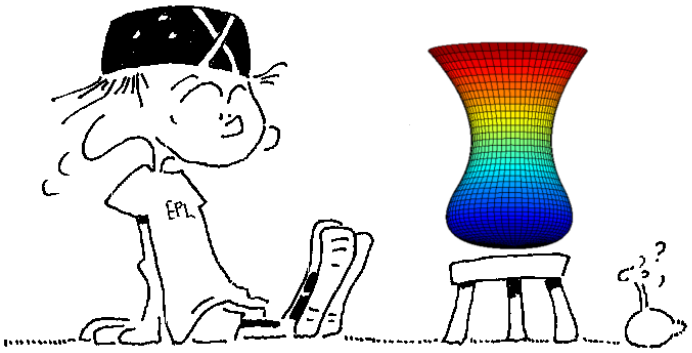
Ecole Polytechnique de Louvain



MATHEMATIQUES ET METHODES NUMERIQUES

...ou les aspects facétieux du calcul sur un ordinateur

V. Legat

Notes pour les cours LEPL1104
Année académique 2023-2024 (version 8.5 15-01-2024)

python™

Ce texte est toujours une version provisoire. Malgré tout le soin apporté à sa rédaction, il est possible que quelques erreurs soient toujours présentes dans le texte. Tout commentaire, critique ou suggestion de votre part, est évidemment le bienvenu. Il vous est possible de m'envoyer vos commentaires directement par courrier électronique à l'adresse suivante : `vincent.legat@uclouvain.be`

Les éventuels errata du texte seront disponibles sur le site Web du cours.

J'adresse mes remerciements à tous les étudiants et étudiantes qui m'ont aidé à améliorer de manière constante ce texte. Grâce à leur oeil impitoyable, les nombreuses coquilles, fautes d'orthographe et imprécisions ont été progressivement éliminées...

Comment résoudre numériquement
une équation différentielle ?

Avant-propos

Les modèles mathématiques des sciences et des techniques se présentent très souvent sous la forme de systèmes d'équations différentielles qui lient des fonctions inconnues à leurs dérivées par rapport au temps et à l'espace. Des conditions aux limites sont en général requises pour compléter le modèle. Par exemple, la conservation de la quantité de mouvement est utilisée pour prédire le mouvement d'un solide. Pour tirer une information pertinente sur le processus physique modélisé, il est essentiel de trouver la fonction satisfaisant l'équation différentielle et les conditions aux limites. Il s'agit donc de *résoudre un problème différentiel aux conditions aux limites*.

La solution d'un problème différentiel peut parfois être obtenue analytiquement. Toutefois, dans la plupart des cas, cela n'est pas possible et l'unique possibilité est de calculer une fonction approchée au moyen de méthodes numériques. L'idée de base consiste à ne rechercher que la valeur des fonctions inconnues en un grand nombre de points : il s'agit de la *discrétisation*. Au lieu de résoudre un problème différentiel ou *problème continu*, on résout un grand système algébrique qu'on appelle le *problème discret*. Il est ici essentiel de comprendre la différence entre une solution analytique exacte et une solution numérique approchée. Il est aussi essentiel de comprendre comment une solution approchée peut être interprétée comme la projection orthogonale d'une fonction dans un espace discret de taille finie.

Comment résoudre analytiquement une équation différentielle ? Des techniques analytiques de résolution des équations différentielles ordinaires linéaires existent et permettent d'en obtenir une solution exacte. L'analyse de valeurs propres pour étudier la stabilité d'un système linéaire est un outil essentiel pour l'ingénieur et fait l'objet de notre étude.

Comment obtenir un problème discret ? C'est le rôle d'une méthode numérique. Il ne s'agit pas d'apprendre à écrire des logiciels numériques mais de préparer à devenir ultérieurement des utilisateurs avertis des logiciels de simulation numérique. L'analyse et l'estimation des erreurs des différentes méthodes accompagnent la présentation des méthodes numériques. On examinera aussi les problèmes liés aux erreurs d'arrondi, à la stabilité des méthodes numériques, au coût des algorithmes ainsi déduits et à la capacité à traiter des problèmes difficiles. On utilisera *Matlab* pour les séances d'exercices et dans le cadre du projet.

Pour obtenir une méthode numérique pour la résolution d'une équation différentielle ordinaire, plusieurs étapes sont nécessaires :

- Tout d'abord, nous verrons comment il est possible d'approcher une fonction par une combinaison linéaire de fonctions connues. L'approximation de la fonction sera complètement décrite par l'ensemble des coefficients. Il s'agit des problèmes

d'interpolation et d'approximation.

- Ensuite, nous présenterons des méthodes permettant d'effectuer des intégrations et des dérivations de manière numérique.
- Nous présenterons une manière de résoudre numériquement une équation différentielle ordinaire, en utilisant les méthodes de l'interpolation-approximation et celles de l'intégration numérique.
- Il sera alors tentant d'aborder brièvement le cas plus réaliste de systèmes d'équations différentielles ordinaires non-linéaires. Il nous faudra alors résoudre des systèmes algébriques non-linéaires.
- Finalement, nous résoudrons numériquement des équations aux dérivées partielles.

A chaque étape, l'approche numérique sera présentée, en parallèle à une approche analytique classique pour les problèmes d'intégrations, les résolutions d'équations différentielles ordinaires et l'analyse de la stabilité par le biais du calcul des valeurs propres. Pour chaque type de problème, l'étudiant devra être à même, face à un problème donné, de choisir entre une voie analytique et une voie numérique. Parmi les méthodes numériques, il devra pouvoir choisir celle qui convient le mieux. Il devra en connaître les conditions de convergence, les caractéristiques de coût, de complexité et de stabilité. Il devra être capable d'utiliser ou de programmer des méthodes simples avec le langage `python` .

La modélisation mathématique et la simulation informatique sont des outils essentiels pour l'ingénieur. Des prédictions sont obtenues par la résolution numérique d'un système d'équations différentielles qui modélisent, par exemple, le mouvement d'un véhicule. Les résultats ne sont pas toujours fiables, mais parfois ils fournissent de bonnes prédictions. La fiabilité est affectée par quatre sources distinctes d'erreurs.

- Les données initiales ne sont connues qu'approximativement.
- Le système d'équations différentielles du modèle ne décrit que très approximativement la réalité de la nature.
- La solution discrète calculée par l'ordinateur n'est qu'une approximation numérique de la solution du problème continu.
- Et finalement, l'arithmétique en virgule flottante utilisée dans un ordinateur ne fournira elle-même qu'une approximation de la solution discrète que l'on obtiendrait avec un calculateur parfait.

La somme de ces quatre approximations (c'est-à-dire, le bruit sur les données, l'erreur de modélisation, l'erreur de discrétisation et les erreurs d'arrondi) forme l'erreur de la prédiction qui est parfois excessive. Il est dès lors essentiel d'estimer l'erreur totale en estimant individuellement chaque terme afin de tenter d'améliorer la précision où cela est possible et nécessaire.

Evidemment, les méthodes numériques ne servent pas qu'à résoudre numériquement des équations différentielles ordinaires... Il ne s'agit que d'un exemple d'application particulièrement remarquable qui formera l'objectif final de notre propos. Mais, toutes les briques que nous avons introduites ont également bien d'autres applications.

Références

- Charles F. Van Loan, *Introduction to Scientific Computing,* Second Edition, Prentice Hall, Upper Saddle River, ISBN 0-13949157-0 (1999).
- Jacques Rappaz, Marco Picasso, *Introduction à l'analyse numérique,* Presses polytechniques et universitaires romandes, Lausanne, ISBN 2-88074363-X (2000).
- André Fortin, *Analyse numérique pour ingénieurs,* Seconde Edition, Presses internationales polytechniques, Montréal, ISBN 2-55300936-4 (2001).
- William L. Briggs, Van Emden Henson, Steve F. McCormick, *A Multigrid Tutorial,* Second Edition, SIAM, Philadelphia, ISBN 0-89871462-1 (2000).
- Brigitte Lucquin, Olivier Pironneau, *Introduction to Scientific Computing,* John Wiley & Sons, New York, ISBN 0-47197266-X (1998).
- Alfio Quarteroni, Fausto Saleri, *Scientific Computing with MATLAB,* Springer-Verlag, Berlin, ISBN 3-35044363-0 (2003).
- Desmond J. Higham, Nicholas J. Higham *Matlab Guide,* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, ISBN 0-89871469-9 (2000).
- Michael T. Heath *Scientific Computing : an Introduction Survey,* McGraw Hill, New-York, ISBN 0-07-115336-5 (1997).
- K. E. Atkinson, *An Introduction to Numerical Analysis,* Second Edition, John Wiley & Sons, New York (1989).
- S. D. Conte, C. de Boor, *Elementary Numerical Analysis, An Algorithmic Approach,* Third Edition, McGraw-Hill Book Company, New York (1980).
- B.M. Irons, N.G. Shrive, *Numerical Methods in Engineering and Applied Sciences : numbers are fun,* Second Edition, John Wiley and Sons (1987).
- John H. Mathews, *Numerical Methods for Mathematics, Science and Engineering,* Second Edition, Prentice Hall, Englewood Cliffs, ISBN 0-13624990-6 (1992).
- W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery *Numerical Recipes in C: The Art of Scientific Computing,* Second Edition, Cambridge University Press, Cambridge (1994).

Table des matières

Fun, frustrations and tricks of the trade
(Titre du premier chapitre du livre de B.M. Irons)

Chapitre 1

Comment approximer une fonction ?

La procédure de calcul utilisée dans les logiciels informatiques pour évaluer les fonctions telles que $\cos(x)$, $\sin(x)$ *ou* $\exp(x)$ *fait appel à l'approximation polynomiale. De même, la modélisation de courbes et de surfaces en infographie ou en conception assistée par ordinateur est basée sur l'approximation polynomiale.*

L'état de l'art utilise des fonctions rationnelles sous la forme de quotients de polynômes. Typiquement, on utilise des courbes Non-Uniform Rational B-Splines (NURBS) ou des courbes de Bézier pour la représentation et la modélisation d'objets complexes. L'objet de ce chapitre est de fournir une brève introduction au problème de l'approximation et de l'interpolation d'une fonction ou de données discrètes.

Soit une fonction quelconque $u(x)$ qui n'est pas vraiment disponible ou qui est difficile à manipuler algébriquement. Cette fonction appartient à un espace vectoriel $\mathcal{U}$ qui contient des fonctions définies sur un domaine $[a, b]$: il pourrait, ainsi, s'agir de l'espace des fonctions continues. Maintenant, on souhaite remplacer la fonction $u(x)$ par une approximation $u^h(x)$ qui appartient à un sous-espace des fonctions disponibles $\mathcal{U}^h \subset \mathcal{U}$. A titre d'exemple, on pourrait souhaiter approximer la fonction cosinus par une expression polynomiale.

Pour simplifier les choses, décidons de rechercher une approximation sous la forme d'une combinaison linéaire :

$$u(x) \approx u^h(x) = \sum_{j=0}^{n} a_j \ \phi_j(x) \tag{1.1}$$

où a_j sont des *paramètres* inconnus, tandis que ϕ_j sont des *fonctions de base* spécifiées a priori et appartenant à l'espace $\mathcal{U}$. Les fonctions de base sont choisies afin qu'aucune d'entre elles ne puisse être obtenue par combinaison linéaire des autres. Il y a donc $n + 1$ degrés de liberté pour définir une approximation particulière appartenant au sous-espace d'approximation ou espace discret $\mathcal{U}^h \subset \mathcal{U}$. Ces fonctions constituent donc une

base du sous-espace $\mathcal{U}^h \subset \mathcal{U}$. On approche donc u appartenant à un espace de dimension infinie, par une approximation u^h appartenant à un espace de dimension finie $\mathcal{U}^h$. L'approximation peut être entièrement caractérisée par les valeurs des paramètres a_j.

La dimension de $\mathcal{U}^h$ est clairement $n+1$ et les fonctions de base forment une base de cet espace dont tous les éléments sont obtenus par une combinaison linéaire unique des éléments de cette base. Le problème de l'approximation consiste donc à trouver ces $n+1$ paramètres afin que la fonction u^h soit la plus proche possible de u selon un critère qu'il faudra préciser. On observe donc que la construction d'une approximation sera caractérisée par deux aspects

- le choix des fonctions de base,
- la manière de déterminer les paramètres inconnus.

Un premier exemple d'approximation est le polynôme construit avec le développement de Taylor autour d'un point X_0. Sur base de la valeur de $u(x)$ et des dérivées successives en ce point, on approxime une fonction par un polynôme au voisinage de X_0. Toutefois, dans les problèmes pratiques, on ne dispose pas de la fonction u, et encore plus rarement de ses dérivées !

Si la fonction u n'est pas disponible, il faut au moins une information partielle à son sujet. Par exemple, on sait que les valeurs aux abscisses $a \leq X_0 < X_1 < X_2 < \ldots < X_m \leq b$ sont connues et données par

$$u(X_i) = U_i, \quad i = 0, 1, 2, \ldots, m.$$

Dans de nombreuses applications, comme la prise de mesures expérimentales ponctuelles, les points (X_i, U_i)[1] constituent la seule information connue sur la fonction $u(x)$. Intuitivement, on choisit des abscisses équidistantes pour effectuer une campagne de mesures, mais il ne s'agit pas d'office du meilleur choix. Il s'agit maintenant de trouver une approximation u^h d'une fonction u uniquement sur la base de l'ensemble de points $(X_i, u(X_i))$. Deux cas de figure peuvent se présenter:

- La fonction $u^h(x)$ passe *exactement* par les points. Elle vérifie donc les relations

$$u^h(X_i) = U_i.$$

  Dans ce cas, on dira de $u^h(x)$ qu'elle *interpole* la fonction $u(x)$ aux abscisses X_i. L'intervalle $[X_0, X_m]$ est appelé *intervalle d'interpolation*, et les valeurs de la fonction d'interpolation $u^h(x)$ pour x choisi *en dehors* de l'intervalle $[X_0, X_m]$ sont appelées *valeurs extrapolées*.

[1] ou une idée de ces points, car comme chacun le sait, les mesures expérimentales ne sont jamais totalement fiables...

- La fonction $u^h(x)$ ne passe pas par les points, mais s'en rapproche seulement selon un critère de qualité à préciser. On a donc

$$u^h(X_i) \approx U_i.$$

  Typiquement, ce sera le cas si l'on souhaite approximer, par une droite, un grand ensemble de données expérimentales ne se trouvant pas exactement sur une droite : c'est le cas du problème de la *régression linéaire*.

Dans les deux cas, on aura en général $u^h(x) \neq u(x)$ pour $x \neq X_i$. Même lorsque $u(x)$ est un élément du sous-espace d'approximation $\mathcal{U}^h \subset \mathcal{U}$, nous ne pouvons pas être toujours certains que l'approximation obtenue sera systématiquement égale à u, bien qu'on puisse espérer que cela soit le cas pour une procédure d'approximation construite avec un minimum de bon sens. L'interpolation et l'approximation au sens des moindres carrés de données (X_i, U_i) sont illustrées à la Figure 1.1.

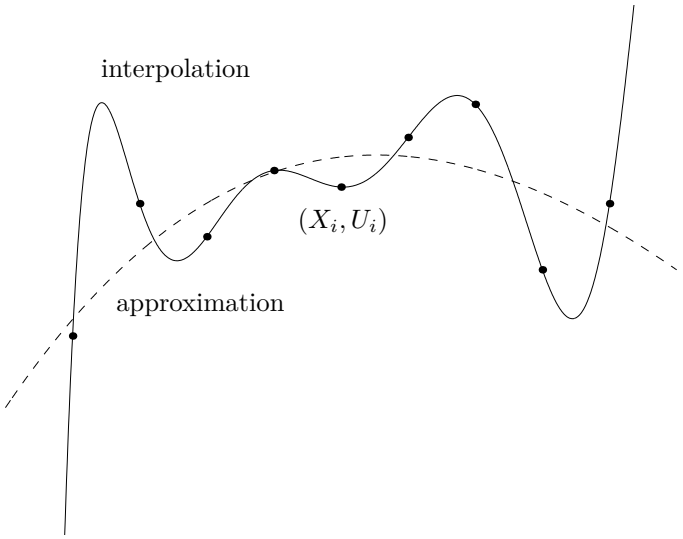


Figure 1.1: Neuf points expérimentaux : interpolation par un polynôme de degré 8 et approximation par un polynôme de degré 4 au sens des moindres carrés.

1.1 Interpolation

Considérons d'abord $n+1$ abscisses distinctes X_i pour lesquelles on connaît les ordonnées U_i d'une fonction inconnue u.

$$(X_i, U_i), \quad i = 0, 1, 2, \ldots, n.$$

Choisissons une famille de $n+1$ fonctions ϕ_j linéairement indépendantes. Il s'agit alors de trouver les paramètres a_j afin que la fonction

$$u^h(x) = \sum_{j=0}^{n} a_j \ \phi_j(x)$$

interpole la fonction $u(x)$ aux abscisses X_i, c'est-à-dire telle que $u^h(X_i) = U_i$. En d'autres mots, le problème peut être formulé comme suit :

Trouver $(a_0, \ldots, a_n) \in \mathbb{R}^{n+1}$ tels que

$$\underbrace{\sum_{j=0}^{n} a_j \ \phi_j(X_i)}_{u^h(X_i)} = U_i \qquad i = 0, 1, \ldots, n. \tag{1.2}$$

Il s'agit donc de résoudre le système linéaire.

$$\begin{bmatrix} \phi_0(X_0) & \phi_1(X_0) & \ldots & \phi_n(X_0) \\ \phi_0(X_1) & \phi_1(X_1) & \ldots & \phi_n(X_1) \\ \phi_0(X_2) & \phi_1(X_2) & \ldots & \phi_n(X_2) \\ \phi_0(X_3) & \phi_1(X_3) & \ldots & \phi_n(X_3) \\ \phi_0(X_4) & \phi_1(X_4) & \ldots & \phi_n(X_4) \\ \vdots & \vdots & & \vdots \\ \phi_0(X_n) & \phi_1(X_n) & \ldots & \phi_n(X_n) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ \vdots \\ U_n \end{bmatrix} \tag{1.3}$$

Notons immédiatement que l'interpolation n'est rien d'autre qu'un cas particulier d'approximation et est donc également une approximation !

1.1.1 Matrice de Vandermonde

Comme les polynômes sont faciles à évaluer, dériver ou intégrer, il est très fréquent de choisir des *polynômes* comme fonctions de base $\phi_j(x)$. Evidemment, il existe plusieurs

possibilités de choix qu'il faudra effectuer de la manière la plus astucieuse. Commençons par adopter comme fonctions de base les monômes

$$\phi_j(x) = x^j \qquad\qquad j = 0, 1, 2, \ldots, n.$$

L'interpolation recherchée est un polynôme de degré n de la forme

$$u^h(x) = \sum_{j=0}^{n} a_j \; x^j$$

Le système (1.3) pour l'obtention des coefficients a_j devient

$$\begin{bmatrix} 1 & X_0 & \ldots & X_0^n \\ 1 & X_1 & \ldots & X_1^n \\ \vdots & \vdots & & \vdots \\ 1 & X_n & \ldots & X_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_n \end{bmatrix}.$$

La matrice de ce système est la *matrice de Vandermonde*. Si les abscisses d'interpolation X_i sont distinctes, le déterminant de la matrice de Vandermonde ne s'annule jamais et le système linéaire possède une solution unique. Ainsi, *il existe un et un seul polynôme d'interpolation de degré n au plus qui passe par $n+1$ points d'abscisses distinctes.*

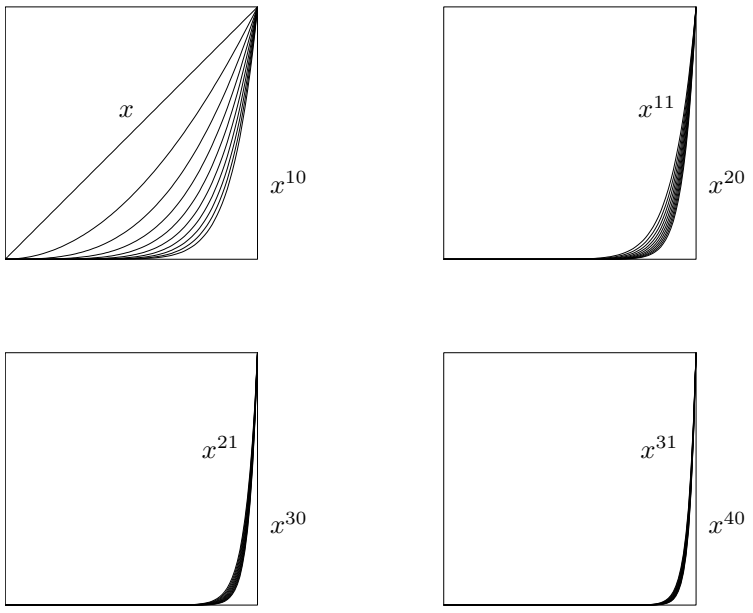


Figure 1.2: Les quarante premiers monômes x^j sur l'intervalle $[0, 1]$: les fonctions deviennent progressivement de plus en plus semblables.

Toutefois, le conditionnement de la matrice de Vandermonde augmente fortement avec la taille $(n+1)$ du système. De manière intuitive, la Figure 1.2 illustre le comportement des 40 premiers monômes x^j sur l'intervalle [0,1]. Nous voyons que les fonctions x^j deviennent de plus en plus semblables lorsque j augmente. Les colonnes associées de la matrice sont donc presque égales, et son déterminant devient de plus en plus petit. C'est pourquoi on dit que le système linéaire devient de plus en plus *mal conditionné* lorsque le nombre de points d'interpolation augmente. En outre, nous allons voir qu'il est possible d'éviter de devoir résoudre un système linéaire pour calculer le polynôme d'interpolation. *Cette méthode est donc rarement utilisée.*

1.1.2 Formule d'interpolation de Lagrange

Les fonctions de base de Lagrange sont déterminées de sorte que la matrice du système (1.3) soit égale à la matrice unité. A chaque abscisse d'interpolation X_i, on associe une fonction de base $\phi_i(x)$ définie par

$$\phi_i(x) \quad = \quad \frac{(x-X_0)(x-X_1)\ \ldots\ (x-X_{i-1})(x-X_{i+1})\ \ldots\ (x-X_n)}{(X_i-X_0)(X_i-X_1)\ldots(X_i-X_{i-1})(X_i-X_{i+1})\ldots(X_i-X_n)}. \qquad (1.4)$$

Puisque $\phi_i(X_j)$ vaut l'unité si $i=j$ et zéro dans les autres cas, la résolution du système est immédiate et nous en déduisons

$$u^h(x) = \sum_{i=0}^{n} U_i \phi_i(x)$$

Insistons sur le fait que ce polynôme est identique à celui calculé en résolvant le système linéaire de Vandermonde !

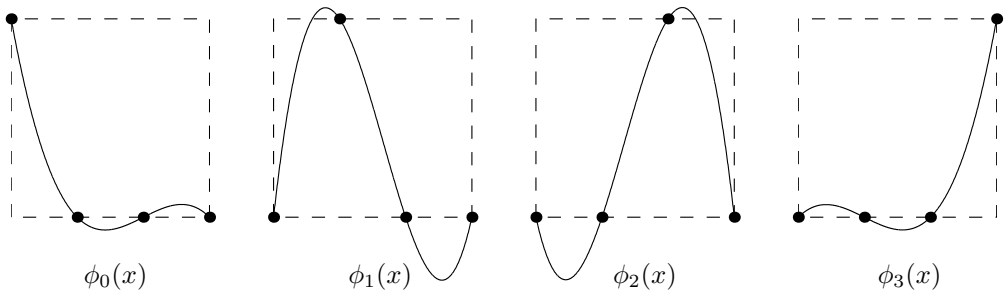


Figure 1.3: Fonctions de base de Lagrange pour les abscisses $X_0 = 0$, $X_1 = \frac{1}{3}$, $X_2 = \frac{2}{3}$ et $X_3 = 1$ sur l'intervalle $[0, 1]$.

1.1.3 Erreur d'interpolation

Remplacer une fonction u par une interpolation polynomiale u^h introduit une *erreur* donnée par

$$e^h(x) = u(x) - u^h(x) \tag{1.5}$$

Cette fonction $e^h(x)$ est appelée *erreur d'interpolation*. Cela signifie que l'interpolation polynomiale $u^h(x)$ procure une approximation de la fonction $u(x)$ avec une erreur $e^h(x)$. Le théorème suivant fournit une expression analytique du terme d'erreur.

Théorème 1.1.

Soit $X_0 < X_1 < \ldots < X_n$ les abscisses des points d'interpolation. Si la fonction $u(x)$ est définie sur l'intervalle $[X_0, X_n]$ et qu'elle est $(n+1)$ fois dérivable sur $]X_0, X_n[$, alors pour tout $x \in]X_0, X_n[$, il existe $\xi(x) \in]X_0, X_n[$ tel que :

$$e^h(x) = \frac{u^{(n+1)}(\xi(x))}{(n+1)!}\ (x - X_0)(x - X_1)(x - X_2)\cdots(x - X_n).$$

Démonstration : nous n'allons pas effectuer une démonstration complète et nous allons nous restreindre au cas où $u^h(x)$ est une fonction linéaire ($n = 1$). Définissons une fonction $g(t)$ comme suit :

$$g(t) = u(t) - u^h(t) - e^h(x)\frac{(t - X_0)(t - X_1)}{(x - X_0)(x - X_1)}$$

Notons que x, X_0 et X_1 sont des constantes et que la variable est t. Il est facile d'observer que g s'annule, lorsqu'on l'évalue pour $t = x$, $t = X_0$ et $t = X_1$.

Considérons ensuite une valeur de x telle que $X_0 < x < X_1$ et appliquons le théorème de Rolle pour g sur les intervalles $[X_0, x]$ et $[x, X_1]$: on obtient des valeurs $X_0 < c < x$ et $x < d < X_1$ telles que $g'(c) = g'(d) = 0$. En appliquant encore une fois ce théorème pour g' sur l'intervalle $[c, d]$, on obtient maintenant une valeur ξ telle que :

$$
\begin{array}{rcl}
g''(\xi) & = & 0 \\
& \downarrow & \\
u''(\xi) - \underbrace{(u^h)''(\xi)}_{=0} - e^h(x)\dfrac{2}{(x-X_0)(x-X_1)} & = & 0 \\
& \downarrow & \\
e^h(x) & = & \dfrac{u''(\xi)}{2}(x-X_0)(x-X_1) \qquad \square
\end{array}
$$

Avant d'aller plus loin, il est possible de faire quelques observations sur cette expression analytique de l'erreur d'interpolation :

- Bien entendu, nous avons que $e^h(X_i) = 0$!
- Cette expression fait intervenir la dérivée d'ordre $(n+1)$ de la fonction u qui est *a priori* inconnue. Il est donc peu probable de connaître ses dérivées et encore moins, la valeur de sa dérivée d'ordre $(n+1)$ en un point $\xi(x)$ également inconnu et qui, en plus, a l'audace de varier avec x. En d'autres mots, voilà une relation qui semble bien peu utile... Et pourtant !
- Il existe une forte similarité entre l'expression analytique de l'erreur d'interpolation et l'expression analytique du reste du développement en série de Taylor. Dans les deux cas, on montre l'existence d'un point $\xi(x)$ permettant d'évaluer l'erreur, mais que l'on ne peut généralement pas déterminer.

  $$u(x) - u_n(x) = \frac{u^{(n+1)}(\xi(x))}{(n+1)!}\ (x - X_0)^{n+1}$$

  où le polynôme de Taylor de degré n autour du point X_0 est donné par

  $$u_n(x) = u(X_0) + u'(X_0)(x - X_0) + u''(X_0)\frac{(x-X_0)^2}{2!} \ldots + u^{(n)}(X_0)\frac{(x-X_0)^n}{n!}$$

- Puisque le terme d'erreur en un point x fait intervenir des coefficients de la forme $(x - X_i)$, il y a tout intérêt à choisir des points de mesure X_i qui sont les plus proches du point x, si on souhaite que l'erreur en ce point soit la plus petite possible. Cela correspond bien à l'intuition !
- La fonction $(x-X_0)(x-X_1)\ldots(x-X_n)$ est un polynôme de degré $(n+1)$ et possède les $(n+1)$ racines réelles X_i. Nous allons observer qu'une telle fonction peut osciller avec des très grandes amplitudes. Cette observation permet de craindre que les interpolations d'ordre élevé puissent présenter de très grandes erreurs entre les X_i.

Bornes d'erreur et ordre d'interpolation

L'expression analytique de l'erreur d'interpolation ne permet pas d'évaluer numériquement cette erreur. Toutefois, il est souvent souhaitable de pouvoir estimer cette erreur, même de manière grossière. On peut se poser la question suivante : est-il possible d'obtenir *a priori* une *borne supérieure* de l'erreur d'interpolation $e^h(x)$ en valeur absolue ?

Admettons que la fonction u satisfasse aux hypothèses du théorème 1.1. Notons immédiatement que pour une fonction inconnue, ce n'est pas évident à vérifier et que cela nécessite un acte de foi ! Admettons, en outre, que l'on puisse trouver une estimation C_{n+1} qui borne la valeur absolue de la dérivée $(n+1)^{ième}$ de $u(x)$. Notons, à nouveau, qu'en général, une telle constante n'est ni disponible, ni estimable dans les applications pratiques. Mais, supposons que toutes les hypothèses soient satisfaites et choisissons une abscisse $x \in]X_0, X_n[$ en considérant le cas de $n+1$ abscisses équidistantes avec $X_{i+1} = X_i + h$. Nous pouvons écrire que

$$
\begin{array}{rcl}
|e^h(x)| & \leq & \dfrac{C_{n+1}}{(n+1)!} \underbrace{|(x-X_0)(x-X_1)\cdots(x-X_n)|}_{\leq n!\ h^{n+1}/4} \\
 & \Big\downarrow & \text{En définissant une nouvelle constante } C = \frac{n! C_{n+1}}{4(n+1)!} \\
|e^h(x)| & \leq & Ch^{n+1} \\
 & \Big\downarrow & \\
e^h(x) & = & \mathcal{O}(h^{n+1})
\end{array}
\tag{1.6}
$$

où nous avons introduit la notation $\mathcal{O}(h^{n+1})$. On dit ainsi qu'en un point donné x, la dépendance[2] du terme d'erreur $e^h(x)$ par rapport à h est un grand ordre de h^{n+1}.

Définition 1.1.

On dit qu'une fonction $f(h)$ est un grand ordre de h^m au voisinage de 0, s'il existe une constante C telle que :

$$\left|\frac{f(h)}{h^m}\right| \leq C$$

au voisinage de 0. On écrit alors $f(h) = \mathcal{O}(h^m)$.

[2]Attention, c'est un peu délicat ! Comme h devient également une variable, il faudrait d'abord introduire $e(x,h)$ pour désigner la fonction à deux variables qui renvoie la valeur de l'erreur $e^h(x)$ pour des valeurs de x et de h. Ensuite, pour une valeur fixée de $x = \alpha$, on va s'intéresser à la fonction $f(h) = e(\alpha, h)$.

Bien que relativement floue, cette définition exprime bien l'idée d'une telle fonction $\mathcal{O}(h^m)$. Lorsque h est assez petit, la fonction $\mathcal{O}(h^m)$ décroît comme Ch^m. Plus m est grand, plus la décroissance est rapide. Ainsi, une fonction $\mathcal{O}(h^3)$ décroît plus vite qu'une fonction $\mathcal{O}(h^2)$, qui elle-même décroît plus vite qu'une fonction $\mathcal{O}(h)$. Pour avoir une idée intuitive du comportement de telles fonctions, il suffit d'observer que lorsque h est divisé par deux, la fonction $\mathcal{O}(h^m)$ diminue selon un facteur approximatif de 2^m. En effet, si l'on remplace h par $h/2$ dans Ch^m, on obtient :

$$C\left(\frac{h}{2}\right)^m = \frac{1}{2^m}\, Ch^m$$

Définition 1.2. *Une approximation numérique dont le terme d'erreur est $\mathcal{O}(h^m)$ est dite d'ordre m.*

Suivant cette définition, le polynôme de Taylor de degré n autour de X_0 est généralement (*mais pas toujours*) une approximation d'ordre $(n+1)$ pour des valeurs de x comprises entre X_0 et $X_0 + h$. De même, l'interpolation polynomiale est souvent (*mais pas toujours*) une approximation de degré n et d'ordre $(n+1)$. On observe immédiatement qu'il y a une confusion possible entre le degré et l'ordre et il faudra donc veiller à être précis.

Finalement, c'est le moment de justifier l'usage de la notation u^h pour décrire une approximation ou une interpolation d'une fonction u. L'idée est qu'une telle approximation pourrait être paramétrée par une grandeur caractéristique, telle que la distance entre deux abscisses. Il s'agit évidemment d'un raccourci, car la caractérisation d'une approximation numérique devrait contenir tous les paramètres nécessaires pour la calculer : c'est-à-dire, le degré d'une approximation polynomiale et toutes les abscisses des données, ainsi que la manière de choisir l'approximation. Est-ce que toutes les données doivent avoir la même importance, par exemple ?

1.1.4 Convergence de l'interpolation polynomiale

Intuitivement, on espère qu'en augmentant le nombre de données (c'est-à-dire, en augmentant n), cette erreur va diminuer. En d'autres mots, on espère utiliser une méthode d'interpolation qui converge.

Définition 1.3.

Une interpolation est dite convergente *si l'erreur d'interpolation tend vers zéro lorsque le nombre de degrés de liberté, c'est-à-dire n tend vers l'infini :*

$$\lim_{n\to\infty} e^h(x) = 0 \quad \textit{pour } x \in [X_0, X_n].$$

Si nous augmentons le nombre d'abscisses d'interpolation X_i, comprises dans un intervalle $[a, b]$ fini donné, nous sommes tentés de croire que la précision de l'interpolation va augmenter, c'est-à-dire que l'erreur d'interpolation $e_n(x)$ va tendre vers zéro pour tout x dans $[a, b]$ lorsque n augmente. En fait, on peut prouver ce résultat de convergence pour des fonctions $u(x)$ dont toutes les dérivées sont bornées sur $[a, b]$ par une constante : par exemple, les fonctions e^x, $\sin(x)$ et $\cos(x)$. La Figure 1.4 illustre la convergence de l'interpolation polynomiale de la fonction $\cos(x)$ sur l'intervalle $[0, \pi]$. Les abscisses d'interpolation x_i sont choisies équidistantes dans $[0, \frac{\pi}{2}]$.

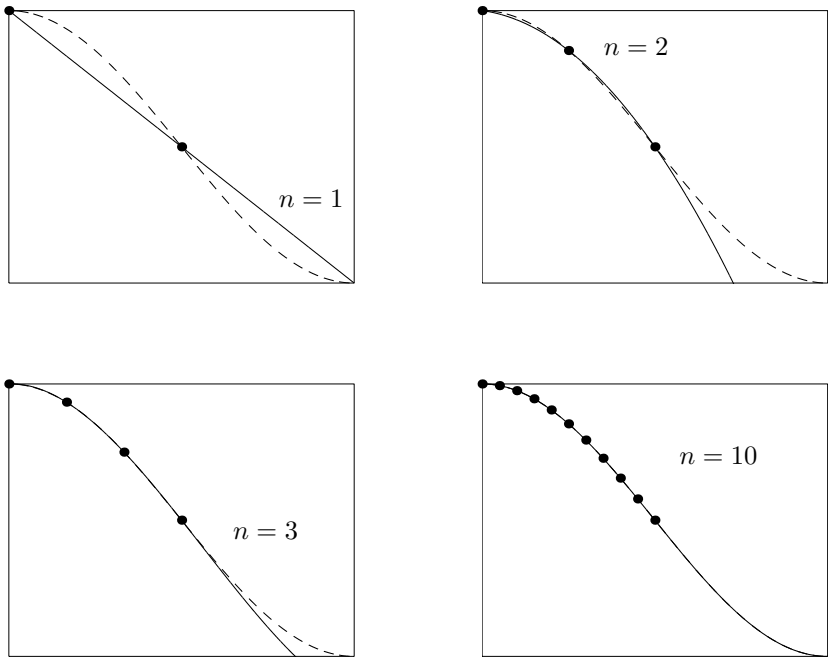


Figure 1.4: Convergence de l'interpolation polynomiale de $\cos(x)$ avec des abscisses équidistantes dans $[0, \frac{\pi}{2}]$. On observe que l'extrapolation entre $\frac{\pi}{2}$ et π converge également, mais nettement plus lentement.

Phénomène de Runge

Pour des fonctions $u(x)$ moins régulières, par contre, il peut arriver que l'interpolation polynomiale *ne converge pas* lorsque le nombre de points d'interpolation augmente. L'expression analytique de l'erreur d'interpolation comprend trois parties. La présence du terme $(n+1)!$ au dénominateur tend bien à réduire l'erreur lorsque n augmente. Par contre, cet effet bénéfique peut dans certains cas être compensé par le mauvais comportement de la dérivée $u^{(n+1)}(\xi)$ et du produit $(x - X_0)(x - X_1) \cdots (x - X_n)$ de sorte que l'erreur d'interpolation augmente lorsque h diminue.

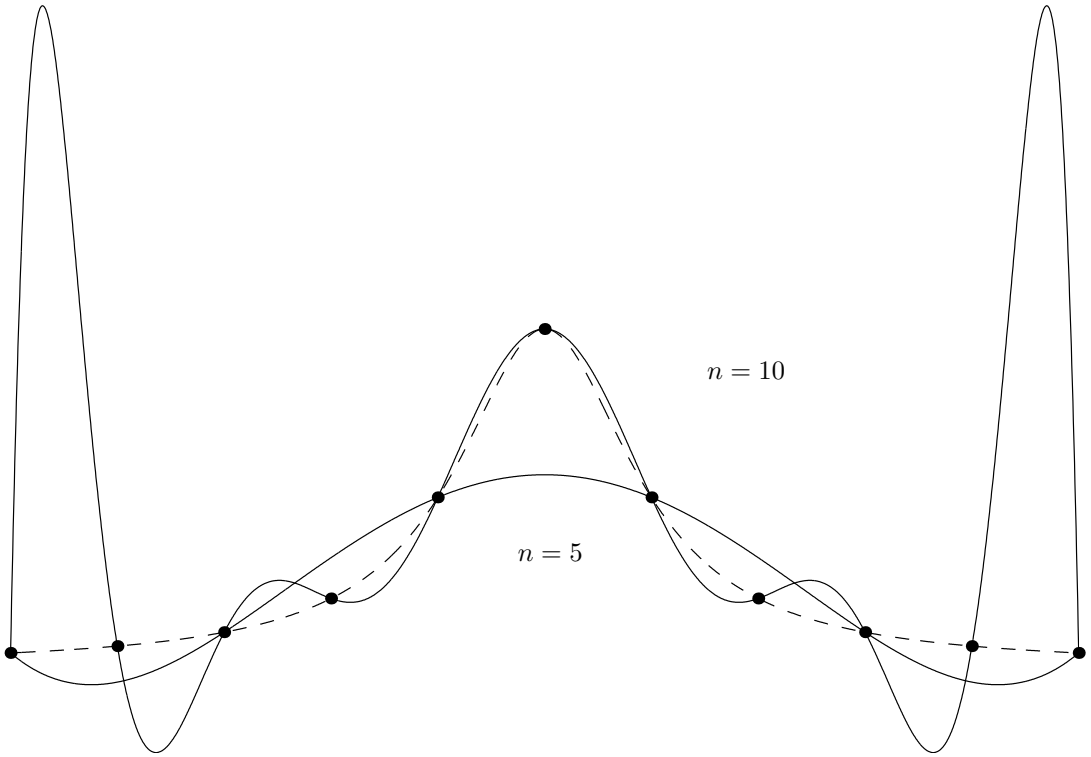


Figure 1.5: Interpolation polynomiale de la fonction de Runge en des points d'abscisses équidistantes.

Interpolons la fonction de Runge

$$u(x) = \frac{1}{1 + 25x^2} \tag{1.7}$$

sur l'intervalle $[-1, 1]$, et ce en un nombre croissant d'abscisses X_i équidistantes. La Figure 1.5 compare la fonction $f(x)$ aux interpolants polynomiaux de degré 5 et 10. On observe le *phénomène de Runge*, à savoir la présence de fortes oscillations de l'interpolant *près des bords* de l'intervalle d'interpolation, lorsque le degré n augmente. En d'autres mots, on observe que les interpolations polynomiales successives ne convergent pas vers la fonction de Runge.

1.1.5 Interpolation aux abscisses de Chebyshev

Après avoir observé le phénomène de Runge avec un choix particulier d'abscisses X_i, en l'occurrence des abscisses *équidistantes*, on pourrait décider de rechercher un meilleur choix des abscisses. Pour un nombre $n+1$ donné d'abscisses dans l'intervalle[3] $[-1, 1]$, on va donc rechercher la position d'abscisses X_i telles qu'on minimise la norme du maximum du carré de l'erreur commise par l'interpolation de degré n d'un polynôme de degré $n+1$.

Il est possible de démontrer[4] qu'il est équivalent d'exiger que cette erreur soit un polynôme de degré $n+1$ ayant des oscillations d'amplitude égale entre chaque abscisse. En d'autres mots, sur l'intervalle $[-1, 1]$, on peut écrire que $-E \leq e^h(x) \leq E$ est un polynôme de degré $n+1$, dont les maxima et les minima intermédiaires valent respectivement E et $-E$. Les valeurs absolues aux extrémités de l'intervalle valent également E. Cette situation est représentée sur la Figure (1.6) : on voit que l'erreur s'inscrit dans une boite. On peut alors effectuer les observations suivantes :

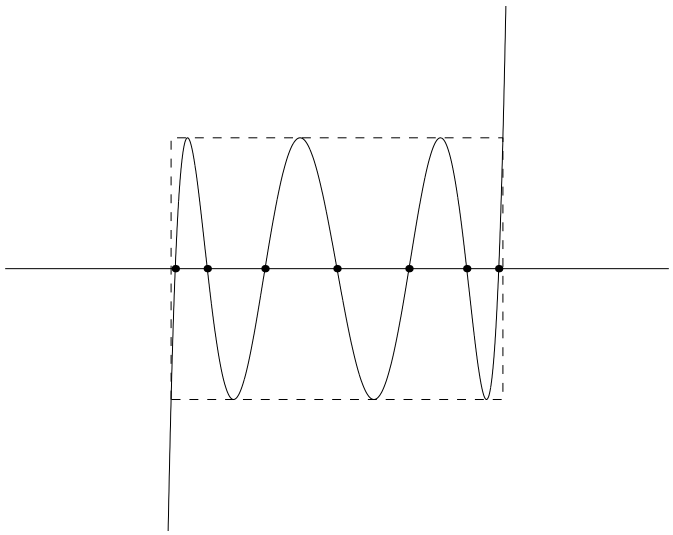


Figure 1.6: Allure de l'erreur en utilisant une interpolation de degré $n = 6$ d'un polynôme de degré $n+1$, en faisant usage des abscisses de Chebyshev. L'erreur représentée est donc un polynôme de degré $n+1 = 7$.

- La fonction $e(x)$ est un polynôme de degré $n+1$ qui s'annule aux $n+1$ abscisses d'interpolation.

[3]Pour passer à un intervalle quelconque, il suffit d'effectuer un changement de variable adéquat.

[4]Intuitivement, on voit bien que choisir les abscisses laisse $n+1$ degrés de liberté qui permettent bien d'exiger que les valeurs absolues des minima et maxima locaux soient égales entre elles. On voit bien aussi que si une valeur est légèrement supérieure aux autres, il est avantageux de diminuer celle-ci en acceptant d'augmenter les autres afin de les égaler : puisque c'est uniquement la plus grande valeur qui est pénalisante.

- La fonction $e'(x)$ est un polynôme de degré n qui s'annule aux n minima et maxima intermédiaires.
- La fonction $(e'(x))^2$ est un polynôme de degré $2n$ qui a des racines doubles aux n minima ou maxima intermédiaires.
- La fonction $E^2 - e^2(x)$ est un polynôme de degré $2n+2$ qui a des racines doubles aux n minima ou maxima intermédiaires et des racines simples aux extrémités de l'intervalle.

On en déduit que l'égalité suivante doit être respectée : il s'agit de deux polynômes de même degré avec les mêmes racines et le même terme dominant. Il suffit ensuite de résoudre l'équation différentielle qui en résulte...

$$
\begin{aligned}
(e'(x))^2 (1-x^2) &= (n+1)^2 (E^2 - e^2(x)) \\
&\downarrow \\
\frac{\frac{e'(x)}{E}}{\sqrt{1-\left(\frac{e(x)}{E}\right)^2}} &= \pm(n+1)\frac{1}{\sqrt{1-x^2}} \\
&\downarrow \\
\arccos\left(\frac{e(x)}{E}\right) &= \pm\left((n+1)\arccos(x) + C\right) \\
&\downarrow \quad \text{En vertu de la parité du cosinus !} \\
e(x) &= E\cos\left((n+1)\arccos(x) + C\right) \\
&\downarrow \quad \text{En imposant que e(1) = E} \\
e(x) &= E\underbrace{\cos\left((n+1)\arccos(x)\right)}_{T_{n+1}(x)}
\end{aligned}
$$

On observe que l'erreur est proportionnelle à $\cos\left((n+1)\arccos(x)\right)$. Et cette expression est un polynôme ! Il s'agit du $(n+1)^{ième}$ *polynôme de Chebyshev*, noté $T_{n+1}(x)$. On peut immédiatement observer que $T_0(x) = 1$ et $T_1(x) = x$ et les autres polynômes sont obtenus par une formule de récurrence.

Théorème 1.2.

Les polynômes de Chebyshev $T_{n+1}(x) = \cos\left((n+1)\arccos(x)\right)$ *définis sur l'intervalle* $[-1,1]$ *satisfont la relation de récurrence*

$$T_{i+1}(x) = 2x\ T_i(x) - T_{i-1}(x), \quad i = 1, 2, 3, \ldots,$$

avec $T_0(x) = 1$ *et* $T_1(x) = x$.

Démonstration : Définissons $\theta = \arccos(x)$ et écrivons :

$$\begin{aligned} T_{i+1}(x) &= \cos((i+1)\theta) \\ &= \cos(\theta)\cos(i\theta) - \sin(\theta)\sin(i\theta) \end{aligned}$$

$$\begin{aligned} T_{i-1}(x) &= \cos((i-1)\theta) \\ &= \cos(\theta)\cos(i\theta) + \sin(\theta)\sin(i\theta) \end{aligned}$$

$$\begin{aligned} T_{i+1}(x) + T_{i-1}(x) &= 2\cos(\theta)\cos(i\theta) \\ &= 2xT_i(x) \end{aligned}$$ □

Les $n+1$ racines de ce polynôme de Chebyshev sont les abscisses de Chebyshev.

$$0 = \overbrace{\cos\left((n+1)\arccos(X_i)\right)}^{T_{n+1}(X_i)} \qquad i = 0, \ldots, n$$

$$\downarrow$$

$$\frac{\pi/2 + i\pi}{(n+1)} = \arccos(X_i) \qquad i = 0, \ldots, n$$

$$\downarrow$$

$$\cos\left(\frac{(2i+1)\pi}{2(n+1)}\right) = X_i \qquad i = 0, \ldots, n$$

Ces abscisses ne sont pas équidistantes. Elles seront plus nombreuses près des bords.

Choisissons, maintenant, comme abscisses pour interpoler la fonction de Runge les 11 racines du polynôme de Chebyshev $T_{11}(x)$. Les abscisses d'interpolation de Chebyshev sont concentrées aux extrémités de l'intervalle $[-1,1]$, précisément là où l'interpolation en des abscisses équidistantes était très mauvaise sur la Figure 1.5. La Figure 1.7 montre que le polynôme d'interpolation avec les abscisses de Chebyshev est une bien meilleure approximation de la fonction de Runge.

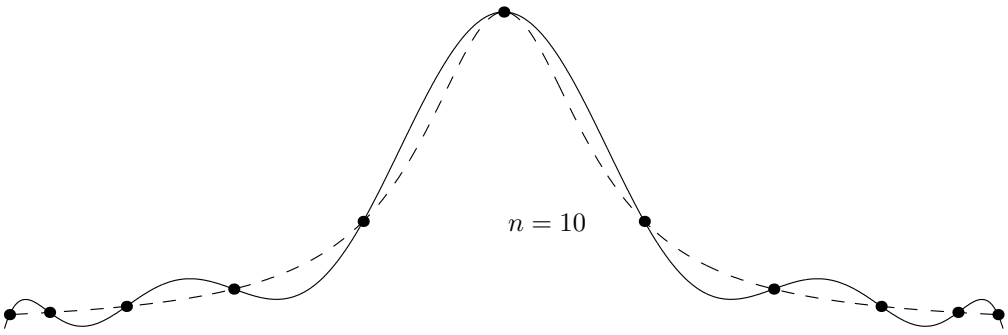


Figure 1.7: Polynôme d'interpolation de la fonction de Runge : abscisses d'interpolation de Chebyshev.

Bien que les abscisses d'interpolation de Chebyshev conduisent à des interpolants polynomiaux de degré élevé de très bonne qualité pour certaines fonctions, ce n'est pas le cas pour des données tout à fait quelconques. En règle générale, l'interpolation polynomiale de degré élevé est peu recommandable pour des fonctions $u(x)$ peu régulières.

1.1.6 Interpolation par splines cubiques

Jusqu'à présent, nous avons considéré le problème d'interpolation polynomiale de façon *globale*, c'est-à-dire sur l'entièreté d'un intervalle $[a, b]$ contenant les abscisses des données. Nous venons aussi de constater que l'utilisation de polynômes de degré élevé peut produire des résultats très décevants, alors qu'il est souvent nécessaire d'obtenir des courbes très régulières ou très lisses[5] passant exactement par un grand nombre de points $(X_i, u(X_i))$. C'est le cas en conception assistée par ordinateur (CAO), où l'on veut concevoir des formes régulières.

Une autre approche, consiste à découper cet intervalle en n sous-intervalles $[X_{i-1}, X_i]$ avec $i = 1, 2, \ldots, n$ et $X_0 < X_1 < \ldots < X_n$ et de construire un polynôme de degré plus faible sur chaque intervalle. La fonction globale u^h est alors définie *par morceaux* sur $[a, b]$. C'est ce qu'on appelle l'*interpolation polynomiale par morceaux*.

Par exemple, on peut construire une *interpolation linéaire par morceaux* en reliant chaque paire de points par un segment de droite comme illustré sur la Figure 1.8. On utilise aussi l'expression *splines linéaires*. Toutefois, on imagine mal comment une telle courbe pourrait permettre de faire la conception d'une aile d'avion ou d'une carrosserie de voiture. Il est donc nécessaire d'effectuer avec soin la jonction entre les différents segments de courbe. Un choix populaire consiste à utiliser dans chaque intervalle $[X_{i-1}, X_i]$ un

[5]Evidemment, il y a une question que l'on doit se poser ici : comment mesurer la régularité d'une fonction ? Une façon de procéder est de la mesurer par le biais de ses dérivées. Plus une fonction est différentiable, plus la courbe qui lui est associée est lisse et plus la fonction est régulière.

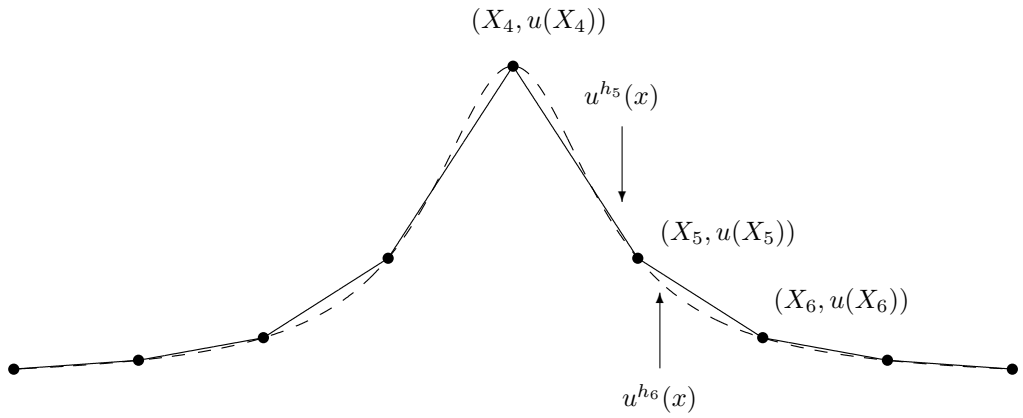


Figure 1.8: Interpolation linéaire par morceaux de la fonction de Runge $u(x) = 1/(1 + 25x^2)$: 8 segments de droite forment la fonction u^h.

polynôme de degré trois de la forme suivante

$$u^{h_i}(x) = a_i + b_i x + c_i x^2 + d_i x^3 \qquad\qquad i = 1, 2, \ldots, n$$

et à relier ces différents polynômes entre eux de façon à ce que la courbe globale soit deux fois différentiable. C'est le principe de *l'interpolation par splines cubiques* illustrée sur la Figure 1.9. Supposons donc que l'on ait $n+1$ points d'interpolation et donc n intervalles, il y a donc $4n$ coefficients à déterminer le plus efficacement possible. Une approche astucieuse permet de les obtenir par la résolution d'un système linéaire tridiagonal[6] de dimension $n + 1$ qui est très aisé à résoudre.

Il s'agit maintenant de déterminer les conditions que nous pouvons imposer à ces $4n$ coefficients.

- Tout d'abord, les polynômes u^{h_1} et u^{h_n} passent par les deux extrémités.

$$\begin{aligned} u^{h_1}(X_0) &= U_0 \\ u^{h_n}(X_n) &= U_n \end{aligned}$$

  où U_i dénote la valeur connue de $u(X_i)$. Cela fait deux équations.

- Ensuite pour chaque point intérieur $X_i (i = 1, 2, \ldots, n-1)$, il y a deux polynômes

[6]Une matrice est dite tridiagonale si ses seuls termes non nuls sont situés sur la diagonale principale et les deux diagonales adjacentes. Tous les autres termes sont nuls.

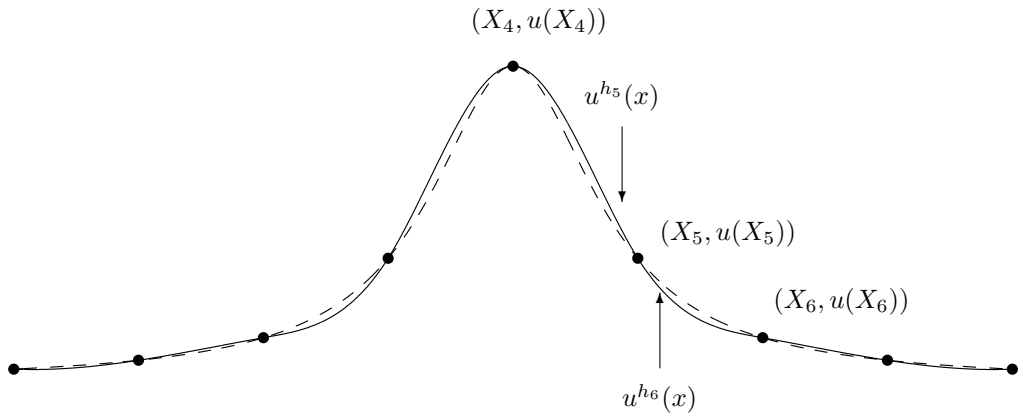


Figure 1.9: Interpolation de la fonction de Runge $u(x) = 1/(1+25x^2)$ par des splines cubiques par 9 points : 8 polynômes de degré trois forment la fonction u^h qui est deux fois différentiable.

qui doivent passer par le point.

$$\begin{aligned} u^{h_i}(X_i) &= U_i & i = 1, \ldots, n-1 \\ u^{h_{i+1}}(X_i) &= U_i & i = 1, \ldots, n-1 \end{aligned}$$

Cela fait $(2n-2)$ équations supplémentaires.

