Méthodes numériques appliquées à la conception par éléments finis

David Dureisseix

▶ **To cite this version:**

David Dureisseix. Méthodes numériques appliquées à la conception par éléments finis. DEA. Montpellier, France. 2008, pp.98. cel-00351713v2

HAL Id: cel-00351713
https://cel.hal.science/cel-00351713v2

Submitted on 3 Jan 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - ShareAlike 4.0 International License

Méthodes numériques appliquées à la conception par éléments finis

David Dureisseix

UNIVERSITÉ MONTPELLIER 2
SCIENCES ET TECHNIQUES

Université de Montpellier 2 — 2013
Ce document est sous licence Creative Commons 3.0 France:
— paternité;
— pas d'utilisation commerciale;
— partage des conditions initiales à l'identique;
http://creativecommons.org/licenses/by-nc-sa/3.0/deed.fr

Table des matières

1 Résolution des systèmes linéaires

1.1 Conditionnement **11**
1.1.1 Définition . . . 11
1.1.2 Influence des perturbations . . . 12

1.2 Techniques de stockage et coût **14**
1.2.1 Paramètres influents . . . 14
1.2.2 Coût . . . 14

1.3 Méthodes directes **16**
1.3.1 Méthode d'élimination de Gauss . . . 16
1.3.2 Factorisation de Cholesky . . . 18
1.3.3 Factorisation de Crout . . . 18
1.3.4 Lien avec la condensation de Schur . . . 19
1.3.5 Autres techniques . . . 20

1.4 Quelques particularités en éléments finis **20**
1.4.1 Conditions aux limites en déplacement . . . 20
1.4.2 Relations linéaires entre degrés de liberté . . . 22
1.4.3 Système singulier : semi défini positif . . . 23

1.5 Méthodes itératives stationnaires **24**
1.5.1 Méthode de Jacobi . . . 25
1.5.2 Méthode de Gauss-Seidel . . . 25
1.5.3 Méthode de relaxation . . . 26
1.5.4 Méthode d'Uzawa . . . 27

1.6 Méthodes itératives non-stationnaires **28**
1.6.1 Méthode générique de projection . . . 28
1.6.2 Méthode du gradient . . . 29
1.6.3 Méthode du gradient conjugué . . . 29
1.6.4 Préconditionnement . . . 30
1.6.5 Autres méthodes itératives . . . 30

1.7 Critères d'arrêt **31**

2 Systèmes aux valeurs propres

2.1 Préliminaires **33**
2.1.1 Rappels . . . 33
2.1.2 Domaine d'emploi des différentes méthodes . . . 34

2.2 Méthode de Jacobi **34**

2.3 Méthode des puissances **36**

2.4 Méthode des itérations inverses **38**

2.5 Méthode des itérations inverses avec décalage spectral 39
2.5.1 Principe et convergence . . . 39
2.5.2 Exemples d'utilisation dans un code éléments finis . . . 39
2.6 Transformation de Householder 40
2.6.1 Factorisation QR . . . 40
2.6.2 Détermination des valeurs propres par la méthode QR . . . 41
2.6.3 Forme de Hessenberg . . . 42
2.7 Exemple d'utilisation dans un code éléments finis 44
2.7.1 Réduction dans un sous-espace . . . 44
2.7.2 Technique itérative . . . 44
2.8 Méthode de Lanczos 44

3 Résolution de systèmes non-linéaires particuliers

3.1 Non-linéarités matérielles de type plasticité 47
3.1.1 Formulation du problème . . . 47
3.1.2 Résolution pour un incrément de temps . . . 48
3.1.3 Méthode de la plus grande pente . . . 48
3.1.4 Méthodes de gradient conjugué . . . 49
3.1.5 Méthode de Newton-Raphson . . . 49
3.1.6 Méthode de Newton-Raphson modifiée . . . 49
3.1.7 Mises à jour de type Quasi-Newton . . . 50

4 Résolution de systèmes différentiels

4.1 Équation scalaire du premier ordre 53
4.1.1 Problème de Cauchy . . . 53
4.1.2 Ré-écriture du problème . . . 53
4.1.3 Schémas explicites à un pas . . . 54
4.1.4 Quelques problèmes pratiques . . . 55
4.1.5 A-stabilité . . . 56
4.1.6 Méthode des trapèzes généralisée . . . 57
4.1.7 Méthode à pas multiples : Adams . . . 59
4.1.8 Méthode de prédiction-correction basée sur Adams . . . 61
4.1.9 Méthode de Runge-Kutta . . . 61
4.1.10 Autres méthodes . . . 63
4.2 Systèmes différentiels du premier ordre 64
4.2.1 Emploi des méthodes précédentes . . . 64
4.2.2 Influence de l'ordre d'intégration . . . 65
4.3 Systèmes différentiels du deuxième ordre 66
4.3.1 Méthode de Newmark . . . 66
4.3.2 Méthode de Gear implicite à deux pas . . . 71
4.3.3 θ-méthode . . . 71
4.3.4 α-HHT . . . 72
4.3.5 Autres méthodes . . . 72

4.4 Intégration de modèles de comportement **72**
4.4.1 Exemple de modèle de plasticité 73
4.4.2 Méthodes avec retour radial 74
4.4.3 Modèle de viscoplasticité 76
4.5 Cas de la dynamique non régulière **77**

Bibliographie

A Méthodes particulières
A.1 Factorisation de cholesky et orthogonalisation d'un sous-espace **87**
A.2 Technique d'accélération d'Aitken **87**

B Bornage par les valeurs propres élémentaires
B.1 Système aux valeurs propres généralisé **89**
B.2 Système aux valeurs propres **90**

C Repères biographiques

Index

Introduction

Ce document de travail (avec renvoi à des ressources de références s'il y a besoin d'approfondir) est utile à ceux qui sont intéressés ou amenés à utiliser des méthodes de simulation (essentiellement par éléments finis) dans le but de dimensionner des structures mécaniques.

Il ne s'agit pas pour autant d'un document sur les éléments finis, puisqu'on suppose que les problèmes ont déjà été formulés dans ce cadre, et que l'on se place en aval, lorsque la résolution des problèmes éléments finis est en jeu. On se ramènera ainsi fréquemment aux techniques effectivement utilisées dans des codes de calcul industriels. Pour cela, on sera amené à utiliser des méthodes classiques d'algèbre linéaire, de calcul matriciel, adaptées aux calculs sur ordinateurs (algorithmique).

Selon les problèmes qui se posent, on est amené à utiliser un grand nombre d'outils différents mais ce document ne se veut pas exhaustif. Il ne traite en particulier pas les cas suivants :

— la recherche de racine de $f(\mathrm{U}) = 0$;
— les problèmes de minimum non quadratiques (bisection, point fixe, Newton, Newton-Raphson, quasi-Newton, sécante, BFGS) ;
— les problèmes non linéaires de type grands déplacements ou contact unilatéral.

Structuré en quatre chapitres, il aborde les techniques de résolution de grands systèmes linéaires (de façon directe ou itérative), de problèmes de recherche de valeurs et vecteurs propres, et enfin de systèmes différentiels.

Pour un aperçu moins détaillé mais plus large des méthodes numériques, le lecteur pourra se reporter à [49].

Il semble aussi utilise de préciser que ce document est en perpétuelle évolution et que de ce fait, son organisation ainsi que sa table des matières peuvent afficher quelque incohérence.

En cas d'erreur

Les différents algorithmes et techniques présentés ne sont pas garantis contre les erreurs et autres coquilles... De la même façon, tout document est perfectible, et celui-ci ne fait pas exception à la règle. Aussi, si vous en trouvez, signalez-les : c'est aussi une méthode itérative... pour réduire le nombre de bugs. J'ai déjà eu des retours qui m'ont permis d'améliorer la chose. Merci donc en particulier à Martin Genet.

Notations

Un vecteur est représenté par une lettre romane minuscule (comme $\mathbf{x}$), une matrice par une lettre romane majuscule (comme $\mathbf{A}$), un scalaire par une lettre grecque minuscule (comme α). La composante i du vecteur $\mathbf{x}$ est notée x_i (c'est un scalaire) et la composante i, j de la matrice $\mathbf{A}$, $\mathrm{A}_{i,j}$: c'est un scalaire également. La colonne j de $\mathbf{A}$, par exemple, sera classiquement notée $\mathbf{A}_{1:n,j}$. Les vecteurs de la base dans laquelle les différentes matrices sont écrites seront notés $\mathbf{e}_i$. À chaque fois, l'itéré numéro i est désigné par une mise en exposant (comme $\mathbf{x}^{(i)}$) et en général, une même lettre désignera en fait un même emplacement en mémoire ; par exemple, $\mathbf{x}^{(i)}$ écrase $\mathbf{x}^{(i-1)}$ en mémoire.

Pour approfondir

Pour des renseignements généraux complémentaires sur les techniques employées, il est utile de ce référer par exemple à [6, 28] et pour les algorithmes et leur implantation, aux librairies de netlib.

De façon générale, la bibliographie présentée regroupe deux types de références : les anciennes, voire très anciennes, en tant que base historique, et les plus récentes, en tant qu'ouvrages contemporains. Les biographies utilisées se sont largement inspirées des bases de données disponibles aux adresses suivantes :

— http://www-history.mcs.st-andrews.ac.uk/
— http://www.math.bme.hu/html/index.html
— http://www.wikipedia.org

Cadre de travail

Actuellement, et pour ce qui nous intéresse, nous avons tout intérêt à utiliser des librairies scientifiques existantes de façon intensive : on ne peut prétendre à reprogrammer de façon plus efficace et plus robuste des algorithmes standards que des spécialistes de ce domaine et qui travaillent sur ces librairies depuis un grand nombre d'années. Il est cependant nécessaire de connaître le principe d'un certain nombre de ces méthodes, leur domaine d'emploi, leur limite et d'avoir une idée des cas à problème pour aider au choix d'une méthode. C'est dans cette direction que se place ce document.

Parmi ces librairies, en grande partie du domaine public, certaines émergent ou commencent à émerger comme des standards de fait. Par exemple, la librairie LAPACK (Linear Algebra PACKage) [2] :

— http://www.netlib.org/lapack/

qui succède en grande partie aux anciennes libraries EISPACK, pour la résolution de systèmes aux valeurs propres :

— http://www.netlib.org/eispack/

et LINPACK pour la résolution de systèmes linéaires :

— http://www.netlib.org/linpack/

Sa description est celle de la FAQ (*Frequently Asked Questions*) :

> LAPACK provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. The associated matrix factorizations (LU, Cholesky, QR, SVD, Schur, generalized Schur) are also provided, as are related computations such as reordering of the Schur factorizations and estimating condition numbers. Dense and banded matrices are handled, but not general sparse matrices. In all areas, similar functionality is provided for real and complex matrices, in both single and double precision.

Citons aussi sa version parallélisée, ScaLAPACK (Scalable LAPACK, [8]) :

> The ScaLAPACK project is a collaborative effort involving several institutions (Oak Ridge National Laboratory, Rice University, University of California, Berkeley, University of California, Los Angeles, University of Illinois, University

> of Tennessee, Knoxville), and comprises four components : dense and band matrix software (ScaLAPACK), large sparse eigenvalue software (PARPACK and ARPACK), sparse direct systems software (CAPSS and MFACT), preconditioners for large sparse iterative solvers (ParPre). The ScaLAPACK (or Scalable LAPACK) library includes a subset of LAPACK routines redesigned for distributed memory MIMD parallel computers. It is currently written in a Single-Program-Multiple-Data style using explicit message passing for interprocessor communication. It assumes matrices are laid out in a two-dimensional block cyclic decomposition. Like LAPACK, the ScaLAPACK routines are based on block-partitioned algorithms in order to minimize the frequency of data movement between different levels of the memory hierarchy. (For such machines, the memory hierarchy includes the off-processor memory of other processors, in addition to the hierarchy of registers, cache, and local memory on each processor.) The fundamental building blocks of the ScaLAPACK library are distributed memory versions (PBLAS) of the Level 1, 2 and 3 BLAS, and a set of Basic Linear Algebra Communication Subprograms (BLACS) for communication tasks that arise frequently in parallel linear algebra computations. In the ScaLAPACK routines, all interprocessor communication occurs within the PBLAS and the BLACS. One of the design goals of ScaLAPACK was to have the ScaLAPACK routines resemble their LAPACK equivalents as much as possible.

D'un niveau d'utilisation plus bas, et utilisées dans les librairies précédentes, on trouve aussi des utilitaires intéressants dans les librairies ARPACK :

— http://www.caam.rice.edu/software/ARPACK/

> ARPACK is a collection of Fortran77 subroutines designed to solve large scale eigenvalue problems. The package is designed to compute a few eigenvalues and corresponding eigenvectors of a general $n \times n$ matrix $\mathbf{A}$. It is most appropriate for large sparse or structured matrices $\mathbf{A}$ where structured means that a matrix-vector product $w \leftarrow \mathrm{A}v$ requires order n rather than the usual order n^2 floating point operations. This software is based upon an algorithmic variant of the Arnoldi process called the Implicitly Restarted Arnoldi Method (IRAM). When the matrix $\mathbf{A}$ is symmetric it reduces to a variant of the Lanczos process called the Implicitly Restarted Lanczos Method (IRLM). These variants may be viewed as a synthesis of the Arnoldi/Lanczos process with the Implicitly Shifted QR technique that is suitable for large scale problems. For many standard problems, a matrix factorization is not required. Only the action of the matrix on a vector is needed.

et BLAS (Basic Linear Algebra Subprograms) :

— http://www.netlib.org/blas/

> The BLAS (Basic Linear Algebra Subprograms) are high quality "building block" routines for performing basic vector and matrix operations. Level 1 BLAS do vector-vector operations, Level 2 BLAS do matrix-vector operations, and Level 3 BLAS do matrix-matrix operations. Because the BLAS are efficient,

portable, and widely available, they are commonly used in the development of high quality linear algebra software, LINPACK and LAPACK for example.

Les développements concernant ces librairies sont toujours en cours, en particulier les versions creuses SPBLAS (SParse BLAS) et parallèles PBLAS (Parallel BLAS). Enfin, des librairies scientifiques généralistes sont aussi disponibles, comme la GSL (GNU Scientific Library) :

— http://www.gnu.org/software/gsl/

1 Résolution des systèmes linéaires

On est souvent, voire tout le temps, amené à résoudre des systèmes linéaires de la forme $\mathbf{Ax} = \mathbf{b}$ pour la simulation sur ordinateur ; par exemple en éléments finis [54], une analyse statique conduit à résoudre un système que l'on notera $\mathbf{Ku} = \mathbf{f}$ où $\mathbf{K}$ est la matrice de rigidité, $\mathbf{u}$ le déplacement en certains points de la structure et $\mathbf{f}$, les forces généralisées.

Une analyse non-linéaire implicite ou la recherche d'une base modale sont souvent basées sur la *brique de base* qu'est la résolution de systèmes linéaires. Dans ces cas de figure, comme dans d'autres, les matrices $\mathbf{A}$ et $\mathbf{b}$ correspondantes ont souvent des particularités dont il sera judicieux de tirer parti pour les avantages, ou qu'il faudra avoir en tête pour les inconvénients. Par exemple :

1. une grande dimension $\dim \mathbf{A} = n$, classiquement de 10^3 à 10^6, donc :
   — on n'inverse pas $\mathbf{A}$ (ce serait résoudre n systèmes linéaires) ;
   — il faut prendre en compte des considérations de stockage adapté, à la fois pour réduire les ressources en mémoire et le nombre d'opérations à effectuer.
2. la nécessité de traiter plusieurs second membres :
   — simultanément ;
   — successivement.
3. des propriétés souvent rencontrées :
   — des coefficients réels, une matrice $\mathbf{A}$ carrée,
   — $\mathbf{A}$ inversible $\Leftrightarrow$ $\mathbf{A}$ régulière $\Leftrightarrow$ $\det \mathbf{A} \neq 0$
   — $\mathbf{A}$ symétrique $\Leftrightarrow$ $\mathbf{A} = \mathbf{A}^\top$ avec $\forall \mathbf{x}, \mathbf{y}, \mathbf{x} \cdot \mathbf{y} = \mathbf{x}^\top \mathbf{y}, \mathbf{x}^\top \mathbf{A} \mathbf{y} = \mathbf{y}^\top \mathbf{A}^\top \mathbf{x}$
   — $\mathbf{A}$ positive $\Leftrightarrow$ $\forall \mathbf{x}, \mathbf{x}^\top \mathbf{A} \mathbf{x} \geqslant 0$
   — $\mathbf{A}$ définie $\Leftrightarrow$ $\mathbf{x}^\top \mathbf{A} \mathbf{x} = 0 \Rightarrow \mathbf{x} = \mathbf{0}$
   — $\mathbf{A}$ mal conditionnée, typiquement $\operatorname{cond} \mathbf{K} = \mathcal{O}(\frac{1}{h^2})$ où h est la taille de la maille en éléments finis massifs.

Pour plus d'informations concernant ces particularités et les techniques mises en œuvre, on pourra se référer à [4, 59].

1.1 Conditionnement

1.1.1 Définition

Pour $\mathbf{A}$ symétrique et inversible,

$$\operatorname{cond} \mathbf{A} = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\| \tag{1.1}$$

Avec une norme matricielle cohérente avec celle définie pour les vecteurs :

$$\|\mathbf{A}\| = \sup_{\|\mathbf{x}\| \neq 0} \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|} \tag{1.2}$$

où $\|\mathbf{x}\|^2 = \mathbf{x}^\top \mathbf{x} = \sum_i x_i^2$ est la norme euclidienne, on a :

$$\operatorname{cond} \mathbf{A} = \frac{|\lambda|_{\max}}{|\lambda|_{\min}} \tag{1.3}$$

où les λ sont les valeurs propres de $\mathbf{A}$. En général, il est très coûteux d'atteindre le conditionnement de $\mathbf{A}$, par contre un certains nombre de techniques peuvent permettre de l'estimer. On pourra en particulier s'inspirer des techniques développées dans le chapitre concernant la recherche de valeurs propres. Un conditionnement grand conduit à des problèmes de *précision* (cumul d'arrondis numériques). Pour les illustrer, on est amené à considérer un système perturbé (par des perturbations généralement supposées petites) $\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$.

Exemple 1.1 Considérons le système $\mathbf{Ax} = \mathbf{b}$ suivant :

$$\begin{bmatrix} 1 & 1 \\ 1 & 1+\varepsilon \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \Rightarrow \mathbf{x} = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \tag{1.4}$$

Le polynôme caractéristique $\lambda^2 - (2+\varepsilon)\lambda + \varepsilon = 0$ permet de déterminer les valeurs propres, à savoir :

$$\lambda_{1,2} = \frac{2 + \varepsilon \pm \sqrt{4+\varepsilon^2}}{2} \tag{1.5}$$

Avec ε petit, le conditionnement est donc très grand puisque $\operatorname{cond}\mathbf{A} \approx 4/\varepsilon$. Considérons maintenant le système $\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$ perturbé tel que :

$$\begin{bmatrix} 1 & 1 \\ 1 & 1+\varepsilon \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2+\alpha \end{pmatrix} \Rightarrow \mathbf{x} = \begin{pmatrix} 2 - \frac{\alpha}{\varepsilon} \\ \frac{\alpha}{\varepsilon} \end{pmatrix} \tag{1.6}$$

Ce résultat entraîne la discussion suivante :

- si $\alpha \ll \varepsilon \ll 1$ alors $\mathbf{x} \approx [2\ 0]^\top$;
- si $\alpha \approx \varepsilon \ll 1$, ce qui est tout à fait possible avec les arrondis numériques, la solution est très différente puisque $\mathbf{x} \approx [1\ 1]^\top$.

Exemple 1.2 Examinons le système suivant :

$$\mathbf{A} = \begin{bmatrix} 1{,}2969 & 0{,}8648 \\ 0{,}2161 & 0{,}1441 \end{bmatrix} \quad \mathbf{b} = \begin{pmatrix} 0{,}8642 \\ 0{,}1440 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} 0{,}8642 \\ 0{,}1440 \end{pmatrix} \tag{1.7}$$

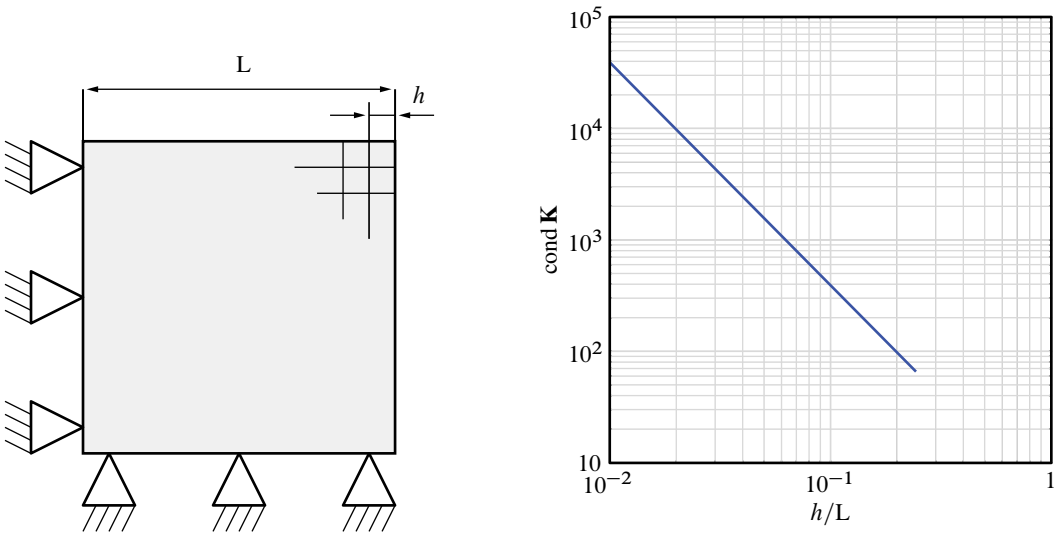
Le conditionnement de $\mathbf{A}$ est $3{,}3 \cdot 10^8$; le résidu est petit puisque $\mathbf{b} - \mathbf{Ax} = [-10^{-8}\ \ 10^{-8}]^\top$ alors que la solution exacte est très différente de $\mathbf{x}$ puisque c'est $[2\ \ -2]^\top$. Il semble donc que le conditionnement du système soit très fortement lié à la qualité de la solution (en terme d'influence des arrondis numériques) que l'on puisse espérer avoir.

Dans le cas des éléments finis massifs (autre que plaques en flexion, poutres...), le conditionnement varie en $\mathcal{O}(1/h^2)$. La figure 1.1 illustre cela dans un cas bidimensionnel très simple. Le conditionnement a été obtenu en calculant explicitement la plus grande et la plus petite valeur propre de $\mathbf{K}$ avec les méthodes qui seront présentées ultérieurement.

1.1.2 Influence des perturbations

Considérons le système $\mathbf{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$. Si $\tilde{\mathbf{b}} = \mathbf{b} + \boldsymbol{\delta}\mathbf{b}$ avec $\|\boldsymbol{\delta}\mathbf{b}\| \ll \|\mathbf{b}\|$, a-t-on $\|\boldsymbol{\delta}\mathbf{x}\| \ll \|\mathbf{x}\|$ pour $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\delta}\mathbf{x}$? Calculons simplement, en utilisant l"inégalité de Milkowski :

$$\mathbf{Ax} = \mathbf{b} \ \Rightarrow\ \|\mathbf{Ax} = \mathbf{b}\| \ \Rightarrow\ \|\mathbf{A}\| \cdot \|\mathbf{x}\| \geqslant \|\mathbf{b}\| \tag{1.8}$$



(a) maillage (b) conditionnement

Figure 1.1 — Conditionnement en éléments finis

autrement dit :

$$\frac{1}{\|\mathbf{x}\|} \leqslant \frac{\|\mathbf{A}\|}{\|\mathbf{b}\|} \tag{1.9}$$

et maintenant :

$$\mathbf{A}(\mathbf{x}+\boldsymbol{\delta}\mathbf{x}) = \mathbf{b}+\boldsymbol{\delta}\mathbf{b} \;\Rightarrow\; \mathbf{A}\boldsymbol{\delta}\mathbf{x} = \boldsymbol{\delta}\mathbf{b} \;\Rightarrow\; \boldsymbol{\delta}\mathbf{x} = \mathbf{A}^{-1}\boldsymbol{\delta}\mathbf{b} \;\Rightarrow\; \|\boldsymbol{\delta}\mathbf{x}\| \leqslant \|\mathbf{A}^{-1}\|\cdot\|\boldsymbol{\delta}\mathbf{b}\| \tag{1.10}$$

soit :

$$\|\boldsymbol{\delta}\mathbf{x}\| \leqslant \|\mathbf{A}^{-1}\| \cdot \|\boldsymbol{\delta}\mathbf{b}\| \tag{1.11}$$

par conséquent :

$$\frac{\|\boldsymbol{\delta}\mathbf{x}\|}{\|\mathbf{x}\|} \leqslant \operatorname{cond}\mathbf{A}\frac{\|\boldsymbol{\delta}\mathbf{b}\|}{\|\mathbf{b}\|} \tag{1.12}$$

Considérons maintenant le système perturbé $\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \mathbf{b}$. Si $\tilde{\mathbf{A}} = \mathbf{A}+\boldsymbol{\delta}\mathbf{A}$ et $\tilde{\mathbf{x}} = \mathbf{x}+\boldsymbol{\delta}\mathbf{x}$, alors :

$$(\mathbf{A}+\boldsymbol{\delta}\mathbf{A})(\mathbf{x}+\boldsymbol{\delta}\mathbf{x}) = \mathbf{b} \tag{1.13}$$

implique :

$$\mathbf{A}\boldsymbol{\delta}\mathbf{x} + \boldsymbol{\delta}\mathbf{A}(\mathbf{x}+\boldsymbol{\delta}\mathbf{x}) = \mathbf{0} \tag{1.14}$$

soit :

$$\boldsymbol{\delta}\mathbf{x} = -\mathbf{A}^{-1}\boldsymbol{\delta}\mathbf{A}(\mathbf{x}+\boldsymbol{\delta}\mathbf{x}) \tag{1.15}$$

ce qui entraîne finalement :

$$\|\boldsymbol{\delta}\mathbf{x}\| \leqslant \|\mathbf{A}^{-1}\| \cdot \|\boldsymbol{\delta}\mathbf{A}\| \cdot \|\mathbf{x} + \boldsymbol{\delta}\mathbf{x}\| \tag{1.16}$$

autrement dit :

$$\frac{\|\boldsymbol{\delta}\mathbf{x}\|}{\|\mathbf{x} + \boldsymbol{\delta}\mathbf{x}\|} \leqslant \operatorname{cond}\mathbf{A}\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|} \tag{1.17}$$

Il n'est donc pas étonnant qu'on se trouve face à une difficulté lorsque le système à résoudre est très mal conditionné, autant avec les méthodes directes qui fournissent une solution erronée qu'avec les méthodes itératives qui ne convergeront pas.

1.2 Techniques de stockage et coût

1.2.1 Paramètres influents

Dans les applications envisagées, en particulier pour les systèmes de grande taille, il est nécessaire d'envisager des techniques de stockage adaptées. Par exemple, si la taille du système est $n = 10^3$, le stockage de $\mathbf{A}$ pleine nécessite 8 Mo de mémoire ; avec $n = 10^6$, 8 000 Go sont nécessaires.

Pire encore, le nombre d'opérations pour résoudre le système $\mathbf{Ax} = \mathbf{b}$ est lié au nombre de termes non nuls (en fait au nombre de termes stockés, donc supposés non nuls) de la matrice, et aussi de la numérotation des inconnues, comme on le verra. Par exemple, une méthode de Cramer sur une matrice pleine demande un nombre d'opérations en virgule flottante de $n_{\text{op}} \approx (n+1)!$, ce qui devient rapidement inextricable, même avec les ordinateurs les plus puissants [15, 48].

Dans le cas des systèmes issus d'une discrétisation par éléments finis, le système est dit creux [1]. Il faut donc en tenir compte au niveau du stockage, en utilisant par exemple un stockage :

- bande (largeur de bande, *bandwidth*, l_b) ;
- ligne de ciel, *skyline* ;
- morse ;
- semi-morse.

Quant à l'influence de la renumérotation des degrés de liberté (ddl), des méthodes heuristiques sont employées car la détermination de la numérotation optimale est un problème extrêmement coûteux (en fait, plus coûteux que le système que l'on cherche à résoudre). Les techniques les plus employées se nomment :

- Cuthill-MacKee inverse [12] ;
- *nested dissection* [24] ;
- degré minimum [25, 64]...

1.2.2 Coût

Le coût d'un algorithme s'estime de différentes manières. Une première indication est la taille de la mémoire requise pour traiter un problème donné. Une autre est le temps mis

1. en anglais, un système creux est dit *sparse*

pour effectuer la résolution du problème. Deux types de prise de temps sont généralement employés : le temps CPU (Central Processing Unit), lié aux opérations effectuées, et le temps horloge, lié au temps extérieur écoulé pendant cette résolution. Il est certain que ce dernier est une mesure très globale puisqu'elle fait intervenir la machine utilisée [2], les compétences du programmeur, et peuvent dépendre fortement du problème traité. Elle ne caractérise donc pas uniquement l'algorithme mais plutôt globalement le logiciel réalisé pour un problème donné sur une machine donnée.

D'autres indicateurs sont également disponibles, comme le nombre d'opérations en virgule flottante nécessaire à la résolution. C'est ce que l'on appelle la *complexité* de l'algorithme. Elle devient plus liée à l'algorithme même en faisant des estimations sur les types de problèmes à traiter, mais elle ne fait pas intervenir les éventuels transferts en mémoire, opérations sur les entiers...

La vraie qualification d'une technique doit donc faire appel à tous ces indicateurs. Ici, seule la complexité sera présentée par la suite. L'estimation du coût effectif en temps CPU des différentes opérations en virgule flottante peut ensuite être proposée par des essais numériques et des prises de temps sur des architectures de machines cibles.

À titre d'indication, un petit test a été réalisé dans ce sens, qui fait intervenir essentiellement les performances du processeur. Les coûts indiqués dans le tableau 1.1 ont été obtenus en janvier 2000. Entre parenthèses, on trouvera les coûts obtenus l'année ou les deux années précédentes. Le code est écrit en Fortran 77 ; les différentes machines testées ainsi que les options de compilation sont les suivantes :

- fronsac est une station de travail HP du LMT-Cachan (HP 9000/879, système B.10.20), la compilation est faite avec f77 +O2 ;
- bacchus est une machine parallèle SGI 02000 du Pôle Parallélisme Île de France Sud (SGI IP27 système 6.5), et la compilation, f90 -O2 ;
- artemis est une machine IBM SP de l'Antenne de Bretagne de l'ENS de Cachan (IBM AIX système 3), et la compilation, f77 -O2.

Les opérations dénommées `daxpy` (série de deux opérations en fait : addition et multiplication), `dscal` (série de multiplications), `dgemv` (calcul d'un produit matrice $\mathbf{A}$ par vecteur $\mathbf{x}$, pondéré par un coefficient β et ajouté à un précédent vecteur $\mathbf{y}$, lui même multiplié par un coefficient α) sont appelées avec la librairie `BLAS`, déjà mentionnée. L'opération dénommée $\alpha\mathbf{y} + \beta\mathbf{A}\mathbf{x}$ est donc l'équivalent de `dgemv`, programmé directement en Fortran. À partir de ces quelques résultats, on peut tirer quelques conclusions : outre le fait qu'utiliser les options d'optimisation des compilateurs est intéressante, l'utilisation de librairies (ici la `BLAS`) est plus efficace que reprogrammer des routines élémentaires soi-même. En fait, l'utilisation de routines standards augmente la lisibilité du programme, permet de programmer à un niveau plus élevé et donc réduit les temps de « debuggage », et permet souvent d'atteindre de meilleures performances, puisque ces librairies sont programmées de façon plus efficace (souvent directement en assembleur) et sont optimisées pour la machine sur laquelle elles sont utilisées.

D'autre part, pour l'utilisateur normal, le coût effectif (en secondes CPU) dépend d'un grand nombre de paramètres comme l'efficacité du compilateur, de l'environnement de développement, des temps de latence de la mémoire, des caches, de l'accès aux disques, etc. En particulier, le changement de version du système d'exploitation d'une année sur l'autre

2. les performances du système, du compilateur, les accès aux disques

type d'opération (moyennée sur 10^8 termes)	coût (10^{-8} s CPU)		
	fronsac	bacchus	artemis
addition	5,2 (5,7 ; 7)	5,54 (6,2)	2,67
multiplication	5,2 (5,7 ; 7)	5,54 (6,2)	2,67
division	9	9 (11,7)	2,73
racine carrée	9	14,5 (18,3)	18,2
`daxpy`	8,9 (9,6 ; 10)	4,66 (5,5)	2,02
`dscal`	4,97 (5,3 ; 6)	2,56 (3,9)	1,35
type d'opération (moyennée sur 100 vecteurs de 1000 termes)	coût (10^{-2} s CPU)		
	fronsac	bacchus	artemis
$\alpha\mathbf{y} + \beta\mathbf{A}\mathbf{x}$	4,95 (5,2)	3,4 (3,9)	0,84
`dgemv`	-	2 (2,3)	1,1

Tableau 1.1 — Estimation du coût des opérations élémentaires

influe sur les résultats du test présenté précédemment.

1.3 Méthodes directes

Encore majoritairement employée dans les codes actuels pour leur robustesse, et d'une prévision fine du coût (nombre d'opérations, stockage nécessaire), elles deviennent coûteuses en temps et en encombrement lorsque le système à résoudre devient très grand. Pour plus d'informations sur ces méthodes, on pourra avantageusement consulter [17].

1.3.1 Méthode d'élimination de Gauss

On simplifie le système pour le rendre triangulaire supérieur par utilisation de combinaison linéaires de lignes (c'est-à-dire d'équations).

Élimination de Gauss par bloc

Écrivons le système à résoudre sous une forme de blocs :

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix} \tag{1.18}$$

Si $\mathbf{A}_{11}$ est inversible, la première équation conduit à exprimer $\mathbf{x}_1$ en fonction de $\mathbf{x}_2$:

$$\mathbf{x}_1 = \mathbf{A}_{11}^{-1}(\mathbf{b}_1 - \mathbf{A}_{12}\mathbf{x}_2) \tag{1.19}$$

En utilisant cette expression dans la deuxième équation, on se ramène à un système avec $\mathbf{x}_2$ comme seule inconnue :

$$\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\mathbf{x}_2 = \mathbf{b}_2 - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{b}_1 \quad \text{soit} \quad \mathbf{S}\mathbf{x}_2 = \mathbf{c} \tag{1.20}$$

$\mathbf{S}$ est appelé complément de Schur ou matrice condensée sur les degrés de liberté $\mathbf{x}_2$. Connaissant $\mathbf{x}_2$, on remonte à $\mathbf{x}_1$ avec (1.19). Le système de départ (1.18) est équivalent à :

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{c} \end{pmatrix} \tag{1.21}$$

et on peut le réécrire :

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A}_{21}\mathbf{A}_{11}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A}_{21}\mathbf{A}_{11}^{-1} & \mathbf{I} \end{bmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{c} \end{pmatrix} \tag{1.22}$$

Utilisation pour la factorisation LU

Si on prend comme bloc 11 la première équation scalaire du système, il faut avoir $\mathbf{A}_{11} \neq 0$ (c'est le premier *pivot*). En continuant sous la forme de l'équation (1.22), récursivement sur les sous-matrices qui restent, on obtient la factorisation $\mathbf{A} = \mathbf{LU}$ (en $n-1$ étapes) avec $\mathbf{U}$ (pour *upper*), matrice triangulaire supérieure et $\mathbf{L}$ (pour *lower*), matrice triangulaire inférieure avec diagonale unitaire.

