DÉPARTEMENT STPI
TROISIÈME ANNÉE
PRÉORIENTATION ICBE

Analyse Numérique I & II

F. Deluzet, A. Huard, A. Liné, J. Morchain, P. Poncet, G.Quinio, P. Villedieu

2005/2006

Table des matières

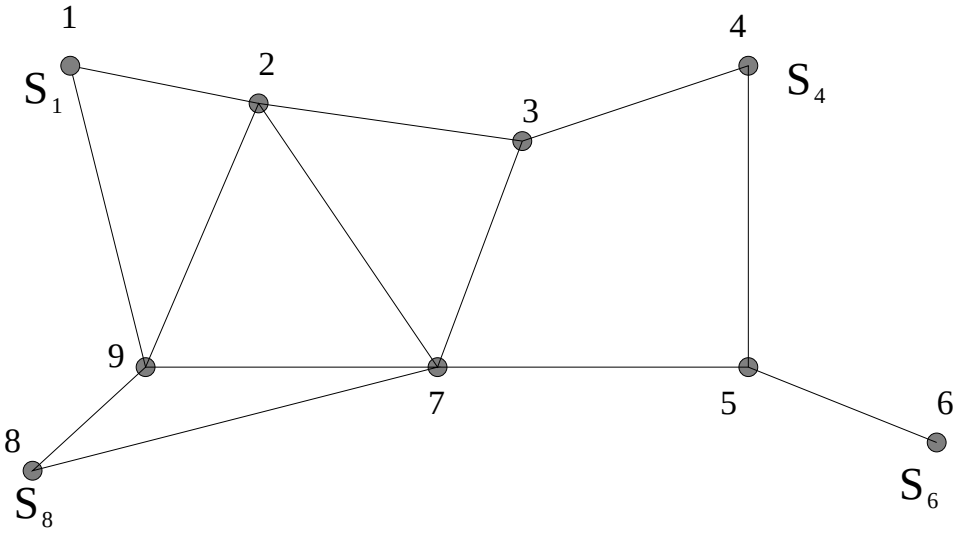
Chapitre 1

Résolution numérique des systèmes linéaires

L'étude des méthodes de résolution des systèmes linéaires est une étape obligatoire d'un cours de calcul scientifique ; presque tous les calculs passent par la résolution de tels systèmes : nous en avons rencontré à l'occasion de problèmes d'interpolation et d'approximation, et l'ingénieur se trouve fréquement confronté à la résolution de tels systèmes.

1.1 Problèmes de réseaux

Un réseau est un ensemble de noeuds P_i et d'arètes $E_{i,j}$ reliant certains de ces noeuds :
- lignes éléctriques,
- canalisations d'eau, égouts,...



Dans chaque arète circule un fluide ; à chaque noeud est associé un potentiel ; l'intensité (ou le débit) du fluide est proportionnelle à la différence de potentiel entre les deux extrémités de l'arète où il circule ; c'est la loi d'Ohm pour les circuits électriques :

$$q_{i,j} = k_{i,j}(u_i - u_j)$$

Une loi physique de conservation (de Kirchoff dans le cas électrique) impose un équilibre : la somme algébrique des intensités en chaque noeud est égale à la valeur de la source (ou du puit) qu'il figure (0 s'il est isolé).

Au noeud P_i, on a dans le cas du circuit électrique :

$$S_i = \sum_j q_{i,j} = \sum_j k_{i,j}(u_i - u_j)$$

Cette somme est étendue aux noeuds P_j adjacents de P_i ; considérons le reseau représenté par la figure ci-dessus. Les équations d'équilibre s'écrivent :

$$\left\{\begin{array}{lcl} S_1 & = & k_{1,2}(u_1-u_2)+k_{1,9}(u_1-u_9) \\ 0 & = & k_{2,1}(u_2-u_1)+k_{2,9}(u_2-u_9)+k_{2,7}(u_2-u_7)+k_{2,3}(u_2-u_3) \\ \cdots & = & \cdots \\ S_8 & = & k_{8,7}(u_8-u_7)+k_{8,9}(u_8-u_9) \\ 0 & = & k_{9,1}(u_9-u_1)+k_{9,2}(u_9-u_2)+k_{9,7}(u_9-u_7)+k_{9,8}(u_9-u_8) \end{array}\right.$$

de sorte que l'équilibre du système est connu en résolvant le système linéaire

$$Au = S$$

avec une matrice A dont les coefficients non nuls sont représentés ci-dessous par une $*$:

$$A = \begin{pmatrix} * & * & & & & & & * \\ * & * & * & & & * & & * \\ & * & * & * & & * & & \\ & & * & * & * & & & \\ & & & * & * & * & * & & \\ & & & & * & * & & & \\ & * & * & & * & & * & * & * \\ & & & & & & * & * & * \\ * & * & & & & & * & * & * \end{pmatrix}$$

Le second membre est défini par $S^T = (S_1, 0, 0, S_4, 0, S_6, 0, S_8, 0)$.