- Finalement, on impose la continuité des dérivées premières et secondes en chaque point intérieur $X_i (i = 1, 2, \ldots, n-1)$.

$$\begin{aligned} (u^{h_i})'(X_i) &= (u^{h_{i+1}})'(X_i) & i = 1, \ldots, n-1 \\ (u^{h_i})''(X_i) &= (u^{h_{i+1}})''(X_i) & i = 1, \ldots, n-1 \end{aligned}$$

Cela fait $(2n-2)$ équations supplémentaires.

En conclusion, on a $4n-2$ équations pour $4n$ coefficients inconnus. La détermination d'un unique interpolant spline cubique exige donc *deux conditions supplémentaires*. Comme nous le verrons plus loin, il existe plusieurs façons de compléter le système d'équations.

La résolution du système est basée sur l'observation suivante : puisque la spline est composée de polynômes de degré trois et est deux fois différentiable, la dérivée seconde de celle-ci est composée de segments de droites sur chaque intervalle reliant des points (X_i, U_i''), comme indiqué sur la Figure 1.10. En d'autres mots, dans l'intervalle $[X_{i-1}, X_i]$, la dérivée seconde $(u^{h_i})''$ est un segment de droite passant par les points (X_{i-1}, U_{i-1}'') et (X_i, U_i'') et peut s'écrire en vertu de la formule d'interpolation de Lagrange sous la forme :

$$(u^{h_i})'' = U''_{i-1}\frac{(x - X_i)}{(X_{i-1} - X_i)} + U''_i\frac{(x - X_{i-1})}{(X_i - X_{i-1})}$$

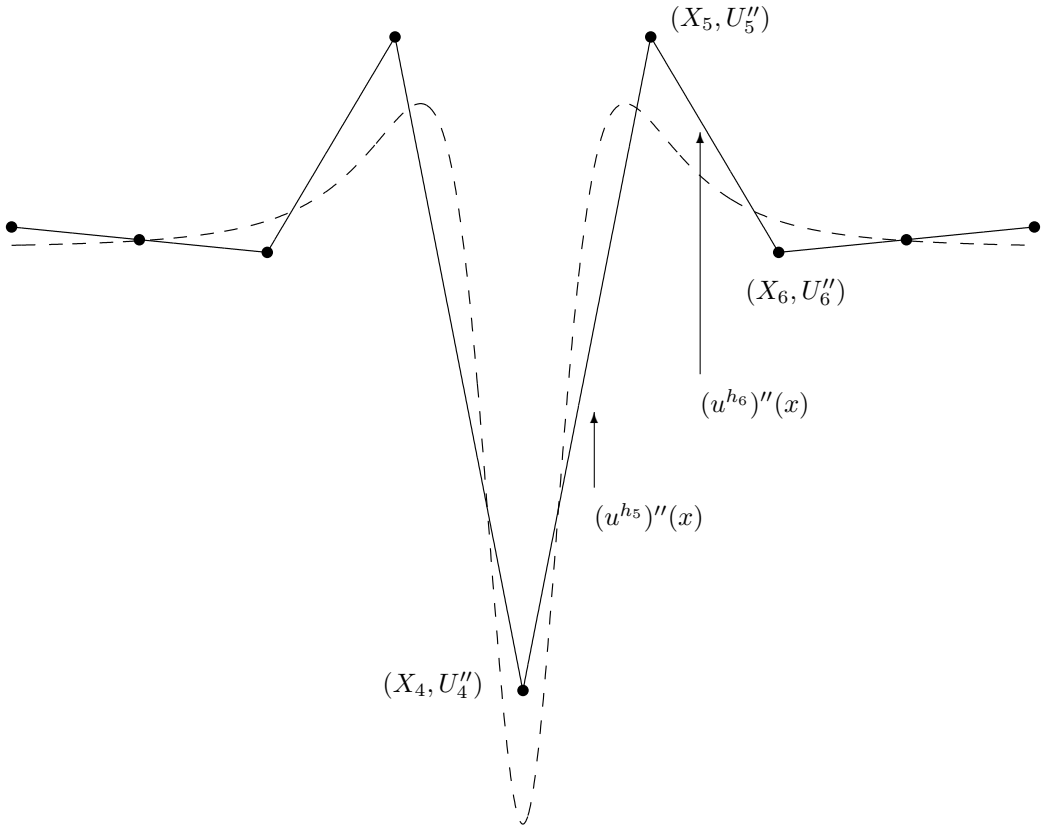


Figure 1.10: Dérivée seconde de la fonction de Runge $u(x) = 1/(1 + 25x^2)$ et des splines cubiques. La fonction $(u^h)''(x)$ apparaît comme une approximation linéaire (et non une interpolation !) par morceaux de $u''(x) = (5000x^2)/(1+25x^2)^3 - 50/(1+25x^2)^2$ représentée en traits discontinus. En d'autres mots, (X_i, U''_i) n'est pas égal à $(X_i, u''(X_i))$.

En notant $h_i = X_i - X_{i-1}$ pour $i = 1, 2, \ldots, n$, nous pouvons alors écrire :

$$\begin{aligned}
(u^{h_i})'' &= U''_{i-1}\frac{(X_i - x)}{h_i} + U''_i\frac{(x - X_{i-1})}{h_i} \\
&\Big\downarrow \quad \text{En intégrant deux fois,} \\
(u^{h_i}) &= U''_{i-1}\frac{(X_i - x)^3}{6h_i} + U''_i\frac{(x - X_{i-1})^3}{6h_i} + A_i\frac{(X_i - x)}{h_i} + B_i\frac{(x - X_{i-1})}{h_i}
\end{aligned}$$

où A_i et B_i sont les deux constantes arbitraires d'intégration définies de manière astucieuse, en s'inspirant de la formule de Lagrange. En effet, pour déterminer ces deux constantes, on va imposer le passage de u^{h_i} par les points $(X_{i-1}, u(X_{i-1}))$ et $(X_i, u(X_i))$. Ce qui revient à écrire :

$$\begin{array}{rcl} U_{i-1} & = & U''_{i-1}\dfrac{h_i^3}{6h_i} + A_i\dfrac{h_i}{h_i} \\ \downarrow & & \\ A_i & = & U_{i-1} - \dfrac{U''_{i-1}\ h_i^2}{6} \end{array} \quad \text{et} \quad \begin{array}{rcl} U_i & = & U''_i\dfrac{h_i^3}{6h_i} + B_i\dfrac{h_i}{h_i} \\ \downarrow & & \\ B_i & = & U_i - \dfrac{U''_i\ h_i^2}{6} \end{array}$$

et de conclure immédiatement que :

$$\boxed{\begin{array}{rcl} u^{h_i}(x) & = & \dfrac{U''_{i-1}}{6h_i}(X_i - x)^3 + \dfrac{U''_i}{6h_i}(x - X_{i-1})^3 \\ & & + \left(\dfrac{U_{i-1}}{h_i} - \dfrac{U''_{i-1}\ h_i}{6}\right)(X_i - x) \\ & & + \left(\dfrac{U_i}{h_i} - \dfrac{U''_i\ h_i}{6}\right)(x - X_{i-1}) \end{array}} \tag{1.8}$$

où les $n+1$ coefficients U''_i restent à déterminer. Des $(4n-2)$ conditions retenues, seule la continuité de la dérivée première aux points intérieurs n'a pas encore été utilisée. En dérivant la relation (1.8), imposer cette continuité en $x = X_i$ avec $i = 1, 2, \ldots, n-1$ s'exprime sous la forme :

$$\begin{array}{rcl} (u^{h_i})'(X_i) & = & (u^{h_{i+1}})'(X_i) \\ \downarrow & & \\ \dfrac{U''_i h_i}{2} + \dfrac{(U_i - U_{i-1})}{h_i} - \dfrac{(U''_i - U''_{i-1})h_i}{6} & = & -\dfrac{U''_i h_{i+1}}{2} + \dfrac{(U_{i+1} - U_i)}{h_{i+1}} - \dfrac{(U''_{i+1} - U''_i)h_{i+1}}{6} \\ \dfrac{(2U''_i + U''_{i-1})h_i}{6} + \dfrac{(U_i - U_{i-1})}{h_i} & = & \dfrac{(U_{i+1} - U_i)}{h_{i+1}} - \dfrac{(U''_{i+1} + 2U''_i)h_{i+1}}{6} \end{array}$$

et d'obtenir le système linéaire :

$$\boxed{\frac{h_i}{6}\,U''_{i-1} + \frac{2(h_i+h_{i+1})}{6}\,U''_i + \frac{h_{i+1}}{6}\,U''_{i+1} \;=\; \frac{(U_{i+1}-U_i)}{h_{i+1}} - \frac{(U_i-U_{i-1})}{h_i} \qquad i=1,\ldots,n-1} \tag{1.9}$$

Comme annoncé, nous avons bien obtenu un système tridiagonal de $n-1$ équations à $n+1$ inconnues U''_i. Pour pouvoir résoudre ce système, il faut aussi, comme nous l'avions prévu, spécifier deux conditions supplémentaires. Le choix le plus simple consiste à imposer :

$$U''_0 = 0 \quad \text{et} \quad U''_n = 0$$

On qualifie de *spline naturelle* la courbe qui en résulte. Un autre choix consiste à imposer :

$$U''_0 = U''_1 \quad \text{et} \quad U''_n = U''_{n-1}$$

Cela revient à imposer une courbure constante sur les premier et dernier intervalles. D'autres choix sont possibles. Tout dépend de l'information disponible. Par exemple, il est possible que la pente soit connue aux extrémités ou que la courbe recherchée soit périodique. On peut alors se servir de cette information pour obtenir les deux conditions manquantes.

Abscisses équidistantes

Dans le cas d'abscisses équidistantes ($h_i = h$), les équations (1.9) se simplifient fortement : on obtient pour une spline naturelle un système de la forme

$$\frac{h^2}{6}\begin{bmatrix} 1 & 0 & & & & & & & \\ 1 & 4 & 1 & & & & & & \\ & 1 & 4 & 1 & & & & & \\ & & 1 & 4 & 1 & & & & \\ & & & 1 & 4 & 1 & & & \\ & & & & \cdot & \cdot & \cdot & & \\ & & & & & 1 & 4 & 1 & \\ & & & & & & 1 & 4 & 1 \\ & & & & & & & 0 & 1 \end{bmatrix} \begin{bmatrix} U''_0 \\ U''_1 \\ U''_2 \\ U''_3 \\ U''_4 \\ \vdots \\ U''_{n-2} \\ U''_{n-1} \\ U''_n \end{bmatrix} = \begin{bmatrix} 0 \\ U_0 - 2U_1 + U_2 \\ U_1 - 2U_2 + U_3 \\ U_2 - 2U_3 + U_4 \\ U_3 - 2U_4 + U_5 \\ \vdots \\ \vdots \\ U_{n-2} - 2U_{n-1} + U_n \\ 0 \end{bmatrix}$$

La matrice du système est bien *tri-diagonale*.

1.2 Approximation

Considérons maintenant $m+1$ abscisses distinctes X_i pour lesquelles on connaît les ordonnées U_i d'une fonction inconnue u.

$$(X_i, U_i), \quad i = 0, 1, 2, \ldots, m.$$

Choisissons une famille de $n+1$ fonctions ϕ_j linéairement indépendantes avec $n < m$. Il s'agit alors de trouver les paramètres a_j afin que la fonction

$$u^h(x) = \sum_{j=0}^{n} a_j \ \phi_j(x)$$

approxime seulement au mieux la fonction $u(x)$ aux abscisses X_i, c'est-à-dire telle que $u^h(X_i) \approx u(X_i)$. Ce qui revient à écrire le système de $m+1$ équations à $n+1$ inconnues

$$\begin{bmatrix} \phi_0(X_0) & \phi_1(X_0) & \ldots & \phi_n(X_0) \\ \phi_0(X_1) & \phi_1(X_1) & \ldots & \phi_n(X_1) \\ \phi_0(X_2) & \phi_1(X_2) & \ldots & \phi_n(X_2) \\ \phi_0(X_3) & \phi_1(X_3) & \ldots & \phi_n(X_3) \\ \phi_0(X_4) & \phi_1(X_4) & \ldots & \phi_n(X_4) \\ \phi_0(X_5) & \phi_1(X_5) & \ldots & \phi_n(X_5) \\ \phi_0(X_6) & \phi_1(X_6) & \ldots & \phi_n(X_6) \\ \phi_0(X_7) & \phi_1(X_7) & \ldots & \phi_n(X_7) \\ \vdots & \vdots & & \vdots \\ \phi_0(X_m) & \phi_1(X_m) & \ldots & \phi_n(X_m) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} \approx \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \\ U_7 \\ \vdots \\ U_m \end{bmatrix}.$$

Comme le nombre $m+1$ d'équations est supérieur au nombre $n+1$ d'inconnues, il n'est, en général, pas possible de trouver des valeurs à ces inconnues afin que toutes les équations soient satisfaites. Le respect approximatif des équations est mesuré par le vecteur de $m+1$ résidus R_i définis par :

$$R_i = U_i - \underbrace{\sum_{j=0}^{n} \phi_j(X_i) \ a_j}_{u^h(X_i)} \qquad i = 0, 1, 2, \ldots, m. \tag{1.10}$$

Il s'agira alors de déterminer les valeurs particulières des $n+1$ coefficients a_j qui *minimisent* les résidus (1.10) selon un critère à préciser. Parmi divers critères utilisés en pratique, celui des *moindres carrés* est particulièrement populaire. Dans la méthode d'approximation par moindres carrés, on cherche les coefficients a_j qui minimisent la somme des carrés des résidus ou des écarts verticaux entre les points (X_i, U_i) et $(X_i, u^h(X_i))$, comme illustré à la Figure 1.11.

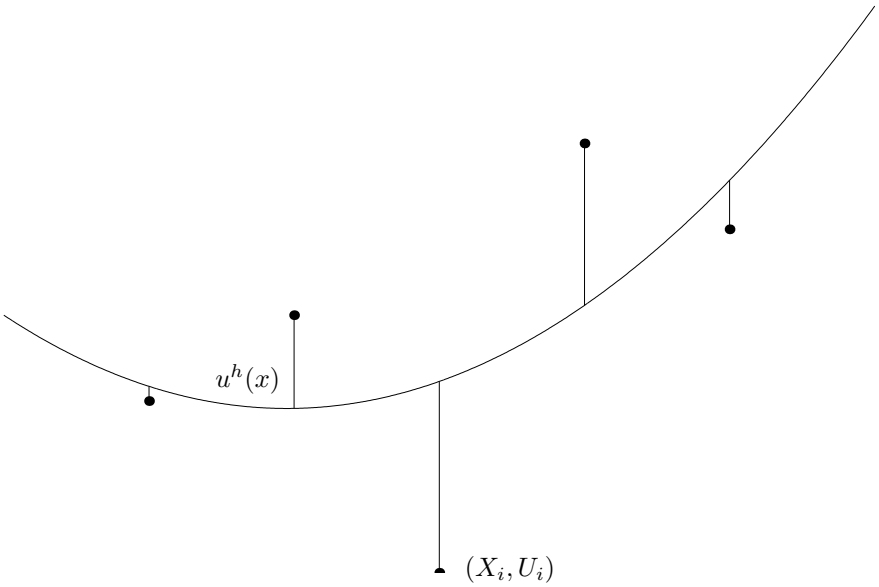


Figure 1.11: Approximation par moindres carrés d'un nuage de points.

En d'autres mots, les paramètres a_j de l'approximation par moindres carrés sont obtenus en minimisant le carré de la norme euclidienne du vecteur résidu. Le problème général de l'approximation *d'un nuage de points* par moindres carrés peut donc s'écrire sous la forme :

Trouver $(a_0, \ldots, a_n) \in \mathbb{R}^{n+1}$ tels que

$$\underbrace{\sum_{i=0}^{m} \left(U_i - \sum_{j=0}^{n} \phi_j(X_i)\ a_j \right)^2}_{J(a_0, \ldots, a_n)} \quad \text{soit minimal.} \tag{1.11}$$

L'expression $J(a_0, \ldots, a_n)$ est une fonction de $(n+1)$ variables réelles. Les valeurs des coefficients a_j qui minimisent cette fonction doivent donc vérifier les $(n+1)$ conditions :

$$
\begin{aligned}
\left.\frac{\partial J}{\partial a_k}\right|_{(a_0,\ldots,a_n)} &= 0 && k=0,1,\ldots,n \\
&\downarrow \\
\sum_{i=0}^{m} -2\phi_k(X_i)\left(U_i - \sum_{j=0}^{n}\phi_j(X_i)\ a_j\right) &= 0 && k=0,1,\ldots,n \\
&\downarrow \\
\sum_{i=0}^{m}\phi_k(X_i)\ \sum_{j=0}^{n}\phi_j(X_i)\ a_j &= \sum_{i=0}^{m}\phi_k(X_i)U_i && k=0,1,\ldots,n \\
&\downarrow \\
\sum_{j=0}^{n}\left(\sum_{i=0}^{m}\phi_k(X_i)\ \phi_j(X_i)\right) a_j &= \sum_{i=0}^{m}\phi_k(X_i)U_i && k=0,1,\ldots,n
\end{aligned}
$$

Ces $n+1$ équations appelées *équations normales* forment un système algébrique linéaire de $n+1$ équations à $n+1$ inconnues. On peut alors trouver les paramètres a_j caractérisant l'approximation $u^h(x)$ recherchée. Le problème général (1.11) de l'approximation *d'un nuage de points* par moindres carrés peut donc être écrit sous la formulation strictement équivalente[7] :

Trouver $(a_0,\ldots,a_n)\in\mathbb{R}^{n+1}$ tels que

$$
\sum_{j=0}^{n}\left(\sum_{i=0}^{m}\phi_k(X_i)\ \phi_j(X_i)\right)\ a_j = \sum_{i=0}^{m}\phi_k(X_i)U_i \qquad k=0,1,\ldots,n \tag{1.12}
$$

Ce qu'on peut écrire sous la forme matricielle suivante.

$$
\begin{bmatrix}
\sum_{i=0}^{m}\phi_0(X_i)\phi_0(X_i) & \sum_{i=0}^{m}\phi_0(X_i)\phi_1(X_i) & \ldots & \sum_{i=0}^{m}\phi_0(X_i)\phi_n(X_i) \\
\sum_{i=0}^{m}\phi_1(X_i)\phi_0(X_i) & \sum_{i=0}^{m}\phi_1(X_i)\phi_1(X_i) & \ldots & \sum_{i=0}^{m}\phi_1(X_i)\phi_n(X_i) \\
\sum_{i=0}^{m}\phi_2(X_i)\phi_0(X_i) & \sum_{i=0}^{m}\phi_2(X_i)\phi_1(X_i) & \ldots & \sum_{i=0}^{m}\phi_2(X_i)\phi_n(X_i) \\
 & \vdots & & \vdots \\
\sum_{i=0}^{m}\phi_n(X_i)\phi_0(X_i) & \sum_{i=0}^{m}\phi_n(X_i)\phi_1(X_i) & \ldots & \sum_{i=0}^{m}\phi_n(X_i)\phi_n(X_i)
\end{bmatrix}
\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}
=
\begin{bmatrix}
\sum_{i=0}^{m}\phi_0(X_i)U_i \\
\sum_{i=0}^{m}\phi_1(X_i)U_i \\
\sum_{i=0}^{m}\phi_2(X_i)U_i \\
\vdots \\
\sum_{i=0}^{m}\phi_n(X_i)U_i
\end{bmatrix}
$$

[7] Pour en être totalement convaincu, il faudrait aussi se préoccuper des dérivées partielles secondes de la fonction J. Pourquoi et qu'en concluez-vous ?

1.2.1 Régressions polynomiales

Il est toujours tentant de choisir des *polynômes* comme fonctions de base $\phi_j(x)$. Commençons par adopter comme fonctions de base les monômes x^j. Typiquement, ce sera le cas si l'on souhaite approximer, par une droite ou une parabole, un grand ensemble de données expérimentales ne se trouvant pas exactement sur une droite : c'est le cas du problème de la *régression linéaire* et de la *régression quadratique*.

Considérons le cas d'une régression quadratique et recherchons les coefficients d'un polynôme de degré 2 de la forme

$$u^h(x) = a_0 + a_1 x + a_2 x^2$$

qui fournit une meilleure approximation au sens des moindres carrés de $m+1$ données expérimentales (X_i, U_i). Le système (1.12) des équations normales pour l'obtention des coefficients a_j devient

$$\begin{bmatrix} (m+1) & \sum_{i=0}^{m} X_i & \sum_{i=0}^{m} X_i^2 \\ \sum_{i=0}^{m} X_i & \sum_{i=0}^{m} X_i^2 & \sum_{i=0}^{m} X_i^3 \\ \sum_{i=0}^{m} X_i^2 & \sum_{i=0}^{m} X_i^3 & \sum_{i=0}^{m} X_i^4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{m} U_i \\ \sum_{i=0}^{m} X_i U_i \\ \sum_{i=0}^{m} X_i^2 U_i \end{bmatrix}.$$

Comme pour l'interpolation polynomiale, il n'est en général pas prudent d'utiliser des approximations par moindres carrés de degré élevé. A titre d'exemple, considérons la fonction $u(x) = 1.44x^{-2} + 0.24$ et les abscisses $0.25, 1.00, 1.50, 2.00, 2.40, 5.00$ indiquées sur la Figure 1.12. Calculons ensuite les meilleures approximations polynomiales par moindres carrés pour ces données discrètes.

$$\begin{array}{lrcl} n=2 & u^h(x) & = & 22.93 - 16.96x + 2.55x^2 \\ n=3 & u^h(x) & = & 33.03 - 46.51x + 19.51x^2 - 2.30x^3 \\ n=4 & u^h(x) & = & 39.92 - 80.93x + 58.39x^2 - 17.15x^3 + 1.68x^4 \\ n=5 & u^h(x) & = & 46.02 - 118.14x + 119.36x^2 - 57.51x^3 + 13.03x^4 - 1.09x^5 \end{array}$$

où le polynôme d'approximation de degré 5 est évidemment le polynôme d'interpolation de la fonction aux abscisses X_i.

1.2.2 Et si nous pouvons disposer de la fonction $u(x)$...

Si nous pouvons disposer de la fonction $u(x)$ lors du calcul des paramètres a_j , il est possible d'exploiter toute l'information disponible pour obtenir une approximation polynomiale

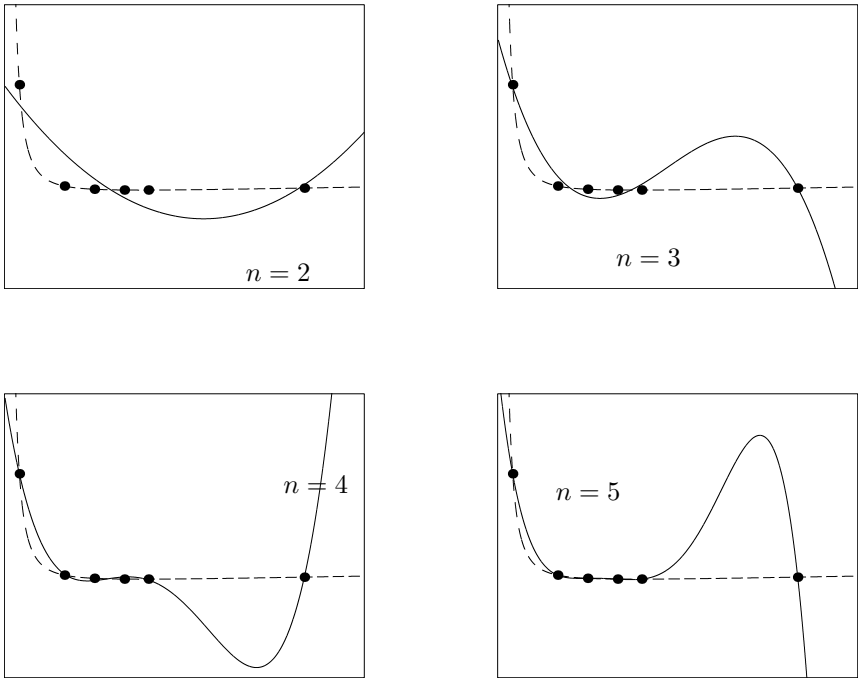


Figure 1.12: Approximations par moindres carrés de données discrètes. L'usage de polynômes de degré élevé peut également produire des résultats apparemment surprenants.

$u^h(x)$ sur un intervalle d'approximation $[a, b]$. On pourrait ainsi calculer les paramètres a_j de l'approximation par moindres carrés en minimisant l'intégrale du carré de l'écart $u(x) - u^h(x)$ en tout point de l'intervalle

$$J(a_0, a_1, \ldots, a_n) = \int_a^b \left(u(x) - \sum_{j=0}^{n} \phi_j(x)\ a_j \right)^2 dx.$$

Une telle manière de procéder fournira une approximation qui tentera de minimiser l'erreur de manière globale sur l'intervalle et qui ne traitera plus de manière privilégiée un certain nombre d'abscisses. Toutefois, une telle approche n'est réalisable que pour des fonctions $u(x)$ dont une expression est disponible et intégrable lors du calcul des a_j.

A nouveau, l'expression $J(a_0, a_1, , \ldots, a_n)$ est une fonction de $(n+1)$ variables réelles. Les valeurs des coefficients a_j qui minimisent cette fonction doivent donc vérifier les $(n+1)$ conditions :

$$
\begin{aligned}
\left.\frac{\partial J}{\partial a_k}\right|_{(a_0,\ldots,a_n)} &= 0 && k = 0, 1, \ldots, n \\
&\downarrow \\
\int_a^b -2\phi_k(x)\left(u(x) - \sum_{j=0}^{n} \phi_j(x)\ a_j\right) dx &= 0 && k = 0, 1, \ldots, n \\
&\downarrow \\
\sum_{j=0}^{n}\left(\int_a^b \phi_k(x)\phi_j(x)dx\right)\ a_j &= \int_a^b \phi_k(x)u(x)dx && k = 0, 1, \ldots, n
\end{aligned}
$$

Il s'agit à nouveau de résoudre un système linéaire. Ce qu'on peut écrire sous la forme matricielle suivante :

$$
\begin{bmatrix}
\int_a^b \phi_0(x)\phi_0(x)dx & \int_a^b \phi_0(x)\phi_1(x)dx & \ldots & \int_a^b \phi_0(x)\phi_n(x)dx \\
\int_a^b \phi_1(x)\phi_0(x)dx & \int_a^b \phi_1(x)\phi_1(x)dx & \ldots & \int_a^b \phi_1(x)\phi_n(x)dx \\
\int_a^b \phi_2(x)\phi_0(x)dx & \int_a^b \phi_2(x)\phi_1(x)dx & \ldots & \int_a^b \phi_2(x)\phi_n(x)dx \\
 & \vdots & & \vdots \\
\int_a^b \phi_n(x)\phi_0(x)dx & \int_a^b \phi_n(x)\phi_1(x)dx & \ldots & \int_a^b \phi_n(x)\phi_n(x)dx
\end{bmatrix}
\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} =
\begin{bmatrix}
\int_a^b \phi_0(x)u(x)dx \\
\int_a^b \phi_1(x)u(x)dx \\
\int_a^b \phi_2(x)u(x)dx \\
\vdots \\
\int_a^b \phi_n(x)u(x)dx
\end{bmatrix}
$$

Pour évaluer les expressions qui apparaissent dans la matrice et le membre de droite, le recours à des techniques d'intégration numérique s'impose fréquemment, en particulier, lorsque la fonction u est difficile à manipuler. Le problème général de l'approximation *d'une fonction* par moindres carrés sur un intervalle $[a, b]$ s'écrit :

Trouver $(a_0, \ldots, a_n) \in \mathbb{R}^{n+1}$ tels que

$$
\sum_{j=0}^{n}\left(\int_a^b \phi_k(x)\phi_j(x)dx\right)\ a_j = \int_a^b \phi_k(x)u(x)dx \qquad k = 0, 1, \ldots, n \tag{1.13}
$$

On observe que l'approximation aux moindres carrés d'un nuage de points peut être considérée comme un calcul lui-même très approximatif (en remplaçant une intégrale par la somme des valeurs pour quelques abscisses particulières) de l'approximation aux moindres carrés de la fonction correspondante.

1.2.3 L'approximation au sens des moindres carrés est une projection orthogonale...

Introduisons les notations suivantes :

$$\begin{aligned} < f, g > &= \int_a^b f g \ dx, \\ \|f\| &= \sqrt{\int_a^b f^2 \ dx} \end{aligned} \tag{1.14}$$

En définissant de manière adéquate l'ensemble $\mathcal{U}$, on peut montrer que l'expression $< ., . >$ définit un produit scalaire entre deux fonctions de cet ensemble et que $\|.\|$ est la norme qui est associée.

Il est dès lors possible d'interpréter l'approximation au sens des moindres carrés comme la projection orthogonale de u sur le sous-ensemble $\mathcal{U}^h \subset \mathcal{U}$. Il s'agit bien d'un problème de minimisation de la norme de l'écart $u - u^h$. Le problème (1.13) peut être écrit comme suit : on exige que l'écart, c'est-à-dire l'erreur $e^h(x) = u(x) - u^h(x)$, soit orthogonal à tout élément de la base de $\mathcal{U}^h$, c'est-à-dire à tout élément du sous-ensemble d'approximation.

$$\begin{aligned} < \widehat{u}^h \ , \ \underbrace{(u - u^h)}_{e^h} > &= 0, && \forall \widehat{u}^h \in \mathcal{U}^h, \\ & \downarrow \\ \int_a^b \phi_k(x) \left(u(x) - \sum_{j=0}^{n} \phi_j(x) \ a_j \right) dx &= 0, && k = 0, 1, \ldots, n \\ & \downarrow \\ \sum_{j=0}^{n} \left(\int_a^b \phi_k(x) \phi_j(x) dx \right) \ a_j &= \int_a^b \phi_k(x) u(x) dx && k = 0, 1, \ldots, n \end{aligned}$$

On retrouve exactement les équations du problème (1.13).

Choisir judicieusement les fonctions de base peut évidemment grandement simplifier le calcul de la projection orthogonale ou de l'approximation aux moindres carrés. On essaie ainsi de considérer des bases de fonctions orthogonales pour le produit scalaire considéré. Par exemple, choisissons comme fonctions de base pour un intervalle $[-\pi, \pi]$, la famille :

$$\phi_i(x) = \cos(ix) \qquad i = 1, 2, \ldots, n \tag{1.15}$$

On peut observer que les intégrales

$$\begin{aligned} \int_{-\pi}^{\pi} \cos(ix) \cos(jx) \ dx &= 0 && i \neq j \\ &= \pi && i = j \end{aligned}$$

et en déduire qu'une telle famille de n fonctions de base forme une base orthogonale. Pour une telle base, le calcul de la projection orthogonale peut être directement obtenu par les expressions

$$a_i = \frac{1}{\pi} \int_{-\pi}^{\pi} u(x) \cos(ix)\ dx \tag{1.16}$$

et il n'est plus nécessaire de résoudre un système algébrique pour obtenir l'approximation, puisque la matrice du système se réduit à une matrice diagonale !

Introduisons, maintenant, un second produit scalaire :

$$\begin{aligned} < f, g >_* &= \sum_{j=0}^{m} f(X_j) g(X_j), \\ \|f\|_* &= \sqrt{\sum_{j=0}^{m} f^2(X_j)}. \end{aligned} \tag{1.17}$$

Le problème (1.12) peut être écrit comme la recherche d'une projection orthogonale pour ce produit scalaire que nous venons de définir.

$$\begin{aligned} < \widehat{u}^h\ ,\ \underbrace{(u - u^h)}_{e^h} >_* &= 0, && \forall \widehat{u}^h \in \mathcal{U}^h, \\ & \downarrow \\ \sum_{i=0}^{m} \phi_k(X_i) \left(u(X_i) - \sum_{j=0}^{n} \phi_j(X_i)\ a_j \right) &= 0, && k = 0, 1, \ldots, n \\ & \downarrow \\ \sum_{j=0}^{n} \left(\sum_{i=0}^{m} \phi_k(X_i)\ \phi_j(X_i) \right)\ a_j &= \sum_{i=0}^{m} \phi_k(X_i) U_i && k = 0, 1, \ldots, n \end{aligned}$$

Nous avons bien ré-obtenu les équations (1.12).

1.2.4 Introduction aux NURBS

Les logiciels de conception assistée par ordinateur, d'animation sur ordinateur et les non moins célèbres jeux doivent représenter des courbes et des surfaces d'une complexité de plus en plus grande. Et cette complexité s'accompagne d'une exigence de rapidité dans la génération de ces surfaces et de ces courbes paramétrées. Ces contraintes ont amené le développement d'outils tels que les *courbes ou surfaces de Bézier* ou *B-splines rationnelles non-uniformes*. Dans cette section, nous allons rapidement introduire le concept de NURBS (en anglais, *Non-Uniform Rational B-Splines*).

Fonctions B-splines

Commençons par définir ce que l'on nomme des *B-splines*, ce que l'on appelle des fonctions de base, ou fonctions de pondération ou encore des fonctions de mélange. Nous allons utiliser t comme paramètre et considérer $n+1$ abscisses appelées noeuds :

$$T_0 \leq T_1 \leq T_2 \leq \ldots \leq T_n,$$

On observe donc que les noeuds sont classés par ordre croissant et que certains noeuds pourront être doubles ou triples. Un noeud T_i est de multiplicité m s'il est répété m fois. On définit ensuite les B-splines comme suit

Définition 1.4.

Soit les $n+1$ noeuds $T_0 \leq T_1 \leq T_2 \leq \ldots \leq T_n$.
Les B-splines de degré p sont les $n-p$ fonctions $B_i^p(t)$. Une fonction B_i^p est nulle sauf dans l'intervalle $[\ T_i, T_{i+1+p}\ [$ où elle est définie par la relation de récurrence :

$$B_i^p(t) = \frac{(t-T_i)}{(T_{i+p}-T_i)} B_i^{p-1}(t) + \frac{(T_{i+1+p}-t)}{(T_{i+1+p}-T_{i+1})} B_{i+1}^{p-1}(t)$$

avec $i = 0, \ldots, n-p-1$ et en partant de :

$$B_i^0(t) = \begin{cases} 1, & si \ \ t \in [T_i, T_{i+1}[\\ \\ 0, & ailleurs \end{cases}$$

On observe immédiatement que pour des noeuds de multiplicité supérieure à un, la formule de récurrence peut faire apparaître une valeur nulle à l'un ou l'autre dénominateur. On complète donc la définition en spécifiant qu'il ne faut tenir compte que des termes dont les dénominateurs ne s'annulent pas.

On peut aussi observer que la fonction B_i^p a des valeurs non nulles sur un intervalle englobant $p+2$ noeuds. Afin de mieux appréhender ces fonctions de base, considérons quelques exemples simples.

- Prenons comme vecteur de noeuds $\mathbf{T} = (0, 1, 2, 3)$. Nous avons donc trois fonctions

B_i^0, deux fonctions B_i^1 et une fonction B_i^2 : quatre de ces fonctions sont représentées sur la Figure 1.13.

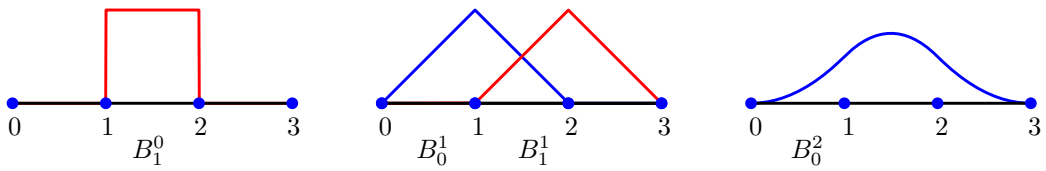


Figure 1.13: Quelques splines de base pour $\mathbf{T} = (0, 1, 2, 3)$.

- Considérons ensuite le vecteur de noeuds $(0, 1, 2, 3, 4, 5, 6, 7)$ et les $7 - 3 = 4$ splines de base de degré 3. On remarque que chaque fonction B_i^3 a un support de $3 + 2 = 5$ noeuds. Ces splines cubiques sont cependant différentes de celles introduites dans la section 1.1.6. Ensuite, illustrons le rôle des noeuds en considérant le cas de noeuds non équidistants et multiples sur la Figure 1.14.

- Comme dernier exemple, prenons le vecteur $\mathbf{T} = (0, 0, 0, 0, 1, 1, 1, 1)$ et les quatre splines de base qui y sont associées sur la Figure 1.15. Il n'y a que deux noeuds distincts de multiplicité 4. On obtient ainsi les fonctions de base de Bézier qui sont également connues sous le nom de polynômes de Bernstein.

On peut ainsi observer quelques propriétés pratiques des B-splines :

- Une spline $B_i^p(t)$ est toujours comprise entre 0 et 1.
- Une spline $B_i^p(t)$ ne peut être non nulle que dans l'intervalle $[\, T_i, T_{i+1+p} \,[$.
- Dans l'intervalle $[\, T_i, T_{i+1} \,[$, seules les splines $B_{i-p}^p(t), \ldots, B_i^p(t)$ sont non nulles.
- Une spline $B_i^p(t)$ ne vaut exactement 1 qu'en un noeud de multiplicité supérieure ou égale à p.

Nous allons finalement donner, sans les démontrer, deux théorèmes fondamentaux pour les fonctions B-splines.

Théorème 1.3.

A l'exclusion des p premiers et p derniers intervalles, la somme des B-splines vaut l'unité.

$$\sum_{i=0}^{n-p-1} B_i^p(t) = 1 \qquad\qquad T_p \leq t < T_{n-p}$$

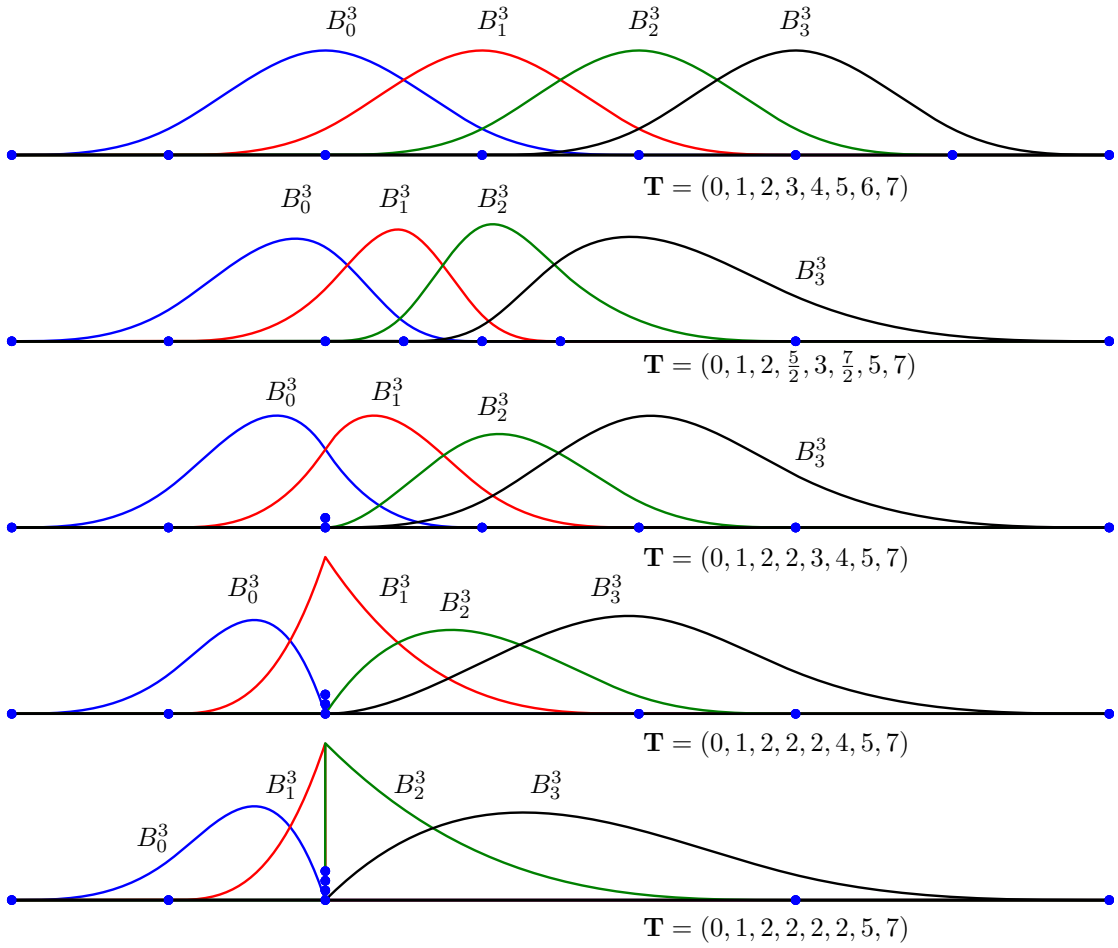


Figure 1.14: Splines de base cubiques $B_i^3(t)$ avec $i = 0, \ldots, 3$ pour différents vecteurs de noeuds.

Théorème 1.4.

Si le vecteur des noeuds est constitué uniquement de points de multiplicité un, les splines de base B_i^p sont $(p-1)$ fois dérivables.
On dit que l'ordre de continuité est $p-1$.
Par contre, à un point de multiplicité m, l'ordre de continuité n'y sera localement que de $p-m$.

Courbes composées de B-splines

On peut engendrer des courbes paramétrées dans le plan en se servant des splines de base que nous venons d'introduire. Pour un vecteur $\mathbf{T}$ de $n+1$ noeuds avec $n-p$ fonctions

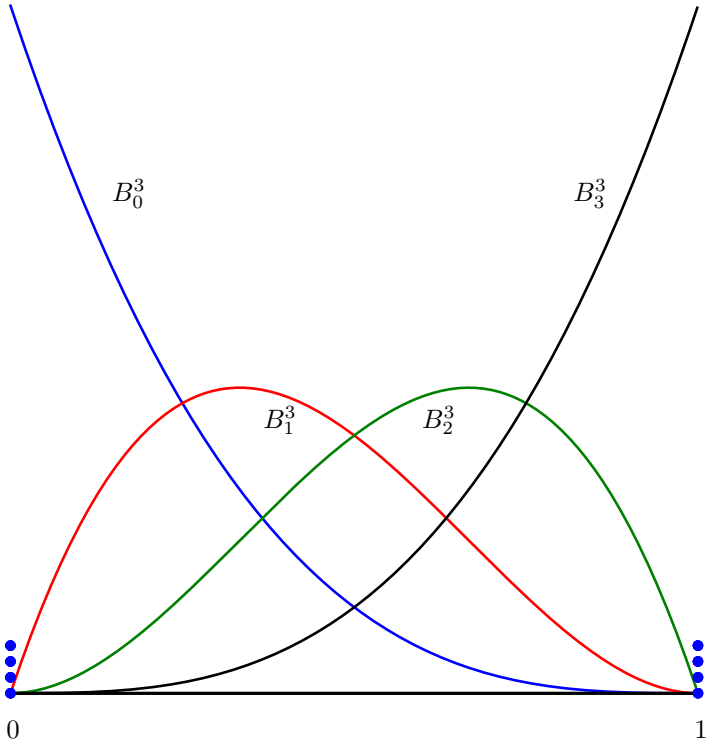


Figure 1.15: Splines de base cubiques pour $\mathbf{T} = (0, 0, 0, 0, 1, 1, 1, 1)$: on retrouve les splines de Bézier qui sont les polynômes de Bernstein.

de base de degré p, nous allons nous donner $n - p$ points de contrôle (X_i, Y_i) et engendrer une courbe définie par :

$$\begin{cases} x^h(t) = \sum_{i=0}^{n-p-1} B_i^p(t)\ X_i \\ y^h(t) = \sum_{i=0}^{n-p-1} B_i^p(t)\ Y_i \end{cases} \qquad T_p \leq t < T_{n-p} \tag{1.18}$$

En posant $\mathbf{u}^h(t) = (x^h(t), y^h(t))$ et $\mathbf{P}_i = (X_i, Y_i)$, la relation (1.18) devient :

$$\mathbf{u}^h(t) = \sum_{i=0}^{n-p-1} B_i^p(t)\ \mathbf{P}_i \qquad T_p \leq t < T_{n-p} \tag{1.19}$$

A titre d'exemple, considérons le vecteur de noeuds $\mathbf{T} = (0, 0, 0, 0, 1, 2, 2, 2, 2)$ et une liste de points de contrôle $\mathbf{P} = \{(0, 0); (1, 3); (2, 0); (3, 3); (4, 0)\}$. Il y a 9 noeuds ($n = 8$) et 5 points de contrôle : on va donc utiliser des splines de degré 3. Sur la Figure 1.16, on peut bien observer que la courbe passe par le premier et le dernier point, tout en étant en quelque sorte influencée par les autres points. Cette courbe tend à se rapprocher mais ne

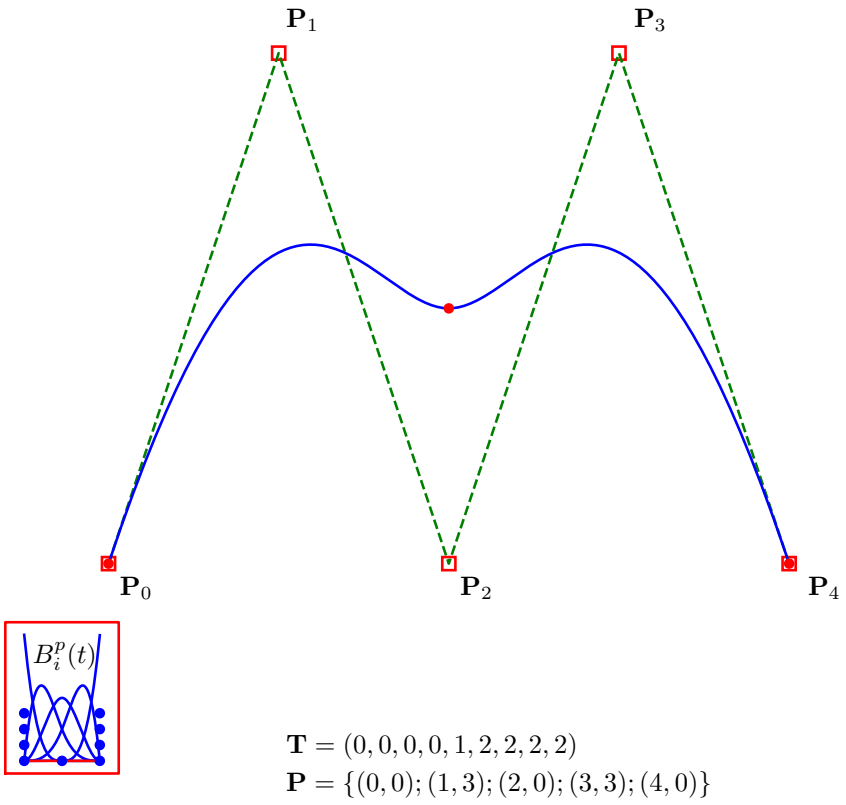


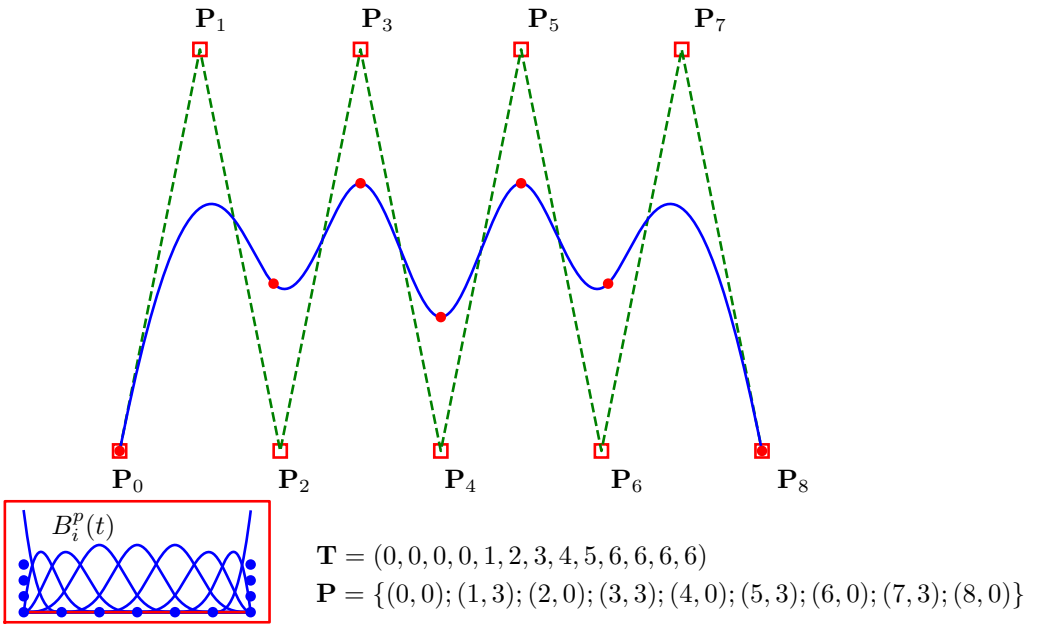
Figure 1.16: B-splines cubiques : les cercles indiquent les noeuds sur la courbe et le long de l'axe du paramètre t, tandis que les carrés représentent les points de contrôle.

passe pas par les points de contrôle : en ce sens, il ne s'agit pas d'une interpolation, mais d'une approximation de ces points. Nous allons utiliser quelques exemples pour mieux comprendre l'impact de points de contrôle et de noeuds multiples dans le cas de splines cubiques qui sont celles les plus utilisées dans les applications pratiques.

- L'utilisation de noeuds triples[8] aux extrémités du vecteur $\mathbf{T}$ permet d'imposer le passage par le premier et le dernier point de contrôle, ce qui n'est pas le cas lorsqu'il n'y a pas de noeuds multiples, comme illustré sur la Figure 1.17. Notons bien que la courbe résultante est obtenue en faisant varier le paramètre t entre T_p et T_{n-p} (et non, entre T_0 et T_n !). En général, on choisit le premier et le dernier noeud de multiplicité $p + 1$ afin que la courbe résultante passe par le premier et le dernier point de contrôle. Les noeuds et les points de contrôle peuvent être répétés. On peut aussi construire des courbes paramétrées dans l'espace de cette façon.

- L'utilisation d'un point de contrôle double va forcer la courbe à se rapprocher net-

[8]Les puristes me feront remarquer que sur la Figure 1.17, la multiplicité est quatre et non trois. C'est exact, mais seule une multiplicité triple est vraiment requise pour forcer le passage par un point. Ajouter un quatrième noeud permet juste de couper la fonction de base en ce noeud, mais ne modifie en rien la forme de la courbe sur le domaine d'intérêt.



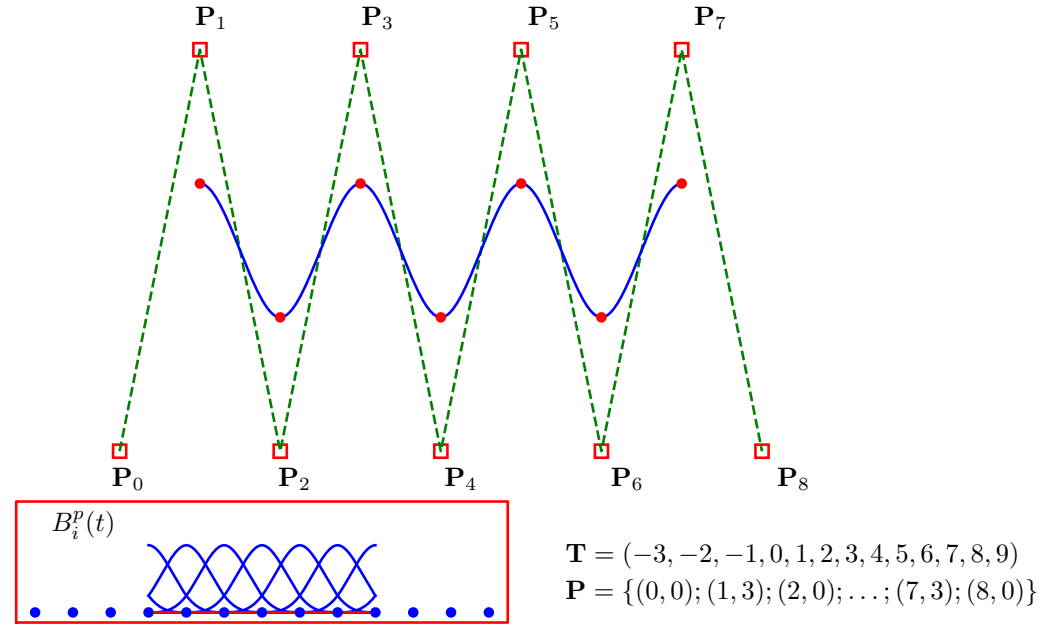


Figure 1.17: B-splines cubiques : impact de noeuds multiples ou simples aux extrémités.

tement plus de celui-ci. Et la courbe passera exactement par un point de contrôle triple. Toutefois, l'ordre de continuité de la courbe sera réduit en ces points : on voit clairement apparaître un point anguleux au passage du point de contrôle triple sur la Figure 1.18.

- Sur la Figure 1.19, on obtient un effet semblable en utilisant des noeuds multiples. L'utilisation de noeuds doubles ou triples permet ainsi également de renforcer le poids de certaines données ou de forcer le passage par un point de contrôle. L'effet n'est toutefois pas identique à celui créé par des points de contrôle multiples. Comme l'espace mémoire est souvent un facteur critique, il est plus intéressant d'introduire des noeuds scalaires multiples que des points de contrôle vectoriels (deux composantes dans le plan, trois composantes dans l'espace). C'est le choix effectué, en général, dans les logiciels de CAO et d'animation.

B-splines rationnelles non-uniformes

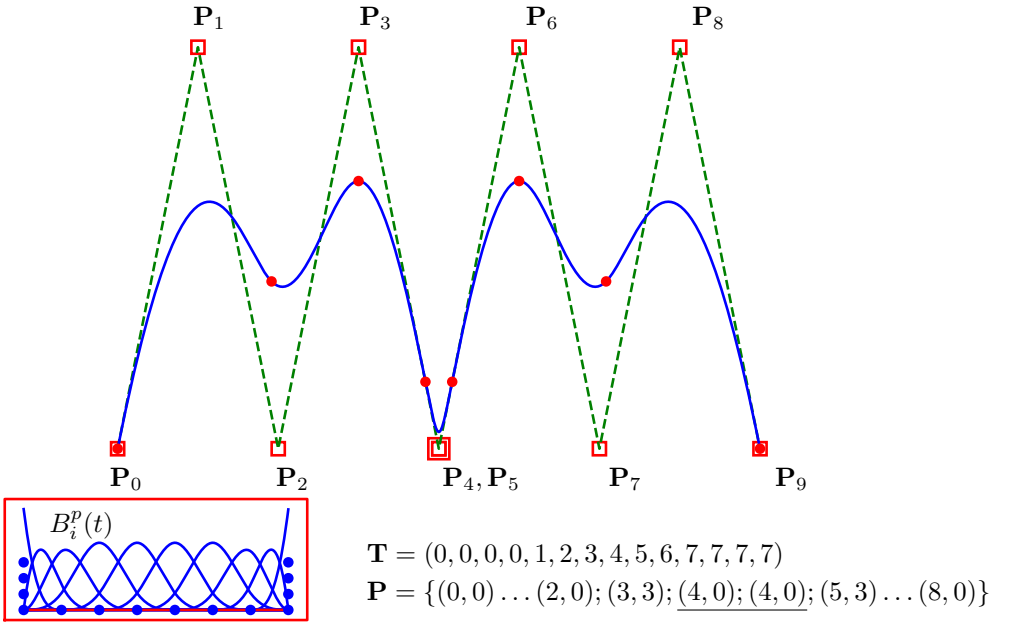
Les B-splines rationnelles non uniformes (ou NURBS) sont, comme leur nom l'indique, un quotient de courbes B-splines. Le qualificatif non-uniforme provient du fait que les noeuds du vecteur $\mathbf{T}$ ne sont pas forcément équidistants. Une des raisons principales de l'introduction des NURBS est qu'il n'est pas possible de représenter exactement des coniques (cercles, ellipses et hyperboles) avec des courbes B-splines. On comprend alors les limitations pour un logiciel de dessin de ne pouvoir représenter de manière exacte un cercle !

