ECOLE DES MINES DE NANCY

Année 2013-2014

Analyse Numérique

Cours de Takéo Takahashi

Polycopié rédigé par Michel Pierre et Antoine Henrot

Cours électif CE33

Semestre S7 : 2013-2014

Ecole des Mines de Nancy - Campus Artem - 54 042 Nancy Cedex
Tel : 03 83 68 45 95 - email : takeo.takahashi@univ-lorraine.fr

Table des matières

1 Les erreurs en Analyse Numérique **9**
- 1.1 Introduction . . . 9
- 1.2 Arithmétique machine . . . 10
  - 1.2.1 Représentation machine des nombres réels . . . 10
  - 1.2.2 Perte de chiffres significatifs . . . 10
- 1.3 Notions de conditionnement et stabilité . . . 12
  - 1.3.1 Conditionnement . . . 12
  - 1.3.2 Stabilité . . . 13
- 1.4 Conclusion . . . 14
- 1.5 Exercices du chapitre 1 . . . 15

2 Résolution d'équations non linéaires **17**
- 2.1 Introduction . . . 17
- 2.2 Cas des fonctions d'une variable . . . 17
  - 2.2.1 Préliminaires : séparation des zéros . . . 17
  - 2.2.2 Quelques algorithmes classiques . . . 18
    - 2.2.2.1 Méthode de dichotomie (ou bisection) . . . 18
    - 2.2.2.2 Méthode de la sécante . . . 19
    - 2.2.2.3 Critère d'arrêt . . . 19
    - 2.2.2.4 Méthode de Newton . . . 20
    - 2.2.2.5 Méthode de point fixe . . . 20
  - 2.2.3 Convergence des algorithmes . . . 22
    - 2.2.3.1 Méthodes de point fixe . . . 22
    - 2.2.3.2 Méthode de Newton . . . 24
    - 2.2.3.3 Méthode de la sécante . . . 26
  - 2.2.4 Comparaison des algorithmes . . . 28
    - 2.2.4.1 Méthode de dichotomie . . . 28
    - 2.2.4.2 Méthode de Newton . . . 29
    - 2.2.4.3 Méthode de la sécante . . . 29
    - 2.2.4.4 Méthode du point fixe . . . 29
  - 2.2.5 Accélération de la convergence . . . 30
    - 2.2.5.1 Méthode de relaxation . . . 30
    - 2.2.5.2 Méthode du Δ^2 d'Aitken . . . 31
- 2.3 Fonctions à valeurs complexes . . . 34
  - 2.3.1 Cas des équations polynômiales . . . 34
    - 2.3.1.1 Localisation des zéros . . . 34

2.3.1.2 Evaluation d'un polynôme : algorithme de Hörner . . . 35
2.3.1.3 Détermination de tous les zéros . . . 37
2.3.1.4 Méthode de Bairstow . . . 38
2.3.1.5 Méthode de Müller . . . 39
2.4 Système d'équations non linéaires . . . 40
2.4.1 Introduction . . . 40
2.4.2 La méthode de Newton-Raphson . . . 41
2.4.3 La méthode de Broyden . . . 42
2.5 Exercices du chapitre 2 . . . 42

3 Interpolation et approximation polynômiale 45
3.1 Interpolation de Lagrange . . . 45
3.1.1 Le polynôme d'interpolation . . . 45
3.1.2 Propriétés des différences divisées . . . 47
3.1.3 Erreur dans l'interpolation de Lagrange . . . 50
3.1.3.1 Analyse du théorème 3.3 . . . 51
3.2 Approximation polynômiale uniforme . . . 52
3.3 Approximation au sens des moindres carrés . . . 56
3.3.1 Introduction . . . 56
3.3.2 Méthode classique des moindres carrés . . . 57
3.3.3 Polynômes orthogonaux . . . 60
3.3.4 Exemples classiques de polynômes orthogonaux . . . 62
3.3.5 Polynômes trigonométriques . . . 64
3.3.6 La transformée de Fourier rapide . . . 67
3.4 Approximation par des fonctions polynômiales par morceaux . . . 67
3.4.1 Approximation linéaire par morceaux . . . 68
3.4.2 Approximation cubique . . . 68
3.4.3 Fonctions splines . . . 69
3.5 Exercices du chapitre 3 . . . 71

4 Dérivation et intégration numérique 73
4.1 Introduction . . . 73
4.2 Dérivation numérique . . . 74
4.2.1 Dérivée première . . . 74
4.2.1.1 Formules à deux points . . . 74
4.2.1.2 Formules à trois points . . . 75
4.2.1.3 Erreur . . . 75
4.2.2 Dérivées d'ordre supérieur . . . 76
4.3 Intégration numérique : méthodes composites . . . 76
4.3.1 Principe . . . 76
4.3.2 Méthode des rectangles . . . 76
4.3.3 Méthode des trapèzes . . . 77
4.3.4 Méthode de Simpson . . . 78
4.3.5 Méthode du trapèze corrigée . . . 78
4.4 Analyse de l'erreur dans les méthodes d'intégration . . . 79
4.4.1 Théorie . . . 79

4.4.2 Application aux méthodes usuelles . . . 80
4.4.2.1 Méthode des rectangles (à gauche ou à droite) . . . 80
4.4.2.2 Formule du point milieu . . . 80
4.4.2.3 Méthode des trapèzes . . . 81
4.4.2.4 Méthode de Simpson . . . 81
4.4.2.5 Méthode des trapèzes corrigés . . . 82
4.4.3 Exemple d'application . . . 82
4.5 Méthodes d'intégration numérique de Gauss . . . 82
4.5.1 Introduction . . . 82
4.5.2 Idée de base des méthodes de Gauss . . . 83
4.5.3 Mise en œuvre de la méthode . . . 84
4.5.3.1 Méthode de Legendre-Gauss . . . 85
4.5.3.2 Méthode de Gauss-Chebyshev . . . 86
4.5.3.3 Méthodes de Laguerre et Hermite . . . 86
4.6 Méthodes d'extrapolation à la limite . . . 86
4.6.1 Extrapolation à la limite de Richardson . . . 87
4.6.2 Méthode de Romberg . . . 89
4.7 Exercices du chapitre 4 . . . 91

5 Résolution numérique d'équations différentielles 93
5.1 Quelques remarques sur (5.1) . . . 93
5.1.1 Quelques soucis d'ordre théorique . . . 94
5.1.2 Régularité de la solution . . . 96
5.2 Méthodes de résolution à un pas . . . 97
5.2.1 La méthode d'Euler . . . 97
5.2.1.1 Estimation de l'erreur de discrétisation . . . 99
5.2.1.2 Influence des erreurs d'arrondis . . . 101
5.2.2 Méthode d'Euler implicite . . . 103
5.3 Les méthodes de Runge-Kutta . . . 103
5.4 Contrôle du pas . . . 107
5.5 Méthodes à pas multiples . . . 109
5.5.1 Méthodes d'Adams-Bashforth à $r+1$ pas . . . 109
5.5.2 Méthodes d'Adams-Moulton à $r+1$ pas . . . 111
5.5.3 Méthode de prédicteur-correcteur . . . 112
5.6 Comparaison des méthodes . . . 112
5.7 Applications à des problèmes aux limites . . . 113
5.8 Schéma de Newmark pour les problèmes d'ordre 2 . . . 114
5.9 Exercices du chapitre 5 . . . 116

6 Résolution de systèmes linéaires 117
6.1 Quelques remarques générales . . . 117
6.1.1 Origines des problèmes . . . 117
6.1.2 Méthodes utilisées . . . 117
6.1.3 Structure de la matrice . . . 118
6.1.3.1 Cas d'une matrice diagonale . . . 118
6.1.3.2 Matrice triangulaire supérieure (ou inférieure) . . . 118

6.1.3.3 Autres cas . . . 119
6.1.3.4 Matrices symétriques . . . 120
6.1.4 Stockage des matrices . . . 120
6.2 Méthodes directes de résolution de $AX = b$ . . . 123
6.2.1 Méthode d'élimination de Gauss . . . 123
6.2.1.1 Calcul du nombre d'opérations élémentaires . . . 126
6.2.2 Dérivés de la méthode de Gauss . . . 126
6.2.2.1 Factorisation LU . . . 127
6.2.2.2 Algorithme de Crout . . . 128
6.2.2.3 Cas particulier des matrices tridiagonales . . . 129
6.2.2.4 Méthode de Gauss-Jordan . . . 130
6.2.3 Factorisation de Cholesky . . . 131
6.2.4 Autres méthodes directes . . . 134
6.3 Analyse du conditionnement d'un système linéaire . . . 134
6.3.1 Quelques préliminaires sur les normes matricielles . . . 134
6.3.1.1 Norme sur l'espace vectoriel des matrices . . . 135
6.3.2 Analyse du conditionnement . . . 137
6.3.2.1 Propriétés de $cond\,(A)$ . . . 138
6.3.3 Exemple de système linéaire mal conditionné . . . 139
6.3.4 Préconditionnement . . . 140
6.4 Méthodes itératives . . . 140
6.4.1 Description des méthodes de Jacobi, Gauss-Seidel, relaxation . . . 140
6.4.2 Itérations par blocs . . . 141
6.4.3 Etude de la convergence des méthodes itératives . . . 142
6.4.3.1 Vitesse de convergence . . . 143
6.4.3.2 Application aux méthodes de Jacobi-Gauss-Seidel-Relaxation144
6.4.4 Application à la méthode des différences finies . . . 147
6.5 La méthode du gradient conjugué . . . 151
6.5.1 Gradient conjugué avec préconditionnement . . . 154
6.6 Exercices du chapitre 6 . . . 154

7 Calcul de valeurs propres et vecteurs propres 155
7.1 Rappels . . . 155
7.2 Origine des problèmes de valeurs propres . . . 157
7.2.1 Vibration d'une corde . . . 157
7.2.2 Vibration d'une membrane . . . 159
7.2.3 Problèmes de valeurs propres généralisés . . . 160
7.2.4 Système mécanique . . . 160
7.2.5 Autres exemples . . . 161
7.3 Méthode de la puissance itérée et de la puissance inverse . . . 161
7.3.1 Méthode de la puissance itérée . . . 161
7.3.2 Méthode de la puissance inverse . . . 163
7.4 Méthode de Jacobi . . . 164
7.5 Méthode de Givens-Householder . . . 168
7.5.1 Principe . . . 168
7.5.2 Description de la méthode de Givens . . . 168

7.5.3 Mise sous forme Hessenberg d'une matrice 171
7.6 La méthode QR 172
7.6.1 Algorithme QR de recherche de valeurs propres 174
7.7 Exercices du chapitre 7 176

Chapitre 1

Les erreurs en Analyse Numérique

1.1 Introduction

Dans le titre de ce chapitre, le mot **erreur** n'est pas pris au sens de faute (raisonnement faux dans la méthode, instruction fausse dans le programme). Il ne concerne que des erreurs *inévitables*. On peut les classer en trois catégories :

- **Les erreurs sur les données**. Elles peuvent être dues à l'imprécision des mesures physiques ou au fait que les données proviennent elle même d'un calcul approché. Elles sont imposées, en quelque sorte, de l'extérieur et nous ne pouvons agir sur elles. Néanmoins, la manière dont elles se propagent au cours des calculs est davantage du ressort du calculateur. L'analyse de cette propagation sera évoquée au cours de ce chapitre. Elle est liée aux notions de conditionnement et de stabilité (voir section 1.3).
- **Les erreurs d'arrondi**. Ce sont les erreurs dues au fait que la machine (dans ce cours, ce terme désignera indifféremment la calculette de poche ou l'ordinateur) ne peut représenter les nombres réels qu'avec un nombre fini de chiffres. A chaque opération mathématique élémentaire, il pourra y avoir une perte de chiffres significatifs. Le calculateur doit donc être vigilant quand le nombre d'opérations est très important. Cela va faire l'objet du prochain paragraphe.
- **Les erreurs d'approximation ou de discrétisation**. Ce sont les erreurs qu'on commet, par exemple, lorsqu'on calcule une intégrale à l'aide d'une somme finie, une dérivée à l'aide de différences finies ou bien la somme d'une série infinie à l'aide d'un nombre fini de ses termes (on parle alors quelquefois d'erreur de troncature). Une situation qu'on rencontrera souvent également consiste à approcher une fonction, solution d'une certaine équation fonctionnelle ou aux dérivées partielles, par une combinaison linéaire finie de fonctions élémentaires. Ce type d'erreurs est bien sûr fortement lié à la méthode employée. Un des buts de l'analyse numérique consiste justement à évaluer ces erreurs de discrétisation pour chaque algorithme mis en place. C'est donc un souci qui nous accompagnera tout au long de ce cours.

1.2 Arithmétique machine

1.2.1 Représentation machine des nombres réels

Le système de représentation machine des nombres réels le plus utilisé en calcul scientifique est celui de la représentation en virgule flottante normalisée. Un nombre $x \neq 0$ s'écrit

$$x \approx \pm m\, b^p$$

où b est la base de numération (entier supérieur ou égal à 2), m la mantisse (ensemble de chiffres de la base b) et p l'exposant (entier relatif). Dire que la représentation est normalisée, c'est supposer

$$b^{-1} \leq m < 1.$$

Par exemple, le nombre 395.2134 en base 10 s'écrit $+.3952134\ 10^{+3}$ ou, plus souvent, $+.3952134\ E{+}3$. Les bases $b = 2$ ou les puissances de 2 sont fréquemment utilisées par les constructeurs d'ordinateurs. Cela signifie, par exemple que tous les calculs internes sont faits en base 2 et seul le résultat affiché est traduit en base 10.

La mantisse m est donc un nombre qui commence toujours par $0.\ldots$ et qui s'écrit avec un nombre maximum N de chiffres significatifs (imposé par le choix de la taille des emplacements mémoires alloués au type *réel*). Signalons la possibilité offertes sur la plupart des ordinateurs (mais pas sur les calculatrices !) de travailler en *double précision*, c'est-à-dire essentiellement avec une mantisse comportant $2N$ chiffres significatifs. Cette possibilité est évidemment coûteuse en temps de calcul et ne doit surtout pas être utilisée systématiquement. Elle est intéressante, en particulier, quand on n'est pas très sûr de la stabilité d'un algorithme, pour pouvoir comparer des résultats obtenus en simple précision et en double précision.

L'exposant p est lui aussi limité par la machine avec laquelle on travaille. Si p est hors des limites permises (i.e. trop grand en valeur absolue), le nombre x ne sera pas représentable en machine : elle affichera alors un message du type *exponent overflow* quand le nombre est trop grand, ou elle affichera 0 s'il est trop petit, voir Exercice 1.1. On appelle quelquefois **capacité** le plus grand nombre que peut écrire la machine et **pas** le plus petit nombre. En général, le pas est l'inverse de la capacité.

La différence entre la valeur exacte d'un nombre x et la valeur x^* de sa représentation est appelée **erreur d'arrondi**. Elle est majorée par $\frac{1}{2}b^{-N}b^p$, soit en valeur relative :

$$\left|\frac{x - x^*}{x}\right| \leq \frac{b^{-N}b^p}{2x} \leq \frac{b^{-N}b^p}{2b^{p-1}} = \frac{b^{1-N}}{2}\,.$$

Cette erreur est systématiquement présente dans tout calcul arithmétique sur nombre réel effectué par un ordinateur. Elle fait que l'arithmétique numérique n'a pas la précision de l'arithmétique mathématique et peut, si on n'y prend garde, conduire à des résultats inexacts, voire aberrants, comme le montrent les exemples suivants et les exercices.

1.2.2 Perte de chiffres significatifs

Pour faciliter la compréhension, nous nous placerons dans l'environnement rassurant de la base 10, les phénomènes étant du même type dans toute base.

Exemple 1.1 Supposons que dans un calcul apparaisse la quantité $x = \pi - 3.1415$ (où $\pi = 3.141592653589793\ldots$). Si on travaille avec 8 chiffres significatifs (comme

beaucoup de calculettes), le nombre π sera représenté par : $\pi^* = 0.31415927\ 10^{-1}$ en virgule flottante normalisée. On aura donc

$$x = 0.31415927\ 10^{-1} - 0.31415\ 10^{-1} = 0.0000927\ 10^{-1} = 0.927\ 10^{-4}$$

On constate que x ne contient en fait que 3 chiffres significatifs et non 8, soit une perte sèche de 5 chiffres significatifs. Ce genre de phénomènes peut surgir à tout moment d'un calcul. Nous verrons de nombreux exemples d'algorithmes conduisant à faire la différence de 2 nombres proches.

Exemple 1.2 : Soit à calculer le quotient

$$A = \frac{XN}{XD} = \frac{\pi - 3,1415}{10^4(\pi - 3,1515) - 0,927}$$

En travaillant avec 8 chiffres significatifs, on a :

$$XD\ = 10^4(0,927.10^{-4}) - 0,927 = 0,0$$

$\pi^* = 3,1415927$	$A = ERREUR$
$\pi^* = 3,14159265$	$A = -0,18530...$
$\pi^* = 3,141592653\ \rceil$	$A = -0,197134...$
$\pi^* = 3,141592654\ \rfloor$	$A = -0,201427...$
$\pi^* = 3,1415926535\ \rceil$	$A = -0,1992548...$
$\pi^* = 3,1415926536\ \rfloor$	$A = -0,1996844...$
$\pi^* = 3,14159265358\ \rceil$	$A = -0,1995984...$
$\pi^* = 3,14159265359\ \rfloor$	$A = -0,19964143...$
$\pi^* = 3,141592653589$	$A = -0,19963713...$
$\pi^* = 3,1415926535897\ \rceil$	$A = -0,199640143...$
$\pi^* = 3,1415926535898\ \rfloor$	$A = -0,1996405743...$
$\pi^* = 3,14159265358979$	$A = -0,1996405312$
$\pi^* = 3,1415927653589793$	$A = -0,1996405439...$

On constate que pour obtenir p chiffres significatifs pour A, il faut représenter π à l'aide de $p + 8$ chiffres significatifs.
Dans le tableau ci dessus, le symbole $\rceil$ signale des représentations avec le même nombre de chiffres, tous étant identiques sauf le dernier.

Exemple 1.3 : L'addition numérique n'est pas associative : en virgule flottante à n chiffres significatifs, $(a + b) + c$ peut être différent de $a + (b + c)$. C'est le cas avec le choix suivant où les calculs sont faits avec 8 chiffres significatifs.

$$\begin{aligned} a &: = 0,23371258.10^{-4} \\ b &: = 0,33678429.10^{2} \\ c &: = -0,33677811.10^{2} \end{aligned}$$

$$a + b = 0,00000023(371258).10^2 + 0,33678429.10^2 = 0,33678452.10^2.$$

On remarque que, dans cette addition, les 6 derniers chiffres de a sont perdus. Ainsi :

$$\begin{aligned} (a + b) + c &= 0,33678452.10^2 - 0,33677811.10^2 \\ &= 0,00000641.10^2 = 0,641.10^{-3}. \end{aligned}$$

Par ailleurs :

$$\begin{aligned} b + c &= 0,33678429.10^2 - 0,33677811.10^2 \\ &= 0,00000618.10^2 = 0,618.10^{-3} \end{aligned}$$

$$a+(b+c)=0,02337125\,(8)\,.10^{-3}+0,61800000.10^{-3}=0,64137126.10^{-3}.$$

Essayons d'analyser le phénomène ci-dessus. Si $vf\,(a+b)$ désigne le résultat de l'addition $a+b$, on a :

$$vf\,(a+b)=(a+b)\,(1+\varepsilon_1)$$

où ε_1 désigne l'erreur relative commise dans le calcul : celle-ci dépend de la précision de la machine. On sait qu'elle est majorée par $\frac{1}{2}\beta^{1-n}$ soit ici 5.10^{-8}. Posons $\eta=vf\,(a+b)$. On a alors

$$\begin{aligned} vf\,((a+b)+c) &= vf\,(\eta+c)=(\eta+c)\,(1+\varepsilon_2) \qquad |\varepsilon_2|\leq 5.10^{-8} \\ &= [(a+b)\,(1+\varepsilon_1)+c]\,(1+\varepsilon_2) \\ &= a+b+c+(a+b)\,\varepsilon_1\,(1+\varepsilon_2)+(a+b+c)\,\varepsilon_2. \end{aligned}$$

Ainsi :

$$\frac{vf\,((a+b)+c)-(a+b+c)}{a+b+c}=\frac{a+b}{a+b+c}\varepsilon_1\,(1+\varepsilon_2)+\varepsilon_2.$$

De la même façon :

$$\frac{vf\,(a+(b+c))-(a+b+c)}{a+b+c}=\frac{b+c}{a+b+c}\varepsilon_3\,(1+\varepsilon_4)+\varepsilon_4.$$

On voit que les erreurs $\varepsilon_1\,(1+\varepsilon_2)$, $\varepsilon_3\,(1+\varepsilon_4)$, environ égales à 5.10^{-8} sont soumises à des coefficients amplificateurs

$$\frac{a+b}{a+b+c}\simeq 5.10^4, \qquad \frac{b+c}{a+b+c}\simeq 0,9.$$

Ceci explique pourquoi le second calcul est plus précis que le premier.

Remarque 1.1 Dans les calculs où interviennent des nombres d'ordres de grandeur différents, il est en général préférable d'effectuer les opérations en groupant ceux d'ordres de grandeur similaires pour éviter les pertes de chiffres significatifs.

1.3 Notions de conditionnement et stabilité

Ces deux notions, toujours présentes en analyse numérique, sont relatives à la propagation plus ou moins importante des erreurs d'arrondi dans un calcul donné. Nous les étudions ici pour le calcul d'une fonction.

$$x\in\mathbb{R}\longmapsto f(x)\in\mathbb{R}.$$

1.3.1 Conditionnement

Le conditionnement décrit la sensibilité de la valeur d'une fonction à une petite variation de son argument, c'est-à-dire :

$$\frac{f\,(x)-f\,(x*)}{f\,(x)} \text{ en fonction de } \frac{x-x*}{x}$$

lorsque $x-x*$ est petit. Pour une fonction suffisamment régulière, on a évidemment :

$$\left|\frac{f\,(x)-f\,(x*)}{f\,(x)}\Big/\frac{x-x*}{x}\right|\simeq\left|\frac{xf'\,(x)}{f\,(x)}\right|.$$

D'où on tire :

Définition 1.1 *On appelle conditionnement d'une fonction numérique f de classe C^1 en un point x, le nombre*

$$cond(f)_x := \left|\frac{xf'(x)}{f(x)}\right|.$$

Exemple 1.4 : $f(x) = \sqrt{x}$

$$\left|\frac{xf'(x)}{f(x)}\right| = \frac{1}{2}.$$

Ceci correspond à un bon conditionnement, puisque l'erreur relative sur f sera au plus moitié d'une erreur relative sur x.

Exemple 1.5 : $f(x) = a - x$

$$\left|\frac{xf'(x)}{f(x)}\right| = \left|\frac{x}{a-x}\right|.$$

Ici, le conditionnement est très mauvais si x est voisin de a. Ceci correspond aux exemples 1.1, 1.2, 1.3.

1.3.2 Stabilité

La stabilité décrit la sensibilité d'un algorithme numérique pour le calcul d'une fonction $f(x)$.

Exemple 1.6 :

$$f(x) = \sqrt{x+1} - \sqrt{x}.$$

Le conditionnement de cette fonction est égal à :

$$\left|\frac{xf'(x)}{f(x)}\right| = \left|\frac{x\left(\sqrt{x} - \sqrt{x+1}\right)}{2\sqrt{x(x+1)}} - \frac{1}{\sqrt{x+1} - \sqrt{x}}\right| = \frac{1}{2}\sqrt{\frac{x}{x+1}}$$

Cette dernière expression étant proche de $\frac{1}{2}$ pour x grand. Donc, si x est grand, le conditionnement de f est bon. Cependant, dans un calcul à 6 chiffres significatifs, on a :

$$f(12345) = \sqrt{12346} - \sqrt{12345} = 111,113 - 111,108 = 0,500000.10^{-2}.$$

Or un calcul précis donne : $f(12345) = 0,4500032....10^{-2}$. On a donc une erreur de 10% ce qui est important et peu en accord avec le bon conditionnement de f. Ceci est dû à l'algorithme utilisé dans ce calcul que l'on peut expliciter comme suit :

$$\begin{aligned} x_0 &: = 12345 \\ x_1 &: = x_0 + 1 \\ x_2 &: = \sqrt{x_1} \\ x_3 &: = \sqrt{x_0} \\ x_4 &: = x_2 - x_3 \end{aligned}$$

Il y a quatre fonctions à intervenir et, à priori, même si le conditionnement de f est bon, il se peut que le conditionnement d'une ou plusieurs fonctions utilisées dans l'algorithme soit supérieur à celui de f. C'est ce qui se produit ici pour la fonction $x_3 \longmapsto x_4 = x_2 - x_3$ (x_2 étant supposé fixe) dont le conditionnement est grand lorsque x_3 est voisin de x_2 comme on l'a vu précédemment.

En conclusion, le choix d'un bon algorithme numérique est essentiel. Par exemple, ci-dessus, un meilleur algorithme est obtenu en utilisant :

$$f(x) = \sqrt{x+1} - \sqrt{x} = \frac{1}{\sqrt{x+1} + \sqrt{x}}.$$

Dans ce cas, toujours avec 6 chiffres significatifs :

$$f(12345) = \frac{1}{\sqrt{12346} + \sqrt{12345}} = \frac{1}{222,221} = 0,450002.10^{-2}$$

ce qui donne une erreur relative de $0,0003\%$. Dans la même optique, l'exemple suivant peut aussi être médité.

Exemple d'instabilité : Calcul de e^{-12} à l'aide de sa série de Taylor

$$e^x = \sum_{n=0}^{N} \frac{x^n}{n!} \ (= S_N) \ \text{ pour } N \text{ grand.}$$

Voici les valeurs successives de cette somme en fonction de N pour $x = -12$. Le calcul est fait avec 10 chiffres significatifs.

N	S_N	N	S_N	N	S_N
2	$-11,0...$	19	$1629,87...$	36	$-0,001432...$
3	$61,0...$	20	$-996,45...$	37	$0,000472...$
4	$-227,0...$	21	$579,34...$	38	$-0,0001454...$
5	$637,0...$	22	$-321,11...$	39	$0,000049726...$
6	$-1436,6...$	23	$170,04...$	40	$-0,000010319...$
7	$2710,6...$	24	$-86,20...$	41	$0,000007694...$
8	$-4398,88...$	25	$41,91...$	42	$0,000002422...$
9	$6265,34...$	26	$-19,58...$	43	$0,000003928...$
10	$-7953,62...$	27	$8,80...$	44	$0,000003508...$
11	$9109,137...$	28	$-3,8130...$	45	$0,000003623...$
12	$-9504,78...$	29	$1,5937...$	46	$0,000003592...$
13	$9109,13...$	30	$-0,6435...$	47	$0,000003600...$
14	$-8072,94...$	31	$0,2513...$	48	$0,000003598...$
15	$6654,55...$	32	$-0,0950...$	49	$0,000003599...$
16	$-5127,44...$	33	$0,0348...$	50	$0,000003598...$
17	$3709,05...$	34	$-0,01238...$		
18	$-2528,47...$	35	$0,004283...$		

La valeur de e^{-12} est en fait de $0,0000061442...$. La formule $e^{-x} = \frac{1}{e^x}$ donne lieu à un calcul plus stable, même si e^x est calculé comme ci-dessus.

1.4 Conclusion

Afin de limiter la propagation des erreurs d'arrondi, il faut essayer d'anticiper en utilisant des algorithmes dont la stabilité est optimisée par un choix d'opérations intermédiaires à bon conditionnement.

Les phénomènes soulignés dans ce chapitre ne pouvant être, en tout état de cause, complètement éliminés, on peut essayer d'évaluer l'erreur totale à laquelle un algorithme est susceptible de donner lieu :

a/ en faisant un calcul en double précision et en confrontant le résultat au même calcul fait en simple précision. C'est ce que nous avons fait dans l'exemple 1.2. Cependant,

cette technique est très coûteuse en temps machine puisqu'elle peut multiplier le temps de calcul par un facteur 8.
b/ en faisant une analyse mathématique de l'erreur : ce peut être une analyse rétrograde de l'erreur comme celle utilisée plus haut pour comparer $(a+b)+c$ et $a+(b+c)$. Des méthodes statistiques peuvent être également utilisées. Nous renvoyons à la littérature spécialisée pour ces questions le plus souvent délicates.

1.5 Exercices du chapitre 1

Exercice 1.1 En écrivant un petit programme, trouver la capacité et le pas de votre calculatrice de poche ou de l'ordinateur.

Exercice 1.2 Calculer les racines de l'équation $x^2 + 111,11x + 1,2121 = 0$ dans une arithmétique à 5 chiffres significatifs (base 10) en utilisant les formules $x = \frac{-b+\sqrt{b^2-4ac}}{2a}$, puis $x = \frac{-2c}{b+\sqrt{b^2-4ac}}$, et analyser les résultats.

Exercice 1.3 Estimer l'erreur faite dans le calcul de $(\cos x)\, e^{10x^2}$ autour de $x = 2$ quand l'erreur sur x est d'ordre 10^{-6}.

Exercice 1.4 Trouver une méthode pour calculer :

$$\sin(\alpha + x) - \sin\alpha$$

lorsque x est voisin de 0, à la précision permise par le nombre de chiffres significatifs utilisés.

Exercice 1.5 Calculer, en arrondissant à quatre chiffres significatifs chaque calcul intermédiaire, la somme des nombres suivants d'abord dans le sens croissant puis dans le sens décroissant. Comparer les résultats.

$$0,1580 \qquad 0,2653 \qquad 0,2581.10^1 \qquad 0,4288.10^1 \qquad 0,6266.10^2 \qquad 0,7555.10^2$$

$$0,7889.10^3 \qquad 0,7767.10^3 \qquad 0,8999.10^4.$$

Exercice 1.6 Les valeurs 1, $\frac{1}{6}$, $\frac{1}{6^2}$, ..., $\frac{1}{6^n}$ sont théoriquement générées par la suite $x_0 = 1$, $x_1 = \frac{1}{6}$, $x_{n+1} = \dfrac{37}{6}x_n - x_{n-1}$ pour tout $n \geq 1$.
Calculer les valeurs successives avec 4 décimales puis avec 8 décimales et comparer... On constate une différence très importante dès la 5ème étape. Expliquer !

Chapitre 2

Résolution d'équations non linéaires

2.1 Introduction

Le but de ce chapitre est de décrire les algorithmes les plus fréquemment utilisés pour résoudre des équations non linéaires du type :

$$f(x) = 0$$

où

1. x est une variable réelle, f une fonction à valeurs réelles
2. x est une variable complexe, f une fonction à valeurs complexes
3. x est un vecteur de $\mathbb{R}^n$, f une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^n$.

Le premier point sera le plus détaillé : la convergence des algorithmes est analysée. On présentera également des techniques d'accélération de convergence. Pour les fonctions complexes, on insistera sur le cas des polynômes. Pour les fonctions de $\mathbb{R}^n$ dans $\mathbb{R}^n$ on présentera rapidement les algorithmes qui seront en fait inspirés de la dimension 1.

2.2 Cas des fonctions d'une variable

2.2.1 Préliminaires : séparation des zéros

Exemple 2.1 : résoudre $x - 0,2\sin x - 0,5 = 0$, x réel.
Exemple 2.2 : résoudre $\cos x = e^{-x}$, x réel.

Le premier travail consiste en une analyse mathématique minimale pour séparer les zéros, c'est-à-dire déterminer des intervalles $[a_i, b_i]$ dans lesquels l'équation considérée a une solution et une seule.
La méthode la plus simple est d'utiliser qu'une fonction continue strictement monotone sur un intervalle $[a_i, b_i]$ et telle que $f(a_i) f(b_i) \leq 0$ a un zéro et un seul dans l'intervalle $[a_i, b_i]$. Par exemple :

$$f_1(x) = x - 0,2\sin x - 0,5$$

$$f_1'(x) = 1 - 0,2\cos x \geq 0 \text{ pour tout } x\,.$$

Donc f_1 est strictement croissante sur $\mathbb{R}$. Comme $f_1(0) = -0,5 < 0$, $f_1(\pi) = \pi - 0,5 > 0$, f_1 a un unique zéro dans $[0, \pi]$.

Un calcul similaire pour l'exemple 2.2 conduit à une impasse si on pose $f_2(x) := \cos x - e^{-x}$ puisque la dérivée $f_2'(x) = \sin x + e^{-x}$ est du même type. Cette situation est bien sûr beaucoup plus fréquente. On peut alors :

A/ choisir plus judicieusement la fonction auxiliaire f_2. Ici, on peut par exemple poser $f_2(x) := e^x \cos x - 1$. Les zéros de f_2 sont exactement les solutions de (2.2). D'autre part :

$$f_2'(x) = e^x(\cos x - \sin x) = \sqrt{2}e^x \cos\left(x - \frac{\pi}{4}\right).$$

Ainsi f_2 est strictement monotons sur les intervalles du type $\left[\frac{\pi}{4} + k\frac{\pi}{2}, \frac{\pi}{4} + (k+1)\frac{\pi}{2}\right]$, k entier. L'étude des signes successifs de $f\left(\frac{\pi}{4} + k\frac{\pi}{2}\right)$ permet alors de localiser les zéros éventuels.

B/ On peut procéder par tatonnements en s'aidant au maximum d'informations complémentaires disponibles. Ainsi, dans l'exemple 2.2, le tracé des représentations graphiques des fonctions $x \longmapsto \cos x$ et $x \longmapsto e^{-x}$ est particulièrement suggestif.

Dans la suite, nous nous placerons le plus souvent dans le cas où $f : [a,b] \to \mathbb{R}$ admet un unique zéro dans l'intervalle $[a,b]$.

2.2.2 Quelques algorithmes classiques

2.2.2.1 Méthode de dichotomie (ou bisection)

Reprenons l'exemple 2.1. Nous avons vu que si $f(x) = x - 0.2\sin x - 0.5$, $f(0) < 0$, $f(\pi) > 0$; d'où un zéro dans $(0,\pi)$. Si on calcule la valeur de f en $\left(\frac{\pi}{2}\right)$, on constate que $f\left(\frac{\pi}{2}\right) = \frac{\pi}{2} - 0,7 > 0$. Donc ce zéro est en fait dans $\left[0, \frac{\pi}{2}\right]$. On peut alors calculer $f\left(\frac{\pi}{4}\right) = 0,14 > 0$, qui montre qu'il est dans $\left[0, \frac{\pi}{4}\right]$.

Il est clair qu'une répétition de ce procédé donne un encadrement de plus en plus précis du zéro cherché et fournit donc un algorithme de calcul de ce zéro.

Algorithme 2.1 : Soit $f : [a_0, b_0] \to R$ continue et telle que $f(a_0) f(b_0) \leq 0$.

$$\left[\begin{array}{c} \text{Pour } n = 0,1,2,...,N, \text{ faire} \\ m := \frac{(a_n + b_n)}{2} \\ \text{Si } f(a_n) f(m) \leq 0,\ a_{n+1} := a_n,\ b_{n+1} := m \\ \text{Sinon} \qquad\qquad a_{n+1} := m,\ b_{n+1} := b_n. \end{array}\right.$$

On a : $a_{n+1} - b_{n+1} = \frac{1}{2}(a_n - b_n)$, soit par récurrence $a_n - b_n = \frac{1}{2^n}(a_0 - b_0)$. Il en résulte que cette méthode est toujours convergente puisque $a_n - b_n$ tend vers 0 quand n tend vers l'infini. On peut choisir le temps d'arrêt N pour que :

$$\frac{1}{2^N}(a_0 - b_0) < \varepsilon = \text{ précision choisie.}$$

On obtient alors un encadrement de la solution cherchée à la précision voulue. Bien que cette situation soit très alléchante, on verra qu'en fait cette méthode converge plutôt lentement comparée à celles qui suivent.

Remarque 2.1 Dans l'exemple 2.1, on a $f(0) = -0,5$, $f(\pi) = 2,64$. Ceci laisse immédiatement penser que le zéro cherché doit être plus près de 0 que de π. Le plus efficace n'est donc pas de tester la valeur de f en le milieu de $[0,\pi]$ mais plutôt en un point plus voisin de 0 tenant compte de cette différence de poids entre $f(0)$ et $f(\pi)$. Ceci conduit aux algorithmes dits de "fausse position" qui convergent généralement plus vite. Nous ne les expliciterons pas ici.

Remarque 2.2 La méthode ci-dessus converge même si f a plusieurs zéros dans l'intervalle $[a_0, b_0]$.

2.2.2.2 Méthode de la sécante

Soit f admettant un zéro dans l'intervalle $[x_{-1}, x_0]$. Pour obtenir une première approximation de ce zéro, l'idée est de remplacer f par son interpolé linéaire sur $[x_{-1}, x_0]$, soit par

$$Y(x) = f(x_0) + (x - x_0) \frac{f(x_0) - f(x_{-1})}{x_0 - x_{-1}},$$

l'unique fonction linéaire dont les valeurs coïncident avec celles de f en x_{-1} et x_0. L'approximation x_1 est alors obtenue en résolvant :

$$Y(x_1) = 0$$

soit

$$x_1 = x_0 - f(x_0) \frac{x_0 - x_{-1}}{f(x_0) - f(x_{-1})}.$$

Géométriquement (cf Figure 2.1), ceci revient à remplacer la courbe d'équation $y =$

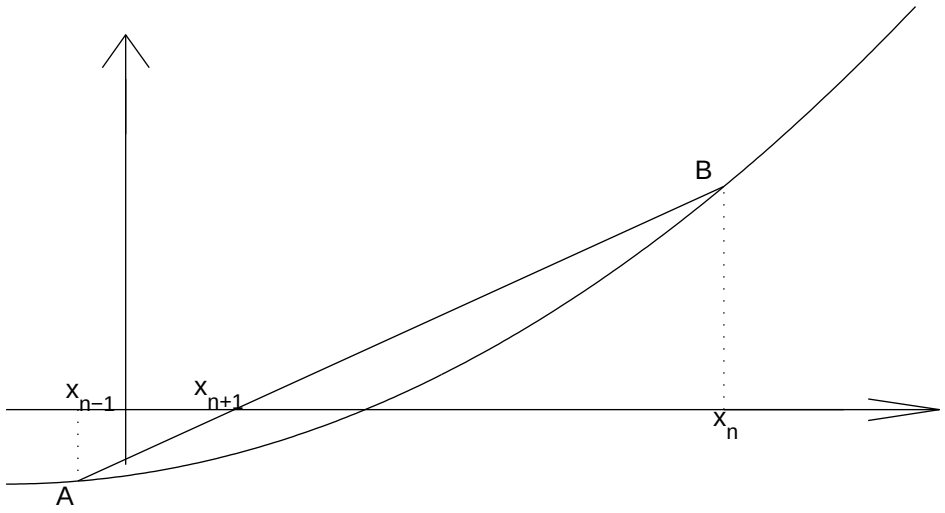

FIGURE 2.1 – Méthode de la sécante

$f(x)$ par la sécante AB et x_{n+1} est l'intersection de AB avec la droite (Ox).
Comme le montre le dessin, x_{n+1} semble plus voisin du zéro cherché que x_{n-1} ou x_n. Pour trouver une meilleure approximation, il suffit de répéter le procédé à l'aide des points (x_n, x_{n+1}). En continuant, on obtient l' :

Algorithme 2.2 : x_0 et x_{-1} étant donnés,

$$\left[\begin{array}{c} \text{Pour } n = 0, 1, 2, ... \\ x_{n+1} = x_n - f(x_n) \dfrac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}. \end{array} \right.$$

2.2.2.3 Critère d'arrêt

Cet algorithme n'est évidemment pas complet tant qu'on n'a pas précisé un critère d'arrêt. Nous verrons plus loin que, généralement, x_n converge vers la solution x_∞ cherchée ce qui signifie que pour n grand, x_n est voisin de x_∞. Sans plus d'informations,

il est difficile de savoir ce que signifie numériquement "n grand", et c'est là une difficulté fréquente en analyse numérique.
Un critère d'arrêt souvent utilisé consiste à choisir *a priori* une tolérance ε et à terminer l'algorithme lorsque

$$|x_{n+1} - x_n| \leq \varepsilon \,. \tag{2.1}$$

Ce n'est pas complètement satisfaisant ; pour s'en convaincre, on peut penser à la suite :

$$x_n = 1 + \frac{1}{2} + ... + \frac{1}{n}$$

qui vérifie $\lim_{n\to\infty} |x_{n+1} - x_n| = 0$ bien que tendant vers l'infini.
Puisqu'on veut résoudre $f(x) = 0$, un autre critère d'arrêt possible consiste à s'arrêter lorsque

$$|f(x_n)| < \varepsilon \,. \tag{2.2}$$

Un numéricien particulièrement scrupuleux pourra décider de ne s'arrêter que lorsque **les deux** critères d'arrêts (2.1) et (2.2) sont **simultanément** vérifiés.
Dans le cas de l'algorithme de la sécante ci-dessus, on peut aussi utiliser :

$$|f(x_n) - f(x_{n-1})| < \varepsilon \,.$$

ce critère d'arrêt a l'avantage d'éviter une possible division par 0.
Enfin, pour éviter à l'ordinateur de tourner sans s'arrêter lorsqu'il n'y a pas convergence, il est évidemment **indispensable de toujours** mettre un critère limitant le nombre total d'itérations.

2.2.2.4 Méthode de Newton

Ici, au lieu d'assimiler la courbe "$y = f(x)$" à une sécante, on l'assimile à la tangente en un point $(x_n, f(x_0))$ soit, la droite d'équation :

$$Y = f(x_n) + f'(x_n)(x - x_n) \,.$$

Son intersection avec l'axe des abscisses fournit une approximation de la solution x_∞. A nouveau, on renouvelle le procédé jusqu'à obtenir une approximation suffisante, d'où l' :

Algorithme 2.3 : x_0 étant donné

$$\left[\begin{array}{l} \text{Pour } n = 0, 1, 2, ... \\ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \end{array}\right.$$

Remarque 2.3 Cet algorithme est initié à l'aide d'un seul point. Il peut être vu comme une modification de l'algorithme précédent où le quotient $\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$ a été remplacé par $\frac{1}{f'(x_n)}$. Nous verrons plus loin qu'il donne lieu à une convergence beaucoup plus rapide.

2.2.2.5 Méthode de point fixe

Elle consiste à d'abord remplacer l'équation

$$(*) \quad f(x) = 0$$

par une équation

$$(**) \quad g(x) = x$$

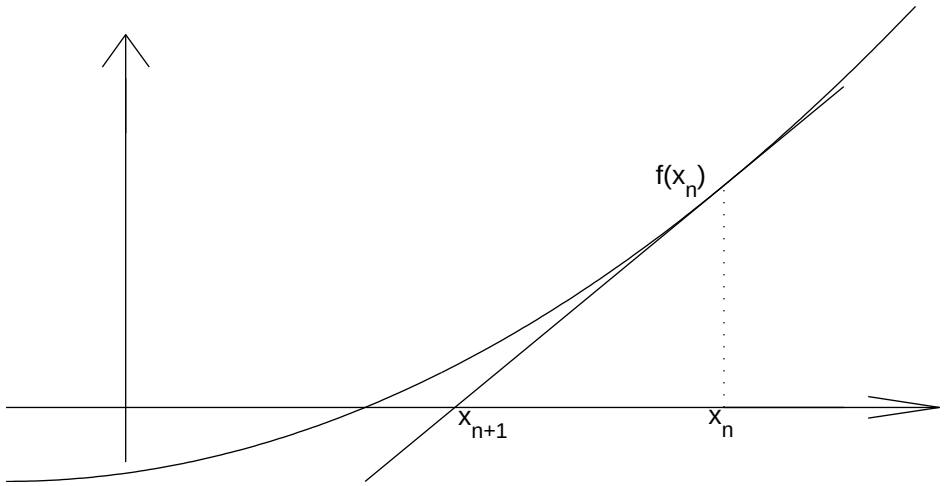

FIGURE 2.2 – Méthode de Newton

ayant mêmes solutions. On est ainsi ramené à la recherche des points fixes de l'application g. Le remplacement de $(*)$ par $(**)$ est toujours possible en posant, par exemple, $g(x) = f(x) + x$. Ce n'est évidemment pas forcément le meilleur choix !

Exemple 2.3 : Pour l'équation

$$x^2 - x - 2 = 0$$

on peut prendre

$$\begin{aligned} g(x) &= x^2 - 2 \\ g(x) &= \sqrt{2+x} \\ g(x) &= 1 + \frac{2}{x} \\ g(x) &= x - \frac{x^2 - x - 2}{m} \text{ pour tout paramètre } m \neq 0. \end{aligned}$$

l'équation étant sous la forme $(**)$, on a alors l'algorithme suivant :

Algorithme 2.4 : On choisit x_0

$$\left[\begin{array}{l} \text{Pour } n = 0, 1, 2, ... \\ \quad x_{n+1} = g(x_n). \end{array} \right.$$

Cette méthode est justifiée par la :

Proposition 2.1 *Soit $g : [a,b] \to [a,b]$ continue et $x_0 \in [a,b]$. Si x_n converge vers x_∞, alors*

$$x_\infty = g(x_\infty).$$

La démonstration est immédiate en passant à la limite dans l'égalité $x_{n+1} = g(x_n)$ et en utilisant la continuité de g au point x_∞.
Une difficulté majeure dans l'algorithme 2.4 est qu'il peut arriver qu'on ne puisse calculer $g(x_n)$. Il suffit de penser à ce qui se passe lorsque $g(x) = -\sqrt{x}$!

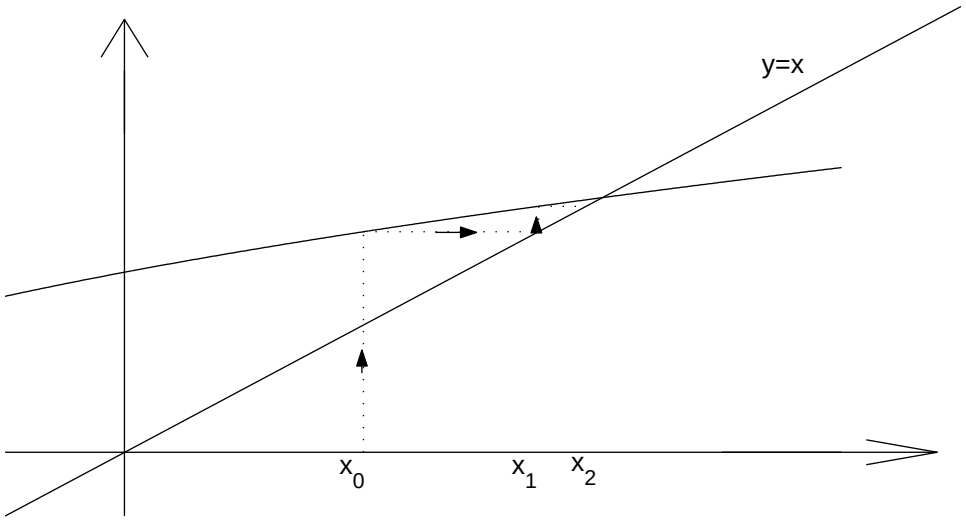

FIGURE 2.3 – Méthode de point fixe pour $g(x) = \sqrt{x+2}$

Remarque 2.4 Il peut être intéressant de construire graphiquement les valeurs successives de x_n dans certains cas : prenons $g(x) = \sqrt{2+x}$ comme dans l'exemple 2.3 :

On utilise la première bissectrice pour reporter les valeurs successives de x_n. On peut se convaincre à l'aide de ce type de construction que certains points fixes ne sont jamais atteints par l'algorithme 2.4 (cf Figure 2.4 ci après) :

Exemple 2.4 : $g(x) = x^2$ Si x_0 est choisi à gauche de 1, la suite x_n converge vers le point fixe 0.
Si x_0 est choisi à droite de 1, la suite x_n tend versl'infini.
A moins de choisir exactement $x_0 = 1$, on voit que la suite ne converge jamais vers 1 : c'est un point fixe instable. On y reviendra dans le paragraphe suivant.
Cet exemple met par ailleurs en évidence l'influence du choix de x_0 sur le comportement de x_n.

2.2.3 Convergence des algorithmes

2.2.3.1 Méthodes de point fixe

Commençons par traiter le cas du point fixe qui est fondamental d'un point de vue théorique.

Théorème 2.1 *Soit* $g : [a,b] \to [a,b]$ *dérivable et telle que*

$$|g'(x)| \leq K \ \ \forall x \in [a,b] \ \ \textit{avec } 0 \leq K < 1 \tag{2.3}$$

Alors, pour tout $x_0 \in [a,b]$, *la suite définie par*

$$x_{n+1} = g(x_n) \ \ \forall n \in N$$

converge vers l'unique point fixe de g.

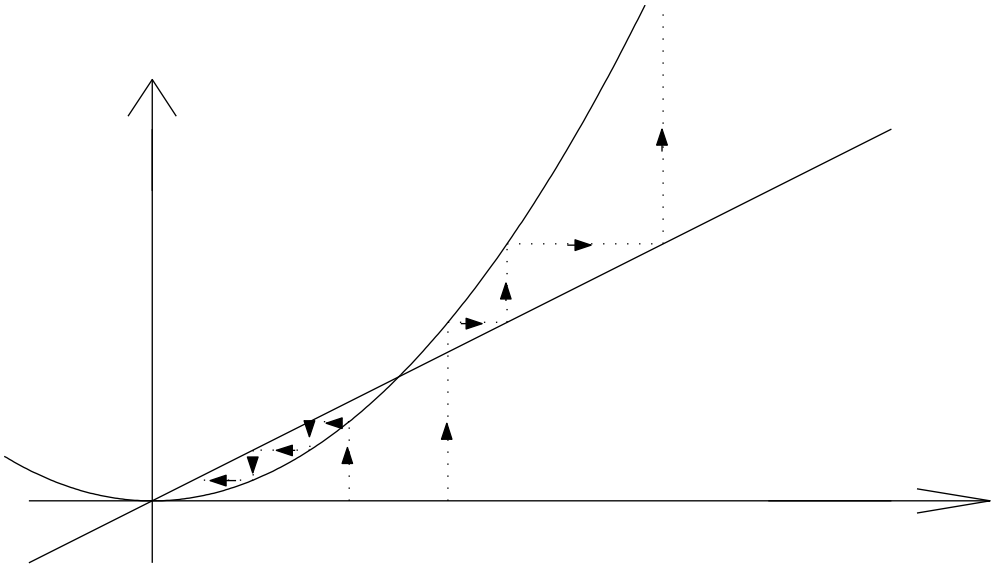

FIGURE 2.4 – Méthode de point fixe pour $g(x) = x^2$

Démonstration : Notons d'abord que la suite est définie pout tout n puisque

$$\forall x \in [a,b] \qquad g(x) \in [a,b] \ .$$

La fonction g a un point fixe et un seul dans l'intervalle $[a,b]$, car la fonction $h(x) = g(x) - x$ vérifie :
(i) h est continue sur $[a,b]$
(ii) $h'(x) = g'(x) - 1 \leq k - 1 < 0$, donc h est strictement monotone
(iii) $h(a) = g(a) - a \geq 0$ puisque $g(a) \in [a,b]$
(iv) $h(b) = g(b) - b \geq 0$ puisque $g(b) \in [a,b]$.
Soit x_∞ ce point fixe, on a :

$$x_{n+1} - x_\infty = g(x_n) - g(x_\infty) = g'(\zeta_n)(x_n - x_\infty)$$

où $\zeta_n \in [a,b]$, d'après le théorème des accroissements finis. Ainsi, en utilisant (2.3), on a :

$$|x_{n+1} - x_\infty| \leq K |x_n - x_\infty| , \tag{2.4}$$

et par récurrence,

$$|x_n - x_\infty| \leq K^n |x_0 - x_\infty| \ \forall n.$$

Puisque $0 \leq K < 1$, ceci prouve que $\lim_{n\to\infty} |x_n - x_\infty| = 0$.

Remarque 2.5 Quand une suite x_n converge vers x_∞ selon la relation (2.4), on dit que la convergence est au moins linéaire. Plus généralement, on définit :

Définition 2.1 *Soit x_n une suite convergeant vers x_∞. S'il existe un nombre p et une constante $C \neq 0$ tels que*

$$\lim_{n\to\infty} \frac{|x_{n+1} - x_\infty|}{|x_n - x_\infty|^p} = C,$$

on dit que la convergence est d'ordre p ; C est la constante d'erreur asymptotique.

S'il existe une constante C et un réel positif p tel que

$$|x_{n+1} - x_\infty| \leq C\,|x_n - x_\infty|^p$$

pour tout n assez grand, on dit que la convergence est au moins d'ordre p.

Remarque 2.6 La convergence vers 0 de $a_n - b_n$ dans l'algorithme 2.1 (la dichotomie) est linéaire.

Remarque 2.7 Il est clair qu'une convergence est d'autant plus rapide que son ordre est grand. En effet, si $|x_n - x_\infty|$ est petit, $|x_n - x_\infty|^2$ est encore plus petit !...

La notion de vitesse de convergence est évidemment essentielle en analyse numérique : afin de diminuer le temps de calcul (et donc le coût !), toutes choses étant égales par ailleurs, on choisira toujours l'algorithme convergeant le plus rapidement. Cependant, il est bien rare que "toutes les choses soient égales par ailleurs". Voir le paragraphe suivant à ce propos.

2.2.3.2 Méthode de Newton

Revenons un instant à la méthode du point fixe. D'après la formule de Taylor à l'ordre 2, on a, en supposant g suffisamment régulière :

$$x_{n+1} - x_\infty = g(x_n) - g(x_\infty) = (x_n - x_\infty)g'(x_\infty) + \frac{1}{2}(x_n - x_\infty)^2 g''(\zeta_n)$$

où $\zeta_n \in [x_n, x_\infty]$. Pour n grand, on a donc

$$x_{n+1} - x_\infty \simeq (x_n - x_\infty)g'(x_\infty)$$

et la vitesse de convergence sera d'autant plus grande que $g'(x_\infty)$ est plus petit. Le cas le plus favorable est celui où $g'(x_\infty) = 0$.
Si M_2 est majorant de $g''(x)$ sur $[a, b]$, on a alors :

$$|x_{n+1} - x_\infty| \leq \frac{M_2}{2}\,|x_n - x_\infty|^2 .$$

La convergence est alors au moins d'ordre 2 !
Si on revient à l'algorithme 2.3 de Newton, on voit qu'il s'agit en fait d'un algorithme de point fixe pour la fonction $g(x) = x - \frac{f(x)}{f'(x)}$. La dérivée de g est donnée par :

$$g'(x) = 1 - \frac{f'(x)}{f'(x)} + \frac{f(x)\,f''(x)}{f'^2(x)} = \frac{f(x)\,f''(x)}{f'^2(x)}.$$

Si $f'(x_\infty) \neq 0$, on a alors, puisque $f(x_\infty) = 0$:

$$g'(x_\infty) = 0.$$

Ceci montre que la méthode de Newton converge de façon quadratique...si elle converge !
Pour s'assurer de sa convergence, on peut essayer d'appliquer à g le théorème 2.1. Il est clair que l'hypothèse (2.3) est vérifiée sur un voisinage suffisamment petit de x_∞ (puisque $g'(x_\infty) = 0$ et on suppose g' continue). L'hypothèse plus difficile à vérifier est que g envoie un voisinage $[a, b]$ de x_∞ dans lui-même. On a en fait le résultat suivant :

Théorème 2.2 *Soit f deux fois continument différentiable sur un intervalle ouvert de centre x_∞ vérifiant :*

$$f(x_\infty) = 0,\ f'(x_\infty) \neq 0.$$

Alors, si x_0 est choisi assez près de x_∞, la méthode de Newton :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\ \forall n \geq 0$$

produit une suite x_n convergeant au moins quadratiquement vers x_∞.

Avant de démontrer ceci, faisons quelques remarques :

Remarque 2.8 Toute l'ambiguité dans ce résultat provient de "si x_0 est choisi assez près de x_∞" : en effet, dans la pratique, il n'y a généralement aucun moyen de savoir dans quelle mesure x_0 est assez voisin de x_∞.

C'est un des inconvénients de la méthode de Newton : une initialisation faite au hasard conduit souvent à une divergence. Ceci est suggéré par la Figure 2.5.

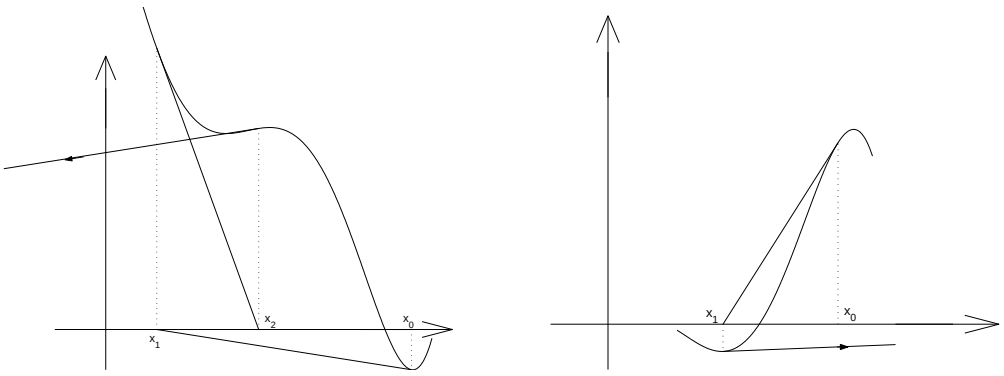

FIGURE 2.5 – *Divergence dans la méthode de Newton*

Remarque 2.9 L'hypothèse $f'(x_\infty) \neq 0$ est naturelle : étant donné le calcul du quotient $\frac{f(x_n)}{f'(x_n)}$, des valeurs petites de $f'(x_n)$ conduisent à une perte de précision. Si $f'(x_\infty) = 0$, il se peut cependant que la méthode converge, mais en général la convergence sera au plus linéaire.

Exemple : $f(x) = x^2$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = \frac{1}{2}x_n\ ;$$

ici x_n converge linéairement vers 0.

Remarque 2.10 Afin de lever l'incertitude de l'initialisation dans la méthode de Newton, on peut utiliser des théorèmes plus globaux comme celui assurant la convergence dès que $x_0 \in [a,b]$ si $f : [a,b] \to R$ est deux fois continuement différentiable et :
i) $f(a)\,f(b) \leq 0$
ii) $f'(x_\infty) \neq 0\ \forall x \in [a,b]$
iii) $f''(x) \geq 0\ (\leq 0)\ \forall x \in [a,b]$
iv) $\frac{|f(a)|}{|f'(a)|} < b-a,\ \frac{|f(b)|}{|f'(b)|} < b-a.$

Démonstration du théorème 2.2 :
Nous allons montrer qu'il existe $\varepsilon > 0$ tel que la fonction

$$g(x) = x - \frac{f(x)}{f'(x)}$$

satisfasse les hypothèses du théorème 2.1 dans l'intervalle $[x_\infty - \varepsilon, x_\infty + \varepsilon]$.
On a :

$$g'(x) = \frac{f(x) f''(x)}{f'^2(x)}$$

Puisque $f'(x_\infty) \neq 0$, il existe ε_1 tel que :

$$\forall x \in [x_\infty - \varepsilon_1, x_\infty + \varepsilon_1],\ f'(x) \neq 0.$$

Il en résulte que g' est continue sur cet intervalle. Puisque $g'(x_\infty) = 0$, il existe $\varepsilon_2 \leq \varepsilon_1$ tel que :

$$\forall x \in [x_\infty - \varepsilon_2, x_\infty + \varepsilon_2],\ |g'(x)| \leq \frac{1}{2} < 1.$$

Par ailleurs, d'après le théorème des accroissements finis :

$$|g(x) - g(x_\infty)| \leq |x - x_\infty| \max_{\zeta \in [x, x_\infty]} |g'(\zeta)| \leq \frac{1}{2} |x - x_\infty|,\ \text{si}\ |x - x_\infty| \leq \varepsilon_2.$$

puisque $g(x_\infty) = x_\infty$, ceci prouve que g envoie $[x_\infty - \varepsilon_2, x_\infty + \varepsilon_2]$ dans lui-même ($|x - x_\infty| \leq \varepsilon_2 \Longrightarrow |g(x) - x_\infty| \leq \varepsilon_2$).
D'après le théorème 2.1 appliqué à g avec $a = x_\infty - \varepsilon_2$, $b = x_\infty + \varepsilon_2$, la suite définie par

$$\begin{aligned} x_0 &\in [x_\infty - \varepsilon_2, x_\infty + \varepsilon_2] \\ x_{n+1} &= g(x_n) = x_n - \frac{f(x_n)}{f'(x_n)} \end{aligned}$$

converge vers x_∞. Les considérations faites précédemment montrent que la convergence est quadratique, le seul point à vérifier étant le comportement de la dérivée seconde de g. Or, on vérifie que

$$g''(x_\infty) = \frac{f''(x_\infty)}{f'(x_\infty)}.$$

Si f est deux fois continument dérivable, g'' est bornée sur un voisinage de x_∞ et on peut directement appliquer les remarques précédentes.

2.2.3.3 Méthode de la sécante

L'algorithme 2.2 correspondant est défini par :

$$x_{n+1} := x_n - f(x_n) \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} = \frac{x_{n-1} f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

soit encore

$$x_{n+1} - x_\infty = \frac{(x_{n-1} - x_\infty) f(x_n) - (x_n - x_\infty) f(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

et, en utilisant $f(x_\infty) = 0$

$$x_{n+1} - x_\infty = (x_n - x_\infty)(x_{n-1} - x_\infty) \frac{\left[\frac{f(x_n) - f(x_\infty)}{x_n - x_\infty} - \frac{f(x_{n-1}) - f(x_\infty)}{x_{n-1} - x_\infty}\right]}{f(x_n) - f(x_{n-1})}.$$

Utilisant les notations standard des différences divisées (cf chapitre suivant), soit :

$$\begin{aligned} f[x] &: = f(x) \\ f[x,y] &: = \frac{f(y)-f(x)}{y-x} = \frac{f[y]-f[x]}{y-x} \\ f[a,x,y] &: = \frac{f[x,y]-f[a,x]}{y-a}, \text{ etc...,} \end{aligned}$$

la formule ci-dessus s'écrit :

$$x_{n+1} - x_\infty = (x_n - x_\infty)(x_{n-1} - x_\infty) \frac{f[x_{n-1}, x_\infty, x_n]}{f[x_{n-1}, x_n]}. \tag{2.5}$$

D'après le théorème des accroissements finis

$$f[x_{n-1}, x_n] = f'(\zeta_n) \text{ où } \zeta_n \in [x_{n-1}, x_n]. \tag{2.6}$$

Nous verrons de même au chapitre suivant que :

$$f[x_{n-1}, x_\infty, x_n] = \frac{1}{2} f''(\eta_n) \text{ où } \eta_n \in \text{ plus petit intervalle contenant } x_{n-1}, x_n, x_\infty. \tag{2.7}$$

Si on suppose que x_n prend ses valeurs dans un intervalle J sur lequel :

$$\forall x \in J \ |f''(x)| \leq M, \ |f'(x)| \geq m > 0, \tag{2.8}$$

d'après (2.5), (2.6), (2.7) et en posant :

$$e_n := |x_n - x_\infty|,$$

on a :

$$e_{n+1} = c_n e_n e_{n-1}, \quad c_n \leq \frac{1}{2} \frac{M}{m} \tag{2.9}$$

(et $\lim_{n\to\infty} c_n = \frac{1}{2} \frac{f''(x_\infty)}{f'(x_\infty)}$ d'après ce qui suit). Si f est deux fois continument dérivable sur un intervalle $[x_\infty - \alpha, x_\infty + \alpha]$ et si $f'(x_\infty) \neq 0$, on montre, comme dans le cas de la méthode de Newton qu'il existe $J = [x_\infty - \varepsilon, x_\infty + \varepsilon]$ $(0 < \varepsilon \leq \alpha)$ et $M, \ m > 0$ tel que (2.8) soit vérifié. Quitte à resteindre encore ε, en utilisant (2.9), on montre que si $x_0 \in J$, alors x_n appartient à J pour tout n, ceci par récurrence. Dans ce cas, l'estimation (2.9) est valable pour tout n. On déduit de cette estimation que e_n tend vers 0 et que la convergence est surlinéaire.
En effet, remarquons d'abord qu'on peut se "ramener" au cas $C = 1$ dans (2.9) en posant $\varepsilon_n = C e_n$. Multipliant alors (2.9) par C, on a

$$\varepsilon_{n+1} \leq (C e_n)(C e_{n-1}) = \varepsilon_n \varepsilon_{n-1}. \tag{2.10}$$

Soit alors p la racine positive de $p^2 - p - 1 = 0$, soit $p = \frac{(1+\sqrt{5})}{2} = 1,618...$ (le nombre d'or !) et $K = \max\left(\varepsilon_0, \sqrt[p]{\varepsilon_1}\right)$. Montrons par récurrence que

$$\varepsilon_n \leq K^{p^n}. \tag{2.11}$$

Ceci est vrai pour $n = 0, 1$ d'après le choix de K. Si c'est vrai jusqu'au rang n, d'après (2.10), on a :

$$\varepsilon_{n+1} \leq K^{p^n} K^{p^{n-1}} = K^{p^{n-1}(1+p)} = K^{p^{n+1}}, \text{ d'après le choix de } p;$$

donc (2.10) est vrai pour tout n.
Ainsi, si on choisit $K < 1$ (c'est-à-dire x_0 et x_1 suffisamment voisins de x_∞), ε_n et donc e_n tendent vers 0 quand n tend vers l'infini, d'après (2.11).
Cette même relation (2.11) suggère que la convergence est au moins d'ordre p. On le montre (cf. Définition 2.1), en remarquant que (2.10) implique :

$$\frac{\varepsilon_{n+1}}{\varepsilon_n^p} \leq \varepsilon_n^{1-p}\varepsilon_{n-1} = \left(\frac{\varepsilon_n}{\varepsilon_{n-1}^p}\right)^{1-p} \text{ car } p(1-p) = -1.$$

Ainsi $\frac{\varepsilon_n}{\varepsilon_{n-1}^p}$ reste inférieur à la suite Y_n définie par

$$\begin{aligned} Y_1 &= \frac{\varepsilon_1}{\varepsilon_0^p}, \\ Y_{n+1} &= Y_n^{1-p} \end{aligned}$$

qui, comme on le vérifie aisément converge vers 1 puisque $|1-p| < 1$.
Donc, pour n assez grand

$$\varepsilon_{n+1} \leq Y_n \varepsilon_n^p \leq (1+\varepsilon)\,\varepsilon_n^p.$$

Ceci prouve que la convergence est au moins d'ordre $p = 1,618...$.