Pour des raisons d'efficacité de stockage, $\mathbf{L}$ et $\mathbf{U}$ écrasent en mémoire A si on ne stocke pas la diagonale de $\mathbf{L}$ qu'on sait unitaire. À chaque itération k, la matrice de départ $\mathbf{A}$ est donc modifiée en une matrice $\mathbf{A}^{(k)}$.

Conditions d'emploi Si à chaque étape de la factorisation, le pivot $\mathrm{A}_{kk}^{(k-1)}$ est non nul, on peut progresser sans problème. La seule condition est que $\mathbf{A}$ soit *inversible*, mais quand on trouve un pivot nul, il faut procéder à des échanges d'équations. En pratique, on a bien entendu aussi des problèmes si le pivot est petit en valeur absolue ; il est parfois conseillé de procéder systématiquement à un pivotage. Cette opération est cependant coûteuse en terme d'accès et de transfert de données en mémoire, surtout si elle nécessite des accès aux disques. Si la matrice assemblée est initialement bande, les permutations détruisent généralement cette propriété, ce qui peut alors devenir très pénalisant.

Stratégies de choix du pivot Il peut s'agir du pivotage partiel ou du pivotage total :

Pivotage partiel On choisit dans la même colonne k que celle du pivot à changer, la ligne l telle que :

$$|\mathrm{A}_{l,k}| = \max_{m=k,\ldots,n} |\mathrm{A}_{m,k}| \tag{1.23}$$

pour remplacer le pivot ;

Pivotage total On peut aussi permuter les colonnes ; on cherche alors le nouveau pivot dans toute la sous-matrice restant à traiter, on choisit donc la ligne l et la colonne c telles que :

$$|\mathrm{A}_{l,c}| = \max_{m,p=k,\ldots,n} |\mathrm{A}_{m,p}| \tag{1.24}$$

Particularités Sans recours au pivotage, il y a unicité de la factorisation. Dans le cas où il est nécessaire de traiter plusieurs second membres, on opère simultanément sur les $\mathbf{b}$. Si les résolutions doivent être réalisées successivement, et pour chaque autre second membre $\mathbf{b}'$, on résoud :

$$\mathbf{L}\mathbf{U}\mathbf{x}' = \mathbf{b}' \tag{1.25}$$

en considérant, dans un premier temps, $\mathbf{L}\mathbf{y}' = \mathbf{b}'$ étape appelée *descente*, puis puis $\mathbf{U}\mathbf{x}' = \mathbf{y}'$, appelée *montée*.

À chaque fois, il ne s'agit plus que de résoudre un système triangulaire (supérieur ou inférieur). On peut d'autre part calculer à faible coût le déterminant de A une fois que l'on connaît sa décomposition $\mathbf{LU}$:

$$\det \mathbf{A} = \det(\mathbf{LU}) = \det \mathbf{L} \det \mathbf{U} = \det \mathbf{U} = \prod_{k=1,n} \mathrm{U}_{k,k} \tag{1.26}$$

Complexité Dans le cas d'une matrice stockée pleine, la factorisation $\mathbf{LU}$ nécessite environ $n^3/3$ opérations ; une montée (ou 1 descente) nécessite quant à elle en gros $n^2/2$ opérations.

1.3.2 Factorisation de Cholesky

Cette technique a été décrite pour la première fois dans [7] en 1924. Elle considère la factorisation $\mathbf{LU}$ quand $\mathbf{A}$ est *symétrique et inversible*, c'est-à-dire $\mathbf{A} = \mathbf{B}^\top\mathbf{B}$ avec $\mathbf{B}$ à diagonale positive. Dans le cas, le stockage et le calcul ne portent que sur un peu plus de la moitié de la matrice, par exemple sur sa partie inférieure.

Propriété Si $\mathbf{A}$ est symétrique, définie, positive (donc inversible), il n'est pas utile de pivoter.

Complexité Pour $\mathbf{A}$ pleine, la factorisation $\mathbf{B}^\top\mathbf{B}$ requière environ $n^3/6$ opérations. Pour $\mathbf{A}$ bande, la factorisation $\mathbf{B}^\top\mathbf{B}$ requière approximativement $nl_b^2/2$ opérations, une montée (ou une descente) coûte de l'ordre de nl_b opérations. Le stockage nécessite aux alentours de nl_b réels (généralement 8 octets). De façon courante, on a $l_b \ll n$ [3].

Un exemple d'utilisation de cette factorisation pour l'orthogonalisation des vecteurs d'un sous-espace est présentée dans l'annexe A.1.

1.3.3 Factorisation de Crout

C'est la méthode la plus employée. La factorisation se met sous la forme $\mathbf{A} = \mathbf{LDL}^\top$ avec $\mathbf{L}$ triangulaire inférieure à diagonale unitaire et $\mathbf{D}$ diagonale. Cette factorisation s'applique pour $\mathbf{A}$ *symétrique, définie positive*, mais traite aussi les cas où les valeurs propres sont négatives ou nulles.

3. $\frac{l_b}{n} = \frac{1}{10}$ à $\frac{1}{100}$

boucle sur $j = 1, 2, \ldots, n$
  boucle sur $i = 1, 2, \ldots, j-1$
    $v_i \leftarrow \mathrm{A}_{j,i}\mathrm{A}_{i,i}$
  fin
  $v_j \leftarrow \mathrm{A}_{j,j} - \mathbf{A}_{j,1:j-1}\mathbf{v}_{1:j-1}$
  $\mathrm{A}_{j,j} \leftarrow v_j$
  $\mathbf{A}_{j+1:n,j} \leftarrow \frac{1}{v_j}(\mathbf{A}_{j+1:n,j} - \mathbf{A}_{j+1:n,1:j-1}\mathbf{v}_{1:j-1})$
fin

Algorithme 1.1 — Factorisation LDL stockée pleine

Complexité Elle est identique à celle de la factorisation de Cholesky, mais on n'a plus besoin d'extraire n racines carrées (pour la diagonale).

L'algorithme 1.1 présente cette factorisation dans le cas d'une matrice symétrique $\mathbf{A}$ stockée pleine. Il requière l'utilisation d'un vecteur $\mathbf{v}$ supplémentaire. Sans pivotage, comme pour Cholesky et pour Gauss, la factorisation conserve le profil de la matrice écrasée en mémoire, comme on le verra dans la suite.

1.3.4 Lien avec la condensation de Schur

Super-éléments

Comme c'était le cas pour la factorisation **LU**, la factorisation de Cholesky ou Crout est très liée à la condensation de Schur. Dans le cas des éléments finis, on peut bâtir ainsi la

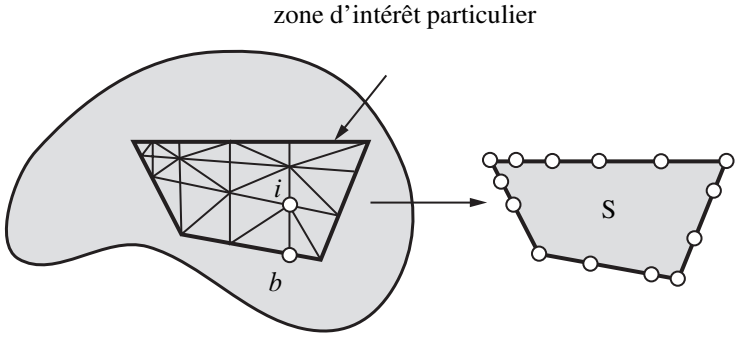


Figure 1.2 — Utilisation d'un super-élément

notion de super-élément. Considérons une partie d'une structure qui représente une zone d'intérêt particulier (par exemple, une zone qui reste en élasticité alors que le reste de la structure plastifie, où une zone dans laquelle des fonctions de bases particulières ont été ajoutées comme dans la méthode de partition de l'unité...) et qui a été maillée par éléments finis, figure 1.2. L'interface avec le reste de la structure se fait par le bord de cette zone. Notons avec un indice i les degrés de libertés des nœuds strictement internes à cette zone et $\mathbf{b}$ ceux des nœuds de l'interface, alors :

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{ib} \\ \mathbf{K}_{bi} & \mathbf{K}_{bb} \end{bmatrix} \tag{1.27}$$

Si on condense sur les nœuds *bord* on obtient :

$$\mathbf{S} = \mathbf{K}_{bb} - \mathbf{K}_{bi}\mathbf{K}_{ii}^{-1}\mathbf{K}_{ib} \tag{1.28}$$

c'est la matrice de rigidité du super-élément considéré. Elle est similaire à une matrice de rigidité élémentaire, pour un *élément* qui peut être lui-même une petite structure.

Conservation de la bande

La figure 1.3 montre, à l'itération k, c'est-à-dire lors de la condensation de l'inconnue k sur les suivantes, les termes de la matrice qui vont être touchés, autrement dit, ceux qui *a priori* sont ou deviennent non-nuls. Clairement, la bande est conservée lors de la factorisation. On peut de la même façon montrer que seuls les termes sous le profil sont modifiés, et donc que la factorisation conserve le profil de la matrice.

1.3.5 Autres techniques

L'adaptation de la factorisation de Crout au cas où A est non-symétrique est classique, c'est la factorisation $\mathbf{LDM}^\top$. D'autres variantes des techniques précédentes permettent de tirer mieux parti des conditions de stockage, en particulier pour optimiser les échanges de données avec la partie des informations stockées sur disque, comme la méthode frontale [38]. D'autres versions encore permettent de réaliser des factorisations en parallèle de façon assez performante, comme les méthodes multi-frontales [16].

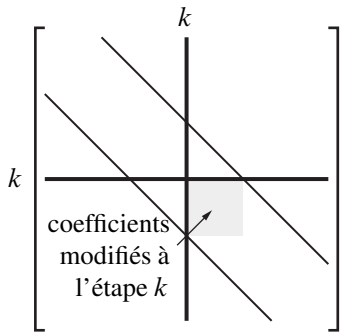


Figure 1.3 — Étape k de la factorisation

1.4 Quelques particularités en éléments finis

1.4.1 Conditions aux limites en déplacement

La prise en compte de déplacements imposés non-nuls a une influence sur l'utilisation des solveurs dans le sens où la matrice de rigidité est modifiée par ces contraintes.

Séparation des degrés de liberté

Le problème correspondant de minimisation du potentiel sous contrainte est le suivant :

$$\min_{\mathbf{Cu}=\mathbf{u}_d} \mathrm{J}_1(\mathbf{u}) = \frac{1}{2}\mathbf{u}^\top \mathbf{K}\mathbf{u} - \mathbf{f}^\top \mathbf{u} \tag{1.29}$$

$\mathbf{C}$ est une matrice booléenne dont le but est d'extraire une partie des degrés de libertés, ceux sur lesquels la condition d'égalité aux déplacements imposés $\mathbf{u}_d$ est imposée.

Reprenons l'écriture par blocs déjà présentés dans l'équation (1.18) en séparant les degrés de liberté imposés $\mathbf{u}_2 = \mathbf{C}\mathbf{u} = \mathbf{u}_d$. Le problème (1.29) est ainsi équivalent à :

$$\min_{\mathbf{u}_1} \frac{1}{2}\mathbf{u}_1^\top \mathbf{K}_{11}\mathbf{u}_1 - \mathbf{f}_1^\top \mathbf{u}_1 + \mathbf{u}_1^\top \mathbf{K}_{12}\mathbf{u}_d + \text{constante} \tag{1.30}$$

dont l'équation d'Euler correspondante est $\mathbf{K}_{11}\mathbf{u}_1 = \mathbf{f}_1 - \mathbf{K}_{12}\mathbf{u}_d$. On a ainsi un problème de taille réduite où les charges extérieures sont modifiées par le terme $-\mathbf{K}_{12}\mathbf{u}_d$ [4]. Cette méthode est effectivement employée dans des codes comme abaqus$^{\text{TM}}$. L'inconvénient de cette méthode réside dans le tri explicite nécessaire dans les degrés de liberté qui rempli donc le terme $\mathbf{K}_{12}$, et qui entraîne un surcoût important lorsque le nombre de degrés de liberté bloqués est grand.

Multiplicateurs de Lagrange

Une autre technique consiste à reformuler le problème (1.29) en relaxant la contrainte à l'aide de multiplicateur de Lagrange $\mathbf{l}$ et donc de transformer le problème de minimum en celui de la recherche du point selle du Lagrangien :

$$\operatorname*{extr}_{\mathbf{u},l} \mathrm{J}_1(\mathbf{u}) - \mathbf{l}^\top(\mathbf{Cu} - \mathbf{u}_d) \tag{1.31}$$

Il n'est plus nécessaire de séparer les degrés de libertés, mais la taille du problème a augmenté ; en effet les équations d'Euler sont cette fois :

$$\underbrace{\begin{bmatrix} \mathbf{K} & -\mathbf{C}^\top \\ -\mathbf{C} & \mathbf{0} \end{bmatrix}}_{\mathbf{A}} \begin{pmatrix} \mathbf{u} \\ \mathbf{l} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ -\mathbf{u}_d \end{pmatrix} \tag{1.32}$$

On peut remarquer que la première équation montre que les multiplicateurs de Lagrange sont bien les réactions à l'appui.

L'inconvénient vient du fait que la taille du problème augmente, et du fait que $\mathbf{A}$ est une matrice symétrique mais plus définie positive ; elle possède des valeurs propres négatives. On a cependant le résultat suivant si $\mathbf{K}$ est symétrique, définie positive :

$$\mathbf{A} \text{ inversible} \Leftrightarrow \mathbf{C} \text{ injective} \Leftrightarrow \ker \mathbf{C} = \tag{1.33}$$

c'est-à-dire que l'on n'écrit que des liaisons indépendantes. Une factorisation de Gauss risque de détruire la structure bande de la matrice $\mathbf{K}$ et la factorisation de Crout va demander des pivotages.

4. ce sont les réactions à l'appui qui permettent d'avoir un déplacement bien égal à $\mathbf{u}_d$

Une modification de cette technique permet, avec une numérotation ad hoc qui conserve la bande, de ne pas avoir besoin de pivoter. Il s'agit d'une technique de double multiplicateurs utilisée dans le code Cast3M^{TM}. Elle peut être illustrée de la manière suivante :

$$\underset{\mathbf{u},\mathbf{l}_1,\mathbf{l}_2}{\text{extr}}\ \mathrm{J}_1(\mathbf{u}) + \underbrace{\frac{1}{2}\begin{pmatrix}\mathbf{l}_1\\ \mathbf{l}_2\\ \mathbf{u}\end{pmatrix}^{\top}\begin{bmatrix}-1 & 1 & 1\\ 1 & -1 & 1\\ 1 & 1 & 0\end{bmatrix}\begin{pmatrix}\mathbf{l}_1\\ \mathbf{l}_2\\ \mathbf{u}_2\end{pmatrix} - \begin{pmatrix}\mathbf{u}_d\\ \mathbf{u}_d\end{pmatrix}^{\top}\begin{pmatrix}\mathbf{l}_1\\ \mathbf{l}_2\end{pmatrix}}_{-\frac{1}{2}(l_1-l_2)^2+(l_1+l_2)^{\top}(\mathrm{C}u-u_d)} \tag{1.34}$$

dont les équations d'Euler sont :

$$\begin{aligned}\mathbf{Ku} &= \mathbf{f} - \mathbf{C}^{\top}(l_1+l_2)\\ l_1 &= l_2\\ \mathbf{Cu} &= \mathbf{u}_d\end{aligned} \tag{1.35}$$

Bien entendu, en cas de nombreux blocages, la taille du problème croît en conséquence.

Valeurs très grandes

On peut aussi considérer une condition aux limites en déplacement imposé comme une modélisation de l'action de l'extérieur. Celui-ci pourrait aussi réagir comme un ressort ayant une grande raideur k, avec un déplacement imposé à la base. Le potentiel élastique de l'ensemble structure + ressort extérieur est alors :

$$\mathrm{J}_1(\mathbf{u}) + \frac{1}{2}(\mathbf{Cu}-\mathbf{u}_d)^{\top}k(\mathbf{Cu}-\mathbf{u}_d) \tag{1.36}$$

Les nouvelles équations d'Euler de cette formulation sont alors :

$$(\mathbf{K}+\mathbf{C}^{\top}k\mathbf{C})\mathbf{u} = \mathbf{f} + \mathbf{C}^{\top}k\mathbf{C}\mathbf{u}_d \tag{1.37}$$

On ne fait donc que rajouter des termes dans la matrice de raideur sur les degrés de libertés incriminés et dans le vecteur des forces généralisées sur les efforts duaux.

Le problème est le choix de k : pour des valeurs trop petites, la condition est mal imposée, pour des valeurs trop grandes, le système devient extrêmement mal conditionné, voire numériquement singulier.

1.4.2 Relations linéaires entre degrés de liberté

Cette situation se présente par exemple fans le cas d'utilisation de symétrie par rapport à un plan non parallèle à un des plan des axes globaux de l'analyse, dans le cas de conditions aux limite périodiques sur une cellule de base... À chaque fois, il s'agit de contraintes de la forme $\mathbf{Cu} = \mathbf{u}_d$ où $\mathbf{C}$ n'est cette fois-ci pas forcément booléenne. Les techniques précédentes s'appliquent bien entendu sur ces cas de figure tout aussi bien... et avec les mêmes limitations.

1.4.3 Système singulier : semi défini positif

Les exemples sont nombreux : il s'agit de structure avec des modes rigides, de problèmes de sous-intégration... il y a alors des déplacements non nuls à énergie de déformation nulle, autrement dit des modes rigides :

$$\frac{1}{2}\mathbf{u}^\top \mathbf{K}\mathbf{u} = 0 \tag{1.38}$$

Dans la suite, on s'intéressera au cas où cas est A symétrique, sachant que la généralisation est tout à fait possible.

Noyau

L'ensemble des solutions indépendantes, stockées dans les colonne de $\mathbf{R}$, telles que $\mathbf{AR} = \mathbf{0}$, forme le noyau de $\mathbf{A}$. S'il y en a p, le rang de $\mathbf{A}$ est $n - p$.

Existence de solutions

Pour qu'il existe des solutions, une condition nécessaire et suffisante est que $\mathbf{b}$ soit orthogonal au noyau de $\mathbf{A}$, c'est-à-dire $\mathbf{R}^\top \mathbf{b} = 0$. Exemple : les efforts ne sollicitent pas les modes de solide rigide.

- Si $\mathbf{A}$ est non symétrique, la condition est que $\mathbf{b}$ appartienne au noyau adjoint de $\mathbf{A}$.
- Si $\mathbf{x_0}$ est une solution de ce système, l'ensemble des solutions est de la forme $\mathbf{x_0} + \mathbf{R}\alpha$ où α est quelconque.

Inverses généralisées (ou pseudo-inverses)

$\mathbf{A}^+$ telle que $\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$. Si $\mathbf{A}$ est inversible, il n'y a qu'une pseudo-inverse $\mathbf{A}^+ = \mathbf{A}^{-1}$. Pour $\mathbf{A}$ symétrique, toutes les pseudo-inverses sont de la forme $\mathbf{A}^+ + \mathbf{B}^\top \mathbf{R}^\top + \mathbf{R}\mathbf{B}$, avec $\mathbf{B}$ quelconque [19].

Factorisation et construction d'une inverse généralisée

Supposons que le système puisse s'écrire sous la forme de blocs à l'image de (1.18), où $\mathbf{A}_{11}$ est la plus grande sous-matrice régulière. On a alors $\mathbf{S} = \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12} = \mathbf{0}$. Une pseudo-inverse de $\mathbf{A}$ est :

$$\begin{bmatrix} \mathbf{A}_{11}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{1.39}$$

et la solution est :

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11}^{-1}\mathbf{b}_1 \\ \mathbf{0} \end{pmatrix} + \begin{bmatrix} -\mathbf{A}_{12} \\ \mathbf{I}_d \end{bmatrix} \mathbf{x}_2 \quad \text{soit} \quad \mathbf{x} = \mathbf{x}_0 + \mathbf{R}\boldsymbol{\alpha} \tag{1.40}$$

La factorisation fait progresser l'élimination tant qu'un pivot non nul existe. Dans notre cas, avec pivotage, on tombe sur une sous-matrice nulle. Pour les matrices semi-définies, il n'est pas nécessaire de pivoter car un pivot nul apparaît lorsque la ligne du pivot est

une combinaison linéaire des précédentes. L'inconnue correspondante est repérée et fixée à **0** [voir $\mathbf{x}_0$ dans (1.40)], c'est-à-dire qu'on ajoute des liaisons complémentaires pour fixer les modes de solide rigide — à condition que le terme correspondant au second membre condensé soit nul aussi. La colonne correspondante de la matrice stocke le vecteur associé [voir **R** dans (1.40)].

1.5 Méthodes itératives stationnaires

De manière générale avec les techniques itératives, il faut se poser des questions telles que :
- quelles sont les conditions d'emploi de ces techniques afin d'assurer la convergence vers la solution cherchée ?
- quel critère d'arrêt des itérations employer ?
- quels sont les coûts de ces méthodes ?

Dans le cas des méthodes itératives stationnaires, on rencontre les particularités suivantes :
- on accède à **A** uniquement par le produit **Aw** (phase la plus coûteuse). Éventuellement, comme en éléments finis, on peut le faire sous forme élémentaire :

$$\mathbf{Aw} = \sum_e \mathbf{A}^e \mathbf{w}^e \tag{1.41}$$

- on construit une suite de solutions approchées :

$$\mathbf{x}^{(k)} \underset{k\to\infty}{\to} \mathbf{A}^{-1}\mathbf{b} \tag{1.42}$$

à partir d'une récurrence linéaire $\mathbf{x}^{(k)} = \mathbf{B}\mathbf{x}^{(k-1)} + \mathbf{c}$ où **B** et **c** sont indépendantes de k [5].

Ce sont des méthodes de type *point fixe* : si on converge, c'est vers :

$$\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{c} \;\Rightarrow\; (\mathbf{x}^{(k)} - \mathbf{x}) = \mathbf{B}(\mathbf{x}^{(k-1)} - \mathbf{x}) \tag{1.43}$$

Pour avoir $\|\mathbf{x}^{(k)} - \mathbf{x}\| \to_{k\to\infty} 0$, il faut et il suffit que $\rho(\mathrm{B}) < 1$ où $\rho(\mathrm{B})$ est le *rayon spectral* de **B** : c'est le sup des valeurs absolues des valeurs propres de **B**. Le taux de convergence asymptotique est défini par :

$$\tau = -\ln \rho(\mathrm{B}) \tag{1.44}$$

Il peut servir à estimer le nombre d'itérations nécessaire pour atteindre le niveau de convergence souhaité ; en effet on a :

$$(\mathbf{x}^{(k)} - \mathbf{x}) = \mathbf{B}^k(\mathbf{x}^{(0)} - \mathbf{x}) \;\Rightarrow\; \frac{\|\mathbf{x}^{(k)} - \mathbf{x}\|}{\|\mathbf{x}^{(0)} - \mathbf{x}\|} \leqslant \|\mathbf{B}^k\| \;\Rightarrow\; -\ln \frac{\|\mathbf{x}^{(k)} - \mathbf{x}\|}{\|\mathbf{x}^{(0)} - \mathbf{x}\|} \leqslant k\tau \tag{1.45}$$

L'intérêt est de pouvoir estimer le nombre d'itérations nécessaires pour diviser le niveau d'erreur par un coefficient donné (utilisation d'un critère d'arrêt), connaissant le taux de convergence asymptotique de la méthode. Pour plus d'informations sur les méthodes itératives stationnaires, le lecteur pourra consulter [29].

5. Cette caractéristique n'existe pas pour les méthodes non stationnaires.

1.5.1 Méthode de Jacobi

Pour les systèmes linéaires, cette méthode, d'algorithme 1.2, est décrite pour mémoire seulement, car elle possède un faible taux de convergence et dans nos applications, est remplacée par d'autre techniques plus efficaces. Elle illustre cependant bien les techniques de point

Vecteur de départ $\mathbf{x}^{(0)}$
tant que R $> \varepsilon$
  itération sur k
  boucle sur $i = 1, n$
    $\bar{\mathbf{x}}_i \leftarrow \mathbf{A}_{i,1:i-1}\mathbf{x}^{(k-1)}_{1:i-1} + \mathbf{A}_{i,i+1:n}\mathbf{x}^{(k-1)}_{i+1:n}$
    $\bar{\mathbf{x}}_i \leftarrow (\mathbf{b}_i - \bar{\mathbf{x}}_i)/\mathrm{A}_{i,i}$
  fin
  $\mathbf{x}^{(k)} \leftarrow \bar{\mathbf{x}}$
fin

Algorithme 1.2 — Méthode de Jacobi

fixe : si on suppose connues toutes les inconnues sauf l'inconnue i, cette dernière peut être déterminée à l'aide de l'équation i. Procédant ainsi sur toutes les inconnues, on obtient un nouvel itéré $\mathbf{x}^k$ à partir de l'ancien $\mathbf{x}^{k-1}$. Cependant, comme la méthode de Gauss-Seidel, elle peut présenter un intérêt dans le cas de systèmes non-linéaires. Sous forme matricielle, cela se traduit par :

$$\mathbf{x}^k = (\mathbf{I} - \mathbf{D}^{-1}\mathbf{A})\mathbf{x}^{k-1} + \mathbf{D}^{-1}\mathbf{b} \tag{1.46}$$

où $\mathbf{D}$ est la matrice diagonale, contenant la diagonale de $\mathbf{A}$. La matrice d'itération est donc ici $\mathbf{B} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$. On peut noter l'utilisation d'un vecteur de stockage $\bar{\mathbf{x}}$ supplémentaire.

Convergence Si $\mathbf{A}$ est symétrique, définie positive, alors $\rho_{\mathrm{J}} = \rho(\mathrm{B}) < 1$, mais en éléments finis, on a une évaluation de ce rayon spectral qui varie comme :

$$\rho_{\mathrm{J}} = 1 - \frac{h^2}{2} + \mathcal{O}(h^4) \tag{1.47}$$

d'où un taux de convergence faible lorsque le maillage est raffiné (donc lorsque h est faible) :

$$\tau \approx \frac{h^2}{2} \tag{1.48}$$

1.5.2 Méthode de Gauss-Seidel

L'idée est similaire à celle de la méthode de Jacobi, à ceci près que l'on utilise les nouvelles valeurs estimées dès qu'elles sont disponibles, et non pas à la fin de l'itération. Sous forme matricielle, cela se traduit par :

$$\mathbf{x}^k = (\mathbf{D} - \mathbf{E})^{-1}(\mathbf{F}\,\mathbf{x}^{k-1} + \mathbf{b}) \tag{1.49}$$

où $-\mathbf{E}$ est la partie triangulaire inférieure de $\mathbf{A}$, et $-\mathbf{F}$ sa partie triangulaire supérieure. La matrice d'itération est ici :

$$\mathbf{B} = (\mathbf{D} - \mathbf{E})^{-1}\mathbf{F} = (\mathbf{I} - \mathbf{D}^{-1}\mathbf{E})^{-1}\mathbf{D}^{-1}\mathbf{F} \tag{1.50}$$

Quand $\mathbf{A}$ est symétrique, définie positive, on est assuré de la convergence puisque $\rho_{\text{GS}} < 1$. Mais l'estimation de la valeur du rayon spectral est difficile cas il dépend en particulier de la numérotation des degrés de liberté employée.

Vecteur de départ $\mathbf{x}^{(0)}$
tant que $\mathrm{R} > \varepsilon$
  itération sur k
  boucle sur $i = 1, 2, \ldots, n$
    $\mathbf{x}_i^{(k-1)} \leftarrow \mathbf{0}$
    $\boldsymbol{\beta} \leftarrow \mathbf{A}_{i,1:n}\mathbf{x}_{1:n}^{(k-1)}$
    $\mathbf{x}_i^{(k)} \leftarrow (\mathbf{b}_i - \boldsymbol{\beta})/\mathrm{A}_{i,i}$
  fin
fin

Algorithme 1.3 — Méthode de Gauss-Seidel

1.5.3 Méthode de relaxation

Cette technique, SOR pour *successive over relaxation*, est attribuée à Southwell [63] et date de 1946. Elle utilise comme base la méthode de Gauss-Seidel ainsi qu'une technique maintenant souvent utilisée avec d'autre approches : la technique de *relaxation*.

Le principe, expliqué par l'algorithme 1.4, consiste à modifier (par relaxation) l'itéré $\mathbf{x}^k$ fourni par l'algorithme de Gauss-Seidel à chaque itération. Pour cela, il faut conserver l'itéré précédent $\mathbf{x}^{k-1}$ et modifier l'itéré courant comme suit :

$$\mathbf{x}^k \leftarrow \omega\mathbf{x}^k + (1-\omega)\mathbf{x}^{k-1} \tag{1.51}$$

où ω est appelé le paramètre de relaxation. Si $\omega < 1$, on parle de sous-relaxation, et si $\omega > 1$, de sur-relaxation.

On peut montrer que la matrice d'itération correspondante est :

$$\mathbf{B} = (\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{E})^{-1}[(1-\omega)\mathbf{I} + \omega\mathbf{D}^{-1}\mathbf{F}] \tag{1.52}$$

Le théorème de Kahan donne une condition nécessaire de convergence : $0 < \omega < 2$. Si $\omega = 0$, l'algorithme ne progresse plus. Le théorème d'Ostrowski-Reich permet de conclure à la convergence $\forall\omega \in]0\,;2[$ quand $\mathbf{A}$ est symétrique, définie positive.

Le choix de ω est fait à partir d'heuristiques, comme $\omega = 2 - \mathcal{O}(h)$ pour une discrétisation éléments finis dont la taille de maille est h. Il existe aussi des techniques d'optimisation de ω en cours de route, comme la technique d'accélération d'Aitken, voir annexe A.2.

On peut aussi s'inspirer de résultats dans des cas particuliers. Par exemple dans un cas tridiagonal bloc, on montre que la valeur optimale de ω est :

$$\omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - \rho_{\text{J}}^2}} \tag{1.53}$$

Vecteur de départ $\mathbf{x}^{(0)}$
tant que $\mathrm{R} > \varepsilon$
  itération sur k
  boucle sur $i = 1, 2, \ldots, n$
    $\boldsymbol{\alpha} \leftarrow \mathbf{x}_i^{(k-1)}$
    $\mathbf{x}_i^{(k-1)} \leftarrow \mathbf{0}$
    $\boldsymbol{\beta} \leftarrow \mathbf{A}_{i,1:n}\mathbf{x}_{1:n}^{(k-1)}$
    $\mathbf{x}_i^{(k)} \leftarrow (1-\omega)\boldsymbol{\alpha} + \omega\boldsymbol{\beta}$
  fin
fin

Algorithme 1.4 — Méthode SOR

La valeur de ρ_{J} est ainsi à estimer sur le problème en question. On a alors $\rho_{\mathrm{SOR}} = \omega_{\mathrm{opt}} - 1$, et, dans le cas éléments finis, $\tau \approx 2h$ (un ordre est gagné par rapport à la méthode de Jacobi).

La technique de relaxation peut être appliquée à d'autres algorithmes. Par exemple, un algorithme de Jacobi relaxé s'écrit sous la forme de l'algorithme 1.5.

Vecteur de départ $\mathbf{x}^{(0)}$
tant que $\mathrm{R} > \varepsilon$
  itération sur k
  boucle sur $i = 1, 2, \ldots, n$
    $\bar{\mathbf{x}}_i \leftarrow \mathbf{A}_{i,1:i-1}\mathbf{x}_{1:i-1}^{(k-1)} + \mathbf{A}_{i,i+1:n}\mathbf{x}_{i+1:n}^{(k-1)}$
    $\bar{\mathbf{x}}_i \leftarrow (\mathbf{b}_i - \bar{\mathbf{x}}_i)/\mathrm{A}_{i,i}$
  fin
  $\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k-1)} + \omega(\bar{\mathbf{x}} - \mathbf{x}^{(k-1)})$
fin

Algorithme 1.5 — Méthode de Jacobi relaxée

1.5.4 Méthode d'Uzawa

Dans certains cas de figure comme les problèmes de point selle ou les liaisons traitées par multiplicateur de Lagrange par exemple, on peut être amené à résoudre des problèmes de la forme :

$$\begin{bmatrix} \mathbf{K} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{l} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \tag{1.54}$$

Pour que le problème continue d'être inversible, il suffit que $\mathbf{C}$ soit injective.

La méthode d'Uzawa développée en 1958 consiste à résoudre itérativement de façon découplée, comme le montre l'algorithme 1.6 où α est un paramètre de la méthode. Pour analyser cet algorithme, on peut condenser la variable $\mathbf{x}^{(k)}$ sur la variable $\mathbf{l}^{(k)}$ pour trouver :

$$\mathbf{l}^{(k)} = (\mathbf{I} - \alpha\mathbf{C}^\top\mathbf{K}^{-1}\mathbf{C})\mathbf{l}^{(k-1)} + \alpha(\mathbf{C}^\top\mathbf{K}^{-1}\mathbf{f} - \mathbf{g}) \tag{1.55}$$

Vecteurs de départ $\mathbf{x}^{(0)}$, $\mathbf{l}^{(0)}$
tant que R $> \varepsilon$, itérer sur k
résoudre $\mathbf{K}\mathbf{x}^{(k)} = \mathbf{b} - \mathbf{C}\mathbf{l}^{(k-1)}$
réactualiser $\mathbf{l}^{(k)} \leftarrow \mathbf{l}^{(k-1)} + \alpha(\mathbf{C}^\top \mathbf{x}^{(k)} - \mathbf{c})$
fin

Algorithme 1.6 — Méthode d'Uzawa

avec $\mathbf{B} = \mathbf{I} - \alpha \mathbf{C}^\top \mathbf{K}^{-1}\mathbf{C}$. Si on appelle $\lambda_1, \ldots, \lambda_p$, les valeurs propres de la matrice $\mathbf{C}^\top \mathbf{K}^{-1}\mathbf{C}$, on en déduit l'équivalence suivante :

$$\varrho(\mathbf{B}) < 1 \;\Leftrightarrow\; \forall i, \quad |1 - \alpha\lambda_i| < 1 \tag{1.56}$$

Avec $\mathbf{C}^\top \mathbf{K}^{-1}\mathbf{C}$ symétrique, définie positive, $\lambda_i > 0$ donc il faut prendre :

$$0 < \alpha < \frac{2}{\max_i \lambda_i} \tag{1.57}$$

1.6 Méthodes itératives non-stationnaires

Ces méthodes se décrivent de la même façon que les précédentes, à ceci près que les opérateurs $\mathbf{B}$ et $\mathbf{c}$ dépendent des itérations.