On a toujours un vecteur $\mathbf{T}$ de $n+1$ noeuds avec $n-p$ fonctions de base de degré p, nous allons nous donner $n-p$ points de contrôle $\mathbf{P}_i$ dans le plan ou dans l'espace et $n-p$ poids W_i et engendrer une courbe définie par :

$$\mathbf{u}^h(t) = \frac{\sum_{i=0}^{n-p-1} W_i B_i^p(t)\ \mathbf{P}_i}{\sum_{k=0}^{n-p-1} W_k B_k^p(t)} \qquad T_p \leq t < T_{n-p} \tag{1.20}$$

On peut immédiatement observer que si tous les poids sont égaux, on peut les éliminer et l'on retombe sur une courbe B-spline, puisque le dénominateur vaut l'unité en vertu du Théorème 1.3. Intuitivement, plus un poids est important de manière relative par rapport aux autres, plus la courbe résultante passera près du point de contrôle correspondant : c'est un peu ce que l'on pouvait obtenir en modifiant la position de noeuds. On voit donc que la définition d'une NURBS est parfois délicate puisque plusieurs paramètres fournissent plus ou moins le même effet. Mais, ce nombre plus élevé de paramètres permet également d'agrandir l'espace des courbes que l'on peut engendrer et qui englobe maintenant les coniques !

Puisqu'une des raisons de l'introduction des NURBS est la possibilité de pouvoir construire des coniques, il est légitime de présenter de tels exemples. Sur la Figure 1.20,



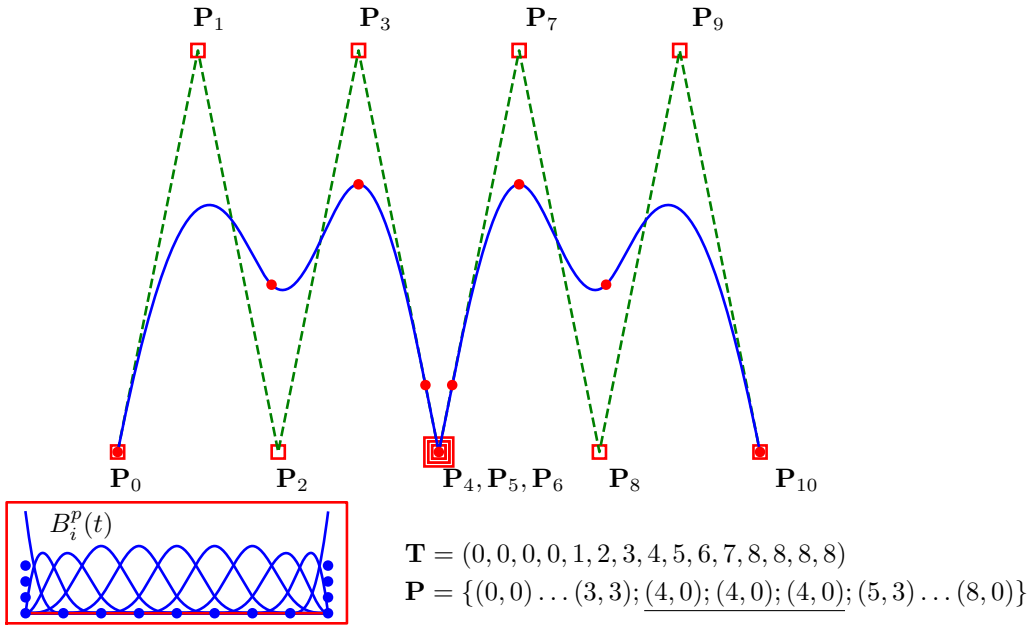
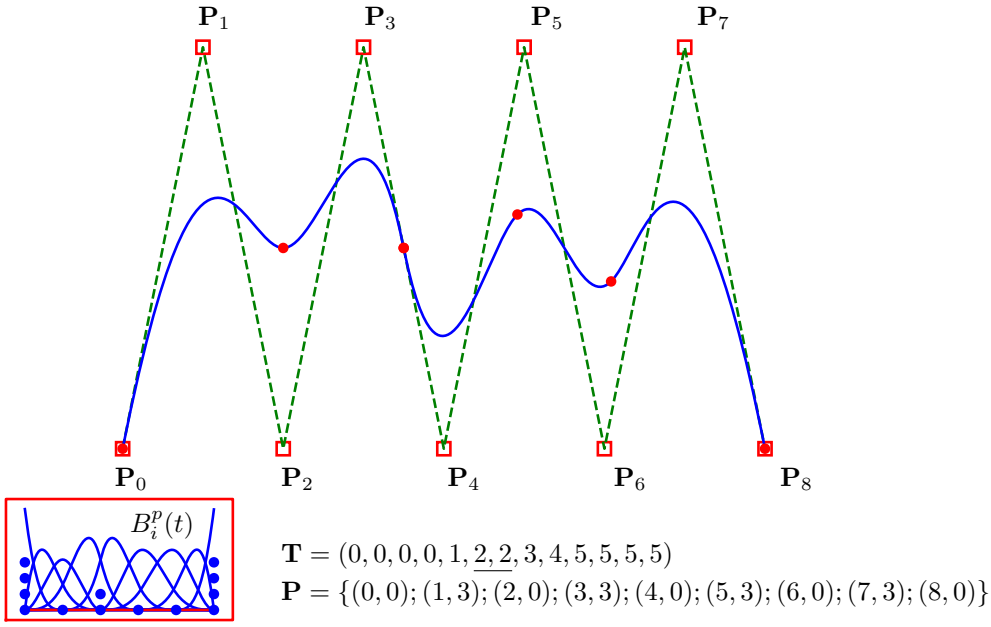


Figure 1.18: B-splines cubiques : effet de points de contrôle multiples à l'intérieur de l'intervalle.



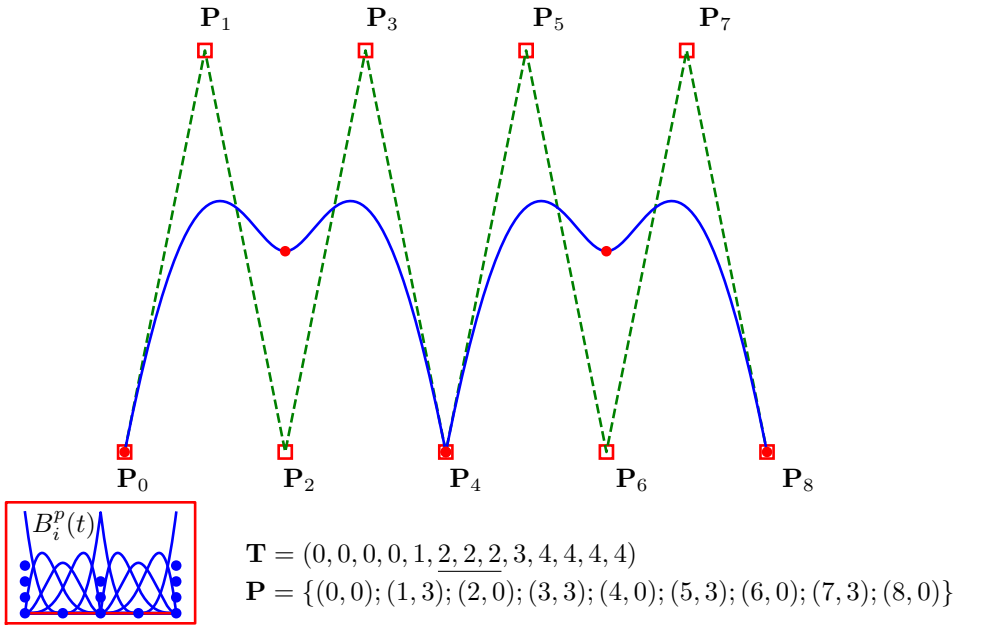


Figure 1.19: B-splines cubiques : effet de noeuds multiples à l'intérieur de l'intervalle.

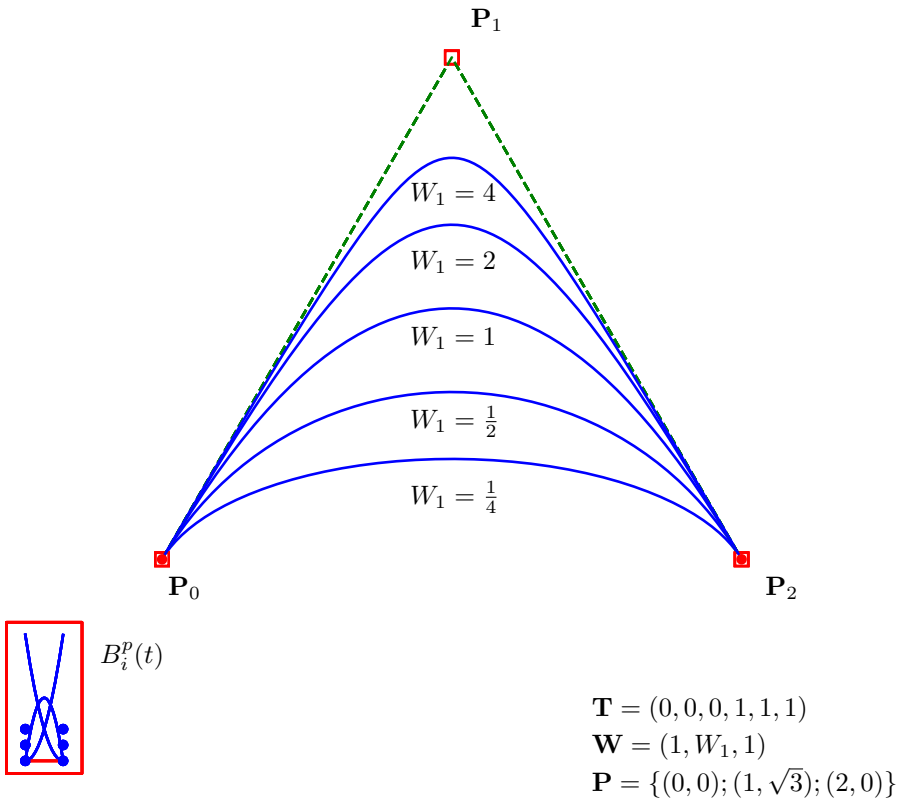


Figure 1.20: Arcs de coniques générés par des NURBS de degré deux pour différents vecteurs de poids.

une NURBS de degré deux génère respectivement un arc d'ellipse ($0 \leq W_1 < 1$), un arc de cercle ($W_1 = \frac{1}{2}$), un arc de parabole ($W_1 = 1$) ou un arc d'hyperbole ($1 < W_1$) en choisissant comme points de contrôle les sommets d'un triangle équilatéral. L'obtention d'un cercle complet est réalisée sur la Figure 1.21.

Notons que l'on peut écrire 1.20 sous la forme suivante :

$$\mathbf{u}^h(t) = \sum_{i=0}^{n-p-1} \frac{W_i B_i^p(t)}{\sum_{k=0}^{n-p-1} W_k B_k^p(t)} \mathbf{P}_i$$

$$\text{En définissant } \phi_i = \frac{W_i B_i^p(t)}{\sum_{k=0}^{n-p-1} W_k B_k^p(t)}$$

$$\mathbf{u}^h(t) = \sum_{i=0}^{n-p-1} \phi_i(t)\, \mathbf{P}_i$$

et retrouver la formule générique de l'approximation. Cela ressemble à l'interpolation de

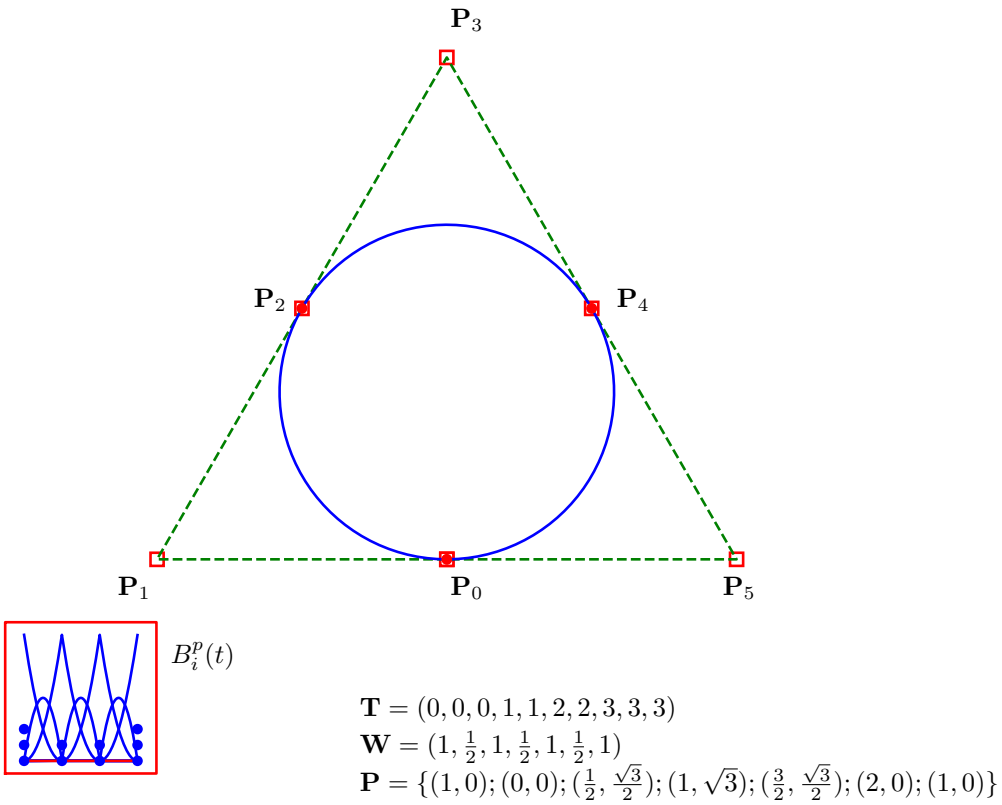


Figure 1.21: Cercle généré par des NURBS de degré deux.

Lagrange, mais rappelons que la NURBS ne passe pas forcément par tous les points de contrôle.

Surfaces construites à partir de B-splines

Pour construire une surface, on se donne un vecteur $\mathbf{T}$ de $n+1$ noeuds avec $n-p$ fonctions de base de degré p et un second vecteur $\mathbf{S}$ de $m+1$ noeuds avec $m-q$ fonctions de base de degré q. Ensuite, on définit un tableau de $(n-p) \times (m-q)$ points de contrôle $\mathbf{P}_{ij}$ dans le plan ou dans l'espace et $(n-p) \times (m-q)$ poids W_{ij} correspondants. On peut alors engendrer une surface définie par :

$$\mathbf{u}^h(t,s) = \frac{\sum_{i=0}^{n-p-1} \sum_{j=0}^{m-q-1} W_{ij} B_i^p(t) B_j^q(s)\ \mathbf{P}_{ij}}{\sum_{k=0}^{n-p-1} \sum_{l=0}^{m-q-1} W_{kl} B_k^p(t) B_l^q(s)} \qquad (1.21)$$

$$T_p \leq t < T_{n-p}$$
$$S_q \leq s < S_{m-q}$$

La surface est obtenue par un produit cartésien de deux courbes l'une de degré p en t et l'autre de degré q en s. Dans de nombreux cas, on choisira $p = q$, mais il est parfois utile de se laisser la possibilité d'utiliser des degrés distincts dans chaque direction. Imposer que les premiers noeuds des vecteurs $\mathbf{T}$ et $\mathbf{S}$ soient de multiplicité $p+1$ et $q+1$ respectivement assure que la surface résultante passera bien par les points de contrôle situés aux quatre coins de la surface.

$$\mathbf{T} = \mathbf{S} = (0, 0, 0, 1, 1, 1)$$

$$\mathbf{P} = \begin{bmatrix} (1,1,1) & (2,1,2) & (3,1,3) \\ (1,2,2) & (2,2,3) & (3,2,1) \\ (1,3,3) & (2,3,1) & (3,3,4) \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

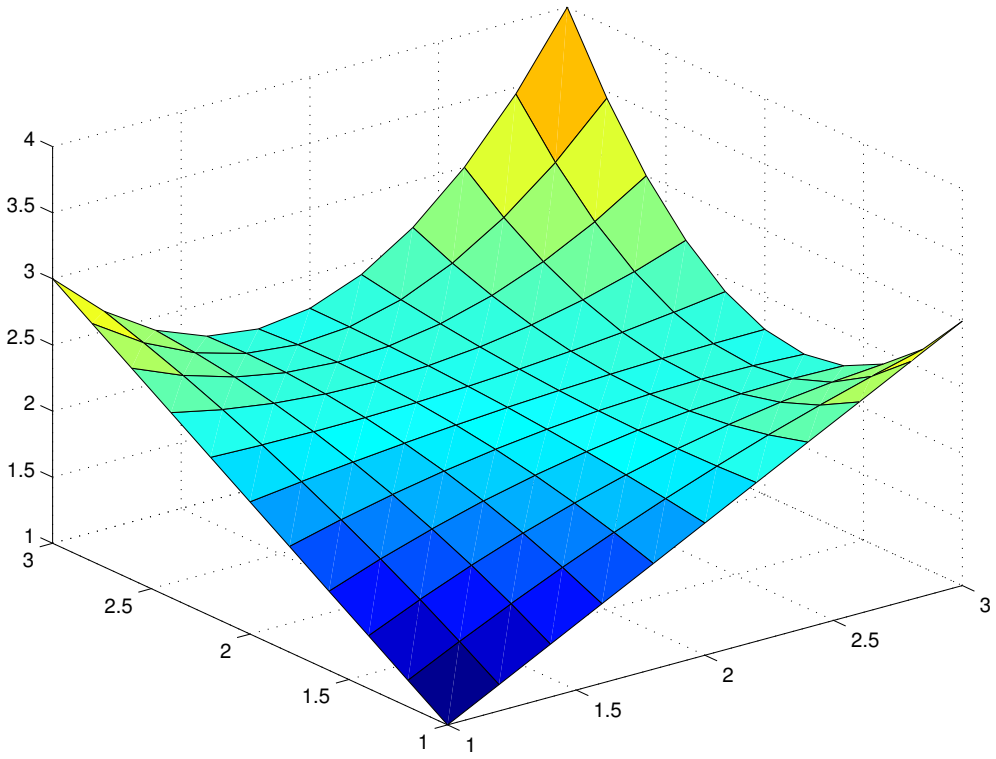


Figure 1.22: Surface générée par des NURBS de degré deux et 9 points de contrôle et 9 poids égaux.

Chapitre 2

Comment intégrer numériquement une fonction ?

Le problème abordé dans ce chapitre est celui du calcul de l'intégrale définie

$$I = \int_a^b u(x)\,dx$$

Le calcul de cette intégrale demande la connaissance de la primitive de $u(x)$. Il se peut que cette primitive soit trop compliquée à calculer et que les méthodes analytiques classiques soient trop lourdes ou impossibles à mettre en oeuvre. Dans certains cas, il n'y a même pas moyen d'exprimer analytiquement la primitive de la fonction $u(x)$ et il est alors indispensable de se tourner vers des méthodes d'intégration numérique.

L'intégration numérique consiste à estimer l'intégrale définie I en évaluant $u(x)$ en un nombre fini de points $a \leq X_0 < X_1 < X_2 < \ldots < X_m \leq b$ et en effectuant une somme pondérée des valeurs $u(X_i)$. Nous cherchons donc une approximation de l'intégrale définie sous la forme suivante :

$$I \approx I^h = \sum_{i=0}^{m} w_i\, u(X_i) \tag{2.1}$$

où les w_i sont appelés les *poids* et les X_i les *points* ou les *abscisses d'intégration*. Une telle formule est appelée règle d'intégration ou *quadrature*. Les méthodes les plus simples utilisent des points X_i équidistants, mais ce n'est pas toujours la stratégie la plus efficace. Lorsque $X_0 = a$ et $X_m = b$, la méthode est dite *fermée*. Dans le cas contraire, il s'agit d'une méthode dite *ouverte*.

La différence entre l'intégrale exacte et la valeur approchée $E^h = I - I^h$ est appelée *l'erreur de discrétisation* ou *l'erreur d'intégration*. Le *degré de précision* d'une méthode d'intégration numérique est le nombre entier positif d tel que l'erreur de discrétisation

soit nulle pour tous les polynômes de degré inférieur ou égal à d mais soit non nulle pour au moins un polynôme de degré $d+1$.

Sans perte de généralité, nous pouvons toujours ramener l'intervalle d'intégration $[a,b]$ à un intervalle standardisé, généralement $[-1,1]$, pour lequel les abscisses et les poids d'intégration pourront être tabulés. Pour ce faire, il suffit d'effectuer le changement de variable

$$x(\xi) = \frac{(b-a)}{2}\,\xi \;+\; \frac{(b+a)}{2} \tag{2.2}$$

On a donc

$$I \;=\; \int_a^b u(x)\,dx \;=\; \int_{-1}^{1} u(x(\xi))\frac{dx}{d\xi}d\xi \;=\; \underbrace{\int_{-1}^{1} u(x(\xi))\,d\xi}_{\approx \sum_{i=0}^{m} w_i\, u(x(\Xi_i))} \;\frac{(b-a)}{2}$$

Par la suite, nous nous limiterons donc souvent à des intégrales sur l'intervalle $[-1,+1]$.

2.1 Méthodes à pas égaux : Newton-Cotes

Considérons, d'abord, le cas d'abscisses équidistantes. Une manière relativement naturelle de construire des règles d'intégration est de partir de l'interpolation polynomiale. Rappelons qu'il existe un unique polynôme de degré m passant par les $m+1$ points distincts $(X_i, u(X_i))$. En utilisant la base des polynômes de Lagrange, on peut donc écrire :

$$\begin{aligned} I \;&\approx\; \overbrace{\int_{-1}^{1} u^h(x)dx}^{I^h} \\ &\downarrow \\ &\approx\; \int_{-1}^{1} \sum_{i=0}^{m} u(X_i)\phi_i(x)dx \\ &\downarrow \\ &\approx\; \sum_{i=0}^{m} u(X_i) \underbrace{\int_{-1}^{1} \phi_i(x)dx}_{w_i} \end{aligned}$$

Notons que les poids w_i ainsi définis sont totalement indépendants de la fonction u et qu'ils peuvent donc être calculés a priori. Avec des abscisses équidistantes comprenant

les extrémités, on obtient ainsi les *méthodes fermées de Newton-Cotes*, en utilisant des polynômes d'interpolation de degré 1, 2, 3 et 4.

$$
\begin{array}{|l|}
\hline
\int_{-1}^{1} u(x)dx \approx U_{-1} + U_1 \\
\qquad\qquad \text{(méthode des trapèzes)} \quad d = 1 \\
\hline
\int_{-1}^{1} u(x)dx \approx \frac{1}{3} U_{-1} + \frac{4}{3} U_0 + \frac{1}{3} U_1 \\
\qquad\qquad \text{(méthode de Simpson)} \quad d = 3 \\
\hline
\int_{-1}^{1} u(x)dx \approx \frac{1}{4} U_{-1} + \frac{3}{4} U_{-1/3} + \frac{3}{4} U_{1/3} + \frac{1}{4} U_1 \\
\qquad\qquad \text{(méthode de Simpson } \frac{3}{8}\text{)} \quad d = 3 \\
\hline
\int_{-1}^{1} u(x)dx \approx \frac{7}{45} U_{-1} + \frac{32}{45} U_{-1/2} + \frac{12}{45} U_0 + \frac{32}{45} U_{1/2} + \frac{7}{45} U_1 \\
\qquad\qquad \text{(méthode de Boole)} \quad d = 5 \\
\hline
\end{array}
\tag{2.3}
$$

où U_i dénote $u(i)$ et d est le degré de précision de la méthode. L'interprétation graphique sur la Figure 2.1 de ces méthodes est évidente.

Cette approche peut évidemment être étendue pour produire des règles d'intégration avec des degrés de précision *théoriques* de plus en plus élevés. On continuerait de la sorte à augmenter la précision (et la complexité) des formules. Un inconvénient apparaît toutefois peu à peu : les poids qui apparaissent dans les formules successives sont de plus en plus déséquilibrés, certains pouvant même devenir négatifs à partir d'un certain moment. Cette disparité croissante des poids entraîne de plus grands risques d'erreurs d'arrondi.

Le calcul des poids w_i à partir des fonctions de base est une opération relativement laborieuse, mais simple. A titre d'illustration, nous calculons ici les poids de la méthode de Simpson.

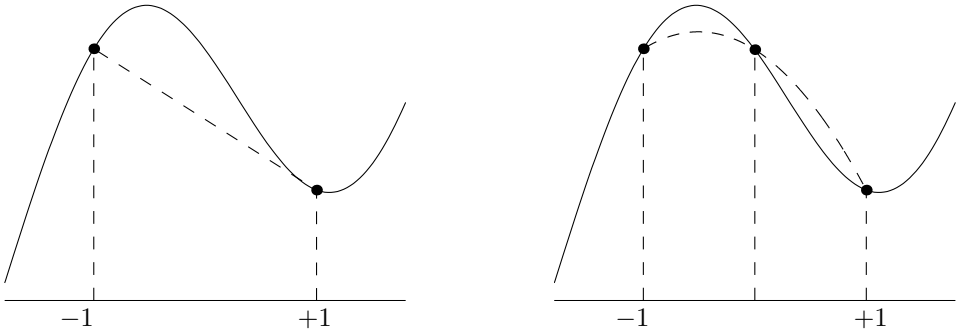


Figure 2.1: Interprétation géométrique de la méthode des trapèzes et de la méthode de Simpson.

$$
\begin{aligned}
\int_{-1}^{1} u(x)dx \;&\approx\; \overbrace{\int_{-1}^{1} u^h(x)dx}^{I^h} \\
&\downarrow \\
&\approx\; \sum_{i=0}^{2} u(X_i) \underbrace{\int_{-1}^{1} \phi_i(x)dx}_{w_i} \\
&\downarrow \quad \text{En fixant } X_0 = -1,\ X_1 = 0 \text{ et } X_2 = 1 \\
&\approx\; U_{-1} \underbrace{\int_{-1}^{1} \frac{(x-1)x}{2}\, dx}_{w_0} \\
&\qquad + U_0 \underbrace{\int_{-1}^{1} \frac{(x-1)(x+1)}{-1}\, dx}_{w_1} \\
&\qquad + U_1 \underbrace{\int_{-1}^{1} \frac{(x+1)x}{2}\, dx}_{w_2}
\end{aligned}
$$

Les trois poids sont immédiatement obtenus en intégrant les trois fonctions de base de Lagrange :

$$
\begin{aligned}
w_0 &= \int_{-1}^{1} \frac{(x-1)x}{2}\, dx &= \left[\frac{x^3}{6} - \frac{x^2}{4}\right]_{-1}^{1} &= \frac{1}{3} \\
w_1 &= \int_{-1}^{1} 1 - x^2\, dx &= \left[x - \frac{x^3}{3}\right]_{-1}^{1} &= 2 - \frac{2}{3} = \frac{4}{3} \\
w_2 &= \int_{-1}^{1} \frac{(x+1)x}{2}\, dx &= \left[\frac{x^3}{6} + \frac{x^2}{4}\right]_{-1}^{1} &= \frac{1}{3}
\end{aligned}
$$

où nous pouvons observer que l'intégration de toutes les puissances impaires d'un polynôme sur un intervalle symétrique produit systématiquement une contribution nulle. Cette observation explique pourquoi le degré de précision de la méthode de Simpson est trois, alors que l'on utilise seulement une parabole pour construire une interpolation de u.

Ce résultat a priori surprenant peut s'expliquer comme suit. Séparons les termes pairs et impairs d'un polynôme quelconque de degré impair $2n+1$ et observons que la contribution des puissances impaires dans une intégrale sur un intervalle symétrique est toujours nulle.

$$
\begin{aligned}
\int_{-1}^{1} \sum_{i=0}^{2n+1} a_i\, x^i\, dx &= \int_{-1}^{1} \sum_{j=0}^{n} a_{2j}\, x^{2j} dx + \underbrace{\int_{-1}^{1} \sum_{j=0}^{n} a_{2j+1}\, x^{2j+1}\, dx}_{=0} \\
&= \int_{-1}^{1} \sum_{i=0}^{2n} a_i\, x^i\, dx
\end{aligned}
$$

En d'autres mots, l'intégrale du polynôme de degré $2n+1$ sera strictement égale à celle du polynôme de degré $2n$.

En conclusion, la méthode de Simpson qui permet d'intégrer de manière exacte n'importe quel polynôme de degré deux, fournira également un résultat exact pour n'importe quel polynôme de degré trois puisque l'intégrale du terme de degré trois est nulle. De façon générale, on a donc démontré qu'une formule symétrique développée pour un degré de précision d pair présente en fait les caractéristiques d'une méthode de degré de précision $d+1$. La formule de Simpson est donc une formule de degré de précision 3. On constate aussi l'intérêt purement théorique de la seconde méthode de Simpson avec quatre abscisses.

2.1.1 Méthodes composites et intervalles juxtaposés

Une manière très intuitive de généraliser ce qui précède est de découper notre intervalle d'intégration en sous-intervalles et d'appliquer sur chaque sous-intervalle une des méthodes que nous venons de présenter. Une telle idée est à la base de ce qu'on appelle les méthodes composites sur des intervalles juxtaposés.

Pour présenter ces méthodes, il est commode d'écrire les relations (2.3), non plus pour un domaine d'intégration standardisé $[-1, 1]$, mais en fonction de l'écart h entre deux abscisses voisines. En faisant usage du changement de variable (2.2), on obtient immédiatement :

$$
\begin{array}{|l|}
\hline
\displaystyle\int_{-h/2}^{h/2} u(x)dx \;\approx\; \tfrac{h}{2}\,\left(U_{-h/2} + \; U_{h/2}\right) \\
\qquad\qquad\qquad\qquad\qquad\qquad \text{(méthode des trapèzes)} \quad d = 1 \\
\hline
\displaystyle\int_{-h}^{h} u(x)dx \;\approx\; \tfrac{h}{3}\,(U_{-h} + 4\;U_0 + \; U_h) \\
\qquad\qquad\qquad\qquad\qquad\qquad \text{(méthode de Simpson)} \quad d = 3 \\
\hline
\displaystyle\int_{-2h}^{2h} u(x)dx \;\approx\; \tfrac{2h}{45}\,(7\;U_{-2h} + 32\;U_{-h} + 12\;U_0 + 32\;U_h + 7\;U_{2h}) \\
\qquad\qquad\qquad\qquad\qquad\qquad \text{(méthode de Boole)} \quad d = 5 \\
\hline
\end{array}
\tag{2.4}
$$

Il est important de réaliser que ces formules d'intégration sont maintenant appliquées pour différents intervalles. En d'autres mots, pour un même intervalle d'intégration, on utilisera $h = (b - a)$ pour la méthode des trapèzes, $h = (b - a)/2$ pour la méthode de Simpson et $h = (b - a)/4$ pour la méthode de Boole. Ce choix a été effectué afin que h corresponde toujours à l'écart entre deux abscisses d'intégration, écart qui est le paramètre naturel de la discrétisation utilisée pour bâtir une règle d'intégration numérique.

Méthode composite des trapèzes

Une manière très intuitive de calculer l'aire sous une courbe en utilisant un grand nombre d'abscisses est obtenue en additionnant les surfaces d'une série de trapèzes placés côte à côte sous la courbe. Cette idée correspond à la *méthode composite des trapèzes*. Nous divisons l'intervalle d'intégration en n sous-intervalles et nous appliquons à chaque sous-intervalle la méthode des trapèzes présentée plus haut. Ceci est illustré à la Figure 2.2.

Partageons l'intervalle d'intégration en n sous-intervalles égaux $[X_{i-1}, X_i]$. La longueur de chaque sous-intervalle sera égale à $X_i - X_{i-1}$ et vaudra h que l'on définit toujours comme l'écart entre deux abscisses d'intégration.

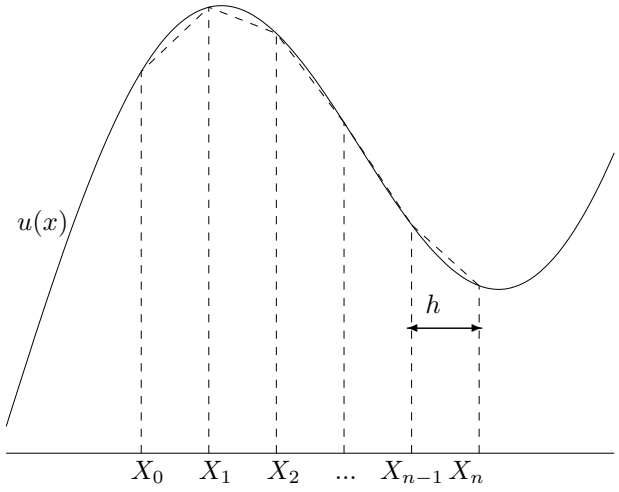


Figure 2.2: Méthode composite des trapèzes - Intervalles juxtaposés.

$$
\begin{aligned}
I &= \int_{X_0}^{X_n} u(x)\ dx \\
&= \sum_{i=1}^{n} \int_{X_{i-1}}^{X_i} u(x)\ dx \\
&\quad \Big\downarrow \quad \text{En utilisant la règle des trapèzes (2.4)} \\
&\quad \quad \text{pour chaque sous-intervalle} \\
I^h &= \sum_{i=1}^{n} \frac{h}{2}\ (U_{i-1} + U_i) \\
&= \tfrac{h}{2}\,(U_0 + 2U_1 + 2U_2 + \ldots + 2U_{n-1} + U_n)
\end{aligned}
$$

Méthode composite de Simpson

Partageons l'intervalle d'intégration en n sous-intervalles égaux $[X_{2i-2}, X_{2i}]$ comprenant chacun trois abscisses. Il y a donc un nombre total de $2n+1$ abscisses. La longueur de chaque sous-intervalle sera égale à $X_{2i} - X_{2i-2}$ et vaudra $2h$.

$$
\begin{aligned}
I &= \int_{X_0}^{X_{2n}} u(x)\, dx \\
&= \sum_{i=1}^{n} \int_{X_{2i-2}}^{X_{2i}} u(x)\, dx \\
&\quad \Big\downarrow \quad \text{En utilisant la règle de Simpson (2.4)} \\
&\quad \quad \text{pour chaque sous-intervalle} \\
I^h &= \sum_{i=1}^{n} \frac{h}{3}\ (U_{2i-2} + 4\ U_{2i-1} + \ U_{2i}) \\
&= \tfrac{h}{3}\,(U_0 + 4U_1 + 2U_2 + 4U_3 + 2U_4 + \ldots + 4U_{2n-1} + U_{2n})
\end{aligned}
$$

Ce qui précède peut facilement être généralisé pour n'importe quelle méthode d'intégration définie sur un intervalle.

2.1.2 Estimation de l'erreur de discrétisation

A titre d'illustration, nous allons estimer l'erreur de discrétisation de la méthode composite de Simpson avec $2n+1$ abscisses équidistantes d'un écart h.

Considérons donc un sous-intervalle $[-h, h]$ de longueur $2h$. Le paramètre h est l'écart entre les points d'intégration X_i. Nous venons de voir que la méthode de Simpson intègre de manière parfaitement exacte un polynôme de degré 3 sur chaque sous-intervalle sur lequel nous utilisons 3 abscisses d'intégration. Paradoxalement, nous avons donc trop peu de points d'intégration pour identifier le polynôme d'interpolation dont l'intégration correspondrait à la valeur obtenue par la formule de Simpson.

Pour obtenir un tel polynôme, nous allons considérer une quatrième abscisse totalement arbitraire α. Nous pouvons alors obtenir un polynôme passant par les abscisses $-h$, h, 0 et α et dire que c'est l'intégration de ce polynôme qui est fournie par la formule de Simpson. Il s'agira ensuite d'intégrer l'erreur d'interpolation commise par ce polynôme de degré 3, sur chaque sous-intervalle de taille $[-h, h]$. L'erreur d'interpolation est alors fournie par le théorème (1.1) :

$$
e^h(x) \;=\; \frac{u^{(4)}(\xi)}{4!}\,(x-h)x(x+h)(x-\alpha) \tag{2.5}
$$

où ξ est un point particulier de l'intervalle $]-h, h[$.

L'erreur sur un sous-intervalle est donnée par l'intégrale de cette erreur d'interpolation que l'on décompose en deux parties dont une est nulle en raison de la symétrie. Pour obtenir l'erreur d'une méthode composite avec n sous-intervalles, il suffit d'écrire :

$$
\begin{aligned}
E^h &= \sum_{i=1}^{n} \int_{-h}^{h} \frac{u^{(4)}(\xi_i)}{4!}\,(x-h)x(x+h)(x-\alpha)\;dx \\
&\Big\downarrow \quad \text{En définissant } C_4 = \max_i \left|u^{(4)}(\xi_i)\right| \\
\left|E^h\right| &\leq n\frac{C_4}{4!}\left|\int_{-h}^{h}(x-h)x(x+h)(x-\alpha)\;dx\right| \\
&\leq n\frac{C_4}{4!}\left|\int_{-h}^{h} x^2(x-h)(x+h)\,dx - \alpha \underbrace{\int_{-h}^{h}(x-h)x(x+h)\,dx}_{=0}\right| \\
&\leq n\frac{C_4}{4!}\left|\left[\frac{x^5}{5} - \frac{h^2x^3}{3}\right]_{-h}^{h}\right| \\
&\leq n\frac{C_4}{90}h^5 \\
&\Big\downarrow \quad \text{Car } 2nh = (b-a) \text{ pour une méthode composite de Simpson} \\
&\leq (b-a)\frac{C_4}{180}h^4
\end{aligned}
$$

Observons que n et h sont deux paramètres liés dans la caractérisation de la procédure numérique. Il est donc essentiel d'en éliminer un en fonction de l'autre. Ici, nous avons choisi d'exprimer l'erreur de discrétisation en termes de h, mais il aurait été tout aussi légitime de le faire en termes de n. Il est important de bien distinguer la longueur des sous-intervalles $(b-a)/n$ et l'écart $h = (b-a)/2n$ entre les abscisses. Ceci est illustré à la Figure 2.3.

Finalement, s'il est possible de trouver une estimation C_4 qui borne la valeur absolue de la dérivée quatrième de $u(x)$ sur l'intervalle d'intégration, on peut alors écrire une borne supérieure de la valeur absolue de l'erreur commise par la méthode de Simpson sous la forme :

$$
\boxed{|E^h| \quad \leq \quad \frac{C_4(b-a)}{180}\;h^4} \tag{2.6}
$$

Il est évidemment possible d'appliquer exactement la même procédure pour obtenir des bornes de l'erreur des autres méthodes de Newton-Cotes et d'obtenir pour un même écart h des abscisses d'intégration, les relations suivantes :

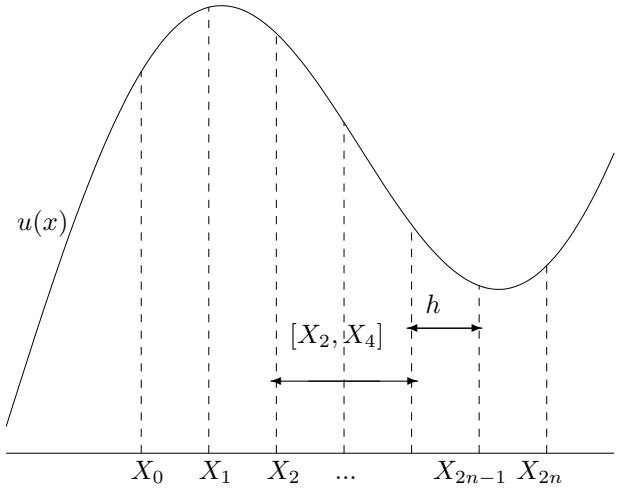


Figure 2.3: Méthode composite de Simpson. Il y a n intervalles juxtaposés de longueur $2h$ et $2n+1$ abscisses d'intégration dont l'écart est h.

$$
\begin{array}{|rcll|}
\hline
\displaystyle\int_{-h/2}^{h/2} u(x)dx & = & \frac{h}{2}\left(U_{-h/2} + U_{h/2}\right) - \frac{h^3}{12}u^{(2)}(\xi) & \\
|E^h| & \leq & \dfrac{C_2(b-a)}{12}\, h^2 \qquad \text{(méthode des trapèzes)} & \mathcal{O}(h^2) \\
\hline
\displaystyle\int_{-h}^{h} u(x)dx & = & \frac{h}{3}\left(U_{-h} + 4\,U_0 + U_h\right) - \frac{h^5}{90}u^{(4)}(\xi) & \\
|E^h| & \leq & \dfrac{C_4(b-a)}{180}\, h^4 \qquad \text{(méthode de Simpson)} & \mathcal{O}(h^4) \\
\hline
\displaystyle\int_{-2h}^{2h} u(x)dx & = & \frac{2h}{45}\left(7\,U_{-2h} + 32\,U_{-h} + 12\,U_0 + 32\,U_h + 7\,U_{2h}\right) - \frac{8h^7}{945}u^{(6)}(\xi) & \\
|E^h| & \leq & \dfrac{2C_6(b-a)}{945}\, h^6 \qquad \text{(méthode de Boole)} & \mathcal{O}(h^6) \\
\hline
\end{array}
\tag{2.7}
$$

Ces relations peuvent être utilisées pour estimer *a priori* le nombre de sous-intervalles requis pour atteindre une précision ϵ. Ainsi, en exprimant que la borne supérieure de l'erreur d'une méthode composite des trapèzes doit être inférieure à ϵ et sachant que $h = (b-a)/n$, on trouve que

$$\begin{aligned} \frac{C_2}{12}\, n\, \frac{(b-a)^3}{n^3} &\leq \epsilon \\ \sqrt{\frac{C_2(b-a)^3}{12\epsilon}} &\leq n \end{aligned}$$

2.2 Méthodes à pas inégaux : Gauss-Legendre

Dans les méthodes de Newton-Cotes, le pas h est constant, de sorte que les abscisses des points d'intégration X_i sont imposées. Il ne reste donc comme degrés de liberté que les valeurs à attribuer aux poids w_i. En ne fixant pas de manière arbitraire les X_i, nous pourrions alors disposer de $n+1$ degrés de liberté supplémentaires pour construire une méthode d'intégration plus efficace dont le degré de précision peut être $2n+1$, puisque nous disposons désormais de $2n+2$ degrés de liberté. C'est l'idée exploitée par la méthode de Gauss-Legendre.

Considérons donc un polynôme quelconque de degré $2n+1$ contenant $2n+2$ termes

$$p(x) = \sum_{i=0}^{2n+1} a_i\, x^i$$

La méthode de Gauss-Legendre consiste à remplacer l'intégrale de ce polynôme sur l'intervalle $[-1,+1]$ par une somme judicieusement pondérée des valeurs du polynôme en un nombre fini de $n+1$ points. Les positions des points X_k et les poids associés w_k sont calculés a priori afin d'obtenir la stricte égalité entre les deux expressions pour n'importe quel polynôme de degré $2n+1$. Cela consiste donc à exiger que :

$$\int_{-1}^{1} p(x)\ _{dx} \quad = \quad \sum_{k=0}^{n} w_k p(X_k)\ ,$$

$\Big\downarrow$ En développant le polynôme,

$$\sum_{i=0}^{2n+1} \int_{-1}^{1} a_i\ x^i\ _{dx} \quad = \quad \sum_{k=0}^{n} w_k \sum_{i=0}^{2n+1} a_i\ X_k^i$$

$\Big\downarrow$ En séparant les termes pairs et impairs,

$$\sum_{j=0}^{n} \int_{-1}^{1} a_{2j} x^{2j}{}_{dx} \ + \ \sum_{j=0}^{n} \int_{-1}^{1} a_{2j+1} x^{2j+1}{}_{dx} \ = \ \sum_{k=0}^{n} w_k \sum_{j=0}^{n} a_{2j} X_k^{2j} \ + \ \sum_{k=0}^{n} w_k \sum_{j=0}^{n} a_{2j+1} X_k^{2j+1}$$

$\Big\downarrow$ En effectuant les intégrales,

$$\sum_{j=0}^{n} \tfrac{2}{(2j+1)} a_{2j} \quad + \quad 0 \quad = \quad \sum_{k=0}^{n} w_k \sum_{j=0}^{n} a_{2j} X_k^{2j} \ + \ \sum_{k=0}^{n} w_k \sum_{j=0}^{n} a_{2j+1} X_k^{2j+1}$$

L'identification des facteurs des coefficients quelconques a_i fournit $2n+2$ équations avec $2n+2$ inconnues ($n+1$ abscisses et $n+1$ poids). Plus précisément, nous devons choisir les points et poids d'intégration, tels que

$$\boxed{\begin{array}{lcll} \displaystyle\sum_{k=0}^{n} w_k X_k^{2j} & = & \dfrac{2}{(2j+1)} & j = 0, \ldots, n, \\ \displaystyle\sum_{k=0}^{n} w_k X_k^{2j+1} & = & 0 & j = 0, \ldots, n. \end{array}} \tag{2.8}$$

Pour un point d'intégration, on a

$$\begin{aligned} a_1 X_0\ w_0 &= 0, \\ a_0 w_0 &= 2a_0, \end{aligned}$$

soit, $X_0 = 0$ et $w_0 = 2$.

Pour deux points d'intégration, on a

$$\begin{aligned} a_3(X_0^3\ w_0 + X_1^3\ w_1) &= 0, \\ a_2(X_0^2\ w_0 + X_1^2\ w_1) &= 2a_2/3, \\ a_1(X_0\ w_0 + X_1\ w_1) &= 0, \\ a_0(w_0 + w_1) &= 2a_0, \end{aligned}$$

d'où on obtient facilement, en tirant profit de la symétrie des équations, les valeurs $w_0 = w_1 = 1$ et $X_0 = -X_1 = \sqrt{3}/3$. On peut ainsi poursuivre le calcul. Cette règle d'intégration est appelée *quadrature de Gauss-Legendre*. Les valeurs des points et poids sont reprises dans la table suivante où l'on tient compte de la symétrie des points et poids.

$n+1$	X_k, X_{n-k}	w_k, w_{n-k}
1	0.000000000000000	2.000000000000000
2	±0.577350269189626	1.000000000000000
3	±0.774596669241483	0.555555555555556
	0.000000000000000	0.888888888888889
4	±0.861136311594053	0.347854845137454
	±0.339981043584856	0.652145154862546
5	±0.906179845938664	0.236926885056189
	±0.538469310105683	0.478628670499366
	0.000000000000000	0.568888888888889
6	±0.932469514203152	0.171324492379170
	±0.661209386466265	0.360761573048139
	±0.238619186083197	0.467913934572691

La généralisation de la règle de Gauss-Legendre à l'intégration d'une fonction $u(x, y)$ sur un carré $[-1, 1] \times [-1, 1]$ est immédiate.

$$\int_{-1}^{1}\int_{-1}^{1} u(x,y)\ dx\ dy \approx \sum_{k=0}^{n}\sum_{l=0}^{n} w_k\ w_l\ u(X_k, Y_l). \tag{2.9}$$

Dans les grands logiciels d'éléments finis pour le calcul de structures, ou la simulation numérique des écoulements de fluides, on fait très fréquemment appel aux quadratures de Gauss-Legendre.

2.3 Méthodes récursives et adaptatives

Une des principales difficultés des méthodes qui viennent d'être présentées réside dans le choix du nombre de points à utiliser. Les estimations d'erreur *a priori* ne fournissent que des bornes supérieures parfois très largement supérieures et il est souvent impossible d'estimer les constantes qui y sont présentes. On est souvent forcé d'estimer grossièrement le nombre de points à utiliser en fonction du caractère plus ou moins régulier de la fonction à intégrer. Une fois le résultat obtenu, l'unique façon de valider le résultat obtenu consiste souvent à recommencer le calcul avec plus de points et à comparer les deux résultats obtenus.

On souhaite donc considérer des *formules de récurrence* qui permettraient de recalculer une intégrale avec plus de points en réutilisant au maximum les calculs effectués pour la première intégrale. Une autre manière de procéder est celle des *méthodes adaptatives* qui adaptent la longueur des sous-intervalles au comportement local de l'intégrand. La difficulté essentielle de cette seconde classe de méthodes est la nécessité d'estimer le comportement de l'intégrand a priori.

Sachant que l'erreur varie comme une certaine puissance du pas h, on souhaite en tirer profit de la manière suivante. A partir de deux ou plusieurs évaluations de l'intégrale pour des pas différents, on souhaite en réaliser une moyenne pondérée dont la précision serait nettement supérieure à celle des résultats obtenus. *L'extrapolation de Richardson* (aussi appelée *extrapolation à la limite*), que nous allons maintenant présenter, convient parfaitement pour cette approche.

2.3.1 Extrapolation de Richardson

L'extrapolation de Richardson permet d'obtenir une estimation de la valeur $f(0)$ d'une fonction f, à partir d'une *suite de valeurs pour des abscisses en progression géométrique*[1]. Typiquement, on aura $f(h)$, $f(\frac{h}{2})$, $f(\frac{h}{4})$, $f(\frac{h}{8})$, $f(\frac{h}{16})\ldots$. Ce qu'on pourrait écrire de manière plus compacte sous la forme

$$f(\frac{h}{2^i}) \qquad i = 0, 1, 2, 3, \ldots \tag{2.10}$$

L'idée de base est d'écrire les développements en série de Taylor pour toutes les valeurs fournies. Ensuite, par le biais de combinaisons linéaires judicieuses, il est possible d'obtenir une estimation de plus en plus précise de la valeur à l'origine.

[1]Comme le fait remarquer judicieusement Adrien Leygue, il est aussi possible de réaliser des extrapolations pour n'importe quel ensemble d'abscisses. En fait, l'idée de base est d'utiliser une extrapolation construite de manière intelligente. L'intérêt majeur d'avoir des abscisses en progression géométrique est de pouvoir calculer la valeur extrapolée par une succession de combinaisons linéaires quasiment identiques. Dans le cas général, il sera nécessaire de recalculer les coefficients de la combinaison linéaire à chaque étape ou de calculer directement le polynôme d'extrapolation. Avec un outil tel que *Matlab*, ce n'est, toutefois, pas un vrai problème.

Elimination du terme du premier ordre

Pour les deux premières abscisses, on peut écrire[2] :

$$\begin{array}{rcccccc} f(h) & = & f(0) & + & h\,f'(0) & + & \frac{h^2}{2}\,f''(0) \quad + \quad \ldots \\ f(\frac{h}{2}) & = & f(0) & + & \frac{h}{2}\,f'(0) & + & \frac{h^2}{8}\,f''(0) \quad + \quad \ldots \end{array} \tag{2.11}$$

Nous pouvons utiliser ces deux relations précédentes pour éliminer le terme en h. Multiplions la seconde équation par 2 et soustrayons ensuite membre à membre la première équation afin d'éliminer le terme du premier ordre. On obtient ainsi une valeur extrapolée de $f(0)$ dont le terme d'erreur est du second ordre $\mathcal{O}(h^2)$.

$$\begin{array}{rcccccc} 2f(\frac{h}{2}) - f(h) & = & (2-1)\,f(0) & + & \left(\frac{h^2}{4} - \frac{h^2}{2}\right) f''(0) & + & \ldots \\ & = & (2-1)\,f(0) & - & (2-1)\,\frac{h^2}{4} f''(0) & + & \ldots \\ & & \Big\downarrow & & & & \\ f(0) & = & \dfrac{\left(2f(\frac{h}{2}) - f(h)\right)}{(2-1)} & + & \dfrac{h^2}{4} f''(0) & + & \ldots \end{array} \tag{2.12}$$

Cette formule d'extrapolation correspond bien à l'intuition : pour obtenir une estimation de $f(0)$, on réalise une moyenne pondérée entre $f(h)$ et $f(\frac{h}{2})$ en attribuant le plus d'importance à la valeur la plus proche de l'origine et en tenant compte de l'évolution constatée entre les deux valeurs !

Elimination du terme du second ordre

On va maintenant utiliser la troisième valeur $f(\frac{h}{4})$, pour améliorer notre estimation à l'origine. On va procéder exactement comme précédemment en écrivant, pour la seconde et la troisième valeur, les développements en série de Taylor :

$$\begin{array}{rcccccc} f(\frac{h}{2}) & = & f(0) & + & \frac{h}{2}\,f'(0) & + & \frac{h^2}{8}\,f''(0) \quad + \quad \ldots \\ f(\frac{h}{4}) & = & f(0) & + & \frac{h}{4}\,f'(0) & + & \frac{h^2}{32}\,f''(0) \quad + \quad \ldots \end{array} \tag{2.13}$$

[2]En toute rigueur mathématique, il faudrait s'assurer que la série de Taylor de f en 0 converge et soit égale à f afin de pouvoir écrire les deux relations. On pourrait évidemment dire que les fonctions indéfiniment dérivables qui ne sont pas égales à leur série de Taylor dans un voisinage d'un certain point de leur domaine sont rarissimes. Mais, dans la vie réelle des méthodes numériques, les fonctions que l'on considère ne sont pas toujours dérivables.... Il n'est donc même pas du tout évident de supposer que f soit indéfiniment dérivable ! Et pour que l'extrapolation de Richardson fonctionne, il faut pourtant que cela soit vérifié !

A nouveau, nous allons utiliser ces deux relations pour éliminer le terme en h. Multiplions la seconde équation par 2 et soustrayons membre à membre la première équation afin d'éliminer le terme du premier ordre. On obtient ainsi une seconde valeur extrapolée de $f(0)$ dont le terme d'erreur est du second ordre $\mathcal{O}(h^2)$.

$$\begin{array}{rcl} 2f(\frac{h}{4}) - f(\frac{h}{2}) &=& (2-1)\,f(0) \;+\; \left(\frac{h^2}{16} - \frac{h^2}{8}\right) f''(0) \quad + \;\ldots \\ &=& (2-1)\,f(0) \;-\; (2-1)\,\frac{h^2}{16} f''(0) \quad + \;\ldots \\ & \downarrow & \\ f(0) &=& \dfrac{\left(2f(\frac{h}{4}) - f(\frac{h}{2})\right)}{(2-1)} \;+\; \dfrac{h^2}{16} f''(0) \;+\; \ldots \end{array} \tag{2.14}$$

Nous avons ainsi une seconde extrapolation de $f(0)$, d'ordre 2. En multipliant (2.14) par 4 et en soustrayant membre à membre l'équation (2.12) afin d'éliminer maintenant le terme du second ordre, on obtient une valeur extrapolée de $f(0)$ dont le terme d'erreur est du troisième ordre $\mathcal{O}(h^3)$.

$$f(0) \;=\; \frac{4\left(\dfrac{\left(2f(\frac{h}{4}) - f(\frac{h}{2})\right)}{(2-1)}\right) - \left(\dfrac{\left(2f(\frac{h}{2}) - f(h)\right)}{(2-1)}\right)}{(4-1)} \;-\; \frac{h^3}{48} f'''(0) \;+\; \ldots \tag{2.15}$$

A ce stade, il est utile d'introduire quelques notations pour alléger les écritures. Appelons $F_{i,j}$ les diverses expressions obtenues pour extrapoler la valeur de f à l'origine. L'indice j indique le degré du dernier terme en h que l'on a éliminé : il s'agit d'une extrapolation dont l'ordre est donc $j+1$. L'indice i signifie que l'extrapolation s'est faite à l'aide des $j+1$ valeurs $f(\frac{h}{2^{i-j}})$ à $f(\frac{h}{2^i})$.