Théorème 2.3 *Soit f deux fois continument différentiable sur un intervalle ouvert de centre x_∞ vérifiant :*

$$f(x_\infty) = 0,\ f'(x_\infty) \neq 0.$$

Alors, si x_0, x_1 sont choisis assez près de x_∞, l'algorithme de la sécante défini par :

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\ \forall n \geq 1$$

converge vers x_∞ et la convergence est au moins d'ordre $p = 1,618...$.

2.2.4 Comparaison des algorithmes

Pour comparer les algorithmes, il est important de tenir compte de la présence ou non des facteurs suivants :
- assurance de la convergence
- vitesse de la convergence
- stabilité de l'algorithme - précision des résultats
- efficacité des calculs (ex : nombre de fonctions à calculer à chaque itération).

2.2.4.1 Méthode de dichotomie

Avantages :
- la convergence est assurée
- on a un encadrement de la solution
- un seul calcul de fonction à chaque itération.

Inconvénients :
- vitesse de convergence linéaire, donc lente
- sensible aux erreurs d'arrondi ; exemple : la fonction $f(x) = e^x - 1 - x - \frac{x^2}{2}$ s'annule théoriquement en $x = 0$ seulement. Numériquement, elle change de signe un grand nombre de fois autour de $x = 0$.

2.2.4.2 Méthode de Newton

Avantages :
- converge rapidement quand elle converge (ce qui compense largement le dernier inconvénient)
- relativement stable et peu sensible aux erreurs d'arrondis si $f'(x_\infty)$ n'est pas trop petit.

Inconvénients :
- peut diverger ou converger vers un autre zéro que celui cherché si la donnée initiale est mal choisie
- nécessite le calcul de la dérivée d'une fonction, ce qui est numériquement difficile si on ne la connait pas explicitement
- chaque étape nécessite deux évaluations de fonctions.

2.2.4.3 Méthode de la sécante

Avantages :
- nécessite une seule évaluation de fonction à chaque étape
- convergence relativement rapide, bien que moins que celle de Newton.

Inconvénients :
- comme la méthode de Newton, peut diverger si les données initiales sont mal choisies
- le calcul de $f(x_n) - f(x_{n-1})$ peut produire des erreurs de chute.

2.2.4.4 Méthode du point fixe

Inconvénients :
- convergence souvent difficile à réaliser en pratique
- certains points fixes -dits instables- ne peuvent être atteints
- convergence lente de type linéaire, en général.

Conclusion : cette dernière méthode a surtout un interêt théorique : son analyse mathématique est relativement aisée et nous a permis d'en déduire celle de la méthode de Newton qui en est en fait un cas particulier. En pratique, elle est souvent utilisée dans de tels cas particuliers. Nous verrons à plusieurs reprises dans la suite qu'elle joue un rôle considérable pour adapter des algorithmes linéaires à des situations non linéaires. Elle est donc fondamentale en tant que concept. D'autre part, couplée avec des techniques d'accélération de la convergence (cf. paragraphe suivant), elle peut être intéressante.

La méthode de Newton a généralement la faveur des utilisateurs : elle est effectivement très rapide et est certainement à conseiller lorsque le calcul de f' est aisé et lorsque des remarques simples permettent de choisir à coup sûr la donnée initiale dans le domaine de convergence. Pour cela, elle peut être, par exemple, précédée d'une application de quelques itérations de la méthode de dichotomie.

Si on veut une méthode d'application plus universelle, la méthode de la sécante est plutôt à conseiller puisqu'elle nécessite seulement la connaissance de f. Elle est relativement rapide et ne nécessite qu'une évaluation de fonction à chaque itération. Ce dernier point peut même la faire considérer comme plus rapide que la méthode de Newton. En effet, si on pose :

$$u_n := x_{2n},$$

le passage de u_n à u_{n+1} nécessite le même travail qu'une itération de la méthode de Newton. Mais on a :

$$|u_{n+1} - x_\infty| \leq C\,|x_{2n+1} - x_\infty|^p \leq C\,|u_n - x_\infty|^{p^2}.$$

Vue ainsi, la méthode est d'ordre $p^2 = 2,618...$! Ce genre de considérations n'est pas à négliger dans une évaluation globale du temps de calcul.

Terminons par un exemple :

Résolution de $x - 0,2\sin x - 0,5 = 0$ à l'aide des quatre algorithmes différents

	Dichotomie $x_{-1} = 0,5$ $x_0 = 1,0$	Sécante $x_{-1} = 0,5$ $x_0 = 1,0$	Newton $x_0 = 1$	Point fixe $x_0 = 1$ $x = 0,2\sin x + 0,5$
1	0,75	0,5	0,5	0,50
2	0,625	0,61212248	0,61629718	0,595885
3	0,5625	0,61549349	0,61546820	0,612248
4	0,59375	0,61546816	0,61546816	0,614941
5	0,609375			0,61538219
6	0,6171875			0,61545412
7	0,6132812			0,61546587
8	0,6152343			0,61546779
9	0,6162109			0,61546810
10	0,6157226			0,61546815
11	0,6154785			
12	0,6153564			
13	0,6154174			
14	0,6154479			
15	0,6154532			
16	0,61547088			
17	0,61546707			
18	0,61546897			
19	0,615468025			
20	0,615468502			

Cet exemple confirme les remarques générales. La méthode de Newton est la plus rapide. Ici, la méthode de la sécante l'est presque autant. Les deux autres sont de type linéaire : celle du point fixe est plus rapide ce qui est cohérent avec le fait que son taux de convergence est
$\max_x \left|\frac{d}{dx}(0,2\sin x + 0,5)\right| = 0,2$ au lieu de $0,5$ pour la première.

2.2.5 Accélération de la convergence

D'une manière générale, étant donnée une suite x_n convergeant vers x_∞, accélérer sa convergence consisite à la remplacer par une suite $\widehat{x}_n$ convergeant plus vite vers x_∞, c'est-à-dire vérifiant :

$$\lim_{n\to\infty} \frac{\widehat{x}_n - x_\infty}{x_n - x_\infty} = 0.$$

Par exemple, si x_n converge linéairement, on cherchera à construire $\widehat{x}_n$ convergeant quadratiquement.

2.2.5.1 Méthode de relaxation

Elle est basée sur une idée très simple. Considérons une méthode de point fixe $x_{n+1} = g(x_n)$ qui ne converge pas ou très lentement vers un point fixe x^*. L'équation qu'on cherche à résoudre $x = g(x)$ peut également s'écrire

$$\alpha x + x = g(x) + \alpha x \tag{2.12}$$

et ce pour tout α réel. L'écriture de l'équation sous la forme (2.12) suggère d'utiliser une nouvelle méthode de point fixe :

$$x_{n+1} = \frac{g(x_n) + \alpha x_n}{\alpha + 1} := G(x_n). \tag{2.13}$$

Maintenant, d'après le Théorème 2.1, cette méthode (2.13) va converger (en choisissant d'initialiser assez près de la solution) dès que

$$|G'(x^*)| = \left| \frac{g'(x^*) + \alpha}{\alpha + 1} \right| < 1$$

et ce d'autant plus rapidement que $\left| \frac{g'(x^*) + \alpha}{\alpha + 1} \right|$ est petit. Comme on est parfaitement libre de choisir le paramètre α (qui s'appelle ici **paramètre de relaxation**), l'idée est de le prendre le plus proche possible de $-g'(x^*)$. Bien sûr, la valeur exacte de $g'(x^*)$ n'est en général pas connue, mais par encadrement on peut en avoir des valeurs aprochées convenables, qu'on peut d'ailleurs réactualiser éventuellement au cours des itérations. En conclusion cette méthode est très simple d'utilisation et peut rendre de précieux services.

2.2.5.2 Méthode du Δ^2 d'Aitken

Afin d'illustrer la méthode, commençons par considérer une suite x_n convergeant vers x_∞ et telle que :

$$x_{n+1} - x_\infty = k\,(x_n - x_\infty) \text{ où } 0 < k < 1.$$

Les nombres k et x_∞ peuvent être exprimés en fonction des valeurs de la suite x_n à l'aide des deux équations :

$$\begin{aligned} x_{n+1} - x_\infty &= k\,(x_n - x_\infty)\,, \\ x_{n+2} - x_\infty &= k\,(x_{n+1} - x_\infty)\,. \end{aligned}$$

On obtient par différence :

$$x_{n+2} - x_{n+1} = k\,(x_{n+1} - x_n)\,,$$

et en substituant dans la première équation mise sous la forme

$$x_\infty = x_n + \frac{x_{n+1} - x_n}{1 - k}$$

on a :

$$x_\infty = x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} + x_n - 2x_{n+1}}.$$

En utilisant les notations des différences finies, soit

$$\begin{aligned} \Delta x_n &= x_{n+1} - x_n \\ \Delta^2 x_n &= \Delta x_{n+1} - \Delta x_n = (x_{n+2} - x_{n+1}) - (x_{n+1} - x_n)\,, \end{aligned}$$

ceci s'écrit encore :

$$x_\infty = x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}.$$

La méthode tire son nom de cette formule. Elle montre ici que la limite x_∞ cherchée peut être exactement connue à l'aide de x_{n+2}, x_{n+1}, et x_n ; n quelconque.

L'idée d'Aitken est de généraliser cette remarque aux suites convergeant linéairement c'est-à-dire telles que :

$$x_{n+1} - x_\infty = k_n (x_n - x_\infty)$$

avec

$$\lim_{n\to\infty} k_n = k \in [0, 1[.$$

Pour une telle suite, le coefficient k_n est "presque" constant si n est grand et on peut alors faire le calcul précédent pour se convaincre que :

$$\widehat{x}_n = x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}$$

doit être très voisin de x_∞. On a en effet :

Théorème 2.4 *Soit une suite x_n vérifiant pour tout entier n, $x_n \neq x_\infty$ et $x_{n+1} - x_\infty = (k + \varepsilon_n)(x_n - x_\infty)$ avec $0 \leq |k| < 1$ et $\lim_{n\to\infty} \varepsilon_n = 0$. Alors la suite définie par :*

$$\widehat{x}_n := x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}$$

vérifie :

$$\lim_{n\to\infty} \frac{\widehat{x}_n - x_\infty}{x_n - x_\infty} = 0.$$

Démonstration : Posons $e_n = x_n - x_\infty$. On a :

$$\begin{aligned} \Delta^2 x_n &= e_{n+2} - 2e_{n+1} + e_n = e_n [(k + \varepsilon_{n+1})(k + \varepsilon_n) - 2(k + \varepsilon_n) + 1] \\ &= e_n \left[(k-1)^2 + \eta_n\right] \text{ où } \eta_n = k(\varepsilon_{n+1} + \varepsilon_n) + \varepsilon_{n+1}\varepsilon_n - 2\varepsilon_n. \end{aligned}$$

Puisque η_n tend vers 0 et $e_n \neq 0$, $\Delta^2 x_n$ est différent de 0 pour n grand et $\widehat{x}_n$ est donc bien défini au moins pour n assez grand. D'autre part

$$\widehat{x}_n - x_\infty = e_n \left[1 - \frac{(k - 1 + \varepsilon_n)^2}{(k-1)^2 + \eta_n}\right].$$

Mais l'expression entre crochets tend vers 0 quand n tend vers l'infini ce qui prouve le résultat.

Application : Algorithme de Steffensen Soit x_n définie par :

$$x_{n+1} = g(x_n) \ \forall n \geq 0$$

où g et x_0 vérifient les hypothèses du théorème 2.1. On sait qu'alors x_n converge au moins linéairement vers x_∞. On peut accélérer cette convergence en utilisant la méthode de Δ^2 d'Aitken. Il est alors plus efficace de repartir avec la nouvelle valeur $\widehat{x}_n$ à chaque nouvelle itération. Ceci donne l'algorithme suivant dit de Steffensen.

Algorithme 2.5 : x_0 donné

$$\left[\begin{array}{c} \text{Pour } n = 0, 1, 2, ... \\ y_n := g(x_n), \ z_n := g(y_n) \\ x_{n+1} := x_n - \dfrac{(y_n - x_n)^2}{z_n - 2y_n + x_n} \end{array}\right.$$

Remarque 2.11 Cet algorithme est à nouveau un algorithme de point fixe

$$\begin{aligned} x_{n+1} &= \psi(x_n) \text{ avec} \\ \psi(x) &= x - \frac{(g(x)-x)^2}{g(g(x)) - 2g(x) + x}. \end{aligned}$$

On peut montrer que si $g'(x_\infty) \neq 0$, alors $\psi'(x_\infty) = 0$. On sait qu'alors l'algorithme associé à ψ converge de façon quadratique. Ainsi, par rapport à l'algorithme associé à g :
– on accélère la convergence quand elle a lieu
– on a un procédé convergeant, même si $|g'(x_\infty)| \geq 1$.

Exemple 2.5 : Nous avons vu que si $g(x) = x^2$, la suite définie par

$$x_{n+1} = x_n^2$$

converge si $|x_0| < 1$ vers le point fixe 0. Le point fixe 1 n'est jamais atteint (sauf dans le cas irréaliste où x_0 est exactement égal à 1). Ceci est cohérent avec le fait que $g'(1) = 2 > 1$.
Calculons la fonction ψ dans ce cas :

$$\psi(x) = x - \frac{\left(x^2 - x\right)^2}{x^4 - 2x^2 + x} = x - \frac{x^2(x-1)^2}{x(x-1)(x^2+x-1)}$$

$$\psi(x) = \frac{x\left(x^2+x-1\right) - x(x-1)}{x^2+x+1} = \frac{x^3}{x^2+x-1} = \frac{x^3}{(x-r_1)(x-r_2)}$$

où $r_{1,2} = \dfrac{-1 \pm \sqrt{5}}{2}$.
Puisque $\psi(x) \sim x^3$ pour x voisin de 0, si x_0 est choisi assez petit, la suite x_n se comportera sensiblement comme :

$$Y_{n+1} = Y_n^3,$$

d'où convergence d'ordre 3 vers 0.
Si x_0 est choisi voisin de 1, puisque $\psi'(1) = 0$ et $\psi''(1) \neq 0$, la suite associée à ψ converge de façon quadratique vers 1. Ici, il y a donc un gain considérable par rapport à g.

Remarque 2.12 Il faut noter que, si l'algorithme 2.5 converge plus vite que celui associé à g, chaque itération comporte deux évaluations de la fonction : on y met le prix. Ainsi, une comparaison plus juste consiste à mettre sur le même plan la suite x_n définie par l'algorithme 2.5 et la suite Y_n définie par :

$$Y_{n+1} = g(g(Y_n)),\ Y_0 = 1$$

où chaque itération comporte aussi deux évaluations de g. Dans l'exemple précédent, on a $Y_{n+1} = Y_n^4$, qui, au voisinage de 0, est d'ordre 4 et donc meilleure que celle donnée par ψ qui est d'ordre 3. Bien sûr, on a dans ce cas $g'(x_\infty) = 0$!

Remarque 2.13 D'une manière générale, il est illusoire d'espérer obtenir des méthodes convergeant plus rapidement sans alourdir le travail à chaque itération, soit en augmentant le nombre d'évaluation de fonctions, soit en évaluant plus de dérivées. Signalons par exemple l'algorithme de Richmond d'ordre 3 pour résoudre $f(x) = x$ donné par :

$$x_{n+1} = x_n - \frac{2f(x_n)f'(x_n)}{f'(x_n)^2 - f(x_n)f''(x_n)}.$$

Cet algorithme parfois utilisé converge bien sûr rapidement, mais nécessite l'évaluation de f, f', f'' à chaque étape.

Théoriquement, il est presque toujours possible d'imaginer des méthodes d'ordre aussi élevé qu'on veut. Du point de vue pratique, elles sont le plus souvent irréalistes. D'autre part, il faut se convaincre qu'une méthode d'ordre 2 est déjà très rapide et qu'en général, il est pratiquement difficile de faire mieux. On peut se reporter à l'exemple traité précédemment : en 4 itérations, la méthode de Newton donne le résultat avec 8 chiffres significatifs ! Un numéricien raisonnable s'en contentera.

2.3 Fonctions à valeurs complexes

2.3.1 Cas des équations polynômiales

Bien que les équations polynômiales (ou algébriques) puissent être résolues à l'aide des méthodes décrites plus haut, elles méritent une attention particulière dans la mesure où elles interviennent très fréquemment dans les applications. Nous verrons, d'une part, comment profiter de la structure polynômiale dans certaines des méthodes générales, d'autre part, nous décrirons de nouveaux algorithmes permettant de trouver les racines réelles ou complexes d'un polynôme.
Rappelons d'abord le théorème fondamental de l'algèbre :

Théorème 2.5 (D'Alembert) *Tout polynôme de degré n admet exactement n zéros réels ou complexes (avec la convention habituelle que des zéros de multiplicité r sont comptés r fois).*

2.3.1.1 Localisation des zéros

La localisation des zéros d'un polynôme est un problème difficile. On peut utiliser des techniques d'analyse complexe (indice ou Théorème de Rouché). Des renseignements peuvent également être obtenus sur les zéros réels à l'aide de la règle des signes de Descartes :

Proposition 2.2 *Soit λ le nombre de zéros strictement positifs d'un polynôme.*
Soit ν le nombre de changements de signe des coefficients.
Alors :

$$\lambda \leq \nu \tag{2.14}$$

$$\nu - \lambda \text{ est pair} \tag{2.15}$$

Exemple 2.6 : Soit $P(x) = x^4 + 4x^2 - x - 1$.
– $\nu = 1$ donc $\lambda \leq 1$
– comme $\nu - \lambda$ est pair, on a nécessairement $\lambda = 1$.
Par conséquent P a une racine positive et une seule.

Les racines négatives peuvent être examinées aussi puisqu'elles sont les racines positives de $P(-x)$:

$$P(-x) = x^4 + 4x^2 + x - 1$$

Par le même raisonnement, P a une racine négative et une seule.
Conclusion : P a
– une racine positive
– une racine négative
– deux racines complexes conjuguées.
Plusieurs théorèmes permettent aussi de localiser les zéros d'un polynôme. Citons en par exemple un :

Théorème 2.6 *Soit $P(x) = a_0 + a_1 x + ... a_n x^n$, $a_n \neq 0$. Soit $r := 1 + \max_{0 \leq k \leq n-1} \left| \frac{a_k}{a_n} \right|$.*
Alors tout zéro x_0 de P vérifie $|x_0| \leq r$.

2.3.1.2 Evaluation d'un polynôme : algorithme de Hörner

L'évaluation d'un polynôme en un point ξ peut se faire en calculant chacun des produits $a_k\xi^k$ et en faisant leur somme. Cette technique est généralement peu utilisée à cause des erreure de chute qu'elle engendre et à cause du nombre trop important d'opérations élémentaires qu'elle met en jeu, surtout quand le degré est élevé.
Voici par exemple ce que donne une évaluation du polynôme :

$$P(x) = (1-x)^6 = 1 - 6x + 15x^2 - 20x^3 + 15x^4 - 6x^5 + x^6$$

à l'aide de sa forme développée. Le calcul a été fait sur un ordinateur CDC 6500 travaillant avec 14 chiffres significatifs. Les points ont été relevés par des segments de droite à l'aide d'un traceur.

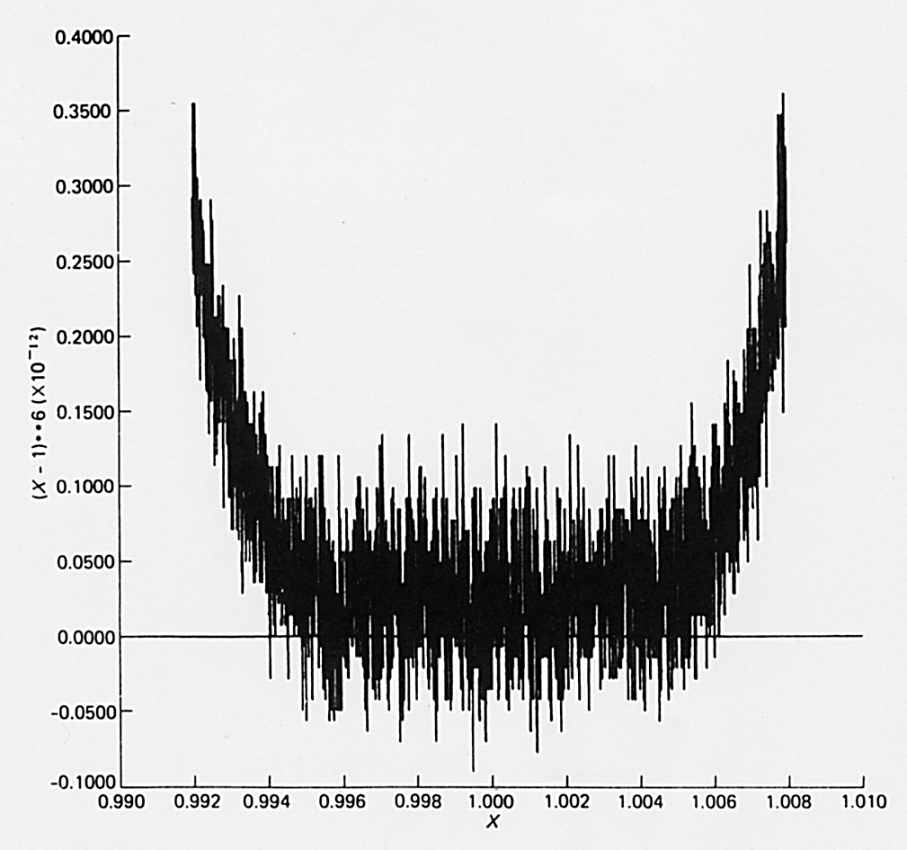

FIGURE 2.6 – Photo tirée du livre "Elementary Numerical Analysis" par S.D. Conte et Carl de Boor.

Remarque 2.14 Imaginer ce que la méthode de dichotomie couplée avec un tel calcul de la fonction P peut donner comme résultat !

On préfère généralement utiliser l'algorithme de Hörner qui repose sur la factorisation suivante du polynôme :

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_2x^2 + a_1x + a_0,$$

soit :

$$P(x) = a_0 + x\left(a_1 + x\left(a_2 + x\left(a_3 + ... + x\left(a_{n-2} + xa_{n-1} + xa_n\right)\right)\ldots\right)\right)$$

d'où :

Algorithme 2.6 : Évaluation de $P(\xi)$

$$\left[\begin{array}{l} b_n := a_n \\ \text{de } i = n-1 \text{ à } 0 \\ b_i := a_i + \xi b_{i+1} \\ P(\xi) := b_0. \end{array}\right.$$

Comme sous-produit intéressant, on obtient le polynôme $P_1(x)$ tel que :

$$P(x) = (x-\xi) P_1(x) + b_0, \tag{2.16}$$

c'est-à-dire le quotient de la division de $P(x)$ par $(x-\xi)$. Il est donné par :

$$P_1(x) = b_n x^{n-1} + b_{n-1} x^{n-2} + ... + b_2 x + b_1.$$

En effet, le quotient de x^k dans $(x-\xi) P_1(x) + b_0$ est égal à :

$$\begin{array}{rl} b_k - \xi b_{k+1} = a_k & \text{pour } k = 1, \ldots, n-1\,, \\ b_n = a_n & \text{pour } k = n \\ b_0 = a_0 & \text{pour } k = 0 \end{array}$$

Si on dérive la formule (2.16), on obtient

$$P'(x) = P_1(x) + (x-\xi) P_1'(x)\,.$$

En particulier :

$$P'(\xi) = P_1(\xi)\,. \tag{2.17}$$

On peut donc utiliser les b_i précédents pour évaluer la dérivée de P au même point ξ, toujours à l'aide de l'algorithme de Hörner.

Algorithme 2.7 : Évaluation de $P'(\xi)$

$$\left[\begin{array}{l} c_n := b_n \\ \text{de } i = n-1 \text{ à } 1 \\ c_i := b_i + \xi c_{i+1} \\ P'(\xi) := c_1. \end{array}\right.$$

Ainsi, l'algorithme de Newton prend la forme suivante pour un polynôme $P(x) = a_0 + a_1 x + ... + a_n x^n$.

Algorithme 2.8 : Résolution de $P(x) = 0$ par la méthode de Newton à partir d'une valeur approchée x_0 d'un zéro réel.

$$\left[\begin{array}{l} \text{Pour } m = 0, 1, ... \\ \quad \xi := x_m \\ \quad b_n := a_n \ ; \ c_n := a_n \ ; \\ \quad \text{Pour } i = n-1 \text{ à } 1 \\ \quad\quad b_i := a_i + \xi b_{i+1} \ ; \ c_i := b_i + \xi c_{i+1} \ ; \\ \quad b_0 := a_0 + \xi b_1 \ ; \\ \quad x_{m+1} := x_m - \frac{b_0}{c_1}. \end{array}\right.$$

Remarque 2.15 Comme dans le cas général, il faut s'attendre à des difficultés lorsque $P'(x_\infty) = 0$, c'est-à-dire lorsque le zéro cherché est multiple.

Remarque 2.16 La formule (2.16) montre, puisque $P(x_\infty) = 0$, que :

$$P(x) = (x - x_\infty) P_1(x)$$

où une approximation de $P_1(x)$ est obtenue à l'aide des derniers coefficients b_i calculés. Pour calculer une autre racine de P, on peut utiliser ce polynôme P_1 de degré moindre et lui appliquer la même méthode (P_1 est appelé polynôme réduit).

Evidemment, les coefficients de ce polynôme ne sont connus que de façon approchée ce qui entachera d'erreur tous les calculs subséquents. On constate, d'une manière générale, que ces erreurs sont moins importantes lorsqu'on calcule les zéros dans l'ordre croissant de leur valeur absolue.

2.3.1.3 Détermination de tous les zéros

Pour pallier les inconvénients précités, Maehly a proposé une méthode qui s'avère efficace au niveau de la précision des résultats : supposons avoir déjà calculé les r zéros $\xi_1, \xi_2, ..., \xi_r$. Le polynôme réduit est alors :

$$P_r(x) = \frac{P(x)}{(x - \xi_1)(x - \xi_2)...(x - \xi_r)}.$$

La méthode de Newton appliquée à P_r, soit

$$x_{i+1} := x_i - \frac{P_r(x_i)}{P_r'(x_i)}$$

peut être mise en oeuvre à l'aide de la formule :

$$x_{i+1} := x_i - \frac{P(x_i)}{P'(x_i) - \sum_{k=1}^{r} \frac{P(x_i)}{(x_i - \xi_k)}}$$

qui se démontre aisément. L'avantage de cette méthode est qu'elle converge quadratiquement vers ξ_{j+1} même si les $\xi_k,\ k = 1, ..., r$, ne sont pas des zéros de P. Elle n'est donc pas sensible aux erreurs précédentes sur les $\xi_k,\ k = 1, ..., r$.

Remarque 2.17 Les coefficients d'un polynôme sont souvent connus seulement de manière approchée (cf. remarque précédente). Il est donc important de connaître la sensibilité d'une racine à une variation des coefficients (cf. notion de conditionnement définie au chapitre précédent).

En bref, on peut dire que :
– une racine multiple est toujours mal conditionnée
– une racine simple peut l'être aussi, et ce pour des polynômes apparemment anodins.
Exemple 2.7 (dû à Wilkinson - voir l'analyse du conditionnement dans "Introduction to Numerical Analysis" par J. Stoer, R. Burlisch) :

$$P(x) = (x - 1)(x - 2)...(x - 20) = \sum_{i=0}^{20} a_i x^{20-i}.$$

P est un polynôme dont toutes les racines sont réelles et bien séparées. Cependant, on peut montrer que, pour une variation de ε du coefficient a_5, la 16ème racine subit une perturbation de

$$\varepsilon \times 3,7 \times 10^{14} !$$

ce qui veut dire qu'un calcul à 14 chiffres décimaux significatifs ne donnera aucun chiffre significatif sur ξ_{16}.

Intéressons-nous maintenant aux zéros complexes d'un polynôme P. Il est clair que si P a ses coefficients réels et si la donnée initiale x_0 est réelle, l'algorithme de Newton ne pourra converger vers un zéro non réel, même si x_0 est proche de ce zéro dans le champ complexe, ceci puisque tous les x_n sont nécessairement réels. Si on veut obtenir un zéro complexe, il faut alors a priori utiliser l'arithmétique complexe. Si on veut l'éviter, on peut préférer l'algorithme suivant qui permet de trouver les zéros complexes tout en travaillant en nombres réels.

2.3.1.4 Méthode de Bairstow

Elle part de l'observation que trouver deux zéros $\alpha,\ \beta$ complexes conjugués de $P(x)$ revient à trouver un polynôme de degré 2, $x^2 - rx - s$ admettant $\alpha,\ \beta$ comme zéros et tel que

$$P(x) = \left(x^2 - rx - s\right) Q(x).$$

Pour r et s fixés, on peut toujours écrire la division euclidienne de P par $x^2 - rx - s$ sous la forme

$$P(x) = \left(x^2 - rx - s\right) Q(x) + A(x - r) + B \tag{2.18}$$

où $\deg Q = \deg P - 2$ et A et B sont des constantes déterminées de façon unique. Ceci étant vrai pour tout r et s, on définit ainsi des fonctions $A(r,s)$ et $B(r,s)$, et le problème posé revient à déterminer r et s solutions de

$$\left\{ \begin{array}{rcl} A(r,s) & = & 0 \\ B(r,s) & = & 0. \end{array} \right. \tag{2.19}$$

Il s'agit d'un système non linéaire de deux équations à deux inconnues. Nous verrons dans le paragraphe suivant que la méthode de Newton que nous avons vue pour une équation à une variable se généralise au cas de plusieurs variables. Ainsi, appliquant cette méthode au système (2.19), on est conduit à l'itération :

$$r_0, s_0 \text{ choisis ("assez près" de la solution)}$$

$$\begin{pmatrix} r_{i+1} \\ s_{i+1} \end{pmatrix} := \begin{pmatrix} r_i \\ s_i \end{pmatrix} - \left[\begin{array}{cc} \frac{\partial A}{\partial r} & \frac{\partial A}{\partial s} \\ \frac{\partial B}{\partial r} & \frac{\partial B}{\partial s} \end{array} \right]^{-1}_{r=r_i,\ s=s_i} \begin{pmatrix} A(r_i, s_i) \\ B(r_i, s_i) \end{pmatrix},$$

où "$[]^{-1}$" signifie qu'on prend l'inverse de la matrice et donc qu'on résout un système linéaire 2×2. On a donc

$$\begin{pmatrix} r_{i+1} \\ s_{i+1} \end{pmatrix} := \begin{pmatrix} r_i \\ s_i \end{pmatrix} - \begin{pmatrix} \frac{AB_s - BA_s}{A_r B_s - B_r A_s} \\ \frac{A_r B - AB_r}{A_r B_s - B_r A_s} \end{pmatrix}_{r=r_i,\ s=s_i}. \tag{2.20}$$

(Ici on a posé $A_r = \frac{\partial A}{\partial r},\ A_s = \frac{\partial A}{\partial s}, ...$). Le calcul de A, B, A_r, A_s, B_r, B_s est mené de la façon suivante.
Posons :

$$\begin{array}{rcl} P(x) & = & a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0. \\ Q(x) & = & b_n x^{n-2} + b_{n-1} x^{n-3} + ... + b_3 x + b_2. \end{array}$$

Identifiant les coefficients des puissances de x dans (2.18), on obtient

$$\left\{ \begin{array}{l} b_n := a_n \\ b_{n-1} := a_{n-1} + r b_n \\ b_{n-2} := a_{n-2} + r b_{n-1} + s b_n \\ \vdots \quad \vdots \quad \vdots b_2 := a_2 + r b_3 + s b_4 \\ A := a_1 + r b_2 + s b_3 \\ B := a_0 + r A + s b_2. \end{array} \right. \tag{2.21}$$

Pour obtenir les dérivées de A et B, on différencie le système (2.21) par rapport à r et s. On remarque que les a_i sont indépendants de r et s et donc à dérivées nulles. Posant alors :

$$\begin{aligned} \forall k &= 2,...,n \;\; C_k = (b_k)_r \,, \;\; C_1 = A_r, \;\; C_0 = B_r \\ \forall k &= 2,...,n \;\; d_k = (b_k)_s \,, \;\; d_1 = A_s, \;\; d_0 = B_s \end{aligned}$$

on obtient :

$$\left\{ \begin{array}{ll} C_n := 0 & d_n := 0 \\ C_{n-1} := b_n & d_{n-1} := 0 \\ C_{n-2} := b_{n-1} + rC_{n-1} & d_{n-2} := b_n \\ C_{n-3} := b_{n-2} + rC_{n-2} + sC_{n-1} & d_{n-3} := b_{n-1} + rd_{n-1} \\ & d_{n-4} := b_{n-2} + rd_{n-2} + sd_{n-1} \\ ..... & ..... \\ C_2 := b_3 + rC_3 + sC_4 & d_2 := b_4 + rd_4 + sd_3 \\ C_1 := b_2 + rC_2 + sC_3 & d_1 := b_3 + rd_3 + sd_2 \\ C_0 := A + rC_1 + sC_2 & d_0 := b_2 + rd_2 + sd_1 \end{array} \right\}$$

On constate qu'en fait les deux algorithmes ci-dessus sont identiques et qu'en particulier $d_0 = C_1 \,(= B_s = A_r)$ et $A_s = d_1 = C_2$. D'où :

Algorithme 2.9 (Bairstow) : les coefficients $a_0, ..., a_n$ sont donnés. On choisit une valeur initiale (r_0, s_0)

$$\left[\begin{array}{l} \text{De } i = 0 \text{ à ... (test d'arrêt)} \\ b_n := a_n, \; b_{n-1} := a_{n-1} + r_i b_n, \; C_{n-1} := b_n, \; C_{n-2} := b_{n-1} + rC_{n-1} \\ \quad \left[\begin{array}{l} \text{de } k = n-2 \text{ à } 1 \\ b_k := a_k + r_i b_{k+1} + s_i b_{k+2} \\ C_{k-1} := b_k + r_i C_k + s_i C_{k+1} \end{array} \right. \\ b_0 := a_0 + r_i b_1 + s_i b_2 \\ r_{i+1} := r_i - \dfrac{b_1 C_1 - b_0 C_2}{C_1^2 - C_0 C_2} \\ s_{i+1} := s_i - \dfrac{C_1 b_0 - b_1 C_0}{C_1^2 - C_0 C_2}. \end{array} \right.$$

2.3.1.5 Méthode de Müller

Nous terminons ce paragraphe par un dernier algorithme qui permet de calculer tous les zéros réels ou complexes d'une équation algébrique ou même d'une équation non linéaire plus générale à inconnue réelle et complexe. C'est en fait une généralisation de la méthode de la sécante : au lieu d'interpoler la fonction f donnée à l'aide d'une fonction linéaire sur $[x_{i-1}, x_i]$, on l'interpole à l'aide du polynôme de degré 2 prenant les valeurs $f(x_{i-2})$, $f(x_{i-1})$, $f(x_i)$ respectivement aux points x_{i-2}, x_{i-1}, x_i.
On verra au chapitre suivant que ce polynôme est donné par :

$$q(x) = f(x_i) + (x - x_i) f[x_i, x_{i-1}] + (x - x_i)(x - x_{i-1}) f[x_i, x_{i-1}, x_{i-2}]$$

où $f[.]$ représente les différences divisées introduites précédemment, qui sont définies par récurrence et que nous étudierons plus amplement au chapitre 3.
Dans l'algorithme, le point x_{i+1} est alors obtenu comme l'un des zéros de $q(x)$. Ce calcul peut être conduit explicitement puisqu'il s'agit d'une équation du second degré. Auparavant, réécrivons $q(x)$ en utilisant :

$$(x - x_i)(x - x_{i-1}) = (x - x_i)^2 + (x - x_i)(x_i - x_{i-1}),$$

soit

$$q(x) = f(x_i) + C_i(x - x_i) + f[x_i, x_{i-1}, x_{i-2}](x - x_i)^2$$

avec

$$C_i = f[x_i, x_{i-1}] + (x_i - x_{i-1}) f[x_i, x_{i-1}, x_{i-2}].$$

Ainsi

$$x_{i+1} := x_i + \frac{-2f(x_i)}{C_i \pm \{C_i^2 - 4f(x_i) f[x_i, x_{i-1}, x_{i-2}]\}^{\frac{1}{2}}}$$

où on choisit entre + et − de façon à rendre le dénominateur le plus grand possible en module (noter qu'on obtiendra des nombres complexes non réels dès que $C_i^2 \leq 4f(x_i) f[x_i, x_{i-1}, x_{i-2}]$, d'où la nécessité de travailler en arithmétique complexe pour obtenir les zéros complexes).
On obtient l'algorithme suivant :

Algorithme 2.10 (Müller) : on choisit $x_0,\ x_1,\ x_2$

$h_1 := x_1 - x_0,\ h_2 := x_2 - x_1$
$f[x_1, x_0] := \frac{f(x_1) - f(x_0)}{h_1}\ ;\ f[x_2, x_1] := \frac{f(x_2) - f(x_1)}{h_2}$
  Pour $i = 2, 3, ...$
  $f[x_i, x_{i-1}, x_{i-2}] := (f[x_i, x_{i-1}] - f[x_{i-1}, x_{i-2}]) / (h_{i+1} + h_i)$
  $C_i := f[x_i, x_{i-1}] + h_i f[x_i, x_{i-1}, x_{i-2}]$
  $h_{i+1} := -2f(x_i) / \left(C_i \pm \left\{C_i^2 - 4f(x_i) f[x_i, x_{i-1}, x_{i-2}]\right\}^{\frac{1}{2}}\right)$
  et on choisit le signe pour que le dénominateur de $|h_{i+1}|$ soit le plus grand possible
  $x_{i+1} := x_i + h_{i+1}$
  on calcule $f(x_{i+1})$
  $f[x_{i+1}, x_i] := (f(x_{i+1}) - f(x_i)) / h_{i+1}$.
Arrêt lorsque $|x_{i+1} - x_i| < \varepsilon |x_{i+1}|$
  ou
$|f(x_{i+1})| < \varepsilon$
  ou
lorsque le nombre maximum d'itérations préfixé est atteint.

Remarque 2.18 Il faut noter que chaque itération de cet algorithme ne nécessite qu'une seule évaluation de f (qu'on peut faire à l'aide de l'algorithme de Hörner si f est un polynôme). On obtient pourtant une méthode d'ordre $1,84... = q$ où q est la plus grande racine de $q^3 - q^2 - q - 1 = 0$. Ceci s'obtient comme dans la méthode de la sécante en montrant que si $e_i = |x_i - x_\infty|$, alors $e_{i+1} \leq C e_i e_{i-1} e_{i-2}$. Cette méthode se révèle très efficace dans la pratique et relativement précise.

2.4 Système d'équations non linéaires

2.4.1 Introduction

Reprenons les méthodes étudiées au paragraphe 2.2. Il est clair que la méthode de dichotomie n'a aucun sens en dimension $N \geq 2$. En revanche, les trois autres méthodes peuvent s'adapter au cas d'une fonction $F : \mathbb{R}^N \longrightarrow \mathbb{R}^N$. Passons rapidement sur les méthodes de point fixe qui nécessitent que le système d'équations à résoudre soit écrit sous la forme

$$\left\{\begin{array}{rcl} x_1 & = & g_1(x_1, x_2, \ldots, x_N) \\ x_2 & = & g_2(x_1, x_2, \ldots, x_N) \\ \vdots & & \vdots \\ x_N & = & g_N(x_1, x_2, \ldots, x_N)\ . \end{array}\right. \tag{2.22}$$

Il est assez rare que la méthode de point fixe écrite à partir du système (2.22) converge, les conditions de convergence étant plus difficiles à réaliser qu'en dimension un. Il faut en effet que pour une certaine norme matricielle (voir chapitre 6), la norme de la matrice Jacobienne de F soit strictement inférieure à 1 au point recherché. On peut bien sûr envisager des méthodes d'accélération de convergence comme au paragraphe 2.2.5. Nous faisons le choix de présenter plutot ici les méthodes généralisant la méthode de Newton et celle de la sécante.

2.4.2 La méthode de Newton-Raphson

Considérons un système d'équations s'écrivant $F(X) = 0$ ou encore, de façon développée

$$\left\{ \begin{array}{ll} f_1(x_1, x_2, \ldots, x_N) & = 0 \\ f_2(x_1, x_2, \ldots, x_N) & = 0 \\ \vdots & \vdots \\ f_N(x_1, x_2, \ldots, x_N) & = 0 \ . \end{array} \right. \tag{2.23}$$

La généralisation naturelle de la formule de Newton unidimensionnelle

$$x_{n+1} = x_n - (f'(x_n))^{-1} f(x_n)$$

fait intervenir la matrice Jacobienne de F :

$$F'(X_n) = \left(\begin{array}{cccc} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_N}{\partial x_1} & \frac{\partial f_N}{\partial x_2} & \cdots & \frac{\partial f_N}{\partial x_N} \end{array} \right) \tag{2.24}$$

toutes les dérivées partielles étant évaluées au point X_n. La méthode de Newton-Raphson s'écrit donc formellement

$$X_{n+1} = X_n - [F'(X_n)]^{-1} F(X_n) \ . \tag{2.25}$$

Le second membre de (2.25) est parfaitement défini car on effectue le produit d'une matrice $N \times N$ par un vecteur de $\mathbb{R}^N$. Dans la pratique, on ne calcule pas explicitement l'inverse de la matrice Jacobienne, ce qui s'avèrerait trop coûteux, et on préfère écrire l'algorithme sous la forme suivante :

Algorithme 2.11 (Newton-Raphson)

$$\left[\begin{array}{l} X_0 \text{ donné , pour } n = 0, 1, \ldots, \text{ test d'arrêt , faire} \\ \text{Résolution du système linéaire } F'(X_n)\delta_n = -F(X_n) \\ X_{n+1} = X_n + \delta_n \end{array} \right.$$

Remarque 2.19 Encore plus qu'en dimension 1, le choix de l'initialisation est crucial et le risque de divergence, si on ne démarre pas à proximité de la solution cherchée, est grand.

Remarque 2.20 La convergence est là aussi d'ordre 2, donc très rapide (quand il y a convergence !)

Remarque 2.21 Dans le cas des systèmes de grande dimension, la méthode de Newton-Raphson s'avère assez coûteuse puisqu'il faut évaluer à chaque itération $N^2 + N$ fonctions (les N^2 dérivées partielles de la matrice Jacobienne, plus les N fonctions coordonnées). De plus, il y a un système linéaire $N \times N$ (dont la matrice est en général pleine)

à résoudre. Pour pallier le premier inconvénient, il est fréquent qu'on ne recalcule pas la matrice Jacobienne à chaque itération, mais seulement de temps en temps. Bien sûr, à ce moment-là, la convergence n'est plus quadratique, mais en temps de calcul il arrive qu'on y gagne beaucoup, en particulier quand les dérivées de F sont très coûteuses à calculer.

2.4.3 La méthode de Broyden

La méthode de Broyden s'inspire directement de la remarque 2.21 ci-dessus en poussant même la logique un peu plus loin. Puisqu'il est coûteux (voire même dans certains cas impossible) de calculer la matrice Jacobienne de F, on va se contenter d'en déterminer une valeur approchée à chaque itération B_n. De plus la suite de matrices B_n se calculera simplement par récurrence.
On peut considérer que la méthode de la sécante, en dimension un, est basée un peu sur le même principe. En effet elle revient à approcher

$$f'(x_n) \quad \text{par} \quad \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} \ .$$

En dimension N, on va simplement imposer à la suite de matrices B_n de vérifier la même relation :

$$B_n(X_n - X_{n-1}) = F(X_n) - F(X_{n-1}) \ . \tag{2.26}$$

Bien sûr, la relation (2.26) ne définit pas la matrice B_n. Elle n'impose que sa valeur dans une direction. Broyden a fait le choix simple de considérer qu'on pouvait passer de B_n à B_{n+1} en rajoutant simplement une matrice de rang 1. Ainsi, sa formule d'actualisation s'écrit,

$$B_{n+1} = B_n + \frac{(\delta F_n - B_n \delta X_n)(\delta X_n)^T}{\|\delta X_n\|^2} \tag{2.27}$$

où on a noté $\delta X_n = X_{n+1} - X_n$ et $\delta F_n = F(X_{n+1}) - F(X_n)$. On vérifie immédiatement que la suite de matrice B_n définie par (2.27) satisfait bien (2.26). La méthode de Broyden s'écrit donc :

Algorithme 2.12 (Broyden)

$$\left[\begin{array}{l} X_0 \text{ et } B_0 \text{ donnés, pour } n = 0, 1, \ldots, \text{ test d'arrêt, faire} \\ \text{Résolution du système linéaire } B_n \delta_n = -F(X_n) \\ X_{n+1} = X_n + \delta_n \ , \quad \delta F_n = F(X_{n+1}) - F(X_n) \\ B_{n+1} = B_n + \dfrac{(\delta F_n - B_n \delta_n)(\delta_n)^T}{\|\delta_n\|^2} \end{array} \right.$$

Remarque 2.22 On peut prendre comme matrice initiale $B_0 = Id$, il faut évidemment attendre un certain temps avant d'avoir une approximation raisonnable de la matrice Jacobienne.

Remarque 2.23 On peut démontrer qu'en général, comme pour la méthode de la sécante, la convergence est superlinéaire. La suite de matrices B_n ne converge pas forcément vers la matrice Jacobienne de F.

2.5 Exercices du chapitre 2

Exercice 2.1 Etudier la convergence des méthodes de point fixe suivantes (on cherche à approcher $\sqrt{2}$). Appliquer des méthodes d'accélération de la convergence pour rendre

ces algorithmes convergents quand ils ne convergent pas ou plus rapides quand ils convergent.

$$x_{n+1} = \frac{2}{x_n}$$
$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{2}{x_n}\right)$$
$$x_{n+1} = 2x_n - \frac{2}{x_n}$$
$$x_{n+1} = \frac{1}{3}\left(2x_n + \frac{2}{x_n}\right)$$

Exercice 2.2 Pour les fonctions suivantes, trouver un algorithme de point fixe (autre que la méthode de Newton!) qui converge vers le plus petit zéro positif de

1. $x^3 - x - 1 = 0$
2. $x - \tan x = 0$
3. $e^{-x} - \cos x = 0$.

Exercice 2.3 Trouver les racines des polynômes suivants

$$x^4 - 56,101x^3 + 785,6561x^2 - 72,7856x + 0,078$$

$$x^7 - 28x^6 + 322x^5 - 1960x^4 + 6769x^3 - Ax^2 + 13068x - 5040$$

avec $A = 13132$, puis $A = 13133$.

Exercice 2.4 En utilisant la formule de Taylor, imaginer un algorithme modifiant la méthode de Newton et assurant une convergence quadratique dans le cas d'une racine multiple de f.

Exercice 2.5 En appliquant le Théorème de Rouché (voirs cours d'analyse complexe) et la règle de Descartes, localiser au mieux les zéros des polynômes

$$x^6 - 6x^4 + 2x^3 + x - 1 \qquad x^5 - 2x^4 + 8x^3 - 3x^2 + 1\,.$$

Chapitre 3

Interpolation et approximation polynômiale

Dans ce chapitre, on dispose d'une fonction f, connue par exemple uniquement par ses valeurs en certains points, et on cherche à remplacer ou à approcher f par une fonction plus simple, le plus souvent par un polynôme. Nous verrons dans ce contexte, *l'interpolation* qui consiste à rechercher un polynôme qui passe exactement par les points donnés, *l'interpolation par morceaux*, en particulier *les fonctions splines* où l'expression du polynôme est différente sur chaque sous-intervalle, *l'approximation uniforme* ou *l'approximation au sens des moindres carrés* ou on cherche à approcher au mieux la fonction, soit pour la norme uniforme (ou norme infinie), soit pour la norme euclidienne. Cette dernière approche conduira naturellement aux fonctions trigonométriques et aux séries de Fourier, et on présentera la *Transformée de Fourier Rapide* ou F.F.T.

3.1 Interpolation de Lagrange

3.1.1 Le polynôme d'interpolation

Soit $f : [a, b] \to \mathbb{R}$ connue en n+1 points distincts $x_0, x_1, ...x_n$ de l'intervalle $[a, b]$. Il s'agit de construire un polynôme P de degré inférieur ou égal à n tel que

$$\forall i = 0, 1, ...n \qquad P(x_i) = f(x_i) \tag{3.1}$$

Théorème 3.1 *Il existe un et un seul polynôme de degré inférieur ou égal à n solution de (3.1). Le polynôme s'écrit*

$$P_n(x) = \sum_{i=0}^{n} f(x_i)\, L_i(x) \tag{3.2}$$

où

$$L_i(x) = \prod_{k=0 \quad k \neq i} \frac{(x - x_k)}{(x_i - x_k)}. \tag{3.3}$$

Remarque 3.1 Le polynôme P_n est appelé polynôme d'interpolation de Lagrange de la fonction f aux points $x_0, x_1, ...x_n$.
Les polynômes $L_i(x)$ sont appelés polynômes de base de Lagrange associés à ces points.

Démonstration du Théorème 3.1 :
Existence : On vérifie directement que le polynôme donné par (3.2) est solution de (3.1) (on utilise le fait que $L_i(x_j) = \delta_{ij}$).
Unicité : Soit Q un autre polynôme solution. Alors $\forall i = 0, 1, ...n \quad Q(x_i) - P(x_i) = 0$. Ainsi $Q - P$ est un polynôme de degré inférieur ou égal à n s'annulant en $n+1$ points. Il est donc identiquement nul.

Exemple 3.1 : Interpolation linéaire. On applique (3.2) avec $n = 1$ pour trouver

$$P_1(x) = f(x_0)\frac{(x - x_1)}{(x_0 - x_1)} + f(x_1)\frac{(x - x_0)}{(x_1 - x_0)}.$$

Ceci s'écrit encore

$$P_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0). \tag{3.4}$$

Remarque 3.2 L'écriture (3.2) du polynôme d'interpolation est intéressante au point de vue théorique, mais peu du point de vue numérique : elle a un caractère peu algorithmique. De plus, son évaluation requiert trop d'opérations élémentaires.

On lui préfère la formule de Newton qui consiste à écrire le polynôme d'interpolation P_n aux points $x_0, x_1, ...x_n$ sous une forme généralisant (3.4) soit :

$$P_n(x) = a_0^n + a_1^n(x - x_0) + ... + a_n^n(x - x_0)(x - x_1)...(x - x_{n-1}).$$

Notons d'abord que tout polynôme de degré inférieur ou égal à n peut se mettre sous cette forme dès que les x_i sont tous distincts. L'avantage de cette écriture est que la partie tronquée

$$a_0^n + a_1^n(x - x_0) + a_2^n(x - x_0)(x - x_1) + ... + a_n^{n-1}(x - x_0)(x - x_1)...(x - x_{n-2})$$

n'est pas autre chose que $P_{n-1}(x)$: en effet, il s'agit d'un polynôme de degré inférieur ou égal à $n - 1$ tel que :

$$\forall i = 0, 1, ...n - 1 \qquad \text{Valeur en } \ x_i = P_n(x_i) = f(x_i).$$

Donc, connaissant P_{n-1}, il suffit de calculer a_n^n pour connaître P_n (a_n^n est aussi le coefficient de x^n dans P_n écrit sous forme usuelle). Les a_n^i sont donnés par la formule de Newton :

Théorème 3.2 (Formule d'interpolation de Newton) *Le polynôme d'interpolation de Lagrange de la fonction f aux points distincts $x_0, x_1, ...x_n$ est donné par :*

$$P_n(x) = \sum_{i=0}^{n} f[x_0, x_1, ...x_i] \prod_{k=0}^{i-1}(x - x_k) \tag{3.5}$$

où $f[.]$ désigne les différences divisées de f définies par :

$$i = 0, ..., n \qquad f[x_i] = f(x_i) \tag{3.6}$$

$$f[x_0, x_1, ..., x_k] := \frac{f[x_1, ..., x_k] - f[x_0, ..., x_{k-1}]}{x_k - x_0}. \tag{3.7}$$

Remarque 3.3 Par convention $\prod_{k=0}^{i-1}(x - x_k) := 1$ si $i < 1$.

Démonstration : On la fait par récurrence sur le nombre de points x_i.
– La formule (3.5) est vraie pour $n = 1$ (cf. (3.4)).

– Supposons la vraie pour n points. On a alors :

$$P_n(x) = P_{n-1}(x) + a_n^n \prod_{k=0}^{n-1}(x - x_k) = \sum_{i=0}^{n-1} f[x_0, x_1, ..., x_i] \prod_{k=0}^{n-2}(x - x_k) + a_n^n \prod_{k=0}^{n-1}(x - x_k).$$

Reste à calculer a_n^n. Considérons

$$Q(x) = \frac{x - x_0}{x_n - x_0} Q_{n-1}(x) + \frac{x_n - x}{x_n - x_0} P_{n-1}(x)$$

où $Q_{n-1}(x) = \sum_{i=1}^{n} f[x_1, ..., x_i] \prod_{k=1}^{n-1}(x - x_k)$ est le polynôme d'interpolation de f aux n points $x_1, x_2, ..., x_n$. On vérifie immédiatement que :

$$\forall i = 0, 1, ..., n \qquad Q(x_i) = f(x_i).$$

Puisque $\deg Q \leq n$, $Q \equiv P_n$. Cette nouvelle expression de P_n nous permet d'affirmer que le coefficient de x^n de P_n est donné par

$$a_n^n = \frac{1}{x_n - x_0} f[x_1, ..., x_n] - \frac{1}{x_n - x_0} f[x_0, x_1, ..., x_n].$$

3.1.2 Propriétés des différences divisées

i) $f[x_0, x_1, ..., x_n]$ est invariant par des permutations sur les x_i : en effet, permuter les x_i ne change pas le polynôme d'interpolation et donc ne change pas le coefficient du terme de plus haut degré.
ii) si $f = Q$ est un polynôme de degré q

$$Q[x_0, x_1, ..., x_n] = \begin{cases} 0 \text{ si } q < n \\ \text{coefficient du terme de plus haut degré de } Q \text{ si } q = n. \end{cases}$$

En effet, si $q \leq n$, l'interpolé de Q n'est autre que Q lui-même.
iii) table des différences divisées : permet de calculer de façon inductive les différences divisées d'une fonction selon le shéma suivant :

$$\begin{array}{llllll}
x_0 & f[x_0] & & & & \\
 & & f[x_0, x_1] & & & \\
x_1 & f[x_1] & & f[x_0, x_1, x_2] & & \\
 & & f[x_1, x_2] & & f[x_0, x_1, x_2, x_3] & \\
x_2 & f[x_2] & & f[x_1, x_2, x_3] & & \ddots \\
 & & f[x_2, x_3] & & & \\
x_3 & f[x_3] & & & & \\
\vdots & \vdots & & \vdots & & \\
\vdots & \vdots & f[x_{n-1}, x_n] & & & \\
x_n & f[x_n] & & & &
\end{array}$$

Les coefficients de la formule de Newton sont donnés par les premiers éléments de chaque colonne (diagonale (1)).
iv) évaluation de $P_n(x)$: Nous avons vu au paragraphe précédent que l'algorithme de Hörner permet l'évaluation d'un polynôme en un point de façon plus rapide et plus stable que la forme développée usuelle. La même remarque vaut ici pour la formule de Newton.

Pour mettre en oeuvre l'algorithme de Hörner, il est plus agréable d'utiliser la formule de Newton aux points $x_n, x_{n-1}, ..., x_1, x_0$ soit

$$P_n(x) = \sum_{i=0}^{n} f[x_i, ..., x_n] \prod_{k=i+1}^{n} (x - x_k).$$

Algorithme 3.1 : Evaluation de $P_n(x)$: on suppose les valeurs $f(x_i)$ mises en mémoire dans d_i.

on obtient ici $d_i = f[x_i, ..., x_n]$ $\longleftarrow$

algorithme de Hörner $\longleftarrow$

$$\left[\begin{array}{l} \left[\begin{array}{c} \text{de } k = 1 \text{ à } n \\ \left[\begin{array}{c} \text{de } i = 0 \text{ à } n-k \\ d_i := (d_{i+1} - d_i)/(x_{i+k} - x_i) \end{array}\right] \\ p := d_0 \\ \left[\begin{array}{c} \text{de } i = 1 \text{ à } n \\ p := d_i + (x - x_i) \times p \end{array}\right] \end{array}\right] \\ P_n(x) := p. \end{array}\right]$$

Remarque 3.4 On vérifie que seule la diagonale (2) est conservée en mémoire. La seule connaissance de cette diagonale est suffisante pour calculer :

$$P_n(x) = [...(x - x_{n-2})[d_2 + (x - x_{n-1})[d_1 + d_0(x - x_n)]]...].$$

Remarque 3.5 Si on veut calculer $P_n(x)$ pour des valeurs successives de n, on remarque que la connaissance de la diagonale de type (2) engendrée par $f(x_k)$ et la connaissance de $f(x_{k+1})$ suffisent à engendrer la diagonale correspondant à $f(x_{k+1})$. On peut alors utiliser l'algorithme suivant :

Algorithme 3.2 : Evaluation de $P_1(x), P_2(x), ...$: on suppose les valeurs $f(x_i)$ mises en mémoire dans d_i $(i = 0, 1, ...N)$.

$$\left[\begin{array}{c} \text{De } n = 1 \text{ à } n \\ \left[\begin{array}{c} \text{de } i = n-1 \text{ à } 0 \\ d_i = (d_{i+1} - d_i)/(x_n - x_i) \end{array}\right] \\ \left[\begin{array}{c} p := d_0 \\ \text{de } i = 1 \text{ à } n \\ p := d_i + (x - x_i) \times p \end{array}\right] \\ P_n(x) := p. \end{array}\right]$$

Cette dernière partie de l'algorithme peut être simplifiée en se rappelant que

$$P_n(x) = P_{n-1}(x) + f[x_0, x_1, ..., x_n](x - x_0)...(x - x_{n-1}).$$

On obtient alors

Algorithme 3.3 :

$$\left[\begin{array}{c} \Psi(x) := 1 \quad P_0(x) = d_0 \\ \text{De } n = 1 \text{ à } N \\ \left[\begin{array}{c} \text{de } i = n-1 \text{ à } 0 \\ d_i := (d_{i+1} - d_i)/(x_n - x_i) \end{array}\right] \\ \left[\begin{array}{c} \Psi(x) := \Psi(x) \times (x - x_{n-1}) \\ P_n(x) := P_{n-1}(x) + d_0 \times \Psi(x). \end{array}\right] \end{array}\right]$$

v) Interpolation entre des points équidistants : Les différences divisées peuvent alors être remplacées par des différences finies. On note :

$$x_i = a + ih, \quad i = 0, ..., N, \quad N = \frac{a-b}{h}$$
$$s := s(x) = \frac{x-x_0}{h} \Longleftrightarrow x = x_0 + sh$$
$$f(x) = f(x_0 + sh) = f_s.$$

Les différences finies sont obtenues par la formule de récurrence

$$\Delta^i f_s = \begin{cases} f_s \text{ si } i = 0 \\ \Delta^{i-1} f_{s+1} - \Delta^{i-1} f_s \text{ si } i > 0 \end{cases}$$

On note aussi $\Delta^1 = \Delta$. Le lien avec les différences finies est :

$$f[x_k, ..., x_{k+i}] = \frac{1}{i! h^i} \Delta^i f_k. \tag{3.8}$$

Ceci se démontre facilement par récurrence sur le nombre de points. La formule d'interpolation de Lagrange devient alors, (au lieu de (3.5)) ce qu'on appelle la formule de Newton progressive :

$$P_n(x) = P_n(x_0 + sh) = \sum_{i=0}^{n} \binom{s}{i} \Delta^i f_0 \tag{3.9}$$

où on pose

$$\binom{s}{i} = \begin{cases} \dfrac{s(s-1)...(s-i+1)}{1.2...i} \text{ si } i > 0 \\ 1 \text{ si } i = 0 \end{cases} \tag{3.10}$$

Remarque 3.6 La notation (3.10) est bien sûr une généralisation de la notation utilisée pour les coefficients binomiaux correspondants à s entier positif (=nombre de façons de choisir i éléments parmi s).

La formule (3.9) se démontre à partir de (3.5) et de (3.8) : on a en effet, en utilisant aussi $x - x_k = (s-k)h$

$$P_n(x) = \sum_{i=0}^{n} \frac{1}{i! h^i} \Delta^i f_0 . h^i \prod_{k=0}^{i-1} (s-k).$$

Remarque 3.7 avant l'ère des calculateurs rapides, ces formules étaient utilisées pour calculer des valeurs approchées d'une fonction f en tout point à partir des valeurs données dans une table (généralement en des points équidistants).

Remarque 3.8 le calcul des différences finies successives permettait également de détecter les erreurs dans les tables. En effet, même de faibles erreurs sur les valeurs de f peuvent entraîner des oscillations importantes des différences finies d'ordre supérieur.

Il existe aussi une formule de Newton régressive utilisée pour faire des calculs en fin de table. Elle s'obtient à l'aide de (3.7), soit

$$P_n(x_n - sh) = \sum_{i=0}^{n} \binom{s+i-1}{i} \overline{\Delta}^i f_n$$

où

$$\overline{\Delta}^i f_s = \begin{cases} f_s \text{ si } i = 0 \\ \Delta^{i-1} f_s - \Delta^{i-1} f_{s-1} \text{ si } i > 0. \end{cases}$$

3.1.3 Erreur dans l'interpolation de Lagrange

Le but de l'interpolation étant de remplacer l'évaluation de $f(x)$ par celle de $P_n(x)$, il est important de connaître l'erreur

$$E_n(x) = f(x) - P_n(x), \quad x \in [a,b]$$

Exemple 3.2 : soit $f(x) = \dfrac{1}{1+x^2}$: pour une interpolation sur l'intervalle $[-5,5]$ avec des points équidistants,

$$x_i = i\frac{10}{n} - 5, \quad i = 0,1,...,n$$

on obtient les résultats suivants : n est le nombre de points et

$$E_n = \max_{-5\leq x\leq 5} [f(x) - P_n(x)] \approx \max_i |f(y_i) - P_n(y_i)|$$

où on prend $y_i = \frac{i}{10} - 5, i = 0,...,100$.

n	2	4	6	8	10	12	14	16
E_n	0.6	0.4	0.6	1.05	1.91	3.6	7.2	14.0

Dans cet exemple, l'erreur augmente très vite avec n !...Ceci est bien sûr paradoxal dans la mesure où on utilise de plus en plus d'informations sur f ! On comprendra mieux ce qui se passe après avoir vu le Théorème d'erreur suivant.

Théorème 3.3 *Soit $f : [a,b] \longrightarrow \mathbb{R}$, $n+1$ fois continument différentiable et P_n le polynôme d'interpolation de Lagrange aux points $x_0, x_1, ..., x_n$ de $[a,b]$. Alors*

$$|f(x) - P_n(x)| \leq \frac{M_{n+1}}{(n+1)!} |\pi_n(x)| \tag{3.11}$$

où

$$M_{n+1} = \max_{a\leq x\leq b} \left| f^{(n+1)}(x) \right| \tag{3.12}$$

et

$$\pi_n(x) = \prod_{i=0}^{n} (x - x_i). \tag{3.13}$$

La démonstration va reposer sur le Lemme suivant.