1.6.1 Méthode générique de projection

Reformulons le problème de départ en définissant le résidu :

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x} \tag{1.58}$$

Il s'agit maintenant d'avoir $\mathbf{r} = \mathbf{0}$. L'idée est ici de se trouver avec un problème de plus

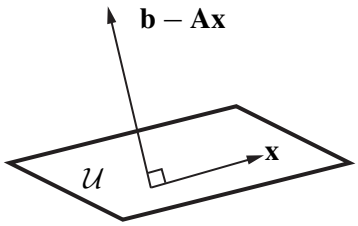


Figure 1.4 — Réduction dans un sous-espace

petite taille, de structure plus simple (par exemple, tridiagonal) par plusieurs étapes de *projection*.

Considérons un sous-espace $\mathcal{U}$ de l'espace dans lequel la solution est cherchée. Le problème approché est défini par $\mathbf{x} = \mathbf{Q}\hat{\mathbf{x}} \in \mathcal{U}$ avec $\mathbf{b} - \mathbf{A}\mathbf{x} \perp \mathcal{U}$, comme décrit sur la figure 1.4. Il est clair que si $\mathcal{U}$ est l'espace complet, ce problème est le problème de départ. Dans tous les cas, il conduit à :

$$\mathbf{Q}^\top(\mathbf{b} - \mathbf{A}\mathbf{Q}\hat{\mathbf{x}}) = \mathbf{0} \;\Rightarrow\; (\mathbf{Q}^\top\mathbf{A}\mathbf{Q})\hat{\mathbf{x}} = \mathbf{Q}^\top\mathbf{b}, \quad \text{soit} \quad \hat{\mathbf{A}}\hat{\mathbf{x}} = \hat{\mathbf{b}} \tag{1.59}$$

Une méthode de projection peut donc se schématiser de la façon suivante :

1. $\mathbf{r} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}$
2. $\hat{\mathbf{x}} = \hat{\mathbf{A}}^{-1}\mathbf{Q}^{\top}\mathbf{r}$
3. $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{Q}\hat{\mathbf{x}}$
4. itération suivante

Le choix de $\mathbf{Q}$ à chaque itération dépend bien entendu de la méthode employée.

1.6.2 Méthode du gradient

Pour $\mathbf{A}$ symétrique, définie positive, la solution approchée est modifiée de la manière suivante :

$$\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)} \tag{1.60}$$

où $\mathbf{d}^{(k)}$ est la direction de recherche dans la direction du gradient :

$$\mathbf{d}^{(k)} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b} \tag{1.61}$$

$\mu^{(k)}$ est le paramètre d'optimalité de la nouvelle solution dans la direction $\mathbf{d}^{(k)}$. Il se calcule comme :

$$\mu^{(k)} = \frac{\mathbf{d}^{(k)\top}(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)})}{\mathbf{d}^{(k)\top}\mathbf{A}\mathbf{d}^{(k)}} = -\frac{\mathbf{d}^{(k)\top}\mathbf{d}^{(k)}}{\mathbf{d}^{(k)\top}\mathbf{A}\mathbf{d}^{(k)}} \tag{1.62}$$

Il s'agit d'une méthode de projection pour laquelle la taille du sous-espace est 1 car on choisit pour $\mathbf{Q}$ le sous-espace généré par le seul vecteur résidu $\mathbf{Q} = \mathbf{r}$.

La convergence de cette méthode est assez lente, d'où l'intervention d'une technique de conjugaison pour la méthode suivante.

1.6.3 Méthode du gradient conjugué

Cette méthode est généralement attribuée à Hestenes et Stiefel [32] en 1952. Elle est utilisée dans le cas où $\mathbf{A}$ est symétrique, définie positive.

L'initialisation consiste à choisir un vecteur de départ $\mathbf{x}^{(0)}$, à calculer le résidu associé $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$ et de prendre comme première direction de recherche $\mathbf{d}^{(0)} = \mathbf{r}^{(0)}$. Ensuite, les itérations sont de la forme :

1. itéré $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)}$ avec le pas optimal :

$$\mu^{(k)} = \frac{\mathbf{r}^{(k)\top}\mathbf{d}^{(k)}}{\mathbf{d}^{(k)\top}\mathbf{A}\mathbf{d}^{(k)}} = \frac{\mathbf{r}^{(k)\top}\mathbf{r}^{(k)}}{\mathbf{d}^{(k)\top}\mathbf{A}\mathbf{d}^{(k)}} \tag{1.63}$$

2. résidu $\mathbf{r}^{(k+1)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k+1)}$;
3. direction de recherche $\mathbf{d}^{(k+1)} = \mathbf{r}^{(k+1)} + \lambda^{(k)}\mathbf{d}^{(k)}$ qu'on orthogonalise par rapport à la précédente :

$$\mathbf{d}^{(k+1)\top}\mathbf{d}^{(k)} = 0 \;\Rightarrow\; \lambda^{(k)} = -\frac{\mathbf{r}^{(k+1)\top}\mathbf{A}\mathbf{d}^{(k)}}{\mathbf{d}^{(k)\top}\mathbf{A}\mathbf{d}^{(k)}} = \frac{\mathbf{r}^{(k+1)\top}\mathbf{r}^{(k+1)}}{\mathbf{r}^{(k)\top}\mathbf{r}^{(k)}} \tag{1.64}$$

4. itération suivante.

Propriétés Si $i < j$, alors $\mathbf{r}^{(i)\top}\mathbf{r}^{(j)} = 0 = \mathbf{d}^{(i)\top}\mathbf{r}^{(j)} = \mathbf{d}^{(i)\top}\mathbf{A}\mathbf{d}^{(j)} = \mathbf{r}^{(i)\top}\mathbf{A}\mathbf{d}^{(j)}$. On construit donc des vecteurs orthogonaux au sens de $\mathbf{A}$, $\mathbf{d}^{(0)}, \mathbf{d}^{(1)}, \ldots$ qui forment une base. On minimise à chaque fois sur un sous-espace de taille de plus en plus grande.

En théorie, cette méthode est donc une méthode directe en n étapes. En pratique, on a progressivement perte d'orthogonalité due aux arrondis : c'est donc en ce sens une méthode itérative et on a la propriété de convergence suivante :

$$\|\mathbf{x}^{(k)} - \mathbf{x}\|_{\mathbf{A}} \leqslant 2\Big[\frac{\sqrt{\operatorname{cond}\mathbf{A}} - 1}{\sqrt{\operatorname{cond}\mathbf{A}} + 1}\Big]^{(k)} \|\mathbf{x}^{(0)} - \mathbf{x}\|_{\mathbf{A}} \tag{1.65}$$

d'où, en éléments finis :

$$\tau = \mathcal{O}\Big(\frac{1}{\sqrt{\operatorname{cond}\mathbf{A}}}\Big) = \mathcal{O}(h) \tag{1.66}$$

1.6.4 Préconditionnement

Si on peut avoir facilement une approximation spectrale $\mathbf{M}$ de $\mathbf{A}$, l'idée est de chercher à résoudre :

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b} \tag{1.67}$$

où $\mathbf{M}$ est symétrique, définie positive. Ce problème devient non symétrique, mais on peut lui substituer :

$$(\mathbf{M}^{-1/2}\mathbf{A}\mathbf{M}^{-1/2})\mathbf{M}^{1/2}\mathbf{x} = \mathbf{M}^{-1/2}\mathbf{b} \tag{1.68}$$

Le choix du préconditionneur dépend du problème à traiter. Dans le meilleur des cas, il est facile à *inverser* et représente bien la matrice $\mathbf{A}$. En fait, si $\mathbf{M} = \mathbf{I}$ il est facile à inverser, mais est très différent de $\mathbf{A}$, et si $\mathbf{M} = \mathbf{A}$ il y a convergence en une itération, mais dans laquelle il faut résoudre le problème de départ !

La convergence est cette fois-ci en :

$$\tau = \mathcal{O}\Big(\frac{1}{\sqrt{\operatorname{cond}\mathbf{M}^{-1}\mathbf{A}}}\Big) \tag{1.69}$$

Comme exemples de préconditionneurs classiques, on a la factorisation de Cholesky incomplète, SSOR, etc. L'algorithme 1.7 présente cette méthode. Le choix et le développement de préconditionneurs adaptés font encore aujourd'hui l'objet de recherches.

1.6.5 Autres méthodes itératives

Pour information, on peut citer :

- les méthodes dites « rapides » dédiées à des applications particulières : Fourier ($\mathcal{O}(n \log n)$), multigrilles ($\mathcal{O}(n)$) ;
- pour $\mathbf{A}$ symétrique et définie : Chebyshev [61] ;
- pour $\mathbf{A}$ symétrique mais non définie : MinRES (résidu minimum, *minimal residual*), SymmLQ (*symmetric* LQ) [55] ;
- pour $\mathbf{A}$ non symétrique connaissant $\mathbf{A}^\top$: QMR (*quasi minimal residual*) [22] ;
- pour $\mathbf{A}$ non symétrique : CGS (*conjugate gradient square*) [62], BiCGStab (*bi-conjugate gradient stabilized*) [65], GMRES (*generalized minimal residual*) [60], GMRES restarted...

Vecteur de départ $\mathbf{x}^{(0)}$
Résidu d'équilibre $\mathbf{r}^{(0)} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
tant que R $> \varepsilon$, itérer sur k
  résoudre $\mathbf{M}\mathbf{z}^{(k-1)} = \mathbf{r}^{(k-1)}$
  $\rho_{k-1} \leftarrow {\mathbf{r}^{(k-1)}}^{\top} \mathbf{z}^{(k-1)}$
  si $k = 1$ **alors**
    $\mathbf{d}^{(1)} \leftarrow \mathbf{z}^{(0)}$
  sinon
    $\beta_{k-1} \leftarrow \frac{\rho_{k-1}}{\rho_{k-2}}$
    $\mathbf{d}^{(k)} \leftarrow \mathbf{z}^{(k-1)} + \beta_{k-1}\mathbf{d}^{(k-1)}$
  fin
  $\mathbf{q}^{(k)} \leftarrow \mathbf{A}\mathbf{d}^{(k)}$
  $\alpha_k \leftarrow \frac{\rho_{k-1}}{{\mathbf{d}^{(k)}}^{\top}\mathbf{q}^{(k)}}$
  $\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k-1)} + \alpha_k \mathbf{d}^{(k)}$
  $\mathbf{r}^{(k)} \leftarrow \mathbf{b}^{(k)} - \mathbf{A}\mathbf{x}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k \mathbf{q}^{(k)}$
fin

Algorithme 1.7 — Gradient conjugué préconditionné

1.7 Critères d'arrêt

Le critère d'arrêt recherché est celui de l'écart en solution tel que $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}$ soit suffisamment petit. Malheureusement, $\mathbf{x}$ est la solution à convergence et on ne peut atteindre l'erreur en solution au cours des itérations. La quantité que l'on peut espérer atteindre est liée au résidu :

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} = -\mathbf{A}\mathbf{e}^{(k)} \tag{1.70}$$

Un critère souvent employé est le suivant :

$$\frac{\|\mathbf{r}^{(k)}\|}{\|\mathbf{A}\mathbf{x}^{(k)}\| + \|\mathbf{b}\|} < \varepsilon \tag{1.71}$$

où ε est un seuil fixé *a priori*, que l'on peut quantifier par rapport à la précision souhaitée et la précision machine. Parfois, on remplace au dénominateur $\|\mathbf{A}\mathbf{x}^{(k)}\|$ par $\|\mathbf{b}\|$. Le problème vient du fait, comme on a déjà pu le voir dans un exemple, que l'erreur en solution et le résidu sont liés de la façon suivante :

$$\|\mathbf{e}^{(k)}\| \leqslant \|\mathbf{A}^{-1}\| \tag{1.72}$$

donc :

$$\frac{\mathbf{r}^{(k)}}{\mathbf{b}} < \varepsilon \;\Rightarrow\; \frac{\mathbf{e}^{(k)}}{x} < \varepsilon \,\mathrm{cond}\,\mathbf{A} \tag{1.73}$$

et, en tout état de cause, le paramètre ε devrait être choisi aussi fonction du conditionnement de $\mathbf{A}$, qu'il faut donc estimer pour chaque problème traité.

Des critères d'arrêt en cas de non-convergence sont aussi mis en place. Par exemple, $\|\mathbf{r}^{(k)}\|$ décroît *trop lentement* ou ne décroît plus (mais il faut faire attention aux plateaux possibles avec certaines méthodes), ou la résolution est trop longue (critère en nombre maxi d'itérations). Il s'agit alors de sortie en cas d'erreur.

2 Systèmes aux valeurs propres

On considère ici les problèmes, dits *aux valeurs propres*, de la forme : trouver les caractéristiques propres, c'est-à-dire les couples $(\lambda, \mathbf{x})$ où $\mathbf{x} \neq \mathbf{0}$ tels que $\mathbf{Ax} = \lambda\mathbf{x}$ et l'on se restreindra aux cas où $\mathbf{A}$ est symétrique.

Un problème aux valeurs propres généralisé est de la forme $\mathbf{Kx} = \omega^2\mathbf{Mx}$, $\mathbf{K}$ symétrique, définie positive et $\mathbf{M}$ symétrique positive, avec $\mathbf{x} \neq \mathbf{0}$. Dans ce cas, on peut utiliser la factorisation de Cholesky de $\mathbf{M}$, soit $\mathbf{M} = \mathbf{LL}^\top$ et avec $\mathbf{y} = \mathbf{L}^\top\mathbf{x}$, on est ramené au problème précédent $\mathbf{L}^{-1}\mathbf{KL}^{-\top}\mathbf{y} = \omega^2\mathbf{y}$.

En général, dans les problèmes qui nous concernent, on ne veut qu'environ 30 valeurs et/ou vecteurs propres, de préférence à basse fréquence (dynamique lente, vibrations [26, 27, 58], flambage...). Dans le cas où il y a de l'amortissement, le système à résoudre est :

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{Kx} = \mathbf{f}(t) \tag{2.1}$$

La matrice d'amortissement, $\mathbf{C}$, est souvent mal maîtrisée et si l'amortissement est peu influent à basse fréquence, ce qui n'est pas évident pour tous les cas, on est ramené à :

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{Kx} = \mathbf{f}(t) \tag{2.2}$$

Avec une discrétisation éléments finis standard, les hautes fréquences obtenues numériquement sont éloignées des vraies.

Enfin, on prévoit parfois d'avoir des problèmes si les valeurs propres sont mal séparées, proches ou même confondues en cas de symétries, par exemple. Une référence de base concernant ces problèmes est [3].

2.1 Préliminaires

2.1.1 Rappels

Matrice quelconque

λ est valeur propre de $\mathbf{A}$ ssi $\det(\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0}$. Dans $\mathbb{C}$, $\mathbf{A}$ admet n valeurs propres distinctes ou non, réelles ou complexes. $\mathbf{A}$ a n valeurs propres distinctes ssi $\mathbf{A}$ est diagonalisable.

Matrice réelle

$\mathbf{A}$ est symétrique ssi $\mathbf{A}$ est diagonalisable dans $\mathbb{R}$. $\mathbf{A}$ est symétrique, définie positive ssi :

- toutes les valeurs propres sont strictement positives ;
- $\exists\mathbf{B}$ symétrique, définie positive telle que $\mathbf{A} = \mathbf{B}^2$.

Dans $\mathbb{C}$, $\mathbf{A}$ admet n valeurs propres distinctes ou non, réelles ou complexes. $\mathbf{A}$ a n valeurs propres distinctes implique que $\mathbf{A}$ est diagonalisable.

Quotient de Rayleigh pour une matrice réelle, symétrique

Le quotient de Rayleigh est défini comme étant, pour $\mathbf{x} \neq \mathbf{0}$:

$$\mathrm{R}(\mathbf{x}) = \frac{\mathbf{x}^\top \mathbf{A}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} \quad \text{ou} \quad \mathrm{R}(\mathbf{x}) = \frac{\mathbf{x}^\top \mathbf{K}\mathbf{x}}{\mathbf{x}^\top \mathbf{M}\mathbf{x}} \tag{2.3}$$

Il vient alors l'équivalence suivante :

$$(\lambda, \mathbf{x}) \text{ est caractéristique propre de } \mathbf{A} \Leftrightarrow \begin{cases} \mathrm{R} \text{ est extremum en } \mathbf{x} \neq \mathbf{0} \\ \mathrm{R}(\mathbf{x}) = \lambda \end{cases} \tag{2.4}$$

La i^{e} valeur propre λ_i est le minimum de $\mathrm{R}(\mathbf{x}), \mathbf{x} \neq \mathbf{0}$ sur le sous espace orthogonal aux précédents vecteurs propres. De plus, en *dimension finie*, la valeur propre maxi est $\lambda_n = \max_{\mathbf{x}\neq\mathbf{0}} \mathrm{R}(\mathbf{x})$ dans l'espace complet de dimension n.

2.1.2 Domaine d'emploi des différentes méthodes

En comparaison de la résolution d'un système linéaire, la résolution d'un système aux valeurs propres est beaucoup plus coûteuse. En fait, il s'agit d'un problème non-linéaire, et beaucoup de méthodes s'appuient sur la résolution d'une succession de systèmes linéaires, en interne. Elles souffriront alors des problèmes affectant ces solveurs, en particulier le mauvais conditionnement. L'ordre de grandeur de la taille des systèmes aux valeurs propres que l'on va considérer dans cette partie est de l'ordre de $n = 250$ à $n = 25\,000$.

La première méthode est celle de l'équation caractéristique. On peut en effet rechercher les zéros du polynôme caractéristique de la matrice, mais si les coefficients de ce polynôme peuvent être assez facilement atteints, une petite perturbation sur ceux-ci conduit à d'énormes variations sur ses racines. Cette méthode est donc réservée à des tailles de système réduites, jusqu'à $n \approx 10$. On peut montrer mathématiquement que, dès que $n \geqslant 5$, la méthode employée ne peut être qu'itérative : le problème ne peut plus être résolu numériquement en un nombre fini d'opérations.

Les méthodes de Jacobi, des puissances et de Householder vont ensuite pouvoir être employées pour des systèmes dont la taille va jusqu'à $n \approx 250$. Au delà, elle doivent être adaptées à un stockage bande, et le remplissage des matrices au cours des itérations doit être étudié. Avec les méthodes de réduction, ces techniques permettent d'atteindre des systèmes dont la taille va jusqu'à $n \approx 2\,500$. Jusqu'à $n \approx 25\,000$, les méthodes des puissances, plus ou moins couplées aux méthodes de sous-espaces, la méthode de Lanczos avec stratégie de réorthogonalisation permettent encore de traiter le problème.

Pour dépasser ces tailles de problème, il est ensuite nécessaire de faire appel à des techniques particulières. Par exemple, les techniques de parallélisation, vectorisation, optimisation des entrées-sorties, les méthodes de condensation et sous-structuration (Guyan-Irons, Craig-Bampton...) Ces dernières méthodes ne seront pas étudiées ici.

2.2 Méthode de Jacobi

Cette méthode, datant de 1846 [39], est décrite ici, dans le cas particulier où $\mathbf{A}$ est symétrique, à des fins de rappel uniquement puisqu'elle est peu efficace pour les problèmes qui nous concernent.

tant que $\mathrm{R} > \varepsilon$, itérer sur k
  choix du terme à annuler $\mathrm{A}_{p,q}$
  $\nu \leftarrow \frac{\mathrm{A}_{q,q}^{(k-1)} - \mathrm{A}_{p,p}^{(k-1)}}{2\mathrm{A}_{p,q}^{(k-1)}}$
  calcul de la tangente t
  calcul du cosinus $c \leftarrow 1/\sqrt{1+t^2}$
  calcul du sinus $s \leftarrow ct$
  $\mathbf{A}^{(k)} \leftarrow \mathbf{A}^{(k-1)}$
  $\mathbf{A}_{p,1:n}^{(k)} \leftarrow c\mathbf{A}_{p,1:n}^{(k-1)} - s\mathbf{A}_{q,1:n}^{(k-1)}$
  $\mathbf{A}_{q,1:n}^{(k)} \leftarrow s\mathbf{A}_{p,1:n}^{(k-1)} + c\mathbf{A}_{q,1:n}^{(k-1)}$
  $\mathrm{A}_{p,p}^{(k)} \leftarrow \mathrm{A}_{p,p}^{(k-1)} - t\mathrm{A}_{p,q}^{(k-1)}$
  $\mathrm{A}_{q,q}^{(k)} \leftarrow \mathrm{A}_{q,q}^{(k-1)} + t\mathrm{A}_{p,q}^{(k-1)}$
  $\mathrm{A}_{p,q}^{(k)} \leftarrow 0$
  $\mathrm{A}_{q,p}^{(k)} \leftarrow 0$
fin

Algorithme 2.1 — Méthode de Jacobi

Dans cette méthode, on cherche à ramener la matrice $\mathbf{A}$ à une forme diagonale pour trouver tous ses valeurs propres et vecteurs propres, et ce par une séquence de transformations orthogonales :

$$\mathbf{A}^{(k+1)} = {\mathbf{R}^{(k)}}^{\top}\mathbf{A}^{(k)}\mathbf{R}^{(k)} \tag{2.5}$$

avec les rotations $\mathbf{R}^{(k)}$ vérifiant ${\mathbf{R}^{(k)}}^{\top}\mathbf{R}^{(k)} = \mathbf{I}$, transformations qui sont des changements de base de type isométries ne modifiant pas les valeurs propres.

On commence par chercher l'élément hors diagonale de module maximal :

$$|\mathrm{A}_{p,q}^{(k)}| = \max_{i \neq j} |\mathrm{A}_{i,j}^{(k)}| \tag{2.6}$$

On construit alors la matrice de rotation d'un angle θ *ad hoc* dans le plan $(\mathbf{e}_p, \mathbf{e}_q)$ de façon à annuler dans l'itéré suivant les termes $\mathrm{A}_{q,p}^{(k+1)}$ et $\mathrm{A}_{p,q}^{(k+1)}$:

$$\mathrm{R}^{(k)} = 1 + (\cos\theta - 1)(\mathbf{e}_p\mathbf{e}_p^{\top} + \mathbf{e}_q\mathbf{e}_q^{\top}) + (\sin\theta - 1)\mathbf{e}_p\mathbf{e}_p^{\top} - (\sin\theta + 1)\mathbf{e}_q\mathbf{e}_q^{\top} \tag{2.7}$$

avec :

$$\mathrm{A}_{p,q}^{(k+1)} = \frac{1}{2}\sin 2\theta\left[\mathrm{A}_{p,p}^{(k)} - \mathrm{A}_{q,q}^{(k)}\right] + \frac{1}{2}\cos 2\theta\,\mathrm{A}_{p,q}^{(k)} = 0 \tag{2.8}$$

ce qui entraîne :

$$\operatorname{cotan} 2\theta = \frac{\mathrm{A}_{p,p}^{(k)} - \mathrm{A}_{q,q}^{(k)}}{\mathrm{A}_{p,q}^{(k)}} \tag{2.9}$$

Seuls changent les termes sur les lignes ou colonnes p ou q. Dans l'algorithme 2.1, la tangente t est calculée comme suit :

$$t = \begin{cases} 1 & \text{si } \nu = 0 \\ -\nu + \operatorname{sgn}(\nu)\sqrt{1+\nu^2} & \text{sinon} \end{cases} \tag{2.10}$$

Au cours des itérations, un terme nul peut redevenir non nul mais :

$$a^{(k)} = \sum_{i \neq j} \mathrm{A}_{i,j}^{(k)\,2} \underset{k\to\infty}{\to} 0 \tag{2.11}$$

En effet :

$$\begin{aligned} a^{(k+1)} &= \sum_{\substack{i\neq j,p \\ q,j\neq p,q}} \underbrace{[\mathrm{A}_{i,j}^{(k+1)}]^2}_{[\mathrm{A}_{i,j}^{(k)}]^2} + \sum_{i\neq p,q} \underbrace{\left([\mathrm{A}_{i,p}^{(k+1)}]^2 + [\mathrm{A}_{i,q}^{(k+1)}]^2\right)}_{[\mathrm{A}_{i,p}^{(k)}]^2 + [\mathrm{A}_{i,q}^{(k)}]^2} + \sum_{j\neq p,q} \underbrace{\left([\mathrm{A}_{p,j}^{(k+1)}]^2 + [\mathrm{A}_{q,j}^{(k+1)}]^2\right)}_{[\mathrm{A}_{p,j}^{(k)}]^2 + [\mathrm{A}_{q,j}^{(k)}]^2} \\ &= a^{(k)} - 2[\mathrm{A}_{p,q}^{(k)}]^2 \end{aligned} \tag{2.12}$$

donc $a^{(k+1)} < a^{(k)}$. Cet algorithme est stable et simple mais la convergence est lente et il s'avère coûteux en accès mémoire si n est grand. On a les estimations suivantes :

$$a^{(k+1)} \leqslant \left(1 - \frac{2}{n(n-1)}\right) a^{(k)} \quad \text{et} \quad \tau \approx \frac{2}{n^2} \tag{2.13}$$

2.3 Méthode des puissances

On se place ici dans le cas où $\mathbf{A}$ est symétrique, définie positive et dont les valeurs propres sont toutes simples, rangées par ordre de module décroissant λ_i. On construit un itéré de

$x^{(0)}$, $\|x^{(0)}\| = 1$
tant que $\mathrm{R} > \varepsilon$, itérer sur k
  $v \leftarrow \mathrm{K}x^{(k-1)}$
  résoudre $\mathrm{M}\mathbf{x}^{(k)} = v$
  $\alpha \leftarrow \|\mathbf{x}^{(k)}\|$
  $\lambda^{(k)} \leftarrow \mathrm{R}(\mathbf{x}^{(k)}) = (\mathbf{x}^{(k)\top}\mathrm{K}\mathbf{x}^{(k)})/(\mathbf{x}^{(k)\top}\mathrm{M}\mathbf{x}^{(k)})$
  normalisation $\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k)}/\alpha$
fin

Algorithme 2.2 — Méthode des puissances

départ $\mathbf{x}^{(0)}$ puis les itérés successifs par récurrence de la manière suivante :

$$\mathbf{x}^{(k+1)} = \mathbf{A}\mathbf{x}^{(k)} \tag{2.14}$$

Si $(\mathbf{v}_i)_i$ est la base propre, on peut écrire les itérés dans cette base. En particulier :

$$\mathbf{x}^{(0)} = \sum_i \alpha_i \mathbf{v}_i \tag{2.15}$$

alors :

$$\mathbf{x}^{(k)} = \sum_i \lambda_i^k \alpha_i \mathbf{v}_i = \lambda_1^k \left[\alpha_1 \mathbf{v}_1 + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^k \mathbf{v}_i\right] \tag{2.16}$$

et si $\alpha_1 \neq 0$ puisque $|\lambda_i|/|\lambda_1| < 1$ alors :

$$\mathbf{x}^{(k)} \underset{k\to\infty}{\to} \lambda_1^k \alpha_1 \mathbf{v}_1 \tag{2.17}$$

En pratique, il est utile de normer $\mathbf{x}^{(k)}$ à chaque itération pour éviter l'explosion et dans ce cas obtenir $\mathbf{x}^{(k)} \to v_1$. Pour comprendre le comportement de la convergence, prenons $\mathbf{x}^{(0)} = \sum_i \mathbf{v}_i$, alors :

$$\frac{\mathbf{x}^{(k)}}{\lambda_1^k} = \mathbf{v}_1 + \sum_{i=2}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^k \mathbf{v}_i \tag{2.18}$$

donc :

$$\frac{\left\| \frac{\mathbf{x}^{(k+1)}}{\lambda_1^{k+1}} - \mathbf{v}_1 \right\|}{\left\| \frac{\mathbf{x}^{(k)}}{\lambda_1^{k}} - \mathbf{v}_1 \right\|} = \left[\frac{\sum_{i=2}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^{2(k+1)}}{\sum_{i=2}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^{2k}} \right]^{1/2} \underset{k\to\infty}{\to} \frac{\lambda_2}{\lambda_1} \tag{2.19}$$

Le taux de convergence est alors estimé comme suit :

$$\tau \approx -\ln \frac{\lambda_2}{\lambda_1} = \ln \frac{\lambda_1}{\lambda_2} \tag{2.20}$$

Bien entendu, on peut avoir des ennuis si au départ $\mathbf{x}^{(0)}$ est orthogonal à $\mathbf{v}_1$. Par rapport à la méthode de Jacobi, la convergence est maintenant indépendante de la taille du problème, n.

Cas de valeurs propres multiples ou proches

Dans le cas de valeurs propres multiples λ_1, la convergence se fait vers λ_1 et vers une combinaison linéaire des vecteurs propres associés.

Dans le cas de valeurs propres proches, $\lambda_2 = \lambda_1 + \delta\lambda$, on a $\tau \approx \delta\lambda/\lambda_1$ et la convergence est donc très lente.

Contrôle de la convergence

Comme critère de convergence sur les valeurs propres, on peut utiliser le fait que $\|\mathbf{x}^{(k+1)}\|/\|\mathbf{x}^{(k)}\|$ se stabilise, et pour la convergence sur les vecteurs propres, $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| < \varepsilon$ ou :

$$\frac{\|\mathbf{A}\mathbf{x}^{(k+1)} - \lambda^{(k+1)}\mathbf{x}^{(k+1)}\|}{\|\mathbf{A}\mathbf{x}^{(k+1)} + \lambda^{(k+1)}\mathbf{x}^{(k+1)}\|} < \varepsilon' \tag{2.21}$$

On conseille aussi d'utiliser la norme infinie pour ces tests.

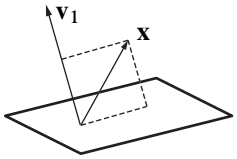


Figure 2.1 — Projection orthogonale à $\mathbf{v}_1$

Obtention des modes suivants

La technique utilisée est la déflation orthogonale : on prend $\mathbf{x}^{(0)}$ orthogonal à $\mathbf{v}_1$. Pour éviter les pertes d'orthogonalité par cumul d'arrondis, on réorthogonalise à chaque itération. Pour cela, on utilise un projecteur comme décrit sur la figure 2.1 :

$$\mathbf{P}_1 = \mathbf{I} - \frac{\mathbf{v}_1 \mathbf{v}_1^\top}{\mathbf{v}_1^\top \mathbf{v}_1} \tag{2.22}$$

d'où :

$$\mathbf{P}_1 \mathbf{x} = \mathbf{x} - \frac{\mathbf{v}_1^\top \mathbf{x}}{\mathbf{v}_1^\top \mathbf{v}_1} \mathbf{v}_1 \tag{2.23}$$

On a alors convergence vers $(\lambda_2, \mathbf{v}_2)$. On peut procéder de façon similaire pour accéder à la troisième caractéristique propre, et ainsi de suite.

2.4 Méthode des itérations inverses

Comme décrit dans l'algorithme 2.3, on choisit encore un $\mathbf{x}^{(0)}$ arbitraire : les solutions classiques sont de prendre un vecteur de composantes aléatoires, normé, quitte à essayer avec plusieurs vecteurs de départ pour vérifier la convergence vers la même valeur. On résoud

$\mathbf{x}^{(0)}$, $\|\mathbf{x}^{(0)}\| = 1$
tant que $\mathrm{R} > \varepsilon$, itérer sur k
  $\mathbf{v} \leftarrow \mathbf{M}\mathbf{x}^{(k-1)}$
  résoudre $\mathrm{K}\mathbf{x}^{(k)} = v$
  $\alpha \leftarrow \|\mathbf{x}^{(k)}\|$
  $\lambda^{(k)} \leftarrow \mathrm{R}(\mathbf{x}^{(k)}) = (\mathbf{x}^{(k)\top}\mathrm{K}\mathbf{x}^{(k)})/(\mathbf{x}^{(k)\top}\mathrm{M}\mathbf{x}^{(k)})$
  normalisation $\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k)}/\alpha$
fin

Algorithme 2.3 — Itérations inverses

ensuite $\mathbf{A}\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$ et on norme $\mathbf{x}^{(k+1)}$. Il s'agit bien de la méthode des puissances, appliquée à $\mathbf{A}^{-1}$, qui commence donc par converger vers la valeur propre de $\mathbf{A}$ de module le plus élevé.

Bien évidemment, il est simple de démontrer que le taux de convergence est tel que :

$$\tau \approx \ln \frac{|\lambda_{n-1}|}{|\lambda_n|} \tag{2.24}$$

Comme pour la méthode des puissances, on converge d'autant mieux que les valeurs propres sont séparées... d'où la méthode du paragraphe suivant.

2.5 Méthode des itérations inverses avec décalage spectral

Appelée encore méthode de Wielandt, cette méthode suppose que l'on connaît une approximation μ de la valeur propre cherchée λ.

2.5.1 Principe et convergence

On utilise la matrice décalée $\mathbf{A}^\star = \mathbf{A} - \mu\mathbf{I}$, on s'intéresse donc à $\mathbf{A}^\star\mathbf{x} = \lambda^\star\mathbf{x}$ et $\lambda^\star = \lambda - \mu$. Comme on converge vers la valeur propre $|\lambda^\star|$ la plus petite, on va donc trouver λ_i la plus proche de μ. Le taux de convergence est :

$$\tau = -\ln\Big(\inf\Big[\frac{|\lambda_i - \mu|}{|\lambda_{i-1} - \mu|}, \frac{|\lambda_i - \mu|}{|\lambda_{i+1} - \mu|}\Big]\Big) \tag{2.25}$$

et il est par conséquent très élevé si μ est très proche de λ_i. Voici quelques remarques :

- μ est proche de λ_i ; le système $\mathbf{A}^\star\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)}$ est très mal conditionné. En fait, la méthode y est peu sensible : l'erreur en solution tend à s'aligner dans la direction du vecteur propre et est éliminée par la normalisation ;
- Cette méthode permet aussi d'accéder aux valeurs propres d'une région donnée du spectre ;
- On peut construire de cette manière la méthode des itérations inverses de Rayleigh : on change l'estimation μ par $\lambda^{(k)}$ à chaque itération. On obtient alors une convergence cubique :

$$\|\mathbf{x}^{(k+1)} - \mathbf{v}\| = \mathcal{O}(\|\mathbf{x}^{(k)} - \mathbf{v}\|^3) \tag{2.26}$$

Cette méthode est coûteuse puisque l'on factorise à chaque itération.