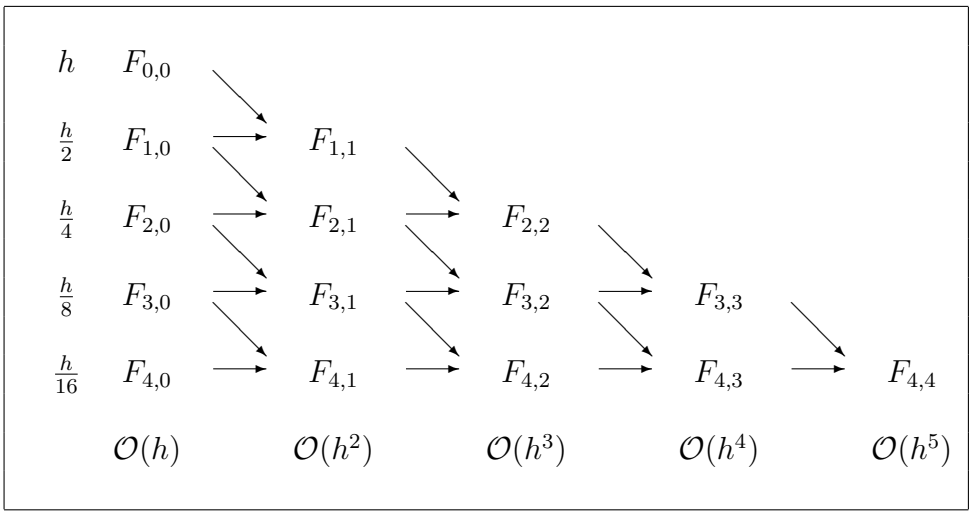
$$\boxed{\begin{array}{lll} F_{i,0} = f(\dfrac{h}{2^i}) & & \\ F_{i,1} = \dfrac{(2F_{i,0} - F_{i-1,0})}{(2-1)} & F_{i,2} = \dfrac{(2^2 F_{i,1} - F_{i-1,1})}{(2^2-1)} & \\ & \ldots & \\ & F_{i,j} = \dfrac{(2^j F_{i,j-1} - F_{i-1,j-1})}{(2^j-1)} & \end{array}} \tag{2.16}$$

Les expressions (2.12) et (2.14) s'écriront $F_{1,1}$ et $F_{2,1}$ comme des extrapolations d'ordre deux. L'expression complexe (2.15) sera simplement notée $F_{2,2}$.

On peut évidemment continuer de la sorte et éliminer successivement les termes d'ordre h^k. On obtient alors les valeurs extrapolées d'ordre $k+1$. On peut montrer que l'ordre du terme d'erreur pour $F_{i,k}$ est donné par :

$$f(0) - F_{i,k} = \mathcal{O}\left(\frac{h}{2^i}\right)^{(k+1)} \tag{2.17}$$

La disposition pratique des calculs se fait comme suit :



En outre, si l'on sait que le développement de $f(x)$ autour de $f(0)$ ne contient pas de termes de certains ordres, on peut sauter leur élimination. Par exemple, si la fonction est paire, les termes en $f'(0)$, $f'''(0)$, $f^{(5)}(0)$, ... sont nuls et l'on peut sauter l'élimination des termes en h, h^3, h^5, ...

2.3.2 Méthode de Romberg

Ce qu'on appelle la *méthode d'intégration de Romberg* consiste à appliquer l'extrapolation de Richardson à une série d'évaluations numériques d'une intégrale par la méthode des trapèzes avec des pas décroissants.

Pour calculer l'intégrale

$$I = \int_a^b u(x)\,dx \tag{2.18}$$

appliquons successivement la méthode des trapèzes, avec 1, 2, 4, 8 et 16 sous-intervalles.

$$
\begin{aligned}
I^h &= \tfrac{h}{2}\,(U_0 + U_1) \\
I^{h/2} &= \tfrac{h}{4}\,(U_0 + 2U_1 + U_2) \\
I^{h/4} &= \tfrac{h}{8}\,(U_0 + 2U_1 + 2U_2 + 2U_3 + U_4) \\
I^{h/8} &= \tfrac{h}{16}\,(U_0 + 2U_1 + 2U_2 + 2U_3 + \ldots + 2U_7 + U_8) \\
I^{h/16} &= \tfrac{h}{32}\,(U_0 + 2U_1 + 2U_2 + 2U_3 + \ldots + 2U_{15} + U_{16})
\end{aligned}
\tag{2.19}
$$

où dans toutes les expressions, nous avons exprimé l'écart entre deux abscisses en fonction de celui qui apparaissait dans la première évaluation. En d'autres mots, nous avons introduit un unique symbole $h = b - a$ pour toutes les règles d'intégration.

On peut alors appliquer une extrapolation de Richardson en considérant que les valeurs obtenues en (2.19) doivent être interprétées comme les valeurs d'une fonction $I(x)$ dont l'extrapolation en $x = 0$ vaudrait l'intégrale recherchée. Il s'agit d'effectuer l'extrapolation à la limite suivante.

Estimer $I(0)$ sachant que

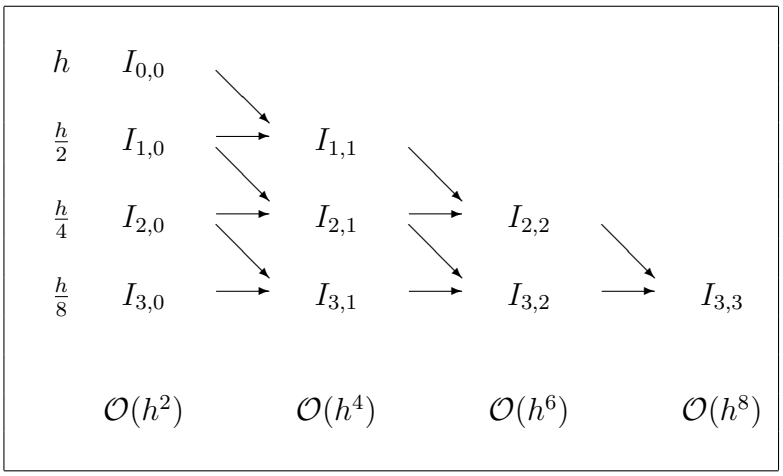
$$
\begin{aligned}
I(h) &= I^h \\
I(h/2) &= I^{h/2} \\
I(h/4) &= I^{h/4} \\
I(h/8) &= I^{h/8} \\
I(h/16) &= I^{h/16}
\end{aligned}
\tag{2.20}
$$

Pour établir la formule d'extrapolation à utiliser, la méthode de Romberg se base sur la propriété que le terme d'erreur ne peut être représenté que par des puissances paires de h. Dans un tel cas, l'extrapolation de Richardson s'écrit :

$$
\begin{aligned}
I_{i,1} &= \frac{(2^2 I_{i,0} - I_{i-1,0})}{(2^2 - 1)} \\
&\ldots \\
I_{i,j} &= \frac{(2^{2j} I_{i,j-1} - I_{i-1,j-1})}{(2^{2j} - 1)}
\end{aligned}
\tag{2.21}
$$

Pour le calcul des $I_{i,1}$, la règle est : *4 fois la meilleure estimation moins la moins bonne, le tout divisé par 3.* Tandis que pour le calcul de $I_{i,2}$, la règle peut se lire : *16 fois la meilleure estimation moins la moins bonne, le tout divisé par 15.* On peut évidemment continuer de la sorte.

Dans la pratique, on construit progressivement le tableau des $I_{i,k}$. On évalue d'abord la règle des trapèzes avec 1 puis 2 intervalles ($I_{0,0}$ et $I_{1,0}$). On y applique l'extrapolation pour trouver $I_{1,1}$. On évalue alors la règle des trapèzes avec 4 intervalles ($I_{2,0}$); une extrapolation permet de calculer $I_{2,1}$ et une seconde extrapolation fournit $I_{2,2}$. Cette dernière est la meilleure approximation. Si elle est trop différente de la meilleure approximation précédente $I_{1,1}$, on recommence une étape en évaluant la formule des trapèzes avec 2 fois plus d'intervalles ($I_{3,0}$), ce qui permet de calculer les extrapolations $I_{3,1}$, $I_{3,2}$ et $I_{3,3}$, cette dernière devenant la meilleure estimation trouvée. Si elle diffère trop des précédentes, on recommence encore.



La méthode de Romberg permet donc de progresser régulièrement et d'arrêter les calculs au moment où une précision suffisante est atteinte. Elle ne demande pas de fixer a priori le nombre de points.

On peut montrer que les approximations d'ordre 4 (les $I_{i,1}$) sont les estimations de l'intégrale par la méthode de Simpson, les approximations d'ordre 6 (les $I_{i,2}$) correspondent à l'application de la méthode de Boole. Par contre, les approximations d'ordre 8 (les $I_{i,3}$) ne correspondent pas à une application quelconque de la méthode de Newton-Cotes d'ordre 8... Une simple comparaison du nombre de points (9 et 7 points respectivement) permet de le déduire. La méthode de Romberg permet donc, à l'aide d'extrapolations peu coûteuses, d'atteindre des degrés de précision croissants très rapidement.

A titre d'exemple, calculons l'intégrale suivante par la méthode de Romberg :

$$I = \int_0^{\pi/2} (x^2 + x + 1) \cos x \, dx \tag{2.22}$$

La solution exacte est 2.038197. On dispose les résultats successifs dans le tableau triangulaire que l'on étend au fur et à mesure des calculs :

i	$I_{i,0}$	$I_{i,1}$	$I_{i,2}$	$I_{i,3}$	$I_{i,4}$
0	0.785398				
1	1.726813	2.040617			
2	1.960534	2.038441	2.038296		
3	2.018794	2.038214	2.038199	2.038197	
4	2.033347	2.038198	2.038197	2.038197	2.038197
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$	$\mathcal{O}(h^{10})$

On constate très clairement dans cette table l'avantage présenté par la méthode de Romberg : la méthode des trapèzes avec 16 intervalles fournit le résultat 2.033347, alors qu'une extrapolation basée sur la méthode des trapèzes avec 1, 2, 4, 8 et 16 intervalles donne le résultat 2.038197 où tous les chiffres sont corrects.

Lorsque l'on a utilisé la méthode des trapèzes avec $m = 2^i$ sous-intervalles et que l'on passe à $2m$ intervalles deux fois plus petits, toutes les abscisses utilisées dans le premier cas se retrouvent dans la deuxième évaluation. Est-il possible d'éviter de recalculer $u(x)$ aux abscisses où ce calcul a déjà été fait? Evidemment, car on peut établir une formule de récurrence

$$I^h \;=\; \tfrac{1}{2} I^{2h} \;+\; 2h \sum_{k=1}^{m} U_{2k-1} \tag{2.23}$$

où $h = (b-a)/2m$ est le nouveau pas. Ainsi par exemple, pour un intervalle d'intégration allant de 0 à 16, le calcul de $I_{2,0}$ (4 intervalles) demande le calcul de $u(x)$ pour $x = 0, 4, 8, 12, 16$. On pourra donc calculer $I_{3,0}$, correspondant à 8 intervalles, par la relation

$$I_{3,0} \;=\; \tfrac{1}{2} I_{2,0} + 4 \Big(\; u(2) \;+\; u(6) \;+\; u(10) \;+\; u(14) \; \Big) \tag{2.24}$$

ce qui ne demande le calcul de $u(x)$ qu'aux 4 nouveaux points. Cette récurrence permet donc d'effectuer de manière plus rapide et efficace le calcul des $I_{i,0}$.

2.3.3 Méthodes adaptatives d'intégration

Les règles composites du type Newton-Cotes sont basées sur des sous-intervalles égaux. Ce choix permet de développer des formules simples. Cependant, lorsque le comportement de $u(x)$ est très différent pour les différentes valeurs de x, il peut être intéressant d'utiliser des sous-intervalles de tailles différentes. On pourrait alors calculer l'intégrale, pour une

précision donnée, avec moins d'évaluations de $u(x)$ si la longueur des sous-intervalles est adaptée en fonction du comportement de l'intégrand.

L'erreur globale d'intégration est le résultat de la somme des erreurs obtenues sur chacun des sous-intervalles. Les méthodes qui adaptent la longueur des sous-intervalles au comportement local de l'intégrand, sont appelées *méthodes adaptatives*. La difficulté essentielle de ces méthodes réside dans l'obligation d'estimer l'erreur commise dans chaque sous-intervalle. La manière de construire de telles méthodes est basée sur la mise au point plus ou moins heuristique de *stratégies adaptatives*.

Une stratégie adaptative peut être comparée à un paradigme de contrôle optimal dans lequel l'erreur de discrétisation est contrôlée par un schéma adaptatif qui orchestre la répartition de la taille des sous-intervalles afin d'obtenir un niveau de précision fixé à un coût minimal. Un tel objectif est obtenu avec un ensemble de sous-intervalles qui contribuent de manière approximativement uniforme à l'erreur globale.

Il convient donc tout d'abord de se fixer un objectif de précision θ. Ensuite la stratégie adaptative peut être vue comme la succession des étapes suivantes :

1 On calcule une première approximation I^h.

2 On estime a posteriori l'erreur commise dans chaque sous-intervalle.
En général, une telle estimation est obtenue en utilisant deux méthodes distinctes d'intégration et en comparant judicieusement les deux résultats.

3 Si l'estimation de l'erreur commise sur un sous-intervalle est excessive, nous divisons ce sous-intervalle afin d'atteindre l'objectif de précision avec un nombre minimum de nouvelles abscisses.

4 On obtient une nouvelle valeur I^h en utilisant les nouvelles abscisses, et on recommence le processus jusqu'à satisfaction de l'objectif.

Chapitre 3

Comment dériver numériquement une fonction ?

Le problème abordé dans ce chapitre consiste à calculer la valeur D correspondant à la dérivée d'une fonction $u(x)$ en un point x

$$D = u'(x)$$

La dérivation numérique est un des outils essentiels pour la construction de méthodes pour la résolution numérique de problèmes aux conditions initiales ou aux conditions aux limites comprenant donc des équations différentielles ordinaires (Initial Value Problem) ou des équations aux dérivées partielles (Boundary Value Problem).

Bien que simples dans leur principe, les méthodes classiques de calcul numérique des dérivées sont numériquement instables : les erreurs d'arrondi détériorent très rapidement la qualité des résultats trouvés. Le second objectif de ce chapitre est donc d'analyser les difficultés liées aux erreurs d'arrondi et au concept de stabilité numérique.

Rappelons-nous la définition de la dérivée d'une fonction

$$\begin{aligned} D &= u'(x) \\ &= \lim_{h\to 0} \frac{u(x+h) - u(x)}{h} \end{aligned} \tag{3.1}$$

Une méthode numérique y apparaît directement en filigrane. Choisissons une suite d'écarts h_k qui tend vers zéro. Ensuite, il s'agira d'estimer la limite de la suite définie par

$$D^{h_k} = \frac{u(x+h_k) - u(x)}{h_k} \qquad k = 1, 2, \ldots, n, \ldots \tag{3.2}$$

Théoriquement, on peut voir D comme la limite de la suite D^{h_k}. Evidemment, en pratique, nous n'allons calculer qu'un nombre limité des termes D^{h_1}, D^{h_2}, ..., D^{h_n} dans la suite (3.2). Deux questions apparaissent... Pourquoi calculer D^{h_1}, D^{h_2}, ..., $D^{h_{n-1}}$? Et quelle valeur de n ou de h_n devons-nous choisir afin que D^{h_n} soit une bonne approximation de la valeur exacte D ? Pour répondre à ces questions, il est donc essentiel de comprendre le comportement de l'erreur de discrétisation en fonction de h. L'*erreur de discrétisation* est évidemment définie comme la différence entre la valeur exacte de la dérivée et la valeur numérique approchée : $E^h = D - D^h$.

Intuitivement, on s'attendrait à ce que l'approximation (3.2) soit d'autant meilleure que le *pas de discrétisation* h est petit. En effectuant un développement de Taylor de $u(x)$ autour de x, on observe :

$$\begin{aligned} u(x+h) &= u(x) + hu'(x) + \frac{h^2}{2}u''(\xi) \\ &\downarrow \\ u'(x) - \frac{u(x+h) - u(x)}{h} &= -\frac{h}{2}u''(\xi) \\ &\downarrow \\ \underbrace{D - D^h}_{E^h} &= -\frac{h}{2}u''(\xi) \end{aligned}$$

où ξ est un point particulier de l'intervalle $[x, x+h]$.

Si nous pouvons trouver une estimation C qui borne la valeur absolue de la dérivée seconde de $u(x)$ sur l'intervalle considéré, on peut alors écrire une borne supérieure de l'erreur de discrétisation sous la forme :

$$\boxed{|E^h| \leq \frac{C}{2}\, h} \tag{3.3}$$

Au vu de ce résultat, la méthode de dérivation numérique introduite est *théoriquement convergente*. On constate aussi que la méthode qui vient d'être développée est *d'ordre 1*. Nous allons toutefois observer qu'en pratique, ce n'est pas aussi simple...

Utilisons cette méthode avec $h_k = 10^{-k}$ pour évaluer la dérivée de la fonction $\exp(x)$ en $x = 1$. Le résultat exact vaut $e^1 = 2.718281828...$ Dans ce cas-ci, la constante C est donnée par e^{1+h}.

h_k	$\exp(1+h_k)$	D^{h_k}	E^{h_k}
$h_0 = 1$	7.389056099	4.670774270	-1.952492442
$h_1 = 0.1$	3.004166024	2.858841955	-0.140560126
$h_2 = 0.01$	2.745601015	2.731918656	-0.013636827
$h_3 = 0.001$	2.721001470	2.719641423	-0.001359594
$h_4 = 0.0001$	2.718553670	2.718417747	-0.000135919
$h_5 = 0.00001$	2.718309011	2.718295420	-0.000013592
$h_6 = 10^{-6}$	2.718284547	2.718283187	-0.000001359
$h_7 = 10^{-7}$	2.718282100	2.718281964	-0.000000136
$h_8 = 10^{-8}$	2.718281856	2.718281777	0.000000051
$h_9 = 10^{-9}$	2.718281831	2.718281600	0.000000229
$h_{10} = 10^{-10}$	2.718281829	2.718278935	0.000002893
$h_{11} = 10^{-11}$	2.718281828	2.718270053	0.000011775
$h_{12} = 10^{-12}$	2.718281828	2.718270053	0.000011775
$h_{13} = 10^{-13}$	2.718281828	2.713385072	0.004896756
$h_{14} = 10^{-14}$	2.718281828	2.664535259	0.053746569
$h_{15} = 10^{-15}$	2.718281828	2.664535259	0.053746569
$h_{16} = 10^{-16}$	2.718281828	0.000000000	2.718281828
$h_{17} = 10^{-17}$	2.718281828	0.000000000	2.718281828
$h_{18} = 10^{-18}$	2.718281828	0.000000000	2.718281828
$h_{19} = 10^{-19}$	2.718281828	0.000000000	2.718281828

Dans cette table, l'erreur réelle décroît d'abord d'un facteur 10 lorsque h est divisé par 10, comme cela doit être le cas pour une méthode d'ordre 1. Cependant, alors que l'erreur de discrétisation devrait théoriquement diminuer lorsque h décroît, on constate dans notre expérience numérique que l'erreur observée commence par diminuer puis remonte à partir d'une certaine valeur de h. Il s'agit d'une véritable catastrophe numérique !

Le calcul effectué avec `python` et un pas $h = 10^{-16}$ fournit une valeur nulle, totalement stupide. Même la meilleure des estimations a une erreur de l'ordre de 10^{-8}. Cette dégradation numérique apparaît lorsque le pas h devient trop petit, par rapport à l'impact des erreurs d'arrondi. Lorsque h devient très petit, les deux quantités dont il faut calculer la différence au numérateur de (3.2) deviennent presque égales. On verra que la soustraction de deux nombres très voisins peut conduire à une perte importante de chiffres significatifs sur le résultat : c'est ce qui se passe ici. Le cas extrême est obtenu avec $h = 10^{-16}$, les deux valeurs ne peuvent même plus être distinguées par la machine : leur différence est donc considérée comme nulle et la dérivée est estimée de manière totalement stupide à une valeur nulle.

Evidemment, l'utilisation de méthodes plus précises, capables de fournir une bonne estimation de la dérivée avec des pas h plus grands, permettra de réduire l'impact des erreurs d'arrondi.

3.1 Différences centrées et unilatérales

Si la fonction $u(x)$ peut être évaluée à gauche et à droite du point où il faut calculer la dérivée, alors la meilleure approximation de la dérivée fera intervenir des abscisses qui sont choisies de manière symétrique autour de x. Sans perte de généralité, nous allons supposer que nous souhaitons estimer la dérivée à l'origine, c'est-à-dire que $x = 0$.

$$\begin{array}{|lcll|}
\hline
u'(0) & = & \dfrac{(U_h - U_{-h})}{2h} - \frac{h^2}{6}u^{(3)}(\xi) & \mathcal{O}(h^2) \\
\hline
u'(0) & = & \dfrac{(-U_{2h} + 8U_h - 8U_{-h} + U_{-2h})}{12h} + \frac{h^4}{30}u^{(5)}(\xi) & \mathcal{O}(h^4) \\
\hline
\end{array} \tag{3.4}$$

où U_x dénote $u(x)$ et ξ est un point particulier d'un intervalle $[a, b]$ contenant toutes les abscisses utilisées. A nouveau, s'il est possible d'estimer une borne pour la valeur de la dérivée troisième (ou cinquième), on observe que l'erreur de discrétisation sera proportionnelle à h^2 (ou h^4). C'est pourquoi on parlera de *différence centrée du second ordre* (ou de *différence centrée du quatrième ordre*).

A titre d'illustration, nous démontrons ici que la première formule de (3.4) fournit bien une estimation du second ordre de la dérivée. Il suffit d'écrire les développements en série de Taylor à partir de l'origine pour évaluer $u(x)$ en h et en $-h$.

$$\begin{aligned}
u(h) &= u(0) + h\,u'(0) + \frac{h^2}{2}\,u''(0) + \frac{h^3}{6}\,u'''(\xi_1) \\
u(-h) &= u(0) - h\,u'(0) + \frac{h^2}{2}\,u''(0) - \frac{h^3}{6}\,u'''(\xi_2)
\end{aligned}$$

où $\xi_1 \in]0, h[$ et $\xi_2 \in]-h, 0[$. En soustrayant membre à membre ces deux relations, on obtient

$$
\begin{aligned}
u(h) - u(-h) &= 2\,h\,u'(0) + \frac{h^3}{3}\left(\frac{u'''(\xi_1) + u'''(\xi_2)}{2}\right) \\
&\Big\downarrow \quad \text{En vertu du théorème de la valeur intermédiaire, on peut trouver } \xi \in [-h, h] \text{ tel que,} \\
u(h) - u(-h) &= 2\,h\,u'(0) + \frac{h^3}{3}u'''(\xi) \\
&\Big\downarrow \\
\frac{(u(h) - u(-h))}{2h} &= u'(0) + \frac{h^2}{6}u'''(\xi) \\
\frac{(U_h - U_{-h})}{2h} &= u'(0) + \frac{h^2}{6}u'''(\xi)
\end{aligned}
$$

Nous avons ainsi montré qu'une différence centrée avec deux points est d'ordre 2. En outre, cette différence ne nécessite pas plus de calculs que la méthode d'ordre 1 : il suffit d'évaluer deux fois la fonction $u(x)$. L'interprétation géométrique de la différence centrée d'ordre deux et d'une différence décentrée d'ordre un est fournie à la Figure 3.1.

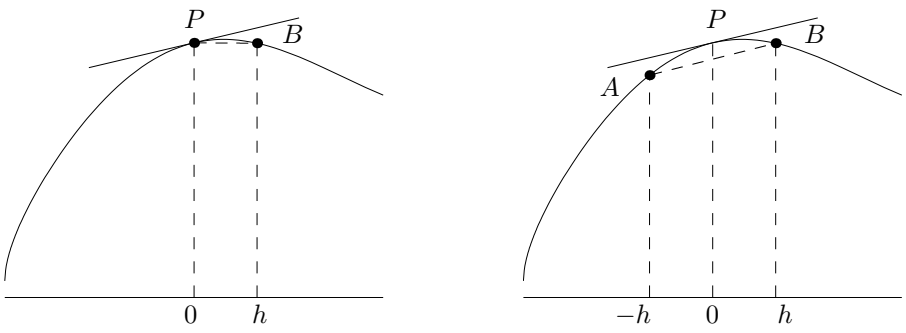


Figure 3.1: Interprétation géométrique des différences du premier et second ordre. La pente de la corde AB est une meilleure approximation de la pente de la tangente en P que celle de la corde PB utilisée pour la différence du premier ordre.

Les formules de différences peuvent aussi être développées en calculant le polynôme d'interpolation passant par les points considérés, puis en dérivant ce polynôme. C'est l'approche que nous allons adopter pour obtenir une formule du second ordre décentrée ou unilatérale. Une telle formule se révèle très utile lorsqu'on ne peut disposer des valeurs de $u(x)$ que d'un seul côté de $x = 0$. Etablissons à titre d'exemple une formule unilatérale

pour le calcul de $u'(0)$, en utilisant les 3 points équidistants $X_0 = 0$, $X_1 = h$ et $X_2 = 2h$.

$$
\begin{aligned}
u'(0) &\approx \overbrace{(u^h)'(0)}^{D^h} \\
&\downarrow \\
&\approx \sum_{i=0}^{2} u(X_i)\phi_i'(0) \\
&\downarrow \quad \text{En fixant } X_0 = 0,\ X_1 = h \text{ et } X_2 = 2h \\
&\approx U_0 \underbrace{\left.\frac{d}{dx}\left(\frac{(x-h)(x-2h)}{2h^2}\right)\right|_{x=0}}_{-3/2h} \\
&\qquad + U_h \underbrace{\left.\frac{d}{dx}\left(\frac{x(x-2h)}{-h^2}\right)\right|_{x=0}}_{2/h} \\
&\qquad + U_{2h} \underbrace{\left.\frac{d}{dx}\left(\frac{x(x-h)}{2h^2}\right)\right|_{x=0}}_{-1/2h}
\end{aligned}
$$

On a donc établi la formule suivante :

$$u'(0) \simeq \frac{(-3U_0 + 4U_h - U_{2h})}{2h} \tag{3.5}$$

On peut montrer, avec des développements en série de Taylor, que cette formule est d'ordre 2. Une telle formule est appelée *différence décentrée d'ordre deux*. On parle aussi de *différence aval*, de *forward difference* ou de *différence unilatérale*. Les formules recourant uniquement à des abscisses à gauche seront appelées *différences amont* ou *backward differences*.

Calcul des dérivées d'ordre supérieur

Pour obtenir des estimations numériques pour des dérivées d'ordre supérieur, on peut utiliser exactement la même approche sur base de développements en série de Taylor ou du calcul des dérivées des fonctions de base d'une interpolation polynomiale de Lagrange.

A titre d'illustration, pour obtenir l'approximation d'une dérivée seconde sous la forme d'une différence centrée d'ordre deux, on écrit les deux séries de Taylor suivantes :

$$u(h) = u(0) + h\,u'(0) + \frac{h^2}{2}\,u''(0) + \frac{h^3}{6}\,u'''(0) + \frac{h^4}{24}\,u^{(4)}(\xi_1)$$

$$u(-h) = u(0) - h\,u'(0) + \frac{h^2}{2}\,u''(0) - \frac{h^3}{6}\,u'''(0) + \frac{h^4}{24}\,u^{(4)}(\xi_2)$$

où $\xi_1 \in]0, h[$ et $\xi_2 \in]-h, 0[$. En additionnant membre à membre ces deux relations, on obtient

$$u(h) + u(-h) = 2u(0) + h^2\,u''(0) + \frac{h^4}{12}\left(\frac{u^{(4)}(\xi_1) + u^{(4)}(\xi_2)}{2}\right)$$

$$\Big\downarrow$$

En vertu du théorème de la valeur intermédiaire, on peut trouver $\xi \in [-h, h]$ tel que,

$$u(h) + u(-h) = 2u(0) + h^2\,u''(0) + \frac{h^4}{12}u^{(4)}(\xi)$$

$$\Big\downarrow$$

$$\frac{(u(h) - 2u(0) + u(-h))}{h^2} = u''(0) + \frac{h^2}{12}u^{(4)}(\xi)$$

$$\frac{(U_h - 2U_0 + U_{-h})}{h^2} = u''(0) + \frac{h^2}{12}u^{(4)}(\xi)$$

Pratiquement, on observe le même comportement que pour l'évaluation numérique de la dérivée première : quand h diminue, l'erreur diminue car l'erreur de discrétisation diminue comme h^2 et h^4, mais à partir d'un certain moment, les erreurs d'arrondi l'emportent et dégradent la solution trouvée. Cet effet de dégradation est même plus précoce que pour les dérivées premières.

En suivant la même procédure que pour la dérivée seconde, on peut établir les formules pour le calcul des dérivées d'ordre supérieur. Les choix les plus populaires sont les schémas de différences centrées d'ordre deux et quatre, ainsi que les schémas de différences amont d'ordre deux.

$$
\begin{array}{|rcl|}
\hline
u'(0) & \approx & \dfrac{(U_h - U_{-h})}{2h} \\
u''(0) & \approx & \dfrac{(U_h - 2U_0 + U_{-h})}{h^2} \\
u^{(3)}(0) & \approx & \dfrac{(U_{2h} - 2U_h + 2U_{-h} - U_{-2h})}{2h^3} \\
u^{(4)}(0) & \approx & \dfrac{(U_{2h} - 4U_h + 6U_0 - 4U_{-h} + U_{-2h})}{h^4} \\
 & & \text{(différences centrées)} \quad \mathcal{O}(h^2) \\
\hline
u'(0) & \approx & \dfrac{(-U_{2h} + 8U_h - 8U_{-h} + U_{-2h})}{12h} \\
u''(0) & \approx & \dfrac{(-U_{2h} + 16U_h - 30U_0 + 16U_{-h} - U_{-2h})}{12h^2} \\
u^{(3)}(0) & \approx & \dfrac{(-U_{3h} + 8U_{2h} - 13U_h + 13U_{-h} - 8U_{-2h} + U_{-3h})}{8h^3} \\
u^{(4)}(0) & \approx & \dfrac{(-U_{3h} + 12U_{2h} - 39U_h + 56U_0 - 39U_{-h} + 12U_{-2h} - U_{-3h})}{6h^4} \\
 & & \text{(différences centrées)} \quad \mathcal{O}(h^4) \\
\hline
u'(0) & \approx & \dfrac{(3U_0 - 4U_{-h} + U_{-2h})}{2h} \\
u''(0) & \approx & \dfrac{(2U_0 - 5U_{-h} + 4U_{-2h} - U_{-3h})}{h^2} \\
u^{(3)}(0) & \approx & \dfrac{(5U_0 - 18U_{-h} + 24U_{-2h} - 14U_{-3h} + 3U_{-4h})}{2h^3} \\
u^{(4)}(0) & \approx & \dfrac{(3U_0 - 14U_{-h} + 26U_{-2h} - 24U_{-3h} + 11U_{-4h} - 2U_{-5h})}{h^4} \\
 & & \text{(différences amont)} \quad \mathcal{O}(h^2) \\
\hline
\end{array}
\tag{3.6}
$$

3.2 Analyse des erreurs d'arrondi

Comme nous venons de l'observer, les méthodes classiques de calcul numérique des dérivées sont numériquement instables : les erreurs d'arrondi détériorent très rapidement la qualité des résultats trouvés. C'est pourquoi il est utile d'essayer de comprendre comment s'effectuent les calculs au sein d'un ordinateur afin de pouvoir prédire l'impact des erreurs d'arrondi.

3.2.1 Arithmétique en virgule flottante

En mathématiques, nous supposons implicitement que les opérations arithmétiques de base $(+,-,\times,/)$ sont réalisées de manière parfaite et que les opérandes de ces mêmes opérations sont également connus de manière parfaite. Cependant, en effectuant une évaluation numérique, un ordinateur ne peut retenir qu'un nombre *fini* de chiffres pour représenter les opérandes et les résultats des calculs intermédiaires. Ceci conduit à un type d'erreur connu sous le nom d'*erreur d'arrondi*.

Afin d'illustrer notre propos, considérons un calculateur utilisant 4 chiffres pour représenter un nombre. Calculons la somme $1.348 + 9.999$. Le résultat exact vaut 11.347 et comporte 5 chiffres. Le calculateur va donc le représenter de manière approchée : 11.35. Ce faisant, il commet une erreur d'arrondi égale à $(11.35 - 11.347) = 0.003$. La propagation des erreurs d'arrondi en cours de calcul peut éventuellement avoir des conséquences catastrophiques sur la précision du résultat final.

Soit $\tilde{x}$ la représentation en précision finie d'un nombre réel x. D'une certaine manière, $\tilde{x}$ peut être compris comme une approximation de x. On dit que $\tilde{x}$ possède d *chiffres significatifs* si d est le plus grand entier positif tel que

$$\left|\frac{x-\tilde{x}}{x}\right| < \tfrac{1}{2}\ 10^{-d} \tag{3.7}$$

Le nombre de chiffres significatifs est évidemment une manière de comprendre la qualité et la fiabilité d'une valeur numérique. Ceci est illustré dans le tableau ci-dessous

x	$\tilde{x}$	nombre de chiffres significatifs : d	
1.000001	0.999998	$\left\|\frac{x-\tilde{x}}{x}\right\| \approx 0.000003 < \frac{1}{2}\ 10^{-5}$	5
0.001234	0.001231	$\left\|\frac{x-\tilde{x}}{x}\right\| \approx 0.002 < \frac{1}{2}\ 10^{-2}$	2
0.000013	0.000009	$\left\|\frac{x-\tilde{x}}{x}\right\| \approx 0.31 < \frac{1}{2}$	0

Dans un ordinateur, les calculs numériques portant sur des nombres réels sont effectués à l'aide de l'*arithmétique en virgule flottante*. Dans une telle arithmétique, un nombre réel est représenté sous une forme normalisée

$$\pm 0.m\ 10^{\pm e}$$

où m est la *mantisse* et e est l'*exposant* de la représentation. La mantisse et l'exposant comportent tous deux un nombre *fini* de chiffres. Dans ce système avec 4 chiffres pour la mantisse et deux chiffres pour l'exposant, quelques exemples de nombres en virgule flottante sont

$$\begin{array}{rrcrrcr}
 & & & & & & \overbrace{}^{\text{11 zéros}} \\
0.5000 & 10^{12} & = & 5.000 & 10^{11} & = & 5\overbrace{00000000000}.000000000000000 \\
-0.3275 & 10^{04} & = & -3.275 & 10^{03} & = & -3275.000000000000000 \\
0.5723 & 10^{01} & = & 5.723 & 10^{00} & = & 5.723000000000000 \\
0.9422 & 10^{-11} & = & 9.422 & 10^{-12} & = & 0.\underbrace{00000000000}_{\text{11 zéros}}9422
\end{array}$$

Pour obtenir une représentation unique ou normalisée d'un nombre en virgule flottante, on impose que le premier chiffre de la mantisse m soit non-nul ; le nombre 1 est donc représenté par $0.1000\ 10^{01}$ (et jamais par $0.0100\ 10^{02}$, $0.0001\ 10^{04}$ ou d'autres possibilités...). Dans un tel système avec 4 chiffres pour la mantisse et deux chiffres pour l'exposant, le plus petit et plus grand nombre représentable sont respectivement $-0.9999\ 10^{99}$ et $0.9999\ 10^{99}$. Nous voyons également que le plus petit nombre strictement positif représentable est $0.1000\ 10^{-99}$.

Les ordinateurs utilisent le même principe en s'appuyant sur le système binaire, et non le système décimal. Pour les nombres réels, les ordinateurs actuels utilisent principalement deux systèmes de représentation en virgule flottante, selon le nombre de chiffres binaires (bits) utilisés pour représenter la mantisse et l'exposant. On parle de calculs en virgule flottante en *simple précision* (32 bits) ou en *double précision* (64 bits).

Les opérations arithmétiques en virgule flottante ont une précision finie. Reprenons notre calculateur avec 4 chiffres pour la mantisse et deux chiffres pour l'exposant. Dans un tel calculateur, chaque nombre est représenté avec trois chiffres significatifs. Effectuons quelques additions avec des données exactes.

x	y	$s = x + y$	$\tilde{s}$
$0.1234\ 10^{01}$	$0.5678\ 10^{-03}$	1.2345678	$0.1234\ 10^{01}$
$0.1000\ 10^{01}$	$0.1000\ 10^{-03}$	1.0001000	$0.1000\ 10^{01}$

Le résultat exact ne peut souvent pas être représenté. Dans le premier cas, une représentation possible est obtenue en choisissant le nombre en virgule flottante $0.1234\ 10^{01}$. Une meilleure représentation aurait pu être le nombre en virgule flottante le plus proche du résultat exact, soit $0.1235\ 10^{01}$. Quel que soit le mode d'approximation retenu[1], la différence entre le résultat exact et sa représentation en virgule flottante approchée génère une erreur que l'on appelle *erreur d'arrondi*.

Dans le second exemple, lorsque deux nombres en virgule flottante de valeurs très différentes sont additionnés, le résultat du calcul est égal strictement (pour le calculateur !) à un des opérandes. Le plus petit nombre positif ϵ que l'on peut ajouter à 1 pour obtenir un résultat en virgule flottante strictement supérieur à 1 est appelé l'*epsilon machine*. Pour le calculateur de notre exemple, $\epsilon = 0.001$ si le résultat est arrondi vers le bas et $\epsilon = 0.0005$ si le résultat est arrondi au nombre le plus proche. L'epsilon machine est une image de la précision relative de l'arithmétique en virgule flottante. On peut montrer que la représentation en virgule flottante d'un nombre approche donc ce nombre avec une erreur relative bornée par ϵ.

Lorsqu'on effectue avec un programme informatique un calcul et que le résultat produit une valeur dépassant le plus grand nombre représentable, on dira qu'il y a *overflow*. Dans la plupart des langages informatiques, un dépassement entraîne l'apparition d'un message d'erreur[2]. Dans d'autres situations, il peut arriver que le calcul doive produire un nombre plus proche de zéro que le plus petit nombre positif représentable. On dira dans ce cas qu'il y a *underflow*, et on imagine déjà les problèmes, si on souhaite procéder à une division dont ce nombre serait le diviseur...

Les dangers de la soustraction...

Calculer la différence de deux nombres proches conduit à la *perte de chiffres significatifs*, lorsque les opérandes ne sont pas exacts, mais sont seulement une représentation approximative en virgule flottante des données exactes. Par exemple, considérons $\tilde{x} = 3.12345678456$ et $\tilde{y} = 3.12345678447$ des représentations avec 11 chiffres significatifs de

[1] En pratique, la manière dont l'arithmétique est effectuée par un ordinateur est définie dans une norme très précise. Il s'agit du standard IEEE 754 qui date de 1985 et qui décrit comment exécuter, normaliser et arrondir les opérations en virgule flottante pour les formats de 32 et 64 bits. Aujourd'hui, tous les ordinateurs satisfont ce standard. Il y avait autrefois des exceptions : les super-ordinateurs des années 80 et 90 construits par *Cray* ne satisfaisaient pas ce standard. En outre, ceci ne vous assure pas que ce qu'on définit comme *float* ou *double* dans un langage informatique, correspond vraiment à une représentation en 32 ou 64 bits. Cela devrait aussi normalement être codifié de manière très rigoureuse au moyen de normes internationales, afin que le même calcul donne le même résultat quelle que soit la machine que vous utilisez. Il y a aussi une exception de taille : la représentation interne d'un réel d'un langage aussi populaire que le C/C^{++} peut formellement dépendre de l'ordinateur ou du compilateur utilisé !

[2] Mais, pas toujours... Typiquement, dans une telle situation, avec un langage tel que le C/C^{++}, l'ordinateur continuera d'effectuer les calculs, sans aucune protestation et fournira donc des résultats totalement incohérents, sans autre forme de procès. Des langages tels que *Fortran*, *Matlab* ou *Java* fournissent un meilleur suivi de telles erreurs. Selon le cas, le calculateur s'arrêtera après envoi d'un message d'erreur. Ou bien, il remplacera froidement le nombre par une valeur totalement différente et continuera les calculs (en signalant éventuellement le problème *overflow* ou *underflow* dans le meilleur des cas).

$x = 3.1234567845578$ et $y = 3.1234567844660$, respectivement. La différence exacte vaut $s = x - y = 0.0000000000918$. Le calcul approché en virgule flottante en utilisant 12 chiffres dans la mantisse donne par contre $\tilde{s} = \tilde{x} - \tilde{y} = 0.00000000009$. Puisque

$$\left|\frac{s - \tilde{s}}{s}\right| \approx 0.2\ 10^{-1} < \tfrac{1}{2}\ 10^{-1},$$

le résultat $\tilde{s}$ ne possède plus qu'un seul chiffre significatif ! On dira d'une méthode numérique ou d'un algorithme de calcul qui entraîne une telle propagation d'erreurs qu'ils sont *numériquement instables*. C'est typiquement, ce qu'on observe dans les formules de dérivation numérique. A ce stade, il est essentiel de clairement distinguer les concepts fondamentaux de stabilité numérique d'une méthode de calcul, de stabilité d'un problème mathématique et de stabilité d'un système physique.

Stabilité d'un problème physique : systèmes chaotiques

On dira d'un problème physique qu'il est *chaotique* si une petite variation des données initiales entraîne une variation totalement imprévisible des résultats. Cette notion de chaos est donc liée à la physique d'un problème et est observée expérimentalement. Elle est indépendante du modèle mathématique utilisé et encore plus de la méthode numérique utilisée pour résoudre ce modèle mathématique. De nombreux problèmes en physique sont dits chaotiques : citons en vrac, la turbulence en mécanique des fluides, les problèmes de fléchissement en génie civil, les comportements chaotiques des nombreux systèmes dynamiques.

Aujourd'hui, la modélisation mathématique et la simulation numérique de tels systèmes restent un défi pour l'ingénieur et le chercheur.

Stabilité d'un problème mathématique : sensibilité

Les données d'un problème mathématique ne sont jamais connues de manière exacte. Nombre de paramètres sont le résultat de mesures expérimentales et même des grandeurs mathématiques parfaites telles que π seront entachées d'erreurs, puisqu'on ne pourra pas représenter ce nombre de manière exacte dans un ordinateur. Le fait d'utiliser des données approximatives modifie quelque peu le problème initial à résoudre. En général, on espère que le fait de travailler sur des données approximatives ne modifie pas de manière importante le résultat. Toutefois, il se peut que cette petite variation des données conduise à une variation importante des résultats.

On dira d'un problème qu'il est *très sensible* ou *mal conditionné* si une petite variation des données entraîne une grande variation des résultats. Cette notion de conditionnement est donc liée au problème mathématique lui-même ; elle est indépendante de la méthode numérique utilisée pour résoudre ce problème. Pour modéliser un problème physique qui n'est pas chaotique, on construira donc un modèle mathématique qui sera le mieux condi-

tionné possible. Par exemple, le système linéaire à résoudre pour obtenir une interpolation polynomiale sera plus ou moins bien conditionné en fonction du choix des fonctions de base. Evidemment, la modélisation mathématique correcte d'un système chaotique risque fort d'être un système mal conditionné ! Un domaine important de la recherche est ainsi la mise au point de formulations de régularisation de problèmes physiquement instables.

Stabilité d'une méthode numérique : stabilité numérique

Le dernier concept fondamental est celui de *stabilité d'une méthode numérique*. Sans entrer dans un formalisme technique, nous dirons d'une méthode qu'elle est instable si elle est sujette à une propagation importante des erreurs numériques de discrétisation et d'arrondi. On observe ainsi que la soustraction de deux nombres proches est une méthode de calcul qui est numériquement instable. En général, on effectuera donc des manipulations algébriques afin de supprimer ce type d'opération lors de la construction d'algorithmes de calcul.

Un problème peut être bien conditionné alors que la méthode numérique choisie pour le résoudre est instable. Dans ce cas, il sera en général impératif d'utiliser une méthode numérique alternative (stable) pour résoudre le problème. Par contre, si le problème de départ est mal conditionné, aucune méthode numérique ne pourra y remédier. Il faudra alors essayer de trouver une formulation mathématique différente du même problème, si on sait que le problème physique sous-jacent est stable.

3.2.2 Erreurs d'arrondi et pas optimal pour les différences

Dans la dérivation numérique, un aspect essentiel est l'impact des erreurs d'arrondi. Nous allons donc effectuer une analyse plus détaillée. Supposons donc que nous souhaitions faire usage de la formule des différences centrées d'ordre deux.

$$u'(0) = \underbrace{\frac{(U_h - U_{-h})}{2h}}_{D^h} + \underbrace{(-1)\frac{h^2}{6}u^{(3)}(\xi)}_{E^h}$$

où E^h représente *l'erreur de discrétisation*. Maintenant, supposons que nous ne disposions que des représentations $\tilde{U}_h$ et $\tilde{U}_{-h}$, comme approximations de U_h ou U_{-h}. Plus précisément, nous écrivons

$$U_h = \tilde{U}_h + \tilde{e}_h$$

$$U_{-h} = \tilde{U}_{-h} + \tilde{e}_{-h}$$

où les termes $\tilde{e}_h$ et $\tilde{e}_{-h}$ sont *les erreurs d'arrondi* associées à la représentation en virgule

flottante des nombres réels U_h et U_{-h}.

L'utilisation des approximations $\tilde{U}_h$ et $\tilde{U}_{-h}$ dans la formule de différences centrées d'ordre deux modifie l'expression de l'erreur. Nous avons désormais la situation suivante :

$$
\begin{array}{rcl}
u'(0) & = & \dfrac{(U_h - U_{-h})}{2h} + (-1)\dfrac{h^2}{6}u^{(3)}(\xi) \\
& \downarrow & \\
u'(0) & = & \underbrace{\dfrac{(\tilde{U}_h - \tilde{U}_{-h})}{2h} + \dfrac{(\tilde{e}_h - \tilde{e}_{-h})}{2h}}_{D^h} + \underbrace{(-1)\dfrac{h^2}{6}u^{(3)}(\xi)}_{E^h} \\
& & \underbrace{\qquad\qquad}_{\tilde{D}^h} \quad \underbrace{\qquad\qquad\qquad\qquad}_{\tilde{E}^h}
\end{array}
$$

où $\tilde{D}^h$ est l'estimation numérique de la valeur de la dérivée obtenue en faisant usage des approximations, tandis que D^h représente l'estimation que l'on obtiendrait avec les valeurs exactes. Ici, nous nous intéressons à la propagation des erreurs et on constate que cette propagation se produira même si nous utilisons un calculateur parfait, puisque nous négligeons l'apparition d'une erreur d'arrondi lors du calcul de $\tilde{D}^h$. L'erreur totale $\tilde{E}^h = D - \tilde{D}^h$ se compose donc de deux termes : $D^h - \tilde{D}^h$ correspondant aux erreurs d'arrondi liées à l'usage d'une arithmétique avec une précision limitée et $E^h = D - D^h$ correspondant à l'erreur de discrétisation habituelle.

Si on peut trouver une estimation C qui borne la valeur absolue de la dérivée troisième de $u(x)$ et si on peut également trouver une estimation ϵ qui borne la valeur absolue des erreurs d'arrondi $\tilde{e}_h$ et $\tilde{e}_{-h}$, on écrit[3] la borne supérieure de la valeur absolue de l'erreur totale pour une formule de différences centrées du second ordre sous la forme :

$$
\boxed{|\tilde{E}^h| \leq \frac{\epsilon}{h} + \frac{C\ h^2}{6}} \tag{3.8}
$$

Tout comme on l'avait déjà observé dans l'évaluation numérique de la dérivée de $\exp(x)$ en $x = 1$, on voit que pour une grande valeur de h, l'erreur d'arrondi est généralement faible par rapport à l'erreur de discrétisation. La situation s'inverse lorsque l'on diminue h. Au vu d'un tel résultat, il est alors tentant d'essayer d'estimer pour quelle valeur de h la borne supérieure de l'erreur totale sera minimale : il suffit de chercher la valeur de h qui annule la dérivée de l'expression (3.8) :

[3]On tient compte du fait que la valeur absolue sur une différence est bornée par la somme des valeurs absolues des termes.

$$h = \left(\frac{3\,\epsilon}{C} \right)^{1/3} \tag{3.9}$$

Imaginons que nous souhaitions utiliser, avec un ordinateur, la formule des différences centrées d'ordre deux pour calculer la dérivée de la fonction $\cos(x)$ en $x = 0.7$. Ici, on peut facilement remplacer C par l'unité et la valeur de ϵ par 10^{-16}. L'allure de l'erreur totale en fonction de h est donnée sur la Figure (3.2), tandis qu'une estimation pour le pas optimal est immédiatement déduite à partir de (3.9). On obtient ainsi une estimation d'un pas optimal $h = 6.7\ 10^{-6}$ pour une erreur de l'ordre de $2.24\ 10^{-11}$. Afin de vérifier la validité d'une telle estimation, utilisons successivement des pas $h_k = 10^{-k}$ et effectuons le calcul de la différence centrée du second ordre avec `python`.

h	$\tilde{D}^h$	$\tilde{E}^h$
1	-0.54209049171057	0.10212719552713
0.1	-0.64314452781256	0.00107315942513
0.01	-0.64420695032992	0.00001073690777
0.001	-0.64421757986810	0.00000010736959
0.0001	-0.64421768616374	0.00000000107395
1.0000000e-005	-0.64421768722345	0.00000000001424
1.0000000e-006	-0.64421768725120	-0.00000000001351
1.0000000e-007	-0.64421768697365	0.00000000026404
1.0000000e-008	-0.64421769030432	-0.00000000306663
1.0000000e-009	-0.64421767920209	0.00000000803561
1.0000000e-010	-0.64421745715748	0.00000023008021
1.0000000e-011	-0.64421801226899	-0.00000032503130
1.0000000e-012	-0.64420691003875	0.00001077719894
1.0000000e-013	-0.64448446579490	-0.00026677855721
1.0000000e-014	-0.64392935428259	0.00028833295510
1.0000000e-015	-0.61062266354384	0.03359502369385
1.0000000e-016	-0.55511151231258	0.08910617492511
1.0000000e-017	0.00000000000000	0.64421768723769
1.0000000e-018	0.00000000000000	0.64421768723769
1.0000000e-019	0.00000000000000	0.64421768723769

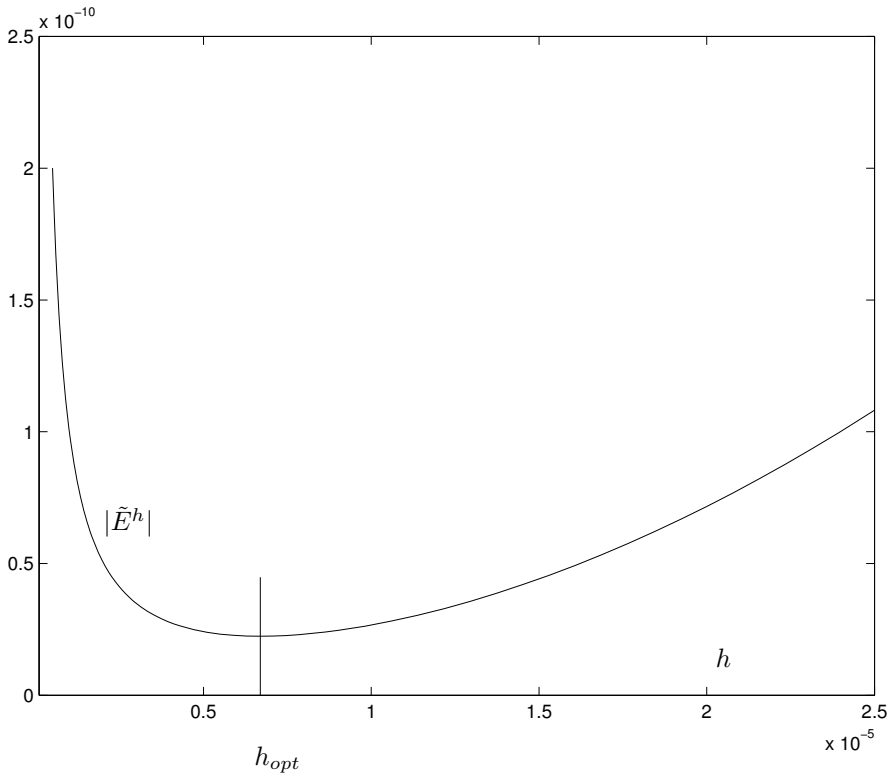


Figure 3.2: Comment trouver le pas optimal pour calculer la dérivée de $\cos(x)$? Calcul avec une formule de différence centrée du second ordre et une arithmétique en virgule flottante à double précision (On estime $\epsilon = 10^{-16}$). On peut alors estimer le pas optimal comme $h_{opt} = 6.6943\ 10^{-6}$ pour une borne sur l'erreur totale de $2.2407\ 10^{-11}$.

On observe que le meilleur résultat est obtenu avec un pas de 10^{-6} et que l'erreur observée vaut $1.3\ 10^{-11}$. Le tableau ci-dessous reprend les résultats d'une telle analyse pour l'estimation de la dérivée première ou seconde par des différences finies d'ordre deux ou d'ordre quatre.

$u'(0) \approx \dfrac{(U_h - U_{-h})}{2h}$ $\lvert\tilde{E}^h\rvert \leq \dfrac{\epsilon}{h} + \dfrac{C_3\, h^2}{6}$	$h_{opt} = \left(\dfrac{3\epsilon}{C_3}\right)^{1/3}$ (différences centrées) $\mathcal{O}(h^2)$
$u'(0) \approx \dfrac{(-U_{2h} + 8U_h - 8U_{-h} + U_{-2h})}{12h}$ $\lvert\tilde{E}^h\rvert \leq \dfrac{3\epsilon}{2h} + \dfrac{C_5\, h^4}{30}$	$h_{opt} = \left(\dfrac{45\epsilon}{4C_5}\right)^{1/5}$ (différences centrées) $\mathcal{O}(h^4)$
$u''(0) \approx \dfrac{(U_h - 2U_0 + U_{-h})}{h^2}$ $\lvert\tilde{E}^h\rvert \leq \dfrac{4\epsilon}{h^2} + \dfrac{C_4 h^2}{12}$	$h_{opt} = \left(\dfrac{48\epsilon}{C_4}\right)^{1/4}$ (différences centrées) $\mathcal{O}(h^2)$
$u''(0) \approx \dfrac{(-U_{2h} + 16U_h - 30U_0 + 16U_{-h} - U_{-2h})}{12h^2}$ $\lvert\tilde{E}^h\rvert \leq \dfrac{16\epsilon}{3h^2} + \dfrac{C_6 h^4}{90}$	$h_{opt} = \left(\dfrac{240\epsilon}{C_6}\right)^{1/6}$ (différences centrées) $\mathcal{O}(h^4)$

(3.10)

Extrapolation de Richardson

On peut conclure des considérations qui précèdent que le calcul numérique des dérivées doit se faire avec un pas h suffisamment petit pour que l'erreur de discrétisation soit acceptable, mais h ne peut pas être trop petit car quand h diminue, on constate que les erreurs d'arrondi augmentent. On se trouve donc confronté au problème du choix de h. Des formules ont été développées pour déterminer le pas h qui minimise la somme des

deux erreurs, mais cela ne signifie pas nécessairement que cette erreur totale minimale soit suffisamment faible.