Lemme 3.1 *Sous les hypothèses du théorème, il existe $\zeta \in [a,b]$ tel que*

$$f[x_0, ..., x_n] = \frac{f^{(n)}(\zeta)}{n!}.$$

Démonstration : La fonction $E_n(x) = f(x) - P_n(x)$ s'annule en $n+1$ points distincts. D'après le théorème de Rolle, $E_n'(x)$ s'annule en au moins n points distincts de $[a,b]$. A nouveau, d'après le même théorème, $E_n''(x)$ s'annule $n-1$ fois, etc... On obtient ainsi que $E_n^{(n)}$ s'annule en un point $\zeta \in [a,b]$, soit $f^{(n)}(\zeta) = P_n^{(n)}(\zeta)$. Comme P_n est de degré n, $P_n^{(n)}(\zeta) = a_n n!$ où a_n est le coefficient de x_n, soit ici $a_n = f[x_0, x_1, ..., x_n]$. Ce qui démontre le lemme.

Démonstration du théorème : Soit $\overline{x} \in [a,b]$; notons Q le polynôme d'interpolation aux points $x_0, x_1, ..., x_n, \overline{x}$. D'après la formule de Newton, il est donné au point $\overline{x}$ par

$$Q(\overline{x}) = P_n(\overline{x}) + f[x_0, x_1, ..., x_n, \overline{x}]\, \pi_n(\overline{x}).$$

Mais puique $Q(\overline{x}) = f(\overline{x})$ ceci donne

$$f(\overline{x}) - P_n(\overline{x}) = f[x_0, x_1, ..., x_n, \overline{x}]\, \pi_n(\overline{x}). \tag{3.14}$$

Il suffit alors d'appliquer le lemme pour conclure à (3.11).

3.1.3.1 Analyse du théorème 3.3

Exemple 3.2 : Soit $f(x) = e^x$; pour $x \in [a,b]$, on a $M_{n+1} = e^b$. D'autre part, on a la majoration facile $|\pi_n(x)| \leq (b-a)^{n+1}$ (pour tout choix des $n+1$ points x_i). Donc, d'après l'estimation (3.11) :

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

En particulier,

$$\lim_{n\to\infty} |f(x) - P_n(x)| = 0.$$

Donc, dans le cas de la fonction exponentielle, plus on prend de points, meilleure est l'interpolation.

Remarque 3.9 On peut en fait démontrer la même chose pour toute fonction développable en série entière au point $\frac{a+b}{2}$ avec un rayon de convergence $r > \frac{3}{2}(b-a)$.

Lorsqu'on prend des points x_i équidistants dans $[a,b]$ - ce qui est fréquemment utilisé vu la simplicité des calculs - on peut montrer que

$$\max_{a\leq x\leq b} |\pi_n(x)| \leq C \frac{e^{-n}}{\sqrt{n}\log n}(b-a)^{n+1} \tag{3.15}$$

où C est une constante positive. Analysons l'exemple 3.1, à l'aide de ce résultat.
Soit $f(x) = \frac{1}{1+x^2}, a = -b > 0$. Pour dériver facilement f, on remarque que $f(x) = \Im m\left(\frac{1}{x-i}\right)$, d'où $f^{(k)}(x) = \Im m\left[\frac{(-1)^k}{(x-i)^{k+1}}\right] k!$; on en déduit, puisque

$$\frac{1}{(x-i)^{k-1}} = \left(\frac{x+i}{x^2+1}\right)^{k+1} = \frac{1}{(x^2+1)^{\frac{k+1}{2}}}(\cos\theta + i\sin\theta)^{k+1}$$

où θ est tel que $\cos\theta = \frac{x}{\sqrt{x^2+1}}, \sin\theta = \frac{1}{\sqrt{x^2+1}}$

$$f^{(k)}(x) = \frac{(-1)^k k!}{(x^2+1)^{\frac{k+1}{2}}} \sin(\theta(k+1)).$$

On constate que, au moins pour k grand, $M_k \sim k!$. Les formules (3.11) et (3.12) donnent donc :

$$\max_{-a\leq x\leq a} |E_n(x)| < C \frac{e^{-n}}{\sqrt{n}\log n}(2a)^{n+1}.$$

Conclusion : Si $2a < e$, $E_n(x)$ converge vers 0 avec n.
Dans l'exemple 3.1, $a = 5$. Dans ce cas, le majorant ci-dessus tend vers l'infini. Ceci ne permet plus de rien affirmer sur $E_n(x)$. Le calcul numérique de l'exemple 3.1 suggère que E_n augmente au lieu de diminuer. On peut en fait montrer que $\max_{|x|\leq 5} |E_n(x)|$ ne tend pas vers 0. Il faut donc être très prudent avec l'interpolation polynômiale.
L'estimation (3.11) du Théorème 3.3 montre que l'erreur dépend d'une part des dérivées de f, d'autre part du maximum de la fonction π_n, qui lui ne dépend que du choix des x_i. On peut chercher à minimiser $\max_{a\leq x\leq b} |\pi_n(x)|$ en choisissant au mieux les points x_i.
Il se trouve qu'un choix de points équidistants n'est pas du tout optimal, loin de là. On verra au paragraphe suivant que le choix optimal est obtenu à l'aide des points de Chebyshev, soit $x_i = \frac{a+b}{2} + \frac{b-a}{2}\cos\frac{(2i+1)\pi}{2n+2}, i = 0, 1, ..., n$.
Les graphes de $|\pi_n(x)|$ pour un choix de points équidistants et pour les points de Chebyshev sont présentés à la figure 3.1

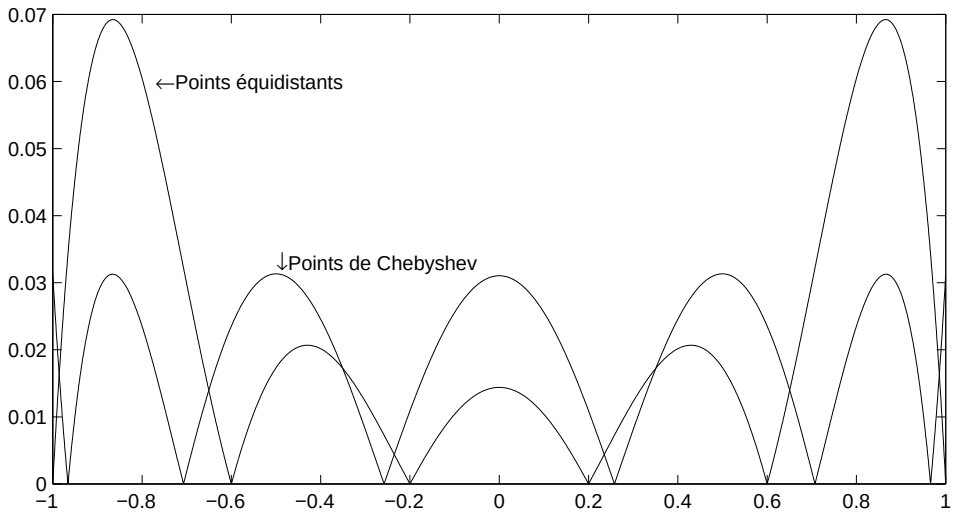

FIGURE 3.1 – Points de Chebyshev et points équidistants

On voit que, pour des points équidistants, $|\pi_n(x)|$ est relativement grand au voisinage des extrémités. Si on se limite à des points équidistants, il vaudra donc mieux se placer à l'intérieur de l'intervalle et surtout éviter les extrémités. On remarque que, pour les points de Chebyshev, le maximum de $|\pi_n(x)|$ est nettement plus petit. D'autre part, tous les maximums relatifs sont égaux : il semble bien que ce choix "régularise" les différents pics en les uniformisant. C'est ce que nous allons étudier maintenant. Nous reviendrons à l'interpolation un peu plus tard en envisageant le cas de l'interpolation par morceaux.

3.2 Approximation polynômiale uniforme

Le problème est le suivant : étant donné une fonction f continue sur $[a,b]$, il s'agit de l'approcher au mieux par un polynôme de degré inférieur ou égal à n, et par "au mieux", nous entendons ici :
Minimiser

$$\max_{a\leq x\leq b} |f(x)-P(x)| = \|f-P\|_\infty \tag{3.16}$$

parmi les polynômes P de degré $\leq n$.

Remarque 3.10 nous verrons plus loin un autre point de vue conduisant à la "méthode des moindres carrés".

Donnons le résultat fondamental dû à Chebishev.

Théorème 3.4 *Il existe un et un seul polynôme P_n tel que*

$$\|f-P_n\|_\infty = \min\{\|f-P\|_\infty ; \deg P \leq n\}.$$

Ce polynôme est caractérisé par la propriété suivante : il existe $n+2$ points distincts $x_0, x_1, ..., x_{n+1}$ dans $[a,b]$ tels que

$$(-1)^i(f(x_i)-P_n(x_i)) = sign(f(x_0)-P_n(x_0))\,\|f-P_n\|_\infty . \tag{3.17}$$

La démonstration de ce théorème, sans faire appel à des résultats compliqués est relativement longue et nous l'admettons ici. Faisons simplement quelques remarques :
i) comme il existe un polynôme de meilleure approximation on peut se demander s'il existe un algorithme permettant de le construire. La réponse est positive. Le plus connu est celui de Rémes. Cependant, dans la pratique, il existe des méthodes plus simples permettant d'obtenir, sinon le meilleur approximant, du moins des approximants "presque optimaux". Cette méthode repose sur l'interpolation aux points de Chebyshev, ou aux points de Chebyshev généralisés (voir plus loin).
ii) la propriété (3.17) caractérise le polynôme de meilleure approximation. C'est un moyen pratique de vérifier qu'un polynôme est bien le meilleur en ce sens.

Exemple 3.4 : Considérons $f(x) = e^x$ sur $[-1, 1]$. Alors son meilleur approximant linéaire est donné par :

$$P(x) = a + bx, \quad b = \frac{e - e^{-1}}{2}, \quad a = \frac{e - bx_1}{2}, \quad x_1 = \log b.$$

En effet, on vérifie que $f(1) - P(1) = -(f(x_1) - P(x_1)) = f(-1) - P(-1) = \frac{e+bx_1}{2} - b$. D'autre part, la dérivée de $f(x) - P(x)$, soit $e^x - b$, ne s'annule qu'au point x_1. Donc $f - P$ atteint son minimum en x_1 et son maximum en -1 et 1. Remarquons que les points x_i de la condition (3.17) contiennent ici les extremités de l'intervalle. Ceci se produit chaque fois que $f^{(n+1)}(x)$ ne change pas de signe sur l'intervalle $[a, b]$.

Exemple 3.5 : Quelle est la meilleure approximation du polynôme x^{n+1} dans l'espace des polynômes de degré inférieur ou égal à n sur $[-1, 1]$?
Pour répondre à cette question, on définit les polynômes de Chebyshev $T_k(x)$ par la relation

$$T_k(\cos\theta) = \cos k\theta \tag{3.18}$$

qui signifie que T_k est le polynôme permettant d'exprimer $\cos k\theta$ en fonction de $\cos\theta$. Ce polynôme existe bien et est défini de façon récurrente par

$$\begin{cases} T_0(x) \equiv 1 \\ T_1(x) \equiv x \\ T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x), k \geq 1. \end{cases} \tag{3.19}$$

Cette relation provient de l'identité trigonométrique

$$\cos(k+1)\theta = 2\cos\theta\cos k\theta - \cos(k-1)\theta.$$

On a ainsi $T_2(x) = 2x^2 - 1, T_3(x) = 4x^3 - 3x$, etc...
A l'aide de (3.19), on obtient facilement par récurrence que :
– T_k est un polynôme de degré k pour tout k
– le coefficient de x^k est 2^{k-1}.
D'autre part, à l'aide de (3.18), on a
– $|T_k(x)| \leq 1$, $\forall x \in [-1, 1]$
– $\begin{cases} T_k(x) = \pm 1 \\ -1 \leq x \leq 1 \end{cases} \iff \begin{cases} x = \cos\theta \\ k\theta = j\pi, \ j \text{ entier} \end{cases} \iff \begin{cases} x_j = \cos\frac{j}{k}\pi \\ j = 0, 1, ..., k. \end{cases}$
Donc $T_k(x)$ prend alternativement les valeurs ± 1 en les $k + 1$ points x_j. Mais ceci montre que, si on pose

$$P_n(x) = x^{n+1} - 2^{-n}T_{n+1}(x)$$

on obtient un polynôme P_n de degré n tel que

$$x^{n+1} - P_n(x) = 2^{-n}T_{n+1}(x)$$

satisfasse (3.17) aux points $\left\{\cos\frac{j}{n+1}\pi, j=0,1,...,n+1\right\}$. Le polynôme P_n est donc le polynôme de meilleure approximation du polynôme x^{n+1} parmi les polynômes de degré inférieur ou égal à n.

On en déduit encore le

Théorème 3.5 *Pour que $\pi_n(x)=(x-x_0)(x-x_1)...(x-x_n)$ soit tel que $\|\pi_n\|_\infty$ soit minimal parmi les choix des x_i, $i=0,1,...,n$ dans l'intervalle $[-1,1]$, il faut et il suffit que*

$$\pi_n(x)=2^{-n}T_{n+1}(x).$$

En particulier, les x_i doivent être exactement les zéros de T_{n+1}.

Démonstration : π_n s'écrit : $\pi_n(x)=x^{n+1}-Q(x)$ où $\deg Q\leq n$. Minimiser $\|\pi_n\|_\infty$ revient donc à minimiser $\left\|x^{n+1}-Q(x)\right\|_\infty$ parmi les polynômes Q de degré inférieur ou égal à n (et tels que $x^{n+1}-Q(x)$ aient des racines réelles et simples). On sait déjà que $Q=P_n$ donné ci-dessus est l'unique solution et qu'alors

$$\pi_n(x)=x^{n+1}-P_n(x)=2^{-n}T_{n+1}(x).$$

Remarque 3.11 Si la même question est posée pour un intervalle $[a,b]$, on s'y ramène en considérant le polynôme

$$\widetilde{\pi_n}(x)=\pi_n(\frac{a+b}{2}+x\frac{b-a}{2}).$$

Minimiser $\max_{a\leq x\leq b}\pi_n(x)$ revient à minimiser $\max_{-1\leq x\leq 1}\widetilde{\pi_n}(x)$. La solution est

$$\widetilde{\pi_n}(x)=2^{-n}T_{n+1}(x),\ \text{d'où}$$

$$\pi_n(x)=2^{-n}T_{n+1}(\frac{2}{b-a}(y-\frac{a+b}{2})).$$

En particulier, les x_i solutions sont donnés par :

$$x_i=\frac{a+b}{2}+\frac{b-a}{2}\cos\frac{(2i+1)\pi}{2(n+1)}, i=0,1,...,n. \tag{3.20}$$

Nous avons vu au paragraphe précédent que, lorsque le nombre de points augmente, le polynôme d'interpolation de Lagrange ne converge pas toujours vers la fonction initiale f. La question reste donc posée de savoir si toute fonction est approchable par des polynômes uniformément sur un intervalle. La réponse est donnée par le classique théorème de Weierstrass.

Théorème 3.6 (Weierstrass) *Soit f une fonction continue sur $[a,b]$ et soit P_n son polynôme de meilleure approximation de degré inférieur ou égal à n. Alors*

$$\lim_{n\to\infty}\left(\max_{a\leq x\leq b}|f(x)-P_n(x)|\right)=0.$$

En d'autres termes, toute fonction continue sur un intervalle est approchable uniformément par une suite de polynômes. Nous renvoyons à la littérature pour une démonstration de ce théorème.

Mentionnons à présent une autre estimation de l'erreur d'interpolation complétant (3.11) :

Théorème 3.7 *Soit f continue sur $[a,b]$ et P_n son polynôme d'interpolation aux points $x_0, x_1, ..., x_n$. Alors*

$$\|f - P_n\|_\infty \leq (1 + \|\Lambda_n\|_\infty) \left(\inf_{\deg P \leq n} \|f - P\|_\infty \right)$$

$$\text{où } \Lambda_n(x) = \sum_{i=0}^{n} |L_i(x)| .$$

Remarque 3.12 $\inf_{\deg P \leq n} \|f - P\|_\infty$ représente la distance de f à l'espace des polynômes de degré inférieur ou égal à n et ne dépend donc que de f. D'autre part, Λ_n ne dépend que du choix des x_i. Il est donc intéressant de les analyser séparément. Signalons sans démonstration :

- $\inf_{\deg P \leq n} \|f - P\|_\infty$ tend vers 0 quand n tend vers l'infini (c'est le théorème de Weierstrass) et ce d'autant plus vite que f est plus régulière.
- Si on note $\overline{\Lambda_n}$ la borne inférieure des $\|\Lambda_n\|_\infty$ quand on fait varier les points x_i,

$$\lim_{n \to \infty} \overline{\Lambda_n} = +\infty.$$

- On peut caractériser les points x_i qui donnent le meilleur $\|\Lambda_n\|_\infty$ (soit $\|\Lambda_n\|_\infty = \overline{\Lambda_n}$). Seulement, ces points sont difficiles à utiliser dans la pratique. On leur préfère par exemple les points de Chebyshev; une raison suffisante est que pour des points de Chebyshev, le $\Lambda_n(x)$ vérifie :

$$\text{quand } n \longrightarrow \infty, \qquad \|\Lambda_n(x)\|_\infty \sim \overline{\Lambda_n} \sim \frac{2}{\pi} \log n.$$

  En d'autres termes, pour ce choix, le $\|\Lambda_n\|_\infty$, même s'il est plus grand, est du même ordre.
- On peut encore légèrement faire mieux à l'aide des points de Chebyshev étendus donnés par

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \frac{\cos \frac{(2i+1)\pi}{2(n+1)}}{\cos \frac{\pi}{2(n+1)}}.$$

  Dans ce cas, $\|\Lambda_n\|_\infty$ est voisin de $\overline{\Lambda_n}$ à $0,02$ près pour tout n.
- En contrepartie, pour des points équidistants, on a

$$\|\Lambda_n\|_\infty \sim \frac{2^{n+1}}{en \log n} \text{ quand } n \longrightarrow \infty.$$

  Ceci, comparé au $\frac{2}{\pi} \log n$ précédent, confirme que les points équidistants sont un mauvais choix pour ce type de questions.
- Un résultat négatif affirme que, quelque soit le choix des points $x_0^n, x_1^n, ..., x_n^n$ où on fait l'interpolation, on peut trouver une fonction continue f pour laquelle le polynôme d'interpolation ne converge pas vers f, et ce même pour les points de Chebyshev. Cependant, dans ce dernier choix, dès que f est de classe C^1 sur $[a,b]$ (et on peut faire mieux), la convergence a lieu.

Nous renvoyons à la bibliographie pour tous ces résultats.

3.3 Approximation au sens des moindres carrés

3.3.1 Introduction

La méthode que nous allons voir maintenant s'appelle en statistiques *ajustement des données*. Jusque ici, nous avons considéré l'approximation d'une fonction par un procédé d'interpolation à l'aide des valeurs $f(x)$ de f en des points x_k. Ceci présuppose que ces valeurs soient connues, et ce de façon assez précise. Il y a beaucoup de situations où ce n'est pas le cas en particulier lorsque les valeurs $f(x)$ proviennent de mesures physiques.

Exemple 3.6 :

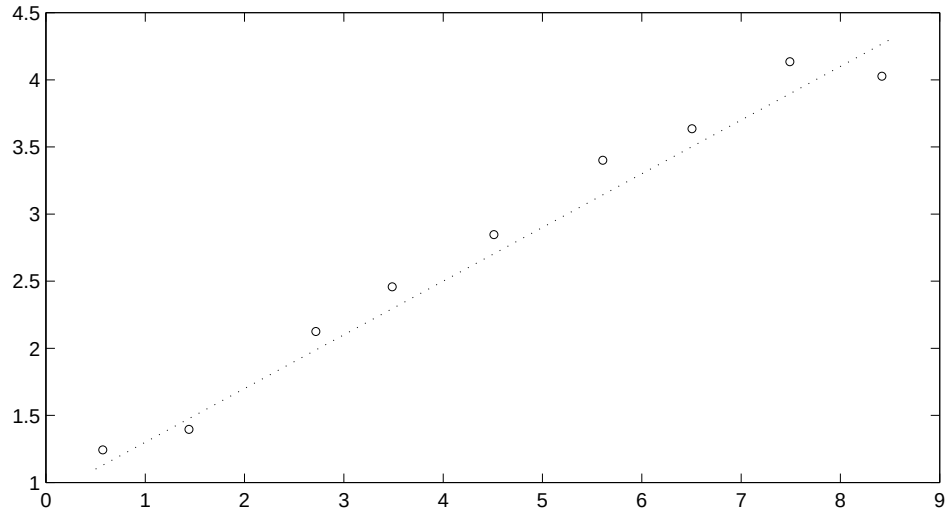

FIGURE 3.2 – Droite des moindres carrés

Le résultat de mesures physiques tel que ci-dessus conduit à penser que $f(x)$ doit être une fonction linéaire. Il ne serait pas très raisonnable de remplacer $f(x)$ par son polynôme d'interpolation en les points x_k dont le calcul dépendrait uniquement des valeurs manifestement erronées $f(x_k)$.
Une analyse succinte du phénomène ci-dessus, fréquent dans la réalité physique, conduit à penser que ces valeurs mesurées $f(x_k)$ contiennent une information juste mais aussi un certain "bruit". Intuitivement, on est conduit à penser que les données $f(x_k)$ contiennent
- une première composante variant lentement : c'est elle qui fournit l'information
- une deuxième composante variant très rapidement : c'est celle qui est due au "bruit" inévitable.

Tout l'art de l'ajustement des données consiste à éliminer la deuxième composante pour ne retenir que la "substantifique moëlle". Le principe de l'étude qui suit consiste à chercher la fonction f sous la forme

$$f(x) = c_1\theta_1(x) + c_2\theta_2(x) + ... + c_n\theta_n(x)$$

où les θ_i sont des fonctions connues (par exemple des polynômes) et les c_i des coefficients à déterminer. Implicitement, n sera assez grand pour contenir l'information, mais suffisamment petit pour se débarrasser des "bruits".
Ainsi dans l'exemple 3.6, on pourra déterminer c_1 et c_2 pour que les déviations

$$\{f(x_k) - (c_1 + c_2x_k)\,;k = 0, ..., p\}$$

soient les plus petites possibles (contrairement à l'interpolation, il n'est pas possible d'annuler ces déviations). Maintenant, il existe plusieurs notions de "le plus petit possible". Celle qu'on a utilisée dans le paragraphe précédent consiste à minimiser :

$$\max_{0\leq k\leq p} |f(x_k) - (c_1 + c_2 x_k)| .$$

On peut tout aussi bien penser à minimiser

$$\sum_{0\leq k\leq p} |f(x_k) - (c_1 + c_2 x_k)| .$$

Ceci conduira en général à des (c_1, c_2) différents.

3.3.2 Méthode classique des moindres carrés

C'est une troisième voie qui est le plus souvent choisie dans la pratique, car elle conduit à un problème de minimisation quadratique en les inconnues c_i, à savoir minimiser

$$\sum_{0\leq k\leq p} |f(x_k) - (c_1 + c_2 x_k)|^2 \tag{3.21}$$

ou plus généralement minimiser

$$\sum_{0\leq k\leq p} (f(x_k) - (c_1\theta_1(x_k) + ... + c_n\theta_n(x_k)))^2 . \tag{3.22}$$

Posons $E(c_1, c_2, ..., c_n) = \sum_{0\leq k\leq p} (f(x_k) - (c_1\theta_1(x_k) + ... + c_n\theta_n(x_k)))^2$. Si E admet un minimum en $c = (c_1, c_2, ..., c_n)$, on a

$$\frac{\partial E}{\partial c_i}(c) = 0, i = 1, ..., n$$

soit

$$\left\{ \begin{array}{l} 2\sum_{0\leq k\leq p} \theta_i(x_k) (f(x_k) - (c_1\theta_1(x_k) + ... + c_n\theta_n(x_k))) = 0 \\ i = 1, ..., n \end{array} \right\} \tag{3.23}$$

ce qui constitue un système linéaire de n équations aux n inconnues $c_1, c_2, ..., c_n$, appelées parfois équations normales. Donnons une autre interprétation, plus géométrique, de ce système (cf Figure 3.3). Notons

$$e = (e_0, e_1, ..., e_p), \quad e_k = f(x_k) - (c_1\theta_1(x_k) + ... + c_n\theta_n(x_k))$$

le vecteur erreur et $\theta_i = (\theta_i(x_0), \theta_i(x_1), ..., \theta_i(x_p))$. Alors, si on note $\langle u, v\rangle = \sum_{i=0}^{p} u_i v_i$ le produit scalaire dans $\mathbb{R}^{p+1}$, les équations (3.23) expriment

$$\forall i = 1, ..., n, \quad \langle e, \theta_i\rangle = 0 \tag{3.24}$$

en d'autre termes, le vecteur erreur e doit être orthogonal à tous les vecteurs θ_i, ce qui, géométriquement, revient à dire que le vecteur $c_1\theta_1 + c_2\theta_2 + ... + c_n\theta_n$ solution est la projection orthogonale du vecteur $P = (f(x_0), ..., f(x_p))$ sur le sous espace engendré par les θ_i.

Exemple 3.7 : Ajustement linéaire : on prend $n = 2$ et $\theta_1(x) \equiv 1, \theta_2(x) \equiv x$. On cherche donc la fonction affine $c_1 + c_2 x$ la plus voisine au sens des moindres carrés des valeurs $(x_k, f(x_k))$.

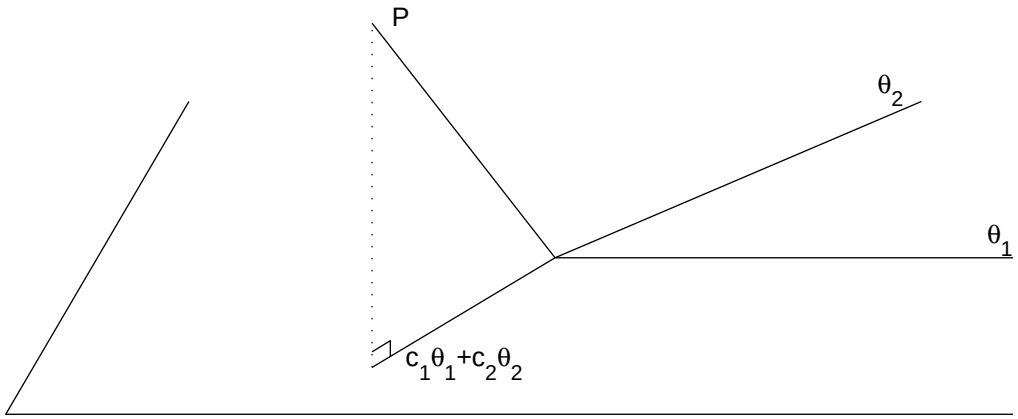

FIGURE 3.3 – Ajustement et projection orthogonale

Le système (3.23) devient

$$\begin{cases} \sum_{0\leq k\leq p} (f(x_k) - (c_1 + c_2 x_k)) = 0 \\ \sum_{0\leq k\leq p} x_k (f(x_k) - (c_1 + c_2 x_k)) = 0 \end{cases}$$

ou

$$\begin{cases} (p+1)c_1 + (\sum_{k=0}^{p} x_k)\, c_2 = \sum_{k=0}^{p} f(x_k) \\ (\sum_{k=0}^{p} x_k)\, c_1 + (\sum_{k=0}^{p} x_k^2)\, c_2 = \sum_{k=0}^{p} x_k f(x_k) \end{cases}$$

Notons :

$$m = \frac{\sum_{k=0}^{p} x_k}{p+1} \text{ (moyenne des } x_k), \quad m_2 := \frac{\sum_{k=0}^{p} x_k^2}{p+1} \text{ (moment d'ordre 2)}$$

$$\overline{f} := \frac{\sum_{k=0}^{p} f(x_k)}{p+1}, \quad \overline{\overline{f}} = \frac{\sum_{k=0}^{p} x_k f(x_k)}{p+1}.$$

Le système devient

$$\begin{cases} c_1 + m c_2 = \overline{f} \\ m c_1 + m_2 c_2 = \overline{\overline{f}}. \end{cases}$$

Son unique solution est donnée par :

$$c_1 = \frac{m_2 \overline{f} - m\overline{\overline{f}}}{m_2 - m^2}, \quad c_2 = \frac{\overline{\overline{f}} - m\overline{f}}{m_2 - m^2}.$$

Remarque 3.13 On vérifie que $m_2 \neq m^2$ si les x_k sont distincts. Dans le cas général, on montre aussi que le système (3.23) a toujours une solution. Il s'écrit

$$Ac = b \tag{3.25}$$

où $c = (c_1, ..., c_n)$, A est la matrice dont les coefficients sont $\{\langle \theta_j, \theta_i \rangle\}_{1\leq i\leq n\,, 1\leq j\leq n}$ (notons que c'est toujours une matrice symétrique) et $b = (b_1, ..., b_n)$ avec $b_i = \langle f, \theta_i \rangle$.

Nous verrons dans le chapitre 6 les techniques de résolution de systèmes linéaires. Nous pouvons cependant déjà souligner que les méthodes sont d'autant plus performantes que la matrice est creuse (c'est-à-dire avec beaucoup de zéros) et à diagonale dominante (i.e. éléments hors diagonale petits devant ceux de la diagonale).

Un cas particulier très intéressant que nous allons développer ici est celui où on choisit les facteurs $\theta_i(x)$ de telle façon que les produits scalaires $\langle \theta_j, \theta_i \rangle$ soient nuls pour $i \neq j$, c'est-à-dire

$$\sum_{k=0}^{p} \theta_j(x_k)\theta_i(x_k) = 0, \quad \forall i \neq j. \tag{3.26}$$

Dans la pratique, on s'arrange pour être si possible dans cette situation (ou proche de cette situation). Sans précaution, on peut avoir des problèmes de conditionnement très sérieux dans la résolution de (3.25). C'est le cas dans l'exemple suivant :

Exemple 3.8 : $p = 3, \theta_1(x) \equiv 1, \theta_2(x) \equiv x, \theta_3(x) \equiv x^2$. On approche une fonction f sur l'intervalle $[10, 15]$ connaissant ses valeurs en 6 points dont les extrémités. On montre que le conditionnement associé (cf. chapitre 6) peut être supérieur à 10^5.

Remarque 3.14 dans la formule (3.26), si les points x_k sont équidistants dans l'intervalle $[a, b]$, on a :

$$0 = \sum_{k=0}^{p} \theta_j \left(a + k\frac{b-a}{p} \right) \theta_i \left(a + k\frac{b-a}{p} \right) \sim \frac{p}{b-a} \int_a^b \theta_j(x)\theta_i(x)dx,$$

si p est assez grand. Ainsi, une version continue de notre problème consiste à considérer les polynômes (ou des fonctions plus générales) θ_i telles que

$$\int_a^b \theta_j(x)\theta_i(x)dx = 0.$$

Cette intégrale représente également un produit scalaire sur l'espace des fonctions et par analogie on dit que les deux fonctions sont orthogonales (le bon espace fonctionnel associé est alors $L^2([a, b])$).

Exemples 3.9 : (a) les fonctions 1 et x sont orthogonales pour le produit scalaire

$$\langle f, g \rangle = \int_{-1}^{1} f(x)g(x)dx$$

$$\text{car } \int_{-1}^{1} 1.xdx = \left[\frac{x^2}{2} \right]_{-1}^{1} = 0.$$

(b) les fonctions $\cos kx$ et $\cos px$ sont orthogonales pour $|k| \neq |p|$ en effet :

$$\int_0^{2\pi} \cos kx \cos pxdx = \frac{1}{2} \int_0^{2\pi} \left[\cos(k+p)x + \cos(k-p)x \right] dx = 0.$$

(c) les fonctions $\cos kx$ et $\sin px$ sont orthogonales en effet :

$$\int_0^{2\pi} \cos kx \sin pxdx = \frac{1}{2} \int_0^{2\pi} \left[\sin(k+p)x + \sin(p-k)x \right] dx = 0.$$

3.3.3 Polynômes orthogonaux

Comme nous serons amenés à les utiliser dans le chapitre sur l'intégration numérique, nous allons maintenant étudier les suites de polynômes orthogonaux pour des produits scalaires du type :

$$\langle P, Q\rangle = \int_a^b P(x)Q(x)w(x)dx \tag{3.27}$$

ou

$$\langle P, Q\rangle = \sum_{n=1}^{N} P(x_n)Q(x_n)w_n \tag{3.28}$$

où w est une fonction continue strictement positive sur $]a, b[$ dans (3.27), (w_n) une suite de nombres strictement positifs dans (3.28) et (x_n) des points distincts de $[a, b]$.
On note que (3.28) est un produit scalaire sur l'espace des polynômes de degré inférieur ou égal à $N-1$. D'autre part, si on suppose que $x^n w(x)$ est intégrable pour tout n, alors (3.27) définit aussi un produit scalaire sur l'espace des polynômes.

Définition 3.1 *Par suite de polynômes orthogonaux, on entend une suite (finie ou infinie)*
$P_0(x), P_1(x), \ldots$ *telle que*

$$P_i \text{ est de degré } i \tag{3.29}$$

$$\langle P_i, P_j\rangle = 0, \quad \text{si } i \neq j. \tag{3.30}$$

Proposition 3.1 *Soit (P_n) une suite de polynômes orthogonaux; alors*
- *Tout polynôme Q de degré inférieur ou égal à k s'écrit de façon unique*

$$Q(x) = d_0P_0(x) + d_1P_1(x) + ... + d_kP_k(x) \qquad (\text{avec } d_i = \frac{\langle P_i, Q\rangle}{\langle P_i, P_i\rangle}). \tag{3.31}$$

- *Si Q est de degré $< k$, alors*

$$\langle Q, P_k\rangle = 0. \tag{3.32}$$

- *Si A_i désigne le coefficient du terme de plus haut degré de P_i, on a la relation de récurrence à trois termes*

$$\widehat{P}_{i+1}(x) = (x - B_i)\widehat{P}_i(x) - C_i\widehat{P}_{i-1}(x) \tag{3.33}$$

$$\text{où } \widehat{P}_i(x) := \frac{P_i(x)}{A_i} \quad \text{(polynôme normalisé),}$$

$$B_i = \frac{\left\langle x\widehat{P}_i(x), \widehat{P}_i(x)\right\rangle}{\left\langle \widehat{P}_i(x), \widehat{P}_i(x)\right\rangle}, \; C_i = \frac{\left\langle \widehat{P}_i(x), \widehat{P}_i(x)\right\rangle}{\left\langle \widehat{P}_i(x), \widehat{P}_{i-1}(x)\right\rangle}.$$

- *P_i a exactement i zéros réels distincts (dans le cas du produit scalaire (3.28), on suppose $i \leq N$).*

Démonstration : Le point (3.31) est classique en algèbre linéaire : les polynômes $P_0, ..., P_k$ forment une base de l'espace des polynômes de degré inférieur ou égal à k, car ils sont tous de degrés différents.
Pour (3.32), on remarque que, si $\deg Q < k$, d'après (3.31),

$$Q(x) = d_0P_0(x) + d_1P_1(x) + ... + d_{k-1}P_{k-1}(x)$$

et d'après la bilinéarité du produit scalaire

$$\langle Q, P_k \rangle = \sum_{i=0}^{k-1} d_i \langle P_i, P_k \rangle = 0.$$

Vérifions (3.33) : on peut toujours écrire

$$\widehat{P}_{i+1}(x) = x\widehat{P}_i(x) + \alpha_i \widehat{P}_i(x) + \alpha_{i-1}\widehat{P}_{i-1}(x) + ... + \alpha_0 \widehat{P}_0(x) \tag{3.34}$$

puisque $\left\{x\widehat{P}_i(x), \widehat{P}_i(x), \widehat{P}_{i-1}(x), ..., \widehat{P}_0(x)\right\}$, étant une famille de polynômes de degrés distincts, est une base de l'espace des polynômes de degré inférieur ou égal à $i+1$. D'autre part, puisque les polynômes sont normalisés, le coefficient de $x\widehat{P}_i(x)$ dans l'écriture $\widehat{P}_{i+1}$ ci-dessus est égal à 1.
Multiplions scalairement (3.34) par $\widehat{P}_i$; on obtient grâce à (3.30) :

$$0 = \left\langle x\widehat{P}_i(x), \widehat{P}_i(x) \right\rangle + \alpha_i \left\langle \widehat{P}_i(x), \widehat{P}_i(x) \right\rangle + 0$$

d'où

$$\alpha_i = -\frac{\left\langle x\widehat{P}_i(x), \widehat{P}_i(x) \right\rangle}{\left\langle \widehat{P}_i, \widehat{P}_i \right\rangle}.$$

Multiplions scalairement (3.34) par $\widehat{P}_j$ où $j \leq i-1$. Toujours grâce à (3.30), on a :

$$0 = \left\langle x\widehat{P}_i, \widehat{P}_j \right\rangle + d_j \left\langle \widehat{P}_j, \widehat{P}_j \right\rangle.$$

Mais $\left\langle x\widehat{P}_i, \widehat{P}_j \right\rangle = \left\langle \widehat{P}_i, x\widehat{P}_j \right\rangle = 0$ si $j+1 \leq i$. Puisque $\left\langle \widehat{P}_j, \widehat{P}_j \right\rangle \neq 0$, ceci implique $\alpha_j = 0$ pour tout $j = 0, 1, ..., i-2$. Pour $j = i-1$, on remarque que

$$\left\langle x\widehat{P}_i, \widehat{P}_{i-1} \right\rangle = \left\langle \widehat{P}_i, x\widehat{P}_{i-1} \right\rangle = \left\langle \widehat{P}_i, \widehat{P}_i \right\rangle + \left\langle \widehat{P}_i, x\widehat{P}_{i-1} - \widehat{P}_i \right\rangle.$$

Mais ce dernier produit scalaire est nul puique $x\widehat{P}_{i-1} - \widehat{P}_i$ est un polynôme de degré inférieur ou égal à $i-1$. On en déduit

$$\alpha_{i-1} = -\frac{\left\langle \widehat{P}_i, \widehat{P}_i \right\rangle}{\left\langle \widehat{P}_{i-1}, \widehat{P}_{i-1} \right\rangle},$$

ce qui termine la démonstration de (3.33).
Pour le dernier point, notons $\zeta_1, ..., \zeta_k$ les racines réelles distinctes de multiplicité impaire et posons

$$R_i(x) = \begin{cases} 1 & \text{si } k = 0 \text{ (pas de racine réelle de multiplicité impaire)} \\ (x-\zeta_1)(x-\zeta_2)...(x-\zeta_k) & \text{si } k > 0. \end{cases}$$

Si $k = i$, P_i a alors i racines réelles distinctes ce qu'on voulait prouver.
Sinon $k < i$, et d'après (3.30)

$$\langle P_i, R_i \rangle = 0.$$

Pour le produit scalaire (3.27), ceci signifie

$$\int_a^b P_i(x) R_i(x) w(x) dx = 0.$$

Mais $P_i = R_i Q_i$ où Q_i est de signe constant d'après le choix de R_i ; ainsi

$$\left\{ \begin{array}{c} \int_a^b R_i(x)^2 Q_i(x) w(x) dx = 0 \\ R_i(x)^2 Q_i(x) w(x) \text{ de signe constant} \end{array} \right\} \Longrightarrow R_i^2 Q_i w \equiv 0;$$

ceci implique en particulier $Q_i = 0$, ce qui est contradictoire avec $\deg Q_i = i - k > 0$. Dans le cas du produit scalaire (3.28), on a

$$\sum_{n=1}^{N} R_i(x_n)^2 Q_i(x_n) w_n = 0 \Longrightarrow Q_i(x_n) = 0 \quad \forall n = 1, ..., N$$

$\Longrightarrow Q_i \equiv 0$ si $N > \deg Q_i$, ce qui se produit dès que $N \geq i$. On a alors une contradiction.

3.3.4 Exemples classiques de polynômes orthogonaux

(1) $[a, b] = [-1, 1]$ et $w(x) = (1 - x)^\alpha (1 + x)^\beta$ avec $\alpha > -1$ et $\beta > -1$: on obtient les polynômes de Jacobi. On leur attribue des noms différents dans les cas particuliers suivants :

- $\alpha = \beta = 0$: polynômes de Legendre. Une fois les polynômes normalisés, on a la relation de récurrence :

$$P_{n+1}(x) = \frac{(2n+1) x P_n(x) - n P_{n-1}(x)}{n+1}.$$

- $\alpha = \beta = -\frac{1}{2}$: on retrouve les polynômes de Chebyshev définis par la relation de récurrence

$$T_{n+1}(x) = 2x T_n(x) - T_{n-1}(x).$$

- $\alpha = \beta = \frac{1}{2}$: polynômes de Chebyshev de seconde espèce (les précédents étant alors de première espèce).

(2) $[a, b[= [0, +\infty[$ et $w(x) = e^{-x}$: on a les polynômes de Laguerre. Plus généralement, si $w(x) = x^\alpha e^{-x}$ avec $\alpha > -1$, on a les polynômes de Laguerre généralisés. La relation de récurrence s'écrit :

$$P_{n+1}(x) = -\frac{1}{n+1}(x - 2n - \alpha - 1) P_n(x) + (n + \alpha) P_{n-1}(x).$$

(3) $]a, b[=]-\infty, +\infty[$, $w(x) = e^{-x^2}$: polynômes d'Hermite.

$$H_{n+1}(x) = 2x H_n(x) - 2n H_{n-1}(x).$$

Remarque 3.15 on vérifie directement que les polynômes de Chebyshev sont orthogonaux pour le produit scalaire

$$\int_{-1}^{1} \frac{f(x) g(x) dx}{(1 - x^2)^{\frac{1}{2}}}$$

car, en posant $x = \cos\theta$, on a :

$$\int_{-1}^{1} \frac{T_n(x) T_p(x) dx}{\sqrt{1 - x^2}} = \int_0^\pi T_n(\cos\theta) T_p(\cos\theta) d\theta = \int_0^\pi \cos n\theta \cos p\theta d\theta = 0 \text{ si } n \neq p.$$

En ce qui concerne l'approximation au sens des moindres carrés d'une fonction f par des polynômes, on utilise généralement des polynômes orthogonaux plutôt que la base canonique usuelle $1, x, x^2 ...$ On a alors le résultat suivant :

Théorème 3.8 *Soit $f : [a,b] \to \mathbb{R}$ telle que $\langle f,f\rangle < \infty$ ou $\langle .\rangle$ est défini par (3.27) (resp. (3.28)). Alors, pour tout $k \geq 1$ (resp. $k \leq N-1$), si $P_0, P_1, ..., P_k$ est une suite de polynômes orthogonaux pour le produit scalaire $\langle .\rangle$, il existe un et un seul polynôme Q de la forme*

$$Q(x) = c_0P_0(x) + c_1P_1(x) + ... + c_kP_k(x)$$

minimisant $\langle f-Q, f-Q\rangle$. Les coefficients sont donnés par

$$c_i = \frac{\langle P_i, f\rangle}{\langle P_i, P_i\rangle}, \quad i = 0, ..., k.$$

Remarque 3.16 $\langle f-Q, f-Q\rangle = \|f-Q\|^2$ représente le carré de la distance de f à Q pour la norme hilbertienne associée au produit scalaire $\langle .\rangle$.

Exemple 3.10 : Trouver le polynôme P de degré inférieur ou égal à 3 qui minimise

$$\int_{-1}^{1} (e^x - P(x))^2\, dx.$$

On utilise comme base de polynômes les polynômes de Legendre, soit

$$L_0(x) = 1,\ L_1(x) = x,\ L_2(x) = \frac{3}{2}\left(x^2 - \frac{1}{3}\right),\ L_3(x) = \frac{5}{2}\left(x^3 - \frac{3}{5}x\right).$$

On calcule alors

$$\begin{aligned}
\langle f, L_0\rangle &= \int_{-1}^{1} e^x dx = e - \frac{1}{e} \\
\langle f, L_1\rangle &= \int_{-1}^{1} xe^x dx = \frac{2}{e} \\
\langle f, L_2\rangle &= \frac{3}{2}\int_{-1}^{1} e^x \left(x^2 - \frac{1}{3}\right) dx = e - \frac{7}{e} \\
\langle f, L_3\rangle &= \frac{5}{2}\int_{-1}^{1} e^x \left(x^3 - \frac{3}{5}\right) dx = -5e + \frac{37}{e}.
\end{aligned}$$

D'autre part, on montre que $\langle L_i, L_i\rangle = \frac{2}{2i+1}$. On en déduit le polynôme solution :

$P(x) = 1,175201194L_0(x) + 1,10363824L_1(x) + 0,3578143506L_2(x) + 0,07045563367L_3(x)$.

Sous forme canonique usuelle, on obtient :

$$P(x) = 0,9962940183 + 0,9979548730x + 0,5367215260x^2 + 0,1761390842x^3.$$

Remarque 3.17 un avantage de l'écriture de P sur la base L_0, L_1, L_2, L_3 est que par exemple la partie tronquée $1,175...L_0 + 1,131...L_1$ est la meilleure approximation linéaire au sens des moindres carrés de e^x sur $[-1,1]$ (un peu comme dans le cas de la formule d'interpolation de Newton).

L'évaluation en un point d'un polynôme donné sous formes de combinaisons linéaires de polynômes orthogonaux se fait sans revenir à l'écriture sur la base canonique, mais en utilisant directement une factorisation de Hörner obtenue à partir de la relation de récurrence (3.33).
Dans le cas de produits scalaires de type

$$\langle P, Q\rangle = \sum_{n=1}^{N} P(x_n)Q(x_n)w_n$$

très importants dans la pratique, on doit générer les polynômes à l'aide de la relation de récurrence (3.33) : $P_0, ..., P_i$ étant connus aux points x_n, on calcule

$$\begin{aligned} S_i &= \langle P_i, P_i \rangle = \sum_{n=1}^{N} \left(P_i\left(x_n\right)\right)^2 w_n \\ B_i &= \sum_{n=1}^{N} \frac{x_n \left(P_i\left(x_n\right)\right)^2 w_n}{S_i} \\ C_i &= \frac{S_i}{S_{i-1}}. \end{aligned}$$

On en déduit $P_{i+1}(x_n)$ en tous les x_n.

3.3.5 Polynômes trigonométriques

Théorème 3.9 *Soit $f : [-\pi, \pi] \to \mathbb{R}$ continue. Alors il existe une unique meilleure approximation au sens des moindres carrés dans l'espace des polynômes trigonométriques de la forme*

$$S(x) = a_0 + \sum_{k=1}^{n} a_k \cos kx + \sum_{k=1}^{n} b_k \sin kx. \tag{3.35}$$

Ses coefficients sont donnés par :

$$\left\{ \begin{array}{l} a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x)dx \\ a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos kx dx \\ b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin kx dx \end{array} \right. \tag{3.36}$$

Remarque 3.18 On reconnait bien sûr là les classiques coefficients de Fourier !

Démonstration : Il s'agit de minimiser $\int_{-\pi}^{\pi} \left(f(x) - S(x)\right)^2 dx$. Comme les fonctions $\{1, \cos x, \cos 2x, \ldots, \cos nx, \sin x, sin2x, \ldots, \sin nx\}$ forment une suite orthogonale par rapport au produit scalaire

$$\langle f, g \rangle = \int_{-\pi}^{\pi} f(x)g(x)dx,$$

d'après l'analyse faite en début de chapitre, on a immédiatement une solution unique donnée par :

$$\begin{aligned} a_0 &= \frac{\langle f, 1 \rangle}{\langle 1, 1 \rangle} = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x)dx \\ a_k &= \frac{\langle f, \cos kx \rangle}{\langle \cos kx, \cos kx \rangle} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos kx dx \\ b_k &= \frac{\langle f, \sin kx \rangle}{\langle \sin kx, \sin kx \rangle} = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin kx dx. \end{aligned}$$

Remarque 3.19 Ce type d'approximation est surtout utilisé pour les fonctions périodiques comme celles représentant les divers phénomènes ondulatoires. En général, une fonction est composée d'un très grand nombre de fréquences : en se limitant à un nombre fini n, on "filtre" l'information en ne gardant que les plus basses fréquences ("information") et en se débarrassant du "bruit" constitué surtout par les hautes fréquences.

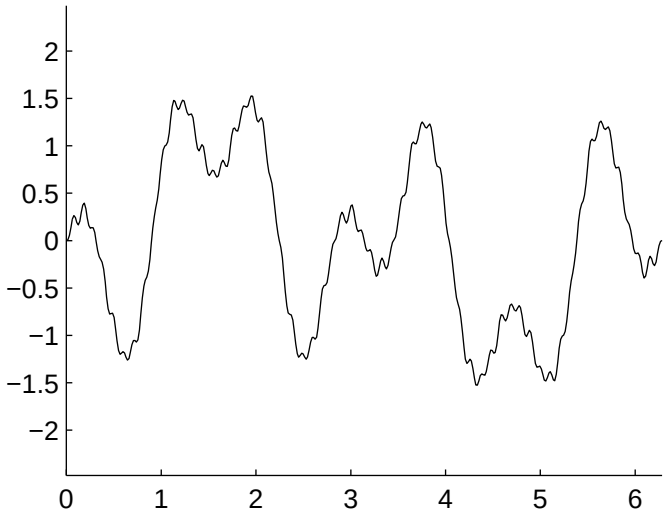

FIGURE 3.4 – Signal bruité

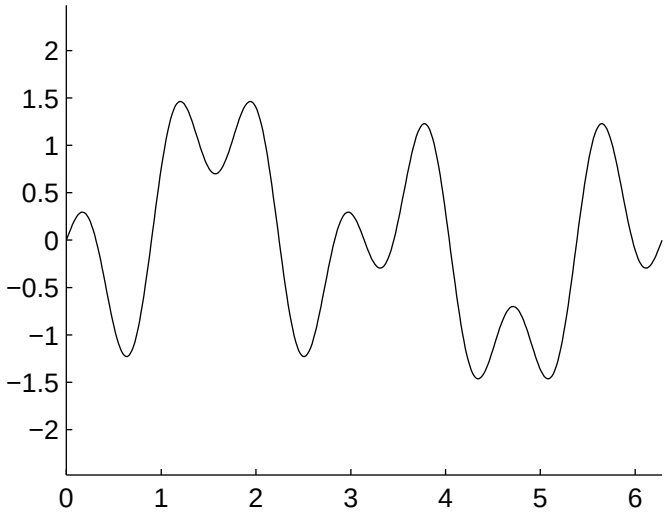

FIGURE 3.5 – Signal débruité

ainsi

devient

Remarque 3.20 Si une fonction f est de période T, on se ramène à l'intervalle $[-\pi, \pi]$ en remarquant que $\widetilde{f}(x) = f\left(\frac{T}{2\pi}x\right)$ est périodique de période 2π. Elle est donc connue pour tout x si elle est connue pour $x \in [-\pi, \pi]$.

On utilise souvent la représentation complexe des coefficients de Fourier :

$$f_n(x) = \sum_{j=-n}^{n} \widehat{f}(j) e^{ijx} \tag{3.37}$$

où

$$\widehat{f}(j) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{-ijx} dx \qquad j = -n, ..., n$$

est une autre expression du polynôme trigonométrique qui approche le mieux f dans $L^2([0,\pi])$. On a d'ailleurs

$$f(x) = \lim_{n\to\infty} f_n(x) = \sum_{j=-\infty}^{+\infty} \widehat{f}(j) e^{ijx} \tag{3.38}$$

la limite ci-dessus étant comprise au sens de la norme

$$\|f - f_n\|_2 = \sqrt{\int_{-\pi}^{\pi} \left(f(x) - f_n(x)\right)^2 dx}$$

pour toute fonction f dans $L^2([0,\pi])$.

De nombreux calculs nécessitent la connaissance des coefficients de Fourier de f. Notons d'abord, que si f est 2π-périodique, ils s'écrivent encore

$$\widehat{f}(j) = \frac{1}{2\pi} \int_{0}^{2\pi} f(x) e^{-ijx} dx.$$

Alors une formule de quadrature très élémentaire conduit à prendre comme approximation

$$\widehat{f}_N(j) = \frac{1}{2\pi} \frac{2\pi}{N} \sum_{n=0}^{N-1} f\left(\frac{2\pi n}{N}\right) e^{-ij(2\pi n/N)}.$$

On montre que cette approximation est convenable si

$$\frac{N}{2} \geq |j|\,.$$

Cette condition se comprend par exemple si $f(x) \equiv 1$; il s'agit alors d'intégrer la fonction $x \to e^{-ijx}$ qui est $\frac{2\pi}{j}$-périodique. Il est clair que les points d'évaluation de la fonction doivent avoir une fréquence plus rapide que celle de la fonction elle-même pour être significatifs.

Nous allons maintenant nous concentrer sur le calcul numérique d'expressions du type

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k}, \qquad x_k = \frac{2\pi k}{N}, \qquad |j| \leq \frac{N}{2}. \tag{3.39}$$

Fait sous la forme ci-dessus, le calcul requiert l'évaluation des produits $f(x_k)e^{-ijx_k}$ puis N sommes et une division pour obtenir c_j. Ceci doit être effectué pour chaque c_j donc environ N fois. Nous négligerons les divisions et nous compterons les opérations élémentaires consistant à calculer un produit $f(x_k)e^{-ijx_k}$ et à l'ajouter à une somme partielle précédente. Ce nombre d'opérations élémentaires est donc ici de l'ordre de N^2. Ainsi un choix de 1000 points nécessite un million d'opérations. Ceci a été un obstacle majeur à l'utilisation de l'analyse de Fourier jusqu'à ce que Cooley et Tukey proposent une méthode qui, après quelques améliorations, a fait passer le nombre d'opérations élémentaires à un ordre $O(N \log N)$ (soit autour de quelques milliers si $N = 1000$) : il s'agit de la Transformée de Fourier rapide (F.F.T.=Fast Fourier Transform) que nous développons maintenant

3.3.6 La transformée de Fourier rapide

Nous en décrirons seulement le principe qui repose sur une décomposition de l'entier N : nous supposerons ici que $N = PQ$, P et Q entiers supérieurs ou égaux à 2. Notons

$$w_N = e^{-i2\pi/N}, \quad w_q = e^{-i2\pi/Q}, \quad Z = [Z_1, ..., Z_N]$$

Avec des changements de notation évidents à partir de (3.39), il s'agit de calculer le vecteur "Transformée de Fourier discrète"

$$\widehat{Z} = \left[\widehat{Z}_1, ..., \widehat{Z}_N\right]$$

avec

$$\widehat{Z}_j := \sum_{k=1}^{n} Z_k w_N^{(j-1)(k-1)} \qquad j = 1, ..., N$$

On interprète le tableau unidimensionnel Z comme un tableau bidimensionnel $Z = Z(P, Q)$ avec :

$$\begin{aligned} 1 &\leq p \leq P \\ 1 &\leq q \leq Q \\ Z(p,q) &= Z_{p+P(q-1)}. \end{aligned}$$

Alors

$$\widehat{Z}_j = \sum_{p,q} Z(p,q) w_N^{(j-1)(p-1+P(q-1))} = \sum_{p=1}^{P} \left[\sum_{q=1}^{Q} Z(p,q) w_Q^{(j-1)(q-1)}\right] w_N^{(j-1)(p-1)}$$

puisque $w_N^P = e^{-i2\pi P/N} = e^{-i2\pi/Q} = w_Q$. Il se trouve et c'est le point important, que la somme entre crochets est une fonction de j périodique de période Q car

$$w_Q^{(j+Q-1)(q-1)} = w_Q^{j(q-1)} w_Q^{Q(q-1)} = w_Q^{j(q-1)}$$

puique $w_Q^Q = 1$. Ainsi seul intervient le reste $\widehat{j}$ de la division de j par Q :
$j = \lambda Q + \widehat{j}$ $0 \leq \widehat{j} \leq Q$, et il suffit de calculer la somme entre crochets pour $j = \widehat{j}$ variant de 1 à Q.
Ce calcul nécessite Q opérations pour chaque j et p et doit donc être effectué $Q \times P$ fois d'où $Q^2 P = NQ$ opérations pour le calcul des crochets. Ensuite le calcul des $\widehat{Z}_j$ nécessite P opérations et doit être effectué N fois soit NP opérations. L'ensemble des $\widehat{Z}_j$, $j = 1, ..., N$ est donc obtenu avec $N(P+Q)$ opérations ce qui est déjà bien moins que N^2 opérations.

	$N = 1\ 000$	$N^2 = 1\ 000\ 000$
Exemple :	$N = 10 \times 100 = P \times Q$	$N(P+Q) = 1\ 000 \times 110 = 110\ 000$
	$N = 25 \times 40 = P \times Q$	$N(P+Q) = 1\ 000 \times 65 = 65\ 000$

On peut encore diminuer le nombre d'opérations en décomposant N sous la forme $N = P_1 \times P_2 \times ... \times P_k$ est en répétant l'idée ci-dessus. Un cas fréquemment utilisé est celui où $N = 2^k$. Nous renvoyons à la littérature pour les détails de la méthode.

3.4 Approximation par des fonctions polynômiales par morceaux

Etant donnée $f : [a, b] \to \mathbb{R}$ connue en $(n+1)$ points $a = x_0 < x_1 < ... < x_n = b$, on l'approche par des fonctions continues dont la restriction à chaque intervalle $[x_{i-1}, x_i]$ est polynômiale.

3.4.1 Approximation linéaire par morceaux

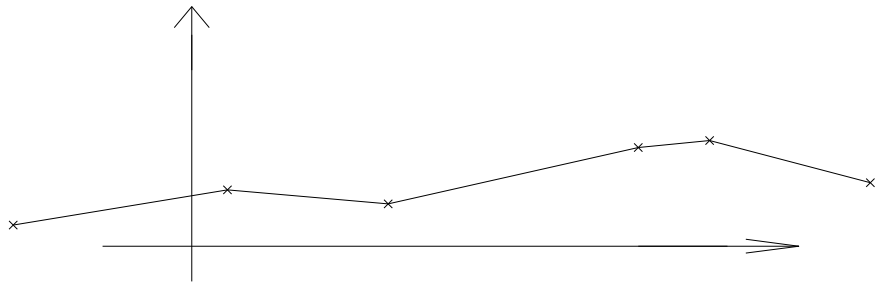

FIGURE 3.6 – Approximation linéaire par morceaux

La restriction de l'approximation P à l'intervalle $[x_{i-1}, x_i]$ est donnée par :

$$P_i(x) = f[x_{i-1}] + f[x_{i-1}, x_i](x - x_{i-1}).$$

La fonction P ainsi construite est évidemment continue sur l'intervalle $[a, b]$. Du temps des tables de fonctions élémentaires (tables de logarithmes ou de fonctions trigonométriques), on obtenait les valeurs hors table par interpolation affine, ce qui revenait exactement à remplacer la fonction par son approximation linéaire par morceaux.

3.4.2 Approximation cubique

Les polynômes de base sont de degré 3. On peut supposer que f est connue en les points $[x_{i-1}, x_{i-2/3}, x_{i-1/3}, x_i]$ où $x_{i-1/3}, x_{i-2/3}$ sont des points distincts de $]x_{i-1}, x_i[$. On prend alors pour P_i l'interpolé de f sur $[x_{i-1}, x_i]$ en ces 4 points.
Une situation plus intéressante consiste à considérer que, outre les valeurs $f(x_i)$, $i = 0, 1, ...N$, on connait aussi les valeurs $f'(x_i)$, $i = 0, 1, ...N$. Dans ce cas, on pourra prendre pour P_i le polynôme de degré 3 tel que :

$$\begin{cases} P_i(x_{i-1}) = f(x_{i-1}) & P_i(x_i) = f(x_i) \\ P_i'(x_{i-1}) = f'(x_{i-1}) & P_i'(x_i) = f'(x_i) \end{cases} \tag{3.40}$$

on sort alors du domaine de l'interpolation de Lagrange pour entrer dans ce qu'on appelle l'interpolation d'Hermite.

Théorème 3.10 *Il existe un et un seul polynôme de degré* 3 *satisfaisant (3.40). Il est donné par la formule de Newton*

$$\begin{cases} P_i(x) = f[x_{i-1}] + f[x_{i-1}, x_{i-1}](x - x_{i-1}) + f[x_{i-1}, x_{i-1}, x_i](x - x_{i-1})^2 \\ \qquad\qquad + f[x_{i-1}, x_{i-1}, x_i, x_i](x - x_{i-1})^2(x - x_i) \end{cases} \tag{3.41}$$

où on définit

$$f[x_{i-1}, x_{i-1}] := \lim_{h \to 0} f[x_{i-1}, x_{i-1} + h] = \lim_{h \to 0} \frac{f[x_{i-1} + h] - f[x_{i-1}]}{h} = f'(x_{i-1})$$

les autres différences divisées étant définies de la façon habituelle.

Remarque 3.21 Ce résultat se généralise bien sûr à un nombre de points quelconque avec des dérivées d'ordre quelconque. La formule de Newton est la même si on étend la définition des différences divisées par continuité comme ci-dessus, soit par récurrence

$$\begin{cases} f[x_i] := f(x_i) \\ f[x_0, x_1, ..., x_k] := \begin{cases} \text{formule usuelle si } x_k \neq x_0 \\ \lim_{h\to 0} f[x_0, ..., x_{k-1}, x_k + h] \text{ si } x_0 = x_k. \end{cases} \end{cases} \tag{3.42}$$

Nous renvoyons à la littérature pour la démonstration du théorème 3.10 et sa version plus générale évoquée dans la remarque ci-dessus. Comme souvent dans la pratique, on ne dispose pas des valeurs de la dérivée, on peut dans la formule (3.40) remplacer $f'(x_i)$ par une valeur approchée s_i de $f'(x_i)$ par exemple

$$s_i = \frac{h_i f[x_i, x_{i+1}] + h_{i+1} f[x_{i-1}, x_i]}{h_i + h_{i+1}} \tag{3.43}$$

avec $h_i = x_i - x_{i-1}$. On parle alors d'interpolation cubique par morceaux de Bessel. Notons que dans ce cas, la fonction d'approximation obtenue est non seulement continue, mais à dérivée continue puisque :

$$P_i'(x_i) = P_{i+1}'(x_i) = s_i.$$

3.4.3 Fonctions splines

Il se trouve que, au lieu de s'imposer les s_i sous la forme (3.43), on peut les choisir pour que la fonction d'approximation soit non seulement à dérivée continue, mais à dérivée seconde continue : il s'agit alors de fonctions splines d'interpolation. Le mot "spline" est utilisé en anglais pour désigner la réglette flexible que les dessinateurs utilisent pour tracer une courbe d'allure "lisse" passant par des points donnés. La méthode que nous allons décrire correspond d'ailleurs à un "lissage" de la courbe passant par les points donnés de coordonnées $(x_i, f(x_i))$. Afin d'analyser cette méthode réécrivons le

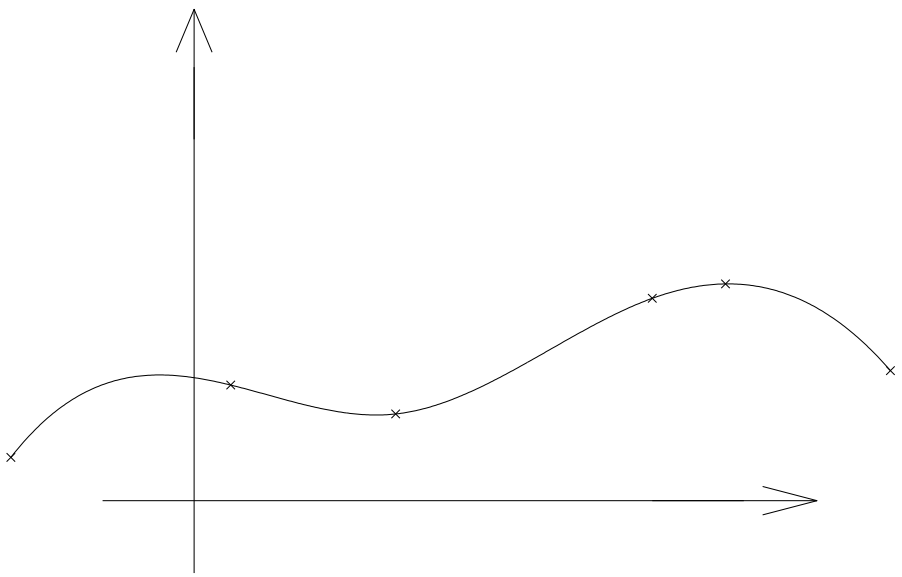

FIGURE 3.7 – Spline cubique

polynôme $P_i(x)$ tel que

$$\begin{cases} P_i(x_{i-1}) = f(x_{i-1}) & P_i(x_i) = f(x_i) \\ P_i'(x_{i-1}) = s_{i-1} & P_i'(x_i) = s_i. \end{cases}$$

Il est donné par la formule (3.41) où on remplace $f'(x_i)$ (resp ; $f'(x_{i-1})$) par s_i (resp. s_{i-1}), soit

$$\begin{aligned} P_i(x) &= f(x_{i-1}) + s_{i-1}(x - x_{i-1}) + h_i^{-1}\left(f[x_{i-1}, x_i] - s_{i-1}\right)(x - x_{i-1})^2 \\ &\quad + C_i (x - x_{i-1})^2 (x - x_i) \end{aligned}$$

avec

$$\begin{aligned} C_i &= \frac{f[x_{i-1}, x_i, x_i] - f[x_{i-1}, x_{i-1}, x_i]}{h_i} \\ &= \left(\frac{s_i - f[x_{i-1}, x_i]}{h_i} - \frac{f[x_{i-1}, x_i] - s_{i-1}}{h_i}\right)/h_i \\ &= \frac{s_i + s_{i-1} - 2f[x_{i-1}, x_i]}{h_i^2}. \end{aligned}$$

On cherche à réaliser

$$P_i''(x_i) = P_{i+1}''(x_i). \tag{3.44}$$

Puisque $\left((x - x_{i-1})^2 (x - x_i)\right)'' = 2(x - x_i) + 4(x - x_{i-1})$, la condition (3.44) s'écrit :

$$2\frac{f[x_{i-1}, x_i] - s_{i-1}}{h_i} + 4\frac{s_i + s_{i-1} - 2f[x_{i-1}, x_i]}{h_i} = 2\frac{f[x_i, x_{i+1}] - s_i}{h_{i+1}} - 2\frac{s_{i+1} + s_i - 2f[x_i, x_{i+1}]}{h_{i+1}}.$$

Ceci conduit au système linéaire en les s_i :

$$\begin{cases} h_{i+1}s_{i-1} + 2(h_i + h_{i+1})s_i + h_i s_{i+1} = 3\left(h_{i+1} f[x_{i+1}, x_i] + h_i f[x_i, x_{i+1}]\right) \\ i = 1, ..., N-1 \end{cases}$$

Ce qui fait $N-1$ équations aux inconnues $s_0, s_1, ..., s_N$. On peut se fixer arbitrairement les valeurs extrêmes s_0 et s_N (par exemple en écrivant que $P''(a) = P''(b) = 0$). On obtient alors un système linéaire à diagonale strictement dominante (|coeff de s_i| > |coeff de s_{i-1}| + |coeff de s_{i-1}|, voir chapitre 6) ; donc il admet une solution unique. Ceci prouve l'existence d'une fonction spline cubique à dérivée seconde continue comme prévu.