2.5.2 Exemples d'utilisation dans un code éléments finis

On considère le système $\mathbf{Kx} = \omega^2\mathbf{Mx}$.

Problème 1 On veut trouver le mode proche de la pulsation ω_0. On procède alors de la façon suivante :

1. décalage : $\mathbf{K}^\star \leftarrow \mathbf{K} - \omega_0^2\mathbf{M}$
2. factorisation $\mathbf{LDL}^\top$ de $\mathbf{K}^\star$; soit $\mathrm{N}^\star$ le nombre de termes strictement négatifs de $\mathbf{D}$
3. $\mathbf{x}^{(0)}$ aléatoire puis itérations inverses avec décalage de ω_0 sur $\mathbf{K}^\star\mathbf{x} = \lambda^\star\mathbf{Mx}$ pour obtenir $(\lambda^\star, \mathbf{x})$
4. $\omega^2 \leftarrow \omega_0^2 + \lambda^\star$
5. le numéro du mode trouvé est $\mathrm{N}^\star$ si $\omega^2 < \omega_0^2$ ou $\mathrm{N}^\star + 1$ si $\omega^2 \geqslant \omega_0^2$.

Problème 2 On veut trouver tous les modes dans l'intervalle $[\omega_1, \omega_2]$. On procède alors de la façon suivante :

1. pour trouver le nombre de modes dans $[\omega_1, \omega_2]$, on compte le nombre de termes strictement négatifs de $\mathbf{D}$ dans les factorisations de $\mathbf{K} - \omega_1^2\mathbf{M}$ et $\mathbf{K} - \omega_2^2\mathbf{M}$;
2. on procède par itérations inverses avec décalage et déflations orthogonales successives.

2.6 Transformation de Householder

2.6.1 Factorisation QR

On cherche à mettre $\mathbf{A}$ sous la forme $\mathbf{A} = \mathbf{QR}$ où $\mathbf{R}$ est triangulaire supérieure et $\mathbf{Q}$ orthogonale, par applications successives de transformations de Householder à $\mathbf{A} \rightarrow \mathbf{HA}$ [34, 67]. $\mathbf{H}$ est de la forme $\mathbf{H} = \mathbf{I} - \mathbf{2uu}^\top$ avec $\|\mathbf{u}\| = 1$. Il s'agit cette fois d'une symétrie par rapport à un plan de normale $\mathbf{u}$ comme illustré sur la figure 2.2. En conséquence, $\mathbf{H}$ est symétrique, orthogonale, et conserve les valeurs propres. Ces transformations de Householder

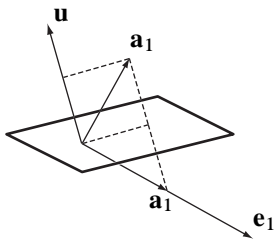


Figure 2.2 — Transformation de Householder de $\mathbf{a}_1$

sont appliquées sur les colonnes de $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n]$ successivement. Par exemple, sur la première, on veut $\mathbf{Ha}_1 = \alpha\mathbf{e}_1$ d'où le choix :

$$\alpha = \|\mathbf{a}_1\| = \|\mathbf{Ha}_1\| > 0 \tag{2.27}$$

alors $\mathbf{a}_1 - 2\mathbf{u}(\mathbf{u}^\top\mathbf{a}_1) = \alpha\mathbf{e}_1$ qui implique $\mathbf{a}_1 - \alpha\mathbf{e}_1 = 2\mathbf{u}(\mathbf{u}^\top\mathbf{a}_1)$. En prenant le produit scalaire par $\mathbf{a}_1$, on obtient :

$$\alpha^2 - \alpha\mathbf{a}_1^\top\mathbf{e}_1 = 2(\mathbf{u}^\top\mathbf{a}_1)^2 \tag{2.28}$$

d'où :

$$\mathbf{u}^\top\mathbf{a}_1 = \sqrt{\frac{1}{2}\alpha(\alpha - \mathbf{a}_1^\top\mathbf{e}_1)} \tag{2.29}$$

On peut noter que le terme $(\alpha - \mathbf{a}_1^\top\mathbf{e}_1)$ est non nul si $\mathbf{a}_1$ n'est pas déjà parallèle à $\mathbf{e}_1$. Finalement, on en déduit le vecteur cherché :

$$\mathbf{u} = \frac{\mathbf{a}_1 - \alpha\mathbf{e}_1}{2\mathbf{u}^\top\mathbf{a}_1} \tag{2.30}$$

Le procédé se poursuit en considérant ensuite la deuxième colonne de $\mathbf{A}$ et $\mathbf{H}$ telle que :

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2 \end{bmatrix} \tag{2.31}$$

Le vecteur suivant $\mathbf{u}_2$ est déterminé à partir de $\mathbf{e}_2$ et $\mathbf{a}_2$. C'est une méthode directe en $n-1$ étapes.

On peut regarder les propriétés de remplissage de $\mathbf{A}$ lors des transformations mais c'est assez délicat. En général, on ne stocke pas la matrice $\mathbf{Q}$; l'application de $\mathbf{Q}$ à un vecteur est coûteuse ($4n$ opérations), cette méthode n'est donc pas appliquée à la résolution de systèmes linéaires. Cet algorithme se trouve aussi sous le nom de Householder-Businger-Golub. La méthode est présentée par l'algorithme 2.4. En ce qui concerne le stockage, $\mathbf{v}_k$

boucle sur $k = 1, 2, \ldots, n-1$
- $\delta \leftarrow \mathbf{A}_{k:n,k}^{(k)\top} \mathbf{A}_{k:n,k}^{(k)}$
- $\alpha \leftarrow \sqrt{\delta}$
- $\beta \leftarrow \frac{1}{\delta - \alpha \mathrm{A}_{k,k}^{(k)}}$
- $\mathbf{v}_{k:n}^{(k)} \leftarrow \mathbf{A}_{k:n,k}^{(k)}$
- $v_k^{(k)} \leftarrow v_k^{(k)} - \alpha$
- $\mathbf{z}_{k+1:n}^{\top} \leftarrow \beta \mathbf{v}_{k:n}^{(k)\top} \mathbf{A}_{k:n,k+1:n}^{(k)}$
- $\mathbf{A}_{k:n,k+1:n}^{(k+1)} \leftarrow \mathbf{A}_{k:n,k+1:n}^{(k)} - \mathbf{v}_{k:n}^{(k)} \mathbf{z}_{k+1:n}^{\top}$
- $\mathrm{R}_{k,k} \leftarrow \alpha$
- $\mathbf{R}_{k,k+1:n} \leftarrow \mathbf{A}_{k,k+1:n}^{(k+1)}$

fin

Algorithme 2.4 — Transformation de Householder pour la factorisation QR

écrase $\mathbf{A}_{1:k,k}$ en mémoire, $\mathbf{R}_{k+1,k+1:n}$ écrase $\mathbf{A}_{k+1,k+1:n}$. $\mathrm{R}_{k,k}$ doit être stocké ailleurs pour chaque itération, d'où un vecteur de dimension n supplémentaire. $\mathbf{Q}$ n'est en général pas stockée, elle pourrait être reconstruite à partir des $\mathbf{v}_k$. Enfin, $\mathbf{z}$ est un vecteur de stockage supplémentaire.

La complexité de cet algorithme pour une matrice pleine de taille n est de l'ordre de $2n^3/3$.

2.6.2 Détermination des valeurs propres par la méthode QR

Cette méthode est attribuée à Francis et Vera N. Kublanovskaya en 1961. On considère le problème $\mathbf{Ax} = \lambda \mathbf{x}$ avec des valeurs propres simples. On procède alors de la façon suivante :

1. $\mathbf{A}^{(1)} = \mathbf{A}$;
2. factorisation QR de $\mathbf{A}^{(k)} = \mathbf{Q}^{(k)} \mathbf{R}^{(k)}$;
3. calcul de $\mathbf{A}^{(k+1)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)}$;
4. itération suivante.

Propriétés

— si les valeurs propres sont toutes simples, $\mathbf{A}^{(k)}$ tend vers la matrice diagonale des valeurs propres. $\mathbf{Q}^{(k)}$ tend vers la matrice des vecteurs propres ;
— il y a conservation des bandes de $\mathbf{A}$. C'est une méthode performante quand $\mathbf{A}$ est tri-diagonale de petite taille... le coût d'une itération est alors d'environ $10n$. Quand $\mathbf{A}$ est pleine, une factorisation coûte de l'ordre de $2n^3/3$ opérations et un produit $\mathbf{QR}$, de l'ordre de $n^3/2$ opérations.

On peut aussi appliquer des translations (décalages) :

$$\begin{aligned} \mathbf{A}^{(k)} - \alpha^{(k)}\mathbf{I} &= \mathbf{Q}^{(k)}\mathbf{R}^{(k)} \\ \mathbf{A}^{(k+1)} &= \mathbf{R}^{(k)}\mathbf{Q}^{(k)} + \alpha^{(k)}\mathbf{I} \end{aligned} \tag{2.32}$$

Les matrices restent toujours semblables, donc conservent les valeurs propres. Concernant la convergence de la méthode, avec des décalages, et en prenant $\alpha^{(k)} = \mathrm{A}_{n,n}^{(k)}$:

$$|\lambda_n - \mathrm{A}_{n,n}^{(k)}| \underset{k\to\infty}{\to} 0 \tag{2.33}$$

alors :

$$\|(\mathbf{A} - \lambda_{k+1}\mathbf{I})\mathbf{q}^{(k)}\| = \mathcal{O}(\|(\mathbf{A} - \lambda_k\mathbf{I})\mathbf{q}^{(k-1)}\|^3) \tag{2.34}$$

Sans translation, la convergence reste linéaire.

2.6.3 Forme de Hessenberg

Comme son nom l'indique, cette méthode est due à Gerhard Hessenberg (1874-1925) pour qui on pourra consulter [31] et est dédiée aux matrices A réelles symétriques. On cherche

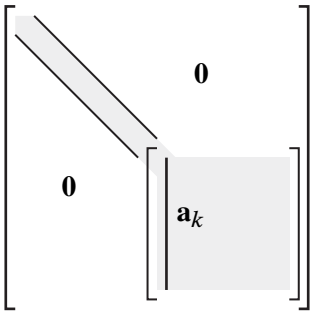


Figure 2.3 — Matrice en cours de tri-diagonalisation

cette fois à rendre $\mathbf{A}$ tri-diagonale. Il s'agit de la même technique colonne par colonne que précédemment. On prend ici une transformation qui ne s'applique qu'à la deuxième colonne pour la première étape :

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{H}} \end{bmatrix} \tag{2.35}$$

mais comme A est symétrique, il faut appliquer les transformations à gauche et à droite :

$$\mathbf{H}^{-1}\mathbf{A}\mathbf{H} = \mathbf{H}^{\top}\mathbf{A}\mathbf{H} = \mathbf{H}\mathbf{A}\mathbf{H} \tag{2.36}$$

$\mathbf{A}^{(k)}$ prend donc la forme de la figure 2.3 et le principe consiste à travailler sur le vecteur $[\mathrm{A}^{(k)}_{k,k+1} \cdots \mathrm{A}^{(k)}_{k,n}]^{\top}$ pour le rendre colinéaire à $\mathbf{e}_{k+1}$. Cette méthode a la propriété intéressante de ne pas modifier les valeurs propres. C'est une méthode directe en $n-2$ étapes et $\mathbf{A}$ reste symétrique. Travaillons sur le premier itéré :

$$\mathbf{A}^{(1)} = \begin{bmatrix} \mathrm{A}_{1,1} & \tilde{\mathbf{a}}_1^{\top} \\ \tilde{\mathbf{a}}_1 & \tilde{\mathbf{A}}^{(1)} \end{bmatrix} \tag{2.37}$$

on prend :

$$\tilde{\mathbf{H}}^{(1)} = 1 - 2\tilde{\mathbf{u}}\tilde{\mathbf{u}}^{\top} \tag{2.38}$$

en souhaitant avoir :

$$\tilde{\mathbf{H}}^{(1)}\tilde{\mathbf{a}}_1 = \alpha\tilde{\mathbf{e}}_1 \tag{2.39}$$

En fait, ici $\tilde{\mathbf{e}}_1 = \mathbf{e}_2$ et on procède comme précédemment pour avoir $\tilde{\mathbf{u}}$. On cherche donc

boucle sur $k = 1, 2, \ldots, n-1$
- $\delta \leftarrow \mathbf{A}^{(k)\top}_{k+1:n,k}\mathbf{A}^{(k)}_{k+1:n,k}$
- $\alpha \leftarrow \sqrt{\delta}$
- $\beta \leftarrow 1/(\delta - \alpha \mathrm{A}^{(k)}_{k+1,k})$
- $\mathbf{v}^{(k)}_{k+1:n} \leftarrow \mathbf{A}^{(k)}_{k+1:n,k}$
- $v^{(k)}_{k+1} \leftarrow v^{(k)}_{k+1} - \alpha$
- $\mathbf{z}_{k+1:n} \leftarrow \beta\mathbf{A}^{(k)}_{k+1:n,k+1:n}\mathbf{v}^{(k)}_{k+1:n}$
- $\gamma \leftarrow \beta/2\mathbf{v}^{(k)\top}_{k+1:n}\mathbf{z}_{k+1:n}$
- $\mathbf{z}_{k+1:n} \leftarrow \mathbf{z}_{k+1:n} - \gamma\mathbf{v}^{(k)}_{k+1:n}$
- $\mathbf{A}^{(k+1)}_{k+1:n,k+1:n} \leftarrow \mathbf{A}^{(k)}_{k+1:n,k+1:n} - \mathbf{v}^{(k)}_{k+1:n}\mathbf{z}^{\top}_{k+1:n} - \mathbf{z}_{k+1:n}\mathbf{v}^{(k)\top}_{k+1:n}$

fin

Algorithme 2.5 — Transformation de Householder

$\tilde{\mathbf{H}}^{(1)} = \mathbf{I} - \beta\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\top}$ avec $\alpha = \tilde{\mathbf{a}}_1^{\top}\tilde{\mathbf{a}}_1$, $\beta = 1/(\alpha(\alpha - \mathrm{A}_{2,1}))$ et $\tilde{\mathbf{y}} = \tilde{\mathbf{a}}_1 - \alpha\tilde{\mathbf{e}}_1$. On calcule ensuite la transformée par :

$$\begin{aligned} \tilde{\mathbf{H}}^{(1)}\tilde{\mathbf{A}}^{(1)}\tilde{\mathbf{H}}^{(1)} &= (1 - \beta\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\top})\tilde{\mathbf{A}}^{(1)}(1 - \beta\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\top}) \\ &= \tilde{\mathbf{A}}^{(1)} - \beta\tilde{\mathbf{y}}(\tilde{\mathbf{y}}^{\top}\tilde{\mathbf{A}}^{(1)}) - \beta(\tilde{\mathbf{A}}^{(1)}\tilde{\mathbf{y}})\tilde{\mathbf{y}}^{\top} + \beta^2\tilde{\mathbf{y}}(\tilde{\mathbf{y}}^{\top}\tilde{\mathbf{A}}^{(1)}\tilde{\mathbf{y}})\tilde{\mathbf{y}}^{\top} \end{aligned} \tag{2.40}$$

et en utilisant les intermédiaires $\tilde{\mathbf{z}} = \beta\tilde{\mathbf{A}}^{(1)}\tilde{\mathbf{y}}$ et $\tilde{\mathbf{t}} = \tilde{\mathbf{z}} - \frac{\beta}{2}(\tilde{\mathbf{y}}^{\top}\tilde{\mathbf{z}})\tilde{\mathbf{y}}$:

$$\begin{aligned} \tilde{\mathbf{H}}^{(1)}\tilde{\mathbf{A}}^{(1)}\tilde{\mathbf{H}}^{(1)} &= \tilde{\mathbf{A}}^{(1)} - \tilde{\mathbf{y}}\tilde{\mathbf{z}}^{\top} - \tilde{\mathbf{z}}\tilde{\mathbf{y}}^{\top} + \beta(\tilde{\mathbf{z}}^{\top}\tilde{\mathbf{y}})\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\top} \\ &= \tilde{\mathbf{A}}^{(1)} - \tilde{\mathbf{y}}\tilde{\mathbf{t}}^{\top} - \tilde{\mathbf{t}}\tilde{\mathbf{y}}^{\top} \end{aligned} \tag{2.41}$$

C'est la technique mise en œuvre dans l'algorithme 2.5, dans lequel $\mathbf{v}_k$ n'est en général pas conservé et où $\mathbf{z}$ est un vecteur de stockage supplémentaire. La réduction s'effectue en $n-1$ étapes.

2.7 Exemple d'utilisation dans un code éléments finis

On considère $\mathbf{Kx} = \lambda\mathbf{Mx}$ avec $\mathbf{K}$ symétrique, définie positive, $\mathbf{M}$ symétrique positive pour l'obtention de m valeurs propres, en basse fréquence.

2.7.1 Réduction dans un sous-espace

On cherche à trouver toutes les m valeurs propres dans un sous-espace de dimension $p \geqslant m$, défini par p vecteurs rangés dans la matrice de base $\mathbf{V} = [\mathbf{v}_1 \ \cdots \ \mathbf{v}_p]$. Éventuellement, les vecteurs $\mathbf{v}_i$ sont orthonormés au sens de $\mathbf{K}$, soit $\mathbf{V}^\top\mathbf{KV} = \hat{\mathbf{K}} = \mathbf{I}$.

On cherche alors les vecteurs propres $\mathbf{x}$ sous la forme $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$, $\hat{\mathbf{x}}$ est de dimension p. Le problème est alors $\mathbf{KV}\hat{\mathbf{x}} = \hat{\lambda}\mathbf{MV}\hat{\mathbf{x}}$, d'où :

$$\mathbf{V}^\top\mathbf{KV}\hat{\mathbf{x}} = \hat{\lambda}\mathbf{V}^\top\mathbf{MV}\hat{\mathbf{x}} \quad \Leftrightarrow \quad \hat{\mathbf{K}}\hat{\mathbf{x}} = \hat{\lambda}\hat{\mathbf{M}}\hat{\mathbf{x}} \tag{2.42}$$

Le problème (2.42) a été construit par une réduction de Ritz. Sa résolution complète permet de trouver p caractéristiques propres $(\hat{\lambda}, \hat{\mathbf{x}})$. On retourne alors dans l'espace de départ où $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$ est l'approximation d'un vecteur propre, $\hat{\lambda}$ est l'approximation de la valeur propre associée, approximations dont la qualité dépend du sous-espace choisi... Ce choix va être discuté dans les paragraphes suivants.

2.7.2 Technique itérative

Rapidement, il s'agit de :

1. choisir p vecteurs de départ ;
2. réduire à $\hat{\mathbf{K}}\hat{\mathbf{x}} = \hat{\lambda}\hat{\mathbf{M}}\hat{\mathbf{x}}$ dont on cherche les valeurs propres (si on a une approximation des valeurs propres, on fait un décalage) par une méthode QR ;
3. itérer en sens inverse avec décalage (souvent une seule itération suffit) pour avoir les approximations des vecteurs propres associés dans l'espace de départ ;
4. passer à l'itération suivante.

Comme la convergence, dépend du rapport des modules des valeurs propres, on a intérêt à conserver $p > m$ approximations. La règle conseillée est choisir une taille du sous-espace :

$$p = \min(2m, m + 8) \tag{2.43}$$

La convergence pour une valeur propre λ_i est proportionnelle à $|\lambda_i|/|\lambda_{p+1}|$.

2.8 Méthode de Lanczos

Cette méthode est due à Lanczos [44] en 1950. On travaille sur un sous-espace généré par la suite des itérés inverses d'un vecteur $\mathbf{v}^{(0)}$ de départ $[\mathbf{v}^{(0)} \ \mathbf{A}^{-1}\mathbf{v}^{(0)} \ \cdots]$ et sur une base orthonormée.

La convergence est très rapide : 3 à 4 itérations inverses par valeur propre convergée, et elle est indépendante de n ; cependant, la taille du sous-espace croît au cours des itérations.

Un inconvénient est la perte rapide d'orthogonalité sur les vecteurs successifs (elle se produit à chaque fois qu'une valeur propre converge), d'où des instabilités numériques. En

$\mathbf{v}^{(0)} \leftarrow \mathbf{0}$, $\beta^{(0)} \leftarrow 0$
Vecteur de départ $\mathbf{v}^{(1)}$, $\|\mathbf{v}^{(1)}\| = 1$
tant que R $> \varepsilon$, itérer sur k
  résoudre $\mathbf{A}\mathbf{z} = \mathbf{v}^{(k)}$
  $\alpha^{(k)} \leftarrow \mathbf{v}^{(k)\top}\mathbf{z}$
  $\mathbf{d} \leftarrow \mathbf{z} - \alpha\mathbf{v}^{(k)} - \beta^{(k-1)}\mathbf{v}^{(k-1)}$
  $\beta^{(k)} \leftarrow 1/\sqrt{\mathbf{d}^\top\mathbf{d}}$
  $\mathbf{v}^{(k+1)} \leftarrow \beta^{(k)}\mathbf{d}$
  recherche de $(\hat{\mathbf{x}}, \hat{\lambda})$ tel que $\mathbf{T}\hat{\mathbf{x}} = \hat{\lambda}\hat{\mathbf{x}}$
fin

Algorithme 2.6 — Méthode de Lanczos

pratique, pour pouvoir traiter de grands problèmes, il est nécessaire d'utiliser des stratégies de ré-orthogonalisation. Ces dernières ne seront pourtant pas décrites ici.

En termes simples, l'algorithme se résume à choisir un vecteur de départ $\mathbf{v}^{(0)}$ puis de résoudre le système $\mathbf{A}\mathbf{z}^{(k)} = \mathbf{v}^{(k)}$, d'orthogonaliser $\mathbf{z}^{(k)}$ par rapport aux $\mathbf{v}^{(j)}$, $j \leqslant k$ puis de normer $\mathbf{v}^{(k+1)} \leftarrow \mathbf{z}^{(k)}/\|\mathbf{z}^{(k)}\|$.

Pour démontrer le taux de convergence, il suffit de chercher $\mathbf{v}^{(k+1)}$ sous la forme :

$$\beta^{(k)}\mathbf{v}^{(k+1)} = \mathbf{A}^{-1}\mathbf{v}^{(k)} - \alpha^{(k)}\mathbf{v}^{(k)} - \beta^{(k-1)}\mathbf{v}^{(k-1)} \tag{2.44}$$

ce que l'on peut montrer par récurrence, avec $\mathbf{v}^{(k)\top}\mathbf{v}^{(j)} = 0$ pour $j = k-1$ et $j = k-2$, et $\mathbf{v}^{(k)\top}\mathbf{v}^{(k)} = 1$, pour avoir orthogonalité par rapport à tous les $\mathbf{v}^{(j)}$, $j < k$. Ces conditions donnent les valeurs de $\alpha^{(k)} = \mathbf{v}^{(k)}\mathbf{A}^{-1}\mathbf{v}^{(k)}$ et $\beta^{(k-1)} = \mathbf{v}^{(k)}\mathbf{A}^{-1}\mathbf{v}^{(k-1)}$. Matriciellement, en notant $\mathbf{V}$ de taille (n,k) la matrice qui contient tous les $\mathbf{v}^{(j)}$ disponibles à l'itération k, la relation (2.44) est équivalente à :

$$\mathbf{A}^{-1}\mathbf{V} - \mathbf{V}\underbrace{\begin{bmatrix} \alpha^{(0)} & \beta^{(0)} & & 0 \\ \beta^{(0)} & \ddots & \ddots & \\ & \ddots & \ddots & \beta^{(k-1)} \\ 0 & & \beta^{(k-1)} & \alpha^{(k)} \end{bmatrix}}_{\mathbf{T}} = \begin{bmatrix} 0 & \cdots & 0 & \beta^{(k)} & v^{(k+1)} \end{bmatrix} \tag{2.45}$$

ce qui implique :

$$\mathbf{V}^\top\mathbf{A}^{-1}\mathbf{V} - \mathbf{T} = \mathbf{0} \tag{2.46}$$

Par conséquent, $\mathbf{T}$ a les mêmes valeurs propres que $\mathbf{V}^\top\mathbf{A}^{-1}\mathbf{V}$ (voir le principe des sous-espaces). On cherche ensuite les caractéristiques propres de $\mathbf{T}$, $\mathbf{T}\hat{\mathbf{x}} = \hat{\lambda}\hat{\mathbf{x}}$, $\hat{\lambda}$ sont des approximations des valeurs propres de $\mathbf{A}^{-1}$ et $\mathbf{V}\hat{\mathbf{x}}$, celles des vecteurs propres associés.

Afin d'illustrer le comportement de cette méthode, la figure 2.4 présente l'estimation des valeurs propres aux cours des itérations de la méthode de Lanczos sur le problème test de Wilkinson (de taille 21) comportant plusieurs valeurs propres multiples et plusieurs valeurs propres très proches (les cercles sont les valeurs propres exactes). On voit que la multiplicité et/ou proximité des valeurs propres échappe à l'algorithme, et que les valeurs propres se mettent à converger quasiment l'une après l'autre.

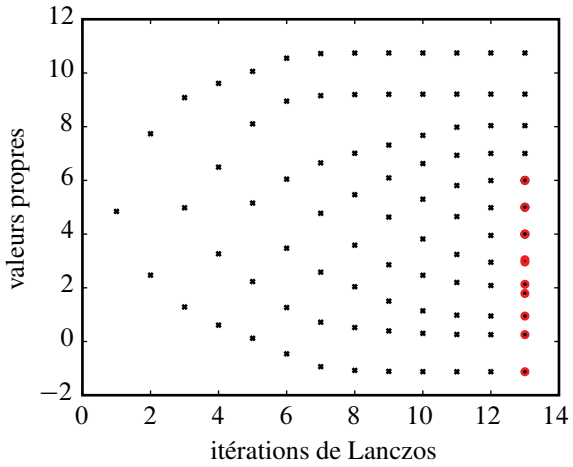


Figure 2.4 — Comportement de la méthode de Lanczos sur le test de Wilkinson

Critère d'arrêt

Classiquement, on borne la norme du résidu $\mathbf{r} = \mathbf{A}^{-1}\mathbf{V}\hat{\mathbf{x}} - \hat{\lambda}\mathbf{V}\hat{\mathbf{x}}$ avec :

$$\mathbf{A}^{-1}\mathbf{V}\hat{\mathbf{x}} = \mathbf{V}\mathbf{T}\hat{\mathbf{x}} + \begin{bmatrix} 0 & \cdots & 0 & \beta^{(k)} & v^{(k+1)} \end{bmatrix}\hat{\mathbf{x}} \tag{2.47}$$

avec $\mathbf{T}\hat{\mathbf{x}} = \hat{\lambda}\hat{\mathbf{x}}$, d'où $\mathbf{r} = \beta^{(k)}\mathbf{v}^{(k+1)}\hat{\mathbf{x}}_k$.

L'algorithme 2.6 décrit cette méthode, pour lequel les $\alpha^{(k)}$ et les $\beta^{(k)}$ sont les composantes de T.

3 Résolution de systèmes non-linéaires particuliers

Dans cette partie, des préoccupations directement issues du calcul de structure par éléments finis sont abordées. Il s'agit ici d'appliquer les outils précédents à certains problèmes particuliers, et non pas de disposer d'un cours entier consacré aux éléments finis. Il en sera de même dans la partie 4.4 pour les relations de comportement du matériau. Pour une présentation dédiée aux éléments finis, le lecteur est invité à se reporter, par exemple, aux nombreux cours de qualité disponibles sur le réseau. Par exemple, ceux de C. Felippa : *Nonlinear Finite Element Methods*, essentiellement pour les non-linéarités d'origine géométriques, et *Advanced Finite Element Methods*, pour la technologie des éléments finis, donnés au *Aerospace Engineering Sciences Department* de l'*University of Colorado at Boulder*.

3.1 Non-linéarités matérielles de type plasticité

Le problème à résoudre est celui du calcul de structures dont le matériau possède un comportement non linéaire. Il peut par exemple s'agir d'un matériau élastique non linéaire (sans dissipation, donc) ou d'un problème d'évolution comme la plasticité ou la visco-plasticité. Dans ce dernier cas, une approche incrémentale transforme le problème d'évolution en une série de problèmes non-linéaires stationnaires à chaque incrément de temps. Ceux-ci sont alors résolus par exemple par des méthodes classiques de type Newton [41], décrites ici. Pour plus d'informations, on pourra consulter [11].

3.1.1 Formulation du problème

Le problème de référence consiste à vérifier :

- les équations d'admissibilité cinématique : les déformations ε doivent être compatibles avec un champ de déplacement admissible U (régulier et satisfaisant les conditions aux limites en déplacement imposé)

$$\varepsilon = \varepsilon(\mathrm{U}) \tag{3.1}$$

- les équations d'admissibilité statique : le champ de contraintes σ doit être en équilibre avec les charges extérieures ;
- les relations de comportement [45]. On considère un matériau dont l'état est décrit par (outre la déformation ou la contrainte) un jeu de variables internes $\mathbb{X}$, avec leurs variables thermodynamiques associées $\mathbb{Y}$.
  La partition de la déformation en partie élastique et non élastique s'écrit :

$$\varepsilon = \varepsilon_e + \varepsilon_p \tag{3.2}$$

  Les lois d'état reliant les variables cinématiques et statiques sont associées à l'énergie libre (qui dépend de l'état) $\psi(\varepsilon_e, \mathbb{X})$:

$$\begin{aligned} \sigma &= \frac{\partial \psi}{\partial \varepsilon_e} \\ \mathbb{Y} &= \frac{\partial \psi}{\partial \mathbb{X}} \end{aligned} \tag{3.3}$$

L'évolution de l'état, pour un modèle standard associé, se déduit du potentiel de dissipation $\Phi(\sigma, \mathbb{Y})$ par la loi de normalité :

$$\begin{aligned} \dot{\varepsilon}_p &= \dot{\lambda}\frac{\partial \Phi}{\partial \sigma} \\ -\dot{\mathbb{X}} &= \dot{\lambda}\frac{\partial \Phi}{\partial \mathbb{Y}} \end{aligned} \tag{3.4}$$

avec, dans le cas non visqueux, $\Phi \leqslant 0$, $\dot{\lambda} \geqslant 0$ et $\Phi\dot{\lambda} = 0$. λ est le multiplicateur plastique.

3.1.2 Résolution pour un incrément de temps

Pour les problèmes qui nous préoccupent, à chaque incrément de temps, on est amené à résoudre, après une discrétisation en déplacement, un problème de la forme :

$$\mathbf{r}(\mathbf{u}) = \mathbf{K}(\mathbf{u})\mathbf{u} - \mathbf{f} = \mathbf{0} \tag{3.5}$$

$\mathbf{r}(\mathbf{u})$ est le *gradient* à annuler. La matrice de rigidité dépend de la solution $\mathbf{u}$, par l'intermédiaire de la relation de comportement du matériau. Les méthodes de type Newton sont habituellement utilisées : si $\mathbf{u}_0$ est une approximation de la solution, cette dernière est $\mathbf{u} = \mathbf{u}_0 + \delta\mathbf{u}$ et un développement de Taylor de $\mathbf{r}$ donne au voisinage de $\mathbf{u}$:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{u}}(\mathbf{u}_0)\delta\mathbf{u} = -\mathbf{r}(\mathbf{u}_0) \tag{3.6}$$

ce qui suggère un algorithme itératif de la forme :

$$\mathbf{u}_{i+1} - \mathbf{u}_i = -s_i \mathbf{H}_i \mathbf{r}_i \tag{3.7}$$

où $\mathbf{r}_i = \mathbf{r}(\mathbf{u}_i)$, $\mathbf{H}_i$ est une approximation de l'inverse de la matrice jacobienne $\mathbf{J}(\mathbf{u}_i) = \frac{\partial \mathbf{r}}{\partial \mathbf{u}}(\mathbf{u}_i)$. s_i est une amplitude destinée à réduire $\mathbf{r}_{i+1}$ [1].

En fait, les méthodes présentées ci-après sont issues d'une classe de problèmes plus vaste, celle de la minimisation de fonctionnelles non-linéaires $f(\mathrm{U})$; dans ce cadre, f est parfois appelée la fonction coût.

3.1.3 Méthode de la plus grande pente

Dans ce cas :

$$\mathbf{H} = \mathbf{I} \tag{3.8}$$

1. Dans un premier temps, on peut choisir $s_i = 1$ mais une procédure de type *line search* peut être utilisée comme une procédure itérative additionnelle pour améliorer sa valeur

3.1.4 Méthodes de gradient conjugué

Ces méthodes n'imposent pas le stockage des matrices $\mathbf{H}_i$ car seuls quelques vecteurs sont nécessaires au calcul de $\mathbf{H}_i\mathbf{r}_i$. Cela est réalisé avec une formule de récurrence sur une direction de recherche $\mathbf{d}$:

$$\mathbf{d}_i = \begin{cases} \mathbf{r}_i & \text{pour } i = 1 \\ \mathbf{r}_i + \zeta_i \mathbf{d}_{i-1} & \text{pour } i > 1 \end{cases} \tag{3.9}$$

et :

$$\mathbf{u}_{i+1} = \mathbf{u}_i - s_i \mathbf{d}_i \tag{3.10}$$

ζ_i est une scalaire utilisé pour conjuguer les directions de recherche, *i.e.* $\mathbf{d}_i^\top \mathbf{d}_{i-1} = 0$, pour lequel deux expressions classiques sont :

1. Fletcher-Reeves [21] :

$$\zeta_i = \frac{\mathbf{r}_i^\top \mathbf{r}_i}{\mathbf{r}_{i-1}^\top \mathbf{r}_{i-1}} \tag{3.11}$$

2. Polak-Ribière [57] :

$$\zeta_i = \frac{\mathbf{r}_i^\top (\mathbf{r}_i - \mathbf{r}_{i-1})}{\mathbf{r}_{i-1}^\top \mathbf{r}_{i-1}} \tag{3.12}$$

Si la fonctionnelle à minimiser est quadratique, donc si $\mathbf{K}$ ne dépend pas de $\mathbf{u}$, alors toutes les directions de recherche successives sont conjuguées entre elles.

Pour mémoire, il exite d'autres méthodes de grandient conjugué, comme la méthode du *shortest residual* [14].

3.1.5 Méthode de Newton-Raphson

Ici, $s_i = 1$ et :

$$\mathbf{H}_i = \mathbf{J}(\mathbf{u}_i)^{-1} = \left[\frac{\partial \mathbf{r}}{\partial \mathbf{u}}(\mathbf{u}_i)\right]^{-1} = \left[\mathbf{K}(\mathbf{u}_i) + \frac{\partial \mathbf{K}}{\partial \mathbf{u}}(\mathbf{u}_i)\right]^{-1} \tag{3.13}$$

à chaque itération, il faut évaluer la matrice jacobienne et résoudre le précédent système linéaire.