On sait aujourd'hui résoudre assez correctement des systèmes à plusieurs millions d'inconnues (et d'équations), lorsqu'ils sont "assez creux", c'est-à-dire lorsque la matrice du système possède beaucoup de coefficients nuls ; pour les systèmes "pleins", on commence à avoir des problèmes au-delà de quelques centaines de milliers d'inconnues. Cela dépend bien sur des performances des ordinateurs que l'on utilise.

Pour les très grands systèmes, on utilise des méthodes itératives : on construit une suite de vecteurs qui converge vers la solution. Ces méthodes sont adaptées aux grands systèmes creux car elles ne manipulent pas la matrice, mais seulement une fonction qui réalise le produit de cette matrice avec un vecteur quelconque.

Pour les systèmes pleins, on utilise des méthodes directes, susceptibles de fournir la solution en arithmétique exacte après un nombre fini d'opérations élémentaires. Ce sont ces méthodes que nous étudierons dans ce cours.

Les logiciels EXCEL et MATLAB proposent des solveurs de systèmes linéaires qui mettent en oeuvre ce type de méthodes. La disponibilité de ces logiciels ne dispense pas de la connaissance du principe des algorithmes qu'ils mettent en oeuvre, ne serait-ce que pour les utiliser à bon escient, et pour comprendre les indications qui sont fournies, ou les messages d'erreur qui sont renvoyés.

1.2 Sensibilité des systèmes linéaires

Un système comme celui que l'on vient de décrire sera construit à partir de valeurs mesurées. On peut se poser la question de la sensibilité de la solution à d'éventuelles erreurs de mesures.

Plus généralement, le système étant résolu par ordinateur en effectuant un certain nombre d'opérations élémentaires, la solution sera-t-elle très affectée par l'accumulation d'erreurs d'arrondis qu'implique cette succession d'opérations.

1.2.1 Exemple

On considère le système linéaire :

$$A\,x = \begin{pmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0.217 \\ 0.254 \end{pmatrix} = b \tag{1.1}$$

dont la solution est $x = (1,\ -1)^T$.

La solution approchée $\widetilde{x^1} = (0.999,\ -1)^T = x + \Delta_1 x$ peut sembler satisfaisante au vu de sa faible différence avec la solution exacte ; elle fournit le résidu :

$$r^1 = A\,\widetilde{x^1} - b = \begin{pmatrix} -0.00078 \\ -0.00091 \end{pmatrix}$$

Le vecteur $\widetilde{x^2} = (0.341,\ -0.087)^T = x + \Delta_2 x$ ne semble pas être un candidat raisonnable pour la résolution de ce système ; et pourtant, cette fois, le résidu est bien plus satisfaisant :

$$r^2 = A\,\widetilde{x^2} - b = \begin{pmatrix} -0.000001 \\ 0 \end{pmatrix}$$

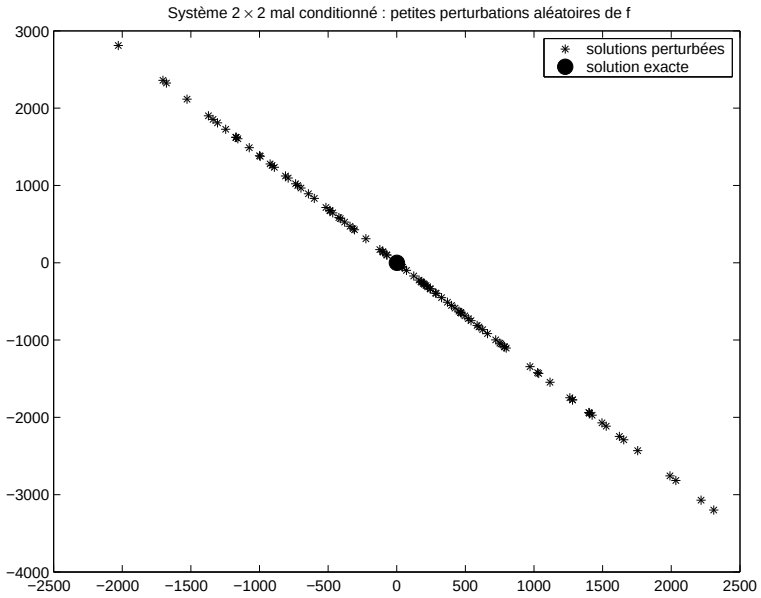
En terme de résidus et d'erreur, on a obtenu, respectivement pour x_1 et x_2 :

$\|\Delta x\|_2$	0.001	1.126
$\|r\|_2$	0.0012	0.000001

On constate ici que dans un cas, une modification très légère de la solution fournit un résidu du même ordre de grandeur, mais dans l'autre une modification assez importante de cette solution fournit un résidu très petit.