Calcul d'erreur : On peut montrer que

$$\max_{a \le x \le b} |f(x) - P(x)| \le \max_{a \le \zeta \le b} \left|f^{(4)}(\zeta)\right| \frac{5\left(\max_i h_i\right)^4}{384}$$

ce qui est une très bonne approximation si la dérivée $f^{(4)}$ est raisonnable et si les intervalles sont petits. D'autre part, au facteur 5 près, cette erreur est du même ordre que celle obtenue dans l'interpolation d'Hermite (selon (3.41)) alors que dans ce cas on utilise deux fois plus d'information sur la fonction, car on a besoin de $f(x_i)$ et $f'(x_i)$ pour tout i.
Ceci suggère que les fonctions splines doivent avoir aux points x_i des dérivées voisines de celles de f. On montre en effet que

$$\max_{a \le x \le b} |f'(x) - P'(x)| \le \max_{a \le \zeta \le b} \left|f^{(4)}(\zeta)\right| \frac{\left(\max_i h_i\right)^3}{24}.$$

Ceci fournit donc une méthode de calcul numérique de la dérivée d'une fonction en des points x_i tout à fait intéressante.

3.5 Exercices du chapitre 3

Exercice 3.1 On veut éditer une table de cosinus pour laquelle l'utilisateur cherchera la valeur d'un cosinus donné par interpolation linéaire entre les valeurs tabulées. Sachant qu'on veut obtenir des valeurs avec une erreur inférieure à 10^{-6}, combien d'entrées dans la table sont-elles nécessaires ?

Exercice 3.2 Déterminer la meilleure approximation linéaire sur $[0,1]$ de $f(x) = x^5 + x^3$, d'abord au sens de la norme uniforme, puis au sens des moindres carrés.

Exercice 3.3 Le tableau suivant donne les valeurs d'une fonction f en trois points.

x	$f(x)$
$x_1 = \sqrt{17} - 4$	0.3771784
$x_2 = \sqrt{10} - 3$	0.4880117
$x_3 = \sqrt{5} - 2$	0.6754903

a) Déterminer une valeur approchée de f en $x_0 = \frac{x_2+x_3}{2}$, par interpolation en x_2 et x_3, puis par interpolation en x_1, x_2 et x_3.
b) Donner dans les deux cas un majorant de l'erreur en fonction des dérivées de f.
c) On constate que les deux fonctions $f_1(x) = \sin \pi x$ et $f_2(x) = \sin \frac{\pi}{x}$ vérifient toutes deux le tableau de valeurs ci-dessus. Calculer $f_1(x_0)$ et $f_2(x_0)$. Expliquer et vérifier la cohérence avec les résultats numériques de a) et b).

Exercice 3.4 On donne les points $(0,2); (1,1); (2,\frac{2}{3}); (3,\frac{1}{2}); (4,\frac{1}{3}); (10,0.1)$.
Chercher des fonctions approchant au mieux ces points au sens des moindres carrés. Comparer les erreurs.

Exercice 3.5 Le tableau suivant donne la viscosité μ (mesurée) de l'éthylène en fonction de la température (en degrés Farenheit) :

T	0	50	100	150	200
μ	242	82.1	30.5	12.6	5.57

Proposer une loi décrivant μ en fonction de T.

Chapitre 4

Dérivation et intégration numérique

4.1 Introduction

Le but de ce chapitre est double : indiquer comment calculer de façon approchée une dérivée et décrire les méthodes de base pour le calcul numérique d'intégrales. Un point commun de ces deux thèmes qui vont nous occuper est l'utilisation de polynômes d'interpolation. L'idée sous-jacente est simple : la fonction avec laquelle on travaille est trop compliquée ou pas assez connue pour qu'on puisse faire des opérations simples comme la dérivation ou l'intégration, on choisit alors de la remplacer par un polynôme et on dérive ou on intègre celui-ci. On présentera assez rapidement les formules de dérivation approchée qui en résultent avec un aperçu sur les méthodes de différences finies. On passera plus de temps sur le calcul d'intégrales du type

$$\int_a^b f(x)\, w(x)\, dx \tag{4.1}$$

où w est une fonction poids (continue strictement positive sur $]a,b[$) et $f(x)\, w(x)$ est intégrable sur $[a,b]$.
Les méthodes d'intégration numérique que nous décrirons consistent toutes à remplacer l'intégrale (4.1) par une expression le plus souvent de la forme :

$$\sum_{i=0}^{N} A_i f(x_i), \tag{4.2}$$

où les x_i sont des points distincts de $[a,b]$ et A_i des coefficients réels, le tout choisi pour que la différence

$$E = \int_a^b f(x)\, w(x)\, dx - \sum_{i=0}^{N} A_i f(x_i)$$

soit petite.
Dans la pratique, il faut évidemment faire la différence entre le cas où la fonction f est connue en un nombre fini de points $y_0,\ y_1, ...,\ y_p$, auquel cas les points x_i devront être des points y_j et celui où la fonction f est connue analytiquement auquel cas on pourra choisir au mieux les points x_i (exemple : calcul de $\int_0^1 e^{-x^2} dx$).
Deux types d'approches sont à retenir :

- Méthodes composites (ou composées) : on divise l'intervalle $[a,b]$ en sous intervalles $[x_{i-1},x_i]$ avec $a = x_0 < x_1 < ... < x_N = b$. Sur chaque intervalle $[x_{i-1},x_i]$ on "remplace" $f(x)$ par un polynôme d'interpolation $P_{i,k}$ de degré $\leq k$ et on remplace $\int_{x_{i-1}}^{x_i} f(x)\,dx$ par $\int_{x_{i-1}}^{x_i} P_{i,k}(x)\,dx$ (ici $w(x) \equiv 1$) : ainsi, sur chaque intervalle $[x_{i-1},x_i]$, on applique une méthode d'intégration élémentaire.
- Méthode de Gauss : elles s'appliquent pour des intervalles $[a,b]$ et des poids w particuliers : on approche f sur $[a,b]$ par un polynôme d'interpolation aux points de $[a,b]$ obtenus à partir des zéros de polynômes orthogonaux associés au poids w sur $[a,b]$. Cette méthode peut être appliquée directement sur l'intervalle $[a,b]$, ou, si $b-a$ est grand, sur des sous-intervalles d'une décomposition.

4.2 Dérivation numérique

Dans ce paragraphe, la fonction f n'est bien sûr pas connue par une formule explicite mais
- ou bien par ses valeurs sur un ensemble discret (en supposant les points assez proches pour que la notion de dérivée ait un sens)
- ou bien, le plus souvent, par un algorithme de calcul ou une formule compliquée qui permet, au moins en théorie, de la calculer en tout point. On suppose bien sûr que la dérivée n'est pas accessible par un procédé analogue. A ce sujet, existent maintenant des techniques de différentiation automatique (par exemple le logiciel Odyssée développé par l'INRIA) qui permettent de différentier chaque pas d'un programme et ainsi de calculer une notion de dérivée, même pour une fonction définie par un programme. Nous ne traiterons pas cette notion ici, nous contentant d'une vision plus ancienne et classique.

Dans toute la suite, on supposera f connue ou calculable aux points $\ldots, x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}, \ldots$ qu'on supposera proches. On notera $h_i = x_{i+1} - x_i$.

4.2.1 Dérivée première

Supposons qu'on veuille calculer une valeur approchée de $f'(x_i)$. Une première idée, déjà évoquée dans le chapitre précédent, consiste à remplacer f par une fonction spline passant par les mêmes points et à prendre la dérivée de la fonction spline. D'un point de vue numérique, cette idée est très bonne, assez stable et donne de bons résultats en terme d'erreur. Malheureusement, elle est assez lourde à mettre en place, nécessite beaucoup de calculs. Elle a de plus un petit côté moralement désagéable : la dérivée est une notion purement locale, en ce sens que la dérivée de f au point x_i ne dépend que des valeurs prises par f *au voisinage* de x_i, alors que la fonction spline dépend globalement de f par l'intermédiaire d'un système linéaire qui fait intervenir *toutes* les valeurs de f.

On préfère donc le plus souvent utiliser une idée plus simple : on écrit un polynôme d'interpolation au voisinage du point x_i et on dérive celui-ci. Les formules vont varier en fonction du nombre de points qu'on choisit pour écrire le polynôme d'interpolation (en général 2 ou 3, plus rarement 4 ou 5).

4.2.1.1 Formules à deux points

Le polynôme d'interpolation sur les deux points x_i, x_{i+1} s'écrit :

$$P(x) = f(x_i) + f[x_i, x_{i+1}](x - x_i).$$

On a donc $P'(x_i) = f[x_i, x_{i+1}]$, ce qui fournit la

$$\textbf{Formule décentrée à droite} \qquad f'(x_i) \simeq \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \,. \tag{4.3}$$

On a bien sûr aussi la

$$\textbf{Formule décentrée à gauche} \qquad f'(x_i) \simeq \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \,. \tag{4.4}$$

4.2.1.2 Formules à trois points

On choisit d'interpoler sur les points x_{i-1}, x_i, x_{i+1} (ce qui est moralement plus satisfaisant). Dans ce cas on a

$$P(x) = f(x_{i-1}) + f[x_{i-1}, x_i](x - x_{i-1}) + f[x_{i-1}, x_i, x_{i+1}](x - x_{i-1})(x - x_i).$$

On a donc

$$P'(x_i) = f[x_{i-1}, x_i] + f[x_{i-1}, x_i, x_{i+1}](x_i - x_{i-1}),$$

ce qui fournit, après simplification la
Formule centrée

$$f'(x_i) \simeq f(x_{i+1}) \frac{h_{i-1}}{h_i(h_{i-1} + h_i)} + f(x_i) \left(\frac{1}{h_{i-1}} - \frac{1}{h_i} \right) - f(x_{i-1}) \frac{h_i}{h_{i-1}(h_{i-1} + h_i)} \,. \tag{4.5}$$

La formule (4.5) se simplifie notablement dans le cas de points équidistants ($h_{i-1} = h_i = h$) pour donner

$$\textbf{Formule centrée - points équidistants} \qquad f'(x_i) \simeq \frac{f(x_{i+1}) - f(x_{i-1})}{2h} \,. \tag{4.6}$$

Remarquons que la formule ci-dessus n'est autre que la moyenne des deux formules décentrés dans le cas de points équidistants.

4.2.1.3 Erreur

Pour le calcul théorique de l'erreur commise quand on remplace $f'(x_i)$ par l'une des formules approchées ci-dessus, on revient à la démonstration effectuée dans le cadre de l'interpolation. Par exemple, dans le cas de la formule décentrée à gauche, on a, d'après (3.14)

$$f(x) = P(x) + f[x_{i-1}, x_i, x](x - x_{i-1})(x - x_i)$$

d'où, par dérivation et en faisant $x = x_i$

$$|f'(x_i) - P'(x_i)| = f[x_{i-1}, x_i, x_i](x_i - x_{i-1})$$

où $f[x_{i-1}, x_i, x_i]$ est définie comme en (3.42). On obtient finalement, grâce au Lemme 3.1 :

$$\left| f'(x_i) - \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \right| \leq \frac{M_2}{2} h_{i-1} \,. \tag{4.7}$$

On démontrerait de même les formules :

$$\left| f'(x_i) - \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \right| \leq \frac{M_2}{2} h_i \,. \tag{4.8}$$

et dans le cas centré, avec des points équidistants

$$\left| f'(x_i) - \frac{f(x_{i+1}) - f(x_{i-1})}{2h} \right| \leq \frac{M_3}{6} h^2 \,. \tag{4.9}$$

4.2.2 Dérivées d'ordre supérieur

Le principe est exactement le même, il faut simplement prendre garde que le degré du polynôme d'interpolation soit suffisant pour que sa dérivée n-ième soit non nulle!
Par exemple, pour la dérivée seconde, on choisit en général d'interpoler sur 3 points, ce qui donne (dans le cas de points équidistants)

Dérivée seconde - points équidistants
$$f''(x_i) \simeq \frac{f(x_{i+1}) + f(x_{i-1}) - 2f(x_i)}{h^2}\,. \tag{4.10}$$

avec une erreur majorée par

$$\left| f''(x_i) - \frac{f(x_{i+1}) + f(x_{i-1}) - 2f(x_i)}{h^2} \right| \leq \frac{M_4}{12}\, h^2\,. \tag{4.11}$$

Remarque 4.1 Application : les méthodes de différences finies. Les formules ci-dessus sont à la base des méthodes de différences finies pour résoudre de façon approchée des équations différentielles ou aux dérivées partielles. Ce thème sera développé plus loin dans le cours, en particulier dans le chapitre 6.

4.3 Intégration numérique : méthodes composites

4.3.1 Principe

On veut calculer $\int_a^b f(x)\,dx$. On décompose l'intervalle $[a,b]$ en $a = x_0 < x_1 < ... < x_n = b$. On a alors

$$\int_a^b f(x)\,dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)\,dx.$$

Sur chaque $[x_{i-1}, x_i]$, on applique une méthode d'intégration élémentaire consistant à remplacer $f(x)$ par

$$\begin{aligned} P_i(x) &= f[x_{i,0}] + f[x_{i,0}, x_{i,1}](x - x_{i,0}) + \dots && (4.12)\\ &\quad + f[x_{i,0}, x_{i,1}, ..., x_{i,l}](x - x_{i,0})...(x - x_{i,l})\,, && (4.13)\end{aligned}$$

soit le polynôme d'interpolation de f en des points $x_{i,0}, ..., x_{i,l}$ de l'intervalle $[a,b]$ (qui peuvent être ou non dans $[x_{i-1}, x_i]$).

4.3.2 Méthode des rectangles

On prend $l = 0$: on approche f par une constante $f(x_{i,0})$ où $x_{i,0} \in [x_{i-1}, x_i]$. D'où la formule de quadrature (i.e. intégration numérique) élémentaire :

$$\int_{x_{i-1}}^{x_i} f(x)\,dx \simeq h_i f(\zeta_i)\,, \quad \zeta_i \in [x_{i-1}, x_i]\,, \quad h_i = x_i - x_{i-1} \tag{4.14}$$

et la formule de quadrature composée :

$$\int_a^b f(x)\,dx \simeq \sum_{i=1}^{n} h_i f(\zeta_i)\,. \tag{4.15}$$

On reconnait là une somme de Riemann dont on sait qu'elle converge vers $\int_a^b f(x)\,dx$ quand $\max\limits_{1\leq i\leq n} h_i \longrightarrow 0$ si f est Riemann-intégrable. Les choix courants pour ζ_i sont :

– rectangles à gauche : $\zeta_i = x_{i-1} \ \rightarrow \ \int_a^b f(x)\,dx \simeq \sum_{i=1}^n h_i f(x_{i-1})$
– rectangles à droite : $\zeta_i = x_i \ \rightarrow \ \int_a^b f(x)\,dx \simeq \sum_{i=1}^n h_i f(x_i)$
– formule du point milieu : $\zeta_i = \dfrac{x_{i-1}+x_i}{2} (:= x_{i-\frac{1}{2}}) \ \rightarrow \ \displaystyle\int_a^b f(x)\,dx \simeq \sum_{i=1}^n h_i f\left(x_{i-\frac{1}{2}}\right)$

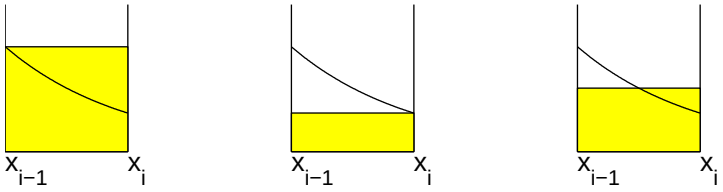

FIGURE 4.1 – Formule des rectangles à gauche, des rectangles à droite, du point milieu

La figure 4.1 suggère que la formule du point milieu doit fournir la meilleure approximation des trois en général (voir le paragraphe sur le calcul d'erreur).

4.3.3 Méthode des trapèzes

On prend $l = 1$ et on remplace f par son interpolé linéaire aux points $[x_{i-1}, x_i]$.

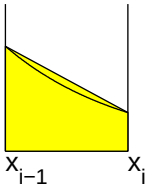

FIGURE 4.2 – Méthode des trapèzes

$$\int_{x_{i-1}}^{x_i} f(x)\,dx \simeq \frac{1}{2}(x_i - x_{i-1})(f(x_i) + f(x_{i-1})).$$

$$\int_a^b f(x)\,dx \simeq \sum_{i=1}^{n-1} \frac{h_i + h_{i+1}}{2} f(x_i) + \frac{1}{2}(h_1 f(x_0) + h_n f(x_n))$$

et dans le cas d'une subdivision régulière ($h_i = h,\ \forall i = 1, ..., n$) :

$$\int_a^b f(x)\,dx \simeq h\left[\sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}(f(x_0) + f(x_n))\right].$$

4.3.4 Méthode de Simpson

Cette fois on interpole f aux points x_{i-1}, x_i et $x_{i-\frac{1}{2}} = \frac{x_{i-1}+x_i}{2}$ soit, en utilisant la formule d'interpolation de Newton (3.5) :

$$\begin{aligned}\int_{x_{i-1}}^{x_i} f(x)\,dx &\simeq \int_{x_{i-1}}^{x_i} \left[f(x_{i-1}) + f[x_{i-1}, x_i](x - x_{i-1}) + f\left[x_{i-1}, x_i, x_{i-\frac{1}{2}}\right](x - x_{i-1})(x - x_i)\right] dx \\ &= h_i f(x_{i-1}) + \frac{1}{2}h_i^2 f[x_{i-1}, x_i] + f\left[x_{i-1}, x_i, x_{i-\frac{1}{2}}\right] \int_{x_{i-1}}^{x_i} (x - x_{i-1})(x - x_i)\,dx\end{aligned}$$

or

$$\int_{x_{i-1}}^{x_i} (x - x_{i-1})(x - x_i)\,dx = \frac{1}{3}h_i^3 - \frac{1}{2}h_i^3 = -\frac{h_i^3}{6} \quad \text{(en écrivant } x - x_i = x - x_{i-1} + x_{i-1} - x_i\text{)}$$

et

$$h_i^2 f[x_{i-1}, x_i] = h_i\left(f(x_i) - f(x_{i-1})\right)$$

$$\begin{aligned}h_i^3 f\left[x_{i-1}, x_{i-\frac{1}{2}}, x_i\right] &= h_i^2 \left(\frac{f(x_i) - f\left(x_{i-\frac{1}{2}}\right)}{\frac{h_i}{2}} - \frac{f\left(x_{i-\frac{1}{2}}\right) - f(x_{i-1})}{\frac{h_i}{2}}\right) \\ &= 2h_i\left(f(x_i) - 2f\left(x_{i-\frac{1}{2}}\right) + f(x_{i-1})\right).\end{aligned}$$

On obtient donc

$$\int_{x_{i-1}}^{x_i} f(x)\,dx = \frac{h_i}{6}\left(f(x_{i-1}) + 4f\left(x_{i-\frac{1}{2}}\right) + f(x_i)\right).$$

La formule composée dans le cas de pas constants devient

$$\int_a^b f(x)\,dx = \frac{h}{6}\left[f_0 + f_n + 2\sum_{i=1}^{n-1} f_i + 4\sum_{i=1}^{n} f_{i-\frac{1}{2}}\right]$$

où on note $f_i = f(x_i)$, $f_{i-\frac{1}{2}} := f\left(x_{i-\frac{1}{2}}\right) = f\left(\frac{x_{i-1}+x_i}{2}\right)$.

Remarque 4.2 Pour cette méthode, il est nécessaire de connaitre f aux points x_i et en leurs milieux, soit en un nombre impair de points. De manière générale, on appelle Formules de Newton-Cotes (fermées) les formules composées lorsque, sur chaque intervalle $[x_{i-1}, x_i]$, f est interpolée en l points équidistants. Les cas $l = 1$ et $l = 2$ correspondent donc respectivement aux formules des trapèzes et de Simpson, le cas $l = 4$ est connu dans la littérature sous le nom de Boole-Villarceau.

4.3.5 Méthode du trapèze corrigée

On utilise l'interpolation d'Hermite sur $[x_{i-1}, x_i]$: $P_i(x)$ est le polynôme de degré ≤ 3 tel que

$$\begin{aligned}P_i(x_{i-1}) &= f(x_{i-1}), \quad P_i(x_i) = f(x_i) \\ P_i'(x_{i-1}) &= f'(x_{i-1}), \quad P_i'(x_i) = f'(x_i).\end{aligned}$$

On obtient (par exemple à l'aide des formules de Newton) :

$$\int_{x_{i-1}}^{x_i} f(x)\,dx \simeq \frac{h_i}{2}\left(f(x_i) + f(x_{i-1})\right) + \frac{h_i^2}{12}\left(f'(x_{i-1}) - f'(x_i)\right).$$

On voit que ceci n'est qu'une amélioration de la méthode des trapèzes d'où son nom. Son inconvénient immédiat est qu'elle nécessite la connaissance des dérivées $f'(x_i)$. Cependant dans le cas d'un pas constant, ces dérivées disparaissent dans la formule composée pour donner :

$$\int_a^b f(x)\,dx \simeq h\left(\sum_{i=1}^{n-1} f_i + \frac{1}{2}(f_0 + f_n)\right) + \frac{h^2}{12}\left(f'(a) - f'(b)\right),$$

où seules apparaissent les dérivées en a et b.

4.4 Analyse de l'erreur dans les méthodes d'intégration

4.4.1 Théorie

Nous évaluons ici l'erreur faite lorsqu'on remplace $\int_a^b f(x)dx$ par $\int_a^b P(x)dx$ où P est le polynôme d'interpolation de f aux points $x_0, ..., x_N$. Ceci incluera le cas de l'interpolation d'Hermite lorsque certains des points x_i sont confondus. Nous savons qu'alors (voir chapitre précédent) les formules de Newton sont identiques.
On a vu au chapitre précédent que si $P_N(x)$ est le polynôme d'interpolation de f en $x_0, x_1, ..., x_N$ alors

$$f(x) - P_N(x) = f[x_0, x_1, ..., x_N, x]\,\psi_N(x) \tag{4.16}$$
$$\text{avec } \psi_N(x) = (x - x_0)(x - x_1)...(x - x_N). \tag{4.17}$$

Ainsi

$$E(f) = \int_a^b f(x)\,dx - \int_a^b P_N(x)\,dx = \int_a^b f[x_0, x_1, ..., x_N, x]\,\psi_N(x)\,. \tag{4.18}$$

L'expression de $E(f)$ se simplifie dans deux cas particuliers.

A $\psi_N(x)$ est de signe constant sur $[a, b]$: on peut alors utiliser le théorème de la moyenne pour les intégrales qui affirme que, si g et h sont continues sur $[a, b]$ et si $h(x) \geq 0$ sur $[a, b]$, il existe $\zeta \in [a, b]$ tel que

$$\int_a^b g(x)\,h(x)\,dx = g(\zeta)\int_a^b h(x)\,dx.$$

Appliqué à (4.18), ceci montre l'existence de $\zeta \in [a, b]$ tel que

$$E(f) = f[x_0, x_1, ..., x_N, \zeta]\int_a^b \psi_N(x)\,dx.$$

D'après le lemme 3.1 du chapitre précédent, on en déduit encore qu'il existe $\eta \in [c, d]$ ($[c, d]$ est un intervalle contenant les x_i, a et b) tel que

$$E(f) = \frac{1}{(N+1)!} f^{(N+1)}(\eta)\int_a^b \psi_N(x)\,dx. \tag{4.19}$$

(On supposera ici et dans toute la suite que f est assez régulière).

B $\int_a^b \psi_N(x)\,dx = 0$; dans ce cas, on utilise

$$f[x_0, x_1, ..., x_N, x] = f[x_0, x_1, ..., x_N, x_{N+1}] + (x_{N+1} - x) f[x_0, x_1, ..., x_N, x_{N+1}, x] \tag{4.20}$$

valable pour x_{N+1} arbitraire. Alors, (4.18) devient

$$E(f) = \int_a^b f[x_0, x_1, ..., x_N, x_{N+1}]\,\psi_N(x)\,dx + \int_a^b f[x_0, x_1, ..., x_N, x_{N+1}, x]\,\psi_{N+1}(x)\,dx$$

$$E(f) = \int_a^b f[x_0, x_1, ..., x_N, x_{N+1}, x]\,\psi_{N+1}(x)\,dx. \tag{4.21}$$

Si on peut choisir x_{N+1} de telle façon que $\psi_{N+1}(x)$ reste de signe constant sur $[a, b]$, on aura d'après (4.19)

$$E(f) = \frac{1}{(N+2)!} f^{(N+2)}(\eta) \int_a^b \psi_{N+1}(x)\,dx \tag{4.22}$$

$$\text{pour un } \eta \in [c, d] \text{ (intervalle contenant les } x_i, a \text{ et } b) \tag{4.23}$$

Définition 4.1 *On dit qu'une méthode d'intégration numérique est d'ordre k si elle est exacte pour tous les polynômes de degré inférieur ou égal à k.*

Remarque 4.3 Lorsque $\psi_N(x)$ est de signe constant sur $[a, b]$, d'après (4.19), on a une méthode d'ordre N puisque $P^{(N+1)}(\eta) = 0$ pour tout polynôme de degré inférieur ou égal à N. Dans la situation B), si (4.19) est vraie, on aura même une méthode d'ordre $N + 1$ (avec un même nombre de points).

4.4.2 Application aux méthodes usuelles

4.4.2.1 Méthode des rectangles (à gauche ou à droite)

$N = 0$, $x_0 = a$, $\psi_0(x) = x - a$. On peut appliquer (4.19) qui donne

$$E(f) = f'(\eta) \int_a^b (x - a)\,dx = \frac{(b-a)^2}{2} f'(\eta).$$

La méthode est d'ordre 0 et exacte seulement pour les constantes.

$$\left| \int_{x_{i-1}}^{x_i} f(x)\,dx - h_i f(x_{i-1}) \right| \leq \frac{h_i^2}{2} f'(\eta_i) \leq \frac{h_i^2}{2} M_1 \tag{4.24}$$

$$\text{où } M_1 := \max_{a \leq x \leq b} |f'(x)| \tag{4.25}$$

Pour la formule composée, il faut ajouter les erreurs provenant de chaque intégration élémentaire. Dans le cas d'un pas constant, on obtient :

$$\left| \int_a^b f(x)\,dx - h \sum_{i=1}^{N} f(x_{i-1}) \right| \leq \frac{M_1}{2} (b - a)\, h.$$

4.4.2.2 Formule du point milieu

$$N = 0, \quad x_0 = \frac{a+b}{2} \quad \psi_0(x) = x - \frac{a+b}{2}.$$

Ici, $\int_a^b \psi_0(x)\,dx = 0$. On va donc obtenir une méthode d'ordre 1 (exacte pour toute fonction linéaire) bien qu'on interpole par une constante. En effet, d'après (4.22) appliqué avec $x_{N+1} = x_0 = x_1 = \frac{a+b}{2}$

$$\left| \int_a^b f(x)\,dx - \frac{b-a}{2}\left(f(a)+f(b)\right) \right| \leq \frac{M_2}{2}\int_a^b \left(x - \frac{b-a}{2}\right)^2 dx = \frac{M_2}{24}(b-a)^3$$

avec $M_2 := \max_{a\leq x\leq b} |f''(x)|$. Pour la formule composée associée, on obtient

$$\left| \int_a^b f(x)\,dx - h\sum_{i=1}^{N} f\left(x_{i-1/2}\right) \right| \leq \frac{M_2}{24}h^2(b-a).$$

4.4.2.3 Méthode des trapèzes

$$N=1, \quad x_0 = a,\ x_1 = b \quad \psi_1(x) = (x-a)(x-b).$$

Comme ψ_1 est de signe constant sur $[a,b]$, on applique (4.19) pour trouver

$$\left| \int_a^b f(x)\,dx - \frac{(b-a)}{2}\left(f(a)+f(b)\right) \right| \leq \frac{|f''(\eta)|}{2}\int_a^b (x-a)(x-b)\,dx$$

soit

$$\left| \int_a^b f(x)\,dx - \frac{(b-a)}{2}\left(f(a)+f(b)\right) \right| \leq \frac{M_2}{12}(b-a)^3.$$

Pour la formule composée de la méthode des trapèzes, on en déduit que l'erreur est majorée par

$$E(f) \leq \frac{M_2}{12}h^2(b-a). \tag{4.26}$$

4.4.2.4 Méthode de Simpson

$$N=2, \quad x_0 = a,\ x_1 = \frac{a+b}{2},\ x_2 = b \quad \psi_2(x) = (x-a)\left(x - \frac{a+b}{2}\right)(x-b).$$

Par raison de symétrie, on vérifie que :

$$\int_a^b \psi_2(x)\,dx = 0.$$

Comme $\psi_3(x) = (x-a)\left(x-\frac{a+b}{2}\right)^2(x-b)$ est de signe constant sur $[a,b]$, on peut appliquer (4.22) pour obtenir

$$\left| \int_a^b f(x)\,dx - \frac{b-a}{6}\left(f(a) + 4f\left(\frac{b-a}{2}\right) + f(b)\right) \right| \leq \frac{M_4}{24}\int_a^b \psi_3(x)\,dx$$

soit tous calculs faits

$$|E(f)| \leq \frac{M_4}{90}\left(\frac{b-a}{2}\right)^5 \tag{4.27}$$

$$\left(M_4 = \max_{a\leq x\leq b}\left|f^{(4)}(x)\right|\right). \tag{4.28}$$

La formule composée correspondante avec pas constant h donnera une erreur majorée par

$$|E(f)| \leq \frac{M_4}{2880} h^4 (b-a).$$

Par ailleurs, la méthode de Simpson (bien que reposant sur une interpolation à trois points) est exacte pour tout polynôme de degré inférieur ou égal à 3.

4.4.2.5 Méthode des trapèzes corrigés

$$N = 3, \quad x_0 = x_1 = a,\ x_2 = x_3 = b \quad \psi_3(x) = (x-a)^2 (x-b)^2.$$

L'application de l'estimation (4.19) donne

$$|E(f)| \leq \frac{M_4}{720} (b-a)^5$$

et pour la formule composée :

$$|E(f)| \leq \frac{M_4}{720} h^4 (b-a).$$

4.4.3 Exemple d'application

Calcul de $\int_0^1 e^{-x^2} dx$.
Utilisons les méthodes élémentaires précédentes à l'aide des valeurs de $f(x) = e^{-x^2}$ aux points 0, $\frac{1}{2}$, 1 soit

$$f(0) = 1,\ f\left(\frac{1}{2}\right) = 0,77880,\ f(1) = e^{-1} = 0,36788.$$

Pour la méthode des trapèzes corrigés, on a aussi besoin de

$$f'(0) = 0,\ f'(1) = -2e^{-1} = -0,73576.$$

D'où les résultats

	rectangle	point milieu	trapèzes	Simpson	trapèzes corrigés
$\int_0^1 e^{-x^2} dx$	1	0,77880	0,68394	0,74718	0,74525
erreur	0,25318	0,03198	−0,06288	0,00036	−0,00157

La valeur exacte est 0,74682. Ces résultats sont tout à fait en accord avec les considérations théoriques précédentes. Noter comment la méthode de Simpson donne une approximation à 4.10^{-4} avec seulement 3 valeurs de f ! Ceci est bien sûr dû au fait que les dérivées d'ordre supérieur ne varient pas trop sur l'intervalle $[0,1]$.

4.5 Méthodes d'intégration numérique de Gauss

4.5.1 Introduction

Les méthodes composées décrites précédemment (sauf pour la méthode des trapèzes corrigés) consistent à remplacer $\int_a^b f(x)\,dx$ par une expression de la forme

$$A_0 f(x_0) + A_1 f(x_1) + ... + A_N f(x_N)$$

où les coefficients A_i sont indépendants de f et où les x_i sont N points de $[a,b]$. Au moins dans les exemples considérés (et c'est un fait général), les méthodes de type Newton-Cotes sont exactes pour les polynômes de degré inférieur ou égal à N (parfois même de degré $N+1$). Il est en fait possible avec le même nombre de points d'obtenir des formules exactes pour des polynômes de degré inférieur ou égal à $2N+1$: c'est le cas des méthodes de Gauss. Outre cet avantage, elles permettent également de calculer des intégrales singulières où la fonction f devient infinie en certains points par exemple

$$\int_0^1 \frac{e^{-x^2}}{\sqrt{x}} dx.$$

Pour recouvrir ce type de situation, on va considérer plus généralement des intégrales de la forme

$$\int_a^b g(x)\, w(x)\, dx$$

où w est une fonction poids qui, dans la pratique, contiendra la singularité (ainsi $w(x) = \frac{1}{\sqrt{x}}$ dans l'exemple ci-dessus).

4.5.2 Idée de base des méthodes de Gauss

Soit $x_0, ..., x_N \in [a,b]$ et $P_N(x)$ le polynôme d'interpolation de g aux points x_i. On a

$$g(x) = P_N(x) + g[x_0, x_1, ..., x_N, x]\, \psi_N(x) \tag{4.29}$$
$$\psi_N(x) = (x-x_0)(x-x_1)...(x-x_N). \tag{4.30}$$

Ceci donne en posant $I(g) = \int_a^b g(x)\, w(x)\, dx$,

$$I(g) - I(P_N) = \int_a^b g[x_0, x_1, ..., x_N, x]\, \psi_N(x)\, w(x)\, dx.$$

Supposons que

$$\int_a^b \psi_N(x)\, w(x)\, dx = 0.$$

Alors, raisonnant comme en (4.20), (4.21) et utilisant

$$g[x_0, x_1, ..., x_N, x] = g[x_0, x_1, ..., x_N, x_{N+1}] + (x - x_{N+1})\, g[x_0, x_1, ..., x_N, x_{N+1}, x]$$

pour x_{N+1} quelconque, on obtient avec $\psi_{N+1}(x) = \psi_N(x)(x - x_{N+1})$.

$$I(g) - I(P_N) = \int_a^b g[x_0, x_1, ..., x_N, x_{N+1}, x]\, \psi_{N+1}(x)\, w(x)\, dx.$$

Si de même

$$\int_a^b \psi_{N+1}(x)\, w(x)\, dx = 0$$

on peut répéter le procédé avec un nouveau point x_{N+2}. De façon générale si

$$\int_a^b \psi_N(x)(x - x_{N+1})...(x - x_{N+m})\, w(x)\, dx = 0,\ m = 0, 1, \ldots \tag{4.31}$$

on obtient à l'aide d'un nouveau point x_{N+m+1}

$$I(g) - I(P_N) = \int_a^b g[x_0, x_1, ..., x_N, x_{N+1}, ..., x_{N+m+1}, x]\, \psi_{N+m+1}(x)\, w(x)\, dx. \tag{4.32}$$

L'idée est de choisir les points $x_0, ...x_N$ initiaux de telle façon que le polynôme de degré $N+1$, $\psi_N(x)$ satisfasse (4.31), c'est-à-dire soit orthogonal au polynôme $(x-x_{N+1})...(x-x_{N+m})$ pour le produit scalaire associé au poids w.
Ainsi, si $P_0, P_1, ..., P_N, P_{N+1}$ est la suite de polynômes orthogonaux (normalisés) associés à ce produit scalaire, on sait déjà d'après la proposition 3.1 du chapitre précédent que :

$$(i)\, P_{N+1} = (x-\zeta_0)(x-\zeta_1)...(x-\zeta_N)$$

où les ζ_i sont réels et distincts :

$$(ii) \int_a^b P_{N+1}(x)\, q(x)\, w(x)\, dx = 0$$

pour tout polynôme q avec $\deg q \leq N$. D'où le choix :

$$\begin{array}{ll} \forall i = 0, 1, ..., N & x_i := \zeta_i \\ \forall i = N+1, ..., 2N+1 & x_i := \zeta_{2N+1-i}. \\ & m = N \end{array}$$

On a alors :

$$\psi_N(x) = P_{N+1}(x)$$

et d'après (4.31), (4.32)

$$I(g) - I(P_N) = \int_a^b g[x_0, ..., x_N, x_{N+1}, ..., x_{2N+1}] (P_{N+1}(x))^2 w(x)\, dx. \tag{4.33}$$

Comme $(P_{N+1}(x))^2$ est de signe constant, on peut appliquer le théorème de la moyenne à cette intégrale, puis appliquant le lemme 3.1 du chapitre précédent (comme dans (4.19)), on obtient l'existence de $\zeta \in [a,b]$ tel que :

$$I(g) - I(P_N) = \frac{g^{(2N+2)}(\zeta)}{(2N+2)!} \int_a^b (P_{N+1}(x))^2 w(x)\, dx. \tag{4.34}$$

On obtient donc une méthode d'intégration numérique d'ordre $2N+1$ avec seulement $N+1$ points !...

Calcul de $I(P_N)$:
Si on écrit le polynôme d'interpolation P_N sous sa forme de Lagrange, on obtient :

$$I(P_N) = \sum_{i=0}^{N} g(x_i) A_i, \text{ avec}$$

$$A_i = \int_a^b l_i(x)\, w(x)\, dx, \; m = N \quad l_i(x) = \prod_{j=0, j\neq i}^{N} \frac{x - x_j}{x_i - x_j}. \tag{4.35}$$

4.5.3 Mise en œuvre de la méthode

Comme application de ce qui précède, examinons tout d'abord le cas du calcul d'une intégrale du type $\int_a^b f(x)w(x)\, dx$ où a et b sont deux réels (finis). Comme les polynômes orthogonaux sont mieux connus sur $[-1,1]$, on s'y ramène toujours (quand on travaille sur un intervalle borné) à l'aide du changement de variable :

$$x = \frac{b+a}{2} + \frac{b-a}{2} t$$

Ainsi :

$$\int_a^b f(x)\, w(x)\, dx = \int_{-1}^{1} f(x(t))\, w(x(t)) x'(t)\, dt = \frac{b-a}{2} \int_{-1}^{1} f(x(t))\, w(x(t)) dt.$$

Ceci étant acquis, il suffit de se limiter aux intégrales du type :

$$I(g) = \int_{-1}^{1} g(x)\, w(x)\, dx.$$

4.5.3.1 Méthode de Legendre-Gauss

C'est le cas $w(x) \equiv 1$: Les polynômes orthogonaux appropriés sont ceux de Legendre, soit :

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_2(x) &= x^2 - \frac{1}{3},\ \zeta_0 = -\frac{1}{\sqrt{3}},\ \zeta_1 = \frac{1}{\sqrt{3}}. \end{aligned}$$

On a ainsi la formule à deux points ($N = 1$)

$$\begin{aligned} \int_{-1}^{1} g(x)\, dx &\approx A_0 g\left(-\frac{1}{\sqrt{3}}\right) + A_1 g\left(\frac{1}{\sqrt{3}}\right) \\ \text{avec } A_0 &= \int_{-1}^{1} \frac{x - \frac{1}{\sqrt{3}}}{-\frac{2}{\sqrt{3}}} dx = 1 \text{ voir (4.35)} \\ A_1 &= \int_{-1}^{1} \frac{x + \frac{1}{\sqrt{3}}}{\frac{2}{\sqrt{3}}} dx = 1. \end{aligned}$$

Plus généralement, si on utilise N points, la formule de Legendre-Gauss va s'écrire

$$\int_{-1}^{1} g(x)\, dx \approx \sum_{i=0}^{N} w_i g(z_i) \tag{4.36}$$

où les poids w_i et les racines z_i sont donnés par le tableau suivant. La quatrième colonne du tableau fournit **un majorant** de l'erreur (comme d'habitude M_n désigne la norme infinie de la dérivée n-ième de la fonction sur l'intervalle $[-1, 1]$. Voici comment sont établies ces formules d'erreur : par exemple pour deux points. D'après (4.34), l'erreur dans la formule à deux points est donnée par $E = \frac{g^{(4)}(\zeta)}{24} \int_{-1}^{1} \left(x^2 - \frac{1}{3}\right)^2 dx$. On vérifie $\int_{-1}^{1} \left(x^2 - \frac{1}{3}\right)^2 dx = \frac{8}{45}$ d'où $E = \frac{g^{(4)}(\zeta)}{135}$.

N	Racines z_i	Poids w_i	Formule d'erreur
1	± 0.5773502691	1	$\frac{M_4}{135}$
2	0	0.8888888888	$\frac{M_6}{15750}$
	± 0.7745966692	0.5555555555	
3	± 0.3399810435	0.6521451548	$\frac{M_8}{3472875}$
	± 0.8611363115	0.3478548451	
4	0	0.5688888888	$\frac{M_{10}}{1237732650}$
	± 0.5384693101	0.4786286704	
	± 0.9061798459	0.2369268850	
5	± 0.2386191860	0.4679139345	$\frac{M_{12}}{648984486150}$
	± 0.6612093864	0.3607615730	
	± 0.9324695142	0.1713244923	

(4.37)

Appliquons ce calcul à l'exemple déjà traité par les méthodes composées soit $\int_0^1 e^{-x^2}dx$.
On pose $x = \frac{1}{2}(1+t)$ et donc

$$\int_0^1 e^{-x^2}dx = \frac{1}{2}\int_{-1}^1 \exp\left(-\frac{(t+1)^2}{4}\right)dt \simeq \frac{1}{2}\left(\exp\left(\frac{-\left(1-\frac{1}{\sqrt{3}}\right)^2}{4}\right)+\exp\left(\frac{-\left(1+\frac{1}{\sqrt{3}}\right)^2}{4}\right)\right).$$

On obtient

$$\begin{aligned}\int_0^1 e^{-x^2}dx &\simeq \frac{1}{2}\left(\exp\left(-\frac{\left(\sqrt{3}-1\right)^2}{12}\right)+\exp\left(-\frac{\left(\sqrt{3}+1\right)^2}{12}\right)\right)\\ &\simeq \frac{1}{2}(0,95632+0,53687)\simeq 0,74659.\end{aligned}$$

Le résultat exact est $0,74682$ soit une erreur de $0,00033$ donc bien meilleure que celle des trapèzes ($-0,06288$ avec deux points) et du même ordre que celle de Simpson ($0,00036$ mais avec trois points).

Remarque 4.4 Comme je l'ai déjà signalé plus haut, il est parfaitement possible (en particulier quand l'intervalle d'intégration est grand) de reprendre l'idée des méthodes composites avec des formules de Gauss. Sur chaque sous-intervalle, on fera alors le changement de variable indiqué au début du paragraphe.

4.5.3.2 Méthode de Gauss-Chebyshev

C'est le cas $w(x) = \frac{1}{\sqrt{1-x^2}}$: Les polynômes appropriés sont les polynômes de Chebyshev dont les zéros sont (voir chapitre précédent)

$$\zeta_i = \cos\frac{(2i+1)\pi}{2(N+1)} \quad i = 0, ..., N.$$

Il se trouve que dans ce cas, tous les coefficients A_i sont égaux et on a la formule de quadrature très attrayante :

$$\int_{-1}^1 \frac{g(x)}{\sqrt{x^2-1}} \approx \frac{\pi}{N+1}\sum_{i=0}^N g\left[\cos\frac{(2i+1)\pi}{2(N+1)}\right]. \tag{4.38}$$

4.5.3.3 Méthodes de Laguerre et Hermite

La méthode de Laguerre correspond au cas $w(x) = e^{-x}$ sur l'intervalle $[a,b[= [0,+\infty[$, tandis que la méthode de Hermitee correspond au cas $w(x) = e^{-x^2}$ sur l'intervalle $]a,b[= [-\infty,+\infty[$. Les formules de récurrence permettant de calculer les polynômes orthogonaux correspondants sont donnés à la section 3.3.4. Nous renvoyons à la littérature pour les formules explicites et plus de détails.

4.6 Méthodes d'extrapolation à la limite

Nous allons décrire ici la méthode de Romberg qui consiste à appliquer la méthode d'extrapolation à la limite de Richardson à partir de la méthode classique des trapèzes dont on améliore ainsi la convergence.
La méthode de Richardson est de portée très générale. Nous allons la présenter ici, indépendamment de la méthode des trapèzes, sur un exemple simple.

4.6.1 Extrapolation à la limite de Richardson

Considérons l'approximation de $f'(a)$ à l'aide des différences finies centrées (voir le paragraphe 4.2.1), soit :

$$D_h f = \frac{f(a+h) - f(a-h)}{2h}. \tag{4.39}$$

On a

$$f'(a) = D_h f + c_1 h^2 + h^2 \varepsilon(h) \tag{4.40}$$

où $c_1 \neq 0$ si $f'''(a) \neq 0$. En effet, d'après la formule de Taylor-Young au voisinage de 0

$$\begin{aligned} f(a+h) &= f(a) + hf'(a) + \frac{h^2}{2} f''(a) + \frac{h^3}{6} f'''(a) + h^3 \varepsilon_1(h) \quad \lim_{h \longrightarrow 0} \varepsilon_1(h) = 0 \\ f(a-h) &= f(a) - hf'(a) + \frac{h^2}{2} f''(a) - \frac{h^3}{6} f'''(a) + h^3 \varepsilon_2(h) \quad \lim_{h \longrightarrow 0} \varepsilon_2(h) = 0. \end{aligned}$$

Par différence

$$f(a+h) - f(a-h) = 2hf'(a) + \frac{h^3}{3} f'''(a) + h^3 \varepsilon_3(h) \quad \lim_{h \longrightarrow 0} \varepsilon_3(h) = 0$$

d'où

$$f'(a) = D_h f - \frac{h^2}{6} f'''(a) + h^2 \varepsilon(h). \tag{4.41}$$

Le terme $h^2 \varepsilon(h)$ est négligeable devant $\frac{h^2}{6} f'''(a)$ si $f'''(a) \neq 0$ et si h est suffisamment petit. Il en résulte qu'une bonne approximation de l'erreur faite en remplaçant $f'(a)$ par $D_h f$ est donnée par $\frac{h^2}{6} f'''(a)$.
Une autre façon d'interpréter (4.41) consiste à remarquer que si on ajoute à $D_h f$ le terme correctif $-\frac{h^2}{6} f'''(a)$, on obtient une bien meilleure approximation de $f'(a)$. Le problème est que dans la pratique on ne connait pas $f'''(a)$! L'idée de base de la méthode de Richardson consiste à déterminer une valeur approchée de $f'''(a)$ en écrivant une 2ème relation obtenue à partir de (4.41) en remplaçant h par $2h$ (ou par $h/2$).
Soit donc

$$f'(a) = D_h f + C_1 h^2 + h^2 \varepsilon(h)$$

où C_1 est inconnu mais indépendant de h. Réécrivons ceci au point $2h$, on a

$$\begin{aligned} f'(a) &= D_{2h} f + 4C_1 h^2 + h^2 \varepsilon_1(h) \\ \text{où } \lim_{h \longrightarrow 0} \varepsilon_1(h) &= \lim_{h \longrightarrow 0} 4\varepsilon(2h) = 0. \end{aligned}$$

Par différence

$$3C_1 h^2 = D_h f - D_{2h} f + h^2 \varepsilon_2(h) \quad \lim_{h \longrightarrow 0} \varepsilon_2(h) = 0.$$

Puisque $h^2 \varepsilon_2(h)$ est "négligeable" devant les autres termes si h est petit et $C_1 \neq 0$, on a :

$$C_1 h^2 \approx \frac{1}{3} (D_h f - D_{2h} f)$$

ce qui donne une valeur approchée du terme correctif cherché en fonction des valeurs de la fonction f. Ainsi :

$$D_h^1 f = D_h f + \frac{1}{3} (D_h f - D_{2h} f) = \frac{4D_h f - D_{2h} f}{3} \tag{4.42}$$

devrait être une meilleure approximation que $D_h f$ de $f'(a)$. Nous allons le vérifier en effectuant un développement limité de f au voisinage de a à un ordre supérieur.

$$\begin{aligned} f(a+h) &= f(a) + hf'(a) + \frac{h^2}{2}f''(a) + \frac{h^3}{6}f'''(a) + \frac{h^4}{24}f^{(4)}(a) + \frac{h^5}{120}f^{(5)}(a) + h^5\varepsilon(h) \\ f(a-h) &= f(a) - hf'(a) + \frac{h^2}{2}f''(a) - \frac{h^3}{6}f'''(a) + \frac{h^4}{24}f^{(4)}(a) - \frac{h^5}{120}f^{(5)}(a) + h^5\varepsilon(h). \end{aligned}$$

Par différence

$$D_h f = f'(a) + \frac{h^2}{6}f'''(a) + \frac{h^4}{120}f^{(5)}(a) + h^4\varepsilon(h). \tag{4.43}$$

Remplaçons h par $2h$ dans cette relation :

$$D_{2h} f = f'(a) + \frac{4h^2}{6}f'''(a) + \frac{16h^4}{120}f^{(5)}(a) + h^4\varepsilon(h). \tag{4.44}$$

(Dans tout ceci et ce qui suit, on convient de noter $\varepsilon(h)$ toute fonction tendant vers 0 avec h).
De (4.42), (4.43), (4.44), on tire

$$D_h^1 f = f'(a) - h^4\frac{f^{(5)}(a)}{30} + h^4\varepsilon(h). \tag{4.45}$$

On obtient ainsi une approximation de $f'(a)$ d'ordre 4 (l'erreur tend vers 0 au moins aussi vite que h^4) et donc nettement meilleure que $D_h f$ qui était d'ordre 2. L'extrapolation à la limite de Richardson consiste donc à obtenir une approximation d'ordre supérieur à l'aide de deux approximations (ici d'ordre 2). On peut évidemment répéter l'idée à partir de (4.45) : partant des deux relations

$$D_h^1 f = f'(a) - C_4 h^4 + h^4\varepsilon(h)$$

$$D_{2h}^1 f = f'(a) - 16C_4 h^4 + h^4\varepsilon(h)$$

on obtient

$$15C_4 h^4 = D_h^1 f - D_{2h}^1 f + h^4\varepsilon(h)$$

d'où une évaluation du terme correctif $C_4 h^4$, et une nouvelle approximation de $f'(a)$

$$D_h^2 f = D_h^1 f + \frac{1}{15}\left(D_h^1 f - D_{2h}^1 f\right) = \frac{16D_h^1 f - D_{2h}^1 f}{16-1}.$$

Par un calcul analogue au précédent, on montre que :

$$D_h^2 f = f'(a) + C_6 h^6 + h^6\varepsilon(h).$$

Remarque 4.5 Afin de s'assurer que $\frac{1}{3}(D_h f - D_{2h} f) \approx C_1 h^2$ ou que $\frac{1}{15}\left(D_h^1 f - D_{2h} f\right) \approx C_4 h^4$ sont des approximations raisonnables, on peut évaluer les quotients

$$\frac{D_h f - D_{2h} f}{D_{h/2} f - D_h f} \approx 4\frac{D_h^1 f - D_{2h}^1 f}{D_{h/2}^1 f - D_h^1 f} \approx 16.$$

Ils doivent être respectivement voisins de 4 et 16 si les approximations en question sont valables. Il est conseillé de le vérifier dans la pratique.

Remarque 4.6 Le nombre 2 utilisé dans le passage de h à $2h$ n'a rien de sacré. Le même travail peut être effectué à l'aide de

$$\begin{aligned} f'(a) &= D_h f + C_1 h^2 + h^2 \varepsilon(h) \\ f'(a) &= D_{qh} f + C_1 q^2 h^2 \varepsilon(h), \ q > 0, q \neq 1 \\ &\rightarrow C_1 h^2 \approx \frac{D_h f - D_{qh} f}{q^2 - 1}. \end{aligned}$$

Si on pose

$$D_{h,q}(f) = D_h f + \frac{D_h f - D_{qh} f}{q^2 - 1}$$

on a également

$$D_{h,q}(f) = f'(a) + C_q h^4 + h^4 \varepsilon(h).$$

Remarque 4.7 La technique d'extrapolation à la limite décrite ici sur l'exemple des approximations $D_h f$ de $f'(a)$ s'applique en fait à toute approximation D_h d'une quantité D qui, au voisinage de $h = 0$ s'écrit sous la forme

$$D_h = D + hD_1 + h^2 D_2 + ... + h^p D_p + h^p \varepsilon(h).$$

Nous allons maintenant l'appliquer à l'approximation de l'intégrale d'une fonction par la méthode des trapèzes.

4.6.2 Méthode de Romberg

Nous avons vu au paragraphe 4.3.3 que la méthode des trapèzes consistait à remplacer $\int_a^b f(x)\,dx$ par

$$T_N = h\left[\sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}(f(x_0) + f(x_N))\right]$$

où $h = \frac{b-a}{N}$ et $x_i = a + i\frac{b-a}{N}$ (dans le cas d'une subdivision régulière, ce que nous supposerons toujours dans la suite).
D'après (4.26), on a

$$\int_a^b f(x)\,dx = T_N + C_2 h^2 + O\left(h^4\right)$$

où C_2 est une constante indépendante de h et $O\left(h^4\right)$ une fonction tendant vers 0 au moins aussi vite que h^4. La situation est donc analogue à l'exemple traité précédemment et la méthode de Richardson s'applique. Ainsi

$$T_{N,q} = T_N + \frac{T_N - T_{N/q}}{q^2 - 1} \tag{4.46}$$

fournit une approximation en h^4 $\left(h = \frac{b-a}{N}\right)$ de $\int_a^b f(x)\,dx$. La situation présente nécessite que $\frac{N}{q}$ soit un entier. On choisir en général $q = 2$ et N pair. Ceci a de plus l'avantage que les évaluations de fonctions faites pour $T_{N/2}$ peuvent servir pour le calcul de T_N. Ainsi pour N pair, on a

$$\begin{aligned} T_N &= h\left(\sum_{i=1}^{N-1} f(a+ih)\right) + \frac{1}{2}(f(a) + f(b)) \\ &= h\left[\sum_{j=1}^{N/2} f(a + (2j-1)h)\right] + h\left[\sum_{j=1}^{N/2-1} f(a+2jh)\right] + h\frac{f(a)+f(b)}{2} \end{aligned}$$

soit

$$T_N = \frac{1}{2}T_{N/2} + h\sum_{j=1}^{N/2} f\left(a + \frac{2i-1}{h}\right). \tag{4.47}$$

La formule (4.46) peut être réitérée. On montre, en effet, que si f est $(2k+2)$ fois continuement dérivable, alors

$$\int_a^b f(x)\,dx = T_N + C_1h^2 + C_2h^4 + ... + C_kh^{2k} + O\left(h^{2k+2}\right).$$

Posons, comme dans (4.46) :

$$T_N^1 = T_N + \frac{T_N - T_{N/2}}{4-1}. \tag{4.48}$$

On montre alors que

$$\int_a^b f(x)\,dx = T_N^1 + C_2^1h^4 + C_3^1h^6 + ... + C_k^1h^{2k} + O\left(h^{2k+2}\right).$$

Pour $m = 1, ..., k$, on pose alors successivement :

$$T_N^m = T_N^{m-1} + \frac{T_N^{m-1} - T_{N/2}^{m-1}}{4^m - 1} \tag{4.49}$$

et T_N^m est une approximation de$\int_a^b f(x)\,dx$ en $O\left(h^{2m+2}\right)$.
La formule (4.49) définit la méthode d'intégration de Romberg. Notons que le calcul de T_N^m nécessite celui de T_N^{m-1} et de $T_{N/2}^{m-1}$. De même, celui de $T_{N/2}^{m-1}$nécessite celui de $T_{N/2}^{m-2}$ et $T_{N/4}^{m-2}$ et ainsi de suite. En particulier, on devra calculer $T_{N/2^m}$, $T_{N/2^{m-1}}$, $...T_{N/2}$ et $N/2^m$ doit donc être un entier. Posons

$$M = \frac{N}{2^m}.$$

On peut visualiser les termes successifs en le tableau triangulaire suivant :

$$\begin{array}{c} T_M^0 \\ T_{2M}^0 \quad T_{2M}^1 \\ T_{4M}^0 \quad T_{4M}^1 \quad T_{4M}^2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ T_{2^mM}^0 \; T_{2^mM}^1 \ldots\ldots\ldots\ldots T_{2^mM}^m. \end{array}$$

Algorithme de Romberg : on se donne $f : [a,b] \longmapsto R$ et un entier positif M.

$$\left[\begin{array}{c} h := \frac{b-a}{M} \\ T_M^0 := h\left(\sum_{i=1}^{m-1} f(a+ih) + \frac{f(a)+f(b)}{2}\right) \\ \text{pour } k = 1, 2, ... \\ h := \frac{h}{2} \\ T_{2^kM}^0 := \frac{1}{2}T_{2^{k-1}M}^0 + h\sum_{i=1}^{2^{k-1}M} f(a+(2i-1)h) \\ \text{De } m = 1 \text{ à } k \\ T_{2^kM}^m := T_{2^kM}^{m-1} + \left(T_{2^kM}^{m-1} - T_{2^{k-1}M}^{m-1}\right)/\left(4^m - 1\right). \end{array}\right.$$

4.7 Exercices du chapitre 4

Exercice 4.1 Calculer $\int_1^3 \frac{(\sin x)^2}{x} dx$ en utilisant successivement

a) la méthode des rectangles sur 5 sous-intervalles,

b) la méthode des trapèzes sur 5 sous-intervalles,

c) la méthode de Simpson sur 5 sous-intervalles,

d) la méthode de Gauss à 5 points

Comparer les résultats et le nombre d'évaluations de fonctions nécessaires par chaque méthode.

Exercice 4.2 Déterminer le nombre de points nécessaires pour évaluer l'intégrale $\int_0^1 \cos(x^2)\, dx$ avec une précision de 10^{-8} par la méthode des rectangles à gauche, la méthode des trapèzes, la méthode de Simpson et la méthode de Gauss composite à 3 points.

Exercice 4.3 Imaginer une méthode de Gauss pour calculer les intégrales

$$\int_0^1 (\cos x)^\alpha \log x \, dx \text{ pour } \alpha = \frac{1}{2}, 1, \frac{3}{2} 2, \frac{5}{2}.$$

Déterminer les premiers polynômes orthogonaux et leurs racines et mettre en oeuvre la méthode de Gauss.

Exercice 4.4 Soit K le carré unité de $\mathbb{R}^2$. Imaginer une formule d'intégration numérique sur K qui soit exacte pour les polynômes de degré inférieur ou égal à 3 en x et en y.

Exercice 4.5 Déterminer les 4 premiers polynômes orthogonaux pour le produit scalaire

$$\int_1^{+\infty} f(t)g(t)e^{-t}\, dt.$$

En déduire une formule d'intégration numérique pour calculer des intégrales du type $\int_1^{+\infty} f(t)e^{-t}\, dt$ qui soit exacte pour les polynômes de degré inférieur ou égal à 7. Tester cette formule avec $f(t) = t^5$, $f(t) = t^6$ et $f(t) = e^{t/2}$.

Exercice 4.6 Calculer $\int_{0.5}^{+\infty} e^{-t^3}\, dt$ avec 4 décimales exactes.

Chapitre 5

Résolution numérique d'équations différentielles

Il s'agit de résoudre numériquement les systèmes d'équations différentielles du type

$$\left\{ \begin{array}{l} y'(t) = f\left(t, y(t)\right),\ t > 0 \\ y\left(0\right) = y_0 \qquad \text{donné} \end{array} \right. \tag{5.1}$$

où $y : [0, +\infty[\to \mathbb{R}^N$ est une fonction vectorielle inconnue de la variable t (représentant le plus souvent le temps), y_0 un vecteur donné de $\mathbb{R}^N$ décrivant l'état initial et f une application de $[0, +\infty[\times \mathbb{R}^N$ dans $\mathbb{R}^N$.

5.1 Quelques remarques sur (5.1)

Si $y\left(t\right) = \left(y_1\left(t\right), y_2\left(t\right), ..., y_N\left(t\right)\right)$, le système (5.1) s'écrit

$$\left\{ \begin{array}{l} y_1'\left(t\right) = f\left(t, y_1\left(t\right), ..., y_N\left(t\right)\right) \\ y_2'\left(t\right) = f\left(t, y_1\left(t\right), ..., y_N\left(t\right)\right) \\ \vdots \\ y_N'\left(t\right) = f\left(t, y_1\left(t\right), ..., y_N\left(t\right)\right) \end{array} \right. , \quad y_i\left(0\right) = y_0^i \qquad i = 1, ..., N. \tag{5.2}$$

La discrétisation spatiale d'équations aux dérivées partielles modélisant les phénomènes d'évolution est une source importante de tels problèmes. Ils apparaissent aussi souvent directement sous la forme (5.2) dans les applications. Notons qu'une équation d'ordre strictement supérieur à 1 peut-être aussi écrite sous la forme (5.2). Considérons par exemple le problème d'ordre 2 :

$$\left\{ \begin{array}{l} y''\left(t\right) = g\left(t, y\left(t\right), y'\left(t\right)\right) \\ y\left(0\right) = y_0,\ y'\left(0\right) = y_1 \end{array} \right. \tag{5.3}$$

où $y : [0, +\infty[\to \mathbb{R}$ est la fonction inconnue, $g : [0, +\infty[\times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, y_0 et y_1 sont des réels donnés (position et vitesse initiales). Posant alors

$$u_1\left(t\right) = y\left(t\right),\ u_2\left(t\right) = y'\left(t\right),$$

on remplace (5.3) par un système équivalent où l'inconnue est le vecteur $U\left(t\right) = \left(\begin{array}{c} u_1\left(t\right) \\ u_2\left(t\right) \end{array} \right)$ soit

$$\frac{dU}{dt}\left(t\right) = \left(\begin{array}{c} u_2\left(t\right) \\ g\left(t, u_1\left(t\right), u_2\left(t\right)\right) \end{array} \right),\ U\left(0\right) = \left(\begin{array}{c} y_0 \\ y_1 \end{array} \right)$$

ce qui est un système de la forme (5.2) avec $N = 2$ et $f(t, u_1, u_2) = \begin{pmatrix} u_2 \\ g(t, u_1, u_2) \end{pmatrix}$.

La même remarque vaut pour les équations d'ordre p

$$\begin{cases} y^{(p)}(t) = g\left(t, y(t), ..., y^{p-1}(t)\right) \\ y^{(p-1)}(0), ..., y'(0), y(0) \text{ donnés} \end{cases}$$

et les méthodes numériques que nous allons décrire pourront leur être appliquées. Nous verrons cependant, en fin de chapitre, une méthode particulière à (5.3) ; la méthode dite de Newmark.