3.1.6 Méthode de Newton-Raphson modifiée

Elle utilise un Jacobien figé, évalué à une précédente estimation de la solution $\mathbf{u}^\star$:

$$\mathbf{H}_i = \mathbf{J}(\mathbf{u}^\star)^{-1} \tag{3.14}$$

3.1.7 Mises à jour de type Quasi-Newton

L'idée est de modifier simplement $\mathbf{H}_i$ à chaque itération plutôt que de le laisser fixé comme MNR, et plutôt que de le recalculer entièrement, comme pour NR.

$$\mathbf{H}_{i+1} = \mathbf{H}_i + \delta\mathbf{H}_i \tag{3.15}$$

en s'assurant que la condition sécante :

$$\mathbf{H}_{i+1}(\mathbf{u}_{i+1} - \mathbf{u}_i) = \mathbf{r}_{i+1} - \mathbf{r}_i \tag{3.16}$$

est satisfaite. Elle correspond à une approximation au premier ordre de la matrice jacobienne au voisinage de $\mathbf{u}_i$.

Si $\mathbf{H}_i$ est mis à jour, on parle de mise à jour inverse, alors qu'une mise à jour directe modifie $\mathbf{H}_i^{-1}$.

Le rang de la mise à jour est le rang de la matrice de correction qui est utilisée. Pour une mise à jour de rang 1, $\delta\mathbf{H} = \mathbf{a}\mathbf{b}^\top$ où $\mathbf{a}$ et $\mathbf{b}$ sont deux vecteurs, la formule de Sherman-Morrison-Woodbury [18, 30, 35] permet d'exprimer directement l'inverse, tant que $\mathbf{H}$ n'est pas singulier et que $1 + \mathbf{b}^\top\mathbf{H}^{-1}\mathbf{a} \neq 0$:

$$(\mathbf{H} + \mathbf{a}\mathbf{b}^\top)^{-1} = \mathbf{H}^{-1} - \frac{\mathbf{H}^{-1}\mathbf{a}\mathbf{b}^\top\mathbf{H}^{-1}}{1 + \mathbf{b}^\top\mathbf{H}^{-1}\mathbf{a}} \tag{3.17}$$

L'algorithme générique consiste à itérer sur :

— direction de recherche $\mathbf{d}_i = -\mathbf{H}_i\mathbf{r}_i$;

— $\min_{s_i>0} \|\mathbf{r}_{i+1}\|$ par exemple par une procédure de *line search* ;

— mise à jour de $\mathbf{H}_i$.

Le but est de construire une approximation de $\mathbf{J}^{-1}(\mathbf{u})$, qui s'améliore au cours des itérations, en utilisant l'information des gradients successifs. Par exemple, avec la précédente correction de rang 1, en vérifiant la symétrie et la condition sécante, on obtient :

$$\mathbf{H}_{i+1} = \mathbf{H}_i + \frac{(\boldsymbol{\delta}_i - \mathbf{H}_i\boldsymbol{\gamma}_i)(\boldsymbol{\delta}_i - \mathbf{H}_i\boldsymbol{\gamma}_i)^\top}{\boldsymbol{\gamma}_i^\top(\boldsymbol{\delta}_i - \mathbf{H}_i\boldsymbol{\gamma}_i)} \tag{3.18}$$

où $\boldsymbol{\delta}_i = \mathbf{u}_{i+1} - \mathbf{u}_i$ et $\boldsymbol{\gamma}_i = \mathbf{r}_{i+1} - \mathbf{r}_i$.

Une propriété intéressante pour l'efficacité est d'avoir $\mathbf{H}$ symétrique, au moins quand le problème à résoudre l'est, et définie positive pour avoir effectivement une direction de descente telle que $\mathbf{d}_i^\top\mathbf{r}_i < 0$. Avec la précédente mise à jour de rang 1, il n'est pas garanti d'avoir $\mathbf{H}_i$ défini positif.

Méthode de Davidson-Fletcher-Powell

Cette méthode de rang 2, proposée dans [20] préserve le caractère défini positif. En démarrant avec $\mathbf{H}_0$ symétrique, défini positif, la mise à jour est :

$$\mathbf{H}_{i+1}^{\text{DFP}} = \mathbf{H}_i + \frac{\boldsymbol{\delta}_i\boldsymbol{\delta}_i^\top}{\boldsymbol{\delta}_i^\top\boldsymbol{\delta}_i} - \frac{(\mathbf{H}_i\boldsymbol{\gamma}_i)(\mathbf{H}_i\boldsymbol{\gamma}_i)^\top}{\boldsymbol{\gamma}_i^\top\mathbf{H}_i\boldsymbol{\gamma}_i} \tag{3.19}$$

Dans le cas où la fonction coût est quadratique, les directions $\mathbf{d}_i$ sont $\mathbf{K}$-orthogonales entre elles ; après n itérations, $\mathbf{H}_n = \mathbf{J}^{-1}$. Si de plus $\mathbf{H}_0 = \mathbf{I}$, il s'agit de la méthode du gradient conjugué.

Méthode de Broyden-Fletcher-Goldfarb-Shanno

Cette méthode de rang 2, proposée dans [9], préserve aussi le caractère défini positif. En démarrant avec $\mathbf{H}_0$ symétrique, défini positif, la mise à jour est :

$$\begin{aligned}\mathbf{H}_{i+1}^{\text{BFGS}} &= \mathbf{H}_{i+1}^{\text{DFP}} + (\boldsymbol{\gamma}_i^\top \mathbf{H}_i \boldsymbol{\gamma}_i)\left(\frac{\boldsymbol{\delta}_i}{\boldsymbol{\delta}_i^\top \boldsymbol{\gamma}_i} - \frac{\mathbf{H}_i \boldsymbol{\gamma}_i}{\boldsymbol{\gamma}_i^\top \mathbf{H}_i \boldsymbol{\gamma}_i}\right)\left(\frac{\boldsymbol{\delta}_i}{\boldsymbol{\delta}_i^\top \boldsymbol{\gamma}_i} - \frac{\mathbf{H}_i \boldsymbol{\gamma}_i}{\boldsymbol{\gamma}_i^\top \mathbf{H}_i \boldsymbol{\gamma}_i}\right)^\top \\ &= \mathbf{H}_i + \left(1 + \frac{\boldsymbol{\gamma}_i^\top \mathbf{H}_i \boldsymbol{\gamma}_i}{\boldsymbol{\delta}_i^\top \boldsymbol{\gamma}_i}\right)\frac{\boldsymbol{\delta}_i \boldsymbol{\delta}_i^\top}{\boldsymbol{\delta}_i^\top \boldsymbol{\gamma}_i} - \frac{\boldsymbol{\delta}_i \boldsymbol{\gamma}_i^\top \mathbf{H}_i + \mathbf{H}_i \boldsymbol{\gamma}_i \boldsymbol{\delta}_i^\top}{\boldsymbol{\delta}_i^\top \boldsymbol{\gamma}_i}\end{aligned} \tag{3.20}$$

Cette méthode est réputée pour être l'une des plus robustes [50].

La famille de méthodes dite de Broyden est définie par :

$$\mathbf{H}_{i+1} = \alpha \mathbf{H}_{i+1}^{\text{BFGS}} + (1-\alpha)\mathbf{H}_{i+1}^{\text{DFP}} \tag{3.21}$$

α étant tel que le caractère défini positif est assuré ($0 \leqslant \alpha \leqslant 1$ le fait).

Versions avec mémoire limitée

Même si elles sont symétriques, les corrections présentées précédemment souffrent d'une grande demande en mémoire, pour les problèmes éléments finis de grande taille (les matrices $\mathbf{H}_i$ ne sont en effet pas creuses). Une solution consiste à ne stocker $\mathbf{H}_i$ uniquement de façon implicite, c'est-à-dire à être capable de construire $\mathbf{H}_i \mathbf{r}_i$ avec $\mathbf{H}_0$ (qui lui est creux) et des vecteurs successifs. De plus, le calcul de la direction de recherche est alors souvent moins coûteux que dans le cas de l'utilisation explicite d'un produit matrice-vecteur.

Par exemple, le gradient conjugué préconditionné (PCG) par $\mathbf{H}_0$, correspond à :

$$\mathbf{H}_{i+1} = \mathbf{V}_i \mathbf{H}_0 \tag{3.22}$$

avec :

$$\mathbf{V}_j = \mathbf{I} - \frac{\boldsymbol{\gamma}_j \boldsymbol{\delta}_j^\top}{\boldsymbol{\delta}_j^\top \boldsymbol{\gamma}_j} \tag{3.23}$$

et la méthode BFGS, à :

$$\mathbf{H}_{i+1} = \left(\prod_{j=0}^{i} \mathbf{V}_j\right)^\top \mathbf{H}_0 \left(\prod_{j=0}^{i} \mathbf{V}_j\right) + \sum_{j=0}^{i} \left(\prod_{k=j+1}^{i} \mathbf{V}_j\right)^\top \frac{\boldsymbol{\delta}_k \boldsymbol{\delta}_k^\top}{\boldsymbol{\delta}_k^\top \boldsymbol{\gamma}_k} \left(\prod_{k=j+1}^{i} \mathbf{V}_j\right) \tag{3.24}$$

Une troncature peut alors être effectuée pour limiter encore les ressources en mémoire, en éliminant les contributions des plus anciens termes. Elle correspond alors à la version *limited memory* de BFGS (L-BFGS) [42, 47].

4 Résolution de systèmes différentiels

Il est possible de consulter sur le sujet les références [36, 40]. Dans les applications qui nous intéressent, on est conduit à résoudre de tels systèmes, par exemple dans les cas suivants : intégration de relation de comportement, problème de thermique transitoire, problème de dynamique rapide...

4.1 Équation scalaire du premier ordre

4.1.1 Problème de Cauchy

Il s'agit de trouver la fonction $x(t)$, $t \in [0, \mathrm{T}]$, telle que :

$$\begin{aligned} &\frac{\mathrm{d}x}{\mathrm{d}t} = f(x,t) \\ &x(0) = u_0 \end{aligned} \tag{4.1}$$

Théorème de Cauchy-Lipschitz

Il y a existence et unicité de la solution si f est assez régulière [1] en x :

$$\exists k > 0,\ \forall t \in [0, \mathrm{T}],\ \forall (y,z) \in \mathbb{R}^2,\ |f(z,t) - f(y,t)| \leqslant k|z-y| \tag{4.2}$$

4.1.2 Ré-écriture du problème

Pour se ramener aux techniques d'intégration, il est plus intéressant d'écrire le problème sous forme intégrale :

$$x = u_0 + \int_0^t f(x,t)\,\mathrm{d}t \tag{4.3}$$

Une *discrétisation* de l'intervalle de temps $[0, \mathrm{T}]$ est la séquence d'instants $0 = t_0 < t_1 < \ldots < t_m = \mathrm{T}$. Le pas de temps est $h = t_{i+1} - t_i$. Le principe est de chercher à approcher la valeur de la fonction aux piquets de temps, $x(t_i)$, par la valeur x_i, ce qui permet de définir *l'erreur* $e_i = |x(t_i) - x_i|$. Dans la suite, on notera $f_i = f(x_i, t)$. Le problème incrémental est, ayant la solution approchée jusqu'à t_i, de la déterminer à t_{i+1} avec :

$$x = x_i + \int_{t_i}^t f(x,t)\,\mathrm{d}t \tag{4.4}$$

Exemple 4.1 — Schéma d'Euler explicite. La figure 4.3(a) illustre l'estimation de l'inté-

1. Plus précisément en termes mathématiques, f est continue et lipschitzienne.

grale cherchée. Le schéma s'écrit :

$$\begin{aligned} x_{i+1} &= x_i + hf_i \\ x_0 &= u_0 \end{aligned} \tag{4.5}$$

4.1.3 Schémas explicites à un pas

Ce sont les schémas de la forme :

$$\begin{aligned} x_{i+1} &= x_i + h\Phi(x_i, t_i, h) \\ x_0 &= u_0 \end{aligned} \tag{4.6}$$

Convergence Si on a la propriété $\sup_i |e_i| \to_{h\to 0} 0$ (en l'absence d'erreur d'arrondi !)

Stabilité On considère le schéma perturbé :

$$\begin{aligned} \tilde{x}_{i+1} &= \tilde{x}_i + h\Phi(\tilde{x}_i, t_i, h) + \varepsilon_{i+1} \\ \tilde{x}_0 &= u_0 + \varepsilon_0 \end{aligned} \tag{4.7}$$

Le schéma considéré est stable ssi :

$$\exists S > 0,\ \sup_i |\tilde{x}_i - x_i| \leqslant S \sum_{i=0}^{m} |\varepsilon_i| \tag{4.8}$$

Consistance Pour définir cette notion, on a besoin de l'erreur de consistance (*local truncation error*) :

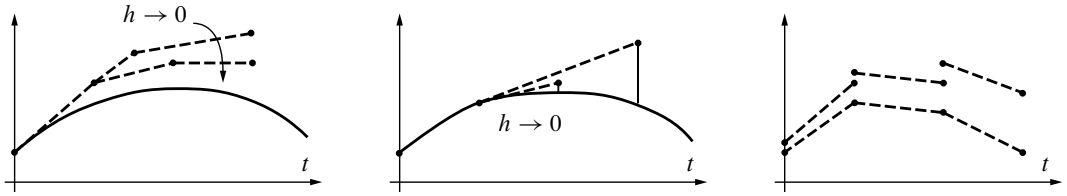
$$c_{i+1} = x(t_{i+1}) - \big(x(t_i) + h\Phi(x(t_i), t_i, h)\big) \tag{4.9}$$

Le schéma est consistant ssi :

$$\sum_i |c_i| \underset{h\to 0}{\to} 0 \tag{4.10}$$

Ordre On dit d'un schéma qu'il est d'ordre p quand $\exists M > 0$, $\sup_i |c_i| \leqslant Mh^{p+1}$.

Les trois premières notions précédentes sont illustrées sur la figure 4.1.



(a) convergence (b) consistance (c) stabilité

Figure 4.1 — Propriétés de base d'un schéma d'intégration

Propriétés

Il existe des liens entre les différentes caractéristiques précédentes, en particulier :

— stabilité et consistance $\Leftrightarrow$ convergence
— Φ lipschitzienne en $x \Rightarrow$ stabilité
— consistance $\Leftrightarrow \Phi(x,t,0) = f(x,t) \Leftrightarrow$ d'ordre $p \geqslant 1$
— d'ordre $p \Rightarrow |e_i| \leqslant \mathrm{N}h^p$

Exemple 4.2 — Schéma d'Euler explicite. Dans ce cas, $\Phi(x,t,h) = f(x,t)$ et par conséquent, le schéma est consistant et stable, donc convergent :

$$c_{i+1} = x(t_{i+1}) - \big(x(t_i) + hf(x(t_i),t_i)\big) = \mathcal{O}(h^2) \tag{4.11}$$

or $x(t_{i+1}) = x(t_i) + h\dot{x}(t_i) + \mathcal{O}(h^2)$ et $\dot{x}(t_i) = f(x(t_i),t_i)$ donc c'est un schéma d'ordre 1.

4.1.4 Quelques problèmes pratiques

Influence des erreurs d'arrondi

Perturbons le schéma :

$$\begin{aligned} \tilde{x}_{i+1} &= \tilde{x}_i + h\Phi(\tilde{x}_i,t_i,h) + \varepsilon_{i+1} \\ \tilde{x}_0 &= u_0 + \varepsilon_0 \end{aligned} \tag{4.12}$$

Si le schéma est convergent, $e_i = |x(t_i) - x_i| \underset{h\to 0}{\to} 0$. Qu'en est-il de $\tilde{e}_i = |x(t_i) - \tilde{x}_i|$?

$$\tilde{e}_i = |x(t_i) - x_i + x_i - \tilde{x}_i| \leqslant \underbrace{|x(t_i) - x_i|}_{①} + \underbrace{|x_i - \tilde{x}_i|}_{②} \tag{4.13}$$

avec, dans l'équation (4.13) :

— le schéma est d'ordre p donc le terme ① $\leqslant \mathrm{N}h^p$;
— le schéma est stable donc le terme ② $\leqslant \mathrm{S}\left[\sum_{i=0}^{m}|\varepsilon| + \sum_{i=1}^{m} h|\varepsilon|\right]$.

Au bilan :

$$\tilde{e}_i \leqslant \mathrm{S}[(m+1)|\varepsilon| + hm|\varepsilon|] + \mathrm{N}h^p \tag{4.14}$$

avec $hm = \mathrm{T}$, d'où :

$$\tilde{e}_i \leqslant \mathrm{S}|\varepsilon|\left(1 + \mathrm{T} + \frac{\mathrm{T}}{h}\right) + \mathrm{N}h^p \tag{4.15}$$

Il n'est donc pas forcément avantageux de réduire trop fortement le pas de temps pour la précision du schéma. En particulier, on peut dire que si le pas de temps n'est pas grand devant les erreurs numériques d'évaluation de la fonction Φ, la solution est entachée d'erreurs par cumul des erreurs numériques.

Problème numériquement mal posé

Considérons le problème suivant :

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= 4x - 1 \\ x(0) &= 1/4 \end{aligned} \tag{4.16}$$

dont la solution est $x(t) = 1/4$. Perturbons maintenant les conditions initiales, soit $x(0) = 1/4 + \varepsilon$. La nouvelle solution $x(t) = 1/4 + \varepsilon \mathrm{e}^{4t}$ est alors très différente de la précédente. Ce problème est numériquement mal posé. On risque d'avoir des difficultés à le résoudre car il est très sensible aux conditions initiales [2].

4.1.5 A-stabilité

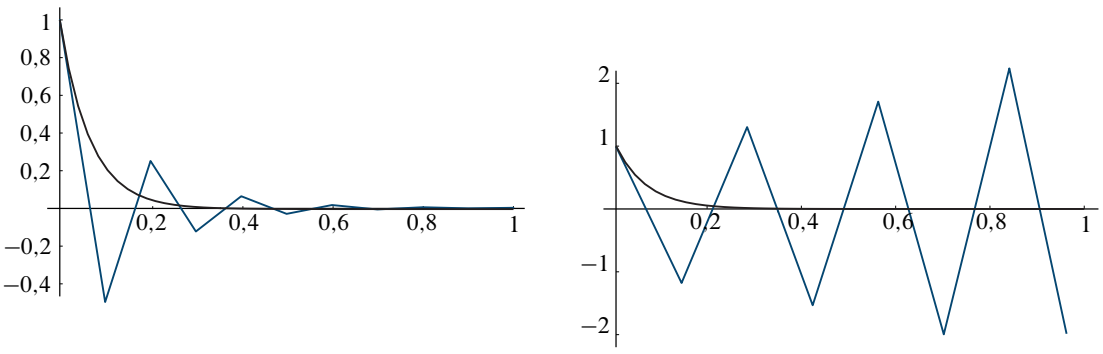
De façon pratique, la propriété de stabilité d'un schéma n'est pas suffisante. Prenons par exemple le schéma d'Euler explicite déjà décrit comme stable, et résolvons le problème :

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= -\omega^2 x \\ x(0) &= 1 \end{aligned} \tag{4.17}$$

dont la solution exacte est :

$$x(t) = \mathrm{e}^{-\omega^2 t} \tag{4.18}$$

La figure 4.2 présente les solutions obtenues numériquement en même temps que la solution exacte pour deux pas de temps. Ces solutions diffèrent fortement, et de plus en plus au cours du temps, si le pas de temps n'est pas assez petit. On parlera alors d'un schéma conditionnellement stable. Un schéma inconditionnellement stable (ou A-stable pour *absolute stability*) [13] se définit pour les équations linéaires. On teste alors le schéma sur un problème du type :



(a) $h = 1/10$ (b) $h = 1/7$

Figure 4.2 — Stabilité du schéma d'Euler explicite

$$\frac{\mathrm{d}x}{\mathrm{d}t} = cx \tag{4.19}$$

2. On peut d'ailleurs se poser des questions quant à la notion même de solution pour un tel problème.

où c est une constante complexe et pour lequel où le schéma est tel que $x_{i+1} = r(ch)x_i$, où r est un polynôme ou une fraction rationnelle. On définit alors le domaine de A-stabilité comme suit :

$$\mathrm{D}_{\text{stab}} = \{z \in \mathbb{C}, |r(z)| \leqslant 1\} \tag{4.20}$$

Un schéma est alors dit A-stable (ou inconditionnellement stable) si et seulement si :

$$\{z \in \mathbb{C}, \Re(z) \leqslant 0\} \subset \mathrm{D}_{\text{stab}} \tag{4.21}$$

Exemple 4.3 — Schéma d'Euler explicite. La relation est la suivante :

$$x_{i+1} = x_i + hcx_i = (1 + hc)x_i \;\Rightarrow\; r(z) = 1 + z \tag{4.22}$$

La condition de stabilité est donc :

$$|1 + hc| \leqslant 1 \tag{4.23}$$

Pour $c = -\omega^2 \in \mathbb{R}^-$, elle s'écrit $h \leqslant 2/\omega^2$. Il s'agit donc d'un schéma conditionnellement stable. Pour l'exemple précédent, avec $c = -15$ (la solution tend vers 0), le critère de stabilité donne $0 \leqslant h \leqslant 1/7{,}5$. Ceci est bien en accord avec les résultats de la figure 4.2. La condition de stabilité dépend bien entendu du schéma, mais aussi du problème. Son interprétation est que, pour une solution bornée, il faut que le schéma donne une approximation bornée.

4.1.6 Méthode des trapèzes généralisée

Cette méthode est décrite avec un paramètre $\alpha \in [0, 1]$ par :

$$\begin{aligned} x_{i+1} &= x_i + h\left[(1-\alpha) f_i + \alpha f_{i+1}\right] \\ x_0 &= u_0 \end{aligned} \tag{4.24}$$

Le calcul de x_{i+1} fait intervenir $\alpha f_{i+1} = \alpha f(x_{i+1}, t_{i+1})$. Il s'agit donc d'une méthode *implicite* si $\alpha \neq 0$.

Le cas $\alpha = 0$ conduit à la méthode d'Euler explicite ; dans le cas où $\alpha = 1$, il s'agit de la méthode d'Euler implicite ; enfin si $\alpha = 1/2$, c'est la méthode des trapèzes, appelée aussi méthode de Crank-Nicolson et qui date de 1947. La figure 4.3 illustre l'approximation de l'intégration réalisée par chacun de ces cas particuliers.

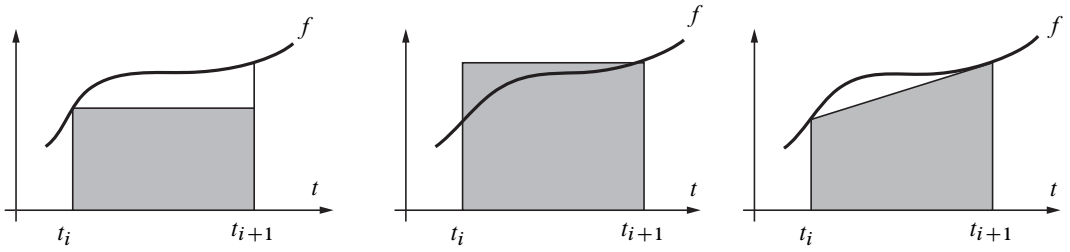
Stabilité

On considère le problème :

$$\frac{\mathrm{d}x}{\mathrm{d}t} = cx \tag{4.25}$$

le schéma donne :

$$x_{i+1} = x_i + h[(1-\alpha)cx_i + \alpha c x_{i+1}] \tag{4.26}$$



(a) $\alpha = 0$ (b) $\alpha = 1$ (c) $\alpha = 0{,}5$

Figure 4.3 — Différents types d'intégration

donc :

$$x_{i+1} = \underbrace{(1-\alpha hc)^{-1}(1+(1-\alpha)hc)}_{\mathrm{B}}\, x_i \tag{4.27}$$

Il faut avoir $\varrho(\mathrm{B}) < 1$, ce qui est le cas si toutes les valeurs propres de B sont simples, c'est-à-dire si la condition suivante est vérifiée :

$$\frac{|1+(1-\alpha)hc|}{|1-\alpha hc|} \leqslant 1 \tag{4.28}$$

— si $\alpha \geqslant 1/2$ le schéma est inconditionnellement stable ;
— si $\alpha < 1/2$ avec $c = -\omega^2$, la condition de stabilité s'écrit $h \leqslant 2/((1-2\alpha)\omega^2)$.

Les différents domaines de stabilité correspondants sont tracés sur la figure 4.4.

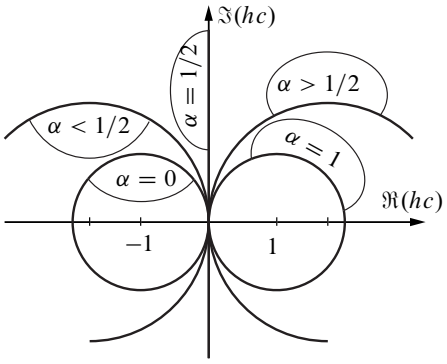


Figure 4.4 — Domaines de stabilité pour la méthode des trapèzes généralisés

Ordre

En utilisant la même méthode que celle déjà employée pour le schéma d'Euler explicite, on peut facilement montrer que le schéma des trapèzes généralisés est d'ordre 1 si $\alpha \neq 1/2$, et qu'il est d'ordre 2 pour $\alpha = 1/2$, c'est-à-dire pour Crank-Nicolson.

4.1.7 Méthode à pas multiples : Adams

L'idée des méthodes à pas multiple est d'utiliser non seulement l'évaluation f_i mais aussi un certain nombre de valeurs passées f_{i-1}...

À titre d'illustration, la méthode d'Adams, datant de 1883, utilisant un développement de Taylor de la solution est maintenant décrite.

Adams explicite

On utilise pour cela un développement de Taylor (avant) de $x(t_{i+1})$:

$$x(t_{i+1}) = x(t_i) + h\dot{x}(t_i) + \frac{h^2}{2!}\ddot{x}(t_i) + \dots \tag{4.29}$$

avec $\dot{x}(t_i) = f(x(t_i), t_i)$ et $\ddot{x}(t_i) = \frac{\mathrm{d}f}{\mathrm{d}t}(x(t_i), t_i)$. On garde ensuite les termes d'un ordre suffisant pour obtenir l'ordre voulu du schéma. Par exemple, pour Adams explicite d'ordre 1, on conserve les termes en $\mathcal{O}(h)$; on obtient $x_{i+1} = x_i + hf_i + \mathcal{O}(h^2)$, c'est Euler explicite !

Pour Adams explicite d'ordre 2, on conserve les termes en $\mathcal{O}(h^2)$ qui impliquent la dérivée première de f ; pour l'estimer on utilise encore le développement d'une fonction g :

$$g(t_{i+1}) = g(t_i) - h\dot{g}(t_i) + \mathcal{O}(h^2) \tag{4.30}$$

qui permet d'avoir l'expression de la dérivée $\dot{g}$:

$$\dot{g}(t_i) = \frac{1}{h}[g(t_i) - g(t_{i+1})] + \mathcal{O}(h) \tag{4.31}$$

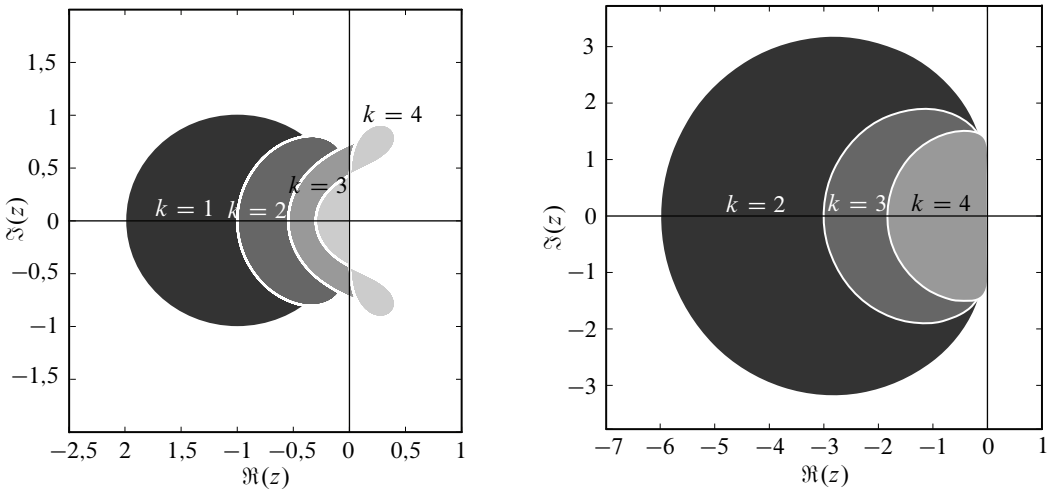
Avec ce procédé appliqué à $\frac{\mathrm{d}f}{\mathrm{d}t}$, on obtient la méthode cherchée :

$$x_{i+1} = x_i + \frac{h}{2}[3f_i - f_{i-1}] + \mathcal{O}(h^3) \tag{4.32}$$

La démarche se prolonge pour les ordres de schéma supérieurs. On obtient ainsi les formules d'Adams-Bashforth :

$$\begin{aligned}
x_i &= x_{i-1} + hf_{i-1} \\
x_i &= x_{i-1} + \frac{h}{2}(3f_{i-1} - f_{i-2}) \\
x_i &= x_{i-1} + \frac{h}{12}(23f_{i-1} - 16f_{i-2} + 5f_{i-3}) \\
x_i &= x_{i-1} + \frac{h}{24}(55f_{i-1} - 59f_{i-2} + 37f_{i-3} - 9f_{i-4}) \\
x_i &= x_{i-1} + \frac{h}{720}(1901f_{i-1} - 2774f_{i-2} + 2616f_{i-3} - 1274f_{i-4} + 251f_{i-5}) \\
x_i &= x_{i-1} + \frac{h}{1440}(4277f_{i-1} - 7923f_{i-2} + 9982f_{i-3} - 7298f_{i-4} + 2877f_{i-5} - 475f_{i-6})
\end{aligned} \tag{4.33}$$

On ne peut cependant pas augmenter l'ordre du schéma impunément : l'inconvénient réside dans un rétrécissement du domaine de stabilité. Celui-ci est illustré sur la figure 4.5.



(a) Adams explicite : Adams-Bashforth

(b) Adams implicite : Adams-Moulton

Figure 4.5 — Domaine de stabilité dans le plan complexe

Adams implicite

L'idée est la même que précédemment, en utilisant un développement de Taylor arrière de :

$$x(t_i) = x(t_{i+1} - h) = x(t_{i+1}) - h\dot{x}(t_{i+1}) + \frac{h^2}{2!}\ddot{x}(t_{i+1}) + \dots \tag{4.34}$$

La méthode d'Adams implicite d'ordre 1 est ainsi :

$$x_i = x_{i+1} - hf_{i+1} + \mathcal{O}(h^2) \tag{4.35}$$

d'où :

$$x_{i+1} = x_i + hf_{i+1} + \mathcal{O}(h^2) \tag{4.36}$$

c'est Euler implicite !

L'utilisation de la même technique que pour Adams explicite permet de construire les schémas implicites d'ordres plus élevés, et on obtient alors les formules d'Adams-Moulton :

$$\begin{aligned}
x_i &= x_{i-1} + hf_i \\
x_i &= x_{i-1} + \frac{h}{2}(f_i + f_{i-1}) \\
x_i &= x_{i-1} + \frac{h}{12}(f_i + 8f_{i-1} - f_{i-2}) \\
x_i &= x_{i-1} + \frac{h}{24}(9f_i + 19f_{i-1} - 5f_{i-2} + f_{i-3}) \\
x_i &= x_{i-1} + \frac{h}{720}(251f_i + 646f_{i-1} - 264f_{i-2} + 106f_{i-3} - 19f_{i-4}) \\
x_i &= x_{i-1} + \frac{h}{1440}(475f_i + 1427f_{i-1} - 798f_{i-2} + 482f_{i-3} - 173f_{i-4} + 27f_{i-5})
\end{aligned} \tag{4.37}$$

Le fait d'utiliser des schémas implicites augmente la stabilité, mais la même tendance est observée lorsque l'ordre croît, voir figure 4.5.

Amorçage du schéma

Aux premiers pas de temps, on ne dispose pas forcément de tous les f_{i-q}. Dans ce cas, on peut utiliser une méthode à 1 pas pour démarrer l'intégration jusqu'à avoir suffisamment progressé pour pouvoir appliquer la méthode à pas multiples voulue.

4.1.8 Méthode de prédiction-correction basée sur Adams

Le problème des méthodes implicites réside dans le fait, qu'en non-linéaire au moins, on ne peut exprimer explicitement la valeur au pas courant. On est alors obligé de mettre en œuvre une méthode itérative, généralement de type point fixe.

On part ainsi de la version implicite ; par exemple Adams d'ordre 2 : connaissant $x_{i+1}^{(0)}$, on évalue $f(x_{i+1}^{(0)}, t_{i+1})$ qui permet, en utilisant la formule d'intégration d'Adams implicite, d'atteindre une valeur $x_{i+1}^{(1)}$. Cette dernière étape est appelée *correction*. On itère ainsi jusqu'à une convergence souhaitée. Pour avoir la première valeur $x_{i+1}^{(0)}$, la *prédiction*, on utilise la version implicite de la méthode, par exemple ici, Adams explicite d'ordre 2. En pratique, une à trois itérations suffisent.

On peut aussi apporter une amélioration à faible surcoût avec une correction du prédicteur avec l'erreur de troncature. Pour Adams d'ordre 2, on procède comme suit :

1. prédicteur (AE2) :

$$x_{i+1}^{(0)} \leftarrow x_i + \frac{h}{2}(3f_i - f_{i-1}) \tag{4.38}$$

2. modificateur :

$$\hat{x}_{i+1}^{(0)} \leftarrow x_{i+1}^{(0)} - \frac{1}{12}\left(x_i - x_i^{(0)}\right) \tag{4.39}$$

3. correcteur (AI2) :

$$x_{i+1} \leftarrow x_i + \frac{h}{2}\left(\hat{f}_{i+1} + f_{i-1}\right) \tag{4.40}$$

où $\hat{f}_{i+1} = f(\hat{x}_{i+1}^{(0)}, t_{i+1})$. Le coefficient $\frac{1}{12}$ est celui issu de la formule de (AI3) ? ? ?.

4.1.9 Méthode de Runge-Kutta

Historiquement présentée dans [43] en 1901, cette méthode est de la forme :

$$\begin{aligned} x_{i+1} &= x_i + h\Phi(x,t,h) \\ x_0 &= u_0 \end{aligned} \tag{4.41}$$

avec :

$$\begin{aligned} \Phi(x,t,h) &= \sum_{l=1}^{q} a_l k_l(x,t,h) \\ k_1(x,t,h) &= f(x,t) \\ k_l(x,t,h) &= f\left(x + \sum_{j=1}^{l-1} \beta_j^{(l)} k_j(x,t,h), t + \alpha_l h\right) \end{aligned} \tag{4.42}$$

L'idée est d'intégrer en utilisant des valeurs en des points particuliers, avec des poids d'intégration particuliers. Les constantes a_l, α_l et $\beta_j^{(l)}$ sont ajustées pour obtenir l'ordre souhaité.

