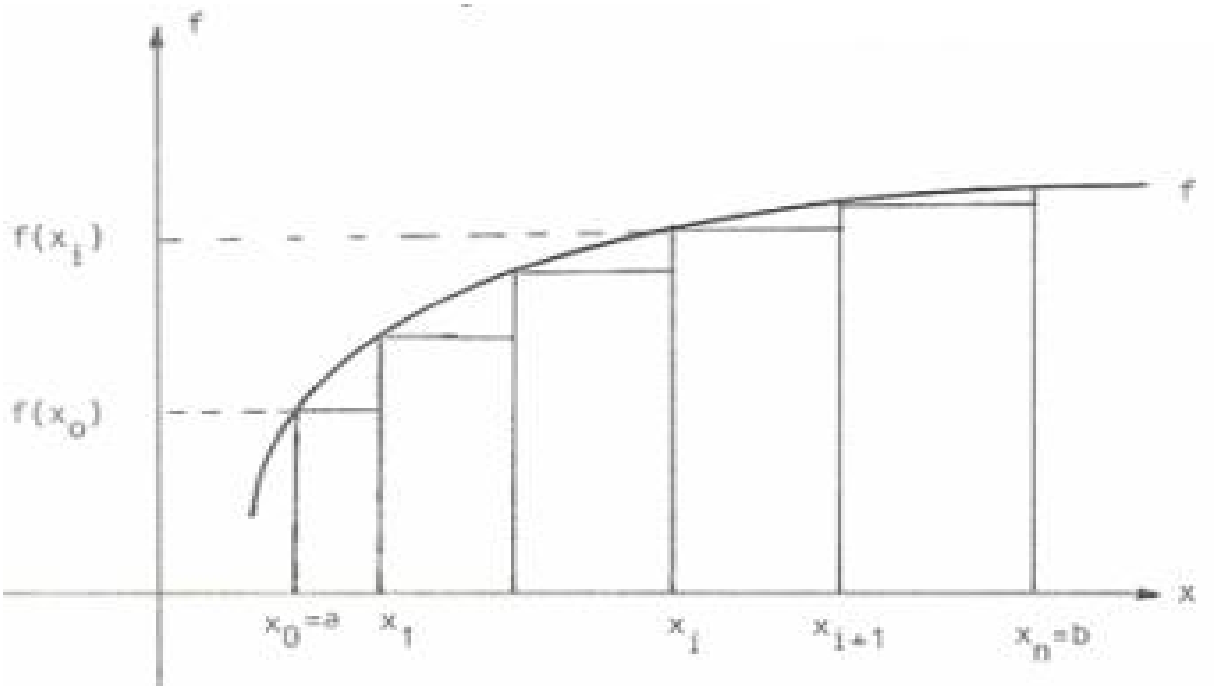

FIGURE 3.2 – Illustration de la méthode des rectangles à gauche

Si on choisit d'approcher $\int_{x_{k-1}}^{x_k} f(t)dt$ par $(x_k - x_{k-1})f(x_{k-1})$, on obtient la *méthode des rectangles à droite.* Toutes les deux sont d'ordre égal à $p = 1$.

3.2.2 Méthode des trapèzes

Principe. Sur l'intervalle $[x_{k-1}, x_k]$, on remplace f par son polynôme d'interpolation aux points x_{k-1} et x_k. On obtient alors pour valeur approchée de $\int_{x_{k-1}}^{x_k} f(t)dt$:

$$\begin{aligned} I_k(f) &= \int {x_{k-1}}^{x_k} \left(f(x_{k-1} + (t - x_{k-1}) \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \right) dt \\ &= (x_k - x_{k-1}) f(x_{k-1}) + \frac{(x_k - x_{k-1}}{2} (f(x_k) - f(x_{k-1}) \\ &= (x_k - x_{k-1}) \frac{f(x_k) + f(x_{k-1})}{2}. \end{aligned}$$

Cette méthode est appelée *méthode des trapèzes* (faire un dessin). C'est une méthode plus précise que la méthode des rectangles.