5.1.1 Quelques soucis d'ordre théorique

Avant d'aborder l'étude numérique de (5.1), il est bon d'avoir à l'esprit quelques idées simples sur l'existence et le comportement des solutions de (5.1) :

Exemples linéaires :

$$\begin{cases} y' = ay \\ y(0) = y_0 \end{cases}, \ a \in \mathbb{R}$$

La solution est bien sûr $y(t) = y_0 e^{at}$. Bien qu'élémentaire, cet exemple nous permettra d'analyser quelques propriétés importantes des algorithmes étudiés plus loin.
Plus généralement, si a dépend de t, l'unique solution est donnée par

$$y(t) = y_0 \exp\left(\int_0^t a(\sigma)\, d\sigma\right)$$

dès que a est une fonction continue de t.
Considérons

$$\begin{cases} y' = Ay \\ y(0) = y_0 \in \mathbb{R}^N \end{cases} \tag{5.4}$$

où $y : [0, +\infty[\to \mathbb{R}^N$ et A est une matrice d'ordre N. Si A est diagonale, le système se découple en N équations scalaires indépendantes. On a alors

$$y_i(t) = y_0^i e^{\lambda_i t} \quad \text{si} \quad A = \begin{bmatrix} \ddots & & 0 \\ & \lambda_i & \\ 0 & & \ddots \end{bmatrix}.$$

Plus généralement, la solution est encore donnée par

$$y(t) = e^{tA} y_0$$

où e^{tA} est la matrice définie par :

$$e^{tA} = \sum_{n=0}^{\infty} \frac{t^n A^n}{n!}.$$

Lorsque la matrice A dépend raisonnablement du temps, on montre encore que (5.4) a une solution unique existant pour tout $t > 0$. Ces deux propriétés peuvent malheureusement aisément disparaître lorsque la fonction f est non linéaire.

Exemples non linéaires

$$\begin{cases} y' = y^2 \\ y(0) = y_0 \end{cases} \tag{5.5}$$

$$\begin{cases} y' = y^{\frac{1}{3}} \\ y(0) = y_0 \end{cases} \tag{5.6}$$

L'équation (5.5) s'intègre explicitement puisque, sur un intervalle où y ne s'annule pas :

$$y' = y^2 \Longleftrightarrow y' y^{-2} = 1 \Longleftrightarrow \frac{d}{dt}\left(-\frac{1}{y}\right) = 1 \Longleftrightarrow \frac{1}{y_0} - \frac{1}{y(t)} = t \Longleftrightarrow y(t) = \frac{y_0}{1 - t y_0}.$$

Ainsi, pour $y_0 > 0$, la solution explose pour $t^* = \frac{1}{y_0}$: il n'y a donc pas existence globale pour tout $t > 0$.
L'équation (5.6) s'intègre de façon identique : on obtient

$$y' y^{-\frac{1}{3}} = 1 \Longleftrightarrow \frac{d}{dt}\left(\frac{3}{2} y^{\frac{2}{3}}\right) = 1 \Longleftrightarrow \frac{3}{2} y^{\frac{2}{3}}(t) = \frac{3}{2} y_0^{\frac{2}{3}} + t.$$

En particulier, si $y_0 = 0$, on obtient comme solution

$$y(t) = \left(\frac{2}{3} t\right)^{\frac{3}{2}}.$$

Seulement, on vérifie que $y(t) = 0$ est aussi solution. Donc, il n'y a pas unicité des solutions de (5.6). En fait il y en a une infinité puisque pour tout $a > 0$

$$y_a(t) = \begin{cases} \left(\frac{2}{3}(t-a)\right)^{\frac{3}{2}} & \text{si } \ t \geq a \\ 0 & \text{si } \ 0 \leq t \leq a \end{cases}$$

est aussi une solution. Enonçons sans démonstration un théorème classique relativement

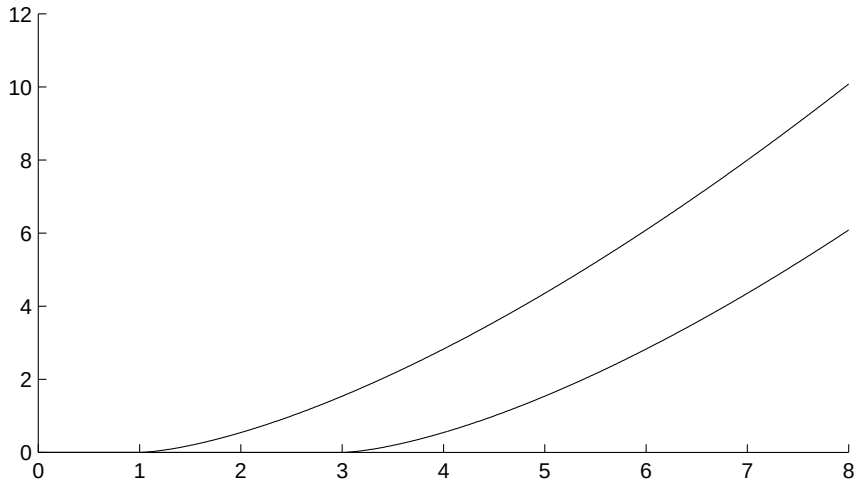

FIGURE 5.1 – Deux solutions distinctes de l'équation différentielle (5.6)

général assurant existence et unicité locale.

Théorème 5.1 *i) On suppose que f est continue de $[0,\alpha] \times \mathbb{R}^N \to \mathbb{R}^N$ avec $\alpha > 0$. Alors, pour tout y_0, il existe au moins une solution continument dérivable sur $[0,T]$ de :*

$$\begin{cases} y'(t) = f(t, y(t)) & 0 \leq t \leq T \\ y(0) = y_0 \end{cases} \tag{5.7}$$

pour un certain $T \in]0, \alpha]$.
ii) On suppose de plus que f *est localement lipschitzienne par rapport à la variable* y, *c'est-à-dire :*

$$\|f(t, y_1) - f(t, y_2)\| \leq L(M) \|y_1 - y_2\| \text{ pour tout } t \in [0, T], \tag{5.8}$$

$$\text{pour tout } \quad y_1,\ y_2 \text{ avec } \|y_1\| \leq M,\ \|y_2\| \leq M$$

où L *est une fonction croissante de* $[0, \infty[$ *dans* $[0, \infty[$ *et* $\|.\|$ *une norme sur* $\mathbb{R}^N$.
Alors, la solution de (5.7) est unique.

Remarque 5.1 Le point i) justifie l'existence de solutions locales pour les problèmes (5.5),(5.6) (voir aussi l'exercice 5.2). En revanche, on voit qu'elles ne sont pas nécessairement globales.

Les exemples (5.4) et (5.5) entrent dans le cadre (ii). En effet :

$$\left|y_1^2 - y_2^2\right| = |y_1 + y_2| \, |y_1 - y_2| \leq 2M \, |y_1 - y_2| \text{ si } |y_1|,\ |y_2| \leq M.$$

Par contre $y \to y^{\frac{1}{3}}$ n'est pas localement lipschitzienne.
On peut en fait montrer que les conclusions de ii) restent valables lorsque (5.8) est remplacé par l'hypothèse de monotonie (plus faible que (5.8)) :

$$\langle f(t, y_1) - f(t, y_2), y_1 - y_2 \rangle \leq L(M) \|y_1 - y_2\|^2 \tag{5.9}$$

où $\langle ., . \rangle$ est le produit scalaire usuel sur $\mathbb{R}^N$ et $\|.\|$ la norme associée.
Ainsi, dans l'exercice 5.2, f vérifie (5.9) avec $L(M) \equiv 0$ (alors qu'elle ne vérifie pas (5.8).

Les hypothèses de monotonie assurent aussi l'existence globale :

Théorème 5.2 *On suppose que* f *est continue sur* $[0, +\infty[\times \mathbb{R}^N$ *et que*

$$\langle f(t, y_1) - f(t, y_2), y_1 - y_2 \rangle \leq L \|y_1 - y_2\|^2 \ \forall y_1, y_2 \in \mathbb{R}^N,\ \forall t > 0. \tag{5.10}$$

Alors le problème (5.7) admet une solution et une seule sur $[0, \infty[$.

Remarque 5.2 Les hypothèses (5.10) recouvrent le cas où f est globalement lipschitzienne en x, soit

$$\|f(t, y_1) - f(t, y_2)\| \leq L \|y_1 - y_2\|. \tag{5.11}$$

C'est en particulier le cas si $f(t, y) = A(t) y$ où $A(t)$ est une matrice à coefficients continus et bornés sur $[0, +\infty[$.

Nous renvoyons à la littérature pour une démonstration de ces résultats (par exemple : "Analyse Numérique des équations différentielles", par M. Crouzeix et A.L. Mignot, Masson).

5.1.2 Régularité de la solution

Par définition, une solution de (5.7) est une fonction dérivable sur $[0, T[$ satisfaisant

$$y'(t) = f(t, y(t)) \quad \forall t \in [0, T],\ y(0) = y_0. \tag{5.12}$$

Si f est continue, par composition, y' est continue. Donc y est une fonction continument dérivable : y est de classe C^1.

Supposons $N = 1$ et que f soit non seulement continue, mais aussi de classe C^1, i.e. $\frac{\partial f}{\partial t}(t,y)$ et $\frac{\partial f}{\partial y}(t,y)$ existent et sont continues sur $[0,T]$. Alors, d'après (5.12), $y'(t)$ est lui-même dérivable et

$$y''(t) = \frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t))\, y'(t).$$

Ceci montre que y'' est continue : y est de classe C^2. Le raisonnement peut être reconduit : y sera de classe C^{p+1} si f est de classe C^p. Par ailleurs, si $\frac{\partial f}{\partial y}$ est continue, d'après le théorème des accroissements finis, on a, pour $|y_1|,\ |y_2| \leq M,\ t \in [0,T]$

$$|f(t,y_1) - f(t,y_2)| \leq L(M)\, \|y_1 - y_2\|$$

où $L(M) = \max_{|y|\leq M,\ 0\leq t\leq T} \left|\frac{\partial f}{\partial t}(t,y)\right|$ et donc f est localement lipschitzienne.

L'hypothèse f est de classe C^1 nous assure existence, unicité, et régularité locale de la solution.

Ces remarques ne sont pas particulières à $N = 1$ mais restent valables pour les systèmes : seules les écritures sont plus lourdes. Ainsi, $\frac{\partial f}{\partial y}$ est remplacée par la différentielle de f. Si

$$f(t, y_1, y_2, ..., y_N) = \begin{pmatrix} f_1(t, y_1, y_2, ..., y_N) \\ \vdots \\ f_N(t, y_1, y_2, ..., y_N) \end{pmatrix}$$

alors nous noterons

$$D_y f(t,y) = \begin{bmatrix} \frac{\partial f_1}{\partial y_1}(t,y) & \frac{\partial f_1}{\partial y_2}(t,y) & \cdots & \frac{\partial f_1}{\partial y_N}(t,y) \\ \frac{\partial f_2}{\partial y_1}(t,y) & \cdots & \cdots & \vdots \\ \vdots & & & \vdots \\ \frac{\partial f_N}{\partial y_1}(t,y) & \cdots & \cdots & \frac{\partial f_N}{\partial y_N}(t,y) \end{bmatrix}$$

la matrice Jacobienne de f. Dérivant (5.12), on a alors

$$y''(t) = \frac{\partial f}{\partial t}(t, y(t)) + D_y f(t, y(t))\, y'(t)$$

ou

$$y''(t) = \frac{\partial f}{\partial t}(t, y(t)) + D_y f(t, y(t))\, f(t, y(t))$$

5.2 Méthodes de résolution à un pas

Nous allons commencer par décrire une méthode très simple mais fondamentale : nous mettrons à profit sa simplicité pour en analyser son erreur de discrétisation, sa consistance, sa convergence et sa stabilité, autant de notions clefs dans l'analyse numérique des équations différentielles.

5.2.1 La méthode d'Euler

Soit donc à résoudre numériquement

$$\begin{cases} y'(t) = f(t, y(t)) \quad 0 \leq t \leq T \\ y(0) = y_0 \end{cases} \tag{5.13}$$

où $y : [0,+\infty[\to \mathbb{R}^N$ est la fonction cherchée, $f : [0,+\infty[\times \mathbb{R}^N \to \mathbb{R}^N$, $y_0 \in \mathbb{R}^N$. Nous supposerons que (5.13) admet une solution sur $[0,T]$. Nous noterons $M_0 = \max_{0\leq t\leq T} \|y(t)\|$ où $\|.\|$ est une norme sur $\mathbb{R}^N$ et nous supposerons que f est de classe C^1 sur $[0,T] \times \mathbb{R}^N$. Ainsi, d'après les remarques précédentes, nous aurons

$$\forall y_1, y_2 \text{ avec } \|y_1\|, \|y_2\| \leq M \qquad \|f(t,y_1) - f(t,y_2)\| \leq L(M) \|y_1 - y_2\| \tag{5.14}$$

$$\forall t \in [0,T] \quad \|y'(t)\| \leq M_1, \ \|y''(t)\| \leq M_2.$$

Nous allons discrétiser (5.13) en introduisant une subdivision

$$0 = t_0 < t_1 < t_2 < ... < t_n = T.$$

On notera $h_n = t_{n+1} - t_n$, $n = 0, ..., N-1$ et $h = \max_{0\leq n\leq N} h_n$. Alors (5.13) implique

$$\begin{cases} \text{pour } n = 0, ..., N-1 & y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+1}} f(t, y(t))\, dt \\ y(0) = y^0 \end{cases} . \tag{5.15}$$

Les méthodes numériques utilisées pour résoudre (5.13) diffèrent par le choix de l'évaluation numérique des intégrales $\int_{t_n}^{t_{n+1}} f(t, y(t))\, dt$.
La méthode d'Euler classique correspond à une méthode des rectangles à gauche, soit :

$$\begin{cases} \text{pour } n = 0, ..., N-1 & y_{n+1} = y_n + h_n f(t_n, y_n). \\ y_0 = y^h \text{ (valeur aprochée de } y^0). \end{cases} \tag{5.16}$$

Il est nécessaire d'analyser dans quelle mesure la valeur calculée approche suffisamment

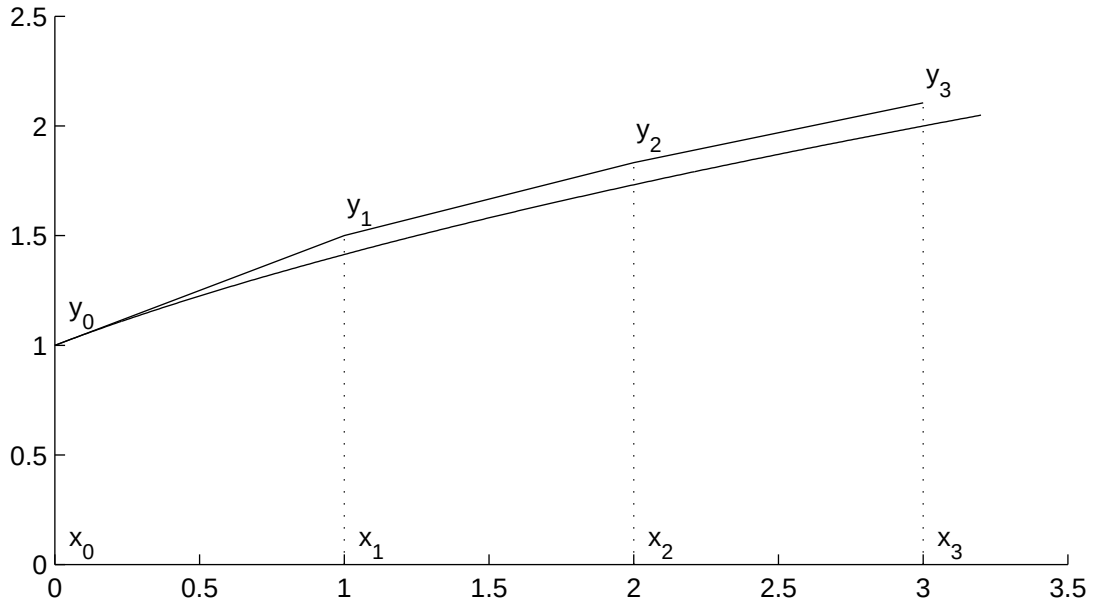

FIGURE 5.2 – La méthode d'Euler pour l'équation y'=1/(2y) avec un pas constant $h_n = 1$: $y_{n+1} = y_n + 1/(2y_n)$

la valeur exacte $y(t_n)$ et donc d'évaluer l'erreur de discrétisation

$$e_n = y(t_n) - y_n.$$

Définition 5.1 *On dira que la méthode est convergente si*

$$\max_{0\leq n\leq N}\|e_n\| \text{ tend vers 0 lorsque } h \text{ tend vers 0 et } y^h \text{ tend vers } y^0.$$

Remarque 5.3 si la méthode est convergente, en choisissant h assez petit, et y^h assez voisin de y^0, on obtient une bonne approximation de $y(t_n),\ n=0,...,N-1$ à l'aide du schéma (5.16). Il semblerait plus naturel de poser directement $y_0=y^0$ dans (5.16). Cependant, dans la pratique, si y^0 est réel, il ne pourra pas toujours être pris en compte de façon exacte à cause des arrondis. Une analyse correcte nécessite donc de supposer $y^0\neq y^h$.

De plus, comme dans tout calcul numérique, il se pose un problème de stabilité, il est nécessaire de connaître les conséquences d'une petite variation de y^h (inévitable dans la pratique) ainsi que celles d'erreurs (également inévitables) sur le calcul de $f(t_n,y_n)$.

Définition 5.2 *Nous dirons que la méthode (5.16) est stable s'il existe une constante K telle que*

$$\max_n\|y_n-z_n\|\leq K\left[\|y_0-z_0\|+\sum_{n=0}^{N-1}\|\varepsilon_n\|\right] \tag{5.17}$$

pour tout z_n solution de

$$z_{n+1}=z_n+h_nf(t_n,z_n)+\varepsilon_n,\ n=0,...,N-1.$$

Cette notion de stabilité implique que de petites perturbations sur les données et sur tous les calculs intermédiaires n'entraînent que de "petites" perturbations du résultat ce qui est absolument nécessaire pour qu'un schéma numérique soit réaliste. Cependant, cette remarque n'est valable de façon pratique que si K est de taille raisonnable (voir exemples plus loin).
Enfin, on dégage aussi une notion de consistance d'un schéma numérique : l'erreur de consistance représente l'erreur qu'on fait au n-ième pas en remplaçant l'équation différentielle par l'équation discrétisée, soit ici :

$$\varepsilon_n=y(t_{n+1})-y(t_n)-h_nf(t_n,y(t_n)). \tag{5.18}$$

Définition 5.3 *On dit que la méthode est consistante si*

$$\lim_{h\to 0}\sum_{n=0}^{N-1}\|\varepsilon_n\|=0.$$

Remarque 5.4 On montre que, pour les méthodes à un pas, consistance et stabilité impliquent convergence. Nous allons le constater ici pour la méthode d'Euler.

5.2.1.1 Estimation de l'erreur de discrétisation

$$e_{n+1}=y(t_{n+1})-y_{n+1}=y(t_{n+1})-y_n-h_nf(t_n,y_n)$$

soit d'après (5.18)

$$e_{n+1}=e_n+h_n(f(t_n,y(t_n))-f(t_n,y_n))+\varepsilon_n. \tag{5.19}$$

Pour des raisons de simplicité, nous supposerons ici que la constante de Lipschitz dans (5.8) est uniforme, c'est-à-dire

$$L(M)\leq L \qquad \forall M>0 \tag{5.20}$$

(Souvent dans la pratique, ce n'est pas le cas, mais on montre que sous les seules hypothèses introduites, on peut choisir T assez petit pour que y_n reste uniformément majoré par un certain M. On prend alors $L = L(\max(M, M_0))$ dans la suite.)
De (5.19), (5.8), (5.20) on déduit

$$\|e_{n+1}\| \leq (1 + Lh_n) \|e_n\| + \|\varepsilon_n\|. \tag{5.21}$$

Afin de faciliter la récurrence dans (5.21), on utilise l'inégalité $1 + x \leq e^x$ pour obtenir

$$\|e_{n+1}\| \leq e^{Lh_n} \|e_n\| + \|\varepsilon_n\|.$$

Par récurrence, on obtient ainsi

$$\|e_n\| \leq e^{Lt_n} \|e_0\| + \sum_{i=0}^{n-1} e^{L(t_n - t_{i+1})} \|\varepsilon_i\|. \tag{5.22}$$

Remarque 5.5 Le calcul que nous venons de faire prouve en fait la stabilité de la méthode; en effet, posant

$$z_{n+1} = z_n + h_n f(t_n, z_n) + \varepsilon_n \tag{5.23}$$

au lieu de (5.18) et en posant $\widehat{e}_n = z_n - y_n$ on aurait de même

$$\|\widehat{e}_n\| \leq e^{Lt_n} \|e_0\| + \sum_{i=0}^{n-1} e^{L(t_n - t_{i+1})} \|\varepsilon_i\|. \tag{5.24}$$

et donc

$$\max_{0 \leq n \leq 1} \|\widehat{e}_n\| \leq K \left[\|\widehat{e}_0\| + \sum_{i=0}^{n-1} \|\varepsilon_i\| \right] \tag{5.25}$$

avec $K = e^{LT}$.

Si nous revenons à (5.22), nous voyons que, lorsque h tend vers 0, $\|e_0\| = \left\|y^0 - y^h\right\|$ tend vers 0. Reste à montrer que le dernier terme tend vers 0, c'est-à-dire montrer la consistance de la méthode, puisque ce dernier terme est majoré par $e^{LT} \sum_{i=0}^{N-1} \|\varepsilon_i\|$.
Or

$$\begin{aligned} \varepsilon_n &= y(t_{n+1}) - y(t_n) - h_n f(t_n, y(t_n)) = \int_{t_n}^{t_{n+1}} (y'(t) - y'(t_n))\, dt \\ &= \int_{t_n}^{t_{n+1}} \left[\int_{t_n}^{t} y''(\sigma)\, d\sigma \right] = \int_{t_n}^{t_{n+1}} (t_{n+1} - \sigma)\, y''(\sigma)\, d\sigma. \end{aligned}$$

On retiendra de ceci que

$$\|\varepsilon_n\| \leq h \int_{t_n}^{t_{n+1}} \|y''(\sigma)\|\, d\sigma \leq hM_2 (t_{n+1} - t_n)$$

$$\sum_{i=0}^{n-1} \|\varepsilon_i\| \leq hM_2 t_n. \tag{5.26}$$

Ainsi, ces estimations jointes à (5.22) donnent

$$\|e_n\| \leq e^{Lt_n} [\|e_0\| + hM_2 t_n]$$

et

$$\max_{0 \leq n \leq N} \|e_n\| \leq e^{LT} [\|e_0\| + hM_2 T] \tag{5.27}$$

ce qui prouve que la méthode d'Euler est convergente.

5.2.1.2 Influence des erreurs d'arrondis

Si on veut calculer $y(T)$ avec une précision donnée ε, d'après (5.27), il faut bien sûr d'abord choisir $\|e_0\|$ assez petit : supposons qu'on puisse le choisir nul. En théorie, on peut alors choisir h assez petit pour que $hM_2T < \varepsilon$. Cependant, dans la pratique, il est illusoire de vouloir choisir h "trop" petit, car alors les erreurs d'arrondis prennent le relais de l'erreur de discrétisation : en effet, le calcul

$$y_{n+1} := y_n + h_n f(t_n, y_n)$$

donne lieu à une erreur μ_n sur le calcul de $f(t_n, y_n)$ et à une erreur d'arrondi ρ_n sur le résultat final : l'algorithme du calculateur est donc

$$z_{n+1} = z_n + h_n (f(t_n, z_n) + \mu_n) + \rho_n.$$

D'après les formules (5.23), (5.24), (5.25), on a

$$\|y_N - z_N\| \leq e^{LT} \left(\|y_0 - z_0\| + \sum_{i=0}^{N-1} (h_n \|\mu_n\| + \|\rho_n\|) \right).$$

Supposons $\|\mu_n\| \leq \mu$ et $\|\rho_n\| \leq \rho$. La somme de l'erreur de discrétisation $\|y(T) - y_N\|$ et des erreurs d'arrondis est donc majorée par (remarquer $\sum_{n=0}^{n-1} \|\rho_n\| \leq N\rho = \frac{T\rho}{h}$) :

$$E_N = e^{LT} \left(hM_2T + \|y_0 - z_0\| + T\mu + \frac{T}{h}\rho \right). \tag{5.28}$$

Ceci montre que les erreurs d'arrondis tendent en général vers l'infini lorsque h tend vers 0 ... ce qui est bien normal puisqu'on accumule alors le nombre d'opérations. On peut essayer de voir quel serait le pas "optimal" qui rendrait E_N minimum. Puisque E_N est de la forme

$$\begin{aligned} E_N &= A + Bh + \rho\frac{C}{h} \\ \text{avec } A &= e^{LT}(\|y_0 - z_0\| + T\mu) \\ B &= M_2 T e^{LT} \\ C &= T e^{LT}, \end{aligned}$$

le minimum est atteint pour $h = h^* = \sqrt{\rho\frac{C}{B}}$: c'est-à-dire que, autour de h^* les erreurs d'arrondis s'équilibrent avec celles dues à la méthode. Un bon choix nécessitera bien sûr $h \geq h^*$: il est inutile d'augmenter le coût de calcul pour rien.

Remarque 5.6 Les calculs ci-dessus sont faits de façon "pessimiste" : souvent les erreurs d'arrondis se compensent dans une certaine mesure (mais c'est aléatoire).

Il est intéressant de noter dans l'expression de E_N comment les erreurs d'arrondi sur la donnée initiale $\|y_0 - z_0\|$ se "propagent" : leur impact sur l'erreur finale est majoré par $e^{LT}\|y_0 - z_0\|$ ce qui est raisonnable ... si e^{LT} n'est pas trop grand : nous allons voir sur des exemples ce que ceci signifie numériquement :

Exemple 1 :

$$\begin{cases} y'(t) = 3y(t) - 3t & t \in [0,6] \\ y(0) = \frac{1}{3} \end{cases} \tag{5.29}$$

La solution est $y(t) = \frac{1}{3} + t$. Ce problème est donc tout à fait inoffensif. Cependant, ici $L = 3$ donc pour $T = 6$,

$$e^{LT} = e^{18} \simeq 0.6510^8.$$

Ainsi, si on considère la solution $z(t)$du même problème avec la donnée initiale $z(0) = \frac{1}{3} + \varepsilon (= 0.33333333$ par exemple) ; par différence

$$\begin{cases} (z-y)' = 3(z-y) \\ (z-y)(0) = \varepsilon \end{cases} \quad \text{et donc } (z-y)(t) = \varepsilon e^{3t}.$$

Ainsi $z(T) = y(T) + \varepsilon e^{18} \simeq y(T) + 0.65\varepsilon 10^8$. En conséquence, si on travaille avec un calculateur travaillant avec une présence relative de 10^{-8}, il sera pratiquement impossible d'approcher convenablement $y(6)$ ... et ceci indépendamment du choix de la méthode numérique. On dit que le problème ci-dessus est mal conditionné ou même ici, numériquement mal posé.

Exemple 2 :

$$\begin{cases} y'(t) = -150\, y(t) + 50 \\ y(0) = \frac{1}{3} \end{cases} . \tag{5.30}$$

La solution est $y(t) \equiv \frac{1}{3}$. La méthode d'Euler conduit à

$$y_{n+1} = y_n + h_n (-150 y_n + 50).$$

Soit encore

$$y_{n+1} - \frac{1}{3} = (1 - 150 h_n) \left(y_n - \frac{1}{3} \right).$$

Si $h_n = h = \frac{1}{50}$, on a donc

$$y_n - \frac{1}{3} = (1 - 150h)^n \left(y_0 - \frac{1}{3} \right) = (-2)^n \left(y_0 - \frac{1}{3} \right). \tag{5.31}$$

En particulier,

$$y_{50} - y(1) = (-2)^{50} \left(y_0 - \frac{1}{3} \right)!$$

Sachant que $2^{50} \simeq 10^{15}$, ceci montre que le pas choisi est trop grand.
Ici, la méthode d'Euler n'est pas un algorithme numériquement stable : bien qu'elle satisfasse à la condition de stabilité théorique (5.17), le coefficient K est trop grand. Cependant, on peut montrer que ce problème est bien conditionné au sens qu'une petite variation de y_0 induit une petite variation de $y(t)$ puisque, grâce au signe $-$ devant $150y(t)$, on montre aisément :

$$|y(t) - \widehat{y}(t)| \leq |y_0 - \widehat{y}_0|.$$

La relation (5.31) met en évidence un autre phénomène : lorsque n grandit, $y_n - \frac{1}{3}$ devient en module de plus en plus grand et est alternativement positif et négatif : au bout d'un temps très court, les calculs ne sont plus significatifs. Analysons ceci de façon un peu plus systématique.

Exemple 3 :

$$\begin{cases} y'(t) = -\lambda y(t) \quad \lambda > 0 \\ y(0) = y_0 \end{cases} . \tag{5.32}$$

Ce problème continu est très stable par rapport à y_0 puisque $\mathrm{y}(t) = y_0 e^{-\lambda t}$ et donc

$$|y(t) - \widehat{y}(t)| \leq e^{-\lambda t} |y_0 - \widehat{y}_0|.$$

La méthode d'Euler appliquée à (5.32) avec pas constant donne

$$y_{n+1} = y_n - \lambda h y_n = (1 - \lambda h) y_n$$

et donc

$$y_n = (1 - \lambda h)^n y_0.$$

Le pas h étant fixé, on voit que, bien que la solution exacte soit bornée, la solution calculée y_n aura un module de plus en plus grand si $|1 - \lambda h| > 1$, d'où un phénomène d'instabilité. La condition de stabilité s'écrit donc

$$\lambda h < 2 \tag{5.33}$$

Ainsi, il sera nécessaire de prendre h d'autant plus petit que λ est grand. Cette restriction apparaît également très souvent dans les systèmes différentiels provenant de la discrétisation d'équations aux dérivées partielles de type parabolique (condition CFL).

5.2.2 Méthode d'Euler implicite

Afin de remédier au problème d'instabilité soulevé ci-dessous, on a souvent recours à une méthode de type implicite comme la suivante

$$\begin{cases} y_{n+1} = y_n + h_n f\left(t_{n+1}, y_{n+1}\right) \\ y_0 = y^h \end{cases} \tag{5.34}$$

qui provient de l'intégration de $\int_{t_n}^{t_{n+1}} f\left(t, y\left(t\right)\right) dt$ par une méthode des rectangles à droite. On voit que la relation (5.34) définit y_{n+1} de façon implicite et qu'il faut donc résoudre un problème pour calculer y_{n+1}. Le coût est évidemment plus lourd que dans la méthode d'Euler explicite vue précédemment, mais en contrepartie la stabilité est grandement améliorée comme on va le voir en reprenant l'exemple (5.32). On a donc, avec un pas constant :

$$y_{n+1} = y_n - \lambda h y_{n+1}$$

soit

$$y_{n+1} = \frac{y_n}{1 + \lambda h},\ y_n = \frac{y_0}{(1 + \lambda h)^n}.$$

En particulier, on a toujours (quel que soit le choix du pas h)

$$\left|y_n\right| \leq \left|y_0\right|.$$

Par ailleurs, on montre que cette méthode converge avec la même vitesse que la précédente.

5.3 Les méthodes de Runge-Kutta

Une méthode à un pas s'écrit de façon générale

$$y_{n+1} = y_n + h_n \phi\left(t_n, y_n, h_n\right). \tag{5.35}$$

Ainsi, y_{n+1} est calculé à partir de y_n par l'intermédiaire de la fonction ϕ. Cette méthode peut être explicite ou implicite. Ainsi, pour la méthode d'Euler explicite, on a

$$\phi\left(t, y, h\right) = f\left(t, y\right)$$

tandis que pour la méthode d'Euler implicite $\phi\left(t, y, h\right)$ est défini de façon implicite; $\phi\left(t, y, h\right)$ est la solution de l'équation

$$Y = y_n + h_n f\left(t_{n+1}, Y\right).$$

Définition 5.4 *On dit que la méthode (5.35) est d'ordre p si l'erreur de consistance vérifie*

$$\sum_{N=0}^{N-1} \|y(t_{n+1}) - y(t_n) - h_n \phi(t_n, y(t_n), h_n)\| \leq K h^p \tag{5.36}$$

où $h = \max_{0 \leq n < N} h_n$ *et ce, pour toute solution p fois continument dérivable de* $y'(t) = f(t, y(t))$.

Exemple : Nous avons vu que la méthode d'Euler explicite est d'ordre 1 (cf. (5.26)). Il en est de même pour la méthode d'Euler implicite. Nous allons le montrer ici en considérant plus généralement :
θ-méthode :

$$y_{n+1} = y_n + h_n [\theta f(t_{n+1}, y_{n+1}) + (1-\theta) f(t_n, y_n)] \tag{5.37}$$

où $\theta \in [0,1]$. Pour $\theta = 0$, c'est explicite ; pour $\theta = 1$ on retrouve la méthode d'Euler implicite (5.34).
Analysons l'erreur de consistance :

$$\varepsilon_n = y(t_{n+1}) - y(t_n) - h_n [\theta f(t_{n+1}, y(t_{n+1})) + (1-\theta) f(t_n, y_n)].$$

Pour simplifier, nous supposerons $N = 1$; les techniques et résultats sont identiques en dimension supérieure. D'après la formule de Taylor :

$$y'(t_{n+1}) = y'(t_n) + h_n y''(t_n) + \frac{h_n^2}{2} y^{(3)}(c_n)$$

où $c_n \in [t_n, t_{n+1}]$. De même

$$y(t_{n+1}) = y(t_n) + h_n y'(t_n) + \frac{h_n^2}{2} y''(t_n) + \frac{h_n^3}{6} y^{(3)}(\widehat{c}_n).$$

On en déduit

$$\begin{aligned} \varepsilon_n &= h_n y'(t_n) + \frac{h_n^2}{2} y''(t_n) + \frac{h_n^3}{6} y^{(3)}(\widehat{c}_n) \\ &\quad - h_n \left[\theta \left(y'(t_n) + h_n y''(t_n) + \frac{h_n^2}{2} y^{(3)}(c_n) \right) + (1-\theta) y'(t_n) \right] \end{aligned}$$

soit

$$\varepsilon_n = h_n^2 y''(t_n) \left[\frac{1}{2} - \theta \right] + h_n^3 \left(\frac{1}{6} y^{(3)}(\widehat{c}_n) - \frac{\theta}{2} y^{(3)}(c_n) \right).$$

Si $\theta \neq \frac{1}{2}$ (par exemple $\theta = 0$ ou $\theta = 1$), l'erreur locale sera en h^2 et l'erreur totale de consistance sera en h et la méthode d'ordre 1. Par contre, si $\theta = \frac{1}{2}$, on obtient une méthode d'ordre 2. Ce choix est souvent appelé méthode de Crank-Nicholson.
Méthode de Crank-Nicholson :

$$y_{n+1} = y_n + h_n \left[\frac{1}{2} f(t_{n+1}, y_{n+1}) + \frac{1}{2} f(t_n, y_n) \right] \tag{5.38}$$

Analysons la stabilité asymptotique sur le cas particulier

$$y'(t) = -\lambda y(t).$$

On a alors

$$y_{n+1} = y_n + h [-\lambda \theta y_{n+1} - \lambda (1-\theta) y_n]$$

$$y_{n+1} = \frac{1-\lambda h\,(1-\theta)}{1+\lambda h\theta} y_n$$

$$y_n = \left(\frac{1-\lambda h\,(1-\theta)}{1+\lambda h\theta}\right)^n y_0.$$

L'approximation y_n restera bornée pour n grand si

$$-1 < \frac{1-\lambda h\,(1-\theta)}{1+\lambda h\theta} < 1.$$

Ceci équivaut à

$$0 < \frac{\lambda h}{1+\lambda h\theta} \leq 2 \text{ ou } 0 < \lambda h \leq 2 + \lambda h.2\theta.$$

D'où la condition de stabilité asymptotique

$$\begin{cases} \cdot \text{ si } \theta \geq \frac{1}{2} : \text{méthode stable } \forall h > 0 \\ \cdot \text{ si } \theta < \frac{1}{2} : \text{méthode stable si } \lambda h < \frac{2}{1-2\theta} \end{cases} \tag{5.39}$$

Conclusion : La méthode de Crank-Nicholson est d'ordre 2 et stable (au sens ci-dessus) pour tout choix de h. Ceci explique qu'elle soit souvent retenue.
Dans la pratique, même une méthode d'ordre 2 se révèle assez souvent insuffisante car nécessitant trop de pas de temps pour atteindre une précision donnée. Il est alors nécessaire d'utiliser une méthode d'ordre supérieur : les plus courantes sont celles de Runge-Kutta qui reposent sur des méthodes d'intégration numérique d'ordre supérieur pour

$$\int_{t_n}^{t_{n+1}} f\,(t, y\,(t))\,dt.$$

Décrivons-en le principe : $t_{n,j} = t_n + h_n c_j,\ j = 1, ..., q$ étant des points de $[t_n, t_{n+1}]$ $(0 \leq c_j \leq 1)$, le remplacement de $f\,(t, y\,(t))$ par un polynôme d'interpolation aux points $t_{n,j}$ conduit à une formule d'intégration numérique du type

$$\int_{t_n}^{t_{n+1}} f\,(t, y\,(t))\,dt \simeq h_n \left(\sum_{j=1}^{q} b_j f\,(t_{n,j}, y\,(t_{n,j}))\right).$$

Les méthodes de Runge-Kutta consistent à remplacer ces évaluations approchées par des égalités soit

$$y_{n+1} = y_n + h_n \left[\sum_{j=1}^{q} b_j f\,(t_{n,j}, y_{n+j})\right], \tag{5.40}$$

les valeurs $y_{n,j}$ étant elles-mêmes évaluées à l'aide de formules d'intégration numérique utilisant les mêmes points $t_{n,j}$:

$$\left\{ y_{n,j} = y_n + h_n \left[\sum_{j=1}^{q} a_{ij} f\,(t_{n,j}, y_{n,j})\right], j = 1, ..., q \right\}. \tag{5.41}$$

Remarque 5.7 – Les formules (5.41) définissent les valeurs $y_{n,j}$ de façon explicite si la matrice $[a_{ij}]$ est strictement triangulaire inférieure, sinon elles sont obtenues de façon implicite.
– La méthode d'Euler et la θ-méthode sont des cas particuliers. Citons quelques méthodes d'ordre supérieur souvent utilisées.

Runge-Kutta d'ordre 2

$$\begin{cases} y_{n+1} = y_n + \frac{1}{2}\left(k_1^n + k_2^n\right) \\ k_1^n = h_n f\left(t_n, y_n\right) \\ k_2^n = h_n f\left(t_n + h_n, y_n + k_1^n\right) \end{cases} \tag{5.42}$$

ou

$$\begin{cases} y_{n+1} = y_n + k_2^n \\ k_1^n = h_n f\left(t_n, y_n\right) \\ k_2^n = h_n f\left(t_n + \frac{h_n}{2}, y_n + \frac{k_1^n}{2}\right) \end{cases} \tag{5.43}$$

et la fameuse méthode d'ordre 4 :

Runge-Kutta d'ordre 4

$$\begin{cases} y_{n+1} = y_n + \frac{1}{6}\left(k_1^n + 2k_2^n + 2k_3^n + k_4^n\right) \\ k_1^n = h_n f\left(t_n, y_n\right) \\ k_2^n = h_n f\left(t_n + \frac{h_n}{2}, y_n + \frac{k_1^n}{2}\right) \\ k_3^n = h_n f\left(t_n + \frac{h_n}{2}, y_n + \frac{k_2^n}{2}\right) \\ k_4^n = h_n f\left(t_{n+1}, y_n + k_3^n\right) \end{cases} . \tag{5.44}$$

On détermine l'ordre de ces méthodes à l'aide de la formule de Taylor. Considérons par exemple

$$\begin{cases} y_{n+1} = y_n + \left(ak_1^n + bk_2^n\right) \\ k_1^n = h_n f\left(t_n, y_n\right) \\ k_2^n = h_n f\left(t_n + \alpha h_n, y_n + \beta k_1^n\right) \end{cases} . \tag{5.45}$$

Il s'agit d'estimer

$$\varepsilon_{n+1} = y\left(t_{n+1}\right) - y\left(t_n\right) + h_n \phi\left(t_n, y\left(t_n\right), h_n\right)$$

où

$$\phi\left(t_n, y\left(t_n\right), h_n\right) = a\lambda + b\mu$$

avec

$$\begin{aligned} \lambda &= f\left(t_n, y\left(t_n\right)\right) \\ \mu &= f\left(t_n + \alpha h_n, y\left(t_n\right) + \beta h_n \lambda\right). \end{aligned}$$

Or

$$f\left(t_n + \alpha h_n, y\left(t_n\right) + \beta h_n \lambda\right) = f\left(t_n, y\left(t_n\right)\right) + \alpha h_n f_t\left(t_n, y\left(t_n\right)\right) + \beta h_n \lambda f_y\left(t_n, y\left(t_n\right)\right) + O\left(h_n^2\right).$$

$$y\left(t_{n+1}\right) - y\left(t_n\right) = h_n y'\left(t_n\right) + \frac{h_n^2}{2} y''\left(t_n\right) + O\left(h_n^3\right).$$

Puisque

$$y''\left(t_n\right) = f_t\left(t_n, y\left(t_n\right)\right) + f_y\left(t_n, y\left(t_n\right)\right).\lambda,$$

on en déduit

$$\varepsilon_{n+1} = h_n \lambda\left(1 - (a+b)\right) + h_n^2\left(\frac{1}{2}\left(f_t + \lambda f_y\right) - b\left(\alpha f_t + \beta \lambda f_y\right)\right) + O\left(h_n^3\right).$$

L'erreur locale est d'ordre 3 (et la méthode d'ordre 2) si

$$\begin{cases} a+b=1 \\ \frac{1}{2}=\alpha b=b\beta \end{cases}.$$

C'est le cas dans les algorithmes (5.43) et (5.44).

Remarque 5.8 On démontre, sous des hypothèses de régularité sur f, que les méthodes de Runge-Kutta sont stables au sens de (5.17). Etant stables et consistantes, elles sont convergentes suivant un principe général déjà mentionné pour les méthodes à un pas.

5.4 Contrôle du pas

Dans la description des algorithmes précédents, nous avons supposé le pas h_n variable. Dans la pratique, il est naturel de chercher à utiliser un pas constant $h_n = h$ et on le fait souvent. Cependant, il est difficile d'évaluer la taille optimale du pas h : il ne doit pas être petit, sinon le coût de calcul est important et les erreurs d'arrondis s'accumulent. Il ne doit pas être trop grand pour que les calculs conservent une précision suffisante. Toute la difficulté consiste bien sûr à choisir le pas pour que la valeur $y(T)$ puisse être connue avec une précision préfixée.

Les algorithmes les plus performants de ce point de vue réalisent un contrôle du pas à chaque étape, dans le but

- de rester dans une marge de précision voulue
- mais aussi d'adapter le pas aux irrégularités éventuelles de la solution : sur des intervalles où la solution varie peu, on pourra "avancer à grands pas". Quand les variations deviennent plus raides, on diminue le pas.

Un contrôle "optimal" du pas dans un problème donné est toujours difficile à déterminer et, souvent, coûteux à mettre en oeuvre. Nous nous contenterons de donner ici quelques idées simples pouvant être utilisées en première approximation.

Etant donné une méthode à un pas de type (5.35), il s'agit à chaque étape de choisir un pas h tel que l'erreur $y(t_n+h) - y_n - h\phi(t_n, y_n, h)$ reste en deçà d'une certaine tolérance. Supposons avoir agi pour le mieux aux étapes précédentes : on ne peut plus alors agir sur y_n et (à moins d'espérer des erreurs de compensation qui sont de toute façon imprévisibles en général), on peut raisonner en supposant que y_n est connue de façon exacte et contrôler alors l'erreur locale de consistance :

$$\varepsilon_n(h) = y(t_n+h) - (y(t_n) + h_n\phi(t_n, y(t_n), h_n)).$$

(Un raisonnement fait de façon globale conduit en fait à la même conclusion : contrôler l'erreur locale de consistance). Ainsi, on pourra s'imposer que

$$|\varepsilon_n(h)| \leq \alpha h \tag{5.46}$$

où α est une certaine tolérance. L'erreur globale de discrétisation sera alors majorée par un facteur de α (cf. (5.22)) (reste bien sûr l'évaluation de ce facteur de type e^{LT}, difficulté toujours présente).

Tout le problème est d'évaluer l'erreur ε_n afin d'en déduire une valeur optimale de h satisfaisant à (5.46). Les diverses méthodes de contrôle du pas diffèrent essentiellement sur la technique d'évaluation de ε_n.

L'une d'elles consiste, étant donné un pas h provenant par exemple de l'étape précédente à calculer 2 valeurs approchées de $y(t_n+h)$, l'une $\overline{y}(t_n+h)$ à l'aide d'une itération de pas h, l'autre $\widehat{y}(t_n+h)$ avec deux itérations de pas $\frac{h}{2}$: l'erreur ε_n est alors évaluée à

l'aide d'un procédé d'extrapolation à la limite de Richardson. En effet, si la méthode est d'ordre p ou plus exactement si l'erreur locale est d'ordre $p+1$, on a en supposant $y_n = y(t_n)$ (cf. remarque plus loin) :

$$\overline{y}(t_n + h) = y(t_n + h) - C_n h^{p+1} + O\left(h^{p+2}\right) \tag{5.47}$$

où en général $C_n \neq 0$ et aussi

$$\widehat{y}(t_n + h) = y(t_n + h) - C_n \left(\frac{h}{2}\right)^{p+1} + O\left(h^{p+2}\right). \tag{5.48}$$

On le vérifie aisément pour la méthode d'Euler explicite : dans ce cas $p = 1$ et $C_n = \frac{1}{2}y''(t_n)$. Négligeant les termes d'ordre $p+2$ dans (5.47), on a

$$\varepsilon_n(h) \simeq |C_n| h^{p+1}. \tag{5.49}$$

Mais par différence entre (5.48) et (5.47), on obtient aussi :

$$\widehat{y}(t_n + h) - \overline{y}(t_n + h) \simeq C_n \left(\frac{h}{2}\right)^{p+1} \left(2^{p+1} - 1\right)$$

d'où une évaluation de l'erreur $\varepsilon_n(h)$ en fonction des valeurs calculées $\widehat{y}(t_n + h)$ et $\overline{y}(t_n + h)$ soit

$$\left|\varepsilon_n\left(\frac{h}{2}\right)\right| \simeq \frac{|\widehat{y}(t_n + h) - \overline{y}(t_n + h)|}{2^{p+1} - 1} \text{ (on note } D_n(h) \text{ cette expression).}$$

D'où une première méthode de contrôle du pas : on se donne une tolérance $\alpha > 0$;

$$\left\{\begin{array}{l} \text{étant donné un pas de départ } h \\ \cdot \text{ on calcule } \overline{y}(t_n + h) \text{ en une itération de pas } h \\ \cdot \text{ on calcule } \widehat{y}(t_n + h) \text{ en deux itérations de pas } h/2 \\ \left|\begin{array}{l} \cdot \text{si } D_n(h) > \alpha h, \text{ on recommence avec } h \text{ remplacé par } h/2 \\ \cdot \text{si } D_n(h) < \alpha h, \text{ on accepte } \widehat{y}(t_n + h) \text{ comme valeur approchée de } y(t_n + h) \\ \quad \text{et on continue : } \left\{\begin{array}{l} -\text{avec le même pas } h \text{ si } \alpha' h < D_n(h) < \alpha h, \text{ où } \alpha' \text{ est une} \\ \text{tolérance inférieure (par exemple } \alpha' = \alpha/2^{p+1}) \\ -\text{avec le pas } 2h \text{ si } D_n(h) < \alpha' h. \end{array}\right. \end{array}\right. \end{array}\right. \tag{5.50}$$

Remarque 5.9 L'introduction de α' permet d'allonger le pas pour ne pas effectuer trop d'itérations inutiles.

Un contrôle plus sophistiqué consiste, au lieu de diviser et multiplier par 2, à prendre comme nouveau pas le pas "optimal" h réalisant l'égalité

$$\alpha \overline{h} = \varepsilon_n\left(\overline{h}\right) \simeq C_n \overline{h}^{p+1} \simeq 2^{p+1} \left(\frac{\overline{h}}{h}\right)^{p+1} D_n(h)$$

soit

$$\overline{h} = \frac{h}{2} \left(\frac{\alpha h}{2 D_n(h)}\right)^{\frac{1}{p}}.$$

On contrôle alors comme suit : partant du pas h de l'étape précédente
- si $h \simeq 2\overline{h}$, on continue avec $\widehat{y}(t_n + h)$ et le même pas h
- si $h >> 2\overline{h}$, on recommence avec h remplacé par $2\overline{h}$
- si $h << 2\overline{h}$, on continue avec $\widehat{y}(t_n + h)$ et le pas $h^* = \min\left(2\overline{h}, h_0\right)$ où h_0 est une limite supérieure de sécurité pour le pas.

Remarque 5.10 – Dans le raisonnement ci-dessus, nous avons supposé $y_n = y(t_n)$ ce qui bien sûr n'est pas réalisé en pratique. On peut cependant justifier les conclusions ci-dessus. On pourra par exemple les vérifier directement sur la méthode d'Euler.
– S'il s'avère nécessaire de prendre des h trop petits ou si on se heurte à plusieurs échecs successifs, la situation est anormale et il faut procéder à des modifications (ordre insuffisant par rapport à la précision demandée, etc...). On veillera à ce que l'ordre de grandeur de la précision requise soit compatible avec l'ordre de la méthode et la précision de la machine.

5.5 Méthodes à pas multiples

Les méthodes à un pas utilisent seulement la valeur approchée y_n de $y(t_n)$ pour calculer une valeur approchée y_{n+1} de $y(t_{n+1})$. Les méthodes à pas multiples utilisent aussi l'information obtenue aux temps précédents $t_{n-1}, t_{n-2}, ..., t_{n-r}$.
Nous décrirons ici les méthodes d'Adams qui consistent à remplacer $f(t, y(t))$ par un polynôme d'interpolation aux points $t_{n-r}, t_{n-r+1}, ..., t_{n-1}, t_n, (t_{n+1})$, dans le calcul de

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\, dt.$$

Si P_n est le polynôme en question, les valeurs approchées y_n seront obtenues par l'équation approchée

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} P_n(t)\, dt. \tag{5.51}$$

Les formules seront implicites ou explicites selon que t_{n+1} est l'un des points d'interpolation ou non.

5.5.1 Méthodes d'Adams-Bashforth à $r+1$ pas

On suppose connues des valeurs approchées y_n de $y(t_n)$ et $f_n, f_{n-1}, ..., f_{n-r}$ de $f(t, y(t))$ respectivement aux points $t_n, t_{n-1}, ..., t_{n-r}$. Le polynôme P_n est choisi comme le polynôme de degré inférieur ou égal à r tel que

$$P_n(t_{n-i}) = f_{n-i} \qquad \forall i = 0, ..., r.$$

Pour $t \in [t_n, t_{n+1}]$, $t_{n+1} = t_n + h$. Posons $t = t_n + sh$ où $s \in [0, 1]$. Alors comme il a été établi dans le chapitre d'interpolation (cf. 3.9),

$$P_n(t_n + sh) = \sum_{i=0}^{r} \left(\begin{array}{c} s+i-1 \\ i \end{array} \right) \Delta^{-i} f_n. \tag{5.52}$$

Ceci est la formule de Newton rétrograde obtenue à l'aide des différences finies rétrograde

$$\Delta^{-i} f_k = \left\{ \begin{array}{ll} f_k & \text{si } i = 0 \\ \Delta^{-i-1} f_k - \Delta^{-i-1} f_{k-1} & \text{si } i \geq 1 \end{array} \right.$$

et $\left(\begin{array}{c} s \\ k \end{array} \right)$ est le coefficient du binôme généralisé aux valeurs non entières soit

$$\left(\begin{array}{c} s \\ k \end{array} \right) = \frac{s(s-1)...(s-k+1)}{1.2...k}.$$

On obtient y_{n+1} à l'aide de (5.51) et (5.52), soit, en effectuant le changement de variables $t = t_n + sh$:

$$y_{n+1} = y_n + \sum_{i=0}^{r} \Delta^{-i} f_n . \int_0^1 \begin{pmatrix} s+i-1 \\ i \end{pmatrix} h\, ds. \tag{5.53}$$

Notons

$$\gamma_i = \int_0^1 \begin{pmatrix} s+i-1 \\ i \end{pmatrix} ds = \int_0^1 \frac{s\,(s+1)\,...\,(s+i-1)}{i!} ds.$$

On montre facilement que les γ_i vérifient la relation

$$\gamma_0 = 1,\ 1 = \frac{\gamma_0}{i+1} + \frac{\gamma_1}{i} + ... + \frac{\gamma_{i-1}}{2} + \gamma_i$$

ce qui permet de les calculer par récurrence. Il est important de noter qu'ils ne dépendent pas de r, ce qui est utile lorsqu'on veut faire varier l'ordre r dans un même calcul. On obtient ainsi

$$\gamma_0 = 1,\ \gamma_1 = \frac{1}{2},\ \gamma_2 = \frac{5}{12},\ \gamma_3 = \frac{3}{8},\ \gamma_4 = \frac{251}{720},\ \gamma_5 = \frac{95}{288}.$$

Dans la pratique, on préfère expliciter la relation (5.53) directement en fonction des valeurs f_{n-i} d'où une formule du type

$$y_{n+1} = y_n + h \sum_{i=0}^{r} b_{i,r} f_{n-i}.$$

A l'aide de la définition des différences divisées, on vérifie

$$b_{r,r} = (-1)^r \gamma_r,\ b_{i,r} = b_{i,r-1} + (-1)^i \begin{pmatrix} r \\ i \end{pmatrix} \gamma_r,\ 0 \le i \le r.$$

On obtient ainsi le tableau suivant :

	$b_{0,r}$	$b_{1,r}$	$b_{2,r}$	$b_{3,r}$	$b_{4,r}$	$b_{5,r}$	$b_{6,r}$	γ_r
$r=0$	1							1
$r=1$	$\frac{3}{2}$	$-\frac{1}{2}$						$\frac{1}{2}$
$r=2$	$\frac{23}{12}$	$-\frac{4}{3}$	$\frac{5}{12}$					$\frac{5}{12}$
$r=3$	$\frac{55}{24}$	$-\frac{59}{24}$	$\frac{37}{24}$	$-\frac{3}{8}$				$\frac{3}{8}$
$r=4$	$\frac{1901}{720}$	$-\frac{1387}{360}$	$\frac{109}{30}$	$-\frac{637}{360}$	$\frac{251}{720}$			$\frac{251}{720}$
$r=5$	$\frac{4277}{1440}$	$-\frac{7923}{1440}$	$\frac{4991}{720}$	$-\frac{3649}{720}$	$\frac{959}{480}$	$-\frac{95}{288}$		$\frac{95}{288}$
$r=6$	$\frac{199441}{60840}$	$-\frac{18817}{2520}$	$\frac{238783}{20160}$	$-\frac{10979}{945}$	$\frac{139313}{20160}$	$-\frac{5783}{2520}$	$\frac{19807}{60840}$	$\frac{19807}{60840}$

On utilise la méthode d'Adams-Bashforth à 4 pas, le plus souvent, soit :

$$y_{n+1} = y_n + \frac{h}{24}\left(55 f_n - 59 f_{n-1} + 37 f_{n-2} - 9 f_{n-3}\right). \tag{5.54}$$

On montre que cette méthode est d'ordre 4 (ie. l'erreur globale de consistance est en h^4) et qu'elle est stable (en un sens analogue à (5.17)) et ce sous des hypothèses naturelles de régularité de f. Cependant, les constantes de stabilité sont souvent grandes ce qui fait qu'on constate souvent une instabilité numérique analogue à celle que nous avons souligné comme étant fréquente dans la méthode d'Euler explicite (cf. paragraphe 5.2.1). Comme toujours, pour pallier cet inconvénient, on préfère utiliser des méthodes implicites.

5.5.2 Méthodes d'Adams-Moulton à $r+1$ pas

On interpole la fonction $f(t, y(t))$ aux points $t_{n+1},\ t_n,\ ...,\ t_{n-r}$ par le polynôme Q_n de degré inférieur ou égal à $r+1$ tel que

$$\begin{cases} Q_n(t_{n-i}) = f_{n-i} \ \ i = 0, 1, ..., r \\ Q_n(t_{n+1}) = f_{n+1} \ \ \text{(valeur encore inconnue).} \end{cases}$$

D'après la formule de Newton régressive, on a

$$Q_n(t_{n+1} + sh) = \sum_{i=0}^{r+1} \begin{pmatrix} s+i-1 \\ i \end{pmatrix} \Delta^{-i} f_{n+1}.$$

On obtient alors y_{n+1} à l'aide de

$$y_{n+1} = y_n + h \sum_{i=0}^{r+1} \gamma_i^* \Delta^{-i} f_{n+1} \tag{5.55}$$

où

$$\gamma_i^* = \int_{-1}^{0} \frac{s(s+1)...(s+i-1)}{i!} ds,\ i \geq 1,\ \gamma_0^* = 1.$$

On vérifie

$$\gamma_i^* = \gamma_i - \gamma_{i-1},\ i \geq 1.$$

Comme précédemment, on préfère écrire (5.55) sous la forme

$$y_{n+1} = y_n + h \sum_{i=-1}^{r} b_{i,r}^* f_{n-i}$$

où, comme on le vérifie aisément, les $b_{i,r}^*$ satisfont à

$$b_{r,r}^* = (-1)^{r+1} \gamma_{r+1}^*,\ b_{i,r}^* = b_{i,r-1}^* + (-1)^{i+1} \begin{pmatrix} r+1 \\ i+1 \end{pmatrix} \gamma_{r+1}^*.$$

On obtient le tableau

	$b_{-1,r}^*$	$b_{0,r}^*$	$b_{1,r}^*$	$b_{2,r}^*$	$b_{3,r}^*$	$b_{4,r}^*$	$b_{5,r}^*$	$b_{6,r}^*$	$b_{7,r}^*$
$r=0$	$\frac{1}{2}$	$\frac{1}{2}$							1
$r=1$	$\frac{5}{12}$	$\frac{2}{3}$	$-\frac{1}{12}$						$-\frac{1}{2}$
$r=2$	$\frac{9}{24}$	$\frac{19}{24}$	$-\frac{5}{24}$	$\frac{1}{24}$					$-\frac{1}{12}$
$r=3$	$\frac{251}{720}$	$\frac{323}{360}$	$-\frac{11}{30}$	$\frac{53}{360}$	$-\frac{19}{720}$				$-\frac{1}{24}$
$r=4$	$\frac{95}{288}$	$\frac{1427}{1440}$	$-\frac{133}{240}$	$\frac{241}{720}$	$-\frac{173}{1440}$	$\frac{3}{160}$			$-\frac{19}{720}$
$r=5$	$\frac{19087}{60480}$	$\frac{2713}{2520}$	$-\frac{15487}{20160}$	$\frac{586}{945}$	$-\frac{6737}{20160}$	$\frac{263}{2520}$	$-\frac{863}{60480}$		$-\frac{3}{160}$
$r=6$	$\frac{36799}{120960}$	$\frac{139849}{120960}$	$-\frac{121797}{120960}$	$\frac{123133}{120960}$	$-\frac{88545}{120960}$	$\frac{41499}{120960}$	$-\frac{11351}{120960}$	$\frac{275}{24192}$	$-\frac{863}{60480}$

on utilise beaucoup la formule d'Adams-Moulton à 3 pas soit

$$y_{n+1} = y_n + \frac{h}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}). \tag{5.56}$$

On montre, sous des hypothèses de régularité, que cette méthode est d'ordre 4 et stable. Les coefficients de stabilité sont bien meilleurs que pour la formule explicite d'ordre 4 (5.54). Bien sûr, il faut "payer le prix" car (5.56) définit y_{n+1} de façon implicite puisque $f_{n+1} = f(t_{n+1}, y_{n+1})$. Il faut donc résoudre un système non linéaire. On peut utiliser les méthodes générales de résolution de tels systèmes ou l'idée générale que nous développons dans le paragraphe suivant.

5.5.3 Méthode de prédicteur-correcteur

Pour résoudre (5.56) on peut utiliser une méthode d'approximations successives (ou de point fixe) consistant à construire la suite $y^0, y^1, y^2, ..., y^p$ définie par

$$\begin{cases} y^{p+1} = y_n + \frac{h}{24}\left(9f\left(t_{n+1}, y^p\right) + 19f_n - 5f_{n-1} + f_{n-2}\right) \\ y^0 \text{ à choisir.} \end{cases} \tag{5.57}$$

On peut mener la suite jusqu'à convergence (en général y^p converge vers y_{n+1} quand p tend vers l'infini). Le plus souvent, on se contente de quelques itérations, voire 1 ou 2. D'autre part, la valeur initiale y^0 est souvent obtenue à l'aide d'un pas d'une méthode explicite de même ordre : il s'agit alors d'une méthode de prédicteur-correcteur : l'évaluation de y^0 correspond à une prédiction ; on corrige ensuite cette valeur à l'aide d'une ou deux itérations de (5.57). Ainsi, le schéma suivant est souvent utilisé :

$$\begin{cases} \text{Prédicteur : Formule d'Adams-Bashforth d'ordre 4} \\ y^0 = y_n + \frac{h}{24}\left(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}\right) \\ \text{Correcteur : une ou deux itérations de la méthode d'Adams-Moulton d'ordre 4} \\ y^{p+1} = y_n + \frac{h}{24}\left(9f\left(t_{n+1}, y^p\right) + 19f_n - 5f_{n-1} + f_{n-2}\right),\ p = 0, 1. \end{cases} \tag{5.58}$$

On montre que la méthode (5.58) est aussi d'ordre 4 ; sa stabilité est nettement meilleure que celle d'Adams-Bashforth ; d'autre part, la résolution du système non linéaire de la formule d'Adams-Moulton est faite de façon "explicite".

Remarque 5.11 cette technique de prédicteur-correcteur peut-être bien sûr utilisée dans de nombreuses autres situations. Ainsi, sans aller jusqu'à l'ordre 4, on peut l'appliquer aux méthodes d'Euler et Crank-Nicholson :

$$\begin{cases} \text{prédicteur} : y^0 = y_n + hf_n \\ \text{correcteur} : y^{p+1} = y_n + \frac{h}{2}\left(f\left(t_{n+1}, y^p\right) + f_n\right),\ p = 0, 1. \end{cases} \tag{5.59}$$

Il existe d'autres prédicteurs-correcteurs multipas : citons les formules de Milne :

$$\begin{cases} \text{prédicteur} : y^0 = y_{n-3} + \frac{4h}{3}\left(2f_n - f_{n-1} + 2f_{n-2}\right) \\ \text{correcteur} : y^{p+1} = y_{n-1} + \frac{h}{3}\left(f\left(t_{n+1}, y^p\right) + 4f_n + f_{n-1}\right). \end{cases} \tag{5.60}$$

Cette méthode correspond à des intégrations de t_{n-r} à t_{n+1} au lieu de t_n à t_{n+1} de l'équation différentielle. Ici, l'intégration est faite avec une méthode de Simpson. On a une méthode d'ordre 4, mais généralement moins stable que celle d'Adams.

5.6 Comparaison des méthodes

Les méthodes les plus couramment utilisées sont celles de Runge-Kutta d'ordre 4 et les méthodes de prédicteur-correcteur d'Adams d'ordre 4. Les codes les plus modernes utilisent de plus un contrôle du pas ainsi qu'un contrôle de l'ordre à chaque étape, adaptant ainsi le pas et l'ordre à la précision requise et aux irrégularités éventuelles de la solution. Des algorithmes multiples assez sophistiqués utilisant ces idées ont été développés en particulier par Gear et fournissent actuellement les méthodes les plus performantes. Signalons que, pour des problèmes mal conditionnés (dits raides), on doit utiliser des méthodes moins performantes mais plus stables : les plus efficaces sont celles des différentiations rétrogrades que nous ne développerons pas ici. Elles sont

évidemment complètement implicites (les itérations éventuelles sont menées jusqu'à la convergence).

En ce qui concerne un choix entre les méthodes de Runge-Kutta et celles d'Adams, signalons les avantages et les inconvénients de chacune des méthodes :

– Runge-Kutta : c'est une méthode explicite, relativement stable et précise, facile à implémenter et aisément adaptable à un pas variable (donc contrôle de pas aisé). De plus, elle est "self-starting", c'est-à-dire qu'une seule valeur $y^0 \simeq y(0)$ suffit pour l'initier.