Exemple 4.4 — Schéma à l'ordre 2. On prend $\Phi = af(x,t) + bf(t+\alpha h, x+\beta h f(x,t))$ et on détermine a, b, α, β pour avoir :

$$\underbrace{\frac{1}{h}\big(x(t+h)-x(t)\big)}_{①} - \underbrace{\Phi\big(x(t),t,h\big)}_{②} = \mathcal{O}(h^2) \tag{4.43}$$

avec, dans l'équation (4.43) :

$$\begin{aligned} ① &= f + \frac{h}{2}\Big(\frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} f\Big) + \mathcal{O}(h^2) \\ ② &= af + b\Big(f + \alpha h \frac{\partial f}{\partial t} + \beta h f \frac{\partial f}{\partial x} + \mathcal{O}(h^2)\Big) \end{aligned}$$

Il faut donc avoir :

$$0 = 1 - a - b \qquad 0 = \frac{h}{2} - b\alpha h \qquad 0 = \frac{h}{2} - b\beta h \tag{4.44}$$

d'où les coefficients cherchés $a = 1-b, \alpha = \beta = \frac{1}{2b}$ et le schéma :

$$\Phi(x,t,h) = (1-b)f(x,t) + bf\Big(x + \frac{h}{2b} f(x,t), t + \frac{h}{2b}\Big) \tag{4.45}$$

où b est une constante arbitraire non nulle. Des cas particuliers existent :
- $b = 1$: il s'agit de la méthode d'Euler modifiée, d'ordre 2 ;
- $b = \frac{1}{2}$: il s'agit de la méthode de Heun, d'ordre 2.

Exemple 4.5 — Schéma à l'ordre 4. Bien entendu, cette technique se poursuit pour les ordres supérieurs. Un ordre souvent utilisé avec cette méthode est l'ordre 4. Le schéma se présente alors comme suit :

$$\begin{aligned} \Phi(x,t,h) &= \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 &= f(x,t) \\ k_2 &= f\Big(x + \frac{h}{2}k_1, t + \frac{h}{2}\Big) \\ k_3 &= f\Big(x + \frac{h}{2}k_2, t + \frac{h}{2}\Big) \\ k_4 &= f(x + hk_3, t + h) \end{aligned} \tag{4.46}$$

Bien entendu, des méthodes de prédiction-correction basées sur la méthode de Runge-Kutta peuvent aussi être mises en place.

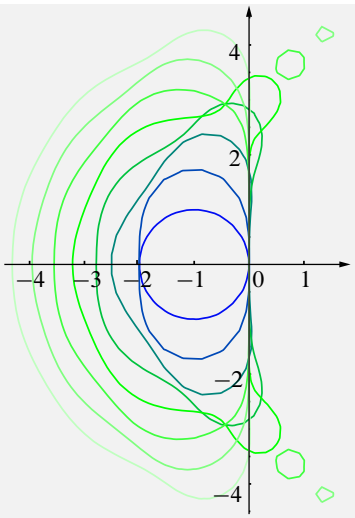


Figure 4.6 — Domaine de stabilité pour Runge-Kutta (ordre 1 à 8)

Cette méthode requière beaucoup d'évaluations de f, ce qui peut la pénaliser au niveau du coût. Concernant les domaines de stabilité, on peut observer la même tendance que pour les méthodes d'Adams, voir figure 4.6.

4.1.10 Autres méthodes

Citons pour mémoire les méthodes de Hanning, de Milne-Simpson, les méthodes de tir (*shooting*), les méthodes de collocation optimale, et une méthode bien adaptée à l'intégration des relations de comportements décrits par variables internes à 1 seul seuil, la méthode de retour radial [53].

Il existe aussi une famille de méthodes appelée Galerkin discontinu en temps, plus inspirée des formulations variationnelles en espace. Pour l'illustrer, considérons à nouveau le problème de Cauchy (4.1). Il peut être réécrit en utilisant une fonction test $x^\star$ à choisir dans un espace adéquat :

$$\int_0^{\mathrm{T}} \Big(\frac{\mathrm{d}x}{\mathrm{d}t} - f(x,t)\Big) x^\star \,\mathrm{d}t = 0 \tag{4.47}$$

L'idée est de pouvoir utiliser des discrétisations en temps qui tolèrent *a priori* des discontinuités aux pas de temps : à $t = t_i$, on peut définir une valeur à gauche x_i^- et une valeur à droite x_i^+ différentes. Le saut est noté $[\![x]\!]_i = x_i^+ - x_i^-$. L'intégrale précédente est donc à prendre en un sens généralisé, qui fait « travailler » les sauts [3] :

$$\int_0^{\mathrm{T}} \Big(\frac{\mathrm{d}x}{\mathrm{d}t} - f(x,t)\Big) x^\star \,\mathrm{d}t = \sum_i \left[\int_{t_i}^{t_{i+1}} \Big(\frac{\mathrm{d}x}{\mathrm{d}t} - f(x,t)\Big) x^\star \,\mathrm{d}t + [\![x]\!]_i x_i^\star\right] = 0 \tag{4.48}$$

La fonction test est choisie dans le même espace que x, et on définit sa valeur en t_i comme $x_i^\star = \alpha x_i^{\star+} + (1-\alpha) x_i^{\star-}$. Dans le cas standard, on prend $\alpha = 1$ [4]. Le problème peut alors

3. On suppose cependant $f(x,t)$ suffisamment régulière.
4. *one-sided upwinding*

s'écrire sur chaque intervalle de temps :

$$\int_{t_i}^{t_{i+1}} \Big(\frac{\mathrm{d}x}{\mathrm{d}t} - f(x,t) \Big) x^\star \, \mathrm{d}t + [\![x]\!]_i x_i^\star = 0 \tag{4.49}$$

Cette méthode est très ressemblante à une méthode de Runge-Kutta à un pas implicite (A-stable) d'ordre $2l+1$ si on prend les espaces polynômiaux de degré inférieur ou égal à l par incrément de temps (Lesaint et Raviart 1974). Il faut aussi prévoir une méthode numérique de calcul d'intégrale en temps qui intègre exactement des polynômes de degré $2l$ (Cockburn et Shu 1989).

4.2 Systèmes différentiels du premier ordre

Cette situation est rencontrée en éléments finis, lors de l'intégration de relations d'un comportement décrit par variables internes et avec un système de taille bien plus grande, lors de calculs de diffusion, par exemple, un calcul de conduction thermique transitoire, qui amène à résoudre un problème de la forme :

$$\begin{aligned} &\mathbf{M}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{g}(t) \\ &\mathbf{x}(0) = \mathbf{u_0} \end{aligned} \tag{4.50}$$

où $\mathbf{K}$, généralement symétrique positive, est la matrice de conduction, et $\mathbf{M}$, généralement symétrique, définie positive, est la matrice de capacité. $\mathbf{x}$ est le vecteur des températures inconnues aux nœuds de la discrétisation.

4.2.1 Emploi des méthodes précédentes

Les méthodes précédentes se généralisent sans grande difficulté à ce cas de figure.

Exemple 4.6 — Schéma d'Euler explicite. Le problème précédent peut s'écrire :

$$\dot{\mathbf{x}} = -\mathbf{M}^{-1}\mathbf{K}\mathbf{x} + \mathbf{M}^{-1}\mathbf{g} \tag{4.51}$$

On applique alors le schéma voulu :

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h(-\mathbf{M}^{-1}\mathbf{K}\mathbf{x}_i + \mathbf{M}^{-1}\mathbf{g}_i) \tag{4.52}$$

Pour éviter de faire intervenir directement $\mathbf{M}^{-1}$, on préfère le mettre sous la forme :

$$\begin{aligned} &\mathbf{M}\mathbf{x}_{i+1} = \mathbf{M}\mathbf{x}_i + h(\mathbf{K}\mathbf{x}_i + \mathbf{g}_i) \\ &\mathbf{x}_0 = \mathbf{u}_0 \end{aligned} \tag{4.53}$$

Stabilité On emploie la même démarche que dans le cas des équations scalaires, dans la base propre (pour les équations linéaires), avec la valeur propre la plus contraignante, c'est-à-dire la pulsation ω la plus grande. Dans le cas d'Euler explicite, on a donc le critère de stabilité $h \leqslant 2/\omega_{\mathrm{M}}^2$ et on peut borner la pulsation propre maxi du système par la pulsation propre maxi élémentaire $\omega_{\mathrm{M}} \leqslant \omega_{\max}^e$, voir annexe B.

Pour avoir un vrai schéma explicite, il faut ne pas avoir à résoudre de système linéaire couplé dans (4.53). Il faut donc avoir une matrice de masse diagonale. Une modification du problème (une approximation supplémentaire dans la modélisation) consiste à remplacer la matrice de masse par une matrice de masse « lumpée » diagonale. Pour cela, plusieurs techniques existent, comme une matrice proportionnelle à la masse diagonale, avec respect de la masse totale par élément, ou même la somme des lignes de la matrice de départ. Outre une approximation supplémentaire [10], on peut se poser la question de la conservation des propriétés du schéma.

4.2.2 Influence de l'ordre d'intégration

Si on considère que le coût est directement lié au nombre d'évaluations de la fonction f, on peut tracer le coût en fonction de l'erreur obtenue pour différents schémas. Choisissons le cas test suivant :

$$\begin{pmatrix}\dot{x}\\ \dot{y}\end{pmatrix} = \begin{pmatrix} y \\ (1-x^2)y - x\end{pmatrix} \quad \text{avec} \quad \begin{pmatrix}x(0)\\ y(0)\end{pmatrix} = \begin{pmatrix}u_0\\ 0\end{pmatrix} \tag{4.54}$$

Il s'agit d'un système différentiel du premier ordre, non linéaire, pour $t \in [0, \mathrm{T}]$. Si $\alpha \approx 2{,}008619861986087484313650940188$ et avec $\mathrm{T} \approx 6{,}663286859323130189699682030 5$, la solution est périodique, de période T. Une mesure de l'erreur peut donc être de prendre à

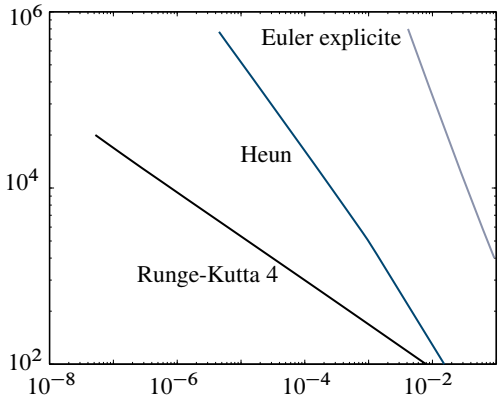


Figure 4.7 — Influence de l'ordre d'un schéma sur le coût

la fin de l'intervalle d'étude :

$$e = \| \begin{pmatrix}x_\mathrm{T}\\ y_\mathrm{T}\end{pmatrix} - \begin{pmatrix}x_0\\ y_0\end{pmatrix} \| \tag{4.55}$$

La figure 4.7 reporte en échelles logarithmiques le nombre d'évaluations de f en fonction du niveau d'erreur e pour plusieurs schémas d'ordres différents : Euler explicite d'ordre 1, Heun d'ordre 2 et Runge-Kutta d'ordre 4. On voit donc l'intérêt de mettre en œuvre des schémas d'ordre élevé, en gardant une stabilité raisonnable.

4.3 Systèmes différentiels du deuxième ordre

On trouve ce genre de problème par exemple en dynamique transitoire des structures :

$$\begin{aligned} &\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{g}(t) \\ &\dot{\mathbf{x}}(0) = \mathbf{v}_0 \\ &\mathbf{x}(0) = \mathbf{u}_0 \end{aligned} \tag{4.56}$$

$\mathbf{M}$ est la matrice de masse, $\mathbf{C}$ est la matrice d'amortissement et $\mathbf{K}$ la matrice de rigidité.

De façon générale, l'ordre d'un système linéaire peut être abaissé, ici à l'ordre 1, mais en doublant la taille du système. Ici, on s'intéressera aux traitement direct du système au second ordre, dans le cas où on n'a pas d'amortissement, c'est-à-dire où $\mathbf{C} = \mathbf{0}$. L'équilibre est donc :

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{g}(t) \tag{4.57}$$

4.3.1 Méthode de Newmark

Il s'agit d'une famille de méthodes implicites à un pas [51] et définie en 1959. Elle se présente sous la forme suivante :

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i + h\dot{\mathbf{x}}_i + \frac{1}{2}h^2\big((1-2\beta)\ddot{\mathbf{x}}_i + 2\beta\ddot{\mathbf{x}}_{i+1}\big) \\ \dot{\mathbf{x}}_{i+1} &= \dot{\mathbf{x}}_i + h\big((1-\gamma)\ddot{\mathbf{x}}_i + \gamma\ddot{\mathbf{x}}_{i+1}\big) \end{aligned} \tag{4.58}$$

On retrouve bien le caractère implicite par l'apparition de quantités indicées $i+1$ au second membre. On peut aussi remarquer que si $\beta = \gamma = 0$, on retrouve des développements de Taylor puisque le schéma de Newmark ressemble à un développement au troisième ordre :

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i + h\dot{\mathbf{x}}_i + \frac{h^2}{2}\ddot{\mathbf{x}}_i + \frac{h^3}{6}6\beta\frac{\ddot{\mathbf{x}}_{i+1} - \ddot{\mathbf{x}}_i}{h} \\ \dot{\mathbf{x}}_{i+1} &= \dot{\mathbf{x}}_i + h\ddot{\mathbf{x}}_i + \frac{h^2}{2}2\gamma\frac{\ddot{\mathbf{x}}_{i+1} - \ddot{\mathbf{x}}_i}{h} \end{aligned} \tag{4.59}$$

Il reste à écrire l'équilibre (4.57) à l'instant t_{i+1} avec le schéma précédent. On peut alors prendre l'accélération comme inconnue :

$$\big(\beta h^2\mathbf{K} + \mathbf{M}\big)\ddot{\mathbf{x}}_{i+1} = \mathbf{g}_{i+1} - \mathbf{K}\Big(\mathbf{x}_i + h\dot{\mathbf{x}}_i + h^2\Big(\frac{1}{2} - \beta\Big)\ddot{\mathbf{x}}_i\Big) \tag{4.60}$$

et on obtient le système à résoudre pour atteindre $\ddot{\mathbf{x}}_{i+1}$; mais on peut aussi l'écrire avec le déplacement comme inconnue :

$$\big(\beta h^2\mathbf{K} + \mathbf{M}\big)\mathbf{x}_{i+1} = \beta h^2\mathbf{g}_{i+1} + \mathbf{M}\Big(\mathbf{x}_i + h\dot{\mathbf{x}}_i + h^2\Big(\frac{1}{2} - \beta\Big)\ddot{\mathbf{x}}_i\Big) \tag{4.61}$$

qui est le système à résoudre pour atteindre $\mathbf{x}_{i+1}$. Dans ce dernier cas, cependant, un problème subsiste si $\beta = 0$. En effet, soit on utilise l'équilibre pour revenir à l'accélération :

$$\ddot{\mathbf{x}}_{i+1} = \mathbf{M}^{-1}(\mathbf{g}_{i+1} - \mathbf{K}\mathbf{x}_{i+1}) \tag{4.62}$$

ce qui peut être coûteux ; soit on utilise la relation inverse de (4.58) :

$$\ddot{\mathbf{x}}_{i+1} = \frac{1}{\beta h^2}(\mathbf{x}_{i+1} - \mathbf{x}_i) - \frac{1}{\beta h}\dot{\mathbf{x}}_i - \frac{1-2\beta}{2\beta}\ddot{\mathbf{x}}_i \tag{4.63}$$

mais on s'aperçoit bien du problème quand $\beta = 0$.

Cas particuliers

— $\gamma = \frac{1}{2}, \beta = \frac{1}{4}$: méthode des accélérations moyennes ;
— $\gamma = \frac{1}{2}, \beta = \frac{1}{6}$: méthode des accélérations linéaires ;
— $\gamma = \frac{1}{2}, \beta = 0$: méthode des différences centrées, qui peut être explicite [5] ;
— $\gamma = \frac{1}{2}, \beta = \frac{1}{12}$: méthode de Fox-Goodwin.

Ordres

On peut montrer qu'il s'agit d'une méthode d'ordre 1 pour $\gamma \neq \frac{1}{2}$ et d'ordre 2 pour $\gamma = \frac{1}{2}$.

Stabilité

Dans le cas d'une seule équation scalaire, $\ddot{x} = -\omega^2 x$, le schéma donne :

$$\underbrace{\begin{bmatrix} 1+(h\omega)^2\beta & 0 \\ h\omega^2\alpha & 1 \end{bmatrix}}_{\mathbf{A}_1}\underbrace{\begin{pmatrix} x_{i+1} \\ \dot{x}_{i+1} \end{pmatrix}}_{\mathbf{y}_{i+1}} = \underbrace{\begin{bmatrix} 1-(h\omega)^2(\frac{1}{2}-\beta) & h \\ -h\omega^2(1-\alpha) & 1 \end{bmatrix}}_{\mathbf{A}_2}\underbrace{\begin{pmatrix} x_i \\ \dot{x}_i \end{pmatrix}}_{\mathbf{y}_i} \tag{4.64}$$

d'où la relation $\mathbf{y}_{i+1} = \mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{y}_i$ et la matrice d'itération $\mathbf{B} = \mathbf{A}_1^{-1}\mathbf{A}_2$. La condition de stabilité est donc $\varrho(\mathbf{B}) < 1$ ou bien $\varrho(\mathbf{B}) \leqslant 1$ avec des valeurs propres simples. L'équation caractéristique permettant de déterminer les valeurs propres de $\mathbf{B}$ est $\lambda^2 - 2b\lambda + c = 0$ avec :

$$2b = 2 - \left(\alpha + \frac{1}{2}\right)\frac{(h\omega)^2}{1+\beta(h\omega)^2} \qquad c = 1 - \left(\alpha - \frac{1}{2}\right)\frac{(h\omega)^2}{1+\beta(h\omega)^2} \tag{4.65}$$

les conditions de stabilité sont donc :

$$c \leqslant 1 \qquad 1 - 2b + c \geqslant 0 \qquad 1 + 2b + c \geqslant 0 \tag{4.66}$$

sauf en ce qui concerne les points isolés $(b, c) = (-1, 1)$ et $(1, 1)$. D'où les conditions en terme de coefficients de la méthode :

$$\gamma \geqslant \frac{1}{2} \qquad 2 + \frac{h\omega(2\beta-\gamma)}{1+\beta(h\omega)^2} \geqslant 0 \tag{4.67}$$

En conclusion, si $2\beta \geqslant \gamma \geqslant \frac{1}{2}$, le schéma est inconditionnellement stable (IS) ; si $\gamma \geqslant \frac{1}{2}$ et $\gamma > 2\beta$, le schéma est conditionnellement stable (CS) et la condition de stabilité est :

$$h \leqslant \frac{1}{\omega\sqrt{\gamma/2-\beta}} \tag{4.68}$$

La figure 4.8 illustre ces résultats.

5. Cette méthode est aussi appelée *Velocity Verlet algorithm* ou *Leap Frog* en dynamique moléculaire !

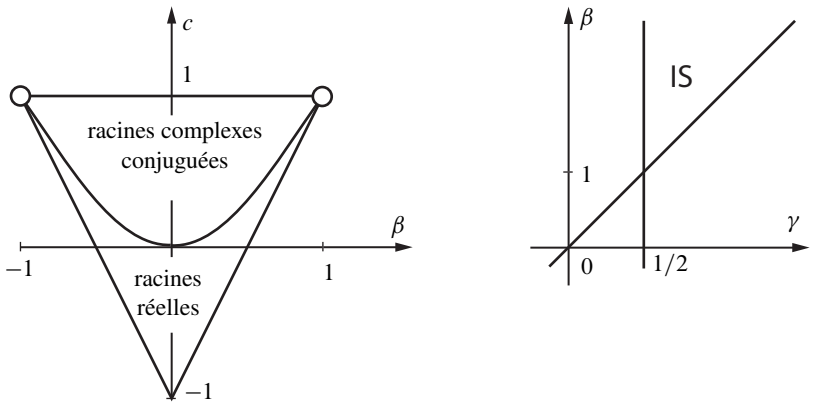


Figure 4.8 — Stabilité des schémas de Newmark

Influence de l'amortissement et comportement hautes fréquences

Considérons le cas où l'amortissement n'est pas négligé dans l'équation scalaire [6] :

$$\ddot{x} + 2\zeta\omega\dot{x} + \omega^2 x = 0 \tag{4.69}$$

On peut montrer dans ce cas que la condition de stabilité s'écrit :

$$\bar{h} \leqslant \frac{\zeta\left(\gamma - \frac{1}{2}\right) + \left(\frac{1}{2}\gamma - \beta + \zeta^2(\gamma - \frac{1}{2})^2\right)^{1/2}}{\frac{1}{2}\gamma - \beta}\frac{1}{\omega} \tag{4.70}$$

Dans le cas où $\gamma > \frac{1}{2}$, l'amortissement a un effet stabilisant puisque sur les conditions de stabilité, on a $\bar{h} < h$.

Concernant les modes à haute fréquence, ils ne sont pas représentatifs du modèle continu mais sont néanmoins sollicités lors de l'intégration et conduisent à la présence de parasites non désirés. On utilise classiquement une *dissipation* numérique pour amortir ces hautes fréquences.

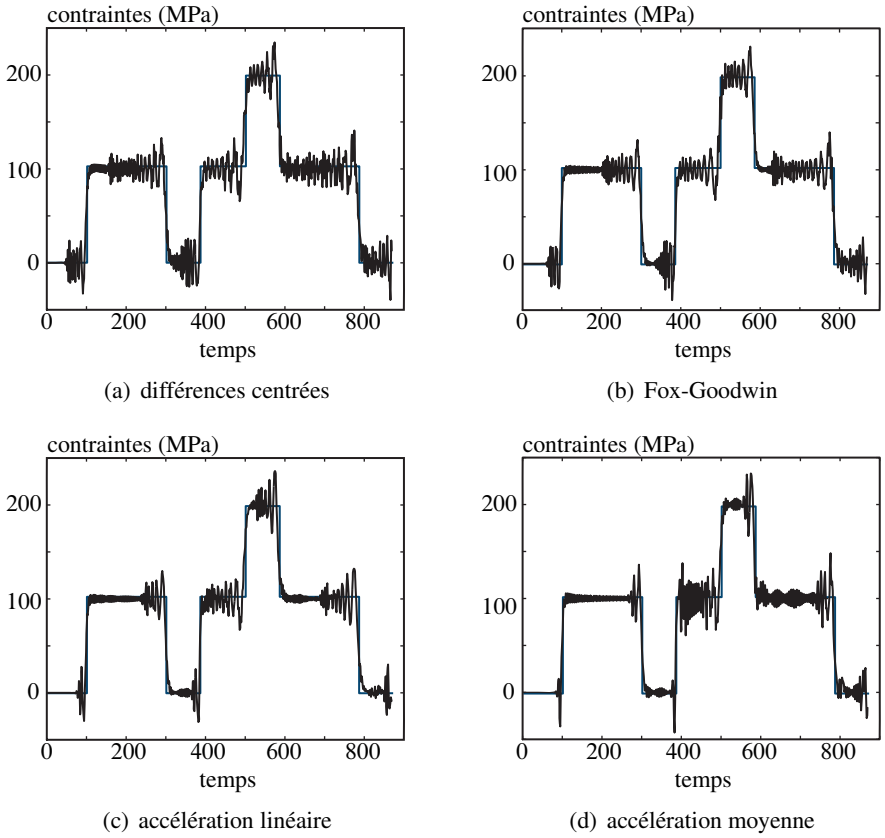
En choisissant $\gamma > \frac{1}{2}$, on prend β pour maximiser l'amortissement (région des racines complexes conjuguées sur la figure 4.8), ce qui conduit à :

$$\beta \leqslant \frac{1}{4}\left(\gamma + \frac{1}{2}\right)^2 \tag{4.71}$$

Le point de dissipation Haute Fréquence maximal est déterminé par le couple ($\beta = 1$, $\gamma = \frac{3}{2}$), mais il présente le gros inconvénient d'être peu précis en Basse Fréquence. Un choix généralement recommandé pour les méthodes de Newmark est alors :

$$\gamma > \frac{1}{2} \qquad \beta = \frac{1}{4}\left(\gamma + \frac{1}{2}\right)^2 \tag{4.72}$$

La figure 4.9 présente l'utilisation des schémas de Newmark sans amortissement pour le cas de réflexions d'ondes dans une poutre unidimensionnelle, ce qui met en évidence la présence de perturbations à hautes fréquences. La figure 4.10 présente quant à elle l'influence de l'amortissement numérique pour ces mêmes schémas.



(a) différences centrées (b) Fox-Goodwin

(c) accélération linéaire (d) accélération moyenne

Figure 4.9 — Parasites hautes fréquences (d'après [46])

Toujours pour illustrer l'amortissement numérique, considérons un bilan d'énergie entre les instants t_i et t_{i+1} [37]. L'écriture du schéma de Newmark (4.58) en différence entre ces deux instants, en utilisant la notation suivante :

$$\begin{aligned} \langle \mathbf{x}_i \rangle &= \frac{1}{2}(\mathbf{x}_{i+1} + \mathbf{x}_i) \\ [\mathbf{x}_i] &= \mathbf{x}_{i+1} - \mathbf{x}_i \end{aligned} \tag{4.73}$$

conduit à :

$$\begin{aligned} [\mathbf{x}_i] &= h\,\langle \dot{\mathbf{x}}_i \rangle + \frac{h^2}{2}(2\beta - \gamma)[\ddot{\mathbf{x}}_i] \\ [\dot{\mathbf{x}}_i] &= h\,\langle \ddot{\mathbf{x}}_i \rangle + h\Big(\gamma - \frac{1}{2}\Big)[\ddot{\mathbf{x}}_i] \end{aligned} \tag{4.74}$$

De même, la différence des deux équilibres (4.57) devient :

$$\mathbf{M}[\ddot{\mathbf{x}}_i] + \mathbf{K}[\dot{\mathbf{x}}_i] - [\mathbf{g}_i] = 0 \tag{4.75}$$

6. Cette écriture dans la base propre d'un système entier correspond à un modèle d'amortissement de Rayleigh.

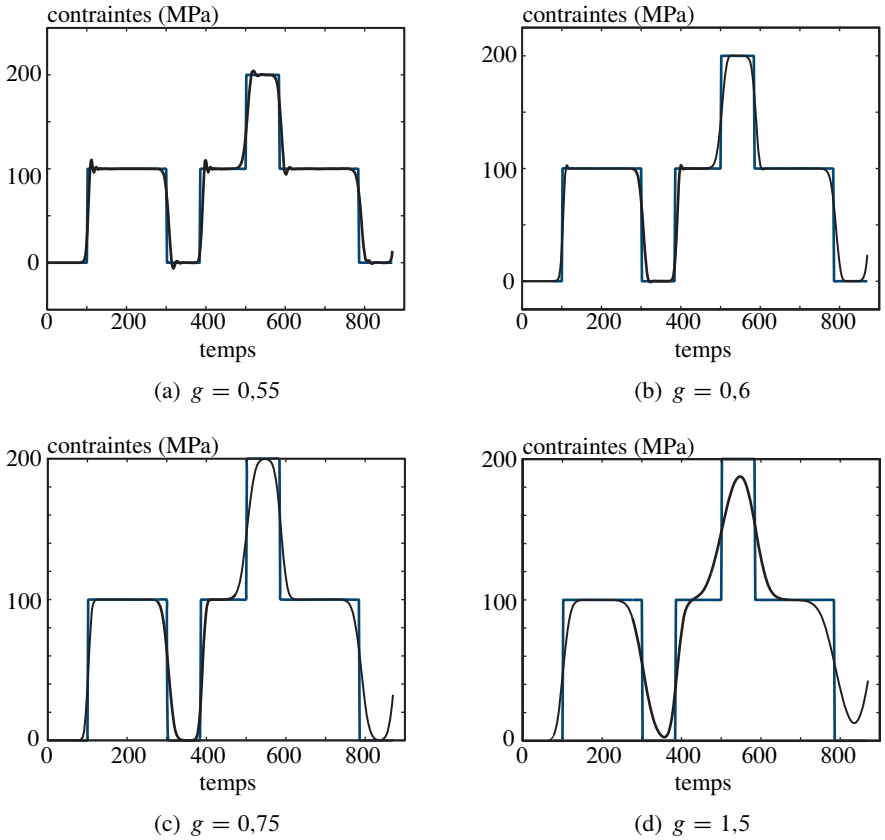


Figure 4.10 — Influence de l'amortissement numérique (d'après [46])

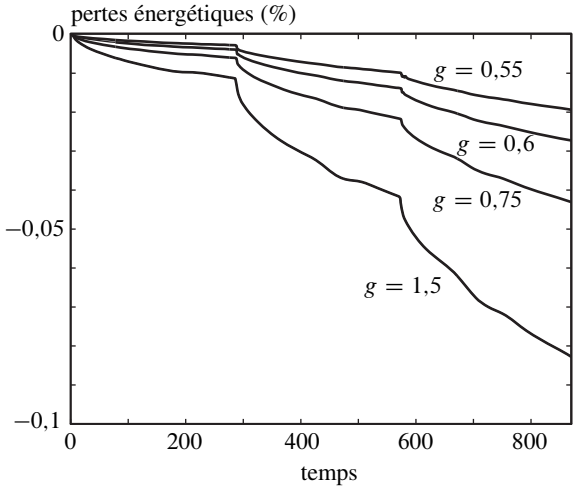


Figure 4.11 — Influence de l'amortissement numérique (suite) (d'après [46])

En multipliant cette dernière relation par $[\dot{\mathbf{x}}_i]^{\mathrm{T}}$ et en utilisant la propriété $\langle \mathbf{x}_i \rangle^{\mathrm{T}} [\mathbf{x}_i] = [\frac{1}{2}\mathbf{x}_i^2]$,

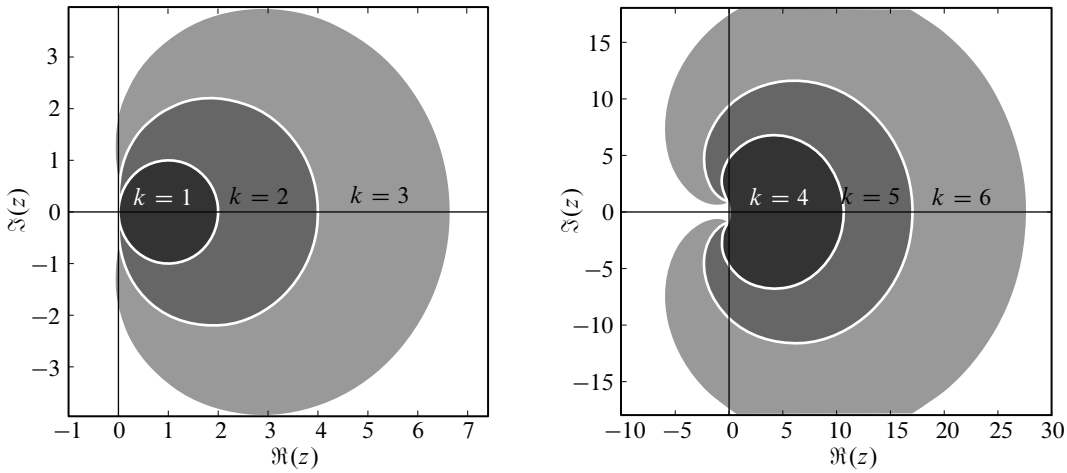


Figure 4.12 — A-stabilité dans le plan complexe pour la méthode de Gear : k est le nombre de pas

avec des matrices $\mathbf{M}$ et $\mathbf{K}$ positives, on obtient le bilan :

$$\left[\frac{1}{2}\ddot{\mathbf{x}}_i^{\mathrm{T}}\mathbf{M}^{\star}\ddot{\mathbf{x}}_i\right] + \left[\frac{1}{2}\dot{\mathbf{x}}_i^{\mathrm{T}}\mathbf{K}\dot{\mathbf{x}}_i\right] = \frac{1}{h}\left[\dot{\mathbf{x}}_i\right]^{\mathrm{T}}\left[\mathbf{g}_i\right] - \left(\gamma - \frac{1}{2}\right)\left[\ddot{\mathbf{x}}_i\right]^{\mathrm{T}}\mathbf{M}^{\star}\left[\ddot{\mathbf{x}}_i\right] \tag{4.76}$$

où $\mathbf{M}^{\star} = \mathbf{M} + \frac{1}{2}h^2(2\beta - \gamma)\mathbf{K}$. On trouve donc, dans l'ordre, un terme lié à la variation de l'énergie cinétique, un terme lié à la variation de l'énergie interne, un terme lié au travail des efforts extérieurs et un dernier terme lié à l'amortissement numérique (dissipation). Cela permet de conclure que si $\gamma \geqslant 0{,}5$, alors la solution est d'amplitude bornée et que si $\gamma = 0{,}5$, le schéma numérique ne dissipe pas d'énergie.

4.3.2 Méthode de Gear implicite à deux pas

Cette méthode [23] proposé en 1969, appelée aussi méthode BDF pour *Backward Differentiation Formulas*, est réputée être très stable. Pour information, elle utilise le développement suivant :

$$\begin{aligned}\dot{\mathbf{x}}_{i+1} &= \frac{1}{2h}(3\mathbf{x}_{i+1} - 4\mathbf{x}_i + \mathbf{x}_{i-1}) \\ \ddot{\mathbf{x}}_{i+1} &= \frac{1}{2h}(3\dot{\mathbf{x}}_{i+1} - 4\dot{\mathbf{x}}_i + \dot{\mathbf{x}}_{i-1})\end{aligned} \tag{4.77}$$

De la même façon, l'équilibre (4.57) à l'instant t_{i+1} donne le système à résoudre pour atteindre $\ddot{\mathbf{x}}_{i+1}$:

$$\left(\mathbf{M} + \frac{4}{9}h^2\mathbf{K}\right)\ddot{\mathbf{x}}_{i+1} = \mathbf{g}_{i+1} - \mathbf{K}\left(\frac{4}{3}\mathbf{x}_i - \frac{1}{3}\mathbf{x}_{i-1} + \frac{8}{9}h\dot{\mathbf{x}}_i - \frac{2}{9}h\dot{\mathbf{x}}_{i-1}\right) \tag{4.78}$$

La figure 4.12 présente le domaine de stabilité de cette méthode.