Devant ce dilemme, on peut tirer avantage des propriétés de l'extrapolation de Richardson. En effet, cette technique permet, à partir de plusieurs évaluations de la dérivée pour des pas différents, de trouver une meilleure évaluation dans laquelle l'erreur de discrétisation a été réduite. On commence par évaluer la dérivée pour des pas h qui ne sont pas trop petits de façon à ce qu'ils ne soient pas entachés d'erreurs d'arrondi. Les erreurs de discrétisation que comportent alors ces évaluations peuvent être éliminées par des extrapolations de Richardson successives.

Reprenons le calcul de la dérivée de $f(x) = \cos(x)$ en $x = 0.7$ par la formule des différences centrées d'ordre 2. On peut y appliquer une extrapolation de Richardson en considérant les valeurs obtenues comme les valeurs d'une fonction $D(h)$, dont l'extrapolation en $h = 0$ vaudrait la dérivée recherchée. Il s'agit d'effectuer l'extrapolation à la limite suivante.

Estimer $D(0)$ sachant que

$$\begin{aligned} D(h) &= -0.64314452781256 \\ D(h/10) &= -0.64420695032992 \\ D(h/100) &= -0.64421757986810 \\ D(h/1000) &= -0.64421768616374 \end{aligned} \tag{3.11}$$

où les valeurs ont été calculées avec $h = 0.1$.

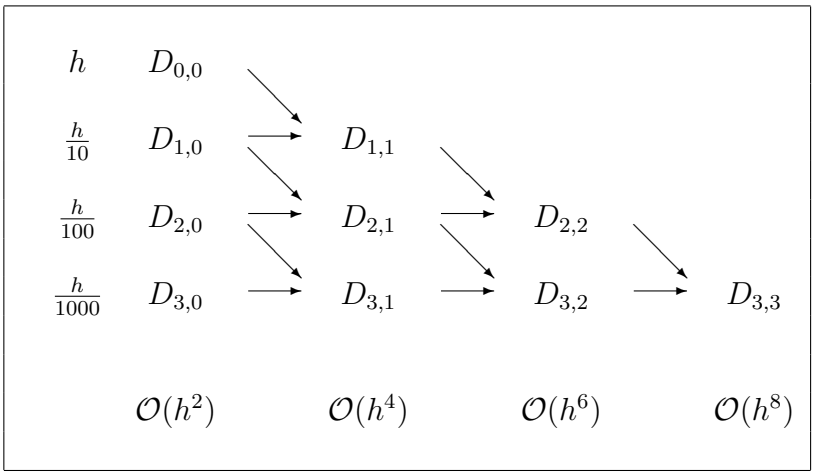
Pour établir la formule d'extrapolation à utiliser, on se base sur la propriété[4] que le terme d'erreur des différences centrées peut être représenté seulement par des puissances paires de h. La formule de l'extrapolation de Richardson peut alors s'écrire :

$$D_{i,k} = \frac{\left(D_{i,k-1} - \frac{1}{10^{2k}} D_{i-1,k-1}\right)}{\left(1 - \frac{1}{10^{2k}}\right)} \tag{3.12}$$

où les termes $D_{i,0} = D(h/10^i)$. Pour le calcul des $D_{i,1}$, la règle est : *100 fois la meilleure estimation moins la moins bonne, le tout divisé par 99.* Tandis que pour le calcul de $D_{i,2}$, la règle peut se lire : *10000 fois la meilleure estimation moins la moins bonne, le tout divisé par 9999.* L'étape suivante correspondra à l'élimination de l'erreur en h^6 : *1000000 fois la meilleure estimation moins la moins bonne, le tout divisé par 999999.*

Dans la pratique, on construit progressivement le tableau des $D_{i,k}$ comme suit :

[4] C'est exactement la même approche que ce que nous avons présenté pour la méthode de Romberg, dans le chapitre sur l'intégration numérique.



Le tableau ci-dessous reprend cette succession d'extrapolations calculées avec *Matlab*. On s'est limité dans la première colonne aux valeurs correspondant à des pas supérieurs à 10^{-4} de façon à ne travailler qu'avec des estimations de départ qui n'ont pas été trop contaminées par les erreurs d'arrondi. Pour donner une idée intuitive de la précision, les chiffres exacts de chaque étape intermédiaire ont été soulignés.

i	$D_{i,0}$	$D_{i,1}$	$D_{i,2}$	$D_{i,3}$
0	-0.64314452781256			
1	-0.64420695032992	-0.64421768187050		
2	-0.64421757986810	-0.64421768723717	-0.64421768723771	
3	-0.64421768616374	-0.64421768723743	-0.64421768723743	-0.64421768723743
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$

La valeur exacte est -0.64421768723769. On constate qu'il est maintenant possible de calculer la dérivée avec une erreur de 10^{-13} alors que l'on n'avait pas pu faire mieux que 10^{-11} sans extrapolation. Il n'est pas indispensable (ni recommandé) de démarrer les extrapolations avec des estimations sur des pas successivement divisés par 10. Dès qu'on atteint les limites de la précision de l'ordinateur, le processus d'extrapolation n'est plus d'aucun secours : le calcul du terme $D_{3,3}$ est totalement inutile.

Effectuons maintenant une extrapolation toujours avec $h = 10^{-1}$ mais une division successive des pas par un facteur deux. Les dénominateurs de l'étape k de l'extrapolation sont alors successivement $2^{2k} - 1$ au lieu de $10^{2k} - 1$. De même, les facteurs 2^{2k} remplacent 10^{2k} dans l'expression des numérateurs. Les résultats obtenus en extrapolant sur les valeurs $D_2(h)$ trouvées avec h, $h/2$, $h/4$ et $h/8$ sont repris ci-dessous.

i	$D_{i,0}$	$D_{i,1}$	$D_{i,2}$	$D_{i,3}$
0	-0.64314452781256			
1	-0.64394929675235	-0.64421755306561		
2	-0.64415058332564	-0.64421767885006	-0.64421768723569	
3	-0.64420091086648	-0.64421768723743	-0.64421768723765	-0.64421768723766
	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$

Bien que partant de valeurs beaucoup moins précises, on obtient une valeur extrapolée un tout petit peu plus précise, car on est moins victime d'erreurs d'arrondi.

Chapitre 4

Comment résoudre numériquement un problème aux valeurs initiales ?

Les équations différentielles sont très souvent utilisées dans la modélisation mathématique de la physique du monde de l'ingénieur. Trouver la solution d'une équation différentielle ordinaire avec une condition initiale est donc un problème courant qu'il est souvent difficile ou impossible de résoudre analytiquement. Dans ce chapitre, nous allons présenter les méthodes numériques les plus populaires pour la résolution d'équations différentielles ordinaires.

Typiquement, nous souhaitons résoudre le problème à la valeur initiale (Initial Value Problem) ou problème de Cauchy défini par

Trouver $u(x)$ tel que

$$\begin{cases} u'(x) &= f(x, u(x)), & x \in [a, b] \\ u(a) &= \bar{u} \end{cases}$$

La fonction à deux variables $f : (x, u) \to f(x, u)$ et la valeur initiale $\bar{u}$ sont des données du problème.

Pour obtenir une *approximation* de la fonction $u(x)$ sur l'intervalle de calcul $[a, b]$, nous allons estimer la valeur de cette fonction en un nombre fini de points $a = X_0 < X_1 < X_2 \ldots < X_m = b$. L'écart entre deux abscisses est appelé *pas de discrétisation* et est noté h. Dans les méthodes les plus simples, il sera constant, mais il peut être judicieux de travailler avec un pas variable $h_i = X_i - X_{i-1}$. Nous recherchons donc une suite de valeurs approchées :

$$u(X_i) \approx u^h(X_i) = U_i \tag{4.1}$$

Il s'agit bien de calculer l'approximation d'une fonction inconnue. Les techniques de résolution d'équations différentielles ordinaires vont donc se baser sur l'approximation de fonctions et l'intégration (ou la dérivation) numérique.

4.1 Stabilité d'une équation différentielle

Avant de présenter les méthodes utilisées pour la résolution de problèmes différentiels avec une valeur initiale, il est utile de présenter quelques équations typiques que l'on risque de rencontrer. En particulier, nous allons nous intéresser au concept de *stabilité* d'une équation différentielle qui est un concept fondamental dans la modélisation mathématique et l'analyse de phénomènes physiques.

Une équation différentielle stable, ...

Pour illustrer le concept de *stabilité* d'une équation différentielle, considérons le cas particulier $f(x,u) = -u$. Le problème à la valeur initiale qui y correspond est défini par :

Trouver $u(x)$ tel que

$$\begin{cases} u'(x) &= -u(x), & x \in [a,b] \\ u(a) &= \bar{u} \end{cases} \tag{4.2}$$

L'ensemble des solutions d'une équation différentielle paramétrées par C est appelé *la famille de solutions* de l'équation différentielle. Pour notre exemple, cette famille de solutions est donnée par $u(x) = Ce^{-x}$. La constante d'intégration C est déterminée de manière unique lorsque l'on recherche la solution du problème (4.2) :

$$u(x) = \bar{u}\ e^{-(x-a)}$$

On observe à la Figure 4.1 que les membres de la famille de solutions $u(x) = Ce^{-x}$ *se rejoignent* lorsque $x \to \infty$. Les équations différentielles $u' = f(x,u)$ qui possèdent cette propriété sont dites *stables*. Afin de comprendre le concept de *stabilité* ou de *sensibilité aux perturbations* d'une équation différentielle, considérons la perturbation du problème (4.2) :

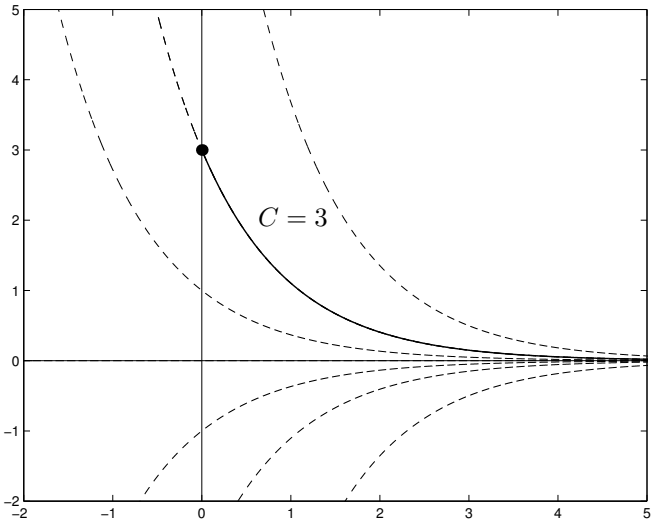


Figure 4.1: Famille de solutions de l'équation différentielle ordinaire $u'(x) = -u(x)$. La courbe caractérisée correspondant à la valeur initiale $u(0) = 3$ est mise en évidence.

Trouver $u_\epsilon(x)$ tel que

$$\begin{cases} u'_\epsilon(x) &= -u_\epsilon(x), & x \in [a,b] \\ u_\epsilon(a) &= \bar{u} + \epsilon \end{cases} \tag{4.3}$$

La condition initiale est *perturbée* d'une quantité ϵ. La solution du problème perturbé est $u_\epsilon(x) = (\bar{u} + \epsilon)\ e^{-(x-a)}$. L'écart entre la solution du problème perturbé et celle du problème non perturbé est donné par

$$\underbrace{u_\epsilon(x) - u(x)}_{e_\epsilon(x)} = \epsilon\ e^{-(x-a)}$$

On observe que cet écart tend vers 0 lorsque x augmente : l'effet de la perturbation s'évanouit pour de grandes valeurs de x et l'équation différentielle $u'(x) = -u(x)$ peut être considérée comme le prototype d'une équation différentielle stable.

... une équation différentielle instable

Considérons maintenant l'équation $u'(x) = u(x)$. La famille de solutions est donnée par $u(x) = Ce^x$. La constante d'intégration C est déterminée de manière unique lorsque l'on recherche la solution du problème à la valeur initiale $u(a) = \bar{u}$:

$$u(x) = \bar{u}\ e^{(x-a)}$$

L'écart entre les solutions des problèmes perturbé et non perturbé vaut

$$\underbrace{u_\epsilon(x) - u(x)}_{e_\epsilon(x)} = \epsilon\ e^{(x-a)}$$

et croît exponentiellement avec x. Les membres de la famille de solutions de l'équation $u'(x) = u(x)$ sont des exponentielles croissantes. Sur la Figure 4.2, ces membres de la famille de solutions $u(x) = Ce^x$ *s'écartent* lorsque $x \to \infty$. L'équation différentielle $u'(x) = u(x)$ peut être considérée comme le prototype d'une équation différentielle instable.

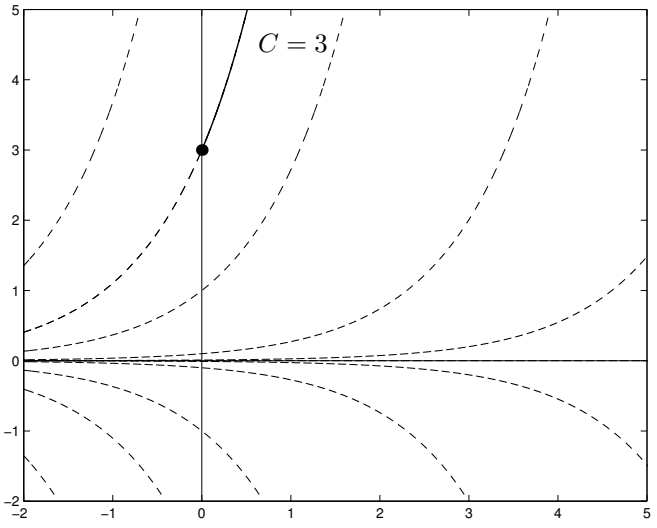


Figure 4.2: Famille de solutions de l'équation différentielle ordinaire $u'(x) = u(x)$. La courbe caractérisée correspondant à la valeur initiale $u(0) = 3$ est mise en évidence.

Une équation différentielle ni stable, ni instable

Toutes les équations différentielles ne sont pas stables ou instables. Afin d'illustrer ce propos, considérons l'équation différentielle $u'(x) = 1 - e^{-x}$. Cette équation différentielle admet la famille de solutions

$$u(x) = C + x + e^{-x}$$

paramétrée par la constante C. L'imposition de la condition initiale en $u(a) = \bar{u}$ permet de choisir le membre de la famille de solutions qui satisfait au problème à la valeur initiale.

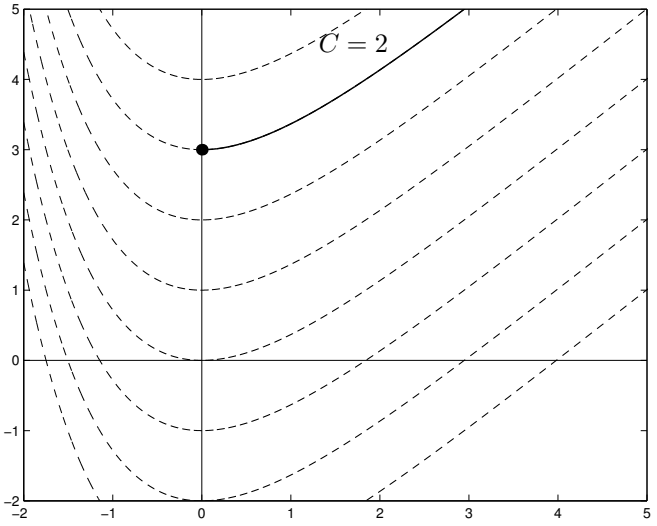


Figure 4.3: Famille de solutions de l'équation différentielle ordinaire $u'(x) = 1 - e^{-x}$. La courbe caractérisée par $C = 2$ pour la valeur initiale $u(0) = 3$ est mise en évidence.

Sur la Figure 4.3, on voit que l'écart entre deux membres de la famille de solutions reste toujours constant. En conséquence, l'écart entre les solutions des problèmes perturbé et non perturbé vaut

$$\underbrace{u_\epsilon(x) - u(x)}_{e_\epsilon(x)} = \epsilon$$

Une équation différentielle un peu instable et puis stable

Soit l'équation différentielle du premier ordre $u'(x) = -10(x-1)\ u(x)$ dont la famille de solutions, paramétrée par C, est

$$u(x) = Ce^{-5(x-1)^2}$$

Sur la Figure 4.4, nous observons que les solutions s'éloignent les unes des autres pour $x < 1$ (comportement instable) pour se rejoindre ensuite lorsque $x > 1$ (comportement stable).

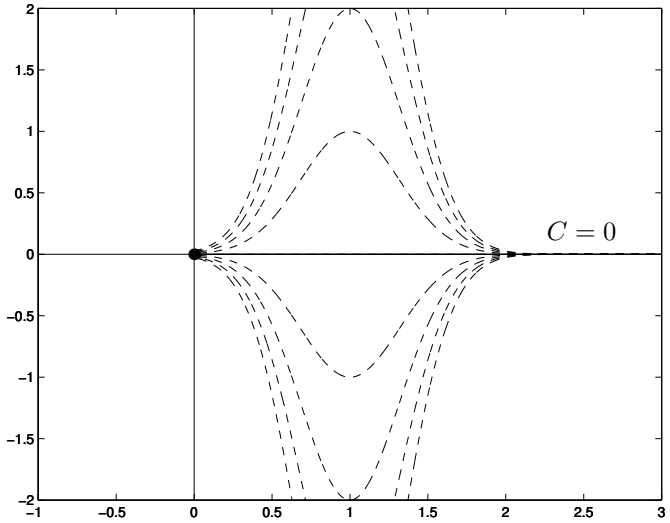


Figure 4.4: Famille de solutions de l'équation différentielle ordinaire $u'(x) = -10(x-1)\ u(x)$. La courbe caractérisée par $C = 0$, c'est-à-dire, la droite horizontale passant par l'origine est la solution passant par ce point.

Comment savoir si un problème sera stable dans le cas général $u'(x) = f(x, u(x))$? Nous allons observer que toute l'information requise pour répondre de manière locale à cette question est incluse dans ce qu'on appelle le *jacobien* d'une équation différentielle. En soustrayant, membre à membre, les expressions différentielles générales correspondant à (4.2) et (4.3), on peut écrire :

$$
\begin{aligned}
e'_\epsilon(x) &= f(x, u_\epsilon(x)) - f(x, u(x)) \\
&\Big\downarrow \quad \text{En effectuant un développement de Taylor de } f(x,u) \\
&\quad \text{pour la seconde variable } u \text{ autour du point } (x, u(x)), \\
&\approx f(x, u(x)) + \left.\frac{\partial f}{\partial u}\right|_{(x,u)=(x,u(x))} \Big(u_\epsilon(x) - u(x) \Big) - f(x, u(x)) \\
&\Big\downarrow \\
&\approx \left.\frac{\partial f}{\partial u}\right|_{(x,u)=(x,u(x))} e_\epsilon(x)
\end{aligned}
$$

Toute l'information concernant la stabilité locale de l'équation différentielle est donc contenue dans la dérivée partielle de f par rapport à la seconde variable. Ce terme est appelé le *jacobien* de l'équation et est défini par

$$
J(x, u(x)) = \left.\frac{\partial f}{\partial u}\right|_{(x,u(x))} \tag{4.4}
$$

Dans le voisinage de $(x, u(x))$, on peut donc dire que l'écart $e_\epsilon(x)$ peut être interprété comme la solution du problème dont on a une idée approximative de l'équation différentielle :

Trouver $e_\epsilon(x)$ tel que

$$
\begin{cases}
e'_\epsilon(x) \approx \underbrace{\left.\dfrac{\partial f}{\partial u}\right|_{(x,u(x))}}_{J(x,u(x))} e_\epsilon(x), & x \in [a, b] \\
e_\epsilon(a) = \epsilon
\end{cases} \tag{4.5}
$$

En calculant la solution du problème à la valeur initiale, on obtient une estimation du comportement de l'écart.

$$
e_\epsilon(x) \approx \epsilon \exp\left(\int_a^x J(s, u(s)) ds \right)
$$

On peut en déduire que *si la solution du problème différentiel* $u' = f(x, u)$ *est telle que le*

jacobien est négatif, alors l'écart entre les solutions des problèmes perturbé et non perturbé tendra vers 0 lorsque x augmente ; l'équation différentielle sera alors qualifiée de stable.

Revenons aux exemples considérés plus haut. Pour $u' = -u$ (problème stable), nous avons $J = -1$, qui est bien négatif. Pour $u' = u$ (problème instable), le jacobien vaut $J = 1$ et est bien positif. Dans le cas particulier où la fonction $f(x, u)$ est indépendante de u, le jacobien J est égal à zéro et l'écart se maintient à la valeur initiale ϵ. Enfin, pour certaines fonctions f, il peut arriver que le jacobien soit parfois négatif et parfois positif. Dans le dernier exemple, on observe un comportement instable, puis stable : on observe aussi que le jacobien $J = -10(x-1)$ est positif pour $x < 1$ et négatif pour $x > 1$. En toute généralité, c'est encore plus compliqué, puisque le jacobien peut également dépendre de $u(x)$...

Equations différentielles stables, mais ... raides

Il est intuitivement raisonnable de penser qu'une méthode numérique aura des difficultés à résoudre un problème différentiel instable. Pouvons-nous, par contre, supposer que la résolution numérique de problèmes différentiels stables sera une tâche aisée ? De manière étonnante, la réponse est négative : nous allons observer que beaucoup de méthodes numériques classiques sont inadéquates pour la résolution de problèmes différentiels *fortement stables* (c'est-à-dire, tels que $J \ll 0$). De tels problèmes sont appelés *problèmes raides* ou *stiff problems*. A titre d'exemple, considérons le problème à la valeur initiale :

Trouver $u(x)$ tel que

$$\left\{ \begin{array}{rcll} u'(x) & = & -\alpha \Big(u(x) - \sin(x) \Big) + \cos(x), & x \in [0, 5] \\ u(0) & = & 1 & \end{array} \right. \tag{4.6}$$

où α est un paramètre permettant d'ajuster la raideur du problème, puisque le jacobien vaut $-\alpha$. La solution du problème est :

$$u(x) = e^{-\alpha x} + \sin(x).$$

Pour $\alpha = 1000$, nous avons un problème raide ($J = -1000 \ll 0$). Comme le montre la Figure 4.5, la solution décroît extrêmement rapidement au voisinage de $x = 0$, après quoi elle adopte le comportement régulier de la fonction sinus. Le problème (4.6) avec $\alpha = 1000$ est fortement stable ou raide ; il sera très difficile à résoudre numériquement et nécessitera l'usage de méthodes spécialement conçues pour de tels problèmes.

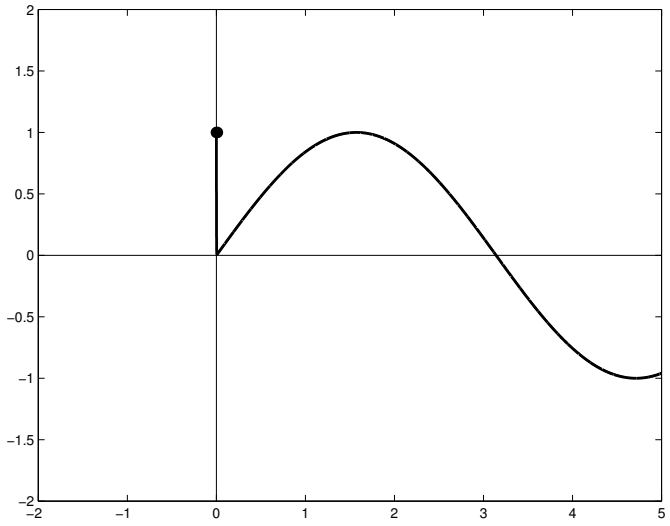


Figure 4.5: Solution $u(x) = e^{-\alpha x} + \sin(x)$ d'un problème raide ($\alpha = 1000$). Quelle que soit la condition initiale, toutes les solutions vont très rapidement adopter le comportement régulier de la fonction sinus.

4.2 Méthodes de Taylor

Nous voulons déterminer de manière approchée la fonction $u(x)$ vérifiant l'équation différentielle scalaire du premier ordre $u'(x) = f(x, u(x))$ sur l'intervalle $[a, b]$ avec la condition initiale $u(a) = U_0$. Nous allons donc chercher une approximation U_i de la fonction u en un nombre fini de points X_i de l'intervalle $[a, b]$. Le choix le plus simple des points X_i est obtenu en subdivisant l'intervalle $[a, b]$ en m sous-intervalles de taille uniforme h :

$$X_i = a + hi \quad i = 0, \ldots, m,$$

où $h = (b - a)/m$ est le pas de la discrétisation. Le problème consiste donc à générer la suite $(U_1, U_2, \ldots, U_m)$ à partir de la condition initiale U_0. Pour obtenir une telle suite d'estimations, nous allons développer de manière systématique des méthodes sur base du développement de Taylor de la solution exacte $u(x)$. On construit ainsi les méthodes dites de Taylor.

Ecrivons le développement en série de Taylor de la solution exacte $u(x)$ de l'équation

$$u'(x) = \underbrace{f(x, u(x))}_{\widehat{f}(x)}$$

autour du point de départ $x = X_0$. Nous introduisons ici $\widehat{f}$ pour identifier la fonction à

une variable $\widehat{f} : x \to f(x, u(x))$.

$$
\begin{aligned}
u(x) &= u(X_0) + (x - X_0)\, u'(X_0) + \frac{(x - X_0)^2}{2!}\, u''(X_0) + \cdots \\
&\Big\downarrow \quad \text{En vertu du problème à la valeur initiale,} \\
&= U_0 + (x - X_0)\, \widehat{f}(X_0) + \frac{(x - X_0)^2}{2!}\, \widehat{f}'(X_0) + \cdots \\
&\Big\downarrow \quad \text{En tenant compte que } \widehat{f}' = \tfrac{\partial f}{\partial x} + \tfrac{\partial f}{\partial u} u' \text{ et que } u' = \widehat{f} \\
&= U_0 + (x - X_0)\, f(X_0, U_0) \\
&\quad + \frac{(x - X_0)^2}{2!} \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u} f \right)_{(X_0, U_0)} \\
&\quad + \frac{(x - X_0)^3}{3!} \left(\frac{\partial^2 f}{\partial x^2} + 2 \frac{\partial^2 f}{\partial x \partial u} f + \frac{\partial^2 f}{\partial u^2} f^2 + \frac{\partial f}{\partial x} \frac{\partial f}{\partial u} + \left(\frac{\partial f}{\partial u} \right)^2 f \right)_{(X_0, U_0)} \\
&\quad + \cdots
\end{aligned}
$$

Le calcul des dérivées successives de $\widehat{f}(x)$ au point X_0 peut donc être réalisé à partir de f et de ses dérivées partielles. Il est donc possible d'écrire les dérivées de u au point X_0 en termes de f et de dérivées partielles de f évaluées en $(X_0, u(X_0))$. On peut ainsi calculer une approximation U_1 de la fonction u en X_1 à l'aide de ce développement tronqué à l'ordre n :

$$
\begin{aligned}
U_1 &= U_0 + (X_1 - X_0) f(X_0, U_0) \\
&\quad + \frac{(X_1 - X_0)^2}{2!} \left. \frac{df}{dx} \right|_{u' = f} (X_0, U_0) \\
&\quad + \cdots \\
&\quad + \frac{(X_1 - X_0)^n}{n!} \left. \frac{d^{(n-1)} f}{dx^{n-1}} \right|_{u' = f} (X_0, U_0)
\end{aligned}
\tag{4.7}
$$

où l'opérateur de dérivation est défini comme la dérivée totale de f sur les trajectoires de l'équation différentielle. En d'autres mots, l'opérateur de dérivation est défini par :

$$\left.\frac{d}{dx}\right|_{u'=f} = \left(\frac{\partial}{\partial x} + f\frac{\partial}{\partial u}\right)$$

Ce calcul est possible si nous disposons de X_0, X_1, U_0 et d'une expression de f et de ses dérivées partielles. Ayant ainsi déterminé U_1, nous pouvons obtenir l'approximation U_2 au point suivant X_2 à l'aide d'un développement similaire à (4.7), effectué cette fois autour de X_1.

Nous obtenons ainsi la *méthode de Taylor d'ordre n*, qui consiste à calculer, de proche en proche, les valeurs approchées U_{i+1} à l'aide de la formule de récurrence

$$\boxed{U_{i+1} = U_i + h \underbrace{\left[f + \frac{h}{2!}\left.\frac{df}{dx}\right|_{u'=f} + \cdots + \frac{h^{n-1}}{n!}\left.\frac{d^{(n-1)}f}{dx^{n-1}}\right|_{u'=f}\right]_{(X_i,U_i)}}_{\Phi(X_i,U_i)}} \tag{4.8}$$

où la fonction Φ est appelée la *fonction d'incrément* de la méthode. Les méthodes de Taylor sont ainsi dites *à pas séparés* ou *single-step methods* : le résultat U_i relatif au point X_i suffit pour calculer la nouvelle approximation U_{i+1} au point X_{i+1}. Mais surtout, ces méthodes sont également dites *méthodes explicites*, car il n'est pas nécessaire d'évaluer $f(X_{i+1}, U_{i+1})$ pour obtenir U_{i+1}. La méthode de Taylor la plus simple (obtenue avec $n = 1$) est la méthode d'Euler explicite :

$$U_{i+1} = U_i + h\ f(X_i, U_i) \tag{4.9}$$

A l'exception de la méthode d'ordre 1, les méthodes de Taylor sont peu utilisées en pratique, en raison du calcul fastidieux des dérivées totales du membre de droite de l'équation différentielle. Il faut noter cependant que l'apparition au cours des dix dernières années de logiciels de calcul symbolique permet d'automatiser facilement cette phase de la mise en oeuvre de l'algorithme de Taylor.

4.2.1 Interprétation géométrique de la méthode d'Euler explicite

La méthode d'Euler explicite peut s'interpréter géométriquement comme suit. Nous démarrons d'un point (X_0, U_0) et calculons la valeur de la pente $f(X_0, U_0)$ de la trajectoire passant par ce point. Nous nous déplaçons horizontalement d'une distance h et verticalement d'une distance $hf(X_0, U_0)$. En d'autres mots, nous nous déplaçons le long de la droite tangente à la courbe $u(x)$ et arrivons au point (X_1, U_1). Il est essentiel d'observer que ce point (X_1, U_1) n'est *pas* sur la courbe de la solution recherchée $u(x)$. On est seulement en train d'obtenir une approximation $U_1 = u^h(X_1) \approx u(X_1)$. Ensuite,

on utilise U_1 comme la meilleure estimation de $u(X_1)$ dont on dispose et on reproduit la même procédure de calcul, comme indiqué sur la Figure (4.6).

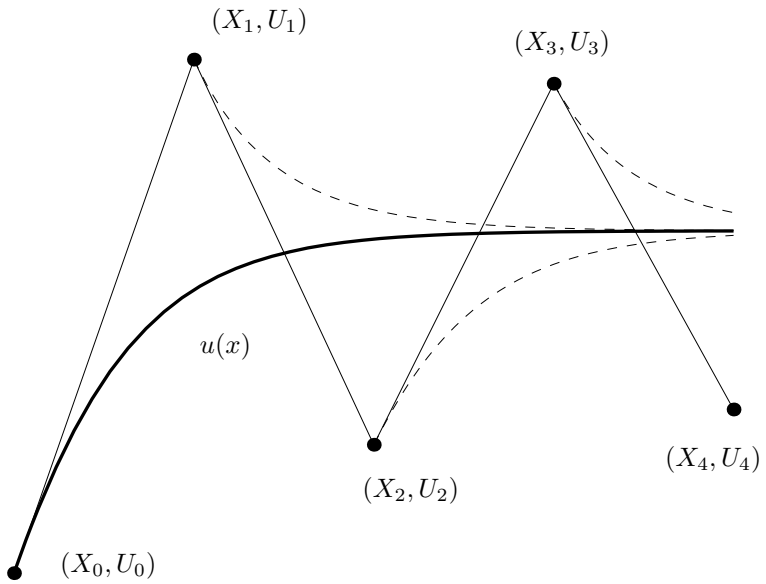


Figure 4.6: Interprétation géométrique de la méthode d'Euler : les courbes appartiennent à la famille de solutions de $u' = 2 - 0.5u^2$. La valeur initiale est $X_0 = 0$ et $U_0 = 1$.

A titre d'exemple, considérons le problème à la valeur initiale suivant :

Trouver $u(x)$ tel que

$$\begin{cases} u'(x) = (x - u(x))/2, & x \in [0, 3] \\ u(0) = 1 \end{cases} \tag{4.10}$$

La solution exacte est $u(x) = 3e^{-x/2} - 2 + x$. Nous allons maintenant comparer les solutions approchées obtenues avec la méthode d'Euler respectivement pour $h = 1$, $h/2$, $h/4$ et $h/8$. Le tableau ci-dessous et la Figure 4.7 illustrent les résultats obtenus. Pour h fixé, chaque couple de points consécutifs (X_i, U_i) et (X_{i+1}, U_{i+1}) est relié par un segment de droite. On constate la convergence vers la solution exacte lorsque h tend vers zéro.

Méthode d'Euler explicite pour $u' = (x-u)/2$					
X_i	$u^h(X_i)$	$u^{h/2}(X_i)$	$u^{h/4}(X_i)$	$u^{h/8}(X_i)$	$u(X_i)$
0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
0.1250				0.9375	0.9432
0.2500			0.8750	0.8867	0.8975
0.3750				0.8469	0.8621
0.5000		0.7500	0.7969	0.8174	0.8364
0.7500			0.7598	0.7868	0.8119
1.0000	0.5000	0.6875	0.7585	0.7902	0.8196
2.0000	0.7500	0.9492	1.0308	1.0682	1.1036
3.0000	1.3750	1.5339	1.6043	1.6374	1.6694

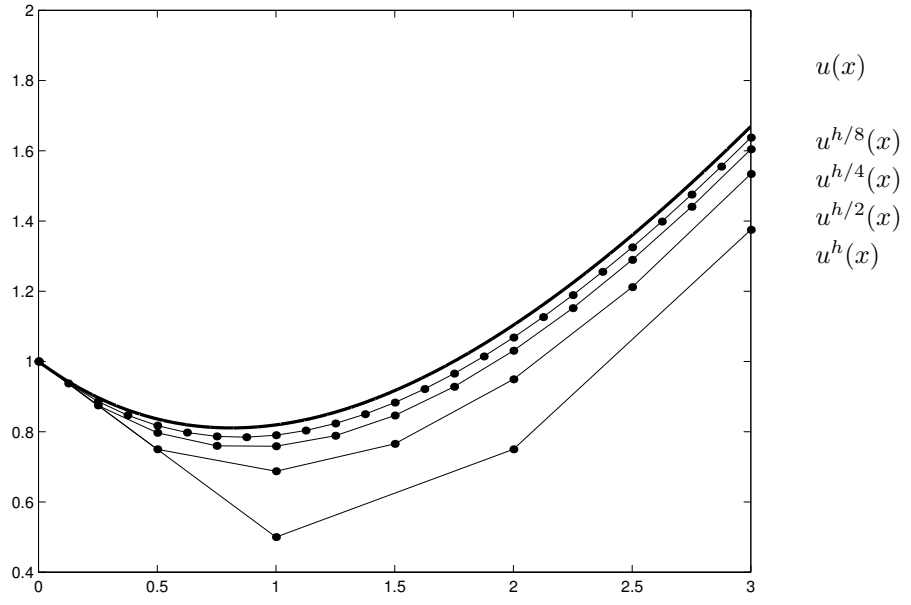


Figure 4.7: Convergence de la méthode d'Euler explicite : comparaison des solutions approchées obtenues avec la méthode d'Euler respectivement pour $h = 1$, $h/2$, $h/4$ et $h/8$. Le problème à résoudre est $u' = (x-u)/2$ avec la condition initiale $u(0) = 1$.

4.2.2 Estimation de l'erreur et analyse de la stabilité

Nous allons maintenant tenter d'estimer l'erreur de discrétisation à chaque pas :

$$e^h(X_i) = u(X_i) - \underbrace{u^h(X_i)}_{U_i}$$

Pour obtenir une telle estimation, nous allons utiliser le théorème du reste de Taylor et écrire que :

$$u(X_{i+1}) = u(X_i) + h\ f(X_i, u(X_i)) + \frac{h^2}{2}\ u''(\xi_{i+1}), \tag{4.11}$$

où ξ_{i+1} est un point particulier de l'intervalle $[X_i, X_{i+1}]$. La différence de la formule de la méthode d'Euler explicite (4.9) et de cette équation (4.11) donne

$$\underbrace{u(X_{i+1}) - U_{i+1}}_{e^h(X_{i+1})} = \underbrace{u(X_i) - U_i}_{e^h(X_i)} + h\ \Big(f(X_i, u(X_i)) - f(X_i, U_i) \Big) + \frac{h^2}{2}\ u''(\xi_{i+1})$$

$\downarrow$

Si f est continue et différentiable par rapport à u,
on peut utiliser le théorème de la valeur moyenne.
Et il existe ζ_i dans l'intervalle entre U_i et $u(X_i)$ tel que :

$$e^h(X_{i+1}) = e^h(X_i) + h\ \underbrace{\Big(u(X_i) - U_i \Big)}_{e^h(X_i)} \left.\frac{\partial f}{\partial u}\right|_{(X_i,\zeta_i)} + \frac{h^2}{2}\ u''(\xi_{i+1})$$

$\downarrow$

En définissant $J_i = \left.\frac{\partial f}{\partial u}\right|_{(X_i,\zeta_i)}$ et $e_{i+1} = \frac{h^2}{2}\ u''(\xi_{i+1})$,

$$e^h(X_{i+1}) = e^h(X_i)\ \Big(1 + hJ_i \Big) + e_{i+1}$$

où le terme $e_{i+1} = (h^2/2)\ u''(\xi_{i+1})$ est appelé *erreur locale de discrétisation* commise lors d'un pas de la méthode d'Euler. Par opposition, l'erreur $e^h(X_{i+1})$ est appelée *erreur globale de discrétisation*. Nous voyons que l'erreur globale du pas précédent contribue à l'erreur globale du pas en cours par l'intermédiaire d'un *facteur de multiplication*

$$a_i = (1 + hJ_i)$$

Ce facteur dépend du pas de discrétisation h. Il dépend également du jacobien de l'équation différentielle. Les concepts d'erreur locale et globale sont illustrés à la Figure 4.8. L'erreur globale à chaque pas de la méthode d'Euler est la somme de deux

termes. Le premier terme provient de l'erreur globale du pas précédent : c'est l'erreur *propagée*. Le second terme est l'erreur locale du pas en cours.

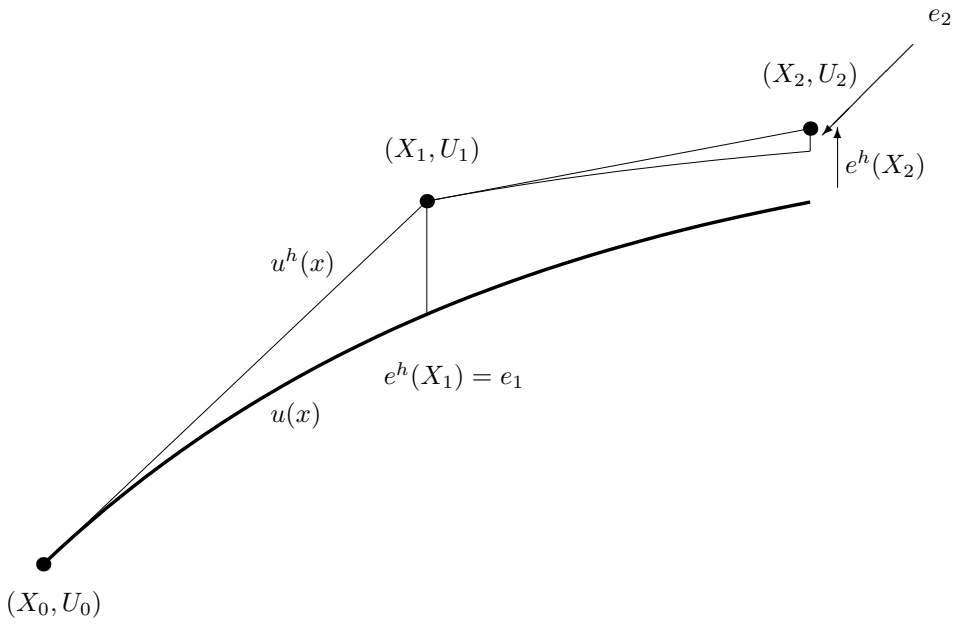


Figure 4.8: Erreurs locale et globale de la méthode d'Euler.

L'évolution de l'erreur globale de la méthode d'Euler explicite en cours d'intégration peut maintenant être écrite pour chaque pas.

$$
\begin{aligned}
e^h(X_1) &= e_1 \\
e^h(X_2) &= a_1\ e_1\ +\ e_2 \\
e^h(X_3) &= a_2 a_1\ e_1\ +\ a_2\ e_2\ +\ e_3
\end{aligned}
$$

$$\dots$$

$$
e^h(X_{i+1}) = \underbrace{a_i a_{i-1} \dots a_2 a_1\ e_1\ +\ a_i \dots a_2\ e_2\ +\ \dots\ +\ a_i e_i}_{\text{erreur propagée}} + e_{i+1}
$$

On peut observer comment les erreurs locales $e_1, e_2 \dots e_i$ commises lors des pas précédents contribuent à l'erreur globale $e^h(X_{i+1})$ à la fin du pas en cours.

Nous sommes maintenant en mesure d'étudier la stabilité de la méthode d'Euler. Pour rappel, une méthode est qualifiée de *stable* si les erreurs numériques ne croissent pas en cours de calcul. Il est clair que l'erreur globale de la méthode d'Euler ne va cesser de croître (en valeur absolue) d'un pas à l'autre si tous les facteurs d'amplification a_k ont une norme supérieure à 1. A l'opposé, si tous les a_k sont de norme inférieure à 1, l'erreur globale au pas courant restera "sous contrôle". Elle sera d'autant moins influencée par

les erreurs locales commises aux pas antérieurs que ceux-ci sont lointains. Dans ce cas, la méthode sera donc stable.

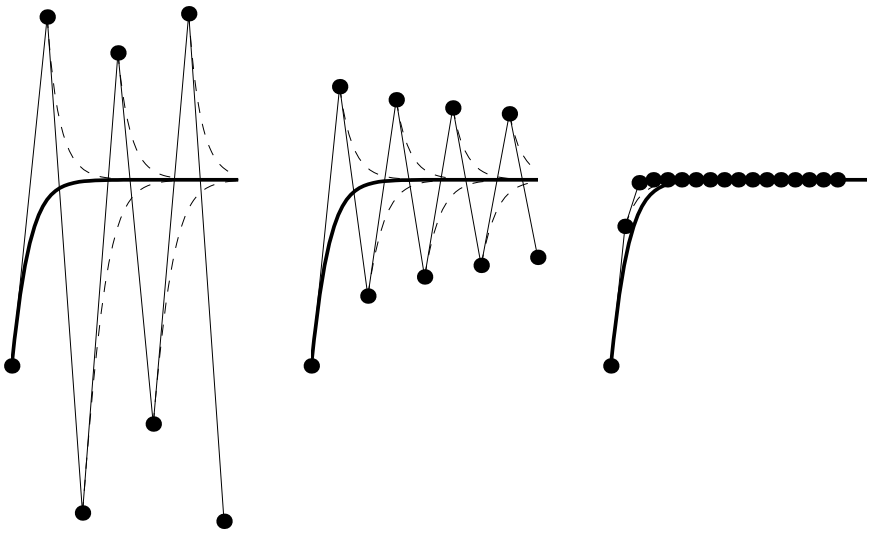


Figure 4.9: Instabilité ou stabilité de la méthode d'Euler : solution approchée de l'équation différentielle stable $u' = 2 - 0.5u^2$ avec la condition initiale $u(0) = 1$. Pour un pas $h = 1.25$ à gauche, on observe un comportement instable de la méthode d'Euler. Pour un pas $h = 0.5$ à droite, la méthode a un comportement stable et l'approximation numérique est excellente à l'exception des premiers pas. Pour le cas $h = 1$, on voit que l'erreur apparue aux premiers pas n'explose pas mais ne diminue pas non plus : c'est la limite entre comportements instables et stables.

La méthode d'Euler est stable si

$$\begin{array}{rcll} |\overbrace{1 + hJ_i}^{a_i}| & < & 1 & \forall i, \\ & & \downarrow & \\ -2 & < & hJ_i < 0 & \forall i, \end{array}$$

Lorsque le problème différentiel à résoudre est instable ($J_i > 0$), cette condition ne peut jamais être satisfaite, quelle que soit la valeur du pas h ($h > 0$) ; la méthode d'Euler est donc toujours instable. Si le problème différentiel est stable ($J_i < 0$), la condition de stabilité se réduit à

$$h < \frac{-2}{J_i}. \tag{4.12}$$

Autrement dit, le pas de discrétisation h utilisé dans la méthode d'Euler doit être plus petit qu'une valeur critique pour que la méthode soit stable. On dira que la méthode d'Euler explicite est *conditionnellement stable*.

La stabilité des méthodes de résolution d'équations différentielles est un sujet important dont l'étude détaillée déborde du cadre d'un cours introductif. Quoique fort partielle, cette analyse peut fournir des informations très utiles au numéricien. L'étude de la stabilité dans la résolution du problème général $u' = f(x, u)$ est complexe, mais peut être approchée par l'évaluation du comportement d'une méthode numérique sur un *problème modèle*

$$\begin{cases} u' = \lambda u \\ u(0) = 1 \end{cases} \tag{4.13}$$

où λ est un nombre complexe quelconque[1]. La solution exacte du problème modèle est donnée par $u(x) = e^{\lambda x}$. Le lien entre le problème modèle et le cas général $u' = f(x, u)$ est plus fort qu'il n'y paraît. Le problème modèle peut être vu comme la linéarisation du cas général en un point précis et λ comme la valeur du jacobien de f en ce point. Une méthode numérique est stable pour ce problème modèle si les erreurs ne croissent pas en cours de calcul. Dans le cas présent, il s'agit d'exiger que U_i tende vers zéro lorsque i tend vers l'infini. Cela revient à déterminer les valeurs de $h\lambda$ telles que le facteur d'amplification soit de norme strictement inférieure à 1

$$|1 + h\lambda| < 1$$

C'est pourquoi, dans la littérature, on représente la région de stabilité de la méthode d'Euler explicite comme un disque dans le plan complexe centré autour de -1 et de rayon 1. Si $h\lambda$ est dans ce cercle, la méthode d'Euler a un comportement numériquement stable. Pour λ réel négatif, on retrouve bien la condition de stabilité $-2 < h\lambda < 0$.

Cette analyse de la stabilité de la méthode d'Euler peut être étendue aux méthodes de Taylor (ainsi qu'à toute autre méthode à pas séparés). Sous des conditions appropriées de régularité du problème différentiel étudié, on peut montrer que l'erreur globale $e^h(X_{i+1})$ d'une méthode à pas séparés vérifie une expression de la forme

$$e^h(X_{i+1}) = a_i \; e^h(X_i) + e_{i+1}$$

[1] Considérer le cas des valeurs réelles et complexes sera utile pour pouvoir généraliser notre propos au cas des systèmes différentiels. Pour le cas scalaire, on pourrait se contenter de considérer uniquement le cas des λ réels.

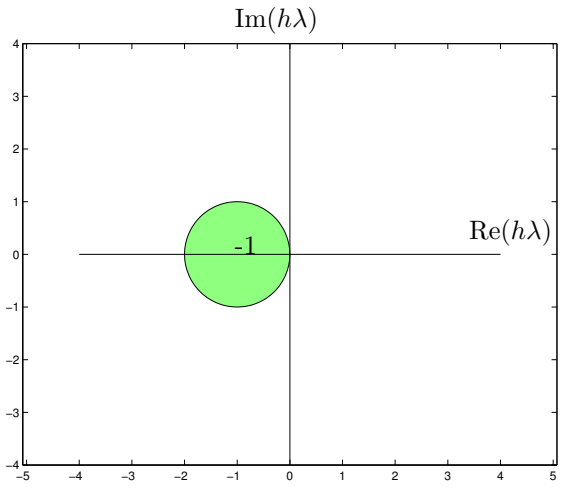


Figure 4.10: Région de stabilité de la méthode d'Euler explicite.

similaire au cas particulier obtenu pour la méthode d'Euler. Dans cette expression, a_i est le facteur d'amplification et e_{i+1} est l'erreur locale de discrétisation. Pour la méthode de Taylor d'ordre n, l'erreur locale est donnée par

$$e_{i+1} = \frac{h^{n+1}}{(n+1)!}\, u^{(n+1)}(\xi_{i+1})$$

où ξ_{i+1} est un point particulier de l'intervalle $[X_i, X_{i+1}]$. En règle générale, nous dirons d'une méthode à pas séparés qu'elle est d'ordre n si l'erreur locale de discrétisation est d'ordre $\mathcal{O}(h^{n+1})$. Dans ce cas, on démontre que l'erreur globale est d'ordre $\mathcal{O}(h^n)$.

Toutefois, toutes les méthodes de Taylor restent conditionnellement stables et ne sont donc pas adéquates pour des problèmes raides. En effet, pour le problème modèle $u' = \lambda u$, la formule de la méthode de Taylor d'ordre n devient

$$U_{i+1} = \left(1 + h\lambda + \frac{h^2\lambda^2}{2!} + \cdots + \frac{h^n\lambda^n}{n!}\right) U_i$$

On en déduit immédiatement que les régions de stabilité de la Figure (4.11) sont définies par

$$\left|1 + h\lambda + \frac{h^2\lambda^2}{2!} + \cdots + \frac{h^n\lambda^n}{n!}\right| < 1$$

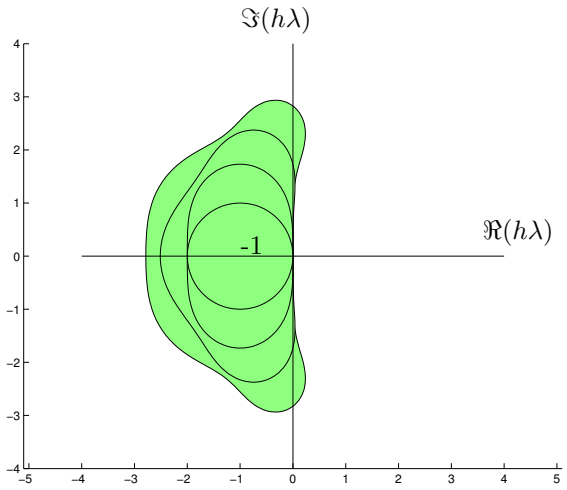


Figure 4.11: Région de stabilité des méthodes de Taylor d'ordre n.

4.3 La méthode d'Euler implicite ou la douce illusion de l'inconditionnellement stable

Est-il possible de modifier la méthode d'Euler que nous venons d'analyser afin d'améliorer ses propriétés de stabilité ? C'est l'objet de cette section, où nous allons introduire la méthode d'Euler implicite.

Tout d'abord, reprenons la méthode d'Euler explicite appliquée à l'équation $u' = f(x,u)$. On peut interpréter cette méthode comme la résolution numérique très grossière d'une intégrale.

$$\int_{X_i}^{X_{i+1}} f(x,u(x))dx \quad = \quad u(X_{i+1}) - u(X_i)$$

$$\Big\downarrow \quad \text{En approximant l'intégrale par un rectangle,}$$

$$h\ f(X_i,U_i) \quad \approx \quad U_{i+1} - U_i$$

On peut bien sûr utiliser d'autres techniques numériques pour calculer cette intégrale. En particulier, on pourrait remplacer $h\ f(X_i,U_i)$ par $h\ f(X_{i+1},U_{i+1})$: Nous obtenons ainsi la *méthode d'Euler implicite*. Cette méthode est dite *implicite* du fait que la grandeur U_{i+1} à calculer apparaît dans le terme en f. Il ne s'agit plus d'une méthode de Taylor. En effet, les méthodes de Taylor sont toujours *explicites* puisque U_{i+1} n'apparaît jamais dans le membre de droite.

En général, l'utilisation d'une méthode implicite complique grandement les calculs. Considérons par exemple le problème différentiel $u' = \sin u$ avec $u(0) = 1$. La méthode d'Euler implicite nécessite de trouver à chaque pas de temps, la solution U_{i+1} de l'équation

$$U_{i+1} = U_i + h \sin U_{i+1}$$

alors que la méthode d'Euler explicite ne nécessite pas un tel travail. Pourquoi donc utiliser de telles techniques ? La réponse est la suivante : les méthodes implicites sont en général beaucoup plus stables que les méthodes explicites.

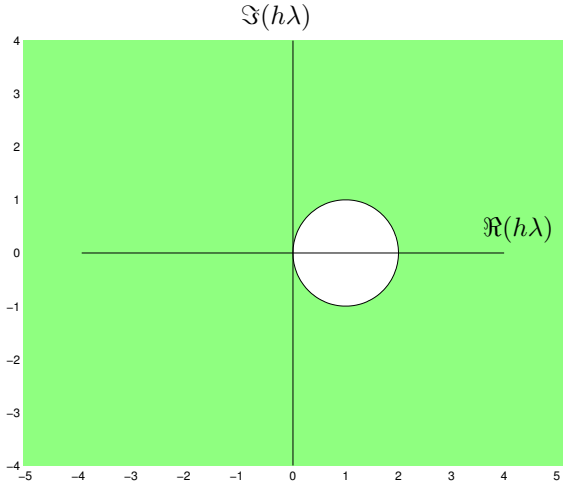


Figure 4.12: Région de stabilité de la méthode d'Euler implicite.

En effectuant une analyse similaire à celle décrite pour la méthode d'Euler explicite, il est possible de montrer que l'inégalité suivante sur le facteur d'amplification de l'erreur propagée doit être satisfaite pour observer un comportement stable.

$$\begin{array}{rcl} \left|\dfrac{1}{1-h\lambda}\right| & < & 1 \\ & \downarrow & \\ 1 & < & |1-h\lambda| \end{array}$$

Le facteur d'amplification de l'erreur globale est de norme strictement inférieure à l'unité pour $h > 0$ lorsque le problème différentiel est stable ($J < 0$). En d'autres termes, le pas de discrétisation h n'est pas limité par une condition de stabilité, mais est régi uniquement par des considérations de précision. La méthode d'Euler implicite est dite *inconditionnellement stable*. De manière surprenante, le critère de stabilité est également satisfait pour des problèmes différentiels instables... Mais, cela n'est guère utile,

car la stabilité apparente des solutions numériques ne correspond en rien aux solutions analytiques !

Une variante particulièrement utile des méthodes d'Euler est la *méthode de Crank-Nicolson* ou *méthode des trapèzes* définie comme la moyenne des méthodes d'Euler explicite et implicite.

$$U_{i+1} = U_i + \frac{h}{2}\Big(f(X_i, U_i) + f(X_{i+1}, U_{i+1}\ )\Big)$$

L'analyse de stabilité de la méthode de Crank-Nicolson s'écrit sous la forme :

$$\begin{array}{rcl} \left|\dfrac{1+h\lambda/2}{1-h\lambda/2}\right| & < & 1 \\ & \downarrow & \\ \Re(h\lambda) & < & 0 \end{array}$$

La région de stabilité de la méthode de Crank-Nicolson est toute la partie du plan complexe située à gauche de l'axe imaginaire pour le problème modèle. C'est donc une méthode inconditionnellement stable qui a, en outre, un ordre de précision quadratique.