Elle présente cependant deux désavantages importants : chaque itération requiert 4 évaluations de fonctions (pour atteindre l'ordre 4). De plus, on n'a aucun contrôle effectif de l'erreur locale ce qui rend difficile un bon contrôle du pas. On peut pallier ce dernier point en couplant deux méthodes, mais ceci augmente encore le coût.

– Adams : pour une précision d'ordre 4, une seule évaluation de fonction est nécessaire à chaque pas dans la méthode explicite, ce qui est bien sûr remarquable. On perd en stabilité par rapport à Runge-Kutta, mais on peut stabiliser en utilisant un correcteur. Ceci augmente le coût mais permet aussi, par comparaison, une estimation de l'erreur locale.

Les difficultés viennent d'une part de l'initiation : pour les méthodes à 4 pas, 4 valeurs sont nécessaires pour initier la méthodes. On peut les obtenir par une méthode de Runge-Kutta appliquée plusieurs fois. on peut aussi prendre une méthode à nombre de pas variables en commençant par un pas et montant progressivement à 4. Il faut alors faire un contrôle sérieux de l'erreur.

Un autre désavantage est que le fait de changer de pas apporte un surcoût important par la nécessité de recalculer les coefficients de la formule. Ceux que nous avons donnés (les $b_{i,r}$) correcpondent à un pas constant. Ils sont extrêmement plus compliqués si les $t_{n+1}, t_n, t_{n-1}, ..., t_{n-r}$ correspondent à des accroissements variables. On préfère alors modifier l'ordre (i.e. le nombre de pas) ce qui donne un surcoût de calcul très réduit. On prend alors le risque d'augmenter l'instabilité...

Conclusion : L'écriture d'un bon programme de résolution de systèmes différentiels passe par la bonne compréhension des algorithmes élémentaires développés dans les paragraphes précédents. On peut à partir de là écrire des algorithmes performants adaptés aux situations particulières des utilisateurs en retenant les idées générales suivantes :

– les méthodes implicites sont plus stables mais plus coûteuses que les explicites.
– augmenter l'ordre augmente aussi l'instabilité.
– un contrôle du pas est généralement nécessaire, mais augmente considérablement le coût du calcul.
– les problèmes raides (mal conditionnés) doivent être abordés avec beaucoup de circonspection et des algorithmes spécifiques.

5.7 Applications à des problèmes aux limites

Soit à résoudre par exemple :

$$\begin{cases} y'' = y^3 + y + 1 \text{ sur } (0,1) \\ y(0) = a,\ y(1) = b,\ a \text{ et } b \text{ donnés.} \end{cases}$$

Il ne s'agit pas ici d'une équation différentielle ordinaire puisque la solution ne dépend plus seulement de "l'état initial" en $t = 0$, mais aussi de $y(1)$ qui est donné. Il s'agit d'un "problème aux limites", donc de nature tout à fait différente. On peut cependant le résoudre à l'aide des techniques précédentes couplées avec une méthode de tir.

Commençons par résoudre le système différentiel ordinaire

$$\begin{cases} y''_\alpha = y_\alpha^3 + y_\alpha + 1,\ t > 0 \\ y_\alpha(0) = a \\ y'_\alpha(0) = \alpha \text{ paramètre.} \end{cases}$$

Ce problème entre dans le cadre précédent en posant $U(t) = \begin{pmatrix} y(t) \\ y'(t) \end{pmatrix}$ comme il a été vu en début de chapitre. On peut donc le résoudre à l'aide des méthodes décrites plus haut. On essaie alors de déterminer le paramètre α pour que $y_\alpha(1) = b$. La Figure 5.3

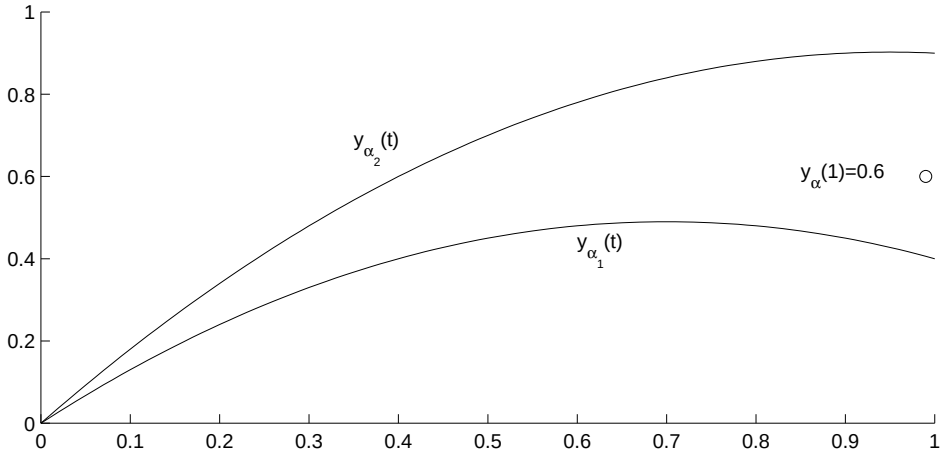

FIGURE 5.3 – La méthode du tir pour résoudre un problème aux limites

illustre bien la terminologie "méthode de tir" : on essaie de "viser" assez juste pour que $y_\alpha(1) = b$.

La valeur $y_\alpha(1)$ est une fonction non linéaire de α et le problème consiste à résoudre l'équation non linéaire $y_\alpha(1) - b = 0$. Partant de deux valeurs initiales α_0, α_1, on peut alors lui appliquer la méthode de la sécante, soit

$$\alpha_{n+1} = \alpha_n - (y_{\alpha_n}(1) - b)\frac{\alpha_n - \alpha_{n-1}}{y_{\alpha_n}(1) - y_{\alpha_{n-1}}(1)}.$$

Evidemment, chaque calcul de $y_\alpha(1)$ nécessite la résolution de l'équation différentielle le long de $(0,1)$. Cette méthode s'avère donc très vite coûteuse. Elle peut cependant être convenable dans certaines situations et l'idée est à retenir.

5.8 Schéma de Newmark pour les problèmes d'ordre 2

De nombreux problèmes de la physique des phénomènes vibratoires nécessitent la résolution d'équations différentielles d'ordre 2 du type

$$\begin{cases} y''(t) = \theta(t, y(t), y'(t)) \quad t > 0 \\ y(0) = y_0 \\ y'(0) = y_1 \end{cases} \tag{5.61}$$

où $y : [0,T] \to \mathbb{R}^N$ est la fonction inconnue, $\theta : [0,T] \times \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$ est une fonction régulière (parfois non linéaire) et $y_0, y_1 \in \mathbb{R}^N$ décrivent l'état initial. Ainsi la discrétisation par éléments finis de l'équation des ondes

$$\frac{\partial^2 u}{\partial t^2} - \Delta u = g,\ u(0) = u_0,\ \frac{\partial u}{\partial t}(0) = 0$$

conduit à une équation du type

$$\begin{cases} M\frac{d^2\zeta}{dt^2}(t) + R\zeta(t) = h(t) \\ \zeta(0) = \zeta_0,\ \frac{d\zeta}{dt}(0) = \zeta_1 \end{cases}$$

où ζ est un vecteur de $\mathbb{R}^N$ représentant la solution u "discrétisée" et M et R sont des matrices d'ordre N. Si M est inversible, cette équation rentre dans le cadre (5.61).
Comme nous l'avons vu en début de chapitre, et comme il vient juste d'être rappelé au paragraphe précédent, le système (5.61) peut s'écrire comme un système d'ordre 1 en dimension $2N$. Nous décrirons ici une autre méthode tenant mieux compte de la particularité de (5.61). Nous la décrirons en supposant $N = 1$, son extension au cas de systèmes étant évidente. Posons $t_{n+1} = t_n + h$. Si la solution $y(t)$ de (5.61) est suffisamment régulière, à l'aide de la formule de Taylor, on montre que

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + h^2\left(\beta y''(t_{n+1}) + \left(\frac{1}{2} - \beta\right)y''(t_n)\right) + O\left(h^3\right) \tag{5.62}$$

$$y'(t_{n+1}) = y'(t_n) + h\left(\gamma y''(t_{n+1}) + (1-\gamma)y''(t_n)\right) + O\left(h^2\right), \tag{5.63}$$

ce pour tout choix des paramètres β et γ. On obtient le schéma de Newmark en remplaçant les dérivées par des différences finies ci-dessus soit

$$y_{n+1} = y_n + hz_n + h^2\left(\beta\theta_{n+1} + \left(\frac{1}{2} - \beta\right)\theta_n\right) \tag{5.64}$$

$$z_{n+1} = z_n + h\left(\gamma\theta_{n+1} + (1-\gamma)\theta_n\right) \tag{5.65}$$

où $\theta_n = \theta(t_n, y_n, z_n)$.

Notons que ce schéma est implicite et nécessite la résolution d'un système (qui peut être non linéaire) pour obtenir (y_{n+1}, z_{n+1}) a partir de (y_n, z_n). Il se simplifie lorsque θ ne dépend pas de y', car on peut alors éliminer z_n dans (5.64), (5.65) en écrivant

$$\begin{aligned} y_{n+2} - 2y_{n+1} + y_n &= (y_{n+2} - y_{n+1}) - (y_{n+1} - y_n) \\ &= h(z_{n+1} - z_n) + h^2\left(\beta\theta_{n+2} + \left(\frac{1}{2} - 2\beta\right)\theta_{n+1} - \left(\frac{1}{2} - \beta\right)\theta_n\right), \end{aligned}$$

c'est-à-dire

$$y_{n+2} - 2y_{n+1} + y_n = h^2\left(\beta\theta_{n+2} + \left(\frac{1}{2} - 2\beta + \gamma\right)\theta_{n+1} + \left(\frac{1}{2} + \beta - \gamma\right)\theta_n\right) \tag{5.66}$$

Ce schéma est explicite si $\beta = 0$, ce pour tout γ.

L'erreur de consistance dans (5.64), (5.65) est évaluée en complétant les développements de Taylor (5.62), (5.63). On obtient facilement
– dans (5.62), $O\left(h^3\right) = h^3\left(\frac{1}{6} - \beta\right)y^{(3)}(t_n) + O\left(h^4\right)$
– dans (5.63), $O\left(h^2\right) = h^2\left(\frac{1}{2} - \gamma\right)y^{(3)}(t_n) + O\left(h^3\right)$.

L'erreur globale de consistance sera donc en h^2 si $\theta = \frac{1}{2}$. La stabilité du schéma peut être analysée à l'aide de l'équation différentielle modèle

$$y'' + \omega^2 y = 0,\ \omega > 0.$$

Une condition nécessaire de stabilité est alors que

$$\gamma \geq \frac{1}{2}$$

et

$$\beta \geq \frac{\gamma}{2} \quad \text{ou} \quad \beta < \frac{\gamma}{2} \quad \text{et} \quad \omega h < \frac{2}{\gamma - 2\beta}.$$

5.9 Exercices du chapitre 5

Exercice 5.1 Résoudre le système (5.4) quand la matrice A est diagonalisable.

Exercice 5.2 Vérifier que les problèmes

$$y' = -y^2 \quad y(0) = y_0 \tag{5.67}$$

$$y' = -y^{\frac{1}{3}} \quad y(0) = y_0 \tag{5.68}$$

ont une solution unique sur $[0, \infty[$.

Exercice 5.3 Soit $y'(t) = Ay(t)$ où A est une matrice carrée $N \times N$ diagonalisable de valeurs propres $0 < \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_N$.
Montrer que si $\lambda_N h < 2$, la méthode d'Euler est stable au sens que y_n reste borné pour n grand.

Exercice 5.4 Pour résoudre numériquement $y' = -\lambda y$, on décide d'approcher $y'(t_n)$ par les différences finies centrées

$$y'(t_n) \simeq \frac{y(t_{n+1}) - y(t_{n-1})}{2h},\ h = t_{n+1} - t_{n-1},\ \forall n.$$

Ceci conduit au schéma

$$y_{n+1} = y_{n-1} - 2\lambda h y_n.$$

Montrer que ce schéma est toujours instable.

Exercice 5.5 Pour résoudre l'équation différentielle

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(0) = y_0 \end{cases}$$

(avec f lipschitzienne par rapport à la deuxième variable), on envisage une méthode à pas multiple :

$$y_{n+1} = y_n + h_n(\beta_0 f_{n-2} + \beta_1 f_{n-1} + \beta_2 f_n.$$

Déterminer les coefficients $\beta_0, \beta_1, \beta_2$ pour que la mt́hode soit d'ordre maximum. Est-elle alors stable, consistante, convergente ?

Exercice 5.6 On considère l'équation différentielle

$$\begin{cases} y'(t) = e^{-ty(t)} \\ y(0) = 0\ . \end{cases}$$

Montrer l'existence et l'unicité locale, puis globale sur $\mathbb{R}$ de la solution. En déduire que celle-ci est impaire. Montrer qu'elle admet une limite quand $t \to +\infty$.

Chapitre 6

Résolution de systèmes linéaires

Les deux problèmes fondamentaux de l'analyse numérique matricielle sont
- la résolution de systèmes linéaires, i.e. la recherche de vecteurs X solutions de

$$AX = b$$

  où A est une matrice -le plus souvent carrée- à coefficients réels (ou complexes) et b un vecteur donné
- le calcul de valeurs propres et de vecteurs propres d'une matrice, i.e. la recherche des scalaires λ (réels ou complexes) et des vecteurs non nuls X tels que

$$AX = \lambda X,$$

  A matrice carrée donnée.

Nous traitons dans ce chapitre le premier type de problèmes, le calcul de valeurs propres et vecteurs propres étant étudié au chapitre 7.

6.1 Quelques remarques générales

6.1.1 Origines des problèmes

Elles sont très variées et apparaissent déjà en plusieurs endroits dans les autres chapitres de ce cours. Citons quelques-uns des problèmes conduisant à la résolution de systèmes linéaires :
- Résolution d'équations aux dérivées partielles (ceci en constitue une source très importante. Voir quelques exemples au paragraphe 6.4.4)
- Approximation au sens des moindres carrés
- Calcul de fonctions splines
- Programmation linéaire (méthode du simplexe, d'Uzawa)
- Algorithmes d'optimisation non linéaire
- etc...

6.1.2 Méthodes utilisées

Elles sont de deux types
- les méthodes directes : celles où on obtient la valeur exacte de la solution (aux erreurs d'arrondi près) en un nombre fini d'opérations,

– les méthodes itératives : elles consistent à construire une suite de vecteurs X^k convergeant vers la solution X cherchée. On s'arrête bien sûr au bout d'un nombre fini n d'itérations choisi pour que X^n soit suffisamment voisin de X.

Remarque 6.1 On n'utilise jamais les formules de Cramer car elles nécessitent le calcul de déterminants qui requièrent un nombre trop important d'opérations élémentaires.

Remarque 6.2 On ne calcule pas A^{-1} pour résoudre $AX = b$. Numériquement, c'est le contraire qui se produit : le calcul de A^{-1}, ainsi d'ailleurs que celui de $\det A$, sont des sous-produits des méthodes de résolution de systèmes. Ainsi, le calcul de A^{-1} s'effectuera en résolvant successivement les systèmes

$$\{AX_i = e_i,\ i = 1, ..., N\}$$

où e_i est le ième vecteur de la base canonique de $\mathbb{R}^N$. La ième colonne de A^{-1} est donnée par X_i comme on le vérifie aisément.

6.1.3 Structure de la matrice

La structure de la matrice a une incidence fondamentale sur la difficulté de la résolution et sur le choix de la méthode

6.1.3.1 Cas d'une matrice diagonale

Tous les éléments sont nuls sauf ceux de la diagonale principale :

$$A = \begin{bmatrix} a_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_N \end{bmatrix}$$

La résolution de $AX = b$ est alors immédiate. Si $X = (x_1, ..., x_{\mathrm{N}})$ et $b = (b_1, ..., b_{\mathrm{N}})$, on a

$$x_i = \frac{b_i}{a_i},\ i = 1, ..., N$$

ceci en supposant bien sûr que tous les a_i sont non nuls.

6.1.3.2 Matrice triangulaire supérieure (ou inférieure)

Tous les éléments au-dessous (ou au-dessus) de la diagonale sont nuls

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1\mathrm{N}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{\mathrm{NN}} \end{bmatrix}$$

Dans ce cas on utilisera toujours une méthode directe de résolution par remontée soit :

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1\mathrm{N}}x_{\mathrm{N}} = b_1 \\ a_{22}x_2 + ... + a_{2\mathrm{N}}x_{\mathrm{N}} = b_2 \\ \qquad\qquad ..... \\ a_{\mathrm{N}-1,\mathrm{N}-1}x_{\mathrm{N}-1} + a_{\mathrm{N}-1,\mathrm{N}}x_{\mathrm{N}} = b_{\mathrm{N}-1} \\ a_{\mathrm{NN}}x_{\mathrm{N}} = b_{\mathrm{N}} \end{cases} \tag{6.1}$$

On commence par résoudre la dernière équation ; on substitue le résultat obtenu pour x_{N} dans la précédente, ce qui permet de calculer $x_{\mathrm{N}-1}$, etc...

$$\begin{cases} x_{\mathrm{N}} = b_{\mathrm{N}} / a_{\mathrm{NN}} \\ x_{\mathrm{N}-1} = \left(b_{\mathrm{N}-1} - a_{\mathrm{N}-1,\mathrm{N}} x_{\mathrm{N}}\right) / a_{\mathrm{N}-1,\mathrm{N}-1} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_i = \left(b_i - a_{i,\mathrm{N}} x_{\mathrm{N}} - a_{i,\mathrm{N}-1} x_{\mathrm{N}-1} - \ldots - a_{i,i+1} x_{i+1}\right) / a_{ii} \end{cases} \tag{6.2}$$

Il est intéressant de compter déjà ici le nombre d'opérations élémentaires (i.e. addition, multiplication, division) nécessaires pour mener à bien le calcul de tous les x_i. Chaque x_i nécessite (une fois x_{i+1}, x_{i+2}, ..., x_{N} calculés) 1 division, $N-i$ multiplications, $N-i$ additions.
Puisque $\sum_{i=N}^{1}(N-i) = \sum_{j=0}^{N-1} j = \frac{N(N-1)}{2}$, il faut au total N divisions, $\frac{N(N-1)}{2}$ multiplications, $\frac{N(N-1)}{2}$ additions, soit N^2 opérations élémentaires.
Le cas des matrices triangulaires inférieures se traite de façon identique par un procédé de descente.

6.1.3.3 Autres cas

Dans le cas général, les méthodes de résolution seront d'autant plus simples que les deux phénomènes suivants seront plus importants :
- la matrice est creuse (c'est-à-dire a beaucoup de zéros)
- les termes non nuls de la matrice sont localisés et en particulier sont rassemblés autour de la diagonale. Ceci conduit à la notion de matrice-bande.

Exemples : *matrice tridiagonale*

$$A = \begin{bmatrix} \bullet & \bullet & 0 & \cdots & 0 \\ \bullet & \bullet & \bullet & \ddots & \vdots \\ 0 & \bullet & \bullet & \bullet & 0 \\ \vdots & \ddots & \bullet & \bullet & \bullet \\ 0 & \cdots & 0 & \bullet & \bullet \end{bmatrix}$$

Tous les éléments sont nuls sauf les $a_{i,j}$ pour lesquels $i-1 \leq j \leq i+1$. Nous verrons plus loin une méthode directe de résolution du système avec un nombre d'opérations $8N-6$, donc très faible.

matrice pentadiagonale :

$$A = \begin{bmatrix} \bullet & \bullet & \bullet & 0 & \cdots & 0 \\ \bullet & \bullet & \bullet & \bullet & \ddots & \vdots \\ \bullet & \bullet & \bullet & \bullet & \bullet & 0 \\ 0 & \bullet & \bullet & \bullet & \bullet & \bullet \\ \vdots & \ddots & \bullet & \bullet & \bullet & \bullet \\ 0 & \cdots & 0 & \bullet & \bullet & \bullet \end{bmatrix}$$

matrice diagonale par blocs

$$A = \begin{bmatrix} A_{11} & & & & \\ & A_{22} & & & \\ & & A_{33} & & \\ & & & A_{44} & \end{bmatrix}$$

Dans ce cas le système associé se réduit à plusieurs "petits" systèmes découplés de dimension réduite (mais pas nécessairement égale), soit

$$\{A_{ii}X_i = b_i,\ i = 1, 2, 3, 4\}$$

où $X = (X_1, X_2, X_3, X_4)\,, b = (b_1, b_2, b_3, b_4)\,.$

On définit de façon analogue les matrices tridiagonales par blocs, auxquelles on étend les techniques développées pour les matrices tridiagonales au sens ci-dessus : celles-ci sont alors appelées tridiagonales par points et apparaissent comme des cas particuliers des tridiagonales par blocs.

6.1.3.4 Matrices symétriques

Une autre propriété de la matrice peut jouer un rôle important, c'est la symétrie. On dit qu'une matrice A est symétrique si elle coincide avec sa transposée, c'est-à-dire si

$$a_{ij} = a_{ji} \qquad \forall i, j.$$

Un sous-ensemble de ces matrices donnera lieu à une étude particulière, c'est celui des matrices symétriques définies positives. On dit qu'une matrice symétrique est positive si

$$\forall X \in \mathbb{R}^N \qquad {}^tXAX \geq 0$$

($X \longrightarrow {}^tXAX$ est une forme quadratique). Elle est de plus définie positive si ${}^tXAX$ n'est nul que si X est nul.

Exemple : $A = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$ est symétrique définie positive car :

$$\begin{aligned} \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2x & -y \\ -x & 2y \end{bmatrix} \\ &= 2\left(x^2 - xy + y^2\right) \\ &= 2\left[\left(x - \frac{1}{2}y\right)^2 + \frac{3}{4}y^2\right] > 0 \text{ si } (x, y) \neq (0, 0)\,. \end{aligned}$$

La matrice $\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ est symétrique positive, non définie car

$$\begin{aligned} \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} x & -y \\ -x & y \end{bmatrix} \\ &= x^2 - 2xy + y^2 = (x - y)^2 \geq 0 \end{aligned}$$

mais $(x - y)^2$ s'annule si $x = y$.

Des méthodes particulières seront développées pour ces matrices qui apparaissent très souvent dans la discrétisation de problèmes variationnels (minimisation de fonctionnelles). Des exemples ont déjà été donnés dans les problèmes de minimisation au sens des moindres carrés (cf le paragraphe sur l'ajustement linéaire).

6.1.4 Stockage des matrices

Le stockage de tous les éléments d'une matrice d'ordre N requiert bien sûr en mémoire le stockage de N^2 nombres réels. La discrétisation de certains problèmes, en particulier les équations aux dérivées partielles en dimension 2 ou 3 conduisent rapidement à des matrices d'ordre N égal à plusieurs millions ; d'où la nécessité "d'optimiser" le stockage des matrices. Indiquons ici quelques techniques souvent utilisées.

– Si la matrice est symétrique, on ne stockera bien sûr que les éléments diagonaux et surdiagonaux soit au total $\frac{N(N+1)}{2}$ si la dimension est N.
– De très grands systèmes (plusieurs millions d'équations) peuvent être facilement traités si les matrices sont creuses (sparse en anglais) comme c'est souvent le cas pour celles provenant de la discrétisation des équations aux dérivées partielles. On obtient alors le plus souvent des matrices-bandes dont la largeur de bande est petite devant la dimension :

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1b} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2b} & a_{2,b+1} & \ddots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots & \ddots & 0 \\ a_{b1} & a_{b2} & \cdots & a_{bb} & a_{b,b+1} & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & a_{\mathrm{N,N}-b+1} & \cdots & \cdots & a_{\mathrm{NN}} \end{bmatrix}$$

avec $b << N$ (par exemple $b = 100$ et $N = 10000$). On essaiera dans la mesure du possible de ne stocker que les éléments non nuls. On peut par exemple utiliser un tableau "redressé" comme suit :

$$M = \begin{bmatrix} 0 & & & a_{11} & a_{12} & \cdots & a_{1b} \\ & & a_{21} & a_{22} & a_{23} & \cdots & a_{2,b+1} \\ a_{b1} & a_{b2} & \cdots & \cdots & \cdots & \cdots & a_{b,2b-1} \\ \vdots & & & & & & 0 \\ \vdots & & & & & & \vdots \\ \vdots & & & & & & \vdots \\ a_{\mathrm{N,N}-b+1} & \cdots & a_{\mathrm{NN}} & 0 & \cdots & \cdots & 0 \end{bmatrix}$$

tableau de dimension $(2b-1) \times N << N^2$ si $b << N$, $a_{IJ} = m_{ij}$ si $\begin{cases} i = I \\ j = J - I + b \end{cases}$.

Cette méthode conduit cependant à mémoriser $b\,(b-1)$ valeurs nulles inutiles. On lui préfère plutôt la méthode suivante :

– matrice profil (ou méthode de la "ligne de ciel") : elle consiste à tracer une ligne fermée contenant les éléments non nuls de la matrice et laissant à l'extérieur le plus de zéros possibles.

Commençons par le cas d'une matrice symétrique.

$$A = \begin{bmatrix} 7 & . & . & . & . & . \\ 2 & 11 & . & . & . & . \\ 0 & 1 & 3 & . & . & . \\ 0 & 0 & 4 & -1 & . & . \\ 6 & 3 & 0 & 2 & -3 & . \\ 0 & 0 & 0 & 12 & 0 & -7 \end{bmatrix}$$

$$\begin{bmatrix} 1 & & & & & \\ 2 & 3 & & & & \\ & 4 & 5 & & & \\ & & 6 & 7 & & \\ 8 & 9 & 10 & 11 & 12 & \\ & & & 13 & 14 & 15 \end{bmatrix}$$ tableau des positions

Le stockage de A est réalisé à l'aide de deux tableaux M et P où M contient les éléments de la matrice à l'intérieur de la ligne de profil et P est un tableau de pointeurs de dimension $N+1$ avec $P(1)=0$, $P(i+1)=$ adresse dans M du ième élément diagonal de A. Ces deux tableaux caractérisent A compte-tenu que les éléments de A sont rangés ligne par ligne et pour une ligne donnée dans l'ordre croissant des colonnes, ce qui donne dans l'exemple ci-dessus :

$$\begin{array}{lllllllllllllll} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ M= & (7 & 2 & 11 & 1 & 3 & 4 & -1 & 6 & 3 & 0 & 2 & -3 & 12 & 0 & -7) \\ P= & (0 & 1 & 3 & 5 & 7 & 12 & 15) & & & & & & & & \end{array}$$

la donnée de M et P permet de retrouver A à l'aide de la formule suivante qu'on obtient de façon élémentaire :

$$\begin{cases} \forall i=1,...,N,\ \forall j=i+1-(P(i+1)-P(i)) \\ \qquad a_{ij}=M(P(i+1)+j-i). \end{cases}$$

On utilise le même principe pour une matrice non symétrique, mais le profil sera toujours choisi symétrique.

$$A=\begin{bmatrix} 7 & 3 & 0 & 0 & 0 & 0 \\ 2 & 11 & -1 & 0 & 4 & 0 \\ 0 & 1 & 3 & 0 & 0 & 0 \\ 0 & 0 & 4 & -1 & -2 & 3 \\ 6 & 3 & 0 & 2 & -3 & 1 \\ 0 & 0 & 0 & 12 & 0 & -7 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 3 & . & . & 15 & . \\ 2 & 4 & 6 & . & 16 & . \\ . & 5 & 7 & 9 & 17 & . \\ . & . & 8 & 10 & 18 & 22 \\ 11 & 12 & 13 & 14 & 19 & 23 \\ . & . & . & 20 & 21 & 24 \end{bmatrix}$$ tableau des positions

La marice A est stockée dans le tableau M suivant la règle du tableau des positions ci-dessus : c'est-à-dire de $i=1$ à N on parcourt la ligne i jusqu'à l'élément diagonal (exclu), puis on descend la colonne i jusqu'à l'élément diagonal (inclus). Ici encore $P(i+1)$ est l'adresse du ième élément diagonal. Avec l'exemple ci-dessus, on a donc :

$$\begin{array}{rcl} M & = & (7\ 2\ 3\ 11\ 1\ -1\ 3\ 4\ 0\ -1\ 6\ 3\ 0\ 2\ 0\ 4\ 0\ -2\ -3\ 12\ 0\ 3\ 1\ -7) \\ P & = & (0\ 1\ 4\ 7\ 10\ 19\ 24) \end{array}$$

On retrouve A à l'aide de M et P par la règle

$$i-\frac{P(i+1)-P(i)+1}{2}<j<i,\ a_{ij}=M\left(\frac{P(i+1)+P(i)+1}{2}+j-i\right)$$

$$i\leq j<i+\frac{P(j+1)-P(j)+1}{2},\ a_{ij}=M(P(j+1)+i-j).$$

Remarque 6.3 Le gain en mémoires occupées n'est évidemment pas probant dans les exemples numériques ci-dessus. Cependant, un calcul simple à l'aide des formules ci-dessus montre que le gain est important pour des matrices-bandes de grandes dimensions.

Si on veut se débarrasser de tous les éléments non nuls, on peut stocker la matrice sous forme de
matrice morse : on remplace la matrice A par trois tableaux M, P_1, P_2 où on a (par exemple) la règle suivante :
- M est le tableau des éléments non nuls de A
- P_1 est un tableau de pointeurs de dimension $N+1$ avec $P_1(1) = 0$ et $P_1(i+1) =$ adresse du ième coefficient diagonal dans M et
- P_2 est un tableau de pointeurs de dimension égale au nombre d'éléments non nuls de A avec $P_2(k) =$ numéro de la colonne de $M(k)$.

Si M est symétrique, la partie triangulaire inférieure (par exemple) est stockée ligne par ligne de la gauche vers la droite. Si M est quelconque, les coefficients sont rangés ligne par ligne de gauche à droite, le coefficient diagonal étant le dernier.
Dans les deux exemples numériques ci-dessus, on obtient les tableaux de positions suivants

$$\begin{bmatrix} 1 & . & . & . & . & . \\ 2 & 3 & . & . & . & . \\ . & 4 & 5 & . & . & . \\ . & . & 6 & 7 & . & . \\ 8 & 9 & . & 10 & 11 & . \\ . & . & . & 12 & . & 13 \end{bmatrix} \quad \begin{bmatrix} 2 & 1 & . & . & . & . \\ 3 & 6 & 4 & . & 5 & . \\ . & 7 & 8 & . & . & . \\ . & . & 9 & 12 & 10 & 11 \\ 13 & 14 & . & 15 & 17 & 16 \\ . & . & . & 18 & . & 19 \end{bmatrix}$$

On en déduit les tableaux M, P_1, P_2. On montre facilement que la connaissance de ces trois tableaux donne A.

6.2 Méthodes directes de résolution de $AX = b$

6.2.1 Méthode d'élimination de Gauss

Principe : il s'agit de la méthode élémentaire d'élimination des variables. Rappelons-la d'abord sur un exemple simple :

$$(S)\begin{cases} ax+by+cz=d \\ a'x+b'y+c'z=d' \\ a''x+b''y+c''z=d'' \end{cases} \quad \text{soit} \quad \begin{cases} P_1(x,y,z)=d \\ P_2(x,y,z)=d' \\ P_3(x,y,z)=d'' \end{cases}$$

Supposons $a \neq 0$: on peut alors éliminer x dans les deuxième et troisième équations en remplaçant (S) par le système équivalent

$$(S_1)\begin{cases} P_1 = d \\ P_2 - \dfrac{a'}{a}P_1 = d' - \dfrac{a'}{a}d \\ P_3 - \dfrac{a''}{a}P_1 = d'' - \dfrac{a''}{a}d \end{cases},$$

qui est de la forme

$$(S_1)\begin{cases} P_1 = d \\ Q_2 = d_2 \\ Q_3 = d_3 \end{cases} \text{ où } \quad \begin{cases} Q_2 = b_2y + c_2z \\ Q_3 = b_3y + c_3z \end{cases}.$$

Supposons $b_2 \neq 0$: on peut alors éliminer y dans la dernière équation en remplaçant (S_1) par le système équivalent

$$(S_2)\left\{\begin{array}{c} P_1 = d \\ Q_2 = d_2 \\ Q_3 - \dfrac{b_3}{b_2}Q_2 = d_3 - \dfrac{b_3}{b_2}d_2 \end{array}\right. \quad \text{soit} \quad \left\{\begin{array}{r} ax + by + cz = d \\ b_2 y + c_2 z = d_2 \\ c'_3 z = d'_3 \end{array}\right.$$

qui est un système triangulaire ; on peut le résoudre à l'aide de la méthode de remontée signalée au paragraphe précédent.
La méthode générale est absolument la même. La seule difficulté technique qui se présente est que l'un des pivots utilisés (a, b_2, c'_3 dans le calcul précédent) peut être nul auquel cas la méthode échoue. Même s'il n'est pas nul mais très petit, ceci peut conduire à des erreurs très importantes. On peut s'en convaincre à l'aide de l'exemple simple suivant : soit à résoudre :

$$(S)\left\{\begin{array}{r} 10^{-4}x + y = 1 \\ x + y = 2 \end{array}\right. .$$

La solution "exacte" de (S) est :

$$x = 1,00010... \simeq 1, \quad y = 0,999990... \simeq 1.$$

Prenant 10^{-4} comme pivot, on est ramené au système équivalent :

$$\left\{\begin{array}{c} 10^{-4}x + y = 1 \\ (x+y) - \frac{1}{10^{-4}}\left(10^{-4}x + y\right) = 2 - \frac{1}{10^{-4}} \end{array}\right. \iff \left\{\begin{array}{r} 10^{-4}x + y = 1 \\ \left(10^4 - 1\right) y = 10^4 - 2 \end{array}\right. .$$

Supposons qu'on travaille avec trois chiffres significatifs : le système est alors équivalent à

$$\left\{\begin{array}{l} 10^{-4}x + y = 1 \\ 9990y = 9990 \end{array}\right. \quad \text{soit} \quad \left\{\begin{array}{l} x = 0 \\ y = 1 \end{array}\right. !$$

Au contraire, un échange préalable des deux équations donnera un résultat satisfaisant même avec seulement trois chiffres significatifs :

$$\begin{array}{l} \left\{\begin{array}{l} x + y = 2 \\ 10^{-4}x + y = 1 \end{array}\right. \iff \left\{\begin{array}{r} x + y = 2 \\ 10^{-4}x + y - 10^{-4}(x+y) = 1 - 2.10^{-4} \end{array}\right. \\ \qquad\qquad \iff \left\{\begin{array}{l} x + y = 2 \\ 0,999y = 0,999 \end{array}\right. \iff \left\{\begin{array}{l} x = 1 \\ y = 1 \end{array}\right. ! \end{array}$$

Nous allons décrire le passage de l'étape p à l'étape $p+1$ dans la méthode de Gauss pour la résolution de

$$AX = b.$$

L'élimination sucessive des variables conduit à un système équivalent

$$A_p X = b^p$$

où A_p est de la forme

$$A_p = \begin{bmatrix} a_{11}^p & a_{12}^p & \cdots & \cdots & \cdots & \cdots & \cdots & a_{1\mathrm{N}}^p \\ 0 & a_{22}^p & \cdots & \cdots & \cdots & \cdots & \cdots & a_{2\mathrm{N}}^p \\ \vdots & \ddots & \ddots & & & & & \vdots \\ \vdots & & \ddots & \ddots & & & & \vdots \\ \vdots & & & \ddots & \ddots & a_{pp}^p & \cdots & a_{p,\mathrm{N}}^p \\ \vdots & & & & 0 & a_{p+1,p}^p & \cdots & a_{p+1,\mathrm{N}}^p \\ \vdots & & & & \vdots & \vdots & & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & a_{\mathrm{N},p}^p & \cdots & a_{\mathrm{NN}}^p \end{bmatrix}.$$

Le passage à l'étape suivante consiste à utiliser la p-ième ligne (c'est-à-dire la p-ième équation) pour faire apparaître des zéros à la place de $a^p_{i,p}$, $i = p+1, ...N$, c'est-à-dire éliminer la variable x_p dans les $N-p$ dernières équations). Auparavant, il faut s'assurer que a^p_{pp} n'est pas nul ni même trop petit. Pour cela, on adopte généralement une stratégie du pivot.

– Stratégie du pivot partiel : on détermine l'élément a^p_{ip} (ou l'un des éléments a^p_{ip}) tel que $p \leq i \leq N$ et

$$\left|a^p_{ip}\right| = \max_{p \leq k \leq N} \left|a^p_{kp}\right|.$$

On permute alors les lignes d'indices i et p pour amener en position de pivot l'élément a^p_{ip}.

– Stratégie du pivot total : on détermine l'élément a^p_{ij} (ou l'un des éléments a^p_{ij}) tel que $p \leq i,\, j \leq N$ et

$$\left|a^p_{ij}\right| = \max_{p \leq l,k \leq N} \left|a^p_{l,k}\right|.$$

On effectue alors des permutations de lignes et de colonnes (c'est-à-dire une permutation sur les inconnues) pour amener en position de pivot l'élément $a^p_{i,j}$.

Remarque 6.4 on se contente le plus souvent de la stratégie du pivot partiel l'autre étant réservée aux systèmes particulièrement délicats ou susceptibles de l'être.

Notons que, si la matrice est inversible, la stratégie de pivot partiel donne toujours un pivot non nul. En effet, on vérifie immédiatement que

$$\det A_p = a^p_{11}.a^p_{22}...a^p_{p-1,p-1}.\det A'_p$$

où

$$A'_p = \begin{bmatrix} a^p_{pp} & \cdots & a^p_{p\mathrm{N}} \\ \vdots & & \vdots \\ a^p_{\mathrm{N}p} & \cdots & a^p_{\mathrm{NN}} \end{bmatrix}.$$

$\det A_p \neq 0 \Longrightarrow \det A'_p \neq 0 \Longrightarrow$ la colonne $\begin{pmatrix} a^p_{pp} \\ \vdots \\ a^p_{\mathrm{N}p} \end{pmatrix}$ n'est pas égale au vecteur nul.

Supposons maintenant qu'une stratégie du pivot ait été effectuée et que (en gardant les mêmes notations pour A_p)

$$a^p_{pp} \neq 0.$$

Pour obtenir A_{p+1}, on conserve la p-ième équation

$$a^p_{pp}x_p + a^p_{p,p+1}x_{p+1} + ... + a^p_{p,N}x_{\mathrm{N}} = b^p_p$$

et on remplace chaque équation

$$a^p_{ip}x_p + a^p_{i,p+1}x_{p+1} + ... + a^p_{i,\mathrm{N}}x_{\mathrm{N}} = b^p_i,\ i = p+1, ...N$$

par l'équation obtenue en lui retranchant la p-ième multipliée par a^p_{ip}/a^p_{pp}. On obtient ainsi la nouvelle matrice A_{p+1} définie par

$$a^{p+1}_{i,j} = a^p_{i,j} - \frac{a^p_{ip}}{a^p_{pp}} a^p_{p,j} \tag{6.3}$$

$$p+1 \leq i \leq N \tag{6.4}$$

$$p+1 \leq j \leq N. \tag{6.5}$$

Les lignes $i = 1$ à p ne sont pas modifiées et les autres coefficients sont nuls. Le second membre est transformé de façon identique

$$b_i^{p+1} = b_i^p - \frac{a_{ip}^p}{a_{pp}^p} b_p^p \tag{6.6}$$

$$p+1 \leq i \leq N. \tag{6.7}$$

Dans la pratique, on assimile le vecteur b^p à une $(N+1)$-ème colonne de A_p en posant $a_{i,N+1}^p := b_i^p$. On obtient ainsi un traitement global. On peut d'ailleurs ajouter plusieurs colonnes ce qui permet de résoudre plusieurs systèmes simultanément.
Une fois la triangulation terminée, on résout le système par la méthode de remontée.

Remarque 6.5 comme sous-produit, on obtient un calcul simple du déterminant de A, à savoir

$$\det A = \pm a_{11}^1 a_{22}^2 ... a_{\mathrm{NN}}^N$$

où $\pm$ dépend du nombre de permutations éventuelles de lignes. On vérifie en effet que le déterminant de A_p n'est pas modifié dans le procédé d'élimination (ou de triangulation).

6.2.1.1 Calcul du nombre d'opérations élémentaires

Pour passer de A_p à A_{p+1} on a
$N-p$ divisions
$(N-p)(N-p+1)$ additions
$(N-p)(N-p+1)$ multiplications, soit

$$\sum_{p=1}^{N-1} (N-p) = \frac{N(N-1)}{2} \text{ divisions}$$

$$\sum_{p=1}^{N-1} (N-p)^2 + (N-p) = \frac{N(N-1)(2N-1)}{6} + \frac{N(N-1)}{2}$$

$$= \frac{N^3 - N}{3} \text{ multiplications et additions}$$

d'où au total $\dfrac{4N^3 + 3N^2 - 7N}{6}$ opérations élémentaires auxquelles il faut ajouter les N^2 nécessaires à la remontée du système. Lorsque N est grand, les termes en N^2 et N sont négligeables devant N^3 et le nombre d'opérations est donc de l'ordre de $\dfrac{2N^3}{3}$.

Remarque 6.6 On montre que ce nombre d'opérations est pratiquement optimal pour la résolution directe d'un système linéaire quelconque (c'est-à-dire sans aucune particularité). C'est pourquoi la méthode de Gauss est souvent utilisée dans le cas des matrices "pleines".

Noter que dans le temps de calcul, il faut aussi tenir compte de la stratégie du pivot : celle-ci peut prendre un temps non négligeable dans le cas d'un pivot total.
Les formules de Cramer nécessitent le calcul de $(n+1)$ déterminants et n divisions. Chaque déterminant calculé selon sa définition requiert $(n-1)\,n!$ multiplications, $n!-1$ additions soit $\left(n^2-1\right)n! + (n+1)! - (n+1)$ opérations. Ainsi pour $n = 10$ cela donne environ 400 000 000 opérations contre environ 900 opérations par la méthode de Gauss.

6.2.2 Dérivés de la méthode de Gauss

Nous allons voir dans ce paragraphe plusieurs variantes de la méthode de Gauss (Crout, Doolittle, Gauss-Jordan) qui auront chacune leur intérêt propre.

6.2.2.1 Factorisation LU

Supposons qu'on doive, dans un programme donné, résoudre plusieurs fois le même système linéaire

$$AX = b$$

avec différents seconds membres b, mais la même matrice A. Si tous les seconds membres b sont initialement connus, on peut alors effectuer simultanément sur tous les seconds membres les manipulations intervenant dans l'élimination de Gauss.

Le plus souvent, dans la pratique, il s'agit de résoudre des systèmes avec des b calculés au cours du processus et donc inconnus initialement. Il n'est alors pas raisonnable de recommencer la triangulation de Gauss à chaque résolution : il faut conserver ce résultat qui consiste en fait à factoriser la matrice A sous la forme $A = LU$ ou L est une matrice triangulaire inférieure et U une matrice triangulaire supérieure (L pour "lower", U pour "upper") : on conservera en mémoire L et U. Par la suite, tout système

$$(S) \qquad AX = b \Longleftrightarrow LUX = b$$

sera remplacé par la résolutions des deux systèmes équivalents

$$\left\{ \begin{array}{l} LY = b \\ UX = Y \end{array} \right. .$$

Il s'agit alors de systèmes triangulaires qui se résolvent par une méthode de "descente" puis de "remontée", soit un nombre d'opérations égal à $2N^2$.

Montrons comment l'élimination de Gauss donne une factorisation de A sous la forme $A = LU$, tout au moins quand aucune stratégie du pivot n'est appliquée ce que nous supposons dans ce qui suit. Notons $l_{ip} = a^p_{ip}/a^p_{pp}$. La matrice triangulaire obtenue en fin de méthode de Gauss est , avec les notations précédentes :

$$U = \left[\begin{array}{ccccccc} a^1_{11} & a^1_{12} & \cdots & \cdots & \cdots & \cdots & a^1_{1\mathrm{N}} \\ 0 & a^2_{22} & a^2_{23} & \cdots & \cdots & \cdots & a^2_{2\mathrm{N}} \\ \vdots & \ddots & \ddots & & & & \vdots \\ \vdots & & \ddots & \ddots & & & \vdots \\ \vdots & & & \ddots & a^p_{pp} & \cdots & a^p_{p\mathrm{N}} \\ \vdots & & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & a^N_{\mathrm{NN}} \end{array} \right].$$

Si nous notons

$$L = \left[\begin{array}{cccc} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ l_{\mathrm{N}1} & \cdots & \cdots & l_{\mathrm{NN}} \end{array} \right] \text{ (noter } l_{ii} = 1)$$

on vérifie qu'on a alors

$$A = LU.$$

En effet le terme général du produit LU est par définition

$$a'_{ij} = \sum_{p=1}^{\min(i,j)} l_{ip} a^p_{pj}.$$

D'après les relations (6.3) définissant l'algorithme de Gauss, une sommation à i, j fixés de $p = 1$ à $\min(i,j) - 1$ donne

$$\sum_{p=1}^{\min(i,j)-1} a_{ij}^{p+1} = \sum_{p=1}^{\min(i,j)-1} a_{ij}^{p} - \sum_{p=1}^{\min(i,j)-1} l_{ip} a_{pj}^{p},$$

soit après réduction des termes identiques

$$\sum_{p=1}^{\min(i,j)-1} l_{ip} a_{pj}^{p} + a_{ij}^{\min(i,j)} = a_{ij}^{1}.$$

Puisque $a_{ij}^{1} = a_{ij}$, pour obtenir $a'_{ij} = a_{ij}$, il suffit de vérifier

$$l_{iq} a_{qj}^{q} = a_{ij}^{q} \text{ si } q = \min(i,j). \tag{6.8}$$

Si $i \leq j$ et donc $q = i$, c'est immédiat puisque $l_{ii} = 1$. Si $i > j$, soit $q = j$, on a par définition de l_{ij}

$$l_{ij} = \frac{a_{ij}^{j}}{a_{jj}^{j}}$$

...ce qui est (6.8).

Remarque 6.7 nous avons ainsi démontré que, si la triangulation de Gauss peut-être conduite sans stratégie du pivot, alors la matrice A admet une factorisation LU. On peut trouver des conditions suffisantes simples pour que ceci soit possible : par exemple lorsque A est symétrique définie positive. Ceci sera explicité au paragraphe suivant (factorisation de Cholesky).

Si A est inversible, nous avons vu que l'élimination de Gauss est toujours possible pourvu qu'on applique une stratégie du pivot partiel qui revient à permuter des lignes de A. Ceci prouve que si A est inversible, il existe une matrice P produit de matrices de transposition telle que PA ait une factorisation LU. Dans la pratique, une fois les permutations nécessaires repérées on les effectue préalablement à la résolution de tout nouveau système associé à A.

6.2.2.2 Algorithme de Crout

C'est le nom donné dans la littérature à une variante de l'algorithme de Gauss : elle suppose a priori l'existence de la décomposition $A = LU$, les coefficients l_{ij}, u_{ij} étant déterminés par le système d'équations

$$\left\{ \sum_{p=1}^{\min(i,j)} l_{ip} u_{pj} = a_{ij} \right\} \begin{array}{l} 1 \leq i \leq N \\ 1 \leq j \leq N \end{array}.$$

Ceci est un système de N^2 équations à $N^2 + N$ inconnues. On peut se fixer N éléments par exemple

$$l_{ii} = 1 \quad \forall i = 1, ..., N \quad \text{ou} \quad u_{ij} = 1 \quad \forall j = 1, ..., N.$$

Le système peut être alors résolu de proche en proche. Par exemple, avec le choix de $l_{ii} = 1$, on est conduit à :

$$\left\{ \begin{array}{l} l_{ii} = 1, i = 1, ..., N \\ \text{pour } r = 1, ..., N \\ \left\{ \begin{array}{ll} u_{rj} = a_{rj} - \sum_{k=1}^{r-1} l_{rk} u_{kj} & j = r, ..., N \\ l_{ir} = \dfrac{a_{ir} - \sum_{k=1}^{r-1} l_{ik} u_{kr}}{u_{rr}} & i = r+1, ..., N \end{array} \right. \end{array} \right. . \tag{6.9}$$

Ceci détermine la matrice cherchée

$$M = \begin{bmatrix} u_{11} & \cdots & \cdots & u_{1N} \\ l_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ l_{N1} & \cdots & \cdots & u_{NN} \end{bmatrix} \quad \text{dans l'ordre}$$

Remarque 6.8 Si les éléments a_{ij} ne sont pas nécessaires ultérieurement, on peut présenter l'algorithme 6.9 sous forme plus "compacte", en écrasant les éléments de A par ceux de M au fur et à mesure qu'ils ne sont plus utilisés, d'où :

Algorithme de Crout compact :

$$\left[\begin{array}{l} \text{De } i = 2 \text{ à } N \quad a_{i1} := \frac{a_{i1}}{a_{11}} \\ \left[\begin{array}{l} \text{De } r = 2 \text{ à } N \\ \left[\begin{array}{l} \text{de } j = r \text{ à } N \\ a_{rj} := a_{rj} - \sum_{k=1}^{r-1} a_{rk} a_{kj} \\ \text{De } i = r+1 \text{ à } N \\ \\ a_{ir} := \dfrac{a_{ir} - \sum_{k=1}^{r-1} a_{ik} a_{kr}}{a_{rr}} \end{array}\right. \end{array}\right. \end{array}\right. .$$

Remarque 6.9 Aucune stratégie du pivot n'étant appliquée, l'algorithme de Crout peut conduire à des u_{rr} nuls (ou petits) et donc à une méthode incorrecte. Il faut ainsi l'appliquer uniquement à des matrices dont la décomposition LU est assurée et pour lesquelles les $|u_{kk}|$ seront assez grands.

- Les algorithmes de Gauss et de Crout ne diffèrent que par l'ordre des opérations : ils sont par ailleurs équivalents comme on le vérifie à l'aide des formules (6.3) et (6.9) : "$u_{rj} = a_{rj} - \sum_{k=1}^{r-1} l_{rk} u_{kj}$" n'est pas autre chose que a_{rj}^r dans la notation (6.3). Dans la méthode de Gauss, on effectue et on mémorise individuellement les produits $l_{rk} u_{kj}$; dans la méthode de Crout, chaque produit scalaire est traité comme un tout ce qui peut être plus avantageux lorsqu'on dispose d'un calculateur accumulant les produits scalaires en double précision.
- Dans l'algorithme de Crout, le déterminant est obtenu par

$$\det A = u_{11} u_{22} ... u_{NN}.$$

6.2.2.3 Cas particulier des matrices tridiagonales

Théorème 6.1 *Soit*

$$A = \begin{bmatrix} b_1 & c_1 & 0 & \cdots & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & \ddots & & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & \ddots & \ddots & c_{N-1} \\ 0 & \cdots & \cdots & \cdots & 0 & a_N & b_N \end{bmatrix}$$

une matrice tridiagonale. On note A_k la matrice extraite de A en ne prenant que les k premières lignes et premières colonnes et on suppose $\det A_k \neq 0, \quad \forall k = 1, ..., N$. *Alors A admet la décomposition*

$$A = \begin{bmatrix} \frac{c_1}{z_1} & & & & \\ a_2 & \ddots & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & a_N & \frac{c_N}{z_N} \end{bmatrix} \begin{bmatrix} 1 & z_1 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \ddots & z_{N-1} \\ & & & & 1 \end{bmatrix} = LU$$

où

$$z_1 = \frac{c_1}{b_1}, \;\; z_k = \frac{c_k}{b_k - a_k z_{k-1}}, \;\; k = 2, 3, ..., N \;\; (c_N = 1 \textit{ par exemple}) \tag{6.10}$$

Le système $AX = b$ se résoud alors avec l'algorithme suivant :

$$w_1 = \frac{d_1}{b_1}, \;\; w_k = \frac{d_k - a_k w_{k-1}}{b_k - a_k z_{k-1}}, \;\; k = 2, 3, ..., N \;\; \text{(descente)} \tag{6.11}$$

$$x_{\mathrm{N}} = w_N, \quad x_k = w_k - z_k x_{k+1}, \;\; k = N-1, N-2, ..., 1 \;\; \text{(remontée)} \tag{6.12}$$

Remarque 6.10 Cette méthode de résolution requiert $3\,(N-1)$ additions (on ne calcule qu'une fois $b_k - a_k z_{k-1}$)
$3\,(N-1)$ multiplications
$2N$ divisions
soit $8N-6$ opérations au total, ce qui constitue une réduction considérable par rapport au cas d'une matrice pleine. Cette méthode directe est fortement conseillée pour la résolution d'un système tridiagonal.

Démonstration du théorème 6.1 : On vérifie directement que le produit des deux matrices ci-dessus redonne A ; la seule chose à prouver est que les nombres z_k sont bien définis. Or, on montre par récurrence que

$$z_k = c_k \frac{\det A_{k-1}}{\det A_k}, \;\; k = 2, ..., N \;\; (\text{par hypothèse } \det A_k \neq 0)\,.$$

En effet, développant $\det A_k$ suivant la dernière ligne, on vérifie que :

$$\det A_k = b_k \det A_{k-1} - a_k c_{k-1} \det A_{k-2}.$$

Ainsi, si $z_{k-1} = c_{k-1} \frac{\det A_{k-2}}{\det A_{k-1}}$, d'après la définition (6.10) de z_k, on a :

$$z_k = \frac{c_k}{b_k - a_k c_{k-1} \frac{\det A_{k-2}}{\det A_{k-1}}} = \frac{c_k \det A_{k-1}}{\det A_k} \text{ d'après (6.11).}$$

La relation se montre directement pour $k = 2$.
Le système (6.11) correspond à la résolution de $LW = d$; le système (6.12) correspond à celle de $UX = W$.

6.2.2.4 Méthode de Gauss-Jordan

Autre variante de la méthode de Gauss : méthode de Gauss-Jordan : au lieu de déterminer M pour que MA soit triangulaire, on détermine M pour que MA soit diagonale :

la résolution du système final est alors plus simple. Cependant, la diagonalisation demande plus de calculs et globalement le nombre d'opérations est sensiblement le même que dans la résolution de Gauss. Pour des raisons d'organisation des calculs, on emploie souvent la technique de Gauss-Jordan pour calculer l'inverse d'une matrice. Comme indiqué précédemment, on résoud alors simultanément les n systèmes linéaires

$$\{AX_i = e_i \ , i = 1, ..., N\}$$

où $e_i = (0, ..., 0, 1, 0, ..., 0)$ (le 1 est sur la ième coordonnée) et X_i est alors le i-ème vecteur colonne de A^{-1}.
Décrivons le passage de l'état p à l'état $p+1$: la matrice A_p a la structure suivante :

$$A_p = \begin{bmatrix} a_{11}^p & 0 & \cdots & 0 & a_{1p}^p & \cdots & a_{1N}^p \\ 0 & a_{22}^p & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots & & \vdots \\ \vdots & & & 0 & a_{pp}^p & \cdots & a_{pN}^p \\ \vdots & & & \vdots & \vdots & & \vdots \\ 0 & \cdots & \cdots & 0 & a_{Np}^p & \cdots & a_{NN}^p \end{bmatrix}$$

On utilise alors la p-ième ligne (c'est-à-dire la p-ième équation) pour faire apparaître des zéros dans toute la colonne de rang p, sauf en a_{pp}^p (c'est-à-dire éliminer l'inconnue x_p dans toutes les autres équations). Comme dans la méthode de Gauss, on utilise donc a_{pp}^p comme pivot (après application éventuelle d'une stratégie), et si $l_{ip} = a_{ip}^p / a_{pp}^p$, A_{p+1} est obtenu par :

$$\begin{cases} a_{ij}^{p+1} = a_{ij}^p - l_{ip} a_{pj}^p \\ i = 1, ..., N, \ \ i \neq p \\ p+1 \leq j \leq N \end{cases} . \tag{6.13}$$

On fait bien sûr les mêmes transformations aux seconds membres que, de manière pratique, on assimile à des colonnes supplémentaires de A ; les formules (6.13) sont alors appliquées à la matrice augmentée (j varie de $p+1$ à $N+k$, si k est le nombre de seconds membres)

6.2.3 Factorisation de Cholesky

Ce paragraphe traite du cas particulier des matrices symétriques définies positives. Pour de telles matrices, la méthode de Gauss sans stratégie du pivot s'applique toujours. En effet, on remarque d'abord que les sous-matrices principales $A_{(k)}$ sont inversibles :

$$A = \left[\begin{array}{c|c} A_{(k)} & \\ \hline & \end{array} \right].$$

Pour cela, soit $v \in \mathbb{R}^k$ tel que $A_{(k)} v = 0$; soit $\widetilde{v} \in \mathbb{R}^N$ défini par

$$\widetilde{v_i} = v_i, \ \ 1 \leq i \leq k \qquad \widetilde{v_i} = 0, \ \ k+1 \leq i \leq N.$$

On a

$${}^t v A_{(k)} v = {}^t \widetilde{v} A v = 0 \Longrightarrow v = 0,$$

puisque A est définie positive.

Maintenant, la condition $\det A_{(k)} \neq 0 \ \forall k$ est suffisante pour assurer que les pivots sont non nuls dans la factorisation de Gauss. En effet, d'après les règles de calcul des déterminants, si A^p est la matrice à l'étape p, on a :

$$\det A^p_{(p)} = \det A_{(p)}$$

où $A^p_{(p)}$ est la sous-matrice principale de A^p d'ordre p.

$$A^p_{(p)} = \begin{bmatrix} a^p_{11} & 0 & \cdots & 0 & a^p_{1p} \\ 0 & a^p_{22} & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ & & \ddots & \ddots & \vdots \\ 0 & & \cdots & 0 & a^p_{pp} \end{bmatrix}$$

Or $\det A^p_{(p)} = a^p_{11}...a^p_{pp}$. Ce qui montre que $a^p_{pp} \neq 0$. Par récurrence, on voit donc qu'on est toujours conduit à des pivots non nuls. On en déduit :

Théorème 6.2 (Factorisation de Choleski) *Soit A une matrice symétrique définie positive. Alors il existe B triangulaire inférieure telle que :*

$$A = B^t B.$$

Remarque 6.11 On obtient bien sûr une décomposition de type LU, mais il n'y a qu'une matrice à déterminer. Cette matrice est parfois appelée racine carrée de A. On peut imposer que ses éléments diagonaux soient strictement positifs. La factorisation est alors unique (le montrer).
Dans la pratique, on utilise un algorithme d'identification (dit de Cholesky) pour calculer les éléments de B, plutôt que la méthode de Gauss. Celle-ci nous permet cependant de montrer l'existence de B.

Démonstration du Théorème 6.2 :
Comme montré ci-dessus, l'algorithme de Gauss sans stratégie de pivot peut être mené à terme et donne donc une factorisation de A en $A = LU$, soit

$$\begin{bmatrix} 1 & & & & & & \\ l_{21} & 1 & & & & & \\ \vdots & & \ddots & & & & \\ \vdots & & & \ddots & & & \\ \vdots & & & & \ddots & & \\ \vdots & & & & & \ddots & \\ l_{\mathrm{N}1} & \cdots & \cdots & \cdots & \cdots & l_{\mathrm{N},\mathrm{N}-1} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \ddots & & & & & \\ \vdots & \ddots & \ddots & & & & \\ \vdots & & \ddots & \ddots & & & \\ \vdots & & & \ddots & u_{kk} & \cdots & \cdots \\ \vdots & & & & \ddots & \ddots & \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & u_{\mathrm{NN}} \end{bmatrix}.$$

Les u_{kk} sont strictement positifs car

$$u_{11}...u_{kk} = \det A_k > 0,$$

ce dernier point résultant du fait que A_k est symétrique définie posisive. On intercale la matrice diagonale $\Lambda = diag\left(\sqrt{u_{ii}}\right)$, soit :

$$A = (L\Lambda)\left(\Lambda^{-1}U\right) = BC.$$

Comme $A = {}^tA,\ BC = {}^tC{}^tB \Longrightarrow C\,({}^tB)^{-1} = B^{-1}{}^tC$.
Puisque $C\,({}^tB)^{-1}$ est triangulaire supérieure et $B^{-1}{}^tC$ triangulaire inférieure, l'égalité prouve qu'elles sont diagonales. Comme les éléments diagonaux sont égaux à 1, ceci prouve $B^{-1}{}^tC = I$ ou $C = {}^tB$.

Méthode pratique de calcul de B : algorithme de Cholesky
On pose a priori

$$B = \begin{bmatrix} b_{11} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ b_{N1} & \cdots & \cdots & b_{NN} \end{bmatrix}$$

et on détermine $\{b_{ij}\}$ à l'aide du système d'équations

$$\left\{ \sum_{k=1}^{i} b_{ik} b_{jk} = a_{ij},\ \ 1 \le i \le j \le N \right\} \tag{6.14}$$

Il y a $\frac{N(N+1)}{2}$ inconnues pour $\frac{N(N+1)}{2}$ équations. La résolution peut se faire de proche en proche, colonne par colonne, comme suit :

Algorithme de Cholesky

$$\left[\begin{array}{c} \text{De } i = 1 \text{ à } N \\ b_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} b_{ik}^2} \\ \left[\begin{array}{c} \text{De } j = i+1 \text{ à } N \\ b_{ji} = \left(a_{ij} - \sum_{k=1}^{i-1} b_{ik} b_{jk} \right) / b_{ii} \end{array} \right. \end{array} \right. . \tag{6.15}$$

Remarque 6.12 – A priori il n'est pas clair que le calcul des racines carrées ci-dessus soit possible dans l'ensemble des réels : cependant, comme nous avons montré directement l'existence de B, nous savons que le système (6.14) admet une solution et qu'ainsi le calcul sera toujours possible. On obtient au passage l'unicité de la solution.
– Le déterminant de A est obtenu comme sous-produit par

$$\det A = b_{11}^2 b_{22}^2 ... b_{NN}^2$$

puisque $\det A = (\det B)^2$.
– Une évaluation du nombre d'opérations conduit à
$\frac{N^3}{6}$ additions, $\frac{N^3}{6}$ multiplications, $\frac{N^2}{2}$ divisions, N extractions de racines carrées. Ceci se compare très avantageusement à la méthode de Gauss donnant $\frac{N^3}{3}$ additions, $\frac{N^3}{3}$ multiplications, $\frac{N^2}{2}$ divisions.

La méthode de Cholesky est donc fortement conseillée (plutôt que Gauss) quand la matrice du système est symétrique, définie, positive. Elle a aussi l'avantage de se révéler plus stable par rapport aux erreurs d'arrondi. Noter également l'estimation a priori

$$|b_{ik}| < \sqrt{a_{ii}}\ \ \forall i = 1, ..., N,\ \ \forall k = 1, ..., i$$

provenant de la relation

$$\sum_{k=1}^{i} b_{ik}^2 = a_{ii}$$

et qui assure que les b_{ik} ne deviendront jamais trop grands.

6.2.4 Autres méthodes directes

– Une autre méthode, connue sous le nom de méthode de Householder, consiste à multiplier la matrice A par des matrices de symétries orthogonale pour aboutir à une factorisation de la forme

$$A = QR$$

où R est triangulaire supérieure et Q une matrice orthogonale. Cette technique nécessite environ deux fois plus d'opérations que celle de Gauss mais le conditionnement du système initial ne peut qu'être amélioré. Cette factorisation sera décrite dans le chapitre sur le calcul des valeurs propres : elle constitue le fondement d'un des plus importants algorithmes de recherche des valeurs propres.

– Signalons aussi la méthode du gradient conjugué qui est à l'heure actuelle de plus en plus utilisée pour la résolution des systèmes creux. Elle repose sur la minimisation de la fonctionnelle quadratique $\Phi(x) = \frac{1}{2}{}^tXAX - bX$ et sera décrite à la fin de ce chapitre. c'est une méthode "semi-directe".

6.3 Analyse du conditionnement d'un système linéaire

Le but de ce paragraphe est d'étudier l'influence d'une petite variation du second membre ou des coefficients d'une matrice d'un système linéaire sur la valeur de la solution. Comme il a déjà été expliqué dans le premier chapitre de ce cours, les arrondis faits sur les données (et inévitables dans un calcul numérique sur machine), peuvent être interprétés comme de telles "petites variations" : il est donc essentiel de connaitre leur impact sur le résultat calculé.