4.3.3 θ-méthode

Historiquement attribuée à Wilson en 1968 [5], cette méthode estime une accélération linéaire entre les instants t_i et $t_{i+\theta} = t_i + \theta h$, $\theta \geqslant 0$ étant un paramètre de la méthode. On a

donc :

$$\ddot{\mathbf{x}}(t_i+\tau) = \frac{\ddot{\mathbf{x}}_{i+\theta} - \ddot{\mathbf{x}}_i}{\theta h}\tau + \ddot{\mathbf{x}}_i \tag{4.79}$$

et par intégration, on a les expressions :

$$\begin{aligned} \dot{\mathbf{x}}_{i+\theta} &= \dot{\mathbf{x}}_i + \frac{1}{2}\theta h(\ddot{\mathbf{x}}_i + \ddot{\mathbf{x}}_{i+\theta}) \\ \mathbf{x}_{i+\theta} &= \mathbf{x}_i + \theta h\dot{\mathbf{x}}_i + \theta^2 h^2\Big(\frac{1}{3}\ddot{\mathbf{x}}_i + \frac{1}{6}\ddot{\mathbf{x}}_{i+\theta}\Big) \end{aligned} \tag{4.80}$$

Cette fois-ci, l'équilibre (4.57) à l'instant $t_{i+\theta}$ conduit au système en $\mathbf{x}_{i+\theta}$:

$$\Big(\mathbf{K} + \frac{6}{\theta^2 h^2}\mathbf{M}\Big)\mathbf{x}_{i+\theta} = \mathbf{g}_{i+\theta} + \mathbf{M}\Big(\frac{6}{\theta^2 h^2}\mathbf{x}_i + \frac{6}{\theta h}\dot{\mathbf{x}}_i + 2\ddot{\mathbf{x}}_i\Big) \tag{4.81}$$

— $\theta \geqslant 1{,}37$: la méthode est inconditionnellement stable ;
— $\theta = 1$: méthode des accélérations linéaires.

4.3.4 α-HHT

Cette méthode [33] proposée en 1977, ressemble *a priori* à la méthode de Newmark, à ceci près qu'elle introduit une modification de l'équilibre (et donc en particulier du second membre) en utilisant un paramètre additionnel α.

L'écriture du déplacement, comme celle de la vitesse sont identique à la méthode de Newmark, et l'équilibre est écrit sous la forme :

$$\mathbf{M}\ddot{\mathbf{x}}_{i+1} + \big((1+\alpha)\mathbf{C}\dot{\mathbf{x}}_{i+1} - \alpha\mathbf{C}\dot{\mathbf{x}}_i\big) + \big((1+\alpha)\mathbf{K}\mathbf{x}_{i+1} - \alpha\mathbf{K}\mathbf{x}_i\big) = \big((1+\alpha)\mathbf{f}_{i+1} - \alpha\mathbf{f}_i\big) \tag{4.82}$$

Le démarrage est réalisé avec $\mathbf{x}_0$ et $\dot{\mathbf{x}}_0$ données ainsi que $\ddot{\mathbf{x}}_0 = \mathbf{M}^{-1}(\mathbf{f}_0 - \mathbf{C}\dot{\mathbf{x}}_0 - \mathbf{K}\mathbf{x}_0)$.

Pour le choix de paramètres de la forme $\beta = (1-\alpha)^2/4$ et $\gamma = 1/2 - \alpha$, le schéma est du second ordre pour les basses fréquences, il est inconditionnellement stable pour $-1/3 \leqslant \alpha \leqslant 0$, la dissipation en hautes fréquences est maximum pour $\alpha = -1/3$, enfin, si on impose de plus $\alpha = 0$ on retrouve la méthode des trapèzes.

4.3.5 Autres méthodes

Il existe un grand nombre de méthodes d'intégration plus ou moins adaptées à un type de problème. En dynamique, citons encore la méthode de Park à trois pas [56].

4.4 Intégration de modèles de comportement

Si on considère le problème du calcul de structure non-linéaire de la section 3.1.1, on était conduit dans les méthodes de résolution à estimer le résidu $\mathbf{r}(\mathbf{u}) = \mathbf{K}(\mathbf{u})\mathbf{u} - \mathbf{f}$ qui dépend de l'état courant au travers du modèle de comportement du matériau :

$$\mathbf{K}(\mathbf{u})\mathbf{u} = \int_\Omega \mathrm{Tr}\boldsymbol{\sigma}\,\boldsymbol{\varepsilon}(\mathbf{u})\,\mathrm{d}\Omega \tag{4.83}$$

puisque la contrainte σ dépend de l'état courant.

4.4.1 Exemple de modèle de plasticité

Utilisons les notations de la section 3.1.1. Les lois d'état sont de la forme :

$$\begin{aligned} \sigma &= \mathrm{D}(\varepsilon - \varepsilon_p) \\ \mathbb{Y} &= \mathrm{G}(\mathbb{X}) \end{aligned} \tag{4.84}$$

où D est l'opérateur de Hooke et G la fonction d'écrouissage, caractéristiques du matériau. Dans le cadre d'un modèle associé, le potentiel de dissipation est pris égal au seuil de plasticité, sur lequel s'annule la fonction caractéristique f dépendant de l'état $f(\sigma, \mathbb{Y})$.

Les lois d'évolution s'écrivent avec l'utilisation d'un multiplicateur plastique $\dot{\lambda}$ à cause du caractère non différentiable du potentiel de dissipation :

$$\dot{\varepsilon}_p = \dot{\lambda}\frac{\partial f}{\partial \sigma} \tag{4.85}$$

$$-\dot{\mathbb{X}} = \dot{\lambda}\frac{\partial f}{\partial \mathbb{Y}} \tag{4.86}$$

On a alors les conditions de complémentarité de Khun-Tucker :

$$f \leqslant 0, \qquad \dot{\lambda} \geqslant 0, \qquad f\dot{\lambda} = 0 \tag{4.87}$$

L'expression du multiplicateur est donnée dans le cas de l'écoulement plastique par la condition de cohérence :

$$\dot{f} = 0 = \frac{\partial f}{\partial \sigma}\dot{\sigma} + \frac{\partial f}{\partial \mathbb{Y}}\dot{\mathbb{Y}} \tag{4.88}$$

(dans le cas contraire $\dot{\lambda} = 0$ et le comportement est localement élastique). En remplaçant $\dot{\sigma}$ et $\dot{\mathbb{Y}}$ par leurs expressions issues de la dérivation des lois d'état, et en utilisant les lois d'évolution, on obtient l'expression du multiplicateur :

$$\dot{\lambda} = \frac{\frac{\partial f}{\partial \sigma}\mathrm{D}\dot{\varepsilon}}{\frac{\partial f}{\partial \sigma}\mathrm{D}\frac{\partial f}{\partial \sigma} + \frac{\partial f}{\partial \mathbb{Y}}\frac{\partial \mathrm{G}}{\partial \mathbb{X}}\frac{\partial f}{\partial \mathbb{Y}}} \tag{4.89}$$

On peut donc formellement écrire (à partir des lois d'évolution, de l'expression précédente du multiplicateur et en remplaçant chaque fois que nécessaire σ et $\mathbb{Y}$ par leurs expressions en ε, ε_p et $\mathbb{X}$ issues des lois d'état) l'évolution du comportement sous la forme d'un système d'équations non-linéaires différentielles ordinaires (ODE) :

$$\begin{pmatrix} \dot{\varepsilon}_p \\ -\dot{\mathbb{X}} \end{pmatrix} = g\left(\begin{pmatrix} \varepsilon_p \\ \mathbb{X} \end{pmatrix}, \varepsilon, \dot{\varepsilon}\right) \tag{4.90}$$

Il est à noter que le temps n'intervient pas directement dans l'évolution, contrairement au cas des comportements visqueux. Remarquons aussi que ce système ne peut être résolu : il comporte trop d'inconnues par rapport au nombre d'équations disponibles. En effet, il faut une sollicitation pour que le comportement évolue. En utilisation dans un code aux éléments finis en déplacement, il est agréable de conserver une déformation admissible tout le temps. Le modèle de comportement est donc généralement piloté en intégration par le

fait que l'évolution de la déformation totale est imposée. Dans le système précédent, ε et $\dot{\varepsilon}$ sont alors des données.

La résolution de ce système différentiel peut être effectuée avec les techniques décrites dans le chapitre 4. On peut ainsi utiliser des méthodes explicites (avec des sous-pas plus nombreux) ou implicites (avec parfois un seul sous-pas par pas de chargement de la structure, c'est-à-dire par résolution de système différentiel). Classiquement, Runge-Kutta d'ordre 4 avec contrôle de la taille du sous-pas et Euler implicite sont utilisés. En fait, ces méthodes sont utilisées sur une partie du problème à résoudre seulement, car souvent, des méthodes de type retour radial sont employées.

4.4.2 Méthodes avec retour radial

Ces méthodes sont employées car elles sont mieux adaptées au type de problème à traiter. On souhaite en effet retourner assez précisément sur le seuil après intégration du système différentiel. De plus, il faut pouvoir prendre en compte le cas de la décharge élastique. Pour cela, on prend en compte la propriété de séparation en partie élastique et plastique des modèles de comportement, pour séparer en deux temps l'intégration. Considérons un incrément de charge sur la structure, et un pilotage en incrément de déformation totale, noté $\Delta\varepsilon$. Cette méthode est aussi considérée comme une méthode de partage d'opérateur (*operator split*, cf. prédiction élastique et correction plastique par un retour sur le seuil ou *return map*). La notation δ représente une variation sur un éventuel sous-pas d'intégration. La technique employée consiste en :

Prédiction élastique Elle est telle que $\Delta\varepsilon_e + \Delta\varepsilon_p = \Delta\varepsilon$, avec $\Delta\varepsilon_p = 0$ et $\Delta\mathbb{X} = 0$, les lois d'état (4.84) donnent sur cet incrément :

$$\begin{pmatrix} \Delta\sigma \\ \Delta\mathbb{Y} \end{pmatrix} = \begin{bmatrix} \mathrm{D} & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta\varepsilon \\ 0 \end{pmatrix} \tag{4.91}$$

Si, au bout de cet incrément, le nouvel état $(\sigma + \Delta\sigma, \varepsilon_p, \mathbb{X}, \mathbb{Y} + \Delta\mathbb{Y})$ vérifie $f \leqslant 0$, on est à l'intérieur du seuil et le comportement est resté élastique. Sinon, il faut corriger la prédiction de la façon suivante.

Correction À partir de l'état déterminé précédemment, on procède à une nouvelle correction (toujours notée avec Δ) $\Delta\varepsilon_e + \Delta\varepsilon_p = 0$ et on cherche à vérifier les équations d'état, les équations d'évolution et, généralement, directement le retour sur le seuil $f = 0$ à la fin de la correction. Cette phase de correction peut être réalisée par utilisation des méthodes d'intégration déjà citées. On peut aussi utiliser des méthodes comme celle du retour radial d'Ortiz et Simo [52, 53].

Retour radial d'Ortiz et Simo

Pour résoudre la phase de correction, l'idée est d'utiliser une méthode de Newton sur le seuil. On cherche à avoir :

$$f(\sigma + \Delta\sigma, \mathbb{Y} + \Delta\mathbb{Y}) = 0 \tag{4.92}$$

on procède par sous-pas (notés avec un δ) en utilisant la matrice jacobienne des dérivées partielles de f lors des développements successifs :

$$f(\sigma + \delta\sigma, \mathbb{Y} + \delta\mathbb{Y}) \approx f(\sigma, \mathbb{Y}) + \frac{\partial f}{\partial \sigma}\delta\sigma + \frac{\partial f}{\partial \mathbb{Y}}\delta\mathbb{Y} \tag{4.93}$$

pour lesquels on cherche à rester sur le seuil :

$$f + \frac{\partial f}{\partial \sigma}\delta\sigma + \frac{\partial f}{\partial \mathbb{Y}}\delta\mathbb{Y} = 0 \tag{4.94}$$

Comme $\delta\varepsilon = 0$ lors de la correction, les équations (4.85) et (4.86) donnent alors la valeur d'un « multiplicateur plastique » pour le sous-pas :

$$\delta\lambda = \frac{f}{\frac{\partial f}{\partial \sigma}\mathrm{D}\frac{\partial f}{\partial \sigma} + \frac{\partial f}{\partial \mathbb{Y}}\frac{\partial \mathrm{G}}{\partial \mathbb{X}}\frac{\partial f}{\partial \mathbb{Y}}} \tag{4.95}$$

Le potentiel de dissipation étant convexe, le dénominateur de l'expression précédente est toujours positif. Après avoir sommé ce sous-incrément à l'état courant, on itère sur la correction jusqu'à annuler le résidu $f(\sigma, \mathbb{Y})$.

Rigidité tangente cohérente

Il a été constaté que si l'on utilise la matrice de rigidité tangente continue précédente, pour itérer avec une méthode de Newton-Raphson (voir le chapitre 3.1.5), on perd la convergence quadratique de la méthode, et que pour la retrouver, il faut utiliser la matrice de rigidité tangente cohérente avec le schéma d'intégration employé.

Ce dernier peut être vu comme un outil pour produire, à partir d'un état donné :

$$s = (\sigma, \mathbb{Y}, \varepsilon_p, \mathbb{X}) \tag{4.96}$$

un nouvel état satisfaisant au comportement du matériau, piloté par exemple par un incrément de déformation totale $\Delta\varepsilon$. Ce nouvel état est noté :

$$s + \Delta s = (\sigma + \Delta\sigma, \mathbb{Y} + \Delta\mathbb{Y}, \varepsilon_p + \Delta\varepsilon_p, \mathbb{X} + \Delta\mathbb{X}) \tag{4.97}$$

Si la rigidité tangente continue à l'état s est $\mathrm{D_T}$ telle que $\dot{\sigma} = \mathrm{D_T}\dot{\varepsilon}$, la matrice tangente cohérente $\mathrm{D_C}$ est définie par $\delta\Delta\sigma = \mathrm{D_C}\delta\Delta\varepsilon$. Elle relie la variation de correction de contrainte obtenue, à la variation de pilotage en incrément de déformation totale : piloter avec $\Delta\varepsilon + \delta\Delta\varepsilon$ à partir du même état s conduira à un nouvel état suivant $s + \delta s + \delta\Delta s$.

On peut obtenir $\mathrm{D_C}$ par une méthode de perturbation (on procède à plusieurs intégrations avec des perturbations différentes $\delta\Delta\varepsilon$ sur le pilotage) mais cela est assez coûteux numériquement, et nécessite le choix de la valeur de la perturbation (trop petite, elle conduit à des problèmes de précision numériques, trop grande, à une mauvaise approximation de la dérivée). Une autre technique consiste à dériver analytiquement le schéma d'intégration, ce qui peut s'avérer assez pénible suivant le schéma et le modèle de comportement.

4.4.3 Modèle de viscoplasticité

Par rapport au modèle précédente, un grand nombre de modèles de viscoplasticité s'écrivent en n'imposant plus de signe sur la valeur de la fonction seuil $f(\sigma, \mathbb{Y})$, mais avec un terme de rappel sur le multiplicateur viscoplastique $\dot{\lambda}$: les conditions de complémentarité de Kuhn-Tucker (4.87) sont alors remplacées par :

$$\dot{\lambda} = \left(\left\langle \frac{f(\sigma, \mathbb{Y})}{m} \right\rangle_+ \right)^n \tag{4.98}$$

où les parties positive et négative sont respectivement notées :

$$\langle x \rangle_+ = \max(0, x) = \frac{x + |x|}{2} \tag{4.99}$$

et :

$$\langle x \rangle_- = \min(0, x) = \frac{x - |x|}{2} = -\langle -x \rangle_+ \tag{4.100}$$

m est un module de viscoplasticité et n un exposant de viscosité.

Il est possible de définir l'équivalent du seuil de plasticité de la façon suivante : si on considère $\bar{f} = f - \langle f \rangle_+$, alors $\bar{f} = 0$ correspond à $f \geqslant 0$ c'est-à-dire à un écoulement viscoplastique, et $\bar{f} < 0$ correspond à $f \leqslant 0$ c'est-à-dire à une évolution élastique, et dans ce cas $\bar{f} = f$. On peut aussi se ramener à une relation de complémentarité : si on note

Ayant les incréments donnés $\Delta\varepsilon$ et Δt,
prédiction élastique avec (4.101)
prédiction du seuil $f(\sigma, \mathbb{Y})$
si $f > 0$ **alors**
  correction viscoplastique
  $\mu = 0$, $\tilde{f} = f$
  tant que $\tilde{f} > 0$
    estimer $\dot{\varepsilon}_p$ et $\dot{\mathbb{X}}$ avec (4.105)
    calculer la correction $\delta\mu$ avec (4.106)
    mise à jour de σ, $\mathbb{Y}$, ε_p, $\mathbb{X}$ avec (4.104)
    mise à jour du multiplicateur $\mu \leftarrow \mu + \delta\mu$
    mise à jour du seuil corrigé $\tilde{f}$
  fin
fin

Algorithme 4.1 — Intégration du comportement viscoplastique

$\mu = \dot{\lambda}$, on doit en effet avoir $\mu \geqslant 0$, $\tilde{f} = f - m\mu^{1/n} \leqslant 0$ et $\mu\tilde{f} = 0$. La principale différence par rapport au cas de la plasticité est la dépendance avec le temps au travers des vitesses des quantités cherchées. L'intégration de la loi de comportement ne sera plus pilotée par le seul incrément de déformation $\Delta\varepsilon$, mais aussi par l'incrément de temps Δt. Le principe consiste encore à prédire une évolution élastique :

$$\begin{aligned} &\Delta\sigma = \mathrm{D}\Delta\varepsilon, \ \Delta\mathbb{Y} = 0 \\ &\Delta\mathbb{X} = 0, \ \Delta\varepsilon_p = 0 \end{aligned} \tag{4.101}$$

et à estimer la valeur du seuil $f(\sigma, \mathbb{Y})$. Si celle-ci est négative, la solution est effectivement élastique. Sinon, il faut procéder à une correction viscoplastique ; cette étape est encore itérative, mais il faut bien réévaluer toutes les quantités (en particulier les dérivées par l'intermédiaire du schéma d'intégration en temps choisi, ici la θ-méthode) par rapport au précédent pas de temps t_i (elle sont ici indicées par i). La correction d'effectue toujours à déformation totale nulle, et l'estimation du seuil corrigé est :

$$\tilde{f}(\sigma + \delta\sigma, \mathbb{Y} + \delta\mathbb{Y}, \mu + \delta\mu) \approx \tilde{f}(\sigma, \mathbb{Y}, \mu) + \frac{\partial \tilde{f}}{\partial \sigma}\delta\sigma + \frac{\partial \tilde{f}}{\partial \mathbb{Y}}\delta\mathbb{Y} + \frac{\partial \tilde{f}}{\partial \mu}\delta\mu \tag{4.102}$$

pour lesquels on cherche à rester sur le seuil corrigé :

$$f(\sigma, \mathbb{Y}) - m\mu^{1/n} + \frac{\partial f}{\partial \sigma}\delta\sigma + \frac{\partial f}{\partial \mathbb{Y}}\delta\mathbb{Y} - \frac{m}{n}\mu^{1/n-1}\delta\mu = 0 \tag{4.103}$$

avec $\delta\sigma = -\mathrm{D}\delta\varepsilon_p$, $\delta\mathbb{Y} = \frac{\partial \mathrm{G}}{\partial \mathbb{X}}\delta\mathbb{X}$ et les incréments :

$$\begin{aligned} \frac{\delta\varepsilon_p}{\theta\Delta t} &= (\mu + \delta\mu)\frac{\partial f}{\partial \sigma} - \dot{\varepsilon}_p \\ \frac{\delta\mathbb{X}}{\theta\Delta t} &= -(\mu + \delta\mu)\frac{\partial f}{\partial \mathbb{Y}} - \dot{\mathbb{X}} \end{aligned} \tag{4.104}$$

où les dérivées sont estimées par rapport au précédent pas de temps (et doivent être réévaluée à chaque itéré de la correction) :

$$\begin{aligned} \dot{\varepsilon}_p &= \frac{\varepsilon_p - \varepsilon_{pi}}{\theta\Delta t} - \frac{1-\theta}{\theta}\dot{\varepsilon}_{pi} \\ \dot{\mathbb{X}} &= \frac{\mathbb{X} - \mathbb{X}_i}{\theta\Delta t} - \frac{1-\theta}{\theta}\dot{\mathbb{X}}_i \end{aligned} \tag{4.105}$$

L'équation (4.102) permet alors d'obtenir l'incrément de multiplicateur viscoplastique :

$$\delta\mu = \frac{\frac{f - m\mu^{1/n}}{\theta\Delta t} + \frac{\partial f}{\partial \sigma}\mathrm{D}\dot{\varepsilon}_p - \frac{\partial f}{\partial \mathbb{Y}}\frac{\partial \mathrm{G}}{\partial \mathbb{X}}\dot{\mathbb{X}}}{\frac{\partial f}{\partial \sigma}\mathrm{D}\frac{\partial f}{\partial \sigma} + \frac{\partial f}{\partial \mathbb{Y}}\frac{\partial \mathrm{G}}{\partial \mathbb{X}}\frac{\partial f}{\partial \mathbb{Y}} + \frac{m}{n}\mu^{1/n-1}\frac{1}{\theta\Delta t}} \tag{4.106}$$

La valeur courante du multiplicateur doit alors aussi être mise à jour par $\mu + \delta\mu$, à chaque itéré de la correction, qui est poursuivie jusqu'à retourner sur le seuil corrigé $\tilde{f}$.

Connaissant la solution à t_i, on procède à une succession de prédictions élastiques jusqu'à rééquilibrer les forces extérieures (c'est-à-dire à annuler le résidu des équations d'équilibre) et d'intégration du comportement selon l'algorithme décrit dans 4.1. La solution à t_{i+1} est celle qui doit vérifier à la fois l'équilibre et le comportement.

4.5 Cas de la dynamique non régulière

Dans les approches précédentes, on a souvent considéré que les évolutions temporelles étaient suffisamment régulières pour pouvoir les approcher avec un développement limité. Ce n'est pas forcément toujours le cas ; par exemple dans le cas de la fissuration dynamique,

lorsque les structures sont soumises à des chocs ou impacts. Dans ce dernier cas, en particulier quand les corps sont des solides rigides, les vitesses ne sont plus dérivables (lors d'un choc, un corps rigide change peu en position, mais beaucoup en vitesse, par exemple dans le cas d'un rebond). Une technique (dite *event driven*) consiste à traiter de façon standard les phases de *vol libre* c'est-à-dire sans impact, à repérer les instants de choc, et à traiter ceux-ci de façon particulière, avec un saut de vitesse de part et d'autre de l'instant de choc.

Avec une modélisation en solides rigides, il manque une équation de comportement de l'impact pour fermer le problème. Un tel comportement peut être modélisé par la loi de Newton[7]. Elle postule que la vitesse normale du point de contact — d'où des difficultés pour modéliser un impact sur une surface entière — après impact (notée v_n^+) est opposée en direction à celle avant impact (notée v_n^-) :

$$v_n^+ = -e v_n^- \tag{4.107}$$

le saut de vitesse lors de l'impact étant $v_n^+ - v_n^-$. Le coefficient *matériau* e, appelé coefficient de restitution d'énergie, traduit le comportement de l'impact ; $0 \leqslant e \leqslant 1$. Le problème est qu'il dépend de beaucoup de paramètres du problème, puisqu'il représente de façon phénoménologique le comportement complexe des propagations / réflections d'ondes lors de l'impact, ou des impacts multiples...

Sans présence de frottement, il n'est pas nécessaire d'écrire autre choses sur les glissements tangentiels lors de l'impact. On se contentera de décrire ce type de problème ici.

Une autre situation particulière est celle des chocs nombreux dans les matériaux granulaires ou soumis à la fragmentation... Dans ce cas, une technique *event driven* est inapplicable à cause des impacts qui peuvent être extrêmement nombreux. Une alternative consiste à employer une technique de *time-stepping*. Le pas de temps est fixé et on s'emploie à déterminer les vitesses immédiatement après les piquets de temps, en considérant l'impact *moyen* pendant le pas de temps courant. On ne s'intéresse alors pas vraiment à l'évolution du système continûment sur le pas de temps, mais plutôt à une configuration possible en chaque piquet de temps.

Considérons tout d'abord un système dynamique régulier (sans choc), rigide, évoluant sous l'action d'une force extérieure f. Son équation du mouvement est de la forme :

$$m\ddot{x} = f \tag{4.108}$$

où l'on suppose m constante. On se propose d'intégrer cette équation différentielle avec un schéma particulier en diminuant l'ordre du système[8]. Pour cela, commençons par intégrer l'équation du mouvement sur un pas de temps $[t_i, t_{i+1}]$; on obtient :

$$m(\dot{x}_{i+1} - \dot{x}_i) = \int_{t_i}^{t_{i+1}} f(t)\ \mathrm{d}t \tag{4.109}$$

L'intégrale de la force extérieure (régulière) peut être approximée par un schéma d'intégration, par exemple la méthode des trapèzes généralisés, voir section 4.1.6. Le problème

7. Elle masque le fait que le solide est quand même légèrement déformable et que le rebond est dû à un échange entre l'énergie cinétique et l'énergie de déformation, lors de la propagation des ondes de chocs dans le solide.

8. On travaillera alors à la fois sur les positions x_i et les vitesses $\dot{x}_i$ aux piquets de temps t_i.

s'écrit alors :

$$m\dot{x}_{i+1} = m\dot{x}_i + (1-\alpha)hf_i + \alpha h f_{i+1} \tag{4.110}$$

La position est récupérée par une intégration similaire de la vitesse :

$$x_{i+1} = x_i + \int_{t_i}^{t_{i+1}} \dot{x}\ \mathrm{d}t \approx x_i + (1-\alpha)h\dot{x}_i + \alpha h\dot{x}_{i+1} \tag{4.111}$$

Le lecteur pourra vérifier qu'il s'agit exactement de l'application de la méthode des trapèzes généralisés au système du premier ordre :

$$\begin{aligned} m\dot{v} &= f \\ \dot{x} &= v \end{aligned} \tag{4.112}$$

Cette façon de procéder va permettre de traiter les systèmes non réguliers comprenant des chocs. En effet, l'équation de la dynamique entre deux instants immédiatement suivant les piquets de temps t_i et t_{i+1} va s'écrire :

$$m(\dot{x}_{i+1} - \dot{x}_i) = \int_{t_i}^{t_{i+1}} f(t)\ \mathrm{d}t + \mathrm{R} \tag{4.113}$$

où R est l'impulsion totale, c'est-à-dire, la somme des impulsions lors des impacts éventuels sur le pas de temps traité.

Dans le cas d'une évolution statique, le contact unilatéral du système contre un mur rigide placé en $x = 0$ par exemple ferait intervenir un effort F et non pas une impulsion R, avec une loi de comportement du contact parfait du type $0 \leqslant x \perp \mathrm{F} \geqslant 0$, qui est une version concise de $x \geqslant 0$, $\mathrm{F} \geqslant 0$, $x \cdot \mathrm{F} = 0$. Le pendant de ce comportement en dynamique est la condition de Signorini-vitesse issue du Lemme de Moreau :

— si $x > 0$, $\mathrm{R} = 0$
— si $x = 0$, $0 \leqslant \dot{x} \perp \mathrm{R} \geqslant 0$

Pour dicrétiser en temps ce comportement, on utilise en général une version explicite sur la configuration — c'est-à-dire qu'on va utiliser un prédicteur pour la position x qui permettra de sélectionner le cas correspondant, et qu'on notera x^p :

— si $x^p > 0$, $\mathrm{R}_{i+1} = 0$
— si $x^p \leqslant 0$, $0 \leqslant \dot{x}_{i+1} \perp \mathrm{R}_{i+1} \geqslant 0$

Un prédicteur classique (saute-mouton) est $x^p = x_i + \frac{h}{2}\dot{x}_i$. L'inconvénient est la possibilité d'avoir une pénétration résiduelle, d'où l'inégalité utilisée sur les valeurs possiblement négatives de x^p, qui tend cependant vers 0 avec le pas de temps. Ce modèle de comportement correspond à un impact plastique (sans rebond), qui dissipe le plus l'énergie. Dans le cas de choc unique, on peut retrouver l'équivalent du modèle de Newton en substituant $\dot{x}_{i+1}$ par une vitesse dite *formelle* :

$$\tilde{v} = \frac{\dot{x}_{i+1} - \dot{x}_i}{1+e} \tag{4.114}$$

Bibliographie

[1] A. AITKEN. « On the iterative solution of a system of linear equations ». *Proceedings of the Royal Society of Edinburgh* 63 1950, p. 52–60. 87

[2] E. ANDERSON, Z. BAI, C. BISCHOF, S. BLACKFORD, J. DEMMEL, J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY et D. SORENSEN. *LAPACK Users' Guide*. 3e édition. Philadelphia : Society for Industrial et Applied Mathematics, 1999. ISBN : 0-89871-447-8. 8

[3] Z. BAI, J. DEMMEL, J. DONGARRA, A. RUHE et H. van der VORST, éditeurs. *Templates for the Solution of Algebraic Eigenvalue Problems : A Practical Guide*. Draft. 1999. 33

[4] R. BARRETT, M. BERRY, T. CHAN, J. DEMMEL, J. DONATO, J. DONGARRA, V. EIJKHOUT, R. POZO, C. ROMINE et H. Van der VORST. *Templates for the Solution of Linear Systems : Building Blocks for Iterative Methods*. 2e édition. Philadelphia : SIAM, 1994. URL : http://www.netlib.org/linalg/html_templates/Templates.html. 11

[5] K. BATHE et E. WILSON. « Stability and accuracy analysis of direct integration methods ». *Earthquake Engineering & Structural Dynamics* 1(3) 1972, p. 283–291. ISSN : 1096-9845. DOI : 10.1002/eqe.4290010308 71

[6] K. BATHE et E. WILSON. *Numerical Analysis in Finite Element Analysis*. 1re édition. Prentice-Hall, 1976. ISBN : 9780136271901. 8

[7] Commandant BENOÎT. « Note sur une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés à un système d'équations linéaires en nombre inférieur à celui des inconnues — Application de la méthode à la résolution d'un système d'équations linéaires (procédé du Commandant Cholesky) ». *Bulletin Géodésique* 2 1924, p. 5–77. 18

[8] L. BLACKFORD, J. CHOI, A. CLEARY, E. D'AZEVEDO, J. DEMMEL, I. DHILLON, J. DONGARRA, S. HAMMARLING, G. HENRY, A. PETITET, K. STANLEY, D. WALKER et R. WHALEY. *ScaLAPACK Users' Guide*. Philadelphia : Society for Industrial et Applied Mathematics, 1997. ISBN : 0-89871-397-8. 8

[9] C. BROYDEN. « The Convergence of a Class of Double-rank Minimization Algorithms. General Considerations ». *IMA Journal of Applied Mathematics* 6(3) 1970, p. 76–90. DOI : 10.1093/imamat/6.1.76 51

[10] J.-P. COMBE. « Sur le contrôle des calculs en dynamique rapide. Application aux problèmes d'impact ». Thèse de doctorat. Laboratoire de Mécanique et Technologie : École Nationale Supérieure de Cachan, 2000. 65

[11] M. CRISFIELD. *Non-Linear Finite Element Analysis of Solids and Structures*. Wiley, 1996. ISBN : 9780471970590. 47

[12] E. CUTHILL et J. MCKEE. « Reducing the Bandwidth of Sparse Symmetric Matrices ». *Proceedings of the 24th National Conference*. New York, USA : ACM, 1969, p. 157–172 DOI : 10.1145/800195.805928. 14

[13] G. DAHLQUIST. « A special stability problem for linear multistep methods ». *BIT. Numerical Mathematics* 3(1) 1963, p. 27–43. DOI : 10.1007/BF01963532 56

[14] Y. DAI et Y. YUAN. « Global convergence of the method of shortest residuals ». *Numerische Mathematik* 83(4) 1999, p. 581–598. ISSN : 0029-599X. DOI : 10.1007/s002119900080 49

[15] J. DONGARRA, H. MEUER et E. STROHMEIER. « TOP 500 supercomputer sites ». *Proceedings of Supercomputer '99 Conference*. 17. Mannheim, 1999.
URL : http://www.top500.org/. 14

[16] I. DUFF. « Parallel implementation of multifrontal schemes ». *Parallel Computing* 3(3) 1986, p. 193–204. ISSN : 0167-8191.
DOI : 10.1016/0167-8191(86)90019-0 20

[17] I. DUFF, A. ERISMAN et J. REID. *Direct Methods for Sparse Matrices*. Oxford : Oxford University Press, 1990. 16

[18] W. DUNCAN. « Some devices for the solution of large sets of simultaneous linear equations ». *Philosophical Magazine*. 7e série 35(249) 1944, p. 660–670.
DOI : 10.1080/14786444408520897 50

[19] C. FARHAT et D. RIXEN. « Encyclopedia of Vibration ». Academic Press, 2002. Chapitre Linear Algebra, p. 710–720. ISBN : 9780122270857. 23

[20] R. FLETCHER et M. POWELL. « A Rapidly Convergent Descent Method for Minimization ». *The Computer Journal* 6(2) 1963, p. 163–168.
DOI : 10.1093/comjnl/6.2.163 50

[21] R. FLETCHER et C. REEVES. « Function minimization by conjugate gradients ». *The Computer Journal* 7(2) 1964, p. 149–154.
DOI : 10.1093/comjnl/7.2.149 49

[22] R. FREUND et N. NACHTIGAL. « QMR : a quasi-minimal residual method for non-Hermitian linear systems ». *Numerische Mathematik* 60(1) 1991, p. 315–339. ISSN : 0029-599X.
DOI : 10.1007/BF01385726 30

[23] C. GEAR. *Numerical Initial Value Problems in Ordinary Differential Equation*. Prentice-Hall, 1971. ISBN : 9780136266068. 71

[24] A. GEORGE. « Nested dissection of a regular finite-element mesh ». *SIAM Journal on Numerical Analysis* 10 1973, p. 345–363.
DOI : 10.1137/0710032 14

[25] A. GEORGE et W. LIU. « The Evolution of the Minimum Degree Ordering Algorithm ». *SIAM Review* 31(1) 1989, p. 1–19. ISSN : 0036-1445.
DOI : 10.1137/1031001 14

[26] M. GÉRADIN et D. RIXEN. *Théorie des Vibrations. Application à la dynamique des structures*. Physique Fondamentale et Appliquée. Masson, 1996. ISBN : 9782225851735. 33

[27] M. GÉRADIN et D. RIXEN. *Mechanical Vibrations. Theory and Application to Structural Dynamics*. 2e édition. Wiley, 1997. ISBN : 9780471975465. 33

[28] G. GOLUB et C. Van LOAN. *Matrix Computations*. 2e édition. Baltimore : The Johns Hopkins University Press, 1989. ISBN : 9780801830105. 8