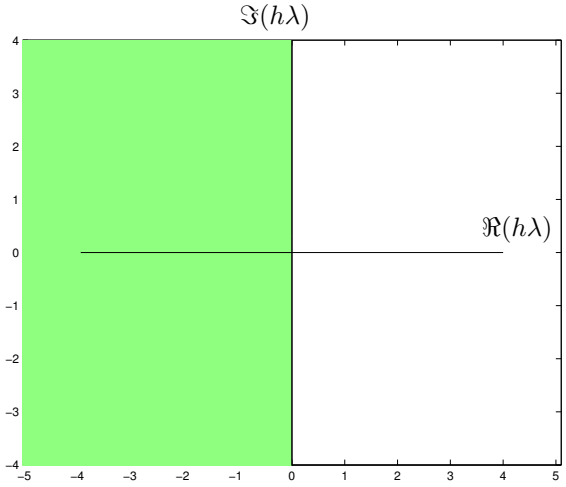


Figure 4.13: Région de stabilité de la méthode de Crank-Nicolson.

Toutefois, les propriétés remarquables de stabilité des méthodes d'Euler implicite et de Crank-Nicolson sont limitées en pratique par la nécessité de résoudre un problème non-linéaire (si f est non-linéaire par rapport à u) à chaque pas de temps : comme nous le verrons à l'avant-dernier chapitre, cette tâche est loin d'être facile à effectuer ! Evidemment, si la fonction f est linéaire par rapport à u, il serait vraiment dommage de ne pas tirer profit de cette méthode, par exemple, pour des problèmes linéaires raides.

4.4 Méthodes de Runge-Kutta

Les méthodes de Taylor permettent la construction des méthodes d'ordre n arbitrairement élevé, s'il est possible d'obtenir une expression analytique des dérivées partielles de f. La classe des *méthodes de Runge-Kutta* possède les mêmes propriétés de précision que les méthodes de Taylor, sans pour autant présenter la lourdeur de mise en oeuvre de ces dernières. Toutefois, comme les techniques de Taylor, les méthodes de Runge-Kutta discutées dans cette section sont explicites : elles ne sont donc pas appropriées à la résolution de problèmes raides.

Les méthodes de Runge-Kutta d'ordre n s'écrivent sous la forme générale :

$$\begin{aligned} U_{i+1} &= U_i + h(\sum_{j=1}^{n} w_j K_j) \\ K_1 &= f(X_i, U_i) \\ K_2 &= f(X_i + \alpha_2 h, U_i + \beta_{21} h K_1) \\ &\dots \\ K_n &= f(X_i + \alpha_n h, U_i + \sum_{m=1}^{n-1} \beta_{nm} h K_m) \end{aligned} \tag{4.14}$$

où les coefficients α_j, β_{jm} et les poids w_j sont choisis de manière à obtenir la précision requise. Plusieurs choix sont possibles, mais certains sont particulièrement populaires.

4.4.1 Méthode de Heun : Runge-Kutta d'ordre deux

En écrivant les relations (4.14) pour le cas particulier du second ordre, on obtient immédiatement :

$$\begin{aligned} U_{i+1} &= U_i + h(w_1 K_1 + w_2 K_2) \\ K_1 &= f(X_i, U_i) \\ K_2 &= f(X_i + \alpha h, U_i + \beta h K_1) \end{aligned} \tag{4.15}$$

Il s'agit de déterminer les constantes $(\alpha, \beta, w_1, w_2)$ qui définissent une méthode de Runge-Kutta particulière. On va réaliser cela afin que l'erreur locale de discrétisation

soit la plus petite possible. Notons que les valeurs $w_1 = 1$, $w_2 = 0$, α et β quelconques, conduisent à la méthode d'Euler explicite qui peut donc être également interprétée comme une méthode de Runge-Kutta d'ordre un.

En utilisant (4.15), nous pouvons écrire :

$$
\begin{aligned}
U_{i+1} &= U_i + h(w_1 K_1 + w_2 K_2) \\
U_{i+1} &= U_i + hw_1 f(X_i, U_i) + hw_2 f(X_i + \alpha h, U_i + \beta h K_1) \\
&\quad \Big\downarrow \quad \text{En effectuant un développement en série de Taylor de l'expression } f(X_i + \alpha h, U_i + \beta h K_1), \text{ autour de } (X_i, U_i), \\
U_{i+1} &= U_i + hw_1\ f(X_i, U_i) \\
&\quad \Big\downarrow \quad + hw_2 \left(f(X_i, U_i) + \alpha h\ \frac{\partial f}{\partial x}(X_i, U_i) + \beta h\ \frac{\partial f}{\partial u}(X_i, U_i)\ f(X_i, U_i) + \mathcal{O}(h^2) \right) \\
U_{i+1} &= U_i + h(w_1 + w_2)\ f(X_i, U_i) \\
&\quad + h^2\ w_2 \left(\alpha \frac{\partial f}{\partial x}(X_i, U_i) + \beta \frac{\partial f}{\partial u}(X_i, U_i)\ f(X_i, U_i) \right) + \mathcal{O}(h^3)
\end{aligned}
$$

Maintenant, écrivons la valeur que l'on obtiendrait avec une méthode de Taylor d'ordre deux. En d'autres mots, calculons U_{i+1} en effectuant un développement de Taylor autour de (X_i, U_i) en utilisant l'équation différentielle comme nous l'avions fait dans la section précédente :

$$
\begin{aligned}
U_{i+1} &= U_i + h\ f(X_i, U_i) \\
&\quad + \frac{h^2}{2} \left(\frac{\partial f}{\partial x}(X_i, U_i) + \frac{\partial f}{\partial u}(X_i, U_i)\ f(X_i, U_i) \right) + \mathcal{O}(h^3)
\end{aligned}
$$

Nous désirons trouver les coefficients $(\alpha, \beta, w_1, w_2)$ de sorte que l'erreur locale de discrétisation de la méthode soit de l'ordre $\mathcal{O}(h^3)$ et que la méthode de Runge-Kutta ainsi construite reproduise exactement la précision de la méthode de Taylor d'ordre deux. Cet objectif sera atteint en égalant les termes de même puissance de h dans les deux expressions. On obtient ainsi les trois relations :

$$\begin{cases} w_1 + w_2 &= 1 \\ \alpha w_2 &= 1/2 \\ \beta w_2 &= 1/2 \end{cases}$$

Ces trois relations ne suffisent pas pour déterminer de manière unique les quatre coefficients $(\alpha, \beta, w_1, w_2)$. Le choix particulier $w_1 = w_2 = 1/2$ et $\alpha = \beta = 1$ conduit à la *méthode de Heun* :

$$\boxed{\begin{aligned} U_{i+1} &= U_i + \tfrac{h}{2}(K_1 + K_2) \\ K_1 &= f(X_i, U_i) \\ K_2 &= f(X_i + h, \underbrace{U_i + hK_1}_{P_{i+1}}) \end{aligned}} \tag{4.16}$$

Interprétation géométrique de la méthode de Heun comme une méthode prédicteur-correcteur

L'interprétation géométrique de la méthode de Heun est illustrée à la Figure 4.14. On peut observer comment cette méthode peut être interprétée en termes de dérivation et d'intégration numériques. Il s'agit d'un calcul en deux étapes :

- Dans un premier temps, nous utilisons la ligne tangente en (X_1, U_1) pour obtenir une première estimation (X_2, P_2) du point d'arrivée. On utilise, ici, la méthode d'Euler explicite, relativement imprécise, pour obtenir une première prédiction du point d'arrivée. Cette méthode d'Euler explicite peut être interprétée comme une différence finie décentrée d'ordre un pour le calcul de $u'(X_1)$ en écrivant :

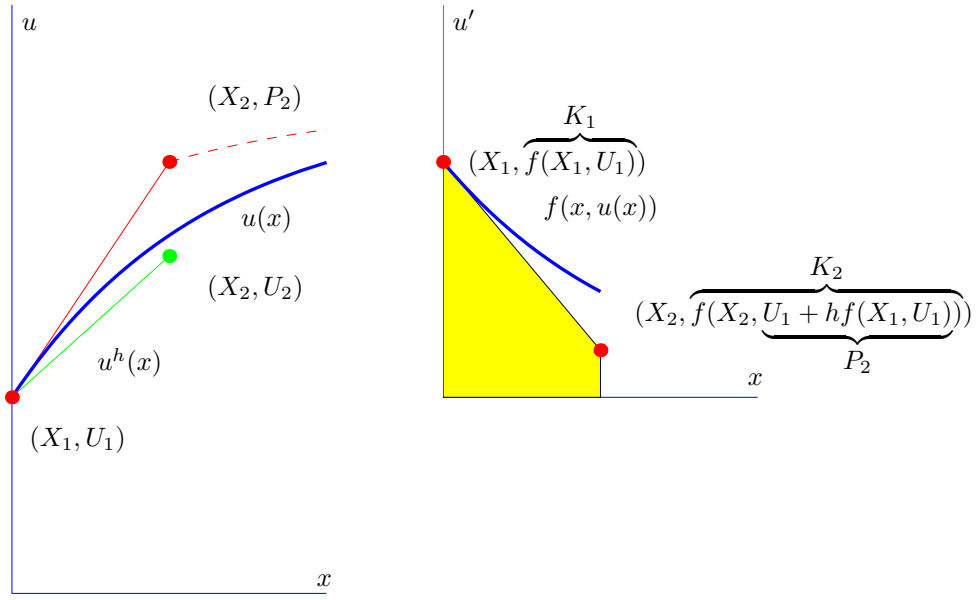


Figure 4.14: Interprétation géométrique de la méthode de Heun : utilisation d'une différence décentrée (c'est-à-dire de la méthode d'Euler explicite) pour obtenir une première prédiction P_2 de $u(X_2)$ et ensuite de la méthode des trapèzes pour une meilleure estimation (ou correction) U_2 de cette valeur $u(X_2)$.

$$f(X_1, U_1) = u'(X_1)$$

$$\downarrow$$

En utilisant une différence décentrée pour estimer $u'(X_1)$,

$$f(X_1, U_1) \approx \frac{u(X_2) - U_1}{h}$$

$$\downarrow$$

En approximant $u(X_2)$ par P_2,

$$f(X_1, U_1) = \frac{P_2 - U_1}{h}$$

$$\downarrow$$

$$U_1 + h \underbrace{f(X_1, U_1)}_{K_1} = P_2$$

L'utilisation d'une différence décentrée est moins précise, mais est requise pour obtenir une méthode explicite.

- Dans un second temps, nous allons utiliser une méthode de dérivation ou d'intégration centrée (et donc plus précise) pour construire une meilleure approximation du point d'arrivée. De telles méthodes requièrent malheureusement la connaissance de ce point : nous utiliserons alors la meilleure prédiction disponible à cet instant : c'est-à-dire (X_2, P_2).

  La seconde étape de la méthode de Heun peut être ainsi interprétée comme l'application de la méthode des trapèzes pour le calcul d'intégrale suivant :

$$\begin{aligned}
\int_{X_1}^{X_2} f(x,u(x))dx &= u(X_2) - u(X_1) \\
&\downarrow \quad \text{En utilisant la méthode des trapèzes,} \\
\frac{h}{2}(f(X_1,u(X_1)) + f(X_2,u(X_2))) &\approx u(X_2) - u(X_1) \\
&\downarrow \quad \text{En approximant } u(X_2) \text{ par } P_2 \text{ pour l'intégrale,} \\
&\quad \text{et par } U_2 \text{ à droite,} \\
\frac{h}{2}(\underbrace{f(X_1,U_1)}_{K_1} + \underbrace{f(X_2,P_2)}_{K_2}) &= U_2 - U_1 \\
&\downarrow \\
U_1 + \frac{h}{2}(K_1 + K_2) &= U_2
\end{aligned}$$

  Il est aussi possible d'interpréter la seconde étape comme une utilisation d'une différence centrée pour le calcul approché de la dérivée u du point milieu $X_{3/2}$ et d'une moyenne de K_1 et K_2 pour l'estimation de $f(X_{3/2}, u(X_{3/2}))$.

Considérons à nouveau l'équation différentielle $u' = (x-u)/2$ à résoudre sur $[0,3]$ avec $u(0) = 1$. Utilisons la méthode de Heun, pour $h = 1$, $h/2$, $h/4$ et $h/8$ pour obtenir la table ci-dessous.

Méthode de Heun pour $u' = (x-u)/2$					
X_i	$u^h(X_i)$	$u^{h/2}(X_i)$	$u^{h/4}(X_i)$	$u^{h/8}(X_i)$	$u(X_i)$
0.0000	1.0000	1.0000	1.0000	1.0000	1.0000
0.1250				0.9434	0.9432
0.2500			0.8984	0.8977	0.8975
0.3750				0.8624	0.8621
0.5000		0.8438	0.8381	0.8368	0.8364
0.7500			0.8141	0.8124	0.8119
1.0000	0.8750	0.8311	0.8222	0.8202	0.8196
2.0000	1.1719	1.1176	1.1068	1.1044	1.1036
3.0000	1.7324	1.6821	1.6723	1.6701	1.6694

4.4.2 Méthode *classique* de Runge-Kutta d'ordre quatre

On génère de manière totalement similaire les méthodes de Runge-Kutta d'ordre supérieur. C'est un vrai travail de bénédictin purement calculatoire que nous ne reproduirons donc pas ici. La méthode classique et très populaire de Runge-Kutta d'ordre 4 est définie comme suit :

$$\begin{aligned} U_{i+1} &= U_i + \tfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= f(X_i, U_i) \\ K_2 &= f(X_i + \tfrac{h}{2}, U_i + \tfrac{h}{2}K_1) \\ K_3 &= f(X_i + \tfrac{h}{2}, U_i + \tfrac{h}{2}K_2) \\ K_4 &= f(X_i + h, U_i + hK_3) \end{aligned} \tag{4.17}$$

Son interprétation géométrique est donnée à la Figure 4.15. On peut ainsi observer que la méthode de Runge-Kutta peut être interprétée comme l'application de la règle d'intégration de Simpson en utilisant les approximations suivantes $f(X_{3/2}, u(X_{3/2})) \approx \frac{1}{2}(K_2 + K_3)$ et $f(X_2, u(X_2)) \approx K_4$

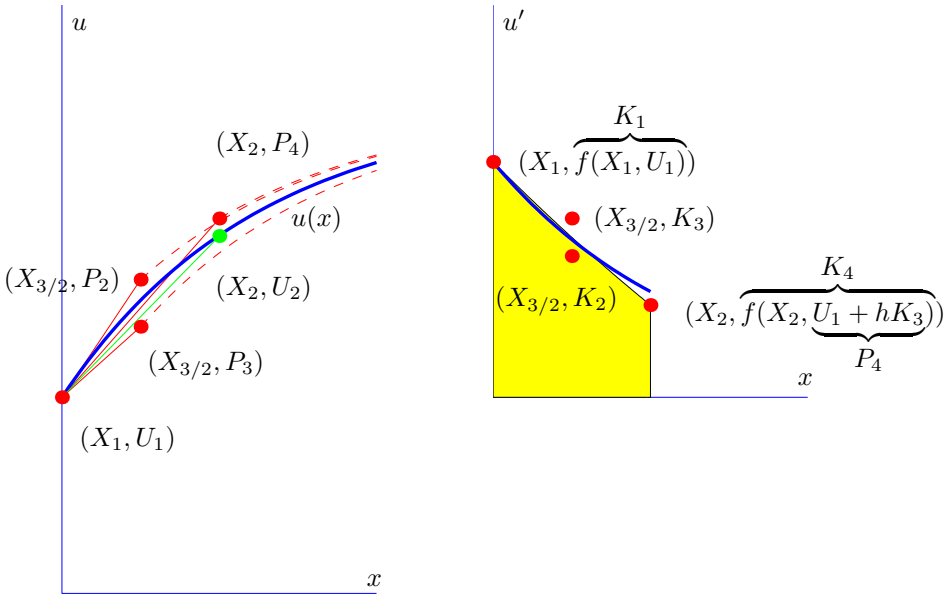


Figure 4.15: Interprétation géométrique de la méthode de Runge-Kutta. D'abord, on utilise une différence décentrée (c'est-à-dire la méthode d'Euler explicite) pour obtenir une première prédiction P_2 de $u(X_{3/2})$. Ensuite, on obtient avec les nouvelles informations une seconde prédiction P_3 pour le même point. Sur base de ces deux prédictions intermédiaires, on réalise une prédiction P_4 pour le point d'arrivée. Finalement, on utilise la méthode de Simpson pour la dernière estimation (ou correction) U_2 de cette valeur $u(X_2)$.

4.4.3 Méthode de Runge-Kutta-Fehlberg

Une manière de garantir la précision de la solution d'un problème de Cauchy est de résoudre le problème deux fois en utilisant respectivement h et $h/2$ comme pas de discrétisation et de comparer les valeurs obtenues aux abscisses d'approximation. Une telle approche requiert malheureusement un coût calcul important. Typiquement pour s'assurer que la méthode classique de Runge-Kutta d'ordre 4 est exacte, on doit effectuer le même calcul sur un maillage deux fois plus fin. Pour passer de X_i à X_{i+1}, il faudra donc effectuer 11 évaluations de f au lieu de 4 évaluations, si on n'effectue pas le calcul de contrôle.

Les *méthodes de Runge-Kutta-Fehlberg* combinent astucieusement deux méthodes de Runge-Kutta d'ordre n et $n+1$. L'erreur locale commise est estimée en cours de calcul à l'aide de l'écart entre les solutions fournies par les deux méthodes. Pour le cas $n = 4$, nous évaluerons seulement six fois la fonction f. C'est une procédure qui permet d'obtenir de manière combinée deux estimations distinctes de $u(X_{i+1})$, l'une d'ordre quatre et la seconde d'ordre cinq. A chaque pas, les deux solutions sont comparées. Si elles sont en bon accord, l'approximation est acceptée. Dans le cas contraire, le pas h peut être réduit. Dans le cas où l'accord est nettement meilleur que la précision requise,

on peut même décider d'agrandir le pas h. Nous venons ainsi de décrire une *méthode adaptative d'intégration* qui gère de manière automatique le pas h_i afin d'obtenir une précision prescrite par l'utilisateur.

La méthode de Runge-Kutta-Fehlberg requiert les calculs suivants à chaque pas :

$$
\begin{aligned}
P_{i+1} &= U_i + h(\tfrac{25}{216}K_1 + \tfrac{1408}{2565}K_3 + \tfrac{2197}{4104}K_4 - \tfrac{1}{5}K_5) \\
U_{i+1} &= U_i + h(\tfrac{16}{135}K_1 + \tfrac{6656}{12825}K_3 + \tfrac{28561}{56430}K_4 - \tfrac{9}{50}K_5 + \tfrac{2}{55}K_6) \\
K_1 &= f(X_i, U_i) \\
K_2 &= f(X_i + \tfrac{h}{4}, U_i + \tfrac{h}{4}K_1) \\
K_3 &= f(X_i + \tfrac{3h}{8}, U_i + \tfrac{3h}{32}K_1 + \tfrac{9h}{32}K_2) \\
K_4 &= f(X_i + \tfrac{12h}{13}, U_i + \tfrac{1932h}{2197}K_1 - \tfrac{7200h}{2197}K_2 + \tfrac{7296h}{2197}K_3) \\
K_5 &= f(X_i + h, U_i + \tfrac{439h}{216}K_1 - 8hK_2 + \tfrac{3680h}{513}K_3 - \tfrac{845h}{4104}K_4) \\
K_6 &= f(X_i + \tfrac{h}{2}, U_i - \tfrac{8h}{27}K_1 + 2hK_2 - \tfrac{3544h}{2565}K_3 + \tfrac{1859h}{4104}K_4 - \tfrac{11h}{40}K_5)
\end{aligned}
$$

L'ensemble de valeurs pour ces coefficients a été choisi tel que la méthode obtenue soit du cinquième ordre, et que les valeurs calculées K_j, combinées à *d'autres* coefficients donnent une méthode de Runge-Kutta d'ordre 4. (Cette méthode n'est pas la méthode classique de Runge-Kutta d'ordre 4). A chaque pas, on calcule ainsi deux approximations de $u(X_{i+1})$: U_{i+1} et P_{i+1}. Une estimation de l'erreur locale θ_{i+1} commise par la méthode d'ordre 4 est donnée par :

$$\theta_{i+1} \approx |U_{i+1} - P_{i+1}| \tag{4.18}$$

En pratique, le résultat de la méthode d'ordre 5, à savoir U_{i+1}, est retenu au pas suivant $X_{i+1} \to X_{i+2}$ pour évaluer les K_j. Cette méthode est donc une véritable méthode d'ordre 5, qui permettra de contrôler le pas h à l'aide de l'erreur locale de la méthode d'ordre 4 associée. D'autres combinaisons sont possibles. Le logiciel *Matlab* implémente une variante des techniques de Runge-Kutta-Fehlberg dans les fonctions `ode23` et `ode45`.

Stratégies adaptatives

Comment adapter de manière automatique le pas afin que l'erreur locale commise soit inférieure à une valeur ϵ ? La connaissance combinée de l'ordre d'une méthode et d'une

estimation de l'erreur locale commise à chaque pas va permettre de calculer la taille du pas de manière adaptative afin de maintenir l'erreur locale en-dessous de la borne imposée par l'utilisateur. Appelons θ_i une estimation de l'erreur locale commise lors du passage entre X_{i-1} et X_i. Nous désirons déterminer la taille du pas h_{i+1} afin que le calcul de U_{i+1} ne soit pas entaché d'une erreur locale supérieure à une tolérance ϵ fixée. En supposant que la méthode soit d'ordre n, nous pouvons espérer que pour des pas suffisamment petits,

$$\begin{aligned} \theta_i &= Ch_i^{n+1} \\ \theta_{i+1} &= Ch_{i+1}^{n+1} \end{aligned}$$

où nous utilisons (de manière un brin abusive !) la même constante pour les deux expressions. Cet abus est toutefois acceptable si les pas sont réellement de petite taille et ne servira qu'à guider une stratégie qui est elle-même de nature heuristique. On peut alors en déduire :

$$\theta_{i+1} \approx \theta_i \left(\frac{h_{i+1}}{h_i} \right)^{n+1}$$

Imposer la condition $\theta_{i+1} \leq \epsilon$ permet de prédire une borne pour le nouveau h_{i+1} :

$$h_{i+1} \leq h_i \left(\frac{\epsilon}{\theta_i} \right)^{\frac{1}{(n+1)}} . \tag{4.19}$$

En pratique, l'utilisateur impose des valeurs minimale et maximale h_{min} et h_{max} que peut prendre le pas. Pour une tolérance ϵ également fixée par l'utilisateur, *une* stratégie de calcul adaptatif de pas est alors :

1 Estimation de l'erreur commise à l'étape courante i avec le pas h_i en effectuant le calcul de θ_i. On utilise la différence entre la prédiction et la correction pour effectuer cette estimation.

$$\theta_i = \left| U_i - P_i \right|$$

2 Est-ce que le critère d'erreur est respecté ? Si $\theta_i > \epsilon$, il faut recommencer le pas courant i avec un plus petit pas de temps. On choisira comme nouveau pas $h = h_i/2$. Il convient aussi de vérifier que le nouveau pas de temps n'est pas inférieur à une limite minimale h_{min} fixée a priori. Si cela arrive, le programme s'arrête de manière définitive avec un constat d'échec !

3 Estimation d'un nouveau pas optimal $\widehat{h}$ à partir de l'estimation θ_i de l'erreur locale qui vient d'être commise au pas précédent.

$$\widehat{h} = h_i \left(\frac{\epsilon}{\theta_i} \right)^{\frac{1}{(n+1)}}$$

4 Si $\widehat{h} > h_i$, on choisit comme nouveau pas : $h_{i+1} = \min(\widehat{h}, \frac{3}{2}h_i, h_{max})$

5 Si $\widehat{h} < h_i$, on choisit comme nouveau pas : $h_{i+1} = \max(\widehat{h}, \frac{1}{2}h_i, h_{min})$

Cet exemple de stratégie est relativement prudent puisqu'on augmente le pas d'un facteur 1.5, alors que la diminution sera d'un facteur 2. On attribue une confiance modérée à l'estimation d'erreur, en évitant des variations trop brusques (dans les deux sens) du pas. Dans le cas d'une réduction de celui-ci, on préfère éventuellement accepter le risque d'un échec au pas suivant et d'une nouvelle diminution du pas de temps, que de le diminuer immédiatement d'un facteur supérieur à deux.

Il est essentiel de fixer une valeur minimale et une valeur maximale pour éviter un comportement dangereux d'une telle stratégie. En l'absence de h_{min}, une telle approche pourrait amener à une boucle infinie de diminution du pas de temps, face à une singularité du problème à intégrer ! La stratégie présentée est à la base de nombreux codes de calcul de grande taille d'éléments finis.

4.5 Méthodes d'Adams-Bashforth-Moulton

Les méthodes de Taylor et de Runge-Kutta sont des méthodes à pas simples, puisqu'elles n'utilisent que l'information du point qui précède pour calculer les valeurs successives.

Typiquement, seul le point initial (X_0, U_0) est utilisé pour obtenir (X_1, U_1) et de manière générale U_i pour l'obtention de U_{i+1}. Ces méthodes n'exploitent donc pas la connaissance des valeurs U_j $(j < i)$ antérieures à U_i calculées en cours d'intégration. L'idée des *méthodes à pas liés* est de tirer profit de l'historique de l'intégration, en utilisant certaines valeurs antérieures U_j. Certaines de ces méthodes ont la forme générique suivante :

$$U_{i+1} = U_i + h \sum_{j=-1}^{n} \beta_j \underbrace{f(X_{i-j}, U_{i-j})}_{F_{i-j}} \qquad (i \geq n). \tag{4.20}$$

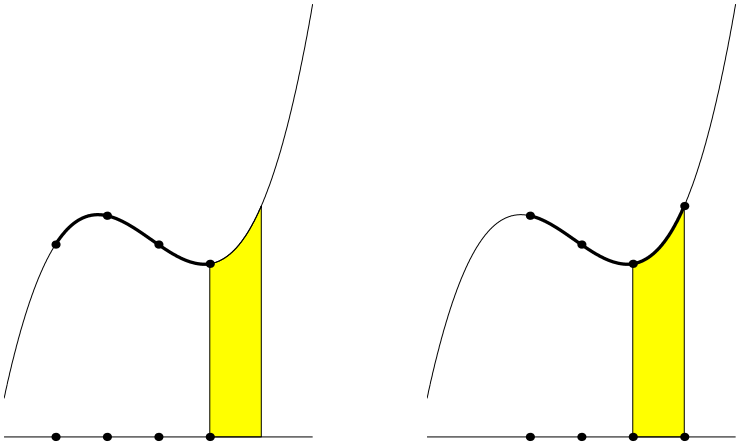
Les β_j sont des coefficients dont les valeurs définissent une méthode particulière. Pour calculer U_{i+1}, nous utilisons les $(n+1)$ valeurs U_{i-n}, U_{i-n+1}, $\ldots$, U_i calculées précédemment. Si $\beta_{-1} \neq 0$, l'inconnue U_{i+1} apparaît dans le membre de droite et on obtient des méthodes implicites. Si $\beta_{-1} = 0$, on obtient des méthodes explicites. De telles méthodes à pas liés ne peuvent être utilisées que lorsque les n premiers résultats U_1, U_2, $\ldots$, U_n sont disponibles. Une autre technique doit donc être utilisée pour démarrer l'intégration.

Pour calculer les coefficients des méthodes à pas liés de la forme (4.20), écrivons :

$$U_{i+1} = U_i + \int_{X_i}^{X_{i+1}} f(x, u(x))dx \tag{4.21}$$

L'idée de base est de calculer de manière approchée l'intégrale de f sur l'intervalle $[X_i, X_{i+1}]$ à l'aide des résultats U_{i-n}, U_{i-n+1}, $\ldots$, U_i obtenus aux pas précédents. Nous pouvons calculer le polynôme d'interpolation de degré n passant par les points (X_{i-n}, F_{i-n}), (X_{i-n+1}, F_{i-n+1}), $\ldots$, (X_i, F_i) et l'intégrer sur l'intervalle $[X_i, X_{i+1}]$. On obtient ainsi les *méthodes explicites d'Adams-Bashforth*. Notons qu'une méthode à pas liés explicite ne demande qu'*une* nouvelle évaluation de f à chaque pas. Par contre, les *méthodes implicites d'Adams-Moulton* sont obtenues en intégrant sur l'intervalle $[X_i, X_{i+1}]$ le polynôme d'interpolation de degré n passant par les points antérieurs (X_{i-n+1}, F_{i-n+1}), $\ldots$, (X_i, F_i) et par le nouveau point, encore inconnu, (X_{i+1}, F_{i+1}). Quelques exemples de méthodes d'Adams-Bashforth et d'Adams-Moulton sont repris ci-dessous. Nous y reconnaissons les méthodes d'Euler explicite et implicite. Les estimations d'erreur locale sont une conséquence directe des résultats liés à l'interpolation polynomiale et à l'intégration numérique.

$$
\begin{array}{|lcll|}
\hline
U_{i+1} & = & U_i + h\ F_i & \\
|e_i| & \leq & \frac{C_2}{2}\ h^2 \qquad \text{(Euler explicite - Adams-Bashforth d'ordre 1)} & \mathcal{O}(h) \\
U_{i+1} & = & U_i + h\ F_{i+1} & \\
|e_i| & \leq & \frac{C_2}{2}\ h^2 \qquad \text{(Euler implicite - Adams-Moulton d'ordre 1)} & \mathcal{O}(h) \\
\hline
U_{i+1} & = & U_i + \dfrac{h}{2}\ \left(-F_{i-1} + 3F_i \right) & \\
|e_i| & \leq & \frac{5C_3}{12}\ h^3 \qquad \text{(Adams-Bashforth d'ordre 2)} & \mathcal{O}(h^2) \\
U_{i+1} & = & U_i + \dfrac{h}{2}\ \left(F_i + F_{i+1} \right) & \\
|e_i| & \leq & \frac{C_3}{12}\ h^3 \qquad \text{(trapèzes - Adams-Moulton d'ordre 2)} & \mathcal{O}(h^2) \\
\hline
U_{i+1} & = & U_i + \dfrac{h}{12}\ \left(5F_{i-2} - 16F_{i-1} + 23F_i \right) & \\
|e_i| & \leq & \frac{3C_4}{8}\ h^4 \qquad \text{(Adams-Bashforth d'ordre 3)} & \mathcal{O}(h^3) \\
U_{i+1} & = & U_i + \dfrac{h}{12}\ \left(-F_{i-1} + 8F_i + 5F_{i+1} \right) & \\
|e_i| & \leq & \frac{C_4}{24}\ h^4 \qquad \text{(Adams-Moulton d'ordre 3)} & \mathcal{O}(h^3) \\
\hline
U_{i+1} & = & U_i + \dfrac{h}{24}\ \left(-9F_{i-3} + 37F_{i-2} - 59F_{i-1} + 55F_i \right) & \\
|e_i| & \leq & \frac{251C_5}{720}\ h^5 \qquad \text{(Adams-Bashforth d'ordre 4)} & \mathcal{O}(h^4) \\
U_{i+1} & = & U_i + \dfrac{h}{24}\ \left(F_{i-2} - 5F_{i-1} + 19F_i + 9F_{i+1} \right) & \\
|e_i| & \leq & \frac{19C_5}{720}\ h^5 \qquad \text{(Adams-Moulton d'ordre 4)} & \mathcal{O}(h^4) \\
\hline
\end{array}
$$



Adams-Bashforth $\mathcal{O}(h^4)$ Adams-Moulton $\mathcal{O}(h^4)$

Figure 4.16: Interprétation géométrique des méthodes d'Adams-Bashforth et d'Adams-Moulton. Dans le cas d'Adams-Bashforth, l'intégration numérique est effectuée sur des valeurs extrapolées et la méthode est explicite. Dans le cas d'Adams-Moulton, on utilise des valeurs interpolées, mais la méthode devient alors implicite.

Les méthodes d'Adams-Bashforth-Moulton sont des méthodes prédicteur-correcteur à pas liés

Les méthodes à pas liés *explicites* fournissent directement le résultat U_{i+1} au prix d'une seule nouvelle évaluation de f, à savoir F_i. Bien entendu, une phase de démarrage est nécessaire pour évaluer les premiers résultats $U_1, U_2, \ldots U_n$. Les méthodes à pas liés *implicites*, par contre, conduisent en général à une équation non linéaire en U_{i+1}.

En effectuant une analyse de stabilité comme nous l'avons fait précédemment, on peut déterminer les régions de stabilité des méthodes explicites d'Adams-Bashforth et des méthodes implicites d'Adams-Moulton. On observerait alors que pour un même ordre, les méthodes implicites sont plus stables que les méthodes explicites. On observerait aussi que la zone de stabilité diminue dans les deux cas, lorsqu'on augmente l'ordre de la méthode. Seules, les méthodes d'Euler implicite et celle des trapèzes sont inconditionnellement stables. Il est malheureusement impossible de construire une méthode inconditionnellement stable d'ordre strictement supérieur à deux.

L'idée vient à l'esprit d'utiliser le résultat fourni par une méthode explicite. Ceci conduit aux méthodes *prédicteur-correcteur*, dans lesquelles une méthode explicite est utilisée pour obtenir la *valeur prédite* P_{i+1}, qui sera ensuite *corrigée* à l'aide de la méthode implicite. La méthode prédicteur-correcteur la plus élémentaire est :

$$\boxed{\begin{array}{ll} \textit{(Euler explicite)} & P_{i+1} = U_i + hf(X_i, U_i) \\ \textit{(Euler implicite)} & U_{i+1} = U_i + hf(X_{i+1}, P_{i+1}) \end{array}}$$

Le correcteur utilise $f(X_{i+1}, P_{i+1})$ comme une approximation de $f(X_{i+1}, U_{i+1})$ dans le calcul de U_{i+1}. Comme a priori cette valeur U_{i+1} est elle-même une meilleure estimation de $u(X_{i+1})$ que P_{i+1}, elle pourrait être utilisée dans le correcteur, pour générer une nouvelle estimation de F_{i+1}, qui, elle-même, permettrait d'améliorer une nouvelle valeur corrigée pour U_{i+1}. Si on poursuit ce processus itératif sur le correcteur et qu'il converge, on obtient la solution exacte de l'équation de la méthode implicite de correction. Toutefois, une telle stratégie n'est pas efficace, il est beaucoup plus intéressant de réduire le pas et de n'effectuer qu'une seule correction sur chaque pas.

Parmi les nombreuses méthodes prédicteur-correcteur développées dans la littérature, les méthodes d'*Adams-Bashforth-Moulton* sont très populaires. Comme leur nom l'indique, elles consistent à choisir une méthode d'Adams-Bashforth comme prédicteur et une méthode d'Adams-Moulton comme correcteur.

Estimation de l'erreur

A titre d'exemple, considérons la méthode d'Adams-Bashforth-Moulton d'ordre quatre :

$$\boxed{\begin{array}{l} P_{i+1} = U_i + \dfrac{h}{24}\left(-9f(X_{i-3}, U_{i-3}) + 37f(X_{i-2}, U_{i-2}) - 59f(X_{i-1}, U_{i-1}) + 55f(X_i, U_i)\right) \\ U_{i+1} = U_i + \dfrac{h}{24}\left(f(X_{i-2}, U_{i-2}) - 5f(X_{i-1}, U_{i-1}) + 19f(X_i, U_i) + 9f(X_{i+1}, P_{i+1})\right) \end{array}}$$

Les termes d'erreur des formules d'intégration numérique utilisées pour obtenir le prédicteur et le correcteur sont tous deux d'ordre $\mathcal{O}(h^5)$. En supposant que U_i soit exact, les erreurs (globales et locales, donc) pour les deux formules peuvent être écrites sous la forme :

$$\begin{aligned} u(X_{i+1}) - P_{i+1} &= \frac{251h^5}{720}u^{(5)}(\xi_{i+1}), \\ u(X_{i+1}) - U_{i+1} &= \frac{-19h^5}{720}u^{(5)}(\zeta_{i+1}). \end{aligned}$$

En supposant que h est très petit, on peut alors estimer que la valeur U_{i+1} obtenue par le correcteur est très proche de celle que l'on obtiendrait en résolvant l'équation non linéaire.

On peut aussi estimer que la dérivée cinquième est presque constante sur l'intervalle et donc que $u^{(5)}(\xi_{i+1}) \approx u^{(5)}(\zeta_{i+1})$. On peut alors éliminer les termes contenant la dérivée cinquième et écrire que[2] :

$$(u(X_{i+1}) - U_{i+1})^2 \approx \underbrace{\left(\frac{-19}{270}\left(U_{i+1} - P_{i+1}\right)\right)^2}_{\theta_{i+1}^2} \tag{4.22}$$

L'importance des méthodes de type prédicteur-correcteur est maintenant claire. L'équation (4.22) donne θ_{i+1} qui est une estimation de la valeur absolue de l'erreur locale commise sur l'intervalle $[X_i, X_{i+1}]$ en se basant sur les deux approximations calculées de $u(X_{i+1})$: la prédiction et la correction. Il est alors possible de construire une stratégie adaptative comme nous l'avons fait dans la section précédente.

4.6 Méthodes de Gear

Les *méthodes de Gear* (davantage connues dans la littérature sous le nom de *méthodes BDF* pour *Backward Differential Formulae*) ne sont plus construites à partir des techniques de dérivation et d'intégration numérique, mais directement à partir de la construction d'un polynôme d'interpolation : elles sont également une très brève introduction au concept plus général des méthodes de résidus pondérés utilisées pour les équations aux dérivées partielles.

Le principe est le suivant. Construisons le polynôme d'interpolation $u_{i+1}^h(x)$ de degré n passant par les points (X_{i-n+1}, U_{i-n+1}), ..., (X_{i+1}, U_{i+1}) :

$$u_{i+1}^h(x) = \sum_{j=0}^{n} U_{i+1-j}\ \phi_j(x) \tag{4.23}$$

où $\phi_j(x)$ sont les polynômes de Lagrange associés aux points d'interpolation X_{i+1-j}. On obtient ainsi une formule pour estimer U_{i+1} en supposant que toutes les ordonnées précédentes $U_{i-n+1} \ldots U_i$ sont connues. L'idée est d'approcher $u(x)$ par $u^h(x)$ dans l'équation différentielle et d'écrire que :

[2] La fraction de la formule est bien correcte ! Si, si ! Et pourtant, chaque année, un nombre non négligeable d'étudiants estiment -à tort- que j'aurais dû écrire 19/251... Pourquoi ont-ils tort ?

$$
\begin{aligned}
u'(X_{i+1}) &= \underbrace{f(X_{i+1}, \overbrace{u(X_{i+1})}^{U_{i+1}})}_{F_{i+1}} \\
&\downarrow \\
(u^h_{i+1})'(X_{i+1}) &\approx F_{i+1} \\
&\downarrow \\
\sum_{j=0}^{n} U_{i+1-j}\ \phi'_j(X_{i+1}) &\approx F_{i+1} \\
&\downarrow \\
U_{i+1} &\approx \frac{1}{\phi'_0(X_{i+1})} \left(-\sum_{j=1}^{n} U_{i+1-j}\ \phi'_j(X_{i+1}) + F_{i+1} \right)
\end{aligned}
$$

où les valeurs des polynômes de Lagrange peuvent évidemment être calculées et tabulées a priori. Observons que nous avons arbitrairement décidé d'imposer le respect de l'équation différentielle au point X_{i+1} et que nous avons généré, de cette manière, une famille de méthodes implicites. On pourrait aussi choisir d'écrire une relation semblable au point X_i et on obtiendrait ainsi une famille de méthodes explicites. On pourrait aussi écrire une combinaison linéaire entre ces deux relations, ou bien imposer un respect seulement approximatif de la relation différentielle sur l'intervalle $[X_i, X_{i+1}]$ et bâtir ainsi ce qu'on appelle les méthodes de résidus pondérés.

A titre d'exemple, construisons la méthode de Gear d'ordre 2. Nous avons, pour $n = 2$ et un pas h constant :

$$
\begin{aligned}
u^h_{i+1}(x) &= U_{i+1}\ \phi_0(x) + U_i\ \phi_1(x) + U_{i-1}\ \phi_2(x) \\
&= U_{i+1}\ \frac{(x-X_i)(x-X_{i-1})}{2h^2} \\
&\quad \Big\downarrow \qquad + U_i\ \frac{(x-X_{i+1})(x-X_{i-1})}{-h^2} + \ U_{i-1}\ \frac{(x-X_{i+1})(x-X_i)}{2h^2} \\
(u^h_{i+1})'(x) &= U_{i+1}\ \frac{(2x-X_i-X_{i-1})}{2h^2} \\
&\quad \Big\downarrow \qquad + U_i\ \frac{(2x-X_{i+1}-X_{i-1})}{-h^2} + \ U_{i-1}\ \frac{(2x-X_{i+1}-X_i)}{2h^2} \\
(u^h_{i+1})'(X_{i+1}) &= U_{i+1}\ \left(\frac{3h}{2h^2}\right) + \ U_i\ \left(\frac{-2h}{h^2}\right) + \ U_{i-1}\ \left(\frac{h}{2h^2}\right) \\
&= \frac{1}{h}\ \left(\frac{3}{2}U_{i+1} - 2U_i + \frac{1}{2}U_{i-1}\right)
\end{aligned}
$$

La méthode de Gear d'ordre 2 s'écrit donc :

$$
U_{i+1} = \frac{4}{3}U_i - \frac{1}{3}U_{i-1} + \frac{2h}{3}F_{i+1}. \tag{4.24}
$$

Les méthodes de Gear les plus utiles en pratique sont données dans le tableau ci-dessous :

$$U_{i+1} = U_i + h\ F_{i+1}$$

(Euler implicite - Gear d'ordre 1) $\mathcal{O}(h)$

$$U_{i+1} = \frac{1}{3}\left(-U_{i-1} + 4U_i\right) + \frac{2h}{3}\ F_{i+1}$$

(Gear d'ordre 2) $\mathcal{O}(h^2)$

$$U_{i+1} = \frac{1}{11}\left(2U_{i-2} - 9U_{i-1} + 18U_i\right) + \frac{6h}{11}\ F_{i+1}$$

(Gear d'ordre 3) $\mathcal{O}(h^3)$

$$U_{i+1} = \frac{1}{25}\left(-3U_{i-3} + 16U_{i-2} - 36U_{i-1} + 48U_i\right) + \frac{12h}{25}\ F_{i+1}$$

(Gear d'ordre 4) $\mathcal{O}(h^4)$

Notons que la méthode de Gear d'ordre 1 est équivalente à la méthode d'Euler implicite. La méthode de Gear d'ordre 2 est également inconditionnellement stable, mais elle n'est *pas* équivalente à la règle du trapèze. On peut montrer que la région de stabilité de ces méthodes *contient l'entièreté de l'axe réel négatif* pour un ordre $n \leq 6$. Ces méthodes semblent donc convenir particulièrement bien à la résolution de problèmes scalaires raides.

4.7 Systèmes d'équations différentielles

Résoudre un *système* d'équations différentielles du premier ordre revient à trouver n fonctions $u_i(x)$ qui vérifient sur $[a, b]$ le système

$$\left\{\begin{array}{rcl} u_1'(x) &=& f_1(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ u_2'(x) &=& f_2(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ \vdots & & \\ u_n'(x) &=& f_n(x, u_1(x), u_2(x), \ldots, u_n(x)) \end{array}\right.$$

avec les valeurs initiales

$$\left\{\begin{array}{rcl} u_1(a) & = & \overline{u}_1 \\ u_2(a) & = & \overline{u}_2 \\ \vdots & & \\ u_n(a) & = & \overline{u}_n \end{array}\right. .$$

Si nous posons

$$\mathbf{u}(x) = \begin{bmatrix} u_1(x) \\ u_2(x) \\ \vdots \\ u_n(x) \end{bmatrix}, \qquad \overline{\mathbf{u}} = \begin{bmatrix} \overline{u}_1 \\ \overline{u}_2 \\ \vdots \\ \overline{u}_n \end{bmatrix},$$

$$\mathbf{f}(x, \mathbf{u}(x)) = \begin{bmatrix} f_1(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ f_2(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ \vdots \\ f_n(x, u_1(x), u_2(x), \ldots, u_n(x)) \end{bmatrix},$$

notre problème différentiel aux valeurs initiales s'écrit sous la forme compacte suivante :

Trouver $\mathbf{u}(x)$ tel que

$$\left\{\begin{array}{rcll} \mathbf{u}'(x) & = & \mathbf{f}(x, \mathbf{u}(x)), & x \in [a, b] \\ \mathbf{u}(a) & = & \overline{\mathbf{u}} & \end{array}\right. \tag{4.25}$$

Rappelons également qu'une équation différentielle scalaire d'ordre n peut toujours être transformée en un système de n équations différentielles du premier ordre. En effet, soit le problème différentiel d'ordre n suivant :

$$u^{(n)}(x) = f(x, u(x), u'(x), u''(x), \ldots, u^{(n-1)}(x)), \tag{4.26}$$

avec les conditions initiales

$$\left\{\begin{array}{rcl} u(a) & = & \overline{u} \\ u'(a) & = & \overline{u}^{(1)} \\ \vdots & & \\ u^{(n-1)}(a) & = & \overline{u}^{(n-1)} \end{array}\right.$$

Définissons les n fonctions inconnues suivantes :

$$\left\{\begin{array}{lcl} u_1(x) & = & u(x) \\ u_2(x) & = & u'(x) \\ \vdots & & \\ u_n(x) & = & u^{(n-1)}(x) \end{array}\right.$$

Le problème différentiel (4.26) d'ordre n est alors équivalent au système de n équations différentielles du premier ordre

$$\left\{\begin{array}{lcl} u_1'(x) & = & u_2(x) \\ u_2'(x) & = & u_3(x) \\ \vdots & & \\ u_{n-1}'(x) & = & u_n(x) \\ u_n'(x) & = & f(x, u_1(x), u_2(x), \ldots, u_n(x)) \end{array}\right.$$

avec les conditions initiales

$$\left\{\begin{array}{lcl} u_1(a) & = & \overline{u} \\ u_2(a) & = & \overline{u}^{(1)} \\ \vdots & & \\ u_n(a) & = & \overline{u}^{(n-1)} \end{array}\right.$$

A titre d'exemple, considérons l'exemple désormais familier de l'équation différentielle du second ordre

$$m\ y''(t) + c\ y'(t) + k\ y(t) = f(t) \tag{4.27}$$

à résoudre sur l'intervalle $[a, b]$ avec les conditions initiales $y(a) = 0$ et $y'(a) = 0$. Ce modèle mathématique décrit un système mécanique masse-amortisseur-ressort, où m est la masse, c est une constante d'amortissement, k est la raideur du ressort et f est la force externe appliquée. L'inconnue $y(t)$ est le déplacement de la masse (mesuré à partir de la position d'équilibre) et $y'(t)$ est sa vitesse. La variable t désigne le temps.

Le problème (4.27) est aisément transformé en un système du premier ordre à deux inconnues de la forme (4.25). En posant $y_1(t) = y(t)$ et $y_2(t) = y'(t) = y_1'(t)$, l'équation du second ordre (4.27) devient

$$\left\{\begin{array}{l} y_1'(t) = y_2(t) \\ \\ y_2'(t) = \frac{1}{m}[f(t) - c\ y_2(t) - k\ y_1(t)] \end{array}\right. \tag{4.28}$$

avec $y_1(a) = 0$ et $y_2(a) = 0$.

Les méthodes numériques décrites plus haut pour résoudre une équation différentielle scalaire $u' = f(x,u)$ sont directement transposables aux systèmes d'équations différentielles. L'objectif est maintenant, pour chacune des fonctions inconnues $u_j(x)$, de trouver une bonne approximation U_{ji} de $u_j(X_i)$:

$$\begin{array}{rcccl} U_{ji} & = & u_j^h(X_i) & \approx & u_j(X_i) \\ & & \downarrow & & \\ \mathbf{U}_i & = & \mathbf{u}^h(X_i) & \approx & \mathbf{u}(X_i) \end{array}$$

où nous utilisons la même convention que plus haut, pour rendre nos notations plus compactes. Par exemple, la méthode d'Euler explicite s'écrit

$$\mathbf{U}_{i+1} = \mathbf{U}_i + h\mathbf{f}(X_i, \mathbf{U}_i)$$

En termes des composantes de $\mathbf{U}$ et $\mathbf{f}$, elle devient

$$U_{ji+1} = U_{ji} + hf_j(X_i, U_{1i}, \ldots, U_{ni}),$$

où U_{ji} est la composante j du vecteur $\mathbf{U}_i$. De même, on généralise aux systèmes les méthodes de Runge-Kutta et les méthodes d'Adams-Bashforth-Moulton. Par exemple, la méthode de Runge-Kutta d'ordre 4 classique (4.17) devient

$$\boxed{\begin{array}{rcl} \mathbf{U}_{i+1} & = & \mathbf{U}_i + \frac{h}{6}(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4) \\ \mathbf{K}_1 & = & \mathbf{f}(X_i, \mathbf{U}_i) \\ \mathbf{K}_2 & = & \mathbf{f}(X_i + \frac{h}{2}, \mathbf{U}_i + \frac{h}{2}\mathbf{K}_1) \\ \mathbf{K}_3 & = & \mathbf{f}(X_i + \frac{h}{2}, \mathbf{U}_i + \frac{h}{2}\mathbf{K}_2) \\ \mathbf{K}_4 & = & \mathbf{f}(X_i + h, \mathbf{U}_i + h\mathbf{K}_3) \end{array}} \qquad (4.29)$$

Quelques mots sur la stabilité des méthodes numériques pour des systèmes d'équations différentielles

Nous pouvons également généraliser la discussion de la stabilité du *système* d'équations différentielles et de la *méthode numérique* utilisée pour le résoudre de manière approchée. Nous nous limiterons ici à l'étude du système différentiel modèle de la forme

$$\mathbf{u}' = \mathbf{A}\mathbf{u} \tag{4.30}$$

où $\mathbf{A}$ est une matrice constante (qui représente la matrice jacobienne en un point du problème général $\mathbf{u}' = \mathbf{f}(x, \mathbf{u})$). Supposons que l'on puisse diagonaliser la matrice $\mathbf{A}$ et écrivons la décomposition

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1},$$

où $\mathbf{D}$ est la matrice diagonale formée des *valeurs propres* $\lambda_1, \lambda_2, \ldots, \lambda_n$ de $\mathbf{A}$. Ces valeurs propres peuvent être des nombres complexes[3]. En effectuant le changement de variable $\mathbf{v}(x) = \mathbf{P}^{-1}\mathbf{u}(x)$, on transforme (4.30) en l'expression

$$\mathbf{v}' = \mathbf{D}\mathbf{v} \tag{4.31}$$

Le problème différentiel initial est équivalent aux n *équations scalaires découplées*

$$v_i'(x) = \lambda_i v_i(x), \qquad\qquad i = 1, \ldots, n.$$

Le système est stable si $\Re(\lambda_i) < 0$ pour tout i. Afin d'étudier la stabilité d'une méthode numérique pour résoudre un système, il suffit d'appliquer aux équations découplées la théorie de stabilité décrite pour des équations différentielles scalaires. Une méthode particulière utilisant un pas h donné sera stable si tous les nombres complexes $h\lambda_i$ se trouvent à l'intérieur de la région de stabilité.

[3]C'est la raison pour laquelle nous avons dessiné les régions de stabilité des méthodes d'Euler et de Taylor dans le plan complexe !

Chapitre 5

Comment trouver la solution d'équations non linéaires ?

Trouver la solution d'équations non linéaires est un problème beaucoup plus difficile qu'on ne pourrait le croire de prime abord. Typiquement, on considère l'équation suivante

$$f(x) = 0$$

où f est une fonction non linéaire. Tous les nombres x pour lesquels on a $f(x) = 0$ sont appelés les racines ou les solutions de l'équation $f(x) = 0$. Certaines équations non linéaires peuvent être résolues analytiquement sans aucune difficulté. Ce n'est évidemment pas le cas pour toutes les équations non linéaires.

Les méthodes qui seront étudiées dans ce chapitre procèdent par *itérations*. Partant d'une valeur approchée x_0, un algorithme itératif permet de calculer les itérées successives $x_1, x_2, x_3, \ldots$ Les bonnes méthodes produisent des itérées qui convergent vers une solution. Le calcul est arrêté lorsqu'une précision prescrite est atteinte. Parfois, la convergence n'est garantie que pour certaines valeurs de x_0. Parfois encore, l'algorithme peut ne pas converger. Dans l'étude des différentes méthodes, on se rendra compte de l'importance cruciale d'un bon choix de la valeur x_0.

L'erreur e_i à l'itération i est l'écart entre la solution exacte et la valeur approchée obtenue après i étapes d'itération : $e_i = x - x_i$. Si la séquence d'itérations converge et s'il existe deux constantes $C < 1$ et $r \geq 1$ telles que

$$\lim_{i \to \infty} \frac{|e_i|}{|e_{i-1}|^r} = C \tag{5.1}$$

on dit que la séquence converge avec un *taux de convergence* r. Cela revient à dire qu'après un assez grand nombre d'étapes, on peut écrire

$$|e_i| < C|e_{i-1}|^r \tag{5.2}$$

Lorsque $r = 1$, on parle de *convergence linéaire*, et lorsque $r > 1$ de *convergence super-linéaire*. Lorsque $r = 2$, on dit que la convergence est *quadratique*. La séquence d'itérées converge d'autant plus rapidement si la valeur de r est élevée.

Il existe des méthodes pour lesquelles la convergence est garantie et d'autres pour lesquelles il faut satisfaire à certaines conditions de convergence, qu'il est souvent impossible ou difficile à vérifier. Les méthodes ont aussi des ordres de convergence différents. En général, il faut choisir une méthode en effectuant un compromis entre *sécurité* de la convergence et *rapidité* du calcul.

Certaines méthodes sont dites *méthodes d'encadrement* ou *bracketing methods*. Partant d'un intervalle déterminé par les deux premières approximations $[x_0, x_1]$, dans lequel on sait que se trouve une solution, on modifie à chaque étape une des limites de cet intervalle de sorte que la solution soit toujours confinée dans un intervalle dont la largeur diminue à chaque itération. Une méthode d'encadrement est donc particulièrement sûre puisqu'elle produit des intervalles de plus en plus petits dans lesquels se trouve coincée la solution. Malheureusement, ces méthodes ne sont pas généralisables pour les systèmes d'équations. A titre d'introduction, nous allons présenter la méthode de bissection.