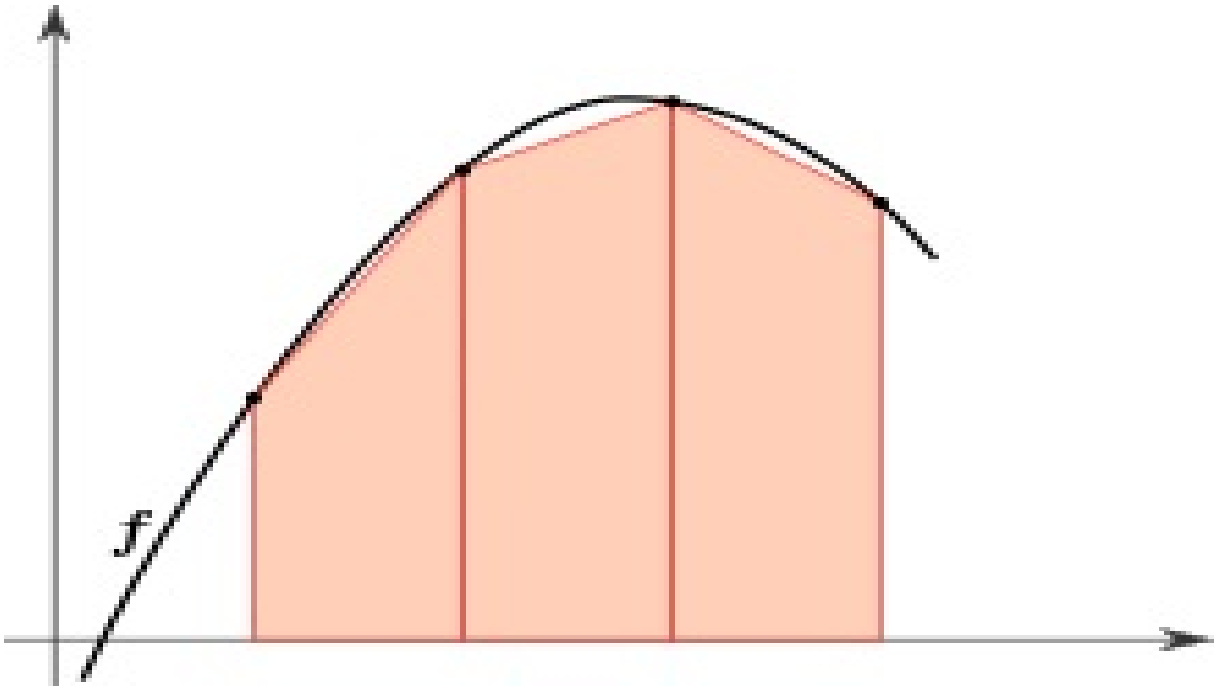

FIGURE 3.3 – Illustration de la méthode des trapèzes

Proposition 3.5. *La méthode des trapèzes est d'ordre* 2. *Pour toute fonction* $f : [a, b] \to \mathbb{R}$ *de classe* $\mathcal{C}^2$, *il existe une constante* $C(f, a, b)$ *telle que*

$$\left| \int_a^b f(t)dt - \frac{b-a}{n} \sum_{k=1}^{n} \frac{f(x_k) + f(x_{k-1})}{2} \right| \leq C(f, a, b) (\frac{b-a}{n})^2.$$

Démonstration. Sur l'intervalle $[x_{k-1}, x_k]$, on rappelle que l'écart entre la fonction f et son polynôme d'interpolation aux points x_{k-1} et x_k est donné par

$$|f(x) - f(x_{k-1}) - (x - x_{k-1}) \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}| \leq \sup_{z \in [a,b]} |f''(z)| \frac{(x - x_k)(x - x_{k-1})}{2}|$$

En intégrant sur $[x_{k-1}, x_k]$, on en déduit alors

$$|\int_{x_{k-1}}^{x_k} f(t)dt - I_k(f)| \leq \sup_{z\in[a,b]} |f''(z)| \int_{x_{k-1}}^{x_k} (t-x_k)(t-x_{k-1})dt \leq \sup_{z\in[a,b]} |f''(z)| \frac{(x_k - x_{k-1})^3}{6}.$$

En sommant ces inégalité pour $k = 1, \ldots, n$, on obtient :

$$\left| \int_a^b f(t)dt - \frac{b-a}{n} \sum_{k=1}^{n} \frac{f(x_k) + f(x_{k-1})}{2} \right| \leq \frac{b-a}{6} \sup_{z\in[a,b]} |f''(z)| \left(\frac{b-a}{n}\right)^2.$$

Ceci achève la démonstration de la proposition : la méthode des trapèzes est d'ordre 2.

3.2.3 Méthode de Simpson

Principe. Sur chaque intervalle $[x_{k-1}, x_k]$, on remplace la fonction f par son polynôme d'interpolation aux points x_{k-1}, $\frac{x_{k-1}+x_k}{2}$, x_k et et la valeur approchée $I_k(f)$ de $\int_{x_{k-1}}^{x_k} f(t)dt$ est donnée par l'intégrale de ce polynôme.

En appliquant la méthode de Newton, le polynôme d'interpolation est donné par

$$p(x) = p(x_{k-1}) + [p(x_{k-1}), p(\frac{x_{k-1}+x_k}{2})](x - x_{k-1}) + [p(x_{k-1}), p(\frac{x_{k-1}+x_k}{2}), p(x_k)](x - x_{k-1})(x - \frac{x_k + x_{k-1}}{2}).$$

On obtient, après calcul,

$$I_k(f) = \frac{(x_k - x_{k-1})}{6} \left(f(x_{k-1} + 4f(\frac{x_k + x_{k-1}}{2}) + f(x_k) \right).$$

On a la propriété suivante :

Proposition 3.6. *La méthode de Simpson est d'ordre 4. Pour toute fonction $f : [a, b] \to \mathbb{R}$ de classe $\mathcal{C}^4$, il existe une constante $C(f, a, b)$ telle que*

$$\left| \int_a^b f(t)dt - \frac{b-a}{n} \sum_{k=1}^{n} \frac{f(x_k) + 4f(\frac{x_k + x_{k-1}}{2}) + f(x_{k-1})}{6} \right| \leq C(f, a, b)(\frac{b-a}{n})^4.$$