6.3.1 Quelques préliminaires sur les normes matricielles

On rappelle tout d'abord :

Définition 6.1 *Etant donné V un espace vectoriel sur $\mathbb{R}$ ou $\mathbb{C}$, on appelle norme sur V toute application (qu'on notera $\|.\|$) de V dans $[0,\infty[$ telle que*

(i) $\|x\| = 0 \Longleftrightarrow x = 0$ *pour* $x \in V$

(ii) $\forall x \in V, \forall \lambda \in \mathbb{R}$ (*ou* $\mathbb{C}$), $\quad \|\lambda x\| = |\lambda| \, \|x\|$ *(homogénéité)*

(iii) $\forall x, y \in V, \quad \|x+y\| \leq \|x\| + \|y\|$ *(inégalité triangulaire).*

Nous serons, en particulier, amenés à utiliser les normes classiques de $\mathbb{R}^N$ (ou $\mathbb{C}^N$) à savoir :

$$\begin{aligned} \|x\|_2 &= \left(\sum_{i=1}^{N} x_i^2 \right)^{1/2} \\ \|x\|_1 &= \sum_{i=1}^{N} |x_i| \\ \|x\|_\infty &= \max_{1 \leq i \leq N} |x_i| \, . \end{aligned}$$

Plus généralement, on peut considérer

$$\|x\|_p = \left(\sum_{i=1}^{N} |x_i|^p \right)^{1/p}, \quad 1 \leq p < \infty.$$

La notation $\|x\|_\infty$ utilisée ci-dessus provient du fait que

$$\forall x \in \mathbb{R}^N, \lim_{p\to\infty} \|x\|_p = \max_{1\leq i\leq N} |x_i| \ \ (= \|x\|_\infty) .$$

Il est bon de connaitre la géométrie des boules-unités pour chacune de ces normes soit $B = \left\{x \in \mathbb{R}^N; \|x\| \leq 1\right\}$. Ainsi, en dimension 2, on a : On remarque que la boule-

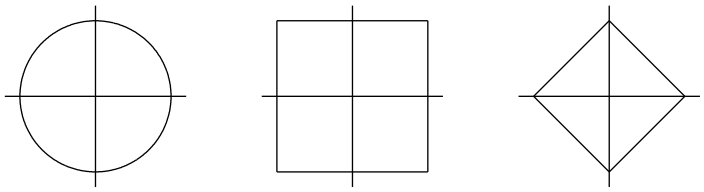

FIGURE 6.1 – Les boules unités, pour la norme $\|.\|_2$ (à gauche), pour la norme $\|.\|_\infty$ (au centre), pour la norme $\|.\|_1$ (à droite)

unité pour la norme euclidienne est "arrondie" : elle ne présente ni angle, ni partie plate contrairement aux deux autres. Cette géométrie particulière explique partiellement pourquoi la norme euclidienne a de "meilleurs" comportements que les autres.

6.3.1.1 Norme sur l'espace vectoriel des matrices

Une matrice carrée d'ordre N est la donnée de N^2 réels (ou complexes) : elle peut donc être identifiée à un élément de $\mathbb{R}^{N^2}$. A ce titre, toute norme sur $\mathbb{R}^{N^2}$ fournit une norme sur l'espace des matrices. Ainsi, si $A = [a_{ij}]$, on peut poser

$$\|A\| = \max_{1\leq i,j\leq N} |a_{ij}| \ \text{ ou } \ \|A\| = \left(\sum_{i,j} a_{ij}^2\right)^{1/2} \text{, etc...}$$

Cependant, pour qu'une norme soit pratique d'utilisation, il faut aussi qu'elle soit compatible avec le produit des matrices, ce qui nous conduit à exiger une quatrième propriété :

$$(iv) \qquad \|AB\| \leq \|A\| \, \|B\| .$$

Définition 6.2 *On appelle norme matricielle toute norme sur l'espace vectoriel des matrices vérifiant* (iv).

Etant donnée une norme vectorielle $\|.\|$ définie sur $\mathbb{R}^N$, il existe une norme matricielle naturellement associée à $\|.\|$: on l'appelle norme matricielle induite par $\|.\|$ et elle est définie par :

$$\|A\| = \max_{\|x\|\neq 0} \frac{\|Ax\|}{\|x\|} \left(= \max_{\|x\|=1} \|Ax\|\right) .$$

Remarque 6.13 – L'égalité entre ces deux expressions est due à la linéarité de A soit $\frac{Ax}{\|x\|} = A\left(\frac{x}{\|x\|}\right)$ et $\left\|\frac{x}{\|x\|}\right\| = 1$. La deuxième expression permet de montrer l'existence du maximum en remarquant que $x \to \|Ax\|$ est continue sur la sphère unité de $\mathbb{R}^N$ qui est compacte.

– On vérifie aisément que l'application $A \to \|A\|$ est bien une norme matricielle. Vérifions par exemple (iv) :

$$\|AB\| = \max_{x \neq 0} \frac{\|ABx\|}{\|x\|}$$

Puisque $\|ABx\| \leq \|A\| \, \|Bx\|$ par définition de $\|A\|$, on a donc

$$\|AB\| \leq \max_{x \neq 0} \|A\| \, \frac{\|Bx\|}{\|x\|} \leq \|A\| \, \|B\| \, .$$

– On garde souvent la même notation pour la norme vectorielle et la norme matricielle induite bien que les deux normes n'agissent pas sur le même espace : cela ne conduit en fait à aucune confusion et favorise au contraire les écritures. Ainsi, on écrira :

$$\|AX\| \leq \|A\| \, \|X\| \, , \, \forall X \in \mathbb{R}^N, \, \forall A \text{ matrice carrée d'ordre } N.$$

– On notera de la même façon $\|.\|_p$ la norme matricielle associée à $\|.\|_p$.

Proposition 6.1

$$\begin{aligned} \textit{(i)} \ \|A\|_\infty &= \max_{1 \leq i \leq N} \left(\sum_{j=1}^{N} |a_{ij}| \right) \\ \textit{(ii)} \ \|A\|_1 &= \max_{1 \leq j \leq N} \left(\sum_{i=1}^{N} |a_{ij}| \right) \\ \textit{(iii)} \ \|A\|_2 &= \sqrt{\rho\left(A^* A\right)} \ \textit{où} \ A^* = {}^t\overline{A} \ \textit{(adjointe de } A\textit{)} \end{aligned}$$

et $\rho(M)$ désigne le rayon spectral de la matrice M soit

$$\rho(M) := \max \{ |\lambda| \, ; \lambda \textit{ valeur propre de } M \} \, .$$

Dans le cas particulier où A est symétrique,$\|A\|_2 = \rho(A)$.

Remarque 6.14 – Toute racine carrée d'une valeur propre de A^*A est appelée valeur singulière de A. Si A est normale (i.e. $AA^* = A^*A$), alors $\sqrt{\rho\left(A^* A\right)} = \rho\left(A\right)$. Dans tous les cas $\rho\left(A^* A\right) = \rho\left(A A^*\right)$.
– $\|A\|_\infty$ et $\|A\|_1$ se calculent simplement à partir de la matrice de A contrairement à $\|A\|_2$.
– On utilise aussi la norme, dite de Fröbenius (ou norme de Schur) :

$$\|A\|_F = \left(\sum_{i,j} a_{ij}^2 \right)^{1/2} = \left[trace \left(A^* A \right) \right]^{1/2} \, .$$

On montre qu'il s'agit d'une norme matricielle, mais non induite par une norme vectorielle (en effet, si I est l'identité, $\|I\|_F = \sqrt{N}$ ... au lieu de 1 pour une norme induite par une norme vectorielle), voir aussi Exercice **??**.

Démonstration de la proposition 6.1 :

$$\begin{aligned} \|Ax\|_\infty &= \max_{1 \leq i \leq N} \left\{ \left| \sum_{j=1}^{N} a_{ij} x_j \right| \right\} \leq \max_{1 \leq i \leq N} \left\{ \sum_{j=1}^{N} |a_{ij}| \, |x_j| \right\} \\ &\leq \max_{1 \leq i \leq N} \left\{ \left(\sum_{j=1}^{N} |a_{ij}| \right) \left(\max_{1 \leq j \leq N} |x_j| \right) \right\} \leq \|x\|_\infty \max_{1 \leq i \leq N} \left(\sum_{j=1} |a_{ij}| \right) . \end{aligned}$$

Ceci démontre que $\|A\| \leq \max_{1\leq i\leq N}\left(\sum_{j=1}^{N}|a_{ij}|\right)$. On montre que l'égalité est réalisée en considérant le vecteur $x=(\varepsilon_j)$ où

$$\varepsilon_j = \begin{cases} 1 \text{ si } a_{i_0 j} \geq 0 \\ -1 \text{ sinon } a_{i_0 j} < 0 \end{cases} \quad \text{et } i_0 \text{ est un indice tel que}$$

$$\sum_{j=1}^{N}|a_{i_0 j}| = \max_{1\leq i\leq N}\left\{\sum_{j=1}^{N}|a_{ij}|\right\}$$

– Le point (ii) s'obtient de façon analogue : noter la dualité entre les normes $\|.\|_1$ et $\|.\|_\infty$ qui va, en fait, bien au delà de cette simple remarque.
– On a $\|A\|_2^2 = \max_{\|v\|_2=1}\|Av\|_2^2 = \max_{\|v\|_2=1} v^*A^*Av$, puisque $\|x\|_2^2 = x^*x = \sum|x_i|^2$. La matrice A^*A est hermitienne puisque $(A^*A)^* = A^*A$.

On sait (cf. cours d'algèbre linéaire classique) qu'il existe une matrice unitaire U telle que
$U^{-1}A^*AU = D$ soit diagonale, la diagonale étant formée des valeurs propres λ de A^*A.
On a donc :

$$\max_{\|v\|_2=1} v^*A^*Av = \max_{\|v\|_2=1} v^*UDU^{-1}v.$$

Puisque U est unitaire, $\left\|U^{-1}v\right\|_2 = 1$. Puisque par ailleurs U^{-1} est bijective, on a donc :

$$\|A\|_2^2 = \max_{\|w\|_2=1} w^*Dw = \max_{\sum_{i=1}^{N}w_i^2=1}\left(\sum_{i=1}^{N}\lambda_i w_i^2\right) = \max_{i=1,\ldots,N}|\lambda_i| = \rho(D) = \rho(A^*A).$$

6.3.2 Analyse du conditionnement

Considérons le système linéaire

$$AX = b. \tag{6.16}$$

Considérons une petite perturbation δb de b. La solution correspondante est perturbée soit

$$A(X+\delta X) = b+\delta b. \tag{6.17}$$

Nous allons mesurer la variation relative de X en fonction de celle de b. Pour cela, nous prenons $\|.\|$ une norme quelconque sur $\mathbb{R}^N$.
De (6.16), (6.17), on tire

$$A(\delta X) = \delta b \text{ soit } \delta X = A^{-1}(\delta b).$$

Utilisant la norme matricielle induite, on a :

$$\|\delta X\| \leq \left\|A^{-1}\right\|\|\delta b\|.$$

Par ailleurs, avec (6.16)

$$\|b\| \leq \|A\|\|X\|.$$

Ces deux inégalités donnent :

$$\frac{\|\delta X\|}{\|X\|} \leq \left\|A^{-1}\right\|\|A\|\frac{\|\delta b\|}{\|b\|}. \tag{6.18}$$

Cette estimation sur la variation de X est en fait optimale puisqu'on peut trouver X_0 tel que $\|AX_0\| = \|A\|\|X_0\|$ et δb tel que $\left\|A^{-1}\right\|\|\delta b\| = \left\|A^{-1}(\delta b)\right\|$.
Ainsi la variation relative sur X sera d'autant plus grande que le nombre $\left\|A^{-1}\right\|\|A\|$ est plus grand : on l'appelle le conditionnement de A relatif à la norme $\|.\|$.

Définition 6.3 *On appelle conditionnement de la matrice A (dans la norme matricielle $\|.\|$), le nombre*

$$cond\,(A) = \left\|A^{-1}\right\| \|A\|\,.$$

Si on travaille avec la norme $\|.\|_p$, on notera le conditionnement associé $cond_p$.

C'est le même nombre qui intervient lors de la variation des coefficients de A : supposons, en effet, que A soit perturbée par une matrice ΔA ; alors

$$\begin{aligned} AX &= b \\ (A+\Delta A)(X+\delta X) &= b \\ &\Longrightarrow A(\delta X) + \Delta A(X+\delta X) = 0 \\ &\Longrightarrow \delta X = -A^{-1}\Delta A(X+\delta X) \\ &\Longrightarrow \|\delta X\| \leq \left\|A^{-1}\right\| \|\Delta A\| \|X+\delta X\| \\ &\Longrightarrow \frac{\|\delta X\|}{\|X+\delta X\|} \leq cond\,(A)\frac{\|\Delta A\|}{\|A\|}. \end{aligned}$$

6.3.2.1 Propriétés de $cond\,(A)$

(i) $cond\,(A) \geq 1$ et le conditionnement est d'autant meilleur qu'il est plus proche de 1. (en effet $AA^{-1} = I \Longrightarrow 1 = \|I\| \leq \|A\| \left\|A^{-1}\right\|$.)

(ii) $cond\,(A) = cond\,(A^{-1})$, $cond\,(\alpha A) = cond\,(A)$.

(iii) si A est normale (i.e. $A^*A = AA^*$), pour la norme $\|.\|_2$, on a

$$cond_2(A) = \frac{\max_i |\lambda_i(A)|}{\min_i |\lambda_i(A)|},\ \lambda_i \text{ valeur propre de } A.$$

Plus généralement, pour une matrice A quelconque

$$cond_2(A) = \frac{\mu_N(A)}{\mu_1(A)},$$

$$\begin{aligned} \mu_N &= \text{plus grande valeur singulière de } A, \\ \mu_1 &= \text{plus petite valeur singulière de } A. \end{aligned}$$

(iv) si A est unitaire (ie $U^* = U^{-1}$), $cond_2(A) = 1$.

Remarque 6.15 – La démonstration de (iii) est immédiate à partir de la définition du conditionnement, de la proposition 6.1 et des remarques qui la suivent.
– La propriété (iii) exprime qu'une matrice dont le spectre (i.e. l'ensemble des valeurs propres) est étendu sera mal conditionnée.
– La propriété (iv) justifie l'emploi de matrices unitaires dans certains procédés de factorisation (cf. méthode de Householder) : ainsi le conditionnement du système final est au moins aussi bon que celui du système initial.
– On ne modifie pas le conditionnement d'une matrice en la multipliant par un scalaire. Par contre, on peut diminuer $cond_2 A$ en multipliant certaines lignes ou certaines colonnes par des coefficients non nuls : il s'agit de l'équilibrage de la matrice. C'est une technique de préconditionnement très utilisée en pratique.

6.3.3 Exemple de système linéaire mal conditionné

(dû à R.S. Wilson, cf. P.G. Ciarlet "Introduction à l'analyse numérique matricielle et à l'optimisation")
Considérons le système

$$\begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 32 \\ 23 \\ 33 \\ 31 \end{bmatrix} \text{ de solution } \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

Le système perturbé

$$\begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix} \begin{bmatrix} u_1 + \delta u_1 \\ u_2 + \delta u_2 \\ u_3 + \delta u_3 \\ u_4 + \delta u_4 \end{bmatrix} = \begin{bmatrix} 32,1 \\ 22,9 \\ 33,1 \\ 30,9 \end{bmatrix} \text{ a pour solution } \begin{bmatrix} 9,2 \\ -12,6 \\ 4,5 \\ -1,1 \end{bmatrix}.$$

Ainsi une erreur relative de 1/200 sur les données entraîne une erreur relative de l'ordre de 10/1 du résultat (erreur amplifiée de 2000).
De même

$$\begin{bmatrix} 10 & 7 & 8,1 & 7,2 \\ 7,08 & 5,04 & 6 & 5 \\ 8 & 5,98 & 9,89 & 9 \\ 6,99 & 4,99 & 9 & 9,98 \end{bmatrix} \begin{bmatrix} u_1 + \Delta u_1 \\ u_2 + \Delta u_2 \\ u_3 + \Delta u_3 \\ u_4 + \Delta u_4 \end{bmatrix} = \begin{bmatrix} 32 \\ 23 \\ 33 \\ 31 \end{bmatrix} \text{ a pour solution } \begin{bmatrix} -81 \\ 137 \\ -34 \\ 22 \end{bmatrix}.$$

Pourtant, la matrice est "bonne" (symétrique, de déterminant 1, donc loin de 0). Son inverse est d'ailleurs donnée par

$$A^{-1} = \begin{bmatrix} 25 & -41 & 10 & -6 \\ -41 & 68 & -17 & 10 \\ 10 & -17 & 5 & -3 \\ -6 & 10 & -3 & 2 \end{bmatrix}.$$

Mais les valeurs propres de A sont

$$\lambda_1 \approx 0,01015 < \lambda_2 \approx 0,8431 < \lambda_3 \approx 3,858 < \lambda_4 \approx 30,2877, \text{ si bien que}$$

$$cond_2(A) = \frac{\lambda_4}{\lambda_1} \approx 2984 \text{ est grand !}$$

D'autre part

$$u = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \delta u = \begin{bmatrix} 8,2 \\ -13,6 \\ 3,5 \\ -2,1 \end{bmatrix}, b = \begin{bmatrix} 32 \\ 23 \\ 33 \\ 31 \end{bmatrix}, \delta b = \begin{bmatrix} 0,1 \\ -0,1 \\ 0,1 \\ -0,1 \end{bmatrix}$$

de sorte que

$$\frac{\|\delta u\|_2}{\|u\|_2} \simeq 8,1985 \text{ et } cond_2(A) \frac{\|\delta b\|_2}{\|b\|_2} \simeq 9,9424.$$

On n'est donc pas loin de l'égalité dans l'estimation (6.18)

$$\frac{\|\delta u\|_2}{\|u\|_2} \leq cond_2(A) \frac{\|\delta b\|_2}{\|b\|_2}.$$

6.3.4 Préconditionnement

Les matrices mal conditionnées sont véritablement à l'origine d'erreurs importantes dans les calculs pratiques, y compris pour les ingénieurs. Par exemple, dans la résolution d'une équation aux dérivées partielles avec une discrétisation de type différences finies ou éléments finis de pas h (très petit), il est classique que le conditionnement de la matrice du système linéaire soit en $O(\frac{1}{h^2})$! Par ailleurs pour les méthodes itératives comme la méthode du gradient conjugué, cf section 6.5, la rapidité de convergence est directement liée au conditionnement : plus celui-ci est mauvais, plus la convergence sera lente. Il est donc souvent utile de remplacer le système linéaire initial par un système équivalent *mieux conditionné*, c'est ce qu'on appelle la technique du préconditionnement. Il y a de nombreuses façons de le faire. Citons simplement ici sans détailler les méthodes de factorisation incomplète ou le préconditionnement SSOR.

6.4 Méthodes itératives

Etant donné le système $AX = B$ à résoudre, les méthodes itératives de résolution de systèmes linéaires consistent à calculer les valeurs successives d'une suite de vecteurs X^k convergeant vers la solution X quand $k \to \infty$.

Principe : On décompose A en $A = M - N$ où M est une matrice "facile" à inverser, au sens que le système $MY = d$ se résout facilement (M diagonale, ou triangulaire, ou diagonale par blocs...). Alors

$$AX = b \Longleftrightarrow MX = NX + b$$

conduit à l'itération

$$MX^{k+1} = NX^k + b.$$

6.4.1 Description des méthodes de Jacobi, Gauss-Seidel, relaxation

On suppose donnée A avec $a_{ii} \neq 0 \ \forall i$. On décompose A sous la forme

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & \cdots & a_{1\mathrm{N}} \\ a_{21} & a_{22} & & & -F & \\ & & & D & & \\ & -E & & & & \\ a_{\mathrm{N}1} & \cdots & \cdots & \cdots & a_{\mathrm{N},\mathrm{N}-1} & a_{\mathrm{NN}} \end{bmatrix} = D - E - F$$

où D est diagonale, E triangulaire inférieure, F triangulaire supérieure.
Une première décomposition conduit à

$$AX = b \Longleftrightarrow DX = (E + F)\, X + b$$

et à la méthode itérative de Jacobi :

$$\begin{cases} X^{k+1} = D^{-1}\,(E + F)\, X^k + D^{-1} b \\ X^0 \text{ arbitraire} \end{cases} \tag{6.19}$$

Algorithme de Jacobi

$$\left[\begin{array}{c} X^0 \text{ choisi} \\ \text{De } k = 0 \text{ à } \ldots \text{ (test d'arrêt)} \\ \left[\begin{array}{c} \text{De } i = 1 \text{ à } N \\ x_i^{k+1} := \left(-\sum_{p \neq i} a_{ip} x_p^k + b_i \right) / a_{ii}. \end{array} \right. \end{array} \right.$$

Remarque 6.16 Dans cet algorithme, il est nécessaire de conserver en mémoire tous les x_i^k aussi longtemps que le calcul des x_i^{k+1} n'est pas achevé. Il est possible d'utiliser 2 fois moins de mémoires en écrasant à chaque fois x_i^k par x_i^{k+1}, ce qui conduit à :

Algorithme de Gauss-Seidel

$$\left[\begin{array}{c} X^0 \text{ choisi} \\ \text{De } k=0 \text{ à } ...\text{(test d'arrêt)} \\ \left[\begin{array}{c} \text{De } i=1 \text{ à } N \\ x_i^{k+1} := \left(-\sum_{p<i} a_{ip}x_p^{k+1} - \sum_{p>i} a_{ip}x_p^k + b_i\right)/a_{ii}. \end{array}\right. \end{array}\right.$$

Remarque 6.17 Sous forme matricielle, cet algorithme s'écrit en fait :

$$DX^{k+1} = EX^{k+1} + FX^k + b$$

ou

$$X^{k+1} = (D-E)^{-1}\left(FX^k + b\right).$$

Cet algorithme peut encore être modifié en introduisant un paramètre supplémentaire qu'on choisira au mieux pour que la convergence soit plus rapide. Selon les notations traditionnelles, on est conduit à :

$$X^{k+1} = (1-\omega)\,X^k + \omega\left[(D-E)^{-1}\left(FX^k + b\right)\right].$$

On vérifie immédiatement que si X^k converge, il converge vers X solution de

$$X = (1-\omega)\,X + \omega\left[(D-E)^{-1}(FX+b)\right] \Longleftrightarrow X = (D-E)^{-1}(FX+b) \Longleftrightarrow AX = b.$$

Algorithme de relaxation

$$\left[\begin{array}{c} X^0 \text{ choisi} \\ \text{De } k=0 \text{ à } ...\text{(test d'arrêt)} \\ \left[\begin{array}{c} \text{De } i=1 \text{ à } N \\ x_i^{k+1} := (1-\omega)\,x_i^k - \omega\left(\sum_{p<i} a_{ip}x_p^{k+1} + \sum_{p>i} a_{ip}x_p^k - b_i\right)/a_{ii}. \end{array}\right. \end{array}\right.$$

Remarque 6.18 Il faut noter que chaque itération nécessite (dans Gauss-Seidel par exemple) : $N\left((N-1)\right.$ multiplications $+(N-1)$ additions $\left.+1 \text{ division}\right) \approx 2N^2$ opérations, si la matrice est pleine. Pour être comparable à la méthode de Gauss, il faut donc que la précision voulue soit obtenue en moins de $\frac{N}{3}$ itérations. Dans la pratique, on constate que les méthodes itératives sont surtout avantageuses lorsque la matrice est creuse : à chaque itération, le nombre d'opérations est d'ordre kN où k est une constante fixe directement proportionnelle au nombre d'éléments non nuls. De plus, il suffit de stocker les éléments non nuls de A. C'est ce dernier point qui est essentiel dans le choix des méthodes itératives pour les grands systèmes creux : une factorisation de type Gauss ou Cholesky, même si elle préserve la structure-bande, remplit les "trous" à l'intérieur de la bande et nécessite donc plus d'espace mémoire.

6.4.2 Itérations par blocs

Supposons que la matrice A se présente naturellement sous la forme

$$A = \left[\begin{array}{ccccc} A_{11} & & & & A_{15} \\ & A_{22} & & & \\ & & A_{33} & & \\ & & & A_{44} & \\ A_{51} & & & & A_{55} \end{array}\right]$$

où les matrices A_{ii} sont inversibles. On peut alors utiliser un algorithme de relaxation par blocs : si $X^k = \left(X_1^k, X_2^k, X_3^k, X_4^k, X_5^k\right)$ où les X_i^k ont des longueurs adaptées à la décomposition ci-dessus, on a :

$$\left[\begin{array}{c} \text{Pour } i = 1, ..., 5 \\ A_{ii}X_i^{k+1} = (1-\omega)\, A_{ii}X_i^k - \omega\left[\sum_{p<i} A_{ip}X_p^{k+1} + \sum_{i<p<5} A_{ip}X_p^k - B_i\right] \end{array}\right. .$$

Ce regroupement par blocs peut accélérer la convergence, mais alourdit bien sûr le coût de chaque itération puisqu'il y a, à chaque pas, 5 sous-systèmes à résoudre. Ce n'est que si cette résolution est rapide que ce choix peut être intéressant.

6.4.3 Etude de la convergence des méthodes itératives

Considérons une méthode itérative du type

$$X^{k+1} = BX^k + d \tag{6.20}$$

appliquée à la résolution de $AX = b$. (On suppose A inversible).

Définition 6.4 *On dit que la méthode est consistante si, lorsque X^k converge vers Y, alors Y est la solution cherchée soit $Y = A^{-1}b$.*
On dit que la méthode est convergente si, pour tout choix X^0 de la donnée initiale, X^k converge vers $A^{-1}b$.

Remarque 6.19 Pour qu'une méthode soit consistante, il faut bien sûr choisir B et d en fonction de A et B convenablement : supposons que $X^k \to_{k\to\infty} Y$.
On a alors

$$\begin{aligned} Y &= BY + d \\ X &= A^{-1}b \\ Y - X &= B\,(Y - X) + d - (I - B)\, A^{-1}b. \end{aligned}$$

La consistance est assurée si $d = (I - B)\, A^{-1}b$ et $I - B$ est inversible.

Théorème 6.3 *Supposons la méthode (6.20) consistante. Alors elle est convergente si et seulement si*

$$\rho\,(B) < 1. \qquad (\rho\,(B) = \textit{rayon spectral de } B)\,. \tag{6.21}$$

Démonstration : Soit X la solution cherchée, i.e. (puisque la méthode est consistante) $X = BX + d$. Retranchant à (6.20), on obtient :

$$X^{k+1} - X = B\left(X^k - X\right)$$

et par récurrence

$$X^k - X = B^k\left(X^0 - X\right). \tag{6.22}$$

Puisque $X^0 - X$ est un vecteur arbitraire, on voit que $X^k - X$ tend vers 0 (pour tout choix de X^0) si et seulement si

$$\lim_{k\to\infty} B^k = 0 \tag{6.23}$$

(au sens que tous les éléments de la matrice B^k tendent vers 0).
Supposons que $\rho\,(B) \geq 1$: alors il existe $\lambda \in \mathbb{C}$ et $X \in \mathbb{C}^N$ tel que $BX = \lambda X$, $|\lambda| \geq 1$, $X \neq 0$ et donc

$$B^k X = \lambda^k X.$$

Mais $\left|\lambda^k\right| \nrightarrow 0$ quand $k \to \infty$. Donc B^k ne tend pas vers 0. La condition $\rho(B) < 1$ est donc nécessaire.
Supposons que $\rho(B) < 1$: Si $\|.\|$ est une norme sur $\mathbb{R}^N$ (ou $\mathbb{C}^N$), d'après (6.22), on a :

$$\left\|X^k - X\right\| \leq \|B\|^k \left\|X^0 - X\right\| \tag{6.24}$$

où $\|B\|$ est la norme matricielle induite de B. S'il existe une norme telle que $\|B\| < 1$, on a immédiatement

$$\lim_{k\to\infty} \left\|X^k - X\right\| = 0 \text{ pour tout choix de } X^0.$$

L'existence de cette norme résulte du lemme suivant :

Lemme 6.1 *Soit B une matrice carrée :*
(i) $\rho(B) \leq \|B\|$ pour toute norme matricielle $\|.\|$.
(ii) $\forall \varepsilon > 0, \exists \|.\|$ norme sur $\mathbb{R}^N$ telle que pour la norme matricielle induite $\|B\| \leq \rho(B) + \varepsilon$.

Démonstration : Le point (i) s'obtient comme suit : soit (λ, X) une valeur propre et un vecteur propre de B.

$$BX = \lambda X \Longrightarrow |\lambda| \|X\| \leq \|B\| \|X\| \Longrightarrow |\lambda| \leq \|B\|.$$

Puisque c'est vrai pour tout λ valeur propre, on en déduit $\rho(B) \leq \|B\|$.
La démonstration de (ii) est un peu plus longue. On peut la trouver dans de nombreux livres (par exemple "Introduction à l'analyse numérique matricielle et à l'optimisation", P.G. Ciarlet, Masson).

6.4.3.1 Vitesse de convergence

Une première estimation est obtenue à l'aide du lemme suivant :

Lemme 6.2 *Pour toute matrice carrée B, et toute norme matricielle*

$$\lim_{k\to\infty} \left\|B^k\right\|^{1/k} = \rho(B). \tag{6.25}$$

Démonstration :

$$\rho(B) = \left(\rho\left(B^k\right)\right)^{1/k} \leq \left\|B^k\right\|^{1/k} \tag{6.26}$$

ce qui donne une inégalité dans un sens.
Soit maintenant $\varepsilon > 0$ et $B_\varepsilon = \frac{B}{\rho(B)+\varepsilon}$; alors $\rho(B_\varepsilon) = \frac{\rho(B)}{\rho(B)+\varepsilon} < 1$. On en déduit

$$\lim_{k\to\infty} B_\varepsilon^k = 0 \text{ soit } \lim_{k\to\infty} \frac{B^k}{(\rho(B)+\varepsilon)^k} = 0.$$

Ainsi pour k assez grand et pour toute norme matricielle :

$$\left\|B^k\right\| \leq (\rho(B)+\varepsilon)^k \text{ ou } \left\|B^k\right\|^{1/k} \leq \rho(B) + \varepsilon. \tag{6.27}$$

De (6.27) et (6.26), on déduit le lemme.

Conséquence du lemme 6.2 : Nous avons vu (cf. (6.22)) que

$$\left\|X^k - X\right\| \leq \left\|B^k\right\| \left\|X^0 - X\right\|.$$

La vitesse de convergence de X^k vers X dépend donc de la vitesse de convergence vers 0 de B^k et donc de $\rho(B)$: il faut que $\rho(B)$ soit strictement inférieur à 1 et la vitesse de convergence sera d'autant plus rapide que $\rho(B)$ est petit. Ainsi, pour k assez grand, on a :

$$\left\|X^k - X\right\| \leq (\rho(B) + \varepsilon)^k \left\|X^0 - X\right\|.$$

De plus, si $\rho\left(\widetilde{B}\right) < \rho(B)$, la convergence de $\widetilde{X}^{k+1} = \widetilde{B}\widetilde{X}^k + \widetilde{d}$ sera plus rapide que celle de X^k. Pour comparer les vitesses de convergence on utilise parfois :

Définition 6.5 *On appelle taux asymptotique de convergence :* $R_\infty(B) = -\log(\rho(B))$.

Ainsi, si $\rho\left(\widetilde{B}\right) = \rho(B)^2 < 1$, on dit (et on vérifie) que la convergence de $\widetilde{X}^k$ est deux fois plus rapide que celle de X^k.

6.4.3.2 Application aux méthodes de Jacobi-Gauss-Seidel-Relaxation

Rappelons les "matrices B" de chaque méthode avec les notations de (6.20) :
Jacobi : $J := D^{-1}(E+F)$
Gauss-Seidel : $L_1 := (D-E)^{-1}F$
Relaxation : $L_\omega := (D-\omega E)^{-1}((1-\omega)D + \omega F)$.

Proposition 6.2 $\rho(L_\omega) \geq |\omega - 1|$. *Donc la méthode de relaxation ne peut converger que si*
$0 < \omega < 2$.

Démonstration : $|\det L_\omega| = |$produit des valeurs propres de $L_\omega| \leq \rho(L_\omega)^N$. Or

$$\det L_\omega = \frac{\det((1-\omega)D + \omega F)}{\det(D - \omega E)} = \frac{(1-\omega)^N \det D}{\det D} = (1-\omega)^N.$$

Ainsi

$$|1-\omega|^N \leq \rho(L_\omega)^N.$$

Nous allons maintenant examiner plusieurs cas de convergence de ces méthodes. Notons d'abord que, dès la dimension 2, elles peuvent être divergentes.
Exemple :

$$\begin{aligned}
A &= \begin{bmatrix} 1 & -2 \\ -2 & 1 \end{bmatrix} \ (A \text{ est symétrique}) \\
D &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad E = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix} \quad F = \begin{bmatrix} 0 & 2 \\ 0 & 0 \end{bmatrix} \\
J &= \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix} \quad \rho(J) = 2 \\
L_1 &= \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 0 & -4 \end{bmatrix} \qquad \rho(L_1) = 4.
\end{aligned}$$

On peut démontrer par exemple :

Théorème 6.4 *On suppose A symétrique définie positive. Alors la méthode de relaxation converge pour tout $\omega \in]0,2[$ (en particulier la méthode de Gauss-Seidel converge).*

On a aussi :

Théorème 6.5 *Si A est à diagonale strictement dominante i.e.*

$$\forall i = 1, ..., N \qquad |a_{ii}| > \sum_{j \neq i} |a_{ij}| \tag{6.28}$$

alors les méthodes de Gauss-Seidel et Jacobi sont convergentes.

On va utiliser pour le démontrer le

Lemme 6.3 (Hadamard-Gerschgörin) *Les valeurs propres d'une matrice quelconque A sont situées dans la réunion des disques*

$$D_i = \left\{ \lambda \in C; |\lambda - a_{ii}| \leq \sum_{j \neq i} |a_{ij}| \right\}, \; i = 1, ..., N.$$

Démonstration : Ecrivons $AX = \lambda X$ soit

$$\forall i = 1, ..., N, \sum_{j=1}^{N} a_{ij} x_j = \lambda x_i.$$

Soit k un indice tel que $|x_k| = \max_i |x_i|$. L'égalité ci-dessus avec $i = k$ implique

$$|(\lambda - a_{kk}) x_k| = \left| \sum_{j \neq i} a_{kj} x_j \right| \leq \sum_{j \neq k} |a_{kj}| \, |x_j|$$

soit

$$|\lambda - a_{kk}| \, |x_k| \leq \left(\sum_{j \neq k} |a_{kj}| \right) |x_k|$$

ce qui démontre le résultat.

Remarque 6.20 Le lemme 6.3 a de nombreuses applications. Notons par exemple qu'il donne une estimation a priori sur la position des valeurs propres d'une matrice quelconque dans le plan complexe. Ce renseignement préliminaire peut être utile pour le calcul effectif de ces valeurs propres (cf. chapitre suivant).

On en déduit aussi le

Corollaire 6.1 *Une matrice à diagonale strictement dominante est inversible.*

En effet, si A n'est pas inversible, 0 est valeur propre et d'après le lemme 6.3, il existe i tel que

$$|a_{ii}| \leq \sum_{j \neq i} |a_{ij}|$$

ce qui contredit la propriété de stricte dominance de la diagonale.

Démonstration du théorème 6.5 : Calculons $\rho(J) = \rho\left(D^{-1}(E + F)\right)$. La matrice J est à diagonale nulle et pour $i \neq j$ on a

$$J_{ij} = -\frac{a_{ij}}{a_{ii}}.$$

D'après le lemme de Hadamard-Gerschgörin, si λ est valeur propre, on a

$$|\lambda - 0| = |\lambda| \leq \sum_{i \neq j} \frac{|a_{ij}|}{|a_{ii}|} < 1$$

et donc $\rho(J) < 1$. Soit maintenant λ une valeur propre de L_1 (éventuellement complexe). Si X est un vecteur propre associé, on a

$$\begin{aligned}(D-E)^{-1}FX &= \lambda X \Longrightarrow FX = \lambda(D-E)X \\ &\Longrightarrow \forall i = 1,...,N \qquad -\sum_{j>i} a_{ij}x_j = \lambda \sum_{j\leq i} a_{ij}x_j.\end{aligned}$$

Soit k un indice tel que $|x_k| = \max_i |x_i|$. La relation ci-dessus écrite avec $i = k$ donne

$$\begin{aligned}\lambda a_{kk}x_k &= -\sum_{j>k} a_{kj}x_j - \lambda \sum_{j<k} a_{kj}x_j \\ |\lambda|\,|a_{kk}| &\leq \sum_{j>k} |a_{kj}| + |\lambda| \sum_{j<k} |a_{kj}|\,.\end{aligned}$$

Si $|\lambda| \neq 0$, puisque A est à diagonale strictement dominante, on obtient

$$\begin{aligned}|\lambda| \sum_{j\neq k} |a_{kj}| &< \textstyle\sum_{j>k} |a_{kj}| + |\lambda| \sum_{j<k} |a_{kj}| \\ &\Longrightarrow |\lambda| < 1 \text{ et donc } \rho(L_1) < 1.\end{aligned}$$

Remarque 6.21 On peut montrer que si $D^{-1}E$ et $D^{-1}F$ sont à éléments positifs ou nuls, alors, dès que la méthode de Jacobi est convergente, il en est de même de celle de Gauss-Seidel. De plus, la convergence de celle-ci est au moins aussi rapide. Ce phénomène n'est pas systématique comme on peut le voir dans l'exercice 6.7.

Cependant, pour de nombreux systèmes, surtout lorsqu'ils proviennent de la discrétisation d'équations aux dérivées partielles, la méthode de Gauss-Seidel converge plus vite que celle de Jacobi. De plus, pour une large famille de tels systèmes, on peut montrer l'existence d'un paramètre ω^* optimal pour lequel la méthode de relaxation est considérablement plus efficace : le nombre d'itérations nécessaires pour atteindre une précision donnée chute considérablement lorsque ω est voisin de ω^*.

Nous allons énoncer sans démonstration un résultat dans ce sens. Pour cela, nous avons besoin de la définition suivante : A étant décomposée comme précédemment, on note :

$$L = D^{-1}E, \quad U = D^{-1}F$$

d'où

$$\begin{aligned}A &= D(I - L - U), \quad J = L + U \\ L_\omega &= (I - \omega L)^{-1}((1-\omega)I + \omega U).\end{aligned}$$

Définition 6.6 *On dira que A est de type (V), si pour $\alpha \neq 0$, les valeurs propres de $J(\alpha) = \alpha L + \frac{1}{\alpha}U$ sont indépendantes de α.*

Remarque 6.22 On montre que les matrices tridiagonales sont de type (V). De nombreuses matrices provenant de la discrétisation d'équations aux dérivées partielles sont également de type (V). Cette notion a été introduite par Varga qui utilise la terminologie "consistently ordered matrices" dont la traduction littérale française est peu heureuse.

On a alors le résultat général suivant :

Théorème 6.6 *Soit A de type (V). Alors*

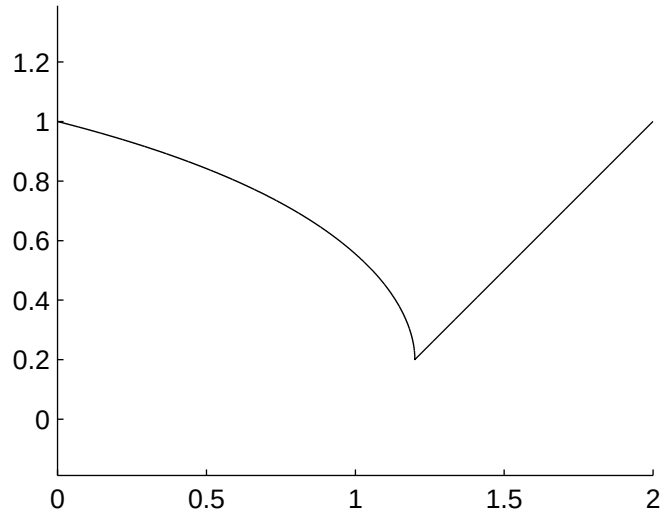

FIGURE 6.2 – Graphe du rayon spectral de L_ω

(i) $\rho(L_1) = \rho(J)^2$.
Donc la méthode de Gauss-Seidel converge si et seulement si celle de Jacobi converge et elle converge alors deux fois plus vite.
(ii) Si, de plus, les valeurs propres de J *sont réelles et* $\rho(J) < 1$, *alors le graphe de* $\rho(L_\omega)$ *a l'allure suivante : Plus précisément, on a :*

$$\omega^* = \frac{2}{1+\sqrt{1-\rho(J)^2}} \quad \rho(L_{\omega^*}) = \left(\frac{\rho(J)}{1+\sqrt{1-\rho(J)^2}}\right)^2$$

et

$$\rho(L_\omega) = \begin{cases} \omega - 1 & si\ \omega^* \le \omega \le 2 \\ 1-\omega+\frac{1}{2}\omega^2\rho(J)^2 + \omega\rho(J)\sqrt{1-\omega+\frac{1}{4}\omega^2\rho(J)^2} & si\ 0 \le \omega \le \omega^*. \end{cases}$$

Remarque 6.23 Le plus souvent, ω^* est déterminé expérimentalement. Noter à ce propos que la dérivée à gauche de $\rho(L_\omega)$ en $\omega = \omega^*$ est infinie. On aura donc plutôt intérêt à surévaluer ω^* car une erreur à droite sur ω^* sera moins sensible qu'une erreur à gauche.

6.4.4 Application à la méthode des différences finies

Nous allons expliciter les résultats du théorème 6.6 pour deux systèmes linéaires provenant de la discrétisation par différences finies d'équations aux dérivées partielles simples.
Considérons d'abord le problème aux limites monodimensionnel standard

$$\begin{cases} -u''(x) = f(x) \quad 0 < x < 1 \\ u(0) = u(1) = 0. \end{cases} \tag{6.29}$$

On montre facilement que si f est continue sur [0,1], ce problème admet une solution u unique. Le calcul numérique de cette solution par la méthode des différences finies

consiste à remplacer la recherche de u par celle d'un vecteur $u_h = (u_1, u_2, ..., u_{\text{N}})$ représentant les valeurs d'une solution approchée en les points $x_1, x_2, ..., x_{\text{N}}$ d'une subdivision de l'intervalle [0,1] que nous supposerons régulière pour simplifier soit

$$x_i = ih,\ i = 0, ..., N+1,\ \ h = \frac{1}{N+1}.$$

Les valeurs en $x_0 = 0$ et $x_{\text{N}+1} = 1$ seront supposées nulles pour satisfaire aux conditions aux limites de (6.29). Le problème (6.29) est alors remplacé par un problème approché où $u''(x_i)$ est remplacé par la différence finie

$$u''(x_i) \approx \frac{u(x_{i+1}) + u(x_{i-1}) - 2u(x_i)}{h^2}.$$

Le vecteur approché u_h cherché est donc défini par le système

$$\begin{cases} -\dfrac{u_{i+1} + u_{i-1} - 2u_i}{h^2} = f_i \quad i = 1, ..., N \\ u_0 = u_{\text{N}+1} = 0 \end{cases} \tag{6.30}$$

où on a noté $f_i = f(x_i)$. Tenant compte des conditions aux limites $u_0 = 0$ dans la première équation et $u_{\text{N}+1} = 0$ dans la dernière, il s'agit donc de résoudre le système linéaire :

$$\frac{1}{h^2} A u_h = f_h \tag{6.31}$$

où $f_h = (f_1, ..., f_N)$ et

$$A = \begin{bmatrix} 2 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}. \tag{6.32}$$

On vérifie que A est une matrice tridiagonale symétrique définie positive. Le système (6.30) pourra être résolu à l'aide de la méthode directe développée dans le théorème 6.1 conduisant à un nombre d'opérations de l'ordre de $8N$. Cette méthode est d'ailleurs fortement conseillée.

Cependant, à titre indicatif, nous allons expliciter les résultats du théorème 6.6 sur cet exemple. La matrice A étant définie par (6.32), on a, avec les notations utilisées précédemment

$$J(\alpha) = \alpha D^{-1}E + \frac{1}{\alpha} D^{-1}F = \frac{1}{2} \begin{bmatrix} 0 & \frac{1}{\alpha} & 0 & \cdots & \cdots & 0 \\ \alpha & \ddots & \ddots & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \frac{1}{\alpha} \\ 0 & \cdots & \cdots & 0 & \alpha & 0 \end{bmatrix}.$$

On vérifie aisément que le vecteur $U^k = \left(U_i^k\right)_{1 \le i \le N}$ tel que

$$U_i^k = \alpha^{i-1} \sin i \frac{k\pi}{N+1},\ \ i = 1, ..., N$$

est vecteur propre de $J(\alpha)$ pour la valeur propre $\lambda_k = \cos\frac{k\pi}{N+1}$ et ce pour $k = 1, ..., N$. Ceci montre que A est de type (V) et

$$\rho(J) = \max_{1\leq k\leq N} |\lambda_k| = \cos\frac{\pi}{N+1}.$$

On en déduit en particulier que, lorsque N est grand,

$$\rho(J) = 1 - \frac{1}{2}\frac{\pi^2}{(N+1)^2} + O\left(\frac{1}{N^4}\right) \tag{6.33}$$

et donc $\rho(J)$ tend vers 1 lorsque N tend vers l'infini. La vitesse de convergence décroît ainsi avec N. D'après le théorème 6.6, on a

$$\rho(L_1) = \rho(J)^2 = 1 - \frac{\pi^2}{(N+1)^2} + O\left(\frac{1}{N^4}\right) \tag{6.34}$$

$$\omega^* = \frac{2}{1+\sqrt{1-\cos^2\left(\frac{\pi}{N+1}\right)}} = \frac{2}{1+\sin\left(\frac{\pi}{N+1}\right)} \tag{6.35}$$

$$\rho(L_{\omega^*}) = \frac{\cos^2\left(\frac{\pi}{N+1}\right)}{\left(1+\sin\left(\frac{\pi}{N+1}\right)\right)^2}. \tag{6.36}$$

Ainsi, pour N grand

$$\rho(L_{\omega^*}) = 1 - \frac{2\pi}{N+1} + O\left(\frac{1}{N^2}\right) \tag{6.37}$$

Les relations (6.33), (6.34), (6.37) permettent de comparer les vitesses de convergence des différentes méthodes lorsque N est grand. Rappelons que, comme conséquence du lemme 6.2, si x_{N} est définie par

$$X_{n+1} = Bx_{\mathrm{N}} + d$$

alors, asymptotiquement

$$\|x_{\mathrm{N}} - X_\infty\| \leq \rho(B)^n \|X_0 - X_\infty\|. \tag{6.38}$$

Notons $c_0 = \|X_0 - X_\infty\|$; pour rendre $\|x_{\mathrm{N}} - X_\infty\|$ inférieur à ε, il faut, en supposant l'estimation (6.38) optimale ce qui est le plus souvent le cas, que $c_0\rho(B)^n \leq \varepsilon$, soit en notant $\alpha = -\log\frac{\varepsilon}{c_0}$ et $R_B = -\log\rho(B)$ (vitesse de convergence)

$$n \geq \frac{\alpha}{R_B}.$$

Pour de grandes dimensions N, d'après (6.33), (6.34), (6.37), on a

$$R_J \sim \frac{\pi^2}{2N^2},\ R_{L_1} \sim \frac{\pi^2}{N^2},\ R_{L_{\omega^*}} \sim \frac{2\pi}{N}.$$

Il sera donc nécessaire de prendre

$$\begin{aligned} n &\sim \frac{2\alpha}{\pi^2}N^2 \text{ pour Jacobi} \\ n &\sim \frac{\alpha}{\pi^2}N^2 \text{ pour Gauss-Seidel} \\ n &\sim \frac{\alpha}{2\pi}N \text{ pour la relaxation optimale.} \end{aligned}$$

Cette dernière méthode est donc plus que N fois plus rapide que celles de Jacobi et Gauss-Seidel.
Afin d'évaluer le temps global de calcul pour éventuellement comparer avec la méthode directe suggérée plus haut, outre le nombre d'itérations, il est nécessaire de tenir compte du nombre d'opérations élémentaires à chaque itération, soit kN où k est une constante de l'ordre de 8. D'autre part, il faut aussi tenir compte d'un choix "raisonnable" de la tolérance ε. Une analyse de l'erreur de discrétisation montre que

$$\max_i |u(x_i) - u_i| \leq Ch^2$$

où u est la solution exacte, (u_i) la solution approchée et C une constante dépendant de u. Il est donc raisonnable de choisir ε d'ordre h^2, soit, pour une certaine constante a

$$\alpha \approx -\log ah^2 \approx \log aN^2 \sim 2\log N \text{ pour } N \text{ grand.}$$

On obtient alors le tableau suivant valable pour N grand

Méthode	Nombre d'itérations	Nombre d'opérations
Jacobi	$\frac{4}{\pi^2}N^2 \log N$	$k\frac{4}{\pi^2}N^3 \log N$
Gauss-Seidel	$\frac{2}{\pi^2}N^2 \log N$	$k\frac{2}{\pi^2}N^3 \log N$
Relaxation optimale	$\frac{1}{\pi}N \log N$	$k\frac{1}{\pi}N^2 \log N$

Remarque 6.24 Comme annoncé, on constate que le nombre d'opérations requis est bien supérieur à celui requis pour la méthode directe du théorème 6.1, soit un facteur de N. Cette différence importante est bien sûr très liée à la structure tridiagonale. Elle devient moins importante pour de grandes matrices à bandes très creuses. De plus, dans ce cas, même si le nombre d'opérations est plus grand, pour des raisons de stockage en mémoire, on pourra préférer une méthode itérative à une factorisation qui "remplit" les bandes et rend ainsi le stockage des matrices-facteurs difficile, voire impossible pour des matrices très grandes.

Pour terminer ce paragraphe, nous énonçons sans détails les résultats relatifs au système discrétisé associé à l'équation aux dérivées partielles bidimensionnelle :

$$\begin{cases} -\frac{\partial^2 u}{\partial x^2}(x,y) - \frac{\partial^2 u}{\partial y^2}(x,y) = f(x,y) \text{ sur } \Omega \subset \mathbb{R}^2 \\ u(x,y) = 0 \text{ sur le bord de } \Omega. \end{cases}$$

Nous supposons que Ω est le carré $[0,1] \times [0,1]$. On le munit d'un maillage uniforme de même pas h dans les deux directions dont les noeuds sont les points (ih, jh), $i, j = 0, 1, ..., N+1$. Le problème discrétisé consiste à trouver une approximation de la solution aux noeuds du maillage, la dérivée seconde en un point (ih, jh) étant remplacée par la formule de différences finies à 5 points

$$u''(ih, jh) \approx \frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j}}{h^2}$$

où $U_{i,j}$ est la valeur de la solution au point (ih, jh). Le système discrétisé s'écrit alors avec $f_{i,j} = f(ih, jh)$ et $h = \frac{1}{N+1}$:

$$\begin{cases} -\frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j}}{h^2} = f_{i,j} \quad i,j = 1, ..., N \\ U_{0,j} = U_{N+1,j} := 0, \ j = 0, ..., N+1 \\ U_{i,0} = U_{i,N+1} := 0, \quad i = 0, ..., N+1 \end{cases}$$

L'écriture matricielle de ce système nécessite le choix d'une numérotation des noeuds. Une numérotation par ligne conduit à un système

$$\frac{1}{h^2}AU_h = f_h$$

où

$$A = \begin{bmatrix}
4 & -1 & 0 & \cdots & 0 & -1 & & & & & & & \\
-1 & 4 & -1 & 0 & \cdots & 0 & \ddots & & & & & & \\
0 & -1 & 4 & -1 & 0 & \cdots & 0 & -1 & & & & & \\
-1 & & & 4 & -1 & & & & \ddots & & & & \\
 & \ddots & & -1 & \ddots & -1 & & & & \ddots & & & \\
 & & -1 & & -1 & 4 & & & & & \ddots & & \\
 & & & \ddots & & & \ddots & & & \ddots & & -1 & \\
 & & & & \ddots & & \ddots & \ddots & & & \ddots & & \ddots \\
 & & & & & \ddots & & & & \ddots & \ddots & & & -1 \\
 & & & & & & -1 & & & 4 & & & -1 & \\
 & & & & & & & \ddots & & -1 & & & \ddots & -1 \\
 & & & & & & & & -1 & 0 & \cdots & 0 & -1 & 4
\end{bmatrix}$$

on montre que A est de type (V). On vérifie que les vecteurs

$$U^{k,l},\ k,l = 1,...,N$$

de composantes

$$U^{k;l}_{i,j} = \sin i\frac{k\pi}{N+1}\sin j\frac{l\pi}{N+1},\ i,j = 1,...,N$$

sont vecteurs propres de $J = L + U$ pour les valeurs propres

$$\lambda^{k,l} = \frac{1}{2}\left(\cos\frac{k\pi}{N+1} + \cos\frac{l\pi}{N+1}\right).$$

Ainsi

$$\rho(J) = \max_{k,l}\left|\lambda^{k,l}\right| = \cos\frac{\pi}{N+1}.$$

On retrouve donc les mêmes formules qu'en (6.33), (6.34), (6.37). Pour atteindre une précision en h^2, le même nombre d'itérations que dans l'exemple précédent est nécessaire. Chaque itération requiert environ kN^2 opérations, d'où un coût total d'environ (à un facteur près) $N^4 \log N$ pour Jacobi et Gauss-Seidel, $N^3 \log N$ pour la relaxation optimale. A titre de comparaison, une factorisation de Cholesky nécessiterait environ $\frac{1}{2}N^4$ opérations.

6.5 La méthode du gradient conjugué

Nous terminons ce chapitre par une présentation succinte d'une méthode de résolution de systèmes linéaires à matrices symétriques définies positives beaucoup plus moderne. Elle est actuellement très employée pour les grandes matrices creuses. Couplée avec des techniques de préconditionnement, elle fournit à l'heure actuelle, l'une des méthodes les plus performantes pour ce type de matrices.

Il s'agit d'une méthode "semi-directe" : en effet, elle consiste à construire une suite de vecteurs X_k qui, en arithmétique exacte converge en moins de N itérations vers la solution du système (supposé d'ordre N). Cependant, les erreurs d'arrondi font que, dans les calculs effectifs, la valeur exacte n'est pas toujours atteinte au bout de N itérations. On doit alors continuer les itérations (au plus de $3N$ à $5N$). Le coût de chaque itération est celui du produit de la matrice A par un vecteur. Elle est donc particulièrement conseillée pour les matrices creuses.
L'idée de départ est la suivante :

Proposition 6.3 *Soit A une matrice symétrique définie positive d'ordre N. Alors le vecteur $X \in \mathbb{R}^N$ est solution de*

$$AX = b \tag{6.39}$$

si et seulement si il réalise le minimum de la fonctionnelle

$$J(Y) = \frac{1}{2}\,{}^tYAY - {}^tbY. \tag{6.40}$$

Démonstration : Un calcul simple montre que

$$J(Y) - J(X) = {}^t(AX - b)(Y - X) + \frac{1}{2}\,{}^t(Y - X)A(Y - X).$$

Si $AX = b$, puisque

$${}^t(Y - X)A(Y - X) > 0 \text{ si } Y \neq X,$$

cette égalité montre que $J(Y) > J(X)$ et donc X réalise le minimum de J.
Inversement, si X réalise le minimum de J, pour tout réel λ

$$J(X + \lambda Y) \geq J(X),$$

ce qui s'écrit encore

$$\frac{\lambda^2}{2}\,{}^tYAY + \lambda\,{}^t(AX - B)Y \geq 0 \quad \forall \lambda \in \mathbb{R}.$$

Divisant par $\lambda > 0$ et faisant tendre λ vers 0, on obtient

$${}^t(AX - b)Y \geq 0.$$

Faisant de même avec $\lambda < 0$, on obtient

$$\forall Y \in \mathbb{R}^N,\ {}^t(AX - b)Y = 0$$

et donc $AX = b$.

La caractérisation de la solution X de (6.39) donnée par la proposition 6.3 conduit à un nouveau point de vue pour le calcul de X à savoir utiliser un algorithme d'optimisation pour déterminer le minimum de J. On retiendra ici l'algorithme du gradient conjugué dont le principe est le suivant.
On choisit un vecteur initial $X_0 \in \mathbb{R}^N$ et on note $r_0 := b - AX_0$. On construit alors des vecteurs $X_1, X_2, ..., X_k$ et $r_1, r_2, ..., r_k$ de $\mathbb{R}^N$ par récurrence suivant la règle

$$\left\{ \begin{array}{c} X_{k+1} \text{ réalise le minimum de } J(X_k + \lambda_0 r_0 + \lambda_1 r_1 + ... + \lambda_k r_k) \\ \text{parmi les } (\lambda_0, ..., \lambda_k) \in \mathbb{R}^{k+1}. \end{array} \right. \tag{6.41}$$

$$r_{k+1} := b - AX_{k+1}. \tag{6.42}$$

Il est clair que la justification essentielle de l'utilisation de cette méthode est que le problème (6.41) peut être résolu de façon simple. Par ailleurs, les "directions" r_i étant, comme on le verra, linéairement indépendantes, (6.41) correspond à une minimisation sur un espace affine de domaine $k+1$. A la N-ième étape, on obtiendra donc nécessairement le minimum de J sur l'espace $\mathbb{R}^N$ tout entier.

Nous donnons maintenant la formulation explicite de (6.41), (6.42) telle qu'elle est utilisée en pratique.

Algorithme du gradient conjugué

– On choisit $X_0 \in \mathbb{R}^N$

– $r_0 := b - AX_0,\ p_0 := r_0$

$$\left[\begin{array}{c} \text{Pour } k = 0, 1, ... \\ \cdot \text{Si } r_k = 0, \text{ stop ; } X_k \text{ est la solution du système} \\ \cdot \text{ Sinon, on calcule} \\ \alpha_k := \dfrac{{}^t r_k r_k}{{}^t p_k A p_k},\ X_{k+1} := X_k + \alpha_k p_k \\ r_{k+1} := r_k - \alpha_k A p_k,\ \ \beta_k = \dfrac{{}^t r_{k+1} r_{k+1}}{{}^t r_k r_k} \\ p_{k+1} := r_{k+1} + \beta_k p_k. \end{array}\right. \tag{6.43}$$

Remarque 6.25 à chaque étape, seul le produit matriciel Ap_k est à effectuer auquel il faut ajouter l'évaluation de deux produits scalaires et quelques combinaisons linéaires de vecteurs. Ceci est évidemment très modeste si la matrice est creuse. Par ailleurs, il faut à chaque étape stocker les 4 vecteurs X_k, p_k, Ap_k, r_k.

Théorème 6.7 *Soit A symétrique définie positive. Alors il existe $l \leq N$ tel que*

$$AX_l = b \text{ (ou } r_l = 0).$$

De plus, on a les propriétés suivantes

$$\forall 0 \leq i < j \leq l \quad {}^t r_i r_j = 0 \tag{6.44}$$

$$\forall 0 \leq i < j \leq l \quad {}^t p_i A p_j = 0 \tag{6.45}$$

$$\forall k \leq l \ \ r_k = b - AX_k. \tag{6.46}$$

Remarque 6.26 Dans ce résultat, il est implicite que les vecteurs X_k, r_k, p_k peuvent être effectivement calculés pour tout $k \leq l$ selon la loi (6.43). En particulier, pour tout $k < l$, on a

$${}^t r_k r_k > 0,\ \ {}^t p_k A p_k > 0. \tag{6.47}$$

La relation (6.45) exprime que les vecteurs p_i sont deux à deux conjugués par rapport à A. C'est cette propriété qui est à l'origine du nom de la méthode. On remarque que connaissant X_k, on calcule X_{k+1} en lui ajoutant un vecteur colinéaire à p_k. Le vecteur p_k doit être interprété comme une direction dans laquelle on se déplace pour atteindre un nouveau point X_{k+1} en lequel J prend une valeur plus petite (on aura bien sûr $J(X_{k+1}) \leq J(X_k)$ d'après (6.41)). La méthode consiste donc à choisir à chaque fois une direction de descente qui est conjuguée des directions précédemment choisies relativement à la matrice A.

Le théorème 6.7 se démontre de façon élémentaire. Il suffit de vérifier par récurrence les relations (6.44), (6.45), (6.46), (6.47). D'après (6.44), les vecteurs r_0, r_1, ..., r_k forment un système orthogonal. On est donc assuré que r_k s'annule pour $k \leq N =$ dimension

de l'espace. Le processus s'arrête donc en un nombre fini d'itérations ce qui, a priori, en fait une méthode directe. En pratique, on constate souvent, qu'à cause des erreurs d'arrondi, r_N n'est pas nul. On poursuit alors le calcul (6.43) au-delà de $k = N$ jusqu'à ce que r_k soit suffisamment petit. En pratique, la méthode est donc plutôt de nature itérative.

Enfin, pour bien justifier l'idée initiale de la méthode, il faudrait vérifier que la suite X_k satisfait à la propriété de minimisation (6.41). Celle-ci laisse entrevoir que la méthode n'est pas particulière aux fonctionnelles du type (6.40) et doit pouvoir s'étendre à des fonctionnelles non linéaires plus générales, ce qui est le cas. Pour une étude de ce type, nous renvoyons à la littérature concernant les problèmes d'optimisation.

6.5.1 Gradient conjugué avec préconditionnement

On applique souvent la méthode ci-dessus après un changement de variable du type $Z = LX$ où L est facile à inverser. Si la matrice L est bien choisie, l'algorithme converge plus vite. Par exemple si L est la matrice triangulaire de Cholesky, la méthode converge en une itération qui correspond en fait à une remontée et une descente et revient à la méthode usuelle de Cholesky. Le plus souvent, L est obtenue par factorisation incomplète de A de façon à bénéficier des zéros de A et à limiter le coût de stockage.

6.6 Exercices du chapitre 6

Exercice 6.1 Montrer que la factorisation $A = LU$ est unique si on impose que les éléments diagonaux de L soient tous égaux à 1 (et A inversible).

Exercice 6.2 Montrer que la structure-bande est conservée dans la factorisation.

Exercice 6.3 Montrer qu'une permutation de lignes de A revient à multiplier A à gauche par une matrice T simple (on l'appelle matrice de transposition).

Exercice 6.4 Montrer que, dans l'algorithme de Crout compact $u_{kk} = \frac{\det A_k}{\det A_{k-1}}$ $(k \geq 2)$ où A_k est la matrice de rang k extraite de A :

$$A = \left[\begin{array}{c|c} A_k & \\ \hline & \end{array}\right]$$

Exercice 6.5 Montrer que $\max_{i,j} |a_{ij}|$ n'est pas une norme matricielle, mais que $\left(\sum_{i,j} a_{ij}^2\right)^{1/2}$ en est une.

Exercice 6.6 Montrer $\rho(A^*A) = \rho(AA^*)$.

Exercice 6.7 Montrer que $\rho(J) < 1 < \rho(L_1)$ lorsque

$$A = \begin{bmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{bmatrix}.$$

Exercice 6.8 On considère la matrice

$$A = \begin{pmatrix} 2 & 2\sqrt{2} \\ 2\sqrt{2} & 9 \end{pmatrix}.$$

Quel est son conditionnement pour les normes matricielles usuelles ? On multiplie la première ligne par un coefficient α, comment choisir α pour que le conditionnement soit minimum ?

Chapitre 7

Calcul de valeurs propres et vecteurs propres

7.1 Rappels

Soit A une matrice carrée d'ordre N à coefficients réels ou complexes et λ un scalaire réel ou complexe. Alors sont équivalents

$$\exists X \in \mathbb{R}^N \left(\text{ou } \mathbb{C}^N\right),\ X \neq 0,\ AX = \lambda X \tag{7.1}$$

$$A - \lambda I \text{ est une matrice non inversible} \tag{7.2}$$

$$\det\left(A - \lambda I\right) = 0 \tag{7.3}$$

Si λ satisfait à une de ces propriétés, on dit que λ est valeur propre de A et tout vecteur satisfaisant à (7.1) est appelé vecteur propre de A associé à la valeur propre λ.

On vérifie que $\lambda \to \det\left(A - \lambda I\right)$ est un polynôme de degré N (dit polynôme caractéristique de A) dont le coefficient de plus haut degré est $(-1)^N$. Ainsi les valeurs propres de A sont les racines d'un polynôme de degré N. Donc A admet exactement N valeurs propres complexes (en comptant les racines multiples éventuelles avec leur multiplicité). D'autre part, la recherche de valeurs propres étant équivalente à la recherche des racines d'un polynôme, les méthodes ne peuvent être qu'itératives et non directes, puisque, d'après le théorème d'Abel, on ne peut "résoudre par radicaux" un polynôme quelconque de degré supérieur ou égal à 5.