5.1 Méthode de bissection

Supposons que l'on trouve un intervalle $[a_0, b_0]$ pour lequel la fonction $f(x)$ présente un signe différent aux deux extrémités a_0 et b_0. Supposons aussi que $f(x)$ soit continue sur cet intervalle. On peut alors garantir que la fonction $f(x)$ a au moins un zéro dans l'intervalle $[a_0, b_0]$. Cette affirmation découle immédiatement du *théorème de la valeur intermédiaire*. La méthode de *bissection* utilise cette propriété des fonctions continues. L'algorithme de la méthode de bissection peut être décrit de la manière suivante :

On fournit un intervalle $[a_0, b_0]$ avec $f(a_0)f(b_0) < 0$ et on calcule trois suites a_i, b_i, x_i par

$$x_i = \frac{(a_i + b_i)}{2}$$

Si $f(x_i) = 0$	On a une solution !
Si $f(x_i)f(a_i) > 0$	$a_{i+1} \leftarrow x_i$ $b_{i+1} \leftarrow b_i$
Sinon	$a_{i+1} \leftarrow a_i$ $b_{i+1} \leftarrow x_i$

(5.3)

où le symbole $\leftarrow$ désigne une assignation. A chaque étape de cet algorithme, on identifie un intervalle $[a_i, b_i]$ tel que les valeurs $f(a_i)$ et $f(b_i)$ donnent $f(a_i)f(b_i) < 0$. On prend ensuite le point milieu de cet intervalle $x_i = (a_i + b_i)/2$ et l'on calcule $f(x_i)$. Si, par chance, $f(x_i) = 0$, une solution de l'équation est trouvée. Si ce n'est pas le cas, on a soit $f(x_i)f(a_i) < 0$, soit $f(x_i)f(b_i) < 0$. Dans le premier cas, on est certain qu'il existe au moins une racine dans l'intervalle $[a_i, x_i]$. On peut alors remplacer les valeurs de b_i par x_i. Dans le deuxième cas, il existe au moins une racine dans l'intervalle $[x_i, b_i]$ et l'on peut remplacer les valeurs de a_i par x_i. Dans les deux cas, on se retrouve dans la situation de départ si ce n'est que l'intervalle a été réduit de moitié. On peut alors recommencer une nouvelle étape de cet algorithme et répéter ces opérations jusqu'à ce que l'intervalle soit suffisamment petit.

Il est aisé de déterminer le taux de convergence de la méthode de bissection. On sait que dans l'intervalle de départ, soit $[a_0, b_0]$, il y a une racine x et on peut donc écrire

$$|e_0| < \frac{b_0 - a_0}{2}$$

$$\downarrow \quad \text{En effectuant une nouvelle itération}$$

$$|e_1| < \frac{b_1 - a_1}{2} = \frac{b_0 - a_0}{2^2}$$

$$\downarrow$$

$$|e_n| < \frac{b_0 - a_0}{2^{n+1}}$$

La méthode de bissection est donc une méthode d'ordre 1. Son taux de convergence est linéaire.

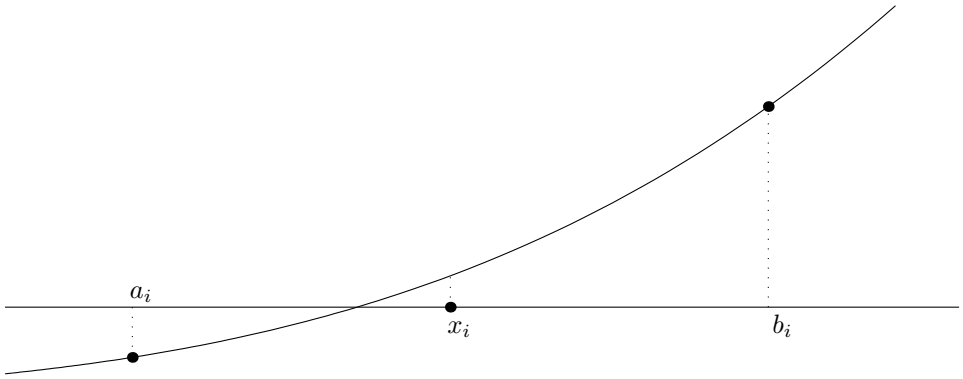


Figure 5.1: Interprétation géométrique de la méthode de la bissection.

$$|e_i| < \frac{1}{2}|e_{i-1}| \tag{5.4}$$

La mise en oeuvre de la méthode de bissection nécessite un intervalle de départ $[a_0, b_0]$ tel que $f(a_0)f(b_0) < 0$. Ceci n'est pas toujours possible. Il faut aussi que la fonction $f(x)$ soit continue sur l'intervalle de départ. Si ce n'est pas le cas, la méthode de bissection pourrait converger vers une valeur qui n'est pas une racine.

5.1.1 Application à un problème aux conditions aux limites

Lorsqu'on a un système d'équations différentielles ordinaires avec des conditions aux limites en plus d'un point de la variable indépendante, le problème est appelé *problème aux conditions aux limites*. Le cas le plus fréquent arrive lorsque des conditions sont appliquées en deux points. Typiquement, on souhaite résoudre un problème de la forme[1]:

[1] Afin de simplifier la présentation, on suppose que f n'est pas une fonction de u'. Toutefois, généraliser notre propos au cas général serait purement technique.

Trouver $u(x)$ tel que

$$\begin{cases} u''(x) &= f(x, u(x)), & x \in [a, b] \\ u(a) &= u^a \\ u(b) &= u^b \end{cases} \tag{5.5}$$

Un tel problème peut être vu, comme nous l'avons montré dans le chapitre précédent, comme un système de deux équations différentielles ordinaires du premier ordre :

Trouver $(u(x), v(x))$ tels que

$$\begin{cases} u'(x) &= v(x) \\ v'(x) &= f(x, u(x)), & x \in [a, b] \\ u(a) &= u^a \\ u(b) &= u^b \end{cases} \tag{5.6}$$

où $v(x)$ représente la dérivée première de $u(x)$. Les méthodes de Taylor, de Runge-Kutta et d'Adams-Bashforth-Moulton peuvent être utilisées pour résoudre un système d'équations différentielles dans le cadre d'un *problème aux valeurs initiales*, c'est-à-dire si l'on prescrit les valeurs $u(a)$ et $v(a)$. Par contre, dans le cas qui nous préoccupe, on n'impose pas la valeur de $v(a)$, mais celle de $u(b)$. Des problèmes similaires se rencontrent dans de nombreuses applications : citons, en vrac, la modélisation de la conduction de la chaleur, de la vibration de corde ou de membrane tendue, de la déflexion d'une poutre ou d'une coque sous une charge, de l'écoulement d'un fluide sur une surface ou de la diffusion dans des matériaux poreux.

Il y a deux approches pour résoudre un tel problème :

- une approche d'essais successifs qui combine la résolution d'un problème non linéaire avec l'utilisation des techniques classiques d'intégration d'équations différentielles dans le cas d'un problème aux valeurs initiales. Il s'agit de la *méthode du tir* ou *shooting method*.

- une seconde approche qui résout en une passe le problème en utilisant des différences finies ou des éléments finis. Toutefois, il sera alors nécessaire de résoudre un système d'équations algébriques pour obtenir la solution du problème.

Méthode du tir

Le principe de la méthode du tir est relativement simple, on fixe a priori une condition initiale arbitraire pour v. Typiquement, une telle condition initiale est trouvée sur base de l'intuition physique et de ce qui semble raisonnable : on impose donc $v(a) = v_0^a$. On intègre les équations différentielles en utilisant une méthode classique pour les problèmes aux valeurs initiales, jusqu'à la valeur b et on obtient ainsi $u_0^b = u_0^h(b)$. En général, la valeur obtenue a peu de chance de correspondre à la valeur désirée u^b.

Sur base de l'écart $r_0 = u^b - u_0^h(b)$, nous allons tenter d'ajuster la condition initiale afin d'atteindre l'objectif. On va donc sélectionner une nouvelle valeur v_1^a, effectuer une nouvelle intégration et obtenir $r_1 = u^b - u_1^h(b)$. Si la procédure itérative converge, on espère que l'écart va tendre vers zéro.

Pour estimer la correction à appliquer sur la valeur v_i^a, on va utiliser la méthode de bissection pour le problème suivant :

$$r(v^a) = 0$$

où la fonction r est définie par :

$$r : v^a \rightarrow \left(u^b - u_{v^a}^h(b)\right)$$

L'expression $u_{v^a}^h$ représente la première composante de la solution approchée du problème (5.6) avec u^a et v^a comme valeurs initiales. Chaque itération de la méthode de bissection nécessitera l'intégration numérique complète du problème aux valeurs initiales. L'utilisation de la méthode de la bissection est toujours considérée comme un algorithme particulièrement efficace dans le monde de l'artillerie : la procédure d'ajustement d'un tir d'artillerie a été très longtemps guidée par une application immédiate de la méthode de la bissection.

Méthode des différences finies

Il y a une autre manière de résoudre le problème (5.5). Nous allons évaluer les valeurs nodales U_i en un nombre fini de points $a = X_0 < X_1 < X_2 \ldots < X_m = b$. Nous recherchons donc une suite de valeurs approchées :

$$u(X_i) \approx u^h(X_i) = U_i \tag{5.7}$$

Pour obtenir les $m-1$ valeurs intermédiaires (car $U_0 = u^a$ et $U_m = u^b$, évidemment), il est nécessaire de trouver $m-1$ relations. Nous allons utiliser une approximation numérique de l'équation différentielle pour les obtenir, en écrivant pour $i = 1, \ldots, m-1$:

$$
\begin{array}{rcll}
0 \;=\; u''(X_i) - f(X_i, U_i) & \approx & (u^h)''(X_i) - f(X_i, U_i) \\
& \Big\downarrow & \text{En utilisant une différence finie centrée d'ordre deux,} \\
& \approx & \dfrac{(U_{i-1} - 2U_i + U_{i+1})}{h^2} - f(X_i, U_i)
\end{array}
$$

Ceci nous fournit les $m+1$ relations suivantes :

$$
\frac{1}{h^2}\begin{bmatrix}
h^2 & 0 & & & & & & & \\
1 & -2 & 1 & & & & & & \\
 & 1 & -2 & 1 & & & & & \\
 & & 1 & -2 & 1 & & & & \\
 & & & 1 & -2 & 1 & & & \\
 & & & & \cdot & \cdot & \cdot & & \\
 & & & & & 1 & -2 & 1 & \\
 & & & & & & 1 & -2 & 1 \\
 & & & & & & & 0 & h^2
\end{bmatrix}
\begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ \vdots \\ U_{m-2} \\ U_{m-1} \\ U_m \end{bmatrix}
=
\begin{bmatrix} u^a \\ f(X_1, U_1) \\ f(X_2, U_2) \\ f(X_3, U_3) \\ f(X_4, U_4) \\ \vdots \\ \vdots \\ f(X_{m-1}, U_{m-1}) \\ u^b \end{bmatrix}
$$

Si f n'est pas une fonction de u (ou est linéaire par rapport à u), il s'agit d'un système linéaire que l'on peut résoudre de manière habituelle. Par contre, si f est non linéaire par rapport à u, ces relations forment un système non linéaire d'équations dont la résolution peut être très délicate. Une propriété importante du système ci-dessus est son caractère creux : on observe que la matrice est tridiagonale et il est évidemment possible de construire des algorithmes spécialisés pour de tels systèmes. Ce sujet sort largement du cadre de ce cours et sera abordé plus tard dans des cours plus spécialisés.

5.2 Méthode du point fixe

La méthode de la bissection a l'avantage de la simplicité et d'une relative robustesse. Malheureusement, son taux de convergence n'est que linéaire et son extension à des systèmes d'équations est impossible.

Réécrivons l'équation $f(x) = 0$ sous une forme[2]

$$x = g(x)$$

[2]Il existe une infinité de façons d'écrire une équation $f(x) = 0$ sous une forme $x = g(x)$.
A titre d'exemple considérons $f(x) = x^3 - 3x + 1$.
On peut ainsi écrire $x = (x^3 + 1)/3$, ou $x = (3x - 1)^{1/3}$ ou encore $x = x - x^3 + 3x - 1$.

et recherchons la solution x telle que $x = g(x)$. Dans un tel formalisme, on dira que l'on recherche le *point fixe* x de la fonction g. L'algorithme de la méthode du point fixe peut alors se décrire de la manière suivante :

On fournit x_0

Tant que $|\Delta x| > \epsilon$, on calcule x_{i+1} à partir de x_i avec

$$x_{i+1} \quad = \quad g(x_i) \tag{5.8}$$

Si on converge, la solution x est le dernier x_{i+1} calculé

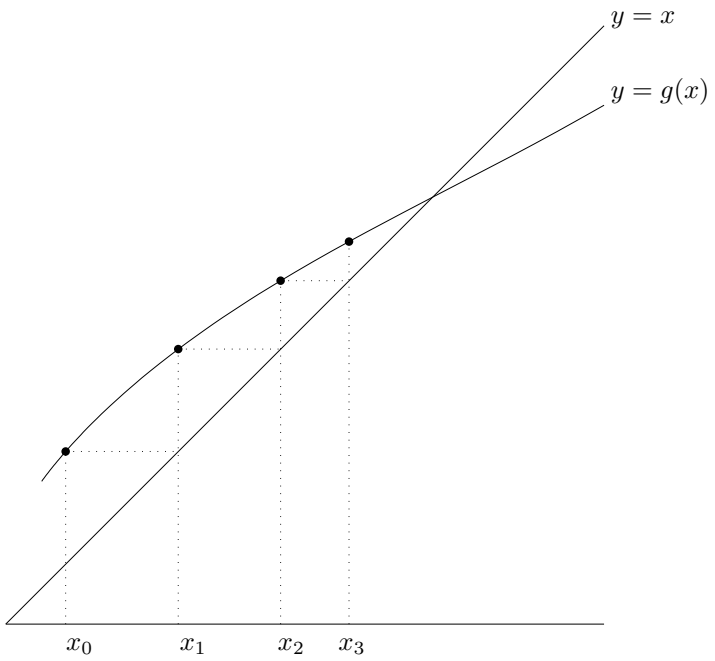


Figure 5.2: Interprétation géométrique de la méthode du point fixe : le nouveau point $(x_{i+1}, g(x_{i+1}))$ est obtenu en effectuant une projection horizontale de $(x_i, g(x_i))$ sur la droite $y = x$ et ensuite une projection verticale sur la courbe $y = g(x)$. On effectue bien ainsi l'itération $x_{i+1} = g(x_i)$.

L'interprétation graphique de la méthode d'itération est donnée à la Figure 5.2. Afin d'obtenir un nouveau point $(x_{i+1}, g(x_{i+1}))$ avec $x_{i+1} = g(x_i)$, on effectue une projection horizontale de $(x_i, g(x_i))$ sur la droite $y = x$ et ensuite une projection verticale sur la courbe $y = g(x)$. Sur le dessin, on peut comprendre intuitivement que la procédure converge car la courbe $y = g(x)$ a toujours une pente inférieure à la bissectrice $y = x$. En d'autres mots, on sent intuitivement que la méthode converge lorsque $|g'(x)| < 1$ pendant l'entièreté du processus itératif.

Convergence de la méthode

Théorème 5.1. *Supposons que $g(x)$ et $g'(x)$ sont continues sur l'intervalle $[a, b]$ qui contient le point fixe unique x de la fonction g. Si la valeur de départ x_0 est choisie dans cet intervalle et si la condition suivante (dite condition de Lipschitz) est satisfaite*

$$|g'(x)| \leq K < 1 \qquad \forall x \in [a, b],$$

alors l'itération $x_{i+1} = g(x_i)$ converge vers x.

Démonstration : En tenant compte que $x_{i+1} = g(x_i)$ et $x = g(x)$, on a

$$\begin{aligned} \underbrace{(x_{i+1} - x)}_{e_{i+1}} &= g(x_i) - g(x) \\ &\downarrow \quad \text{En vertu du théorème de la moyenne.} \\ &= g'(\xi)\ (x_i - x) \\ &\downarrow \quad \text{En vertu de la condition de Lipschitz.} \\ |e_{i+1}| &\leq K\ |e_i| \end{aligned}$$

On peut, dès lors, écrire que

$$0 \ \leq \ \lim_{i \to \infty} |x_i - x| \ \leq \ \lim_{i \to \infty} K^i\ |x_0 - x| \ = \ 0$$

puisque le facteur K^i tend vers 0 quand i tend vers l'infini, avec $0 < K < 1$. On peut donc en conclure que $\lim_{i \to \infty} |e_i| = 0$, ce qui démontre que la séquence des x_i converge vers le point fixe x.

□

Si une solution unique existe dans l'intervalle $[a, b]$, si $x_0 \in [a, b]$ et si la *condition de Lipschitz* est satisfaite, on est assuré de la convergence. Si $|g'(x)| > 1\ \forall x \in [a, b]$, il n'y aura pas convergence. Dans les autres cas, on ne peut rien garantir. Toutefois, il peut arriver que, partant d'un point pour lequel la convergence n'est pas garantie, le

hasard des itérations amène un élément de la séquence x_i dans un intervalle où il y a convergence. Finalement, on peut remarquer que la méthode du point fixe est donc une méthode d'ordre 1. On remarque aussi que la convergence est d'autant plus rapide que K est petit. En d'autres mots, parmi toutes les expressions possibles de la forme $x = g(x)$, il faudra donc privilégier celle conduisant à un $|g'(x)|$ le plus petit possible.

5.2.1 Généralisation pour des systèmes non linéaires

Les systèmes d'équations non linéaires s'écrivent

$$\left\{ \begin{array}{rcl} f_1(x_1, x_2, \ldots, x_n) & = & 0 \\ f_2(x_1, x_2, \ldots, x_n) & = & 0 \\ \cdots & & \\ f_n(x_1, x_2, \ldots, x_n) & = & 0 \end{array} \right. \tag{5.9}$$

Ce système (5.9) peut être écrit sous une forme compacte en utilisant les mêmes notations qu'au chapitre précédent

$$\mathbf{f}(\mathbf{x}) = 0 \tag{5.10}$$

Pour ces systèmes d'équations, les méthodes d'encadrement se révèlent souvent quasiment impossibles à mettre en oeuvre. Par contre, étendre la méthode du point fixe aux systèmes se fait très facilement. En utilisant notre notation compacte, il suffit d'écrire :

On fournit $\mathbf{x}_0$

Tant que $\| \Delta\mathbf{x} \| > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\mathbf{x}_{i+1} = \mathbf{g}(\mathbf{x}_i)$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé

(5.11)

On peut démontrer que la condition de Lipschitz qui garantissait la convergence dans le cas d'une équation à une inconnue peut s'exprimer maintenant comme suit. Si $\mathbf{x}_0$ est suffisamment proche d'une solution $\mathbf{x}$, il y aura convergence des itérations vers cette solution si la condition

$$\sum_{i=1}^{n} \left| \frac{\partial g_j}{\partial x_i} \right| < 1 \qquad\qquad j = 1, \ldots, n$$

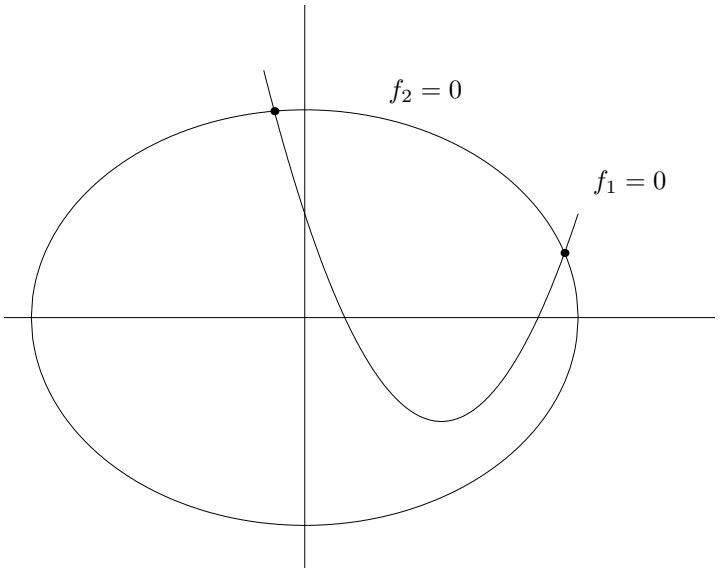


Figure 5.3: Représentation graphique du système non linéaire.

est satisfaite pendant l'entièreté du processus itératif.

A titre d'exemple, considérons le système

$$\left\{ \begin{array}{rcl} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} & = & 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} & = & 0 \end{array} \right.$$

Sur la Figure 5.3, on a représenté les courbes correspondant aux deux équations et on voit qu'elles se croisent en deux points. Il y a donc deux solutions : $\mathbf{s}_1 = (-0.222, 0.994)$ et $\mathbf{s}_2 = (1.901, 0.311)$.

Nous allons maintenant essayer d'obtenir ces deux solutions par la méthode du point fixe. On peut tirer une inconnue de chaque équation et écrire par exemple

$$\left\{ \begin{array}{rcl} x_1 = g_1(x_1, x_2) & = & \dfrac{x_1^2 - x_2 + 0.5}{2} \\ x_2 = g_2(x_1, x_2) & = & \dfrac{-x_1^2 - 4x_2^2 + 8x_2 + 4}{8} \end{array} \right.$$

Ce qui donne les formules d'itération

$$\begin{cases} x_{1,i+1} &= \dfrac{x_{1,i}^2 - x_{2,i} + 0.5}{2} \\ \\ x_{2,i+1} &= \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 8x_{2,i} + 4}{8} \end{cases}$$

Partant près de $\mathbf{s}_1$, on converge vers cette solution.

i	$x_{1,i}$	$x_{2,i}$
0	0.0000000	1.0000000
1	- 0.2500000	1.0000000
2	- 0.2187500	0.9921875
3	- 0.2221680	0.9939880
4	- 0.2223147	0.9938121
5	- 0.2221941	0.9938029
6	- 0.2222163	0.9938095
7	- 0.2222147	0.9938083
8	- 0.2222145	0.9938084
9	- 0.2222146	0.9938084

Par contre, si on part près de $\mathbf{s}_2$, on diverge.

i	$x_{1,i}$	$x_{2,i}$
0	2.0000000	0.0000000
1	2.2500000	0.0000000
2	2.7812500	- 0.1328125
3	4.1840820	- 0.6085510
4	9.3075467	- 2.4820360
5	44.8062311	- 15.8910907
6	1 011.9947186	- 392.6042650
7	512263.2073904	-205477.8225378

Pour comprendre la divergence de la méthode, regardons ce que devient la condition de Lipschitz :

$$\begin{aligned} &|x_1| + |-0.5| < 1, \\ &\left|\frac{-x_1}{4}\right| + |-x_2 + 1| < 1. \end{aligned} \tag{5.12}$$

On constate que ces conditions sont vérifiées pour $-0.5 < x_1 < 0.5$ et $1/8 < x_2 < 15/8$, ce qui explique la convergence à partir du premier point de départ. Par contre, près de la seconde solution, ces conditions ne sont pas respectées et la convergence à partir du deuxième point de départ n'est pas garantie.

On peut vérifier que cette deuxième solution peut être trouvée si l'on tire maintenant une inconnue de chaque équation de la manière suivante

$$\begin{cases} x_1 = g_1(x_1, x_2) = \dfrac{-x_1^2 + 4x_1 + x_2 - 0.5}{2} \\ \\ x_2 = g_2(x_1, x_2) = \dfrac{-x_1^2 - 4x_2^2 + 11x_2 + 4}{11} \end{cases}$$

Ce qui donne les formules d'itération

$$\begin{cases} x_{1,i+1} = \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

En utilisant ces équations à partir de la même condition initiale, on obtient une convergence du processus itératif... qui n'est pas très rapide.

i	$x_{1,i}$	$x_{2,i}$
0	2.0000000	0.0000000
1	1.7500000	0.0000000
2	1.7187500	0.0852273
3	1.7530629	0.1776676
4	1.8083448	0.2504410
8	1.9035947	0.3160782
12	1.9009241	0.3112267
16	1.9006517	0.3111994
20	1.9006772	0.3112196
24	1.9006768	0.3112186

On peut accélérer la convergence en modifiant ces itérations, en utilisant à chaque

étape la dernière valeur trouvée pour chaque inconnue. Les itérations deviennent

$$\begin{cases} x_{1,i+1} &= \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ \\ x_{2,i+1} &= \dfrac{-x_{1,i+1}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

Cette procédure s'appelle l'algorithme de Seidel.

5.2.2 Cas particulier des systèmes linéaires...

Un cas particulier intéressant des systèmes non linéaires est celui des systèmes linéaires. En utilisant la même forme compacte, un système linéaire peut s'écrire sous la forme :

$$\underbrace{\mathbf{Ax} - \mathbf{b}}_{\mathbf{f}(\mathbf{x})} = 0 \tag{5.13}$$

La résolution d'un système d'équations linéaires peut s'effectuer de multiples manières.

- On distingue d'abord les *méthodes directes* basées sur la factorisation de la matrice du système. Il s'agit de l'élimination gaussienne[3]. Il s'agit d'une approche systématique, fiable, largement utilisée, que l'on pourrait imaginer universelle. Ces méthodes directes permettent, à l'aide de manipulations de la matrice du système, de résoudre ce système en un nombre fini et calculable d'opérations. En l'absence d'erreurs d'arrondi, le résultat obtenu est exact.

  On peut montrer que le nombre d'opérations nécessaires pour résoudre un système d'ordre n par une méthode *directe* croît comme n^3. Pour de très grands systèmes, ce nombre d'opérations peut devenir prohibitif. Il est aussi requis de stocker en mémoire cette matrice de grande taille : ce qui peut aussi poser un problème pratique. Dans les cas où la matrice est *creuse*, c'est-à-dire formée d'un grand nombre d'éléments nuls, il est toutefois possible de développer des méthodes directes spécialisées qui tirent profit de cette propriété : il s'agit de solveurs pour matrices bandes ou de solveurs frontaux[4].

- Une autre classe de méthodes de résolution d'un système linéaire d'équations sont les *méthodes itératives*. La plus simple de ces méthodes est l'application de la méthode du point fixe que nous venons d'introduire. De telles méthodes se basent sur un simple calcul itératif : on part d'une approximation de la solution, on met en forme le système à résoudre de telle manière qu'on puisse l'utiliser pour calculer une meilleure approximation de la solution, et on itère le processus jusqu'à convergence.

[3] Oui, il s'agit de la technique d'échelonnement que votre enseignant d'algèbre favori vous a présentée, il y a quelques mois ! On parle aussi de triangularisation ou de factorisation de la matrice du système.

[4] Ce sont des algorithmes compliqués. Nous les présenterons brièvement dans le cours MECA1120 : Introduction aux méthodes d'éléments finis.

Contrairement aux méthodes directes, il n'est pas possible, en général, de fixer a priori le nombre d'opérations nécessaires. En effet, celui-ci dépend de la qualité de l'estimation initiale et de la précision à laquelle on décide d'arrêter les itérations. En outre, on n'est jamais sûr que de telles méthodes convergent toujours. La façon dont est mis en forme le système à résoudre prend ici toute son importance. Nous verrons que le même système d'équations, confronté au même algorithme de résolution, peut conduire à la convergence ou à la divergence suivant la façon dont le système est mis en forme... Toutefois, les méthodes itératives permettent souvent d'obtenir un résultat plus rapidement et avec moins de mémoire[5] dans un grand nombre de cas.

Méthodes itératives de Jacobi et de Gauss-Seidel

Le principe de la *méthode de Jacobi* est très simple. Il suffit d'expliciter une inconnue différente dans chaque équation du système. Partant d'une estimation initiale des différentes inconnues, on les recalcule toutes à partir de ces expressions explicitées. La procédure est recommencée jusqu'à ce que la convergence apparaisse. Il s'agit tout simplement d'un cas particulier de la méthode du point fixe ! Sous forme matricielle, la méthode de Jacobi s'écrit :

$$\mathbf{x}_{i+1} = \underbrace{\mathbf{D}^{-1}\left(\mathbf{D}-\mathbf{A}\right)\mathbf{x}_i + \mathbf{D}^{-1}\mathbf{b}}_{\mathbf{g}_{Jacobi}(\mathbf{x}_i)} \tag{5.14}$$

où $\mathbf{D}$ est la matrice diagonale formée des éléments diagonaux de $\mathbf{A}$. Observons immédiatement que la définition de $\mathbf{g}$ est un choix parmi de nombreuses possibilités. Ce choix est particulièrement simple, mais n'est pas toujours le plus efficace.

Une manière d'améliorer la convergence peut être obtenue en appliquant la technique de Seidel. On prend à chaque occasion la dernière valeur recalculée de chaque inconnue. Ainsi, à la première itération, on calculera la nouvelle valeur x_0 à partir des anciennes valeurs de $x_1, x_2, ..., x_n$, mais pour le calcul de la nouvelle valeur de x_1, on utilisera déjà la nouvelle valeur de x_0 qui vient d'être calculée, sans attendre la fin de la première itération. C'est la *méthode de Gauss-Seidel*. Sous forme matricielle, cela s'écrit :

$$\mathbf{x}_{i+1} = \underbrace{\left(\mathbf{D}+\mathbf{L}\right)^{-1}\overbrace{\left(\mathbf{D}+\mathbf{L}-\mathbf{A}\right)}^{-\mathbf{U}}\mathbf{x}_i + \left(\mathbf{D}+\mathbf{L}\right)^{-1}\mathbf{b}}_{\mathbf{g}_{Gauss-Seidel}(\mathbf{x}_i)} \tag{5.15}$$

où $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ avec $\mathbf{D}$ la matrice diagonale formée des éléments diagonaux de $\mathbf{A}$,

[5] Il est parfois possible d'éviter de devoir stocker la matrice du système, s'il est possible de directement calculer le vecteur $\mathbf{g}(\mathbf{x}_i)$ pour une approximation $\mathbf{x}_i$. C'est même souvent l'intérêt majeur des méthodes itératives pour les systèmes linéaires, aujourd'hui. La limite pour la résolution d'un très grand système sur un ordinateur donné n'est pas le temps (il suffit de faire travailler l'ordinateur la nuit...), mais la mémoire disponible à un coût raisonnable.

U la matrice triangulaire supérieure de **A** avec des zéros sur la diagonale et **L** la matrice triangulaire inférieure de **A** avec des zéros sur la diagonale. Bien que la description matricielle de la méthode de Gauss-Seidel soit un peu plus lourde que celle de Jacobi, il faut insister sur le fait que sa mise en oeuvre n'est pas plus compliquée. Au contraire, dans tous les langages de programmation, une instruction du genre $x = g(x, y, z)$ remplace automatiquement l'ancienne valeur de x par la nouvelle valeur calculée dans le membre de droite. C'est ce que demande la méthode de Gauss-Seidel.

A ce stade d'investigation formelle, on pourrait aussi imaginer une méthode itérative qui semble disposer d'une convergence exceptionnelle, en écrivant :

$$\mathbf{x}_{i+1} = \underbrace{\overbrace{\mathbf{A}^{-1}(\mathbf{A}-\mathbf{A})\mathbf{x}_i}^{0} + \mathbf{A}^{-1}\mathbf{b}}_{\mathbf{g}_{direct}(\mathbf{x}_i)} \tag{5.16}$$

Cette dernière technique converge toujours en une itération quel que soit le point de départ. Le seul inconvénient est qu'elle nécessite de calculer $\mathbf{A}^{-1}\mathbf{b}$, c'est-à-dire de résoudre le système linéaire pour pouvoir effectuer cette unique itération... En d'autres mots, toute la philosophie sous-jacente des méthodes itératives consiste à effectuer à chaque itération une résolution approximative du système, mais la moins coûteuse possible. Plus cette approximation sera efficace, meilleure sera la convergence. Pour obtenir une méthode itérative efficace, on tentera donc d'obtenir une approximation efficace mais peu coûteuse de la factorisation de la matrice **A**. Aujourd'hui, c'est encore un véritable défi pour la recherche en analyse numérique.

Finalement, il existe une modification très simple qui peut être appliquée aux méthodes itératives. Cela consiste simplement à prendre comme nouvelle itérée une moyenne pondérée entre l'ancienne valeur et celle proposée par la procédure itérative :

$$\mathbf{x}_{i+1} = (1-\omega)\ \mathbf{x}_i + \omega\ \mathbf{g}(\mathbf{x}_i) \tag{5.17}$$

où ω est un facteur d'accélération. La méthode de sur-relaxation consiste à exagérer la correction ($\omega > 1$), de façon à tendre plus vite vers la solution finale. Par contre, la méthode de sous-relaxation consiste à ralentir la correction ($\omega < 1$), de façon à éviter une éventuelle divergence. Il existe des techniques complexes qui permettent d'estimer pour un problème donné, une valeur optimale pour ω. On peut aussi utiliser d'autres méthodes complexes ou empiriques pour modifier régulièrement la valeur de ω au cours des itérations afin de converger plus rapidement... La connaissance des propriétés spectrales de la matrice du système est toutefois essentielle comme nous le verrons dans la section suivante.

Convergence des méthodes de Jacobi et de Gauss-Seidel

Les méthodes de Jacobi et Gauss-Seidel conduisent à des formules d'itération de la forme $\mathbf{x}_{i+1} = \mathbf{M}\mathbf{x}_i + \mathbf{c}$. La solution exacte correspond donc à l'égalité $\mathbf{x} = \mathbf{M}\mathbf{x} + \mathbf{c}$. Le vecteur d'erreur à l'étape $i+1$ peut s'écrire comme suit :

$$
\begin{aligned}
\underbrace{(\mathbf{x}_{i+1} - \mathbf{x})}_{\mathbf{e}_{i+1}} &= \mathbf{M}\mathbf{x}_i + \mathbf{c} - \mathbf{M}\mathbf{x} - \mathbf{c} \\
&\downarrow \\
&= \mathbf{M}\,(\mathbf{x}_i - \mathbf{x}) \\
&\downarrow \quad \text{En procédant de la même manière pour chaque étape,} \\
&= \mathbf{M}^{i+1}\ \underbrace{(\mathbf{x}_0 - \mathbf{x})}_{\mathbf{e}_0}
\end{aligned}
$$

Pour que le vecteur d'erreur $\mathbf{e}_i$ tende vers 0 lorsque les itérations progressent, il faut donc que :

$$\lim_{i \to \infty} \mathbf{M}^i\, \mathbf{e}_0 \;=\; 0$$

Est-il maintenant possible de déterminer si cette dernière condition est vérifiée pour une matrice $\mathbf{M}$ donnée? Considérons les valeurs propres λ_i et les vecteurs propres $\mathbf{v}_i$ de la matrice $\mathbf{M}$ d'ordre n. Supposons que la matrice soit diagonalisable. Les n vecteurs propres sont linéairement indépendants, et on peut écrire n'importe quel vecteur comme combinaison linéaire de ces vecteurs propres. Ainsi

$$
\begin{aligned}
\mathbf{e}_0 &= \sum_{j=1}^{n} \alpha_j \mathbf{v}_j \\
&\downarrow \quad \text{Puisque } \mathbf{M}\ \mathbf{v}_i = \lambda_i \mathbf{v}_i, \\
\mathbf{e}_i &= \sum_{j=1}^{n} \alpha_j \lambda_j^i \mathbf{v}_j
\end{aligned}
$$

Lorsque i tend vers l'infini, le vecteur d'erreur tendra donc vers zéro si

$$|\lambda_j| \;<\; 1 \qquad \forall\ j \tag{5.18}$$

En d'autres mots, le *rayon spectral* (maximum des valeurs absolues des valeurs propres) de la matrice $\mathbf{M}$ doit donc être strictement inférieur à 1. Cette condition est suffisante et nécessaire pour que n'importe quel vecteur d'erreur tende vers zéro dans le processus itératif. Toutefois, un vecteur peut tendre vers zéro, même si elle n'est pas satisfaite : en ce sens moins restrictif, elle n'est pas nécessaire. Il se pourrait en effet que, par hasard, le choix du vecteur de départ $\mathbf{x}_0$ corresponde à un vecteur d'erreur $\mathbf{e}_0$ qui n'aurait pas de composante suivant l'un ou l'autre vecteur propre. Un ou plusieurs α_i seraient donc nuls. Dans ce cas, il y a convergence même si les valeurs propres correspondantes λ_i sont, en valeur absolue, supérieures à 1. Bien sûr, on ne peut le savoir a priori. On observe aussi que la convergence est d'autant plus rapide que les valeurs propres λ_i sont petites.

Mais il faut à nouveau constater que la condition (5.18) obtenue n'est pas facile à vérifier. Il faudrait calculer les valeurs propres de la matrice $\mathbf{M}$. On peut cependant en déduire une condition suffisante assez simple, en utilisant le théorème de Gershgorin.

Théorème 5.2.

Soit $\mathbf{M}$ *une matrice diagonalisable d'ordre* n*, on a alors :*

$$|\lambda_i| \leq \sum_{j=1}^{n} |m_{ij}| \qquad i = 1, \ldots, n$$

Démonstration : considérons une valeur propre quelconque λ et le vecteur propre associé $\mathbf{v}$ de la matrice $\mathbf{M}$. Ecrivons, ensuite, la définition de valeur et vecteur propre $\lambda\mathbf{v} = \mathbf{M}\ \mathbf{v}$ en termes de composantes.

$$\lambda\ v_j = \sum_{k=1}^{n} m_{jk}\ v_k \qquad j = 1 \ldots n$$

$$|\lambda|\ |v_j| \leq \sum_{k=1}^{n} |m_{jk}|\ |v_k| \qquad j = 1 \ldots n$$

$\Big\downarrow$ En choisissant j correspondant à la composante v_j la plus grande en valeur absolue,

$$|\lambda|\ |v_j| \leq \sum_{k=1}^{n} |m_{jk}|\ |v_j|$$

$$|\lambda| \leq \sum_{k=1}^{n} |m_{jk}| \qquad \square$$

De ce théorème, on peut déduire une condition suffisante pour que toutes les valeurs propres soient, en valeur absolue, strictement inférieures à 1.

$$\sum_{j=1}^{n} |m_{ij}| < 1 \qquad i = 1, \ldots, n \tag{5.19}$$

En ce qui concerne la méthode de Jacobi, la relation (5.14) permet de repasser de la matrice $\mathbf{M}$ à la matrice $\mathbf{A}$. Une condition suffisante devient alors

$$\sum_{j=1,\neq i}^{n} \left| \frac{a_{ij}}{a_{ii}} \right| < 1 \qquad i = 1, \ldots, n$$

ou, plus simplement,

$$\sum_{j=1,\neq i}^{n} |a_{ij}| \; < \; |a_{ii}| \qquad i = 1, \ldots, n \tag{5.20}$$

Une condition *suffisante* de convergence de la méthode de Jacobi est donc ce qu'on appelle la *dominance diagonale* de la matrice $\mathbf{A}$ du système. Cela signifie que pour chaque ligne de cette matrice, le terme diagonal doit être, en valeur absolue, strictement supérieur à la somme des valeurs absolues des autres termes. On peut montrer que la conclusion est la même pour la méthode de Gauss-Seidel.

5.3 Méthode de Newton-Raphson

Nous allons maintenant essayer d'obtenir un meilleur taux de convergence en modifiant notre stratégie.

Partons d'un point x_0 et développons $f(x)$ en série de Taylor autour du point x_0

$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{(x - x_0)^2}{2} f''(x_0) + \ldots \tag{5.21}$$

Pour trouver x, nous limitons cette série de Taylor après le terme d'ordre 1 et nous exprimons que le souhait est de trouver une valeur de x telle que $f(x) = 0$. Mais comme on a tronqué la série de Taylor, ceci ne constitue qu'une approximation de la solution cherchée. On appelle x_1 la valeur ainsi trouvée et on recommence pour obtenir x_2. On obtient, de cette manière, l'algorithme suivant

On fournit x_0

Tant que $|\Delta x| > \epsilon$, on calcule x_{i+1} à partir de x_i avec

$$\begin{aligned} f'(x_i)\overbrace{(x_{i+1} - x_i)}^{\Delta x} &= -f(x_i) \\ x_{i+1} &= x_i + \Delta x \end{aligned} \tag{5.22}$$

Si on converge, la solution x est le dernier x_{i+1} calculé

L'interprétation graphique de la méthode d'itération de Newton-Raphson est donnée à la Figure 5.4. La tangente à la courbe en x_i a comme coefficient angulaire $f'(x_i)$. On a donc que $\Delta x = -f(x_i)/f'(x_i)$ et cette distance correspond, d'après la formule de Newton-Raphson, à la correction à apporter à x_i pour trouver x_{i+1}. Le point x_{i+1} est donc à l'intersection de la tangente à la courbe en x_i avec l'axe des abscisses.

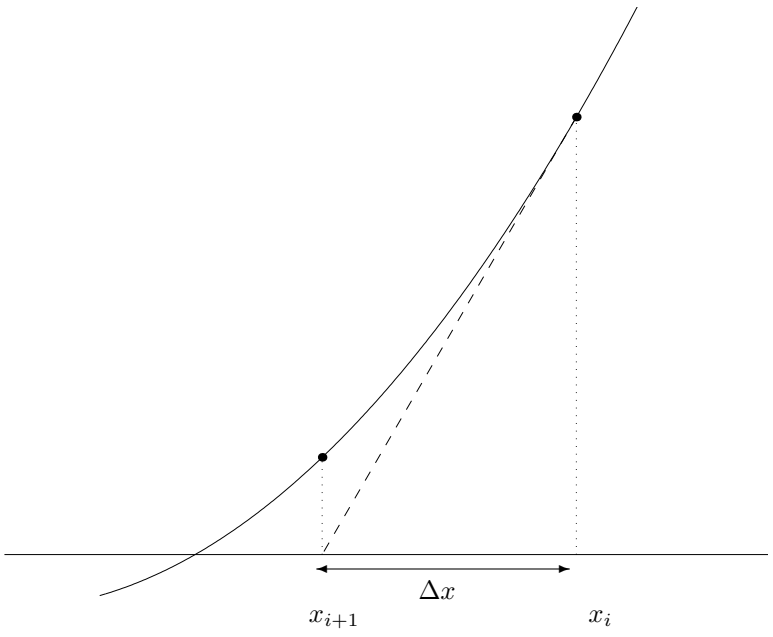


Figure 5.4: Interprétation géométrique de la méthode de Newton-Raphson : le nouveau point est obtenu comme l'intersection de la tangente et de l'axe des abscisses.

Convergence de la méthode

Supposons que $f(x)$ possède deux dérivées continues $f'(x)$ et $f''(x)$, et que nous recherchions une racine simple[6] de notre équation. Si x_0 est *suffisamment proche* de x, nous allons montrer que la méthode itérative de Newton-Raphson jouit d'un taux de convergence quadratique.

Sur base de la définition de l'erreur $e_i = x - x_i$ à l'étape i, on peut écrire :

$$\begin{aligned} e_{i+1} &= x - x_{i+1} \\ &= x - \left(x_i - \frac{f(x_i)}{f'(x_i)} \right) \\ &= \frac{e_i f'(x_i) + f(x_i)}{f'(x_i)} \end{aligned} \tag{5.23}$$

Ecrivons ensuite $f(x) = 0$ en termes d'un développement en série de Taylor autour du point x_i :

$$\begin{aligned} 0 &= f(x) \\ &= f(x_i) + \underbrace{(x - x_i)}_{e_i} f'(x_i) + \underbrace{(x - x_i)^2}_{e_i^2} \frac{f''(\xi)}{2} \end{aligned} \tag{5.24}$$

En combinant (5.23) et (5.24), on obtient finalement

$$e_{i+1} = \underbrace{-\frac{1}{2} \frac{f''(\xi)}{f'(x_i)}}_{C} e_i^2 \tag{5.25}$$

où ξ est un point particulier compris entre x et x_i. Pour avoir une convergence quadratique, il faut que C existe et que sa valeur absolue soit inférieure à l'unité. Ce n'est pas du tout évident, mais on l'observe pourtant dans le voisinage proche d'une racine simple. En effet, on peut démontrer que, si x_0 est *suffisamment près* de la solution x, les x_i successifs se rapprochent indéfiniment de x et donc que la méthode converge. Ce résultat est connu comme le *théorème local de convergence de la méthode de Newton-Raphson* et suppose que la racine cherchée est une racine simple. Dans le cas d'une racine multiple, le taux

[6] En supposant les conditions de continuité et de différentiabilité adéquates, on dira que $f(x) = 0$ a une *racine d'ordre m* en x si et seulement si $f(x) = f'(x) = f''(x) = \ldots = f^{(m-1)}(x) = 0$ et $f^{(m)}(x) \neq 0$. Des racines d'ordre un, deux ou trois sont appelées *racines simples*, *racines doubles* ou *racines triples*, respectivement.

de convergence n'est que linéaire.[7]

Lorsqu'on est dans le domaine de la convergence quadratique de la méthode de Newton et lorsqu'on observe à l'étape i une erreur $|e_i| < 10^{-k}$, on aura à l'étape suivante $|e_{i+1}| < 10^{-2k}$. En d'autres mots, l'estimation x_{i+1} aura approximativement deux fois plus de chiffres significatifs corrects que l'estimation x_i. Cette convergence est donc extrêmement rapide, si on est suffisamment proche de la solution. La méthode de Newton-Raphson convergeant très vite près de la solution sera donc souvent utilisée pour affiner très rapidement une solution que l'on a trouvée approximativement par une méthode à convergence plus lente comme la bissection.

Et quand cela ne marche pas...

Une première difficulté apparaît si on a $f'(x_i) = 0$: la formule d'itération n'est plus applicable. Le point x_{i+1} est expédié à l'infini. Lorsque la valeur de la pente de la tangente $f'(x_i)$ est non nulle, mais très petite, le point x_{i+1} peut se retrouver très loin de x_i. Parfois, on convergera, ainsi, par accident, vers une racine très éloignée de x_0. Si la fonction $f(x)$ est positive et décroît de façon monotone sur l'intervalle $[a, \infty[$ et si $x_0 > a$, la séquence x_i divergera vers $+\infty$. Des situations de bouclage peuvent aussi se produire si des termes de la séquence x_i se répètent.

---

[7] Il suffit de considérer que si x est une racine simple de $f(x)$, alors cette même valeur sera une racine double pour $g(x) = (f(x))^2$. L'application de la méthode de Newton-Raphson à g produit des incréments donnés par la relation :

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

On observe que les incréments sont deux fois plus petits que pour la méthode de Newton-Raphson appliquée à f et on peut en déduire que :

$$\begin{aligned} e_{i+1} &= x - x_{i+1} \\ &= x - \left(x_i - \frac{f(x_i)}{2f'(x_i)} \right) \\ &= \frac{e_i}{2} + \frac{e_i f'(x_i) + f(x_i)}{2f'(x_i)} \\ &= \frac{e_i}{2} + Ce_i^2 \end{aligned}$$

La méthode ne converge donc que linéairement (terme dominant). Les petits futés remarqueront qu'il est possible d'obtenir une méthode qui converge quadratiquement vers une racine de multiplicité m en multipliant l'incrément fourni par la méthode de Newton-Raphson par la multiplicité m de la racine. Evidemment, il faut connaître a priori la multiplicité de la solution que l'on recherche. Ce qui n'est pas évident !

5.3.1 Approximation numérique de la dérivée : méthode de la sécante

Si la dérivation analytique de $f(x)$ est trop compliquée ou même impossible, on recourt à la dérivation numérique. On approxime $f'(x_i)$ par l'expression

$$f'(x_i) \approx \frac{f(x_i + h) - f(x_i - h)}{2h} \tag{5.26}$$

Cela demande d'effectuer deux évaluations de $f(x)$ à chaque itération. Il faut aussi rappeler que pour une dérivation numérique, le choix du pas est particulièrement délicat. Il faut que h soit suffisamment petit pour que l'erreur de discrétisation soit faible. Il faut aussi que h ne soit pas trop petit sous peine de voir les erreurs d'arrondi rendre le résultat numérique inutilisable.

Un choix relativement astucieux est de prendre une différence décentrée sur base de $f(x_i)$ et de $f(x_{i-1})$ pour approximer $f'(x_i)$.

$$f'(x_i) \approx \frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i} \tag{5.27}$$

et on obtient ainsi ce qu'on appelle la *méthode de la sécante*, dont l'interprétation géométrique est reprise sur la Figure 5.5.

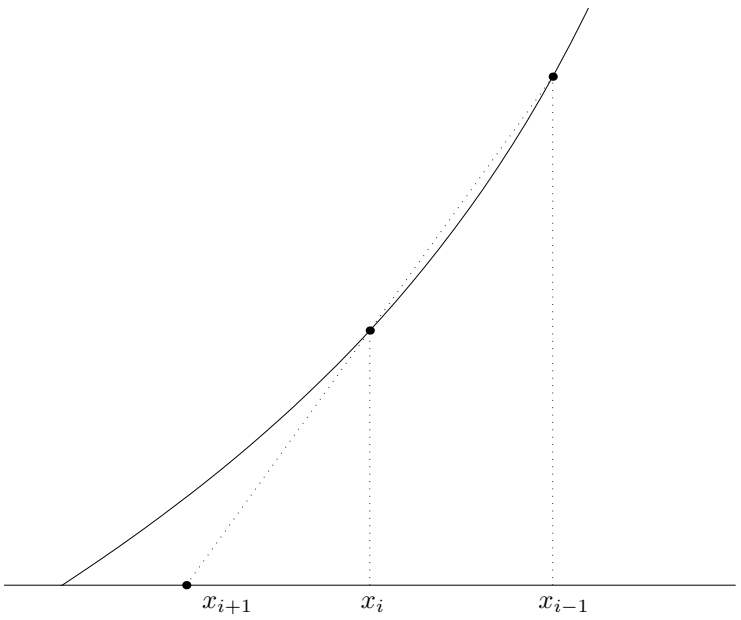


Figure 5.5: Avec des dérivées numériques, la méthode de Newton-Raphson se ramène à la méthode de la sécante.

La méthode de la sécante se ramène donc, à partir de deux estimations successives x_{i-1} et x_i, à calculer l'itération suivante x_{i+1} par la relation

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})} \tag{5.28}$$

Il faut à nouveau remarquer que, contrairement à la méthode de bissection, la méthode de la sécante et celle de Newton-Raphson ne sont pas des méthodes d'*encadrement* puisqu'il n'est pas certain que les intervalles successifs $[x_i, x_{i+1}]$ contiennent la solution cherchée. Il se peut que les valeurs successives x_i ne convergent pas vers la solution.

Toutefois, on peut démontrer que le taux de convergence de la méthode de la sécante est de 1.618 dans une région proche de la solution.

$$|e_{i+1}| = C|e_i|^{1.618} \tag{5.29}$$

Ce taux de convergence de la méthode de la sécante est superlinéaire, mais pas quadratique. On pourrait croire que l'utilisation d'une formule de dérivation numérique a réduit la qualité de la méthode numérique, puisque la méthode de la sécante a un taux de convergence plus faible que celui de Newton-Raphson. Cette comparaison est trompeuse ! Pour être vraiment correct, il faut observer qu'une itération de Newton-Raphson nécessite le calcul de $f(x_i)$ et de $f'(x_i)$, tandis qu'une itération de la sécante ne nécessite que le calcul de $f(x_i)$ (puisque $f(x_{i-1})$ a été calculé à l'itération précédente). En d'autres mots, il faudrait comparer le gain d'une itération de Newton-Raphson, à celui des deux itérations de la sécante. Le rapport est alors nettement plus favorable à la méthode de la sécante, puisque :

$$1.618^2 = 2.618 > 2$$

5.3.2 Généralisation pour des systèmes non linéaires

Reconsidérons le système d'équations non linéaires écrit sous une forme compacte.

$$\mathbf{f}(\mathbf{x}) = 0$$

Pour de tels systèmes, on a souvent recours à la méthode de Newton-Raphson. Evidemment, il faut choisir un bon candidat initial. Cet aspect du problème est très souvent une tâche très délicate qui fait partie des secrets de fabrication des concepteurs de logiciels numériques.

Nous allons effectuer ici un commentaire très prudent. Il n'y a pas de bonnes méthodes numériques pour trouver la solution de systèmes d'équations non linéaires. Il est même

relativement facile de comprendre pourquoi il n'y aura jamais de bonnes méthodes totalement générales. Le cas multidimensionnel correspond à un labyrinthe numérique semé d'embuches et de pièges qui rend la résolution du problème général très ardue. En pratique, seule la méthode de Newton-Raphson fonctionne bien, si on part suffisamment proche de la solution : ce qui est une hypothèse très restrictive. Pour partir près de la solution, il faut avoir une bonne idée intuitive de celle-ci, ce qui n'est pas du tout évident.

Etendre la méthode de Newton-Raphson aux systèmes se fait très facilement. En utilisant notre notation compacte, il suffit d'écrire :

On fournit $\mathbf{x}_0$

Tant que $||\Delta\mathbf{x}|| > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\begin{aligned} \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i)\overbrace{(\mathbf{x}_{i+1}-\mathbf{x}_i)}^{\Delta\mathbf{x}} &= -\mathbf{f}(\mathbf{x}_i) \\ \mathbf{x}_{i+1} &= \mathbf{x}_i + \Delta\mathbf{x} \end{aligned} \tag{5.30}$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé

où on identifie la matrice jacobienne du système non linéaire d'équations sous la forme

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i) = \left.\frac{\partial f_j}{\partial x_k}\right|_{(x_{1,i},x_{2,i},\ldots,x_{n,i})}$$

Pour un système de deux équations à deux inconnues, on peut résumer la méthode comme suit. On a une estimation $x_{1,i}$ et $x_{2,i}$, on calcule Δx_1 et Δx_2, solutions du système linéaire

$$\begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{(x_{1,i},x_{2,i})} & \left.\frac{\partial f_1}{\partial x_2}\right|_{(x_{1,i},x_{2,i})} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{(x_{1,i},x_{2,i})} & \left.\frac{\partial f_2}{\partial x_2}\right|_{(x_{1,i},x_{2,i})} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} -f_1(x_{1,i},x_{2,i}) \\ -f_2(x_{1,i},x_{2,i}) \end{bmatrix} \tag{5.31}$$

On détermine ainsi $x_{1,i+1} = x_{1,i} + \Delta x_1$ et $x_{2,i+1} = x_{2,i} + \Delta x_2$ et on recommence.

A titre d'exemple, reprenons à nouveau le système

$$\left\{\begin{array}{rcl} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} & = & 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} & = & 0 \end{array}\right.$$

Sur la Figure 5.3, on avait vu qu'il y a deux solutions. Maintenant, tentons de trouver ces solutions par la méthode de Newton-Raphson. On obtient d'abord l'expression symbolique de la matrice jacobienne du système

$$\left.\frac{\partial f_j}{\partial x_k}\right|_{(x_1,x_2)} = \begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial f_1}{\partial x_2}\right|_{(x_1,x_2)} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial f_2}{\partial x_2}\right|_{(x_1,x_2)} \end{bmatrix}$$

$$= \begin{bmatrix} 2x_1 - 2 & -1 \\ 2x_1 & 8x_2 \end{bmatrix}$$

Si l'on part de $x_{1,0} = 2$ et $x_{2,0} = 0.25$, le système à résoudre s'écrit

$$\begin{bmatrix} 2 & -1 \\ 4 & 2 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} -0.25 \\ -0.25 \end{bmatrix}$$

On en déduit : $\Delta x_1 = -0.09375$ et $\Delta x_2 = 0.0625$, donc $x_{1,1} = 1.90625$ et $x_{2,1} = 0.3125$. Si l'on poursuit les itérations, on observe bien que la méthode de Newton-Raphson se révèle très rapide lorsqu'on est près d'une solution.

i	$x_{1,i}$	$x_{2,i}$
0	2.000000	0.250000
1	1.906250	0.312500
2	1.900691	0.311213
4	1.900677	0.311219

5.3.3 Application de Newton-Raphson : optimisation non linéaire

On dispose de n mesures (X_i, U_i) d'une fonction inconnue $y = u(x)$ et on souhaite approximer $u(x)$ par une expression

$$u^h(x) = \frac{a}{x+b}$$

en ajustant les deux paramètres réels a et b afin de minimiser la somme du carré des n écarts $U_i - u^h(X_i)$. Ce problème d'approximation correspond clairement à un problème de minimisation d'une fonction à plusieurs variables. Il faut évidemment remarquer que les variables en question sont a et b, soit les paramètres de la fonction $u^h(x)$ et non la variable x !