[29] L. HAGEMANA et D. YOUNG. *Applied Iterative Methods*. New York : Academic Press, 1981. ISBN : 9780486434773. 24

[30] W. HAGER. « Updating the Inverse of a Matrix ». *SIAM Review* 31(2) 1989, p. 221–239. ISSN : 0036-1445.
DOI : 10.1137/1031049 50

[31] G. HESSENBERG. *Transzendenz von* e *und* π *: ein Beitrag zur hoheren Mathematik vom elementaren Standpunkte aus*. Berlin : B. G. Teubner, 1912. 42

[32] M. HESTENES et E. STIEFEL. « Methods of conjugate gradients for solving linear systems ». *Journal of research of the National Bureau of Standards* 49 1952, p. 409–436. 29

[33] H. HILBER, T. HUGHES et R. TAYLOR. « Improved numerical dissipation for time integration algorithms in structural dynamics ». *Earthquake Engineering & Structural Dynamics* 5(3) 1977, p. 283–292. ISSN : 1096-9845. DOI : 10.1002/eqe.4290050306 72

[34] A. HOUSEHOLDER. « Unitary Triangularization of a Nonsymmetric Matrix ». *Journal of the Association of Computing Machinery* 5(4) 1958, p. 339–342. ISSN : 0004-5411. DOI : 10.1145/320941.320947 40

[35] C. HUANG et G. VERCHERY. « An exact structural static reanalysis method ». *Communications in Numerical Methods in Engineering* 13(2) 1997, p. 103–112. ISSN : 1099-0887. DOI : 10.1002/(SICI)1099-0887(199702)13:2<103::AID-CNM36>3.0.CO;2-D 50

[36] T. HUGHES. *The Finite Element Method, Linear Static and Dynamic Finite Element Analysis*. Englewood Cliffs : Prentice-Hall, 1987. ISBN : 9780133170252. 53

[37] T. HUGHES et T. BELYTSCHKO. *Nonlinear Finite Element Analysis*. Zace sevices Ltd - ICE Division, 1995. 69

[38] B. IRONS. « A frontal solution program for finite element analysis ». *International Journal for Numerical Methods in Engineering* 2(1) 1970, p. 5–32. DOI : 10.1002/nme.1620020104 20

[39] C. JACOBI. « Über ein leichtes Verfahren, die in der Theorie der Säcularstörungen vorkommenden Gleichungen numerish aufsulösen ». *Journal für die reine und angewandle Mathematik* 30 1846, p. 51–94. 34

[40] D. JORDAN et P. SMITH. *Nonlinear Ordinary Differential Equations*. Oxford applied mathematics et computing science series, 1987. 53

[41] C. KELLEY. *Solving Nonlinear Equations with Newton's Method*. Tome 1. Fundamentals of Algorithms. Philadelphia : Society for Industrial et Applied Mathematics, 2003. ISBN : 9780898715460. 47

[42] T. KOLDA, D. O'LEARY et L. NAZARETH. « BFGS with Update Skipping and Varying Memory ». *SIAM Journal on Optimization* 8(4) 1998, p. 1060–1083. ISSN : 1052-6234. DOI : 10.1137/S1052623496306450 51

[43] W. KUTTA. « Beitrag zur naherungsweisen Integration totaler Differentialgleichunge ». *Zeitschrift für Mathematik und Physik* 46 1901, p. 435–453. 61

[44] C. LANCZOS. « An iteration method dor the solution of the eigenvalue problem of linear differential and integral operators ». *Journal of research of the National Bureau of Standards* 45 1950, p. 225–280. 44

[45] J. LEMAITRE et J.-L. CHABOCHE. *Mechanics of Solid Materials*. Cambridge University Press, 1994. 47

[46] H. LEMOUSSU. « Approche non incrémentale pour le calcul de choc avec contact unilatéral avec frottement ». Thèse de doctorat. Laboratoire de Mécanique et Technologie : École Nationale Supérieure de Cachan, 1999. 69, 70

[47] D. LIU et J. NOCEDAL. « On the Limited Memory BFGS Method for Large Scale Optimization ». *Numerische Mathematik* 45(3) 1989, p. 503–528. ISSN : 0025-5610. DOI : 10.1007/BF01589116 51

[48] H. MEUER, E. STROHMAIER, J. DONGARRA et H. SIMON. « Top500 Supercomputer Sites ». *16th International Supercomputer Conference : Applications — Architectures — Trends*. 2001. URL : http://www.netlib.org/benchmark/top500.html. 14

[49] B. MOHAMMADI et J.-H. SAIAC. *Pratique de la simulation numérique*. Dunod / Industries et Technologies, 2003. ISBN : 210006407X. 7

[50] L. NAZARETH. « A relationship between BFGS and conjugate gradient algorithms and its implications for new algorithms ». *SIAM Journal on Numerical Analysis* 16 1979, p. 794–800. 51
DOI : 10.1137/0716059

[51] N. NEWMARK. « A method for computation of structural dynamics ». *Proceedings of the American Society of Civil Engineers*. Sous la direction d'EMS. Tome 85. 1959, p. 67–94. 66

[52] M. ORTIZ et E. POPOV. « Accuracy and Stability of Integration Algorithms for Elastoplastic Constitutive Relations ». *International Journal for Numerical Methods in Engineering* 21(9) 1985, p. 1561–1576. 74
DOI : 10.1002/nme.1620210902

[53] M. ORTIZ et J. SIMO. « An analysis of a new class of integration algorithms for elasto-plastic constitutive relations ». *International Journal for Numerical Methods in Engineering* 23(3) 1986, p. 353–366. 63, 74
DOI : 10.1002/nme.1620230303

[54] H. OUDIN. *Méthode des éléments finis*. École Centrale de Nantes, France, 2008.
OAI : cel.archives-ouvertes.fr:cel-00341772. 11

[55] C. PAIGE et M. SAUNDERS. « Solution of sparse indefinite systems of linear equations ». *SIAM Journal on Numerical Analysis* 12 1975, p. 617–629. 30
DOI : 10.1137/0712047

[56] K. PARK. « An Improved Stiffly Stable Method for Direct Integration of Nonlinear Structural Dynamic Equations ». *Journal of Applied Mechanics* 42(2) 1975, p. 464–470. ISSN : 0021-8936. 72
DOI : 10.1115/1.3423600

[57] E. POLAK et G. RIBIÈRE. « Note sur la convergence de méthodes de directions conjuguées ». *Revue française d'informatique et de recherche opérationnelle* 3(1) 1969, p. 35–43. ISSN : 0764-583X. . 49
URL : http://eudml.org/doc/193115

[58] J. RAO. *Advanced Theory of Vibration*. John Wiley & Sons, 1989. 33

[59] Y. SAAD. *Iterative Methods for Sparse Linear Systems*. 2000. 11

[60] Y. SAAD et M. SCHULTZ. « GMRES : A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems ». *SIAM Journal on Scientific and Statistical Computing* 7(3) 1986, p. 856–869. ISSN : 0196-5204. 30
DOI : 10.1137/0907058

[61] G. SHORTLEY. « Use of Tschebyscheff-polynomial operators in the numerical solution of boundary-value problems ». *Journal of Applied Physics* 24 1953, p. 392–396. 30
DOI : 10.1063/1.1721292

[62] P. SONNEVELD. « CGS, A Fast Lanczos-Type Solver for Nonsymmetric Linear systems ». *SIAM Journal on Scientific and Statistical Computing* 10(1) 1989, p. 36–52. 30
DOI : 10.1137/0910004

[63] R. SOUTHWELL. *Relaxation Methods in Theoretical Physics : A Continuation of the Treatise, Relaxation Methods in Engineering Science*. Tome 1. The Oxford engineering science series. The Clarendon Press, 1946. 26

[64] W. TINNEY et J. WALKER. « Direct solutions of sparse network equations by optimally ordered triangular factorization ». 55(11) 1967, p. 1801–1809. ISSN : 0018-9219. 14
DOI : 10.1109/PROC.1967.6011

[65] H. van der VORST. « BI-CGSTAB : A Fast and Smoothly Converging Variant of BI-CG for the Solution of Nonsymmetric Linear Systems ». *SIAM Journal on Scientific and Statistical Computing* 13(2) 1992, p. 631–644. ISSN : 0196-5204. DOI : 10.1137/0913035

[66] J. WILKINSON. *The Algebraic Eigenvalue Problem*. New York : Oxford University Press, 1965. ISBN : 9780198534037.

[67] R. WILLOUGHBY. « Collection of articles honouring Alston S. Householder ». *ACM* 18 1975, p. 3–58.

A Méthodes particulières

A.1 Factorisation de cholesky et orthogonalisation d'un sous-espace

On considère un ensemble de m vecteurs de taille n $\mathbf{v}_i$ indépendants, stockés dans les colonnes d'une matrice rectangulaire $\mathbf{V}$ de taille (n,m). Ils génèrent un sous-espace de l'espace de départ. Dans un certain nombre d'applications, il est utile de construire à partir de ces vecteurs, une nouvelle base orthogonale $\mathbf{U}$. Pour ce faire, on utilise généralement le procédé d'orthogonalisation de Gram-Schmidt. On cherche donc $\mathbf{U}$ engendrant le même sous-espace que $\mathbf{V}$ et tel que $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$.

Pour ce faire, on peut utiliser une technique basée sur la factorisation précédente : si on construit la matrice $\mathbf{M} = \mathbf{V}^\top\mathbf{V}$, pleine et de taille réduite (n,n) et sa factorisation $\mathbf{M} = \mathbf{L}\mathbf{L}^\top$, alors :

$$\mathbf{V}^\top\mathbf{V} = \mathbf{L}\mathbf{L}^\top \Rightarrow \mathbf{L}^{-1}\mathbf{V}^\top\mathbf{V}\mathbf{L}^{-\top} = \mathbf{I} \Rightarrow \mathbf{U}^\top = \mathbf{L}^{-1}\mathbf{V}^\top \tag{A.1}$$

Il faut donc procéder à n résolutions du système triangulaire de taille n :

$$\mathbf{L}\mathbf{U}^\top = \mathbf{V}^\top \tag{A.2}$$

On réalise ainsi une orthonormalisation des vecteurs de départ.

A.2 Technique d'accélération d'Aitken

Cette technique [1] se sert des itérés précédents pour construire une *meilleure* nouvelle approximation. Si on considère une série scalaire η_n qui tend vers $\bar{\eta}$, construite par un algorithme du premier ordre avec un taux de convergence κ ; en supposant les itérés η_n, η_{n-1}, η_{n-2} calculés, l'approximation au premier ordre s'écrit :

$$\begin{aligned} \eta_n - \bar{\eta} &\approx \kappa(\eta_{n-1} - \bar{\eta}) \\ \eta_{n-1} - \bar{\eta} &\approx \kappa(\eta_{n-2} - \bar{\eta}) \end{aligned} \tag{A.3}$$

Ces deux équations permettent de calculer une approximation de $\bar{\eta}$, qui va servir pour construire le nouvel itéré η'_n, et κ, qui permet donc d'estimer le taux de convergence en cours de route. On détermine ainsi l'expression du nouvel itéré :

$$\eta'_n = \eta_n - \frac{(\eta_{n-1} - \eta_n)^2}{\eta_n - 2\eta_{n-1} + \eta_n} \tag{A.4}$$

B Bornage par les valeurs propres élémentaires

On se place ici dans le cas de figure où les problèmes sont issus d'une analyse par éléments finis. L'élément fini courant est noté e et les quantités élémentaires associées porteront aussi l'indice e.

B.1 Système aux valeurs propres généralisé

Le problème est de la forme $\mathbf{Kx} = \lambda\mathbf{Mx}$, et on défini les matrices diagonales par bloc contenant les quantités élémentaires de la façon suivante :

$$\bar{\mathbf{K}} = \begin{bmatrix} \mathbf{K}_e & & \\ & \mathbf{K}_{e'} & \\ & & \ddots \end{bmatrix} \quad \text{et} \quad \bar{\mathbf{M}} = \begin{bmatrix} \mathbf{M}_e & & \\ & \mathbf{M}_{e'} & \\ & & \ddots \end{bmatrix} \tag{B.1}$$

L'assemblage consiste à utiliser formellement une matrice d'assemblage $\mathbf{P}$ booléenne telle que :

$$\mathbf{K} = \mathbf{P}\bar{\mathbf{K}}\mathbf{P}^\top \tag{B.2}$$

Comme $\mathbf{M}$ est issue de la même discrétisation, la même matrice d'assemblage est utilisée :

$$\mathbf{M} = \mathbf{P}\bar{\mathbf{M}}\mathbf{P}^\top \tag{B.3}$$

Considérons le système, de taille supérieure à n :

$$\bar{\mathbf{K}}\bar{\mathbf{x}} = \bar{\lambda}\bar{\mathbf{M}}\bar{\mathbf{x}} \tag{B.4}$$

Il englobe le système initial car il suffit de l'écrire dans le sous-espace où $\bar{\mathbf{x}}$ est issu du *désassemblage* d'un vecteur $\mathbf{x}$ de raille n : $\bar{\mathbf{x}} = \mathbf{P}^\top\mathbf{x}$[1]. Avec le quotient de Rayleigh :

$$\bar{\mathrm{R}}(\bar{\mathbf{x}}) = \frac{\bar{\mathbf{x}}^\top\bar{\mathbf{K}}\bar{\mathbf{x}}}{\bar{\mathbf{x}}^\top\bar{\mathbf{M}}\bar{\mathbf{x}}} \tag{B.5}$$

on sait que :

$$\bar{\lambda}_{\max} = \max_{\bar{\mathbf{x}}\neq\mathbf{0}} \bar{\mathrm{R}}(\bar{\mathbf{x}}) \tag{B.6}$$

donc :

$$\bar{\lambda}_{\max} \underset{\bar{x}\neq\mathbf{0}}{\geqslant} \frac{\bar{\mathbf{x}}^\top\bar{\mathbf{K}}\bar{\mathbf{x}}}{\bar{\mathbf{x}}^\top\bar{\mathbf{M}}\bar{\mathbf{x}}} \tag{B.7}$$

Au vu du problème *désassemblé*, $\bar{\lambda}_{\max}$ est la valeur maximale des valeurs propres des éléments pris séparément et la relation (B.7) est en particulier vraie pour $\bar{\mathbf{x}} = \mathbf{P}^\top\mathbf{x}$ donc :

$$\bar{\lambda}_{\max} \underset{\bar{x}\neq\mathbf{0}}{\geqslant} \frac{\bar{\mathbf{x}}^\top\mathbf{P}\bar{\mathbf{K}}\mathbf{P}^\top\bar{\mathbf{x}}}{\bar{\mathbf{x}}^\top\mathbf{P}\bar{\mathbf{M}}\mathbf{P}^\top\bar{\mathbf{x}}} = \frac{\mathbf{x}^\top\mathbf{K}\mathbf{x}}{\mathbf{x}^\top\mathbf{M}\mathbf{x}} = \mathrm{R}(\mathbf{x}) \tag{B.8}$$

1. Les forces généralisées, elles, s'assembleraient de la façon suivante : $\mathbf{f} = \mathbf{P}\bar{\mathbf{f}}$.

donc :

$$\bar{\lambda}_{\max} \geqslant \max_{\mathbf{x}\neq\mathbf{0}} \mathrm{R}(\mathbf{x}) = \lambda_{\max} \tag{B.9}$$

B.2 Système aux valeurs propres

Cette fois-ci, le problème est de la forme $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$. Concernant les contributions élémentaire, ce problème n'est plus équivalent au problème précédent ; en particulier, si on écrit $\mathbf{A} = \mathbf{M}^{-1}\mathbf{K}$ et comme $\mathbf{M}^{-1}$ est pleine, on n'a plus la connectivité éléments finis dans $\mathbf{A}$, et si on écrit $\mathbf{M} = \mathbf{I}$, $\mathbf{M}$ n'est pas l'assemblage des matrices élémentaires identité. Le résultat précédent ne s'applique donc plus ici. Par contre, on a toujours :

$$\lambda_{\max} = \max_{x\neq\mathbf{0}} \mathrm{R}(\mathbf{x}) \tag{B.10}$$

où :

$$\mathrm{R}(\mathbf{x}) = \frac{\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\mathbf{x}^\top\mathbf{x}} \tag{B.11}$$

donc :

$$\lambda_{\max} \underset{x\neq\mathbf{0}}{\geqslant} \frac{\mathbf{x}^\top\mathbf{A}\mathbf{x}}{\mathbf{x}^\top\mathbf{x}} \tag{B.12}$$

Prenons alors comme vecteur $\mathbf{x}$ particulier un vecteur nul partout sauf sur les nœuds d'un élément quelconque e et notons $\mathbf{x}_e$ sa restriction sur les degrés de liberté de cet élément (vecteur de petite taille, donc). On a $\mathbf{x}_e^\top\mathbf{x}_e = \mathbf{x}^\top\mathbf{x}$ et les contributions élémentaires $\mathbf{x}^\top\mathbf{A}\mathbf{x} = \mathbf{x}^\top\mathbf{P}\bar{\mathbf{A}}\mathbf{P}^\top\mathbf{x} = \mathbf{x}_e^\top\mathbf{A}_e\mathbf{x}_e + a$ où, si toutes les matrices élémentaires $\mathbf{A}_e$ sont positives, $a \geqslant 0$, alors :

$$\lambda_{\max} \underset{\mathbf{x}_e\neq\mathbf{0}}{\geqslant} \frac{\mathbf{x}_e^\top\mathbf{A}_e\mathbf{x}_e + a}{\mathbf{x}_e^\top\mathbf{x}_e} \geqslant \frac{\mathbf{x}_e^\top\mathbf{A}_e\mathbf{x}_e}{\mathbf{x}_e^\top\mathbf{x}_e} \tag{B.13}$$

donc :

$$\lambda_{\max} \geqslant \bar{\lambda}_{\max} \tag{B.14}$$

C'est le résultat contraire au précédent, concernant les systèmes aux valeurs propres généralisés.

C Repères biographiques

Alexander Craig Aitken

Aitken left the Otago Boys' High School in Dunedin in 1913 having won a scholarship to Otago University. He began to study languages and mathematics with the intention of becoming a school teacher but his university career was interrupted by World War I. Back to New Zealand in 1917, Aitken followed his original intention and became a school teacher. At his old school Otago Boys' High School. Encouraged by the new professor of mathematics at Otago University, Aitken came to Scotland in 1923 and studied for a Ph.D. at Edinburgh under Whittaker. In 1925 he was appointed to Edinburgh where he spent the rest of his life.

— Born : 1 April 1895 in Dunedin, New Zealand.
— Died : 3 Nov 1967 in Edinburgh, Scotland.

André-Louis Cholesky

Cholesky was a French military officer involved in geodesy and surveying in Crete and North Africa just before World War I. He entered l'École Polytechnique at the age of 20 and was attached to the Geodesic Section of the Geographic Service, in June 1905. That was the period when the revision of the French triangulation had just been decided to be used as the base of a new cadastral triangulation. Cholesky solved the problem of the adjustment of the grid with the method now named after him to compute solutions to the normal equations for the least squares data fitting problem.

— Born : 15 Oct 1875 in Montguyon (Charente-Inférieure).
— Died : 31 Aug 1918.

Johann Carl Friedrich Gauss

Gaussian elimination, which first appeared in the text Nine Chapters of the Mathematical Art written in 200 BC, was used by Gauss in his work which studied the orbit of the asteroid Pallas. Using observations of Pallas taken between 1803 and 1809, Gauss obtained a system of six linear equations in six unknowns. Gauss gave a systematic method for solving such equations which is precisely Gaussian elimination on the coefficient matrix.

— Born : 30 April 1777 in Brunswick, Duchy of Brunswick (now Germany).
— Died : 23 Feb 1855 in Göttingen, Hanover (now Germany).

Issai Schur

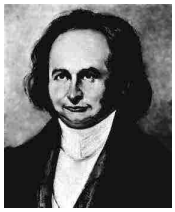
In 1894, Schur entered the University of Berlin to read mathematics and physics. Schur made major steps forward in representation theory of groups, in collaboration with Frobenius, one of his teacher. In 1916, in Berlin, he built his famous school and spent most of the rest of his life there. Schur's own impressive contributions were extended by his students in a number of different directions. They worked on topics such as soluble groups, combinatorics, and matrix theory.

— Born : 10 Jan 1875 in Mogilyov, Mogilyov province, Belarus.
— Died : 10 Jan 1941 in Jerusalem, Palestine.

Joseph-Louis Lagrange

Joseph-Louis Lagrange is usually considered to be a French mathematician, but the Italian Encyclopedia refers to him as an Italian mathematician. They certainly have some justification in this claim since Lagrange was born in Turin and baptised in the name of Giuseppe Lodovico Lagrangia. The papers by Lagrange cover a variety of topics : beautiful results on the calculus of variations, work on the calculus of probabilities... In a work on the foundations of dynamics, Lagrange based his development on the principle of least action and on kinetic energy. The 'Mécanique analytique' which Lagrange had written in Berlin, was published in 1788. It summarized all the work done in the field of mechanics since the time of Newton and is notable for its use of the theory of differential equations. With this work Lagrange transformed mechanics into a branch of mathematical analysis. « Lagrange, in one of the later years of his life, imagined that he had overcome the difficulty (of the parallel axiom). He went so far as to write a paper, which he took with him to the Institute, and began to read it. But in the first paragraph something struck him which he had not observed : he muttered : "Il faut que j'y songe encore", and put the paper in his pocket. » A De Morgan *Budget of Paradoxes.*

— Born : 25 Jan 1736 in Turin, Sardinia-Piedmont (now Italy).
— Died : 10 April 1813 in Paris, France.



Karl Gustav Jacob Jacobi

Jacobi published three treatises on determinants in 1841. These were important in that for the first time the definition of the determinant was made in an algorithmic way and the entries in the determinant were not specified so his results applied equally well to cases were the entries were numbers or to where they were functions. These three papers by Jacobi made the idea of a determinant widely known.

— Born : 10 Dec 1804 in Potsdam, Prussia (now Germany).
— Died : 18 Feb 1851 in Berlin, Germany.

Ludwig Philipp von Seidel

Philipp Seidel entered the University of Berlin in 1840 and studied under Dirichlet and Encke. He moved to Kœnigsberg where he studied under Bessel, Jacobi and Franz Neumann. Seidel obtained his doctorate from Munich in 1846 and he went on to become professor there. Seidel wrote on dioptics and mathematical analysis. His work on lens identified mathematically five coefficients describing the aberration of a lens, now called 'Seidel sums'. These Seidel sums correspond to spherical aberration, coma, astigmatism, Petzval curvature and distortion. He also introduced the concept of nonuniform convergence and applied probability to astronomy.

— Born : 24 Oct 1821 in Zweibrœcken, Germany.
— Died : 13 Aug 1896 in Munich, Germany.

James Hardy Wilkinson

Wilkinson won a Foundation Scholarship to Sir Joseph Williamson's Mathematical School, Rochester at the age of 11. In 1940, Wilkinson began war work which involved mathematical and numerical work on ballistics. Wilkinson continued work becoming more involved in writing many high quality papers on numerical analysis, particularly numerical linear algebra where he developed backward error analysis methods. He worked on numerical methods for solving systems of linear equations and eigenvalue problems. As well as the large numbers of papers on his theoretical work, Wilkinson developed computer software, working on the production of libraries of numerical routines. The NAG (Numerical Algorithms Group) began work in 1970 and much of the linear algebra routines were due to Wilkinson.

— Born : 27 Sept 1919 in Strood, Kent, England.
— Died : 5 Oct 1986 in London, England.

John William Strutt Lord Rayleigh

His first paper in 1865 was on Maxwell's electromagnetic theory. He worked on propagation of sound and, while on an excursion to Egypt taken for health reasons, Strutt wrote Treatise on Sound (1870-1). In 1879, he wrote a paper on travelling waves, this theory has now developed into the theory of solitons. His theory of scattering (1871) was the first correct explanation of why the sky is blue.

— Born : 12 Nov 1842 in Langford Grove (near Maldon), Essex, England.
— Died : 30 June 1919 in Terling Place, Witham, Essex, England.

Helmut Wielandt

Wielandt entered the University of Berlin in 1929 and there he studied mathematics, physics and philosophy. There, he was greatly influenced by Schmidt and Schur. It was on the topic of permutation groups that Wielandt wrote his doctoral dissertation and he was awarded a doctorate in 1935. At the end of World War II Wielandt was appointed associate professor at the University of Mainz, and in 1951, Ordinary Professor at the University of Tœbingen. For 20 years beginning in 1952, Wielandt was managing editor of Mathematische Zeitschrift.

— Born : 19 Dec 1910 in Niedereggenen, Lœrrach, Germany.
— Died : 1984.

Alston Scott Householder

Householder then taught mathematics in a number of different places. He was awarded a Ph.D. by the University of Chicago in 1947 for a thesis on the calculus of variations. However his interests were moving towards applications of mathematics, particularly applications of mathematics to biology. In 1946, after the end of the war, Householder joined the Mathematics Division of Oak Ridge National Laboratory. Here he changed topic, moving into numerical analysis which was increasing in importance due to the advances in computers. He started publishing on this new topic with Some numerical methods for solving systems of linear equations which appeared in 1950. In 1964, Householder published one of his most important books : *The theory of matrices in numerical analysis*.

— Born : 5 May 1904 in Rockford, Illinois, USA.
— Died : 4 July 1993 in Malibu, California, USA.

Cornelius Lanczos

In 1938 at Purdue, Lanczos published his first work in numerical analysis. Two years later he published a matrix method of calculating Fourier coefficients which, over 25 years later, was recognised as the Fast Fourier Transform algorithm. In 1946, with Boeing, he worked on applications of mathematics to aircraft design and was able to develop new numerical methods to solve the problems. In 1949 he moved to the Institute for Numerical Analysis of the National Bureau of Standards in Los Angeles. Here he worked on developing digital computers and was able to produce versions of the numerical methods he had developed earlier to program on the digital computers.

— Born : 2 Feb 1893 in Székesfehérvr, Hungary.
— Died : 25 June 1974 in Budapest, Hungary.

Joseph Raphson

Joseph Raphson's life can only be deduced from a number of pointers. It is through the University of Cambridge records that we know that Raphson attended Jesus College Cambridge and graduated with an M.A. in 1692. Rather remarkably Raphson was made a member of the Royal Society in 1691, the year before he graduated. His election to that Society was on the strength of his book Analysis aequationum universalis which was published in 1690 contained the Newton method for approximating the roots of an equation.

Raphson's ideas of space and philosophy were based on Cabalist ideas. The Cabala was a Jewish mysticism which was influential from the 12th century on and for which several basic doctrines were strong influences on Raphson's philosophical thinking. The doctrines included the withdrawal of the divine light, thereby creating primordial space, the sinking of luminous particles into matter and a *cosmic restoration*.

— Born : 1648 in Middlesex, England.
— Died : 1715.

Carle David Tolmé Runge

As a German mathematician, physicist, and spectroscopist, he is known to be the co-developer and co-eponym of the Runge-Kutta method in the field of numerical analysis. After taking his secondary school teachers examinations he returned to Berlin where he was influenced by Kronecker. Runge then worked on a procedure for the numerical solution of algebraic equations in which the roots were expressed as infinite series of rational functions of the coefficients.

— Born : 30 Aug 1856 in Bremen, Germany.
— Died : 3 Jan 1927 in Göttingen, Germany.

Sir Isaac Newton

Isaac Newton is perhaps the best known renaissance scientist today, living between 1642 and 1727. We think of gravity, celestial mechanics, and calculus when we think of him. He certainly did develop the calculus by building upon the ideas of Fermat and Barrow (the person whose chair he took when he went to Cambridge). But he was not alone in developing calculus. And his focus was really one of mechanics - how do bodies move. His focus was always on motion and is reflected in the terminology he chose for calculus - what we call *derivatives*, he called *fluxions*. And his integrals were simply *inverse fluxions*. You can see how *flux* was his focus. He also developed the *dot* notation for calculus - one dot meant a first derivative, two dots a second, and so forth. Today this is used, but only with respect to time derivatives. A general derivative still needs to specify the variable of differentiation and more often than not, time derivatives are done in this more general manner.

As indicated earlier, Newton and his followers argued vehemently with Huygens and his followers over the nature of light. Newton subscribed to a *corpuscular* theory, where he envisioned light as small compact bodies of energy. Huygens focussed on the wave like nature and developed that theory. The diffraction properties of light were so obvious, that Huygens school eventually won out, and the wave theory of light ruled science for the next three centuries.

— Born : 4 Jan 1643 in Woolsthorpe, Lincolnshire, England.
— Died : 31 March 1727 in London, England.

Leonhard Euler

The publication of many articles and his book Mechanica (1736-37), which extensively presented Newtonian dynamics in the form of mathematical analysis for the first time, started Euler on the way to major mathematical work. He integrated Leibniz's differential calculus and Newton's method of fluxions into mathematical analysis. In number theory he stated the prime number theorem and the law of biquadratic reciprocity. Euler made large bounds in modern analytic geometry and trigonometry. He was the most prolific writer of mathematics of all time. His complete works contains 886 books and papers.

— Born : 15 April 1707 in Basel, Switzerland.
— Died : 18 Sept 1783 in St Petersburg, Russia.

Martin Wilheim Kutta

Kutta was professor at the RWTH Aachen from 1910 to 1911. He then became professor at Stuttgart and remained there until he retired in 1935. In 1901, he had co-developed the Runge-Kutta method, used to solve ordinary differential equations numerically. He is also remembered for the Zhukovsky-Kutta aerofoil, the Kutta-Joukowski theorem and the Kutta condition in aerodynamics.

— Born : 3 Nov 1867 in Pitschen, Upper Silesia (now Byczyna), Poland.
— Died : 25 Dec 1944 in Frstenfeldbruck, Germany.

Index

A

accélération
- d'Aitken 26, 87
- méthode de l'accélération linéaire 67, 72
- méthode de l'accélération moyenne 67
- schéma de Newmark 67

Aitken 91
- technique d'accélération 26, 87

B

Broyden
- Méthode BFGS 51

C

Cauchy
- problème de 53, 63

Cholesky 91
- factorisation de . . . 18, 19, 30, 33, 87

coût
- CPU 15
- d'un algorithme . . 14–16, 18, 21, 42, 63, 65
- d'une méthode 24
- fonction 50
- fonction (optimisation) 48

complexité 15, 18, 19, 41
conditionnement 12, 31, 34
convergence 24, 26
- asymptotique 24
- cubique 39
- gradient conjugué préconditionné 30
- linéaire 42
- méthode de Jacobi 36
- méthode de Lanczos 44
- méthode des itérations inverses . . 39
- méthode des puissances 37
- méthode du gradient 29
- quadratique 75
- schéma d'Euler explicite 54
- taux de 25, 87
- valeurs propres multiples 37

Cramer
- méthode de 14

Crout
- factorisation 18
- factorisation de 19–21

Cuthill-MacKee 14

E

éléments finis . . 7, 11, 12, 19, 47, 89, 90
- code 39
- conditionnement 13
- conditions aux limites 20
- convergence 25
- discrétisation 14, 33
- problème de mémoire 51
- super-élément 19
- système différentiel du premier ordre 64

équation
- caractéristique 34, 68
- d'équilibre 78
- d'état 75
- d'admissibilité cinématique 47
- d'admissibilité statique 47
- d'Euler 21, 22
- de comportement 78
- de la dynamique 79
- différentielle 79
- différentielle non-linéaire 74
- du mouvement 79

Euler 95
- équation d' 21, 22
- schéma d'
  - explicite, 54, 56, 58
  - implicite, 58, 61, 74

F

factorisation
- de Cholesky 18, 19, 30, 33, 87
- de Crout 18–21
- de Gauss 19, 21
- LDL 19, 39
- $LDM^\top$ 20
- LU 17–19
- QR 40, 41

G

Gauss 91
élimination de 16
algorithme de Gauss-Seidel 26
factorisation de 19, 21
méthode de Gauss-Seidel 25, 26

H

Hessenberg
forme de 42
Householder 93
algorithme de Householder-Businger-Golub 41
factorisation QR 41
méthode de 34
transformation de 40, 43

J

Jacobi 92
algorithme de 35
méthode de 25, 27, 34, 37
convergence, 36

K

Kahan
théorème de 26
Kutta 95
Kutta-Joukowski 95
méthode de Runge-Kutta 62–64, 66, 74
Runge-Kutta 94
Zhukovsky-Kutta 95

L

Lagrange 92
multiplicateurs de 21, 27
Lanczos 94
itérations de 46
méthode de 34, 44, 45
Leibniz 95
linéaire
convergence 42
linéaire(s)
équation(s) 56, 65
combinaison(s) 16, 24, 37
méthode des accélérations ... 67, 72
récurrence(s) 24
relation(s) 22
système(s) 11, 25, 34, 41, 49, 65, 66

M

matrice 11
bande 21
booléenne 21
condensée 17
d'amortissement 33, 66
d'assemblage 89
d'itération 26, 68
décalée 39
de capacité 64
de conduction 64
de masse 65, 66
de raideur 22
de rigidité 20, 48, 66
tangente, 75
de rotation 35
jacobienne 48–50, 75
pleine 18
semi-définie 23
sous-matrice 17
symétrique 19
symétrique définie positive 21
triangulaire supérieure 17
valeurs propres 28, 33

N

Newmark
méthode de 66, 69, 72
schéma de 67, 69
stabilité 68
Newton 92, 95
fluxions 95
loi de 78
méthode de 47, 48, 75
modèle de 80
Newton-Raphson 49, 75
quasi-newton 50
norme 11–13, 24, 30, 31, 37, 38, 46
infinie 37
matricielle 11
normer 37, 45

O

orthogonalisation
d'un sous-espace 18, 87
de Gram-Schmidt 87
ré-orthogonalisation de grand problème 45

P

pivot 23
Crout 21
facrotisation LU 17
partiel 17
stratégie 17
total 17
préconditionnement 30
gradient conjugué préconditionné 31, 51
préconditionneur 30

R

Raphson 94
Newton-Raphson 49, 75
Rayleigh 93
amotissement de 68
itérations inverses de 39
quotient de 34, 89
Runge 94
méthode de Runge-Kutta 62–64, 66, 74, 95
Runge-Kutta 94

S

Schur 91, 93
complément de 17
condensation de 19
Seidel 92
algorithme de Gauss-Seidel 26
méthode de Gauss-Seidel 25, 26
Southwell 26
système
aux valeurs propres . . 33, 34, 89, 90
d'exploitation 15
différentiel
du premier ordre, 79
non régulier, 79
dynamique 79
linéaire 11, 14, 34, 41, 49
creux, 14
triangulaire, 16, 18
non-linéaire 25, 47

T

trapèze(s)
méthode des 57–59, 73, 79
schéma des 58

W

Wielandt 93
méthode de 39
Wilkinson 93
test de 45