On peut penser que le calcul numérique de valeurs propres est un cas particulier du calcul numérique des racines d'un polynôme. La difficulté d'évaluer numériquement le polynôme $\det\left(A - \lambda I\right)$ fait que les algorithmes reposant sur cette idée sont en général peu performants. C'est en fait, plutôt l'inverse qui se produit à savoir que le calcul des racines de

$$\lambda^N + a_1\lambda^{N-1} + ... + a_{N-1}\lambda + a_N$$

peut se faire en appliquant un algorithme de recherche de valeurs propres à la matrice-compagnon suivante dont le polynôme caractéristique est, au facteur $(-1)^N$ près, le

polynôme précédent.

$$\begin{bmatrix} -a_1 & -a_2 & \cdots & \cdots & \cdots & \cdots & -a_N \\ 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & 0 & & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & 0 \end{bmatrix}$$

Remarque 7.1 On vérifie immédiatement que les valeurs propres d'une matrice triangulaire sont les éléments diagonaux de cette matrice puisqu'on a alors

$$\det(A - \lambda I) = \prod_{i=1}^{N} (a_{ii} - \lambda).$$

Plusieurs des algorithmes qui vont suivre consisteront précisément à "rendre" la matrice A initiale triangulaire ou même diagonale, ce en construisant une suite $A_0 = A$, A_1, ..., A_k de matrices ayant les mêmes valeurs propres et tendant vers une matrice triangulaire ou diagonale. Pour cela, il est nécessaire de rappeler que la transformation de base laissant invariant l'ensemble des valeurs propres est la similitude.

Définition 7.1 *Deux matrices A et A' sont dites semblables s'il existe une matrice inversible P telle que*

$$A' = P^{-1}AP.$$

On dit que A est diagonalisable si elle est semblable à une matrice diagonale.
Deux matrices semblables ont les mêmes valeurs propres.

Proposition 7.1 *Une matrice A est diagonalisable si et seulement s'il existe une base de $\mathbb{R}^N$ (ou $\mathbb{C}^N$) formée de vecteurs propres de A.*

Démonstration : Supposons $A' = P^{-1}AP$ diagonale. Notons e_i le i-ème vecteur de la base canonique de $\mathbb{R}^N$, soit

$$e_i = (0, ..., 1, 0, ..., 0)$$

$$\uparrow i\text{-ème composante}$$

Notons λ_i, $i = 1, ..., N$ les éléments diagonaux de A'. Alors

$$\begin{aligned} A'e_i &= \lambda_i e_i = P^{-1}APe_i \\ &\Longrightarrow \lambda_i Pe_i = APe_i. \end{aligned}$$

Donc $\{Pe_i,\ i = 1, ..., N\}$ sont des vecteurs propres de A. Comme P est inversible, elle transforme une base en une base. On obtient ainsi une base de vecteurs propres pour A qui ne sont autres que les vecteurs colonnes de P.
Inversement, s'il existe une base de vecteurs propres de A, on note P la matrice dont les vecteurs colonnes sont ces vecteurs propres et on vérifie par le même calcul que $P^{-1}AP$ est alors diagonale, avec sur la diagonale les vecteurs propres de A.

Cas particulier : On dit que A est orthogonalement (resp. unitairement) diagonalisable si la matrice P peut être choisie orthogonale, i.e. $P^{-1} = {}^tP$ (resp. unitaire, i.e. $P^{-1} = {}^t\overline{P}$). Ceci est bien sûr équivalent à l'existence de vecteurs propres orthogonaux pour le produit scalaire $\langle X, Y\rangle = \sum_{i=1}^{N} x_i y_i$ (resp. $\langle X, Y\rangle = \sum_{i=1}^{N} x_i \overline{y_i}$ dans le cas complexe).

Remarque 7.2 "orthogonal" et "unitaire" coïncident lorsque les vecteurs sont réels. Si A est à coefficients réels, ses valeurs et vecteurs propres ne sont pas nécessairement réels. Cependant, les valeurs et vecteurs propres complexes sont conjugués deux à deux.

Rappelons sans démonstration quelques résultats que nous serons conduits à utiliser dans la suite.

Théorème 7.1 *i) Si A a toutes ses valeurs propres distinctes, elle est diagonalisable (si A est réelle, les valeurs propres sont alors réelles).*
ii) Une matrice hermitienne (i.e. ${}^tA = \overline{A}$) a toutes ses valeurs propres réelles et est unitairement diagonalisable.
iii) Une matrice symétrique réelle a toutes ses valeurs propres réelles et est orthogonalement diagonalisable.

7.2 Origine des problèmes de valeurs propres

Sans être exhaustif (loin de là), nous allons citer ici quelques exemples simples conduisant naturellement à la recherche de valeurs et vecteurs propres d'une matrice.

7.2.1 Vibration d'une corde

Considérons une corde tendue entre ses deux extrémités supposées fixes. Il s'agit de déterminer les mouvements vibratoires stationnaires de la corde et par là-même ses fréquences fondamentales.
On démontre que, pour de petits déplacements $u(t,x)$ de la corde, l'évolution de $u(t)$ est régie par l'équation des ondes :

$$\frac{\partial^2 u}{\partial t^2}(t,x) - \omega^2 \frac{\partial^2 u}{\partial x^2}(t,x) = 0 \tag{7.4}$$

ce, en l'absence de forces extérieurs. Si la corde est attachée aux extrémités, il faut ajouter les conditions au bord qui sont

$$u(t,0) = 0, \qquad u(t,l) = 0 \tag{7.5}$$

si les extrémités sont définies par $x = 0$ et $x = l$.
La recherche de mouvements stationnaires consiste à déterminer les solutions du système (7.4), (7.5) sous la forme

$$u(t,x) = u(x)\, e^{i\mu t} \tag{7.6}$$

où μ est la fréquence à déterminer et $u(x)$ la forme stationnaire de la corde à déterminer. On constate que les parties réelles et imaginaires de $u(t,x)$ correspondent à des vibrations de période $\frac{2\pi}{\mu}$ auxquelles la corde peut être soumise en régime libre.
Plus généralement, pour résoudre ce problème, posons a priori

$$u(t,x) = u(x)\, v(t)$$

Alors (7.4) équivaut à

$$v''(t)\, u(x) = \omega^2 u''(x)\, v(t)$$

ce qui, par séparation des variables, équivaut à l'existence d'une constante λ telle que

$$-u''(x) = \lambda u(x)\,, \quad -v''(t) = \lambda \omega^2 v(t)\,.$$

Il faut ajouter à ceci les conditions au bord, à savoir

$$u(0) = u(l) = 0.$$

Considérons alors le problème

$$-u''(x) = \lambda u(x) \tag{7.7}$$
$$u(0) = u(l) = 0. \tag{7.8}$$

Il est clair que $u \equiv 0$ (position d'équilibre) est toujours solution. Ce qui nous intéresse est l'existence éventuelle de solutions non triviales $u(x)$. Ceci ne se produira en fait que pour des valeurs bien particulières de λ qui sont précisément les valeurs propres de l'opérateur $u \to u''$ avec les conditions au bord $u(0) = u(1) = 0$.
Cette situation simple peut en fait s'analyser directement. On vérifiera les résultats suivants :
Si $\lambda < 0$, la solution générale de (7.7) s'écrit

$$u(x) = Ae^{\sqrt{-\lambda}x} + Be^{\sqrt{-\lambda}x}$$

où A et B sont des constantes à déterminer par les conditions (7.8). Dans tous les cas, on obtient $A = B = 0$ donc seulement la solution triviale.
Si $\lambda = 0$, la solution générale de (7.7) s'écrit

$$u(x) = Ax + B$$

ce qui avec (7.8) conduit au même résultat négatif.
Si $\lambda > 0$, la solution générale de (7.7) est

$$u(x) = A\cos\sqrt{\lambda}x + B\sin\sqrt{\lambda}x.$$

On constate qu'on peut satisfaire (7.8) si et seulement si

$$\lambda = \lambda_k = \frac{k^2\pi^2}{l^2},\ k = 1, 2, ...$$

les solutions correspondantes étant données par

$$u_k(x) = \sin\left(\frac{k\pi x}{l}\right).$$

Revenant à l'équation en v, aux λ_k, u_k, on peut associer les solutions

$$v_k(t) = C_k e^{ik\omega t/l},\ C_k \in C.$$

D'où les solutions stationnaires cherchées :

$$u_k(t, x) = C_k \sin\left(\frac{k\pi x}{l}\right) e^{ik\pi\omega t/l}$$

Dans bien des cas, la résolution analytique explicite de tels problèmes n'est pas possible et il est nécessaire de recourir à une résolution numérique.
Le premier travail consiste à remplacer le problème continu de type (7.7), (7.8) par un problème discrétisé approché où les inconnues sont en nombre fini. On utilise par exemple une discrétisation par différences finies ou éléments finis.

Ainsi, on pourra remplacer (7.7), (7.8) par le problème discrétisé

$$\begin{cases} -\dfrac{U_{i+1}+U_{i-1}-2U_i}{h^2} = \lambda U_i,\ i=1,...,N \\ U_0 = 0,\ U_{N+1} = 0 \end{cases} \tag{7.9}$$

correspondant à une subdivision régulière de pas $h = \frac{l}{N+1}$ de l'intervalle $[0,1]$ et utilisant l'approximation par différences finies à 3 points de la dérivée seconde

$$u''(x) \approx \frac{u(x+h)+u(x-h)-2u(x)}{h^2}.$$

On est donc ramené au problème de la recherche des valeurs propres de la matrice

$$A_h = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}$$

Dans ce cas simple, on peut d'ailleurs calculer explicitement les valeurs propres de A_h, soit :

$$\lambda_k^h = \frac{4}{h^2}\sin^2\frac{k\pi}{2(N+1)} \quad 1 \le k \le N$$

associées aux vecteurs propres

$$U^k = \left(U_i^k\right) \quad U_i^k = \sin\frac{k\pi i}{N+1}.$$

On constate que pour k fixé,

$$\lim_{h\downarrow 0}\lambda_k^h = \lambda_k = k^2\pi^2.$$

7.2.2 Vibration d'une membrane

La version bidimensionnelle du problème précédent consiste à déterminer les vibrations propres d'une membrane fixée à un contour rigide Γ. L'équation aux dérivées partielles s'écrit

$$\frac{\partial^2 u}{\partial t^2}(t,x,y) = \omega^2\left[\frac{\partial^2 u}{\partial x^2}(t,x,y)+\frac{\partial^2 u}{\partial y^2}(t,x,y)\right],\ (x,y)\in\Omega$$

où Ω est l'intérieur du contour Γ et

$$u(t,x,y) = 0 \text{ sur } \Gamma.$$

Le problème de valeurs propres associé s'écrit

$$\begin{cases} -\left[\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right] = \lambda u \text{ sur } \Omega \\ u = 0 \text{ sur } \Gamma. \end{cases} \tag{7.10}$$

La discrétisation de (7.10) conduit à la recherche des valeurs propres de la matrice de discrétisation de l'opérateur Laplacien $\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}$.

7.2.3 Problèmes de valeurs propres généralisés

Le plus souvent la discrétisation des opérateurs aux dérivées partielles intervenant dans les problèmes de la physique conduit à des problèmes de valeurs propres généralisées du type

$$Au = \lambda Bu \tag{7.11}$$

où A et B sont des matrices. Si B est diagonale, on est encore dans la situation précédente. Cependant, le plus souvent B est une matrice simple mais non diagonale ; elle sera en général symétrique définie positive et creuse. En théorie, on peut se ramener au cas précédent en remplaçant (7.11) par le système équivalent

$$B^{-1}Au = \lambda u$$

Mais ceci détruit en général les propriétés structurelles de A et B. Par exemple $B^{-1}A$ n'est pas en général symétrique si A et B le sont.
On préférera commencer par une factorisation de Cholesky de B, soit $B = C{}^tC$ où C est triangulaire inférieure. Alors, on écrit (7.11) sous la forme

$$C^{-1}Au = \lambda {}^tCu$$

ce qui équivaut à

$$\begin{cases} C^{-1}A\left({}^tC\right)^{-1}X = \lambda X \\ {}^tCu = X. \end{cases}$$

On a donc à résoudre un problème de valeurs propres à matrice symétrique $C^{-1}A\left({}^tC\right)^{-1}$ puis à résoudre un système triangulaire pour obtenir les vecteurs propres u à partir de X.

7.2.4 Système mécanique

Le calcul des fréquences fondamentales de "petits" mouvements, au voisinage d'une position d'équilibre, d'un système mécanique ayant un nombre fini N de degrés de liberté conduit à une équation différentielle du type

$$Mu''(t) + Su'(t) + Ru(t) = 0$$

où $u(t)$ est un vecteur dont les composantes sont les N degrés de liberté du système et M, S, R des matrices réelles d'ordre N. Ici M est la matrice de "l'énergie cinétique" ou matrice de masse, R la matrice de "rappel" ou matrice de rigidité et S la matrice d'amortissement.
Si on cherche des solutions de la forme $u(t) = e^{\mu t}u$ où u et μ tels que

$$\left(\mu^2 M + \mu S + R\right)u = 0,$$

ceci est un nouveau problème généralisé de valeurs propres. A chaque solution μ correspondra une période fondamentale $T = \frac{2\pi}{\Im m\mu}$. La partie réelle de μ correspond au terme d'amortissement.
On se ramène à la situation précédente en considérant les matrices d'ordre $2N$.

$$A = \begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} \qquad B = \begin{bmatrix} 0 & -I \\ R & S \end{bmatrix}.$$

Si $(\mu, \widetilde{u})$ est solution de $\mu A\,\widetilde{u} + B\,\widetilde{u} = 0$ en posant $\widetilde{u} = (u, v)$ on a :

$$\begin{cases} \mu u - v = 0 \\ \mu Mv + Ru + Sv = 0 \end{cases} \quad \text{soit } \mu^2 Mu + \mu Su + Ru = 0.$$

7.2.5 Autres exemples

Citons maintenant sans détailler d'autres problèmes d'origines très diverses qui conduisent à vouloir calculer des valeurs propres :

- En analyse numérique, on a vu qu'on avait souvent besoin de déterminer le rayon spectral d'une matrice (convergence des méthodes itératives, calcul du conditionnement).
- L'étude du comportement des suites récurrentes linéaires, qui interviennent dans de nombreux domaines à partir du moment où on effectue la modélisation d'un phénomène discret, nécessite la détermination de la plus grande valeur propre d'une matrice (exemples : matrice de transition en biologie, chaînes de Markov).
- Dans le même ordre d'idées, l'étude de la stabilité des systèmes différentiels conduit à rechercher les valeurs propres de la matrice du système ou de son linéarisé.
- Les techniques statistiques d'analyse de données, l'analyse en composante principale par exemple, conduisent à rechercher les plus grandes valeurs propres de la matrice des données. Ces techniques sont employées en économie, géographie, sciences humaines,...

On peut trouver encore d'autres exemples. L'un des faits frappants dans l'inventaire ci-dessus est que, suivant le type de questions, on ne cherchera qu'une valeur propre, ou quelques valeurs propres, ou toutes les valeurs propres. De même, on a parfois besoin des vecteurs propres correspondants, quelquefois non. Le choix de la méthode employée, parmi celles qu'on va décrire dans ce chapitre, sera donc très dépendant de ce qu'on veut.

7.3 Méthode de la puissance itérée et de la puissance inverse

Commençons par un algorithme simple très souvent utilisé quand il s'agit seulement de trouver quelques valeurs et vecteurs propres d'une matrice A.

7.3.1 Méthode de la puissance itérée

Considérons d'abord la suite définie par

$$X_{n+1} = AX_n.$$

Supposons que A soit diagonalisable et soit $e_1, e_2, ...e_{\mathrm{N}}$ une base de vecteurs propres. Notons $\lambda_1, ..., \lambda_{\mathrm{N}}$ les valeurs propres associées dans le même ordre. Ecrivons enfin

$$X_0 = \alpha_1 e_1 + \alpha_2 e_2 + ... + \alpha_{\mathrm{N}} e_{\mathrm{N}}$$

la décomposition de X_0 suivant la base (e_i). Alors

$$X_n = \alpha_1 \lambda_1^n e_1 + \alpha_2 \lambda_2^n e_2 + ... + \alpha_{\mathrm{N}} \lambda_{\mathrm{N}}^n e_{\mathrm{N}}.$$

Supposons maintenant qu'on ait rangé les valeurs propres λ_i de telles façon que :

$$|\lambda_{\mathrm{N}}| > |\lambda_{\mathrm{N}-1}| \geq |\lambda_{\mathrm{N}-2}| ... \geq |\lambda_1|.$$

Alors

$$X_n = \lambda_{\mathrm{N}}^n \left[\alpha_{\mathrm{N}} e_{\mathrm{N}} + \left(\frac{\lambda_{\mathrm{N}-1}}{\lambda_{\mathrm{N}}} \right)^n e_{\mathrm{N}-1} + \ldots + \left(\frac{\lambda_1}{\lambda_{\mathrm{N}}} \right)^n e_1 \right]. \tag{7.12}$$

Puisque, pour $i < N$, $\lim_{n\to\infty} \left(\frac{\lambda_i}{\lambda_N}\right)^n = 0$, on voit que pour n grand

$$X_n \sim \lambda_N^n \alpha_N e_N.$$

Donc
– pour n grand, X_n donne la direction du vecteur propre e_N
– si la j-ième composante de e_N est non nulle, on a

$$\lambda_N \simeq \lim_{n\to\infty} \frac{(X_{n+1})_j}{(X_n)_j}.$$

Nous venons de décrire à peu de chose près la méthode de la puissance itérée permettant de calculer la plus grande valeur propre de A et le vecteur propre associé. La seule modification à apporter vient du fait que comme $\|X_n\|$ peut tendre vers l'infini , on ne peut numériquement l'utiliser tel quel. Il faut donc procéder à une normalisation à chaque étape.

Algorithme de la puissance itérée :

$$\begin{cases} X_0 \text{ choisi} \\ Y_{n+1} = AX_n \\ X_{n+1} = Y_{n+1} / \|Y_{n+1}\|_2 \\ \sigma_{n+1} = {}^t\overline{X_n} Y_{n+1}. \end{cases}$$

Proposition 7.2 *On suppose A diagonalisable, ses valeurs propres vérifiant*

$$|\lambda_1| \leq |\lambda_2| \leq \ldots \leq |\lambda_{N-1}| < |\lambda_N|.$$

On note $X_0 = \alpha_1 e_1 + ... + \alpha_N e_N$ où $e_1, ..., e_N$ sont des vecteurs propres respectivement associés à $\lambda_1, ..., \lambda_N$ et on suppose $\alpha_N \neq 0$. Alors

$$\begin{cases} \lim_{n\to\infty} \sigma_{n+1} = \lambda_N \\ \lim_{n\to\infty} X_{n+1} - \beta^n \varepsilon_N = 0 \end{cases} \tag{7.13}$$

où ε_N est un vecteur colinéaire à e_N et $\beta = \frac{\lambda_N}{|\lambda_N|}$. De plus la convergence est au moins linéaire de rapport $\left|\frac{\lambda_{N-1}}{\lambda_N}\right|$.

Démonstration : Posons $Z_n = A^n X_0$. On vérifie par récurrence que $X_n = \frac{Z_n}{\|Z_n\|_2}$. D'après (7.12), on a :

$$Z_n = \lambda_N^n \left(\alpha_N e_N + \sum_{i=1}^{N-1} \left(\frac{\lambda_i}{\lambda_N}\right)^n \alpha_i e_i \right),$$

soit

$$Z_n = \lambda_N^n \left(\alpha_N e_N + \left(\frac{\lambda_{N-1}}{\lambda_N}\right)^n \theta_n \right)$$

où θ_n est un vecteur borné quand $n \to \infty$. On en déduit

$$\|Z_n\|_2 = |\lambda_N|^n \left(|\alpha_N| \, \|e_N\|_2 + \left|\frac{\lambda_{N-1}}{\lambda_N}\right|^n \rho_n \right)$$

où ρ_n est une suite bornée de réels. Ainsi

$$X_n = \left(\frac{\lambda_N}{|\lambda_N|}\right)^n \left[\frac{\alpha_N e_N}{|\alpha_N| \, \|e_N\|_2} + \left|\frac{\lambda_{N-1}}{\lambda_N}\right|^n \theta_n \right]$$

où θ_n est une suite bornée de vecteurs. Ceci montre le deuxième point de (7.13) avec $\beta = \lambda_N / |\lambda_N|$ et $\varepsilon_N = \alpha_N e_N / |\alpha_N| \|e_N\|_2$. D'après les calculs ci-dessus, on a

$$\sigma_{N+1} = \left\langle \beta^n \left(\varepsilon^N + \left(\frac{\lambda_{N-1}}{\lambda_N} \right)^n \theta_n \right), \beta^n \left(\lambda_N \varepsilon_N + \left(\frac{\lambda_{N-1}}{\lambda_N} \right)^n A\theta_n \right) \right\rangle .$$

Puisque $\langle \varepsilon_N, \varepsilon_N \rangle = 1$, on en déduit

$$\sigma_{N+1} = \lambda_N + \left(\frac{\lambda_{N-1}}{\lambda_N} \right)^n \rho_n$$

où ρ_n est une suite bornée de réels. Ceci montre le deuxième point de (7.13). La vitesse de convergence résulte des estimations ci-dessus.

Remarque 7.3 – La convergence est montrée sous l'hypothèse que la composante du vecteur initial selon le vecteur propre e_N soit non nulle. C'est évidemment invérifiable en pratique. Cependant, on constate que les erreurs d'arrondis sont en général suffisantes pour créer une composante non nulle selon e_N qui est ensuite amplifiée. Ceci fait que la méthode converge pratiquement toujours.

– Si la matrice A est réelle et X_0 réel, les vecteurs X_n et σ_n restent réels. Ceci est compatible avec l'hypothèse faite : en effet, comme λ_N est de module supérieur à celui de toutes les autres valeurs propres, elle est nécessairement réelle.

– On peut se demander ce qui se passe dans cet algorithme lorsque

$$|\lambda_N| = |\lambda_{N-1}| > |\lambda_{N-2}| \geq ... \geq |\lambda_1| .$$

On a alors

$$Z_n = \lambda_N^n \left(\alpha_N e_N + \left(\frac{\lambda_{N-1}}{\lambda_N} \right)^n \alpha_{N-1} e_{N-1} + O\left[\left| \frac{\lambda_{N-2}}{\lambda_N} \right|^n \right] \right) .$$

L'analyse de ce cas est renvoyée en exercice.

7.3.2 Méthode de la puissance inverse

Nous pouvons appliquer la méthode précédente à la matrice A^{-1} ; comme les valeurs propres de A^{-1} sont les inverses de celles de A, on a ainsi un procédé pour obtenir la valeur propre de A de plus petit module et le vecteur propre correspondant sous l'hypothèse que A est diagonalisable et

$$|\lambda_1| < |\lambda_2| \leq ... \leq |\lambda_N| .$$

Plutôt que calculer explicitement A^{-1}, on préfère effectuer une factorisation de la matrice A (par exemple de type LU ou de type Choleski si elle est symétrique). Les itérés successifs s'obtiennent alors par la résolution du système $AY_{n+1} = X_n$. Plus généralement, ce procédé nous permet d'aborder le calcul de n'importe quelle valeur propre λ de A dont on connait une valeur suffisamment approchée $\widetilde{\lambda}$. Il suffit d'appliquer la méthode de la puissance inverse à la matrice translatée $A - \widetilde{\lambda} I$.

Algorithme de la puissance inverse avec translation

$$\begin{cases} X_0 \text{ donné} \\ AY_{n+1} - \widetilde{\lambda} Y_{n+1} = X_n \\ X_{n+1} = Y_{n+1} / \|Y_{n+1}\|_2 \\ \sigma_{n+1} = \widetilde{\lambda} + 1 / {}^t\overline{X_n} Y_{n+1}. \end{cases}$$

Si A est diagonalisable et si $\left|\widetilde{\lambda}-\lambda_i\right| < \left|\widetilde{\lambda}-\lambda_j\right| \quad \forall i \neq j$ d'après la proposition 7.2, $\dfrac{1}{\sigma_{n+1}-\widetilde{\lambda}}$ tend vers la valeur propre de plus grand module de $\left(A-\widetilde{\lambda}I\right)^{-1}$ qui n'est autre que $\dfrac{1}{\lambda_i-\widetilde{\lambda}}$ et donc σ_n tend vers λ_i. On vérifie que X_n tend vers un vecteur propre correspondant.

Remarque 7.4 – La méthode ci-dessus permet en fait de trouver toutes les valeurs propres simples (et même multiples, cf. l'exercice 7.1) d'une matrice diagonalisable.
– La convergence dans l'algorithme précédent est linéaire de rapport

$$\max_{j\neq i}\frac{\left|\widetilde{\lambda}-\lambda_i\right|}{\left|\widetilde{\lambda}-\lambda_j\right|}.$$

On peut accélérer la vitesse de convergence en faisant varier $\widetilde{\lambda}$. Prenant par exemple $\widetilde{\lambda_n}=\sigma_n$ à chaque étape, on accélère très sérieusement la convergence.
– Cette technique peut être aisément étendue au problème de valeurs propres généralisé :

$$AX=\lambda BX.$$

Algorithme de la puissance inverse généralisé

$$\left\{\begin{array}{c} X_0 \text{ choisi, } \widetilde{\lambda} \text{ valeur approchée de } \lambda \\ AY_{n+1}-\widetilde{\lambda}BY_{n+1}=BX_n \\ X_{n+1}=Y_{n+1}/\left\|Y_{n+1}\right\|_2 \\ \sigma_{n+1}=\widetilde{\lambda}+1/{}^t\overline{X_n}Y_{n+1}. \end{array}\right.$$

Remarque 7.5 Comme il a déjà été signalé plus haut, il est préférable de travailler avec la matrice $A-\widetilde{\lambda}B$ plutôt que $B^{-1}A$ pour préserver les structures particulières de A et B.

7.4 Méthode de Jacobi

Cette méthode s'applique aux matrices symétriques réelles. Elle permet d'en trouver simultanément toutes les valeurs propres et tous les vecteurs propres. Elle s'applique bien aux matrices pleines.

Principe : On construit des matrices orthogonales Ω_k telles que les matrices A_k définies par

$$\left\{\begin{array}{c} A_0=A \\ A_{k+1}={}^t\Omega_k A_k\Omega_k \end{array}\right.$$

convergent vers une matrice diagonale. On "lit" alors les valeurs propres de A sur la diagonale de A_k pour k assez grand. Les vecteurs propres de A sont les vecteurs colonnes de

$$O_k=\Omega_1\Omega_2...\Omega_k.$$

Les matrices Ω_k seront des matrices de rotation, soit du type

$$\Omega = \begin{bmatrix} 1 & & 0 & \vdots & & & & \vdots & & & & \\ & \ddots & & \vdots & & 0 & & \vdots & & 0 & & \\ 0 & & 1 & \vdots & & & & \vdots & & & & \\ \cdots & \cdots & \cdots & \cos\theta & \cdots & \cdots & \cdots & \sin\theta & \cdots & \cdots & \cdots & p\text{-ième ligne} \\ & & & \vdots & 1 & & 0 & \vdots & & & & \\ & 0 & & \vdots & & \ddots & & \vdots & & 0 & & \\ & & & \vdots & 0 & & 1 & \vdots & & & & \\ \cdots & \cdots & \cdots & -\sin\theta & \cdots & \cdots & \cdots & \cos\theta & \cdots & \cdots & \cdots & q\text{-ième ligne} \\ & & & \vdots & & & & \vdots & 1 & & 0 & \\ & 0 & & \vdots & & 0 & & \vdots & & \ddots & & \\ & & & \vdots & & & & \vdots & 0 & & 1 & \end{bmatrix}$$

p-ième colonne $\qquad$ q-ième colonne

Si A est symétrique, examinons $B = {}^t\Omega A\Omega$. On a

$$b_{ij} = \sum_{\alpha=1}^{N} \omega_{\alpha i} \sum_{k=1}^{N} a_{\alpha k}\omega_{kj} = \sum_{k=1}^{N}\sum_{\alpha=1}^{N} \omega_{\alpha i}\omega_{kj}a_{\alpha k}.$$

Ainsi

$$\begin{cases} \text{si } i \neq p,q \text{ et } j \neq p,q \quad b_{ij} = a_{ij} \\ \text{si } i = p \text{ et } j \neq p,q \quad b_{pj} = \cos\theta a_{pj} - \sin\theta a_{pj} \\ \text{si } i = q \text{ et } j \neq p \quad b_{qj} = \sin\theta a_{pj} + \cos\theta a_{qj} \\ \text{pour } i = j = p \quad b_{pp} = \cos^2\theta a_{pp} + \sin^2\theta a_{qq} - \sin 2\theta a_{pq} \\ \text{pour } i = j = q \quad b_{qq} = \sin^2\theta a_{pp} + \cos^2\theta a_{qq} + \sin 2\theta a_{pq} \\ \text{pour } i = p,\ j = q \quad b_{pq} = \cos 2\theta a_{pq} + \frac{\sin 2\theta}{2}(a_{pp} - a_{qq}) \\ \text{le reste est obtenu par symétrie.} \end{cases}$$

Au vu de l'expression de b_{pq}, il apparait que si $a_{pq} \neq 0$, on peut choisir θ pour que $b_{pq} = 0$, il suffit de prendre θ tel que

$$\cot 2\theta = \frac{a_{qq} - a_{pp}}{2a_{pq}},\ -\frac{\pi}{4} \leq \theta \leq \frac{\pi}{4}. \tag{7.14}$$

Lemme 7.1 *Si θ est choisi selon (7.14), on a*

$$\sum_{i,j} b_{ij}^2 = \sum_{i,j} a_{ij}^2 \tag{7.15}$$

$$\sum_{i} b_{ii}^2 = \sum_{i} a_{ii}^2 + 2a_{pq}^2 \tag{7.16}$$

Remarque 7.6 Au vu de ce lemme, on voit que si $a_{pq} \neq 0$, la diagonale de B est globalement plus dominante que celle de A. Notons que par différence

$$\sum_{i\neq j} b_{ij}^2 = \sum_{i\neq j} a_{ij}^2 - 2a_{pq}^2.$$

La répétition de ce type d'opérations devrait "creuser" de plus en plus la partie hors diagonale des matrices successives.

Démonstration du lemme 7.1 : Pour (7.15) on utilise

$$trace\left({}^tBB\right) = \sum_{i,j} b_{ij}^2, \quad trace\left({}^tAA\right) = \sum_{i,j} a_{ij}^2.$$

Mais :

$$B = {}^t\Omega A\Omega \Longrightarrow {}^tBB = {}^t\Omega {}^tAA\Omega.$$

Donc

$$trace\left({}^tBB\right) = trace\left({}^t\Omega {}^tAA\Omega\right) = trace\left({}^tAA\right).$$

Pour (7.16), on remarque que

$$\begin{bmatrix} b_{pp} & b_{pq} \\ b_{qp} & b_{qq} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}.$$

par le même raisonnement que ci-dessus, on a

$$b_{pp}^2 + b_{qq}^2 + 2b_{pq}^2 = a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2.$$

Si on choisit θ pour que $b_{pq} = 0$, comme les autres éléments diagonaux de B sont identiques à ceux de A, on en déduit (7.16).

Remarque 7.7 Dans la pratique, on évite de faire appel aux fonctions transcendantes pour le calcul de B en fonction de A. On utilise un calcul reposant sur les formules trigonométriques classiques : si $t = \tan\theta$

$$\cot 2\theta = \frac{1-t^2}{2t}, \ \cos^2\theta = \frac{1}{1+t^2}, \ \sin^2\theta = \frac{t^2}{1+t^2}.$$

Ainsi, puisqu'on peut supposer $|\theta| \leq \frac{\pi}{4}$, posant

$$\lambda = \frac{a_{qq} - a_{pq}}{2a_{pq}} \tag{7.17}$$

la valeur t est calculée comme la racine de plus petit module de

$$t^2 + 2\lambda t - 1 = 0. \tag{7.18}$$

Puis on calcule

$$c = \frac{1}{\sqrt{1+t^2}}, \ s = tc \tag{7.19}$$

$$\begin{cases} b_{pi} = ca_{pi} - sa_{qi} & i \neq p, q \\ b_{qi} = ca_{qi} - sa_{pi} & i \neq p, q \\ b_{pp} = a_{pp} - ta_{pq} \\ b_{qq} = a_{qq} + ta_{pq} \end{cases} \tag{7.20}$$

Algorithme de Jacobi classique

- On pose $A_0 = A$
- On détermine l'élément a_{pq}^k de A_k réalisant le plus grand module parmi les éléments hors diagonaux.
- On calcule A_{k+1} selon les formules (7.17)-(7.20) où A_k joue le rôle de A et A_{k+1} celui de B.

Proposition 7.3 *Les éléments hors diagonaux de A_k tendent vers 0 lorsque k tend vers l'infini.*

Démonstration : Notons $m_k = \max_{i\neq j} \left|a_{ij}^k\right|$. Le couple (p,q) est choisi pour que $\left|a_{pq}^k\right| = m_k$. D'après (7.15), (7.16), on a

$$\sum_{i\neq j} \left(a_{ij}^{k+1}\right)^2 = \sum_{i\neq j} \left(a_{ij}^k\right)^2 - 2m_k^2.$$

Puisque

$$\sum_{i\neq j} \left(a_{ij}^k\right)^2 \leq \left(N^2 - N\right) m_k^2,$$

on en déduit

$$\sum_{i\neq j} \left(a_{ij}^{k+1}\right)^2 \leq \left(1 - \frac{2}{N\left(N-1\right)}\right) \sum_{i\neq j} \left(a_{ij}^k\right)^2.$$

Donc la suite $\sum_{i\neq j} \left(a_{ij}^{k+1}\right)^2$ converge de façon géométrique vers 0.

Proposition 7.4 *La diagonale D_k de A_k converge vers une matrice $D = diag\left(\lambda_{\sigma(i)}\right)$ où σ est une permutation sur $\{1, 2, ..., N\}$.*

Démonstration : On a, puisque A et A_k sont semblables et par définition de D

$$\det\left(A - \lambda I\right) = \det\left(A_k - \lambda I\right) = \det\left(D - \lambda I\right)$$

et d'après la proposition 7.3

$$\lim_{k\to\infty} \det\left(D_k - \lambda I\right) = \det\left(D - \lambda I\right).$$

En particulier les racines du premier polynôme convergent vers celles du second. Reste à vérifier qu'il n'y a pas échange entre deux racines d'une itération à l'autre. Or, on vérifie directement à l'aide des formules (7.20) que D_k converge vers D. En effet

$$\begin{aligned} a_{ii}^{k+1} - a_{ii}^k &= 0 \text{ si } i = p, q \\ \left|a_{pp}^{k+1} - a_{pp}^k\right| &\leq t\left|a_{pq}^k\right| \leq \left|a_{pq}^k\right| \to_{k\to\infty} 0 \\ \left|a_{qq}^{k+1} - a_{qq}^k\right| &\leq t\left|a_{pq}^k\right| \leq \left|a_{pq}^k\right| \to_{k\to\infty} 0. \end{aligned}$$

Proposition 7.5 *On suppose que toutes les valeurs propres de A sont distinctes. Alors la matrice*

$$O_k = \Omega_1\Omega_2...\Omega_k$$

converge vers une matrice orthogonale dont les vecteurs colonnes sont vecteurs propres de A.

La démonstration de ce dernier point est laissé au lecteur.

Remarque 7.8 La détermination de a_{pq}^k comme indiquée dans la méthode classique est relativement coûteuse si la taille de la matrice est importante. On lui préfère souvent les choix suivants :

Méthode de Jacobi cyclique : on annule successivement tous les éléments hors diagonaux par un balayage cyclique par exemple ligne par ligne de la gauche jusqu'à la diagonale. Bien sûr, si un élément est nul, on passe au suivant.

Méthode de Jacobi avec seuil : on procède au même balayage, mais on omet d'annuler les éléments inférieurs à un certain seuil qu'on diminue à chaque balayage.

On démontre que ces procédés donnent lieu aux mêmes résultats de convergence que précédemment.

7.5 Méthode de Givens-Householder

Elle est particulièrement bien adaptée à la recherche des valeurs propres d'une matrice symétrique réelle qui sont situées dans un intervalle présélectionné ou de rang donné. Par contre, elle ne fournit pas les vecteurs propres correspondants.

7.5.1 Principe

Il y a deux étapes :

1) Etant donnée A symétrique, on détermine une matrice orthogonale Ω tel que ${}^t\Omega A\Omega$ soit tridiagonale : ceci se fait en un nombre fini d'opérations à l'aide de matrices dites de Householder.

2) On est alors ramené au calcul des valeurs propres d'une matrice tridiagonale symétrique : pour cela, on utilise une méthode de bissection, dite de Givens.

7.5.2 Description de la méthode de Givens

Soit

$$B_N = \begin{pmatrix} b_1 & c_1 & 0 & & 0 \\ c_1 & \ddots & \ddots & & \\ 0 & \ddots & \ddots & \ddots & 0 \\ & & \ddots & \ddots & c_{N-1} \\ 0 & & 0 & c_{N-1} & b_N \end{pmatrix}$$

On note $P_N(\lambda) = \det(B_N - \lambda I)$ le polynôme caractéristique de B_N. Ces polynômes ont la propriété remarquable de satisfaire à une formule simple de récurrence à 3 termes et de constituer ce que l'on appelle une suite de Sturm. Cette propriété a pour conséquence une disposition particulière des zéros de P_N par rapport à ceux de P_{N-1}.

Proposition 7.6

$$\begin{array}{rcl} P_0(\lambda) & = & 1 \\ P_1(\lambda) & = & b_1 - \lambda \\ \forall N \geq 2, P_N(\lambda) & = & (b_N - \lambda)P_{N-1}(\lambda) - c_{N-1}^2 P_{N-2}(\lambda) \end{array} \tag{7.21}$$

Ceci se démontre aisément en développant $\det(B_N - \lambda I)$ suivant la dernière ligne.

Proposition 7.7 *Les racines de $P_N(\lambda)$ sont toutes réelles, distinctes et séparées par celles de $P_{N-1}(\lambda)$. De plus, $\lim_{\lambda\to-\infty} P_N(\lambda) = +\infty$, $\forall N \geq 1$.*

Ainsi les racines de $P_1, P_2, ...$, sont disposées suivant le shéma suivant : Ceci se démontre aisément par récurrence à l'aide de (7.21). Les détails sont laissés au lecteur. On en déduit la conséquence suivante sur le nombre de racines de $P_i(\lambda)$ inférieures à un nombre réel μ.

Notation : On note $\mathcal{N}(i,\mu)$ le nombre de changements de signes dans la suite $\{+, P_1(\mu), P_2(\mu), ..., P_i(\mu)\}$ avec la convention qu'il n'y a pas de changement de signe lorsqu'un élément est nul.

Proposition 7.8 *Pour $i \geq 1$ et $\mu \in R$, $\mathcal{N}(i,\mu)$ est égal au nombre de racines de $P_i(\mu)$ qui sont strictement inférieures à μ.*

Ceci se démontre par récurrence. On peut se convaincre du bien-fondé de ce résultat à l'aide du dessin ci-dessus. On en déduit :

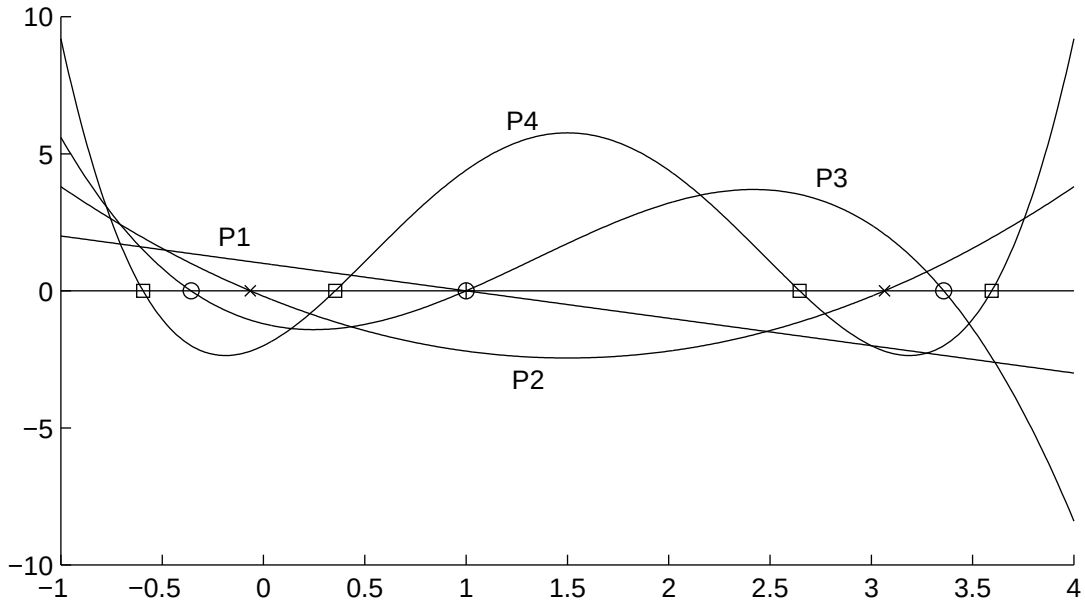

FIGURE 7.1 – Les racines des polynômes P_1, P_2, P_3, P_4 sont entrelacées

Algorithme de Givens : Soit $\lambda_1 \leq ... \leq \lambda_N$ les valeurs propres de B_N. Supposons qu'on veuille calculer λ_k pour un certain $k \in \{1, ..., N\}$.

(i) On détermine un intervalle $[a_0, b_0]$ dans lequel on est sûr de trouver λ_k.

(ii) On prend $c_0 := \frac{a_0+b_0}{2}$ et on calcule $\mathcal{N}(N, c_0)$.

– si $\mathcal{N}(N, c_0) \geq k$, alors $\lambda_k \in [a_0, c_0]$ et on pose $a_1 := a_0, b_1 := c_0$

– si $\mathcal{N}(N, c_0) < k$, alors $\lambda_k \in]c_0, b_0]$ et on pose $a_1 := c_0, b_1 := b_0$

(iii) On recommence alors la même opération avec $c_1 = \frac{a_1+b_1}{2}$, puis avec a_2, b_2, c_2, etc.

On construit ainsi une suite de segments emboités $[a_i, b_i]$ dont l'intersection est précisément la valeur cherchée λ_k. Aux erreurs d'arrondi près, on peut donc obtenir λ_k avec une précision arbitraire.

Remarque 7.9 les évaluations successives des P_j se font bien sûr à l'aide de la formule de récurrence (7.21).

Reste maintenant à décrire une technique de tridiagonalisation des matrices symétriques. Elle est en fait un cas particulier de la mise sous forme Hessenberg d'une matrice soit

$$\begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ 0 & \bullet & \bullet & \bullet & \bullet \\ \vdots & \ddots & \bullet & \bullet & \bullet \\ 0 & \dots & 0 & \bullet & \bullet \end{bmatrix}$$

où $a_{ij} = 0$ si $j < i - 1$. En effet, on remarque qu'une matrice de Hessenberg symérique est tridiagonale.

Nous allons ici décrire en détail cette technique car elle sera doublement utilisée au paragraphe suivant (factorisation QR), d'une part parce qu'on met souvent une matrice sous forme Hessenberg avant de calculer sa factorisation QR pour diminuer le coût de

calcul, d'autre part parce que l'outil est le même, à savoir l'utilisation de matrices de Householder.

Définition 7.2 *On appelle matrice de Householder une matrice H_v de la forme*

$$H_v = I - 2\frac{vv^*}{v^*v}$$

où $v \in \mathbb{C}^N$ et $v^* = {}^t\overline{v}$.

Remarque 7.10 Noter que $v^*v = {}^t\overline{v}v = \sum_{i=1}^{N} |v_i|^2$ est le carré de la norme euclidienne de v. Par contre

$$vv^* = [v_1 v_2 ... v_{\mathrm{N}}] \begin{bmatrix} \overline{v}_1 \\ \overline{v}_2 \\ \vdots \\ \overline{v}_{\mathrm{N}} \end{bmatrix} = \begin{bmatrix} |v_1|^2 & v_1\overline{v}_2 & ... & v_1\overline{v}_{\mathrm{N}} \\ v_2\overline{v}_1 & |v_2|^2 & ... & v_2\overline{v}_{\mathrm{N}} \\ \vdots & & & \vdots \\ v_{\mathrm{N}}\overline{v}_1 & ... & ... & |v_{\mathrm{N}}|^2 \end{bmatrix}$$

De plus, si on introduit le vecteur unitaire $u = \frac{v}{\|v\|_2}$, on a $H_v = H_u = I - 2uu^*$.

– On vérifie que ces matrices sont unitaires et hermitiennes. Ces matrices ont une interprétation géométrique simple. Supposons par exemple u réel et $N = 3$. Alors H_u est la matrice de la symétrie orthogonale par rapport au plan orthogonal à u.

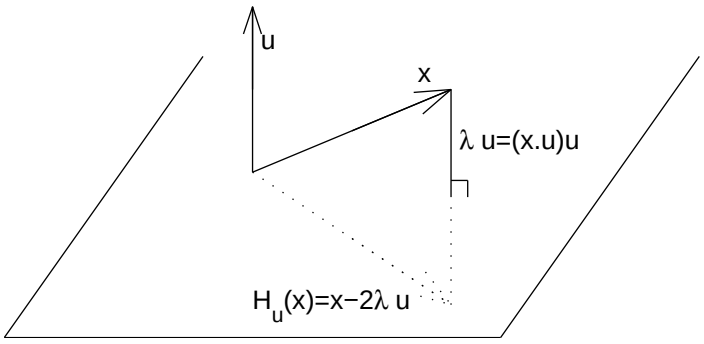

FIGURE 7.2 – $H_u(x)$ est le symétrique de x par rapport au plan orthogonal à u.

En effet le transformé X de x dans la symétrie orthogonale par rapport au plan orthogonal à u est caractérisé par

$$\begin{aligned} X - x &= -2\lambda u \quad \text{où} \quad \lambda = {}^t ux \\ \text{et donc } X &= x - 2u\left({}^t ux\right) = x - 2\left(u\,{}^t u\right)x = H_u x. \end{aligned}$$

– Diverses factorisations de matrices reposent sur l'utilisation de telles matrices de symétries et plus particulièrement de la propriété suivante.

Théorème 7.2 *Soit $a = (a_1, ..., a_N) \in \mathbb{R}^N$ avec*

$$\sum_{i=2}^{N} |a_i| \neq 0.$$

Alors il existe une matrice de Householder H_v telle que $H_v a$ ait ses $(N-1)$ dernières composantes nulles. Plus précisément, si $e_1, ..., e_N$ est la base canonique de $\mathbb{R}^N$, on a

$$H_v a = -ke_1, \quad H_{\widehat{v}} a = ke_1 \tag{7.22}$$

où

$$k = sgn(a_1) \|a\|_2, \ v = a + ke_1, \ \widehat{v} = a - ke_1. \tag{7.23}$$

Remarque 7.11 L'intérêt du Théorème 7.2 dans les techniques de factorisation est immédiat. En effet, si a est un vecteur-colonne d'une matrice M, on "fait apparaître des zéros" dans M en multipliant M par H_v puisque la colonne a est remplacée par la colonne $H_v a$ qui a des zéros partout sauf sur la première composante. Ceci sera utilisé de façon essentielle dans la factorisation QR.

Nous laissons au lecteur la vérification du Théorème 7.2. Nous allons plutôt interpréter géométriquement le Théorème 7.2. Supposons $a \in \mathbb{R}^3$. La relation (7.22) exprime que a n'est pas colinéaire à e_1. On peut alors déterminer aisément une symétrie orthogonale transformant a en un vecteur colinéaire à e_1. On voit qu'il y a deux solutions : les symétries orthogonales par rapport aux plans bissecteurs de l'angle (a, e_1). Ils sont orthogonaux aux vecteurs $v = a + \|a\|_2 e_1$ et $\widehat{v} = a - \|a\|_2 e_1$ et on a

$$H_v a = -\|a\|_2 e_1, \ H_{\widehat{v}} a = \|a\|_2 e_1$$

ce qui donne bien les relations (7.22), (7.23) dans le cas réel.

Calcul pratique : (dans le cas réel)

$$\begin{cases} \cdot \text{ Calculer } \|a\|_2, \ v := a \pm \|a\|_2 e_1 \\ \cdot \text{ Calculer } \sigma = \frac{1}{2}({}^t vv) = \|a\|_2 (\|a\|_2 \pm a_1) \\ \text{et choisir } + \text{ ou } - \text{ pour que } \sigma \text{ soit le plus grand possible.} \\ \cdot \text{ Pour appliquer } H_v \text{ à un vecteur } b, \text{ calculer } {}^t vb \text{ et } H_v b = b - \dfrac{{}^t vb}{\sigma} v. \end{cases}$$

7.5.3 Mise sous forme Hessenberg d'une matrice

Principe : On détermine des matrices de Householder $H_1, H_2, ..., H_{N-2}$ telles que $A_{N-1} = H_{N-2}...H_1 A H_1 ... H_{N-2}$ soit une matrice de Hessenberg. Puisque les matrices H_i sont hermitiennes (${}^t\overline{H} = H = H^{-1}$), A_{N-1} est semblable à A et a donc les mêmes valeurs propres.

Si, de plus, A est symétrique réelle et que les H_i sont réelles, A_{N-1} est symétrique et donc tridiagonale. Comme cas particulier, nous allons donc obtenir la technique de tridiagonalisation annoncée au début du paragraphe qui constitue la première étape de la méthode de Givens-Householder.

Décrivons maintenant le calcul de H_{k-1}. Supposons que $A_{k-1} = H_{k-2}...H_1 A H_1 ... H_{k-2}$ soit pour $k \geq 2$ de la forme :

$$A_{k-1} = \begin{bmatrix} \bullet & \bullet & \dots & & & \dots & \bullet \\ \bullet & \bullet & \dots & & & \dots & \bullet \\ 0 & \bullet & \bullet & \dots & & \dots & \bullet \\ \vdots & \ddots & \ddots & \bullet & & & \vdots \\ & & 0 & \bullet & & & \\ & & \vdots & \vdots & & & \\ & & & & & & \vdots \\ 0 & \dots & 0 & \bullet & \dots & \dots & \bullet \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ k-\text{ième ligne} \end{matrix}$$

$$k-\text{ième colonne}$$

Considérons le vecteur $a = \left(a_{k,k-1}^{k-1} a_{k+1,k-1}^{k-1}, ..., a_{\mathrm{N},k-1}^{k-1}\right)$ constitué par la queue de la $(k-1)$-ième colonne de A_{k-1}. Soit H_v l'une des matrices de Householder de dimension $N-k+1$ associées au vecteur a de $\mathbb{R}^{N-k+1}$ par le théorème 7.2 et donc telle que $H_v a$ soit colinéaire à e_k. Posons alors

$$H_{k-1} = \begin{bmatrix} I_{k-1} & 0 \\ 0 & H_v \end{bmatrix}$$

où I_{k-1} est la matrice unité de dimension $(k-1)$. Si on note

$$A_{k-1} = \begin{bmatrix} A_1^{k-1} & A_2^{k-1} \\ A_3^{k-1} & A_4^{k-1} \end{bmatrix}$$

une multiplication effectuée par blocs montre que :

$$H_{k-1}A_{k-1} = \begin{bmatrix} I_{k-1} & 0 \\ 0 & H_v \end{bmatrix} \begin{bmatrix} A_1^{k-1} & A_2^{k-1} \\ A_3^{k-1} & A_4^{k-1} \end{bmatrix} = \begin{bmatrix} A_1^{k-1} & A_2^{k-1} \\ H_v A_3^{k-1} & H_v A_4^{k-1} \end{bmatrix}$$

$$A_k = H_{k-1}A_{k-1}H_{k-1} = \begin{bmatrix} A_1^{k-1} & A_2^{k-1} \\ H_v A_3^{k-1} & H_v A_4^{k-1} \end{bmatrix} \begin{bmatrix} I_{k-1} & 0 \\ 0 & H_v \end{bmatrix} = \begin{bmatrix} A_1^{k-1} & A_2^{k-1} H_v \\ H_v A_3^{k-1} & H_v A_4^{k-1} H_v \end{bmatrix}$$

En particulier $H_v A_3^{k-1}$ est de la forme

$$H_v A_3^{k-1} = \begin{bmatrix} 0 & \cdots & 0 & \bullet \\ 0 & \cdots & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

et donc A_k a la même structure que A_{k-1}, k étant remplacé par $k+1$. Finalement, $A_{\mathrm{N}-1}$ est une matrice de Hessenberg.

Remarque 7.12 L'application du théorème 7.2 nécessite que a soit non colinéaire à e_k. Si ce n'est pas le cas, le "travail" est déjà fait et on passe immédiatement à l'étape suivante (on a alors $H_k = I$).
Si A est symétrique réelle, les H_i sont réelles. Dans ce cas, on se limitera bien sûr au calcul des éléments sur et au-dessous de la diagonale, le reste s'en déduisant par symétrie.

7.6 La méthode QR

C'est probablement la méthode la plus couramment utilisée pour trouver toutes les valeurs propres d'une matrice quelconque, notamment non symétrique. L'ensemble des vecteurs propres peut être également obtenu.
Commençons par un résultat de factorisation :

Théorème 7.3 *Soit A une matrice à coefficients complexes (respectivement réels). Alors, il existe Q unitaire (respectivement orthogonale) et R triangulaire supérieure telle que $A = QR$.*

Démonstration et algorithme de construction de Q et R.
On utilise des matrices de Householder $H_1, H_2, ..., H_{\mathrm{N}-1}$ telles que $R = H_{\mathrm{N}-1}H_{\mathrm{N}-2}...H_1 A$ soit triangulaire supérieure. Posant alors $Q = (H_{\mathrm{N}-1}H_{\mathrm{N}-2}...H_1)^{-1} = H_1 H_2 ... H_{\mathrm{N}-1}$ on obtient la factorisation voulue.

La construction des matrices successives H_k repose sur le théorème 7.2. Supposons $A_k = H_{k-1}H_{k-2}...H_1A$ de la forme suivante pour $k \geq 1$:

$$A_k = \begin{bmatrix} \bullet & \bullet & \dots & & & \dots & \bullet \\ 0 & \bullet & \bullet & \dots & & \dots & \bullet \\ \vdots & \ddots & \ddots & \bullet & & & \vdots \\ & & 0 & \bullet & & & \\ & & \vdots & \vdots & & & \\ & & & & & & \vdots \\ 0 & \dots & 0 & \bullet & \dots & \dots & \bullet \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ k\text{-ième ligne} \end{matrix}$$

k-ième colonne

Notons $a = \left(a^k_{k,k}, a^k_{k+1,k}, ..., a^k_{N,k}\right)$ la queue de la k-ième colonne de A_k.
Soit H_v l'une des matrices de Householder de dimension $N-k+1$ associées au vecteur a selon le théorème 7.2 et donc telle que $H_v a$ soit colinéaire à e_k. Posons alors :

$$H_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & H_v \end{bmatrix}$$

où I_{k-1} est la matrice unité de dimension $k-1$. Si on note

$$A_k = \begin{bmatrix} A^k_1 & A^k_2 \\ 0 & A^k_3 \end{bmatrix}$$

la décomposition par blocs correspondants de A_k, une multiplication par blocs donne :

$$A_{k+1} = H_v A_k = \begin{bmatrix} A^k_1 & A^k_2 \\ 0 & H_v A^k_4 \end{bmatrix}$$

Compte tenu de $H_v A = 0$, la matrice $H_v A^k_4$ est de la forme :

$$\begin{bmatrix} \bullet & \bullet & \cdots & \cdots \\ 0 & \bullet & \cdots & \cdots \\ \vdots & \vdots & & \\ 0 & \bullet & \cdots & \cdots \end{bmatrix}$$

Donc A_{k+1} a la même forme que A_k avec k remplacé par $k+1$. Itérant jusqu'à $k = N-1$, on obtient une matrice triangulaire.

Remarque 7.13 Si a est colinéaire à e_k, on passe bien sûr directement à l'étape suivante en posant $A_{k+1} := A_k$.

Une nouvelle méthode directe de résolution d'un système linéaire :
Soit à résoudre $AX = b$. Appliquant la décomposition ci-dessus, ce système équivaut à :

$$H_{N-1}...H_1AX = H_{N-1}...H_1b$$

soit

$$RX = H_{N-1}...H_1b$$

où R est triangulaire supérieure. On peut alors le résoudre par remontée. On obtient ainsi une nouvelle méthode de résolution directe pour les systèmes linéaires.
Pour N grand, le nombre d'opérations élémentaires de la factorisation QR est de l'ordre de $\frac{4}{3}N^3$ soit deux fois plus important que pour une factorisation LU de Gauss. En

contrepartie, il faut noter qu'aucune stratégie du pivot n'est nécessaire et surtout, comme les facteurs H_k sont unitaires, le conditionnement de R est égal à celui de A. (si H unitaire, $cond_2(HA) = cond_2(A)$ voir le chapitre précédent). Dans certaines situations délicates, ceci peut constituer un avantage déterminant de cette méthode de résolution sur celle de Gauss.

7.6.1 Algorithme QR de recherche de valeurs propres

Etant donnée A, on forme les suites de matrices A_i, Q_i, R_i de la façon suivante :

$$\left[\begin{array}{l} A_0 := A \\ \text{Pour } i = 0, 1, ... \\ (1) \text{ On effectue la décomposition } A_i = Q_i R_i \text{ avec } Q_i \text{ unitaire et } R_i \\ \text{triangulaire supérieure} \\ (2)\ A_{i+1} := R_i Q_i \end{array}\right. \tag{7.24}$$

On remarque que $R_i = Q_i^{-1} A_i$ et donc $A_{i+1} = Q_i^{-1} A_i Q_i$, ainsi A_{i+1} est semblable à A_i et, par récurrence, tous les A_i sont semblables à A (et ont donc les mêmes valeurs propres).

Ainsi, si A_i tend vers une matrice triangulaire lorsque i devient grand, on pourra lire sur la diagonale les valeurs propres de A. C'est ce qui se produit au moins dans le cas fréquent suivant.

Théorème 7.4 *On suppose les valeurs propres de A toutes de modules différents, soit*

$$|\lambda_1| > |\lambda_2| > ... > |\lambda_N|\,.$$

Il existe donc une matrice inversible P telle que $P^{-1}AP$ soit la matrice diagonale $diag\,(\lambda_1, \lambda_2, ..., \lambda_N)$; on fait de plus l'hypothèse technique que P admet une factorisation LU.

Alors la suite de matrices A_k définie par (7.24) vérifie

$$\begin{array}{rcl} \forall\ 1 \leq j < i \leq N & \lim\limits_{k\to\infty} (A_k)_{ij} = 0 \\ \forall\ 1 \leq i \leq N & \lim\limits_{k\to\infty} (A_k)_{ii} = \lambda_i. \end{array}$$

Remarque 7.14 (1) L'hypothèse technique ci-dessus n'est pas vraiment essentielle. Si elle n'est pas satisfaite, la méthode QR converge cependant, mais les λ_i ne sont plus rangées dans l'ordre décroissant des modules.

(2) Si les modules des valeurs propres ne sont pas tous différents, (A_i) tend (en général) vers une matrice triangulaire par blocs, un bloc correspondant à un module. Cette situation se présente en particulier pour des matrices réelles ayant des valeurs propres non réelles (elles sont alors conjuguées deux à deux).

(3) On montre facilement que si A est de Hessenberg ou une matrice bande hermitienne, alors les A_i ont la même structure. Il en résulte qu'on commence toujours par réduire une matrice sous forme Hessenberg (ou sous une forme tridiagonale si A est hermitienne) avant d'appliquer l'algorithme QR. Le coût de calcul est alors considérablement réduit.

(4) L'algorithme QR permet aussi le plus souvent l'obtention des vecteurs propres. Il est difficiles de donner des énoncés suffisamment généraux, mais on peut s'en convaincre à l'aide du raisonnement approximatif suivant. Notons $\Omega_k = Q_1 Q_2 ... Q_k$. On a alors :

$$A_k = \Omega_k^{-1} A \Omega_k.$$

Supposons que pour k grand, $A_k = T$ soit exactement triangulaire. Alors :

$$A\Omega_k = \Omega_k T$$

En particulier

$$A\Omega_k e_1 = \Omega_k T e_1 = \lambda_1 \Omega_k e_1$$

et $\Omega_k e_1$, c'est-à-dire le premier vecteur colonne de Ω_k, est vecteur propre de A pour la valeur propre de λ_1. Ensuite, on vérifie que pour $i \geq 2$, le vecteur $q^i = \left(q^i_j\right)_{j=1,\ldots,N}$ est vecteur propre pour λ_i s'il vérifie

$$\left\{ \begin{array}{l} q^i_j = 0 \quad \forall j = i+1, ..., N \\ q^i_i = 1 \\ q^i_j = -\left(t_{j,j+1} q^i_{j+1} + ... + t_{ji} q^i_i\right) / \left(\lambda_j - \lambda_i\right) \quad \forall j = i-1, ...1. \end{array} \right.$$

Si on ne s'intéresse qu'à quelques vecteurs propres, on peut aussi utiliser la méthode de la puissance itérée avec translation une fois déterminée la valeur propre.

(5) Si la matrice initiale A est réelle, les matrices A_i sont réelles et ne peuvent converger vers une matrice triangulaire que si toutes les valeurs propres sont réelles. Pour atteindre les valeurs propres complexes, on peut procéder à une translation d'argument complexe pour "séparer" les modules. Il faut alors travailler en arithmétique complexe. Certains algorithmes couplant deux translations complexes conjuguées successives permettent de s'en passer. Nous renvoyons à la littérature spécialisée pour une description de ces méthodes plus sophistiquées.

(6) Algorithme QR avec translations : déjà évoqué au point (5) ci-dessus, il permet surtout d'accélérer très sensiblement la vitesse de convergence de l'algorithme. En général, la convergence est linéaire : les éléments A_{ij} tendent vers 0 comme $\left(\frac{\lambda_i}{\lambda_j}\right)^k$ sous les hypothèses du théorème 7.4. Si la matrice A est sous forme Hessenberg, la vitesse de convergence est donc régie par les rapports $\frac{\lambda_i}{\lambda_{i-1}}$. On s'arrange donc pour effectuer des translations pour diminuer ces rapports. Ainsi, à chaque étape, on choisit un réel s_k et A_{k+1} est déterminé comme suit

$$\begin{array}{rcl} A_k - s_k I & = & Q_k R_k \\ A_{k+1} & = & R_k Q_k + s_k I = Q_k^{-1} A_k Q_k. \end{array}$$

On peut choisir s_k pour que le rapport $\left|\lambda_{\mathrm{N}} - s_k\right| / \left|\lambda_{\mathrm{N}-1} - s_k\right|$ soit aussi petit que possible. Un choix fréquent consiste à choisir pour s_k l'une des valeurs propres de la matrice

$$\left[\begin{array}{cc} a^k_{\mathrm{N}-1,\mathrm{N}-1} & a^k_{\mathrm{N}-1,\mathrm{N}} \\ a^k_{\mathrm{N},\mathrm{N}-1} & a^k_{\mathrm{N},\mathrm{N}} \end{array} \right]$$

Pour des matrices tridiagonales symétriques, ceci conduit souvent à une convergence cubique de $a^k_{\mathrm{N},\mathrm{N}-1}$ vers 0. Lorsque ce terme est jugé suffisamment petit, on peut poursuivre les calculs en laissant tomber la N-ième ligne et la N-ième colonne de A^k.

(7) Pour terminer, insistons sur le fait que la méthode QR appliquée avec les améliorations évoquées ci-dessus (réduction préalable des matrices, translations appropriées) est une méthode très performante. Même pour une matrice symétrique, les expériences montrent qu'elle est environ 10 fois plus rapide que la méthode de Jacobi lorsqu'on ne calcule que les valeurs propres et encore 4 fois plus rapide si on calcule à la fois valeurs propres et vecteurs propres.

7.7 Exercices du chapitre 7

Exercice 7.1 Que se passe-t-il dans la méthode de la puissance itérée lorsque $\lambda_{N-1} = \lambda_N$, $\lambda_{N-1} = -\lambda_N$, puis $\lambda_{N-1} = \overline{\lambda_N}$ (ce cas correspond à celui d'une matrice réelle dont la valeur propre de plus grand module est complexe).

Exercice 7.2 Vérifier les relations (7.22), (7.23).

Exercice 7.3 Écrire un algorithme de tridiagonalisation d'une matrice symétrique réelle.

Exercice 7.4 Effectuer les deux premières itérations de la méthode de Jacobi pour la matrice

$$A = \begin{pmatrix} 2 & 3 & 1 \\ 3 & 2 & 2 \\ 1 & 2 & 3 \end{pmatrix}.$$

Exercice 7.5 On reprend la suite $\{+, P_1(\mu), P_2(\mu), ..., P_i(\mu)\}$ qui intervient dans la méthode de Givens et on note $\mathcal{M}(i,\mu)$ le nombre de paires consécutives de même signe dans cette suite. Montrer que pour tout entier i, $\mathcal{M}(i,\mu) + \mathcal{N}(i,\mu) = i$ et en déduire que $\mathcal{M}(i,\mu)$ représente le nombre de racines du polynôme P_i qui sont $\geq \mu$.