Il s'agit donc de minimiser la fonction suivante :

$$f(a,b) = \sum_{i=1}^{n} \left(U_i - \frac{a}{(X_i+b)} \right)^2$$

On en déduit les deux équations (non linéaires) que doivent satisfaire a et b.

$$\begin{aligned} 0 &= \frac{\partial f}{\partial a} = \sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{-1}{(X_i+b)} = -2\sum_{i=1}^{n}\frac{U_i}{(X_i+b)} + 2a\sum_{i=1}^{n}\frac{1}{(X_i+b)^2} \\ 0 &= \frac{\partial f}{\partial b} = \sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{a}{(X_i+b)^2} = 2a\sum_{i=1}^{n}\frac{U_i}{(X_i+b)^2} - 2a^2\sum_{i=1}^{n}\frac{1}{(X_i+b)^3} \end{aligned}$$

On va utiliser la méthode de Newton-Raphson pour la résolution de ces deux équations à partir d'une estimation initiale (a_0, b_0). Le schéma de Newton-Raphson pour les deux équations s'écrit :

On calcule (a_{k+1}, b_{k+1}) à partir de (a_k, b_k) avec

$$\begin{bmatrix} \frac{\partial^2 f}{\partial a^2}(a_k,b_k) & \frac{\partial^2 f}{\partial b \partial a}(a_k,b_k) \\ \frac{\partial^2 f}{\partial a \partial b}(a_k,b_k) & \frac{\partial^2 f}{\partial b^2}(a_k,b_k) \end{bmatrix} \begin{bmatrix} \Delta a \\ \Delta b \end{bmatrix} = - \begin{bmatrix} \frac{\partial f}{\partial a}(a_k,b_k) \\ \frac{\partial f}{\partial b}(a_k,b_k) \end{bmatrix}$$

$$\begin{aligned} a_{k+1} &= a_k + \Delta a \\ b_{k+1} &= b_k + \Delta b \end{aligned}$$

avec les dérivées partielles secondes données par les expressions suivantes :

$$\begin{aligned}
\frac{\partial^2 f}{\partial a^2} &= 2\sum_{i=1}^{n} \frac{1}{(X_i+b)^2} \\
\frac{\partial^2 f}{\partial b^2} &= -4a\sum_{i=1}^{n} \frac{U_i}{(X_i+b)^3} + 6a^2\sum_{i=1}^{n} \frac{1}{(X_i+b)^4} \\
\frac{\partial^2 f}{\partial a \partial b} = \frac{\partial^2 f}{\partial b \partial a} &= 2\sum_{i=1}^{n} \frac{U_i}{(X_i+b)^2} - 4a\sum_{i=1}^{n} \frac{1}{(X_i+b)^3}
\end{aligned}$$

A titre d'exemple, le schéma de Newton-Raphson convergera quadratiquement vers un minimum[8], si on choisit un candidat initial proche de la solution... ce qui est à nouveau difficile à réaliser en pratique. Observons toutefois qu'un tel processus itératif recèle des dangers potentiels : l'apparition d'une valeur de b opposée à une des données X_i impliquera l'apparition de dénominateurs nuls dans toutes les expressions du schéma de Newton-Raphson et va donc poser quelques difficultés. Typiquement, on observera...

```
Warning: Divide by zero.
ans = Inf
```

5.3.4 Application de Newton-Raphson : Euler implicite pour une EDO non linéaire raide

Considérons le problème différentiel aux valeurs initiales :

Trouver $(u(x), v(x))$ tels que

$$\begin{cases} u'(x) = -u^2(x) + v(x) \\ v'(x) = -50\ v^2(x) + x, \qquad x \in [0,4] \\ u(0) = 1 \\ v(0) = 1 \end{cases} \tag{5.32}$$

La matrice jacobienne du système s'écrit

[8] Pourquoi un minimum et non un maximum ?

$$\begin{bmatrix} \left.\frac{\partial f_1}{\partial u}\right|_x & \left.\frac{\partial f_1}{\partial v}\right|_x \\ \left.\frac{\partial f_2}{\partial u}\right|_x & \left.\frac{\partial f_2}{\partial v}\right|_x \end{bmatrix} = \begin{bmatrix} -2u & 1 \\ 0 & -100v \end{bmatrix}$$

Les valeurs propres vérifient la relation

$$\det \begin{bmatrix} -2u(x) - \lambda & 1 \\ 0 & -100\ v(x) - \lambda \end{bmatrix} = 0$$

$$\downarrow$$

$$\begin{aligned} \lambda_1(x) &= -100\ v(x) \\ \lambda_2(x) &= -2\ u(x) \end{aligned}$$

La Figure 5.6 illustre les solutions exactes $u(x)$ et $v(x)$ sur l'intervalle $[0, 4]$. A l'évidence, le problème différentiel est raide sur presque tout l'intervalle $[0, 4]$, mais plus particulièrement, au démarrage, où $\lambda_1 = -100$.

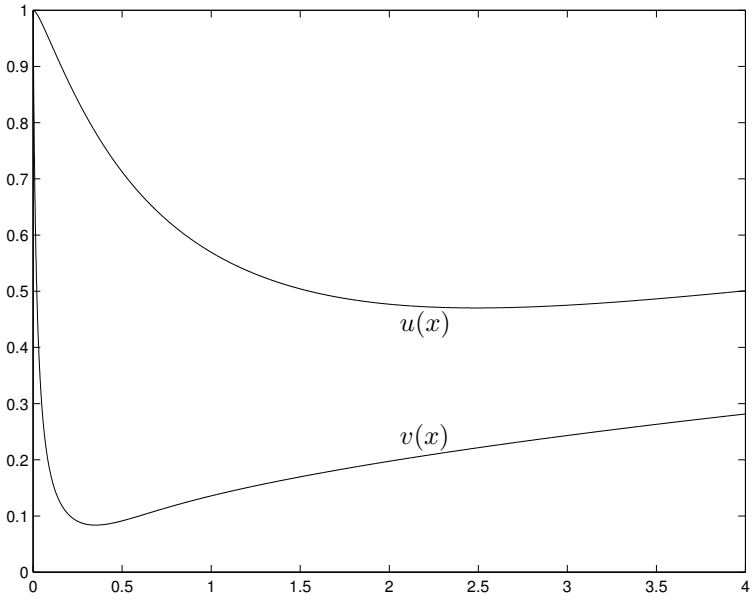


Figure 5.6: Solution exacte du système non linéaire.

Appliquer la méthode d'Euler explicite revient à écrire

$$\begin{cases} U_{i+1} = U_i + h\ (-U_i^2 + V_i) \\ V_{i+1} = V_i + h\ (-50\ V_i^2 + X_i) \end{cases}$$

Malheureusement, le pas h est contraint par la condition de stabilité $h < 0.02$. La méthode d'Euler implicite permettra de ne pas limiter le pas de temps, mais en contrepartie, un problème algébrique non linéaire devra être résolu à chaque pas de temps. Appliquer la méthode d'Euler implicite nécessite d'écrire

$$\left\{ \begin{array}{rcl} U_{i+1} & = & U_i + h \ \left(-U_{i+1}^2 + V_{i+1} \right) \\ V_{i+1} & = & V_i + h \ \left(-50 \ V_{i+1}^2 + X_{i+1} \right) \end{array} \right.$$

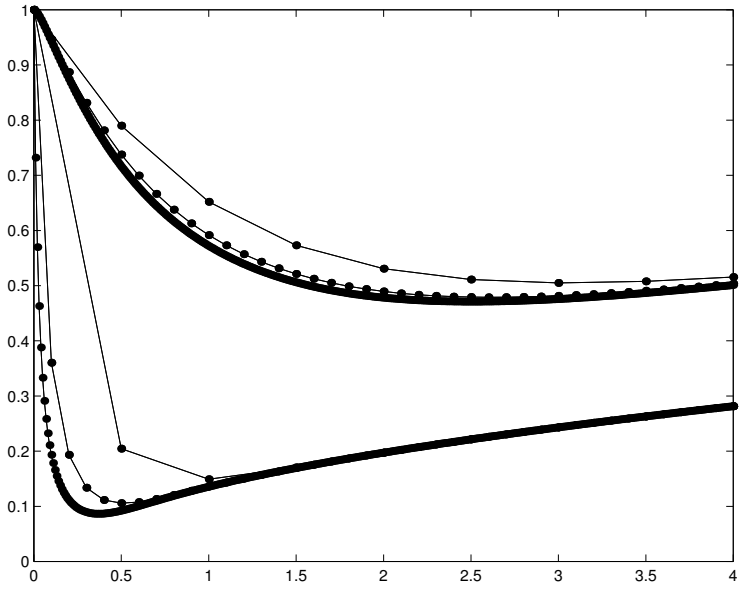


Figure 5.7: Méthode d'Euler implicite : analyse de convergence avec $h = 0.5$, $h = 0.1$ et $h = 0.01$.

Ces équations sont non linéaires en U_{i+1} et V_{i+1}. On peut les écrire sous la forme

$$\left\{ \begin{array}{rcl} \overbrace{U_{i+1} - U_i - h \ \left(-U_{i+1}^2 + V_{i+1} \right)}^{g_1(U_{i+1}, V_{i+1})} & = & 0 \\ \underbrace{V_{i+1} - V_i - h \ \left(-50 \ V_{i+1}^2 + X_{i+1} \right)}_{g_2(U_{i+1}, V_{i+1})} & = & 0 \end{array} \right. \tag{5.33}$$

que nous pouvons écrire sous la forme compacte

$$\mathbf{g}(\mathbf{x}) = 0 \tag{5.34}$$

où le vecteur $\mathbf{x} = \mathbf{U}_{i+1}$ contient les deux inconnues U_{i+1} et V_{i+1}. Nous allons résoudre (5.34) par la méthode de Newton-Raphson. La matrice jacobienne du schéma de Newton-Raphson s'écrit

$$\begin{bmatrix} \left.\frac{\partial g_1}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_1}{\partial x_2}\right|_{(x_1,x_2)} \\ \left.\frac{\partial g_2}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_2}{\partial x_2}\right|_{(x_1,x_2)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - h \begin{bmatrix} -2x_1 & 1 \\ 0 & -100x_2 \end{bmatrix}$$

On peut observer que la matrice jacobienne du schéma de Newton-Raphson correspond à la matrice identité à laquelle on a soustrait la matrice jacobienne du problème différentiel multipliée par un facteur h. On peut alors imbriquer le schéma de Newton-Raphson dans la boucle de calcul des pas d'intégration temporelle :

On fournit $\mathbf{U}_0$

Pour chaque i, on calcule $\mathbf{x} = \mathbf{U}_{i+1}$ à partir de $\mathbf{U}_i$ en résolvant le problème

$$\underbrace{\mathbf{U}_{i+1} - \mathbf{U}_i - h\ \mathbf{f}(X_{i+1}, \mathbf{U}_{i+1})}_{\mathbf{g}(\mathbf{x})} = 0$$

par la méthode de Newton-Raphson en partant de $\mathbf{U}_i$

On fournit $\mathbf{x}_0 = \mathbf{U}_i$

Tant que $||\Delta\mathbf{x}|| > \epsilon$, on calcule $\mathbf{x}_{j+1}$ à partir de $\mathbf{x}_j$ avec

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x}_j) \overbrace{(\mathbf{x}_{j+1} - \mathbf{x}_j)}^{\Delta\mathbf{x}} = -\mathbf{g}(\mathbf{x}_j)$$

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \Delta\mathbf{x}$$

Si on converge, la solution $\mathbf{U}_{i+1}$ est le dernier $\mathbf{x}_{j+1}$ calculé

A chaque pas, on démarre les itérations de Newton-Raphson avec les valeurs convergées au pas précédent. Comme les itérations du schéma de Newton-Raphson sont bien imbriquées dans la boucle de calcul de chaque pas de la méthode d'intégration temporelle, il est essentiel d'obtenir la convergence de la méthode de Newton-Raphson pour que le

schéma global fonctionne : une telle exigence impose parfois des limites très strictes sur le pas de temps afin que le candidat initial de la méthode de Newton-Raphson soit suffisamment proche de la solution. On peut aussi montrer qu'il est inutile d'exiger une précision plus grande pour la convergence du schéma de Newton-Raphson que celle du schéma d'intégration temporelle. Les résultats obtenus à la Figure 5.7 illustrent la convergence globale du schéma d'Euler implicite.

Chapitre 6

Comment résoudre un problème aux conditions aux limites ?

La plupart des modèles mathématiques se présentent sous la forme de systèmes d'équations différentielles. Des conditions initiales et des conditions limites sont en général requises pour compléter le modèle. Pour résoudre numériquement de tels problèmes, l'idée de base consiste à ne rechercher que la valeur des fonctions inconnues qu'en un grand nombre fini de points : il s'agit de la discrétisation. Au lieu de résoudre un problème différentiel ou problème continu, nous allons résoudre un grand système linéaire qu'on appelle le problème discret. Comment obtenir un tel problème discret ? C'est le rôle d'une méthode numérique telle que les différences finies ou les éléments finis.

Il est usuel de classer les équations aux dérivées partielles en trois grandes catégories : elliptique, parabolique et hyperbolique. Les équations appartenant à une même classe ont des propriétés mathématiques semblables et modélisent également des comportements physiques semblables. Typiquement, nous allons considérer les équations modèles les plus simples pour chaque catégorie.

- L'équation de Poisson bidimensionnelle est le prototype le plus simple d'équation *elliptique* et représente typiquement la diffusion stationnaire de la chaleur.

$$\underbrace{k\left(\frac{\partial^2 u}{\partial x^2}(x,y)+\frac{\partial^2 u}{\partial y^2}(x,y)\right)}_{\boldsymbol{\nabla}^2 u(x,y)}+f(x,y)=0 \tag{6.1}$$

  Il s'agit donc de déterminer la fonction inconnue $u(x,y)$ qui représente la température, tandis que k et f sont respectivement la conductivité thermique et la densité de puissance calorifique fournie à distance. Mathématiquement, on peut montrer que pour obtenir un problème bien posé (existence d'une solution unique dépendant continûment des données), il est requis d'imposer une condition aux limites sur la

totalité de la frontière du domaine de calcul, soit en y prescrivant la température soit en y prescrivant le flux de chaleur[1]. Les solutions sont continues à l'intérieur du domaine de calcul, même si les conditions aux limites sont discontinues.

- L'équation unidimensionnelle de la chaleur fournit le modèle le plus élémentaire d'équation *parabolique* et représente la diffusion thermique instationnaire.

$$\rho c \frac{\partial u}{\partial t}(x,t) = k \frac{\partial^2 u}{\partial x^2}(x,t) \tag{6.2}$$

où ρ et c représentent respectivement la masse volumique et la chaleur spécifique du matériau. Les solutions décroissent de manière exponentielle dans le temps à partir d'une condition initiale et tendent vers une solution d'équilibre. L'information se propage à une vitesse infinie et toutes les discontinuités présentes dans la solution initiale et les conditions aux limites disparaissent instantanément.

- L'équation unidimensionnelle d'onde est finalement notre prototype d'équation *hyperbolique*.

$$\frac{\partial^2 u}{\partial t^2}(x,t) = c^2 \frac{\partial^2 u}{\partial x^2}(x,t) \tag{6.3}$$

où c est la vitesse finie de propagation de l'information. Dans ce cas-ci, les discontinuités subsisteront et seront propagées dans le domaine de calcul. Pour obtenir un problème aux conditions aux limites bien posé, il faudra désormais imposer une solution initiale et des conditions aux frontières lorsque les caractéristiques sont entrantes.

Cette terminologie provient d'une analyse de l'opérateur différentiel qui est tout à fait analogue à celle effectuée avec les coniques et les sections coniques. Pour résoudre des problèmes aux conditions aux limites, la méthode la plus simple est celle des différences finies.

[1]En réalité, c'est même un fifrelin plus compliqué, il faut que la température soit au moins prescrite en un point pour avoir une solution unique. En effet, imposer une source nulle et un flux nul sur l'ensemble de la frontière permet de déduire que la température sera constante... Mais la valeur de la constante restera indéterminée.

6.1 Equation de Poisson

Afin d'illustrer la méthode des différences finies, considérons le problème aux conditions aux limites suivant :

$$\boxed{\begin{array}{l}\text{Trouver } u(x,y) \text{ tel que} \\ \qquad\qquad \boldsymbol{\nabla}^2 u(x,y) + 1 = 0, \quad (x,y) \in \Omega, \\ \qquad\qquad u(x,y) = 0, \quad (x,y) \in \partial\Omega,\end{array}} \tag{6.4}$$

où le domaine Ω est le carré dont le côté vaut deux et dont le centre est l'origine du plan. Exceptionnellement, il est possible de trouver une solution analytique, en tirant profit de la symétrie du problème (fonction source et conditions aux limites). On pressent que la solution analytique s'écrira sous la forme suivante :

$$u(x,y) = \sum_{i,j \text{ impairs}} C_{ij} \sin\left(\frac{i\pi(x+1)}{2}\right) \sin\left(\frac{j\pi(y+1)}{2}\right)$$

Cette série satisfait les conditions aux limites. Pour obtenir les coefficients C_{ij}, il reste à satisfaire l'équation de Poisson et à écrire :

$$\begin{aligned} \boldsymbol{\nabla}^2 u(x,y) + 1 &= 0, \\ \sum_{i,j \text{ impairs}} \frac{-\pi^2(i^2+j^2)}{4} C_{ij} \sin\left(\frac{i\pi(x+1)}{2}\right) \sin\left(\frac{j\pi(y+1)}{2}\right) + 1 &= 0, \\ &\Big\downarrow \\ &\text{En vertu de l'orthogonalité des sinus,} \\ &\Big\downarrow \\ \frac{\pi^2(i^2+j^2)}{4} C_{ij} - \underbrace{\int_\Omega \sin\left(\frac{i\pi(x+1)}{2}\right) \sin\left(\frac{j\pi(y+1)}{2}\right) d\Omega}_{16/(ij\pi^2)} &= 0, \end{aligned}$$

On déduit finalement que la solution analytique s'écrit sous la forme suivante :

$$\boxed{u(x,y) = \sum_{i,j \text{ impairs}} \frac{64}{\pi^4(i^2+j^2)ij} \sin\left(\frac{i\pi(x+1)}{2}\right) \sin\left(\frac{j\pi(y+1)}{2}\right)}$$

6.1.1 La méthode des différences finies

Nous allons maintenant essayer d'obtenir une solution approchée par la méthode des différences finies. La première étape consiste à définir une grille ou un maillage uniforme de n^2 points avec un pas $h = 2/(n-1)$ et les coordonnées des noeuds du maillage sont

$$\mathbf{X}_{ij} \;=\; (X_i, Y_j) \;=\; (-1+ih, -1+jh) \qquad\qquad i,j = 0,\ldots,n-1$$

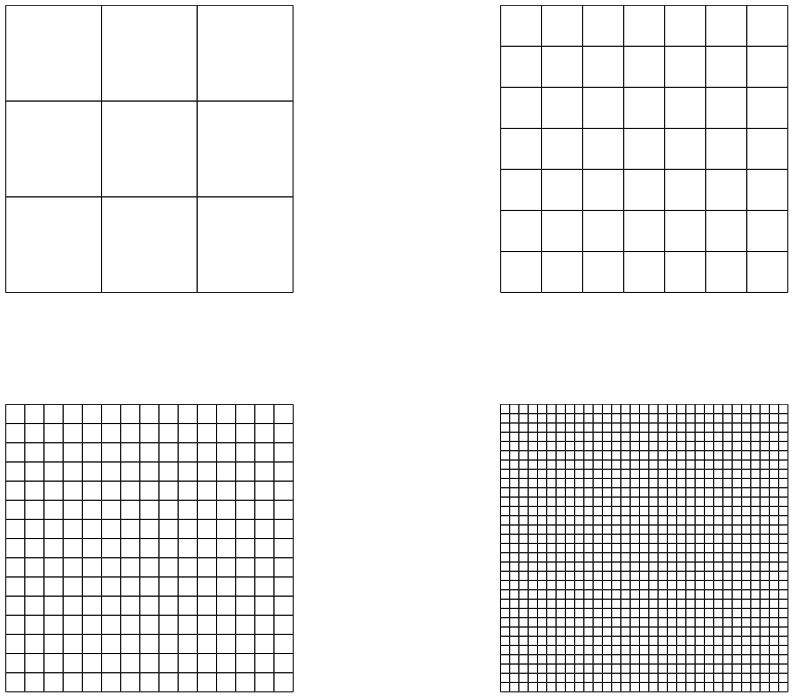


Figure 6.1: Grilles de 4^2, 8^2, 16^2 et 32^2 noeuds.

qui se trouvent dans le domaine Ω. Quatre grilles sont illustrées sur la Figure 6.1. Les valeurs approchées de la fonction inconnue en ces points du maillage seront notées :

$$U_{ij} \;=\; u^h(\mathbf{X}_{ij}) \;\approx\; u(\mathbf{X}_{ij})$$

La seconde étape consiste à approcher les dérivées partielles du laplacien par des différences finies du second ordre afin d'obtenir une approximation discrète du laplacien avec 5 noeuds.

$$\left(\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{h^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{h^2}\right) + 1 = 0$$

$$\downarrow$$

$$\frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j}}{h^2} + 1 = 0$$

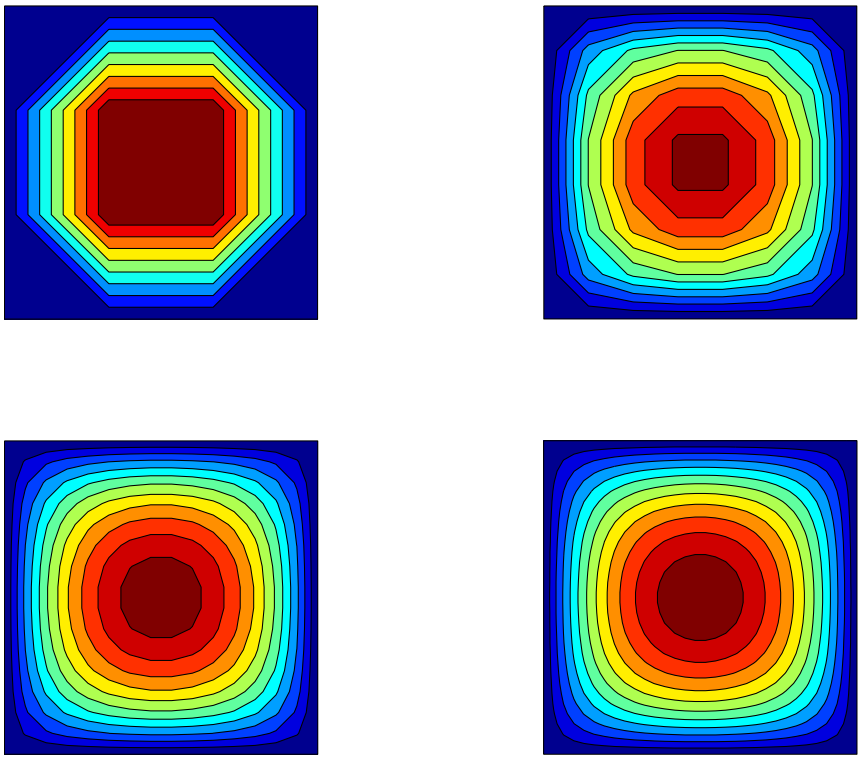


Figure 6.2: Solution par différences finies avec des maillages de 4^2, 8^2, 16^2 et 32^2 noeuds : isolignes de 0 à 0.5 avec un intervalle de 0.025.

On obtient ainsi $(n-2)^2$ équations linéaires à $(n-2)^2$ inconnues en tenant compte que les valeurs nodales sur la frontière sont connues par les conditions aux limites. Il suffira donc de résoudre un grand système linéaire pour obtenir toutes les valeurs nodales. Pour la résolution d'un tel système, on peut faire appel à une méthode directe ou une approche itérative telle que nous l'avons vue précédemment. Ce système linéaire peut être de grande taille mais présente une caractéristique très intéressante : il est creux et peut donc être résolu efficacement par des méthodes adaptées. Les solutions ainsi obtenues sont reprises sur la Figure 6.2 et convergent vers la solution analytique...

Pour estimer l'erreur numérique, le plus simple est de raffiner successivement la grille

et de comparer les erreurs obtenues avec la solution exacte :

$n-1$	h	$e^h = \max \lvert u(X_i, Y_j) - \underbrace{u^h(X_i, Y_j)}_{U_{ij}} \rvert$
3	0.66667	1.9161507e-002
6	0.33332	4.4728014e-003
12	0.16667	1.0188850e-003
24	0.08332	2.4094496e-004

En observant un tel tableau, on constate que l'erreur diminue approximativement d'un facteur quatre, lorsqu'on divise le pas par deux. Cette observation est parfaitement en accord avec la théorie.

Théorème 6.1.

Soit $u(x, y)$ la solution exacte d'un problème de Poisson aux conditions aux limites sur un domaine $\Omega \subset \mathbb{R}^2$.
Si la fonction u et toutes ses dérivées partielles jusqu'au quatrième ordre sont continues sur le domaine fermé $\overline{\Omega}$, alors il existe une constante positive telle que :

$$\max \lvert u(X_i, Y_j) - \underbrace{u^h(X_i, Y_j)}_{U_{ij}} \rvert \leq C \ M \ h^2$$

où M est la valeur maximale atteinte par une des dérivées quatrièmes de u sur $\overline{\Omega}$ et u^h est la solution discrète obtenue au moyen d'un schéma à 5 points basé sur des différences finies centrées du second ordre.

La démonstration (pas évidente) de ce résultat pourra être trouvée dans le livre d'Isaacson et Keller (Analysis of Numerical Methods, 1966).

6.2 Equation de la chaleur

L'extension de la méthode des différences finies à une équation parabolique telle que (6.2) nécessite l'introduction simultanée d'une discrétisation temporelle et spatiale. Nous allons ainsi introduire à chaque pas de temps Δt et pas spatial Δx, une valeur nodale

$$U_i^n = u^h(X_i, T_n) \approx u(X_i, T_n),$$

où i et n sont respectivement les indices spatiaux et temporels. Immédiatement, on peut déduire, à titre d'exemple, que $U_{i+1}^n \approx u(X_i + \Delta x, T_n)$ et $U_i^{n+1} \approx u(X_i, T_n + \Delta t)$. Il est aisé d'utiliser des différences finies avec un maillage de pas spatial Δx et une méthode d'Euler explicite avec un pas temporel Δt.

$$\begin{aligned}
\rho c \frac{\partial u}{\partial t} &= k \frac{\partial^2 u}{\partial x^2} \\
&\downarrow \quad \text{En définissant } \alpha = \frac{k}{\rho c}, \\
\left(\frac{U_i^{n+1} - U_i^n}{\Delta t} \right) &= \alpha \left(\frac{U_{i+1}^n - 2U_i^n + U_{i-1}^n}{(\Delta x)^2} \right) \\
&\downarrow \quad \text{En définissant } \beta = \frac{\alpha \Delta t}{(\Delta x)^2}, \\
U_i^{n+1} &= U_i^n + \beta \Big(U_{i+1}^n + U_{i-1}^n - 2U_i^n \Big)
\end{aligned}$$

Cette dernière expression permet d'obtenir n'importe quelle valeur nodale au temps $n+1$ à partir des valeurs du temps n. Elle définit une itération pour un vecteur qui devrait normalement converger vers la solution de régime. C'est exactement la même procédure itérative que l'on avait déjà utilisée lorsqu'on a abordé l'intégration de systèmes d'équations différentielles ordinaires. En effectuant quelques manipulations symboliques, on peut réécrire l'itération du vecteur d'inconnues spatiales de la manière suivante.

$$
\begin{aligned}
U_i^{n+1} &= U_i^n + \beta\Big(U_{i+1}^n + U_{i-1}^n - 2U_i^n\Big) \\
&\downarrow \quad \text{En passant à une notation matricielle,} \\
\begin{bmatrix} U_1^{n+1} \\ U_2^{n+1} \\ U_3^{n+1} \\ U_4^{n+1} \\ U_5^{n+1} \\ \\ \\ U_m^{n+1} \end{bmatrix} &= \begin{bmatrix} U_1^n \\ U_2^n \\ U_3^n \\ U_4^n \\ U_5^n \\ \\ \\ U_m^n \end{bmatrix} + \beta \begin{bmatrix} -2 & 1 & & & & & & \\ 1 & -2 & 1 & & & & & \\ & 1 & -2 & 1 & & & & \\ & & 1 & -2 & 1 & & & \\ & & & 1 & -2 & 1 & & \\ & & & & & & 1 & -2 \end{bmatrix} \begin{bmatrix} U_1^n \\ U_2^n \\ U_3^n \\ U_4^n \\ U_5^n \\ \\ \\ U_m^n \end{bmatrix} \\
&\downarrow \quad \text{En définissant adéquatement } u_n \text{ et } A, \\
\mathbf{u}_{n+1} &= \mathbf{u}_n + \beta \mathbf{A}\mathbf{u}_n
\end{aligned}
$$

Ce qui correspond exactement à l'itération d'Euler explicite pour le système d'équations différentielles ordinaires linéaires défini par :

$$\mathbf{u}'(t) = \frac{\alpha}{(\Delta x)^2}\mathbf{A}\mathbf{u}(t)$$

6.2.1 Analyse de la stabilité

Physiquement, nous savons que le champ de température doit progressivement converger vers la solution de régime. Mathématiquement, on peut également démontrer le caractère stable du problème continu. Il s'agit maintenant de voir si notre méthode numérique présentera également un comportement stable... Nous avons démontré que la méthode d'Euler explicite n'est que conditionnellement stable. En particulier, la condition suivante doit être satisfaite pour toute valeur propre λ_i de la matrice $\mathbf{A}$:

$$\boxed{|1 + \beta\lambda_i| \leq 1}$$

Evidemment, il faut connaître ses valeurs propres qui dépendent de la taille de la matrice $\mathbf{A}$. Toutefois, un tel calcul peut être effectué numériquement. Sur la Figure 6.3,

nous superposons les valeurs de $\beta\lambda_i$ ainsi obtenues et la zone de stabilité d'Euler explicite dans le plan complexe, pour $m = 6, 11, 21$ et 41 en choisissant Δx et Δt afin que $\beta = 0.5$.

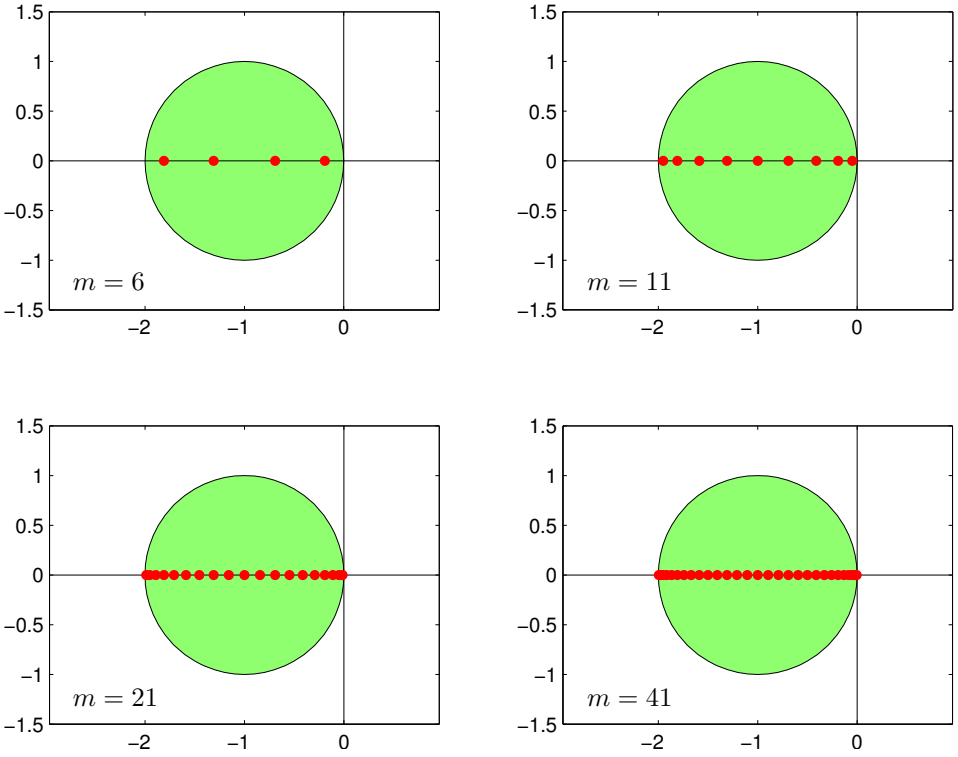


Figure 6.3: Superposition des valeurs de $\beta\lambda_i$ ainsi obtenues et de la zone de stabilité d'Euler explicite dans le plan complexe, pour $m = 6, 11, 21$ et 41 en imposant $\beta = 0.5$.

On observe que toutes les valeurs propres occupent progressivement tout le diamètre sur l'axe réel du cercle définissant la zone de stabilité de la méthode d'Euler explicite. En d'autres mots, on voit qu'il sera nécessaire de maintenir β inférieur à 0.5. Cette contrainte peut devenir extrêmement pénalisante, puisqu'elle impose la relation suivante entre le pas spatial et le pas temporel.

$$\beta = \frac{\alpha \Delta t}{(\Delta x)^2} \leq \frac{1}{2}$$

$$\downarrow$$

$$\Delta t \leq \frac{(\Delta x)^2}{2\alpha}$$

Il s'agit de la *condition CFL* de ce problème. C'est donc une limite sur le pas de

temps pour une intégration temporelle explicite d'une équation aux dérivées partielles. En analyse numérique, une condition de Courant-Friedrichs-Lewy est une contrainte sur le pas de temps pour des méthodes de résolution d'équations aux dérivées partielles afin que ces méthodes convergent et ne produisent pas des résultats totalement erronés. Une telle contrainte a été initialement décrite dans un article de 1928 dû à Richard Courant, Kurt Friedrichs et Hans Lewy. A titre d'illustration, discrétisons un intervalle d'une longueur unitaire avec m points. Pour éviter les instabilités numériques, il faut donc vraiment prendre un très petit pas de temps, lorsqu'on introduit simplement un pas spatial de petite taille. Les pas de temps pour une valeur unitaire de α sont fournis dans la table :

m	Δx	Δt
6	0.2	0.02
11	0.1	0.005
21	0.05	0.00125
41	0.025	0.0003125

Une telle contrainte est particulièrement pénalisante lorsqu'on souhaite intégrer un modèle numérique sur de très longues périodes, par exemple en souhaitant effectuer des prédictions climatiques sur des milliers d'années avec parfois des contraintes de stabilité requérant des pas temporels de l'ordre de la seconde...

Il faut toutefois remarquer que nous avons déduit la condition CFL de manière expérimentale en calculant numériquement les valeurs propres de la matrice $\mathbf{A}$. Il est également possible d'obtenir cette condition en obtenant une estimation du facteur d'amplification d'une perturbation quelconque de la forme suivante :

$$U_j^n = U^n e^{ikX_j}$$

où j et n sont respectivement les indices spatiaux et temporels. Le nombre U^n représente l'amplitude de la perturbation à l'itération temporelle n. On suppose donc une perturbation qui s'écrit sous la forme d'une exponentielle complexe avec un nombre d'onde k quelconque. Le nombre U représente lui le facteur d'amplification de la perturbation puisque $U^{n+1} = U\ U^n$. Notons que l'amplitude et le facteur d'amplification peuvent être des nombres complexes comme nous l'observerons à la fin de ce chapitre.

Cette perturbation évoluera comme suit lors d'un pas de temps :

$$
\begin{aligned}
U_j^{n+1} &= U_j^n + \beta\Big(U_{j+1}^n + U_{j-1}^n - 2U_j^n\Big) \\
&= U_j^n\Big(1 + \beta\Big(e^{ik\Delta x} + e^{-ik\Delta x} - 2\Big)\Big) \\
&= U_j^n\Big(1 + \beta\Big(2\cos(k\Delta x) - 2\Big)\Big)
\end{aligned}
$$

Le facteur d'amplification de la perturbation est donc donné par :

$$
U = \Big(1 + \beta\Big(2\cos(k\Delta x) - 2\Big)\Big)
$$

Pour que la perturbation ne s'accroisse pas, il faut que pour toutes valeurs de k, la valeur absolue du facteur d'amplification reste inférieure à l'unité.

$$
\begin{aligned}
|1 + \beta\Big(2\cos(k\Delta x) - 2\Big)| &\leq 1 \\
&\Big\downarrow \quad \text{En prenant le cas le plus défavorable } k\Delta x = \pi, \\
-1 &\leq 1 - 4\beta \\
2\beta &\leq 1
\end{aligned}
$$

On obtient donc finalement, comme condition de stabilité, l'inégalité de la condition CFL pour les pas de discrétisations spatiale et temporelle.

$$
\boxed{\beta = \frac{\alpha\Delta t}{(\Delta x)^2} \leq \frac{1}{2}}
$$

Il y a un lien extrêmement étroit entre les deux démarches que nous venons de présenter. Nous allons juste l'illustrer pour le cas $m = 11$ et $\beta = 0.5$ ($L = 1$ et $\Delta x = 0.1$). Sur la Figure 6.4, nous représentons $\mathbf{u}$ ainsi que $\beta\mathbf{Au}$ pour les perturbations particulières :

$$
U_j = \Big[\sin(\hat{k}\pi\Delta x), \sin(2\hat{k}\pi\Delta x), \cdots, \sin(9\hat{k}\pi\Delta x)\Big]^T \qquad \hat{k} = 1, \ldots, 9
$$

En calculant le rapport entre $\mathbf{u}$ et $\beta\mathbf{Au}$, on voit que les perturbations sont davantage amorties si le nombre d'onde est petit. C'est bien $\hat{k} = 9$ qui est le cas le plus critique. Ces

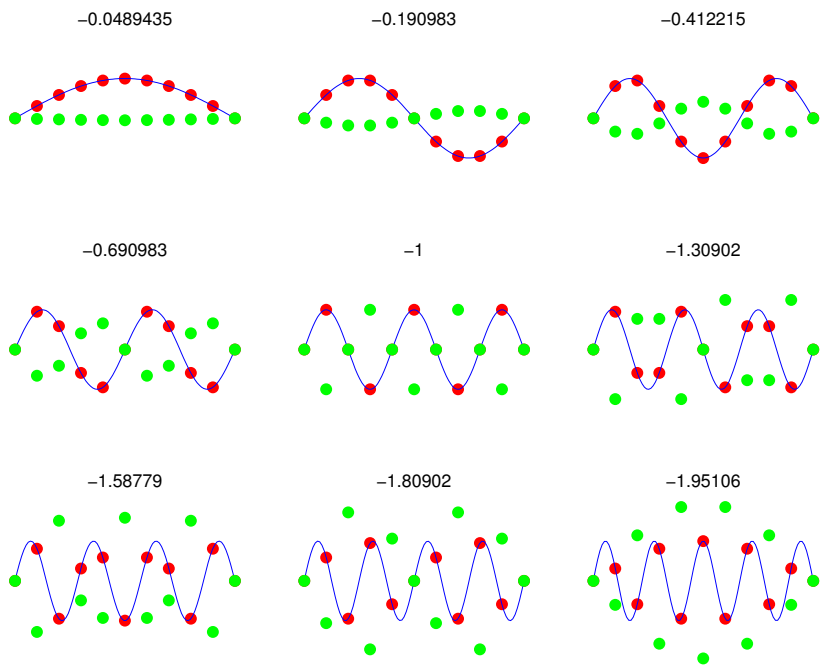


Figure 6.4: Perturbations sinusoïdales $\mathbf{u}$ et $\beta\mathbf{A}\mathbf{u}$ pour $m = 11$ et $\beta = 0.5$. La valeur numérique est le rapport entre les deux vecteurs.

vecteurs $\mathbf{u}$ et ce facteur d'amplification ne sont rien d'autre que les vecteurs et valeurs propres de $\mathbf{A}$. En d'autres mots, n'importe quelle perturbation du vecteur discret peut être écrite comme une combinaison linéaire des vecteurs propres de la matrice $\mathbf{A}$. En effectuant une analyse de stabilité sur la perturbation (ou le vecteur propre) la plus critique, on a tout simplement obtenu la plus grande valeur propre de la matrice $\mathbf{A}$.

Est-il possible de contourner la condition CFL ? Oui, c'est l'avantage des méthodes implicites, mais elles nécessiteront la résolution d'un système linéaire à chaque pas de temps. Ce système pourra être de très grande taille et il faudra donc choisir entre effectuer un très grand nombre de pas explicites peu coûteux et un petit nombre de pas de temps implicites beaucoup plus coûteux en termes de ressources.

6.3 Equation d'onde

Finalement, nous allons appliquer la méthode des différences finies à une équation hyperbolique telle que (6.3). A titre d'exemple physique, considérons une corde tendue vibrante dont la longueur à l'équilibre est L et dont les deux extrémités sont fixées : $u(0,t) = 0$ et $u(L,t) = 0$. La masse de la corde par unité de longueur est ρ et la tension à l'équilibre est donnée par T_0. Le déplacement transversal de la corde par rapport à sa position d'équilibre est donné par $u(x,t)$ et satisfait le problème aux conditions limites :

$$\begin{cases} \rho \dfrac{\partial^2 u}{\partial t^2}(x,t) = T_0 \dfrac{\partial^2 u}{\partial x^2}(x,t), \\ \\ u(0,t) = u(L,t) = 0, \qquad\qquad u(x,0) = u_0(x), \qquad\qquad \dfrac{\partial u}{\partial t}(x,0) = 0. \end{cases}$$

Dans ce cas particulier, nous pourrons comparer une solution analytique avec les solutions numériques. Plus précisément, nous allons utiliser les paramètres matériels et la condition initiale donnés ci-dessous :

$$\begin{array}{rcl} L & = & 1.0\ [m] \\ \rho & = & 0.01\ [kg/m] \\ T_0 & = & 0.1\ [N] \\ \\ u(x,0) & = & u_0 \left(1 - \dfrac{x}{L}\right) \left(\dfrac{x}{L}\right)^2\ [mm] \end{array}$$

La solution du problème est parfaitement périodique dans le temps avec une période $T = 2L/c$. La vitesse de propagation de l'onde est donnée par

$$c = \sqrt{\frac{T_0}{\rho}}.$$

6.3.1 Discrétisations spatiale et temporelle

Pour obtenir une méthode numérique, effectuons tout d'abord une discrétisation spatiale par les différences finies. Nous allons ainsi introduire une fonction à une variable décrivant chaque valeur nodale spatiale

$$U_i(t) = u^h(X_i, t) \approx u(X_i, t), \qquad\qquad i = 1, \ldots, m$$

et leur évolution dans le temps est régie par un système de m équations différentielles ordinaires du second ordre.

$$\begin{array}{rcl} \dfrac{\partial^2 u}{\partial t^2} & = & c^2 \dfrac{\partial^2 u}{\partial x^2} \\ & \downarrow & \\ \dfrac{d^2 U_i}{dt^2} & = & c^2 \dfrac{U_{i+1} - 2U_i + U_{i-1}}{(\Delta x)^2} \end{array}$$

La seconde étape consiste maintenant à effectuer la discrétisation temporelle[2]. Deux options sont possibles :

- Nous pouvons écrire les m équations ordinaires du second degré comme un ensemble de $2m$ équations ordinaires du premier degré.

$$\frac{d^2U_i}{dt^2} = c^2\frac{U_{i+1}-2U_i+U_{i-1}}{(\Delta x)^2}$$

$$\downarrow$$

$$\begin{cases} \dfrac{dU_i}{dt} = V_i \\ \dfrac{dV_i}{dt} = c^2\dfrac{U_{i+1}-2U_i+U_{i-1}}{(\Delta x)^2} \end{cases}$$

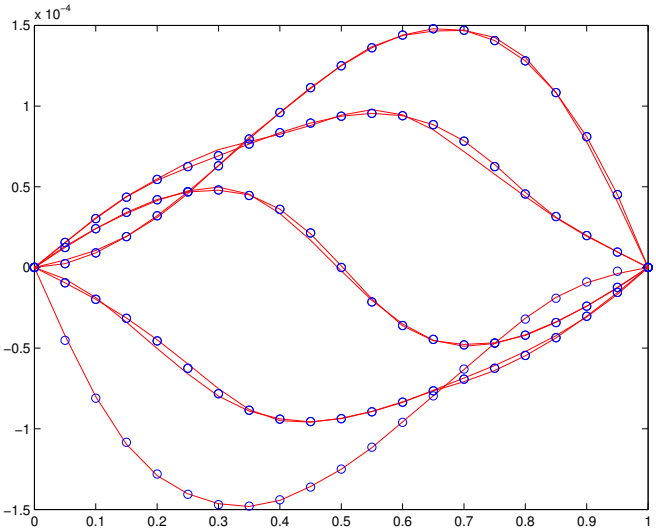


Figure 6.5: Méthode de Runge-Kutta-Felhberg adaptative (ode45) : superposition de la solution numérique calculée sur une période (lignes continues $n = 21$) et de la solution exacte (cercles) pour le problème de la corde vibrante.

Il est dès lors possible d'appliquer toutes les méthodes développées pour les équations différentielles ordinaires. A titre d'exemple, l'application de la méthode d'Euler explicite donne les formules suivantes :

[2]On pourrait parfaitement inverser les deux opérations ou les effectuer de manière simultanée comme nous l'avons fait pour l'équation de la chaleur... Cela ne changerait strictement rien.

$$\begin{cases} \dfrac{U_i^{n+1} - U_i^n}{\Delta t} &= V_i^n \\ \dfrac{V_i^{n+1} - V_i^n}{\Delta t} &= c^2 \dfrac{U_{i+1}^n - 2U_i^n + U_{i-1}^n}{(\Delta x)^2} \end{cases}$$

Sur la Figure 6.5, la solution obtenue par une méthode de Runge-Kutta-Felhberg adaptative (ode45) est présentée pour une période temporelle. Malgré la qualité de la méthode, on observe que la solution numérique ne conserve pas exactement un comportement parfaitement périodique dans le temps.

- Nous pouvons aussi discrétiser directement la dérivée seconde et écrire :

$$\frac{d^2U_i}{dt^2} = c^2 \frac{U_{i+1} - 2U_i + U_{i-1}}{(\Delta x)^2}$$

$$\downarrow$$

$$\frac{U_i^{n+1} - 2U_i^n + U_i^{n-1}}{(\Delta t)^2} = c^2 \frac{U_{i+1}^n - 2U_i^n + U_{i-1}^n}{(\Delta x)^2}$$

Cela fournit une formule d'itération permettant d'obtenir les valeurs au temps $n+1$ à partir de celles aux deux pas de temps précédents. Nous avons donc défini ainsi une méthode d'intégration temporelle à pas liés.

$$\frac{U_i^{n+1} - 2U_i^n + U_i^{n-1}}{(\Delta t)^2} = c^2 \frac{U_{i+1}^n - 2U_i^n + U_{i-1}^n}{(\Delta x)^2}$$

$$\downarrow \quad \text{En définissant } \beta = \frac{c\Delta t}{\Delta x},$$

$$U_i^{n+1} = 2U_i^n + \beta^2 \Big(U_{i+1}^n - 2U_i^n + U_{i-1}^n \Big) - U_i^{n-1}$$

Avec un choix adéquat de β, nous allons observer que cette dernière méthode centrée dans le temps est d'une efficacité optimale. Le choix idéal de β peut être obtenu en effectuant une analyse de stabilité.

6.3.2 Erreurs de dissipation et de dispersion

Considérons, à nouveau, une perturbation quelconque de la forme suivante :

$$U_j^n = U^n e^{ikX_j}$$

où j et n sont toujours les indices spatiaux et temporels. Le nombre U^n représente à nouveau l'amplitude de la perturbation. Cette perturbation évoluera comme suit lors d'un pas de temps :

$$
\begin{aligned}
U_j^{n+1} &= 2U_j^n + \beta^2\Big(U_{j+1}^n - 2U_j^n + U_{j-1}^n\Big) - U_j^{n-1} \\
U^{n+1}e^{ikX_j} &= U^n e^{ikX_j}\Big(2 + \beta^2\Big(e^{ik\Delta x} - 2 + e^{-ik\Delta x}\Big)\Big) - U^{n-1}e^{ikX_j} \\
U^{n+1} &= U^n\Big(2 + \beta^2\Big(2\cos(k\Delta x) - 2\Big)\Big) - U^{n-1} \\
U^{n+1} &= U^n\Big(2 - 4\beta^2\sin^2\left(\tfrac{k\Delta x}{2}\right)\Big) - U^{n-1} \\
\overline{U}^2\ U^{n-1} &= \overline{U}\ U^{n-1}\Big(2 - 4\beta^2\sin^2\left(\tfrac{k\Delta x}{2}\right)\Big) - U^{n-1} \\
&\downarrow \\
\overline{U}^2 &= 2\overline{U}\ \underbrace{\Big(1 - 2\beta^2\sin^2\left(\frac{k\Delta x}{2}\right)\Big)}_{a} - 1
\end{aligned}
$$

Dans le cas de méthodes à pas liés, nous observons à nouveau que l'analyse de stabilité nécessite la résolution d'un polynôme pour obtenir le facteur d'amplification $\overline{U}$. Grâce à cette dernière relation, on voit que le facteur d'amplification $\overline{U}$ est une solution de l'équation du second degré où on définit $a = 1 - 2\beta^2\sin^2(\frac{k\Delta x}{2})$ afin d'alléger les notations.

$$
\begin{aligned}
\overline{U}^2 - 2a\overline{U} + 1 &= 0 \\
&\downarrow \\
\overline{U} &= a \pm \sqrt{a^2 - 1}
\end{aligned}
$$

Pour que la perturbation ne s'accroisse pas, il faut que pour toute valeur de k, le module du facteur d'amplification $\overline{U}$ (qui peut être éventuellement complexe !) reste inférieur à l'unité.

Dans le cas de racines complexes conjuguées, le module de celles-ci vaudra toujours l'unité ($|\overline{U}| = \sqrt{a^2 + 1 - a^2} = 1$). Dans le cas de racines réelles, on observe que le produit des racines vaut l'unité. Imposer que la valeur absolue de ces racines soit inférieure à l'unité, revient à se restreindre au cas unique de la racine double de valeur unitaire. La condition de stabilité revient simplement à exiger que $|a| \leq 1$. Afin de simplifier l'algèbre, il est plus commode d'exiger que $a^2 \leq 1$ et d'écrire :

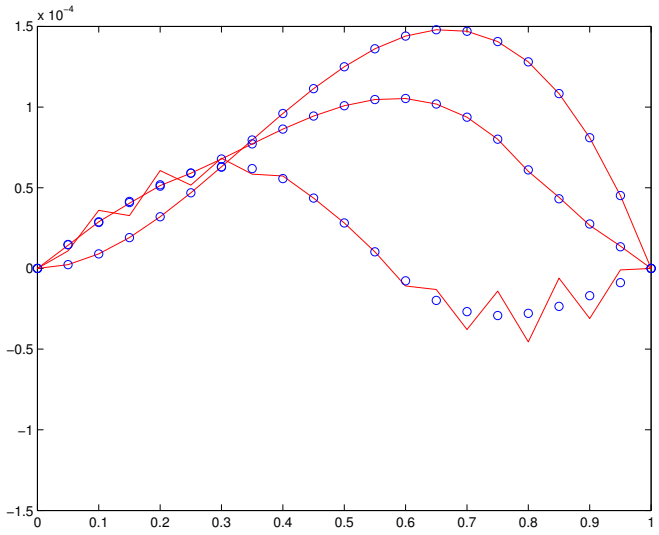


Figure 6.6: Différences finies centrées avec $\beta = 1.1$: développement de l'instabilité (lignes continues $n = 21$) et de la solution exacte (cercles).

$$\left(1 - 2\beta^2 \sin^2(\frac{k\Delta x}{2})\right)^2 \leq 1$$

$$1 - 4\beta^2 \sin^2(\frac{k\Delta x}{2}) + 4\beta^4 \sin^4(\frac{k\Delta x}{2}) \leq 1$$

$$-1 + \beta^2 \sin^2(\frac{k\Delta x}{2}) \leq 0$$

$\downarrow$ En prenant le cas le plus défavorable

$$\beta^2 \leq 1$$

On obtient donc finalement comme condition de stabilité, l'inégalité suivante sur les pas de discrétisations spatiale et temporelle.

$$\boxed{\beta = \frac{c\Delta t}{\Delta x} \leq 1}$$

Dans un tel cas, les perturbations introduiront un comportement oscillant de la solution et ne seront pas dissipées dans le temps : ce qui correspond exactement au comportement physique de l'équation d'onde. Notre schéma sera donc numériquement stable, sans être "trop stable" et sans introduire de dissipation numérique inopportune. Les instabilités numériques prédites par l'analyse de stabilité, sont bien observées en pratique sur la Figure 6.6.

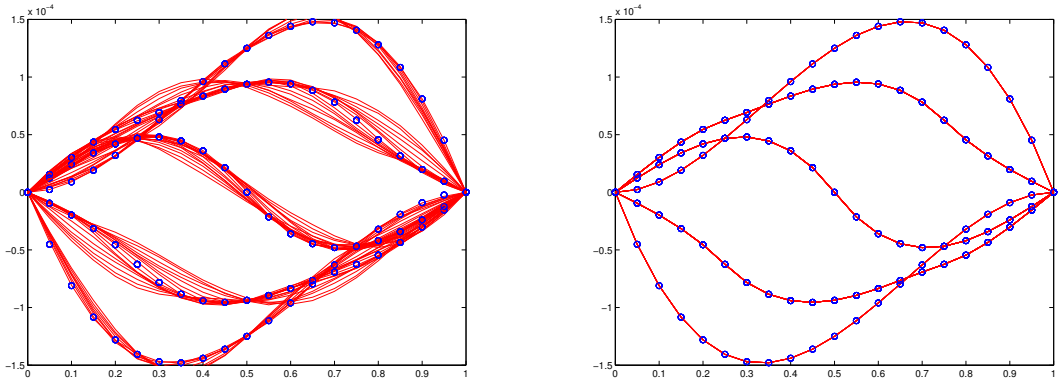


Figure 6.7: Différences finies centrées avec $\beta = 0.5$ (gauche) et $\beta = 1.0$ (droite): superposition de la solution numérique calculée sur huit périodes (lignes continues $n = 21$) et de la solution exacte (cercles).

Il est toutefois utile de remarquer que parmi toutes les valeurs possibles de β, la valeur unitaire permet d'avoir un facteur d'amplification réel exactement égal à l'unité : ce qui correspond à la physique d'une équation d'onde où aucune énergie n'est dissipée. L'introduction d'une plus petite valeur va créer un facteur U complexe de module un qui va créer des erreurs numériques dites de dispersion : l'amplitude des perturbations ne va ni s'accroître, ni diminuer, mais la forme spatiale de la perturbation sera dispersée. En pratique, il faudra donc contrôler à la fois la dissipation et la dispersion des méthodes numériques. Pour des problèmes physiques mettant en jeu des phénomènes ondulatoires, il sera essentiel que ces ondes n'explosent pas artificiellement par une méthode numérique instable, que ces ondes ne soient pas artificiellement étouffées par une méthode numérique trop stable (on parlera de méthode trop dissipative) et aussi que la forme du signal ne soit pas déformée par une méthode numérique trop dispersive. Sur la Figure 6.7, on voit que le choix d'un facteur $\beta = 1$ permet dans le cas particulier d'une équation d'onde unidimensionnelle, d'obtenir une méthode numérique qui n'est ni dissipative, ni dispersive. Hélas, dans les problèmes plus complexes, il n'est pas toujours aussi aisé d'obtenir un tel résultat : l'accumulation d'erreurs de dispersion et de dissipation sur un calcul pendant une longue période de temps mettra évidemment en péril la fiabilité de telles prédictions...