Une autre expérience consiste à modifier légèrement le second membre. Considérons le vecteur $\Delta b = \begin{pmatrix} -0.7603 \\ 0.6496 \end{pmatrix}$ et résolvons $A\,\widetilde{x^3} = b + 10^{-3}\,\Delta b$. Cette fois la solution est approximativement $\begin{pmatrix} -865.76 \\ 1199.84 \end{pmatrix}$ soit en notant $\Delta_3 x = x - \widetilde{x^3}$, $\|\Delta_3 x\|_2 \simeq 1481$.

La figure suivante représente la solution exacte du système $A\,x = b$ et les solutions obtenues pour 100 perturbations aléatoires du second membre, toutes inférieures en norme euclidienne à 0.0029.



On peut imaginer que le second membre est le résultat d'une mesure, ou de calculs précédents : accepteriez vous, par exemple, d'être le premier passager d'un avion en sachant que sa conception est passée par la résolution de ce système !

Le problème de la résolution d'un système linéaire n'est pas toujours un problème facile. On voit déja dans cet exemple très simple que l'on peut se poser la question de savoir si l'on cherche le vecteur solution du système, ou un vecteur qui vérifie le système. A précision donnée, les réponses peuvent être très différentes !

1.2.2 Distance à la singularité

On dit que la matrice A est **singulière** lorsque les systèmes linéaires de matrice A ne sont pas inversibles. Une matrice $A \in M_N(\mathbb{R})$ sera donc singulière si l'une ou l'autre des conditions équivalentes suivantes est vérifiée :

- le déterminant de A est nul,
- une des valeurs propres de A est nulle,
- le rang de A est strictement inférieur à N : c'est le cas si les lignes (ou les colonnes) de A ne sont pas linéairement indépendantes.

Une interprétation géométrique de l'exemple

On constate sur la figure précédente que toutes les solutions perturbées semblent se trouver sur une même droite. Un système linéaire 2×2 s'écrit sous la forme :

$$\left\{ \begin{array}{rcrcl} a_{1,1}\,x_1 & + & a_{1,2}\,x_2 & = & b_1 \\ a_{2,1}\,x_1 & + & a_{2,2}\,x_2 & = & b_2 \end{array} \right.$$

Chacune des équations est l'équation d'une droite du plan $(x_1,\, x_2)$. La solution est donnée par les coordonnées du point d'intersection de ces deux droites.

Pour notre exemple, les coefficients directeurs de ces deux droites sont respectivement -1.3854351 et -1.3854324 ! Elles sont donc presque parallèles. Il n'est pas étonnant que la localisation du point d'intersection soit délicate.

Si l'on modifie par exemple f_1, la première droite est remplacée par une droite qui lui est parallèle. Si les deux droites sont presque parallèles, le nouveau point d'intersection est très éloigné du précédent. La figure 1.2.2 illustre ce phénomène pour deux droites dont les coefficients directeurs sont -1 et $-\frac{10}{9}$, c'est-à-dire quand même assez différents, et qui se coupent en un point P. La droite en pointillé est parallèle à l'une des deux droites ; on a modifié la première composante du second membre pour l'obtenir. Son intersection avec la droite inchangée est le point Q.

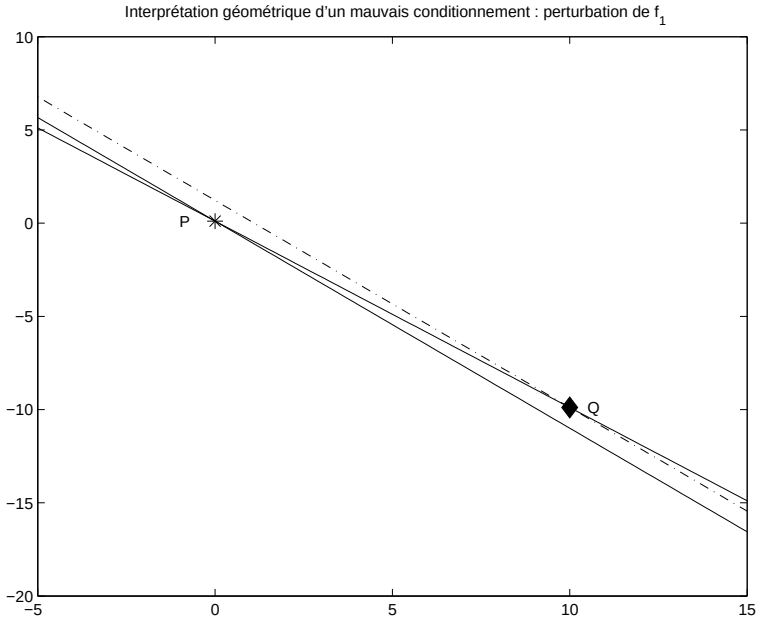


FIG. 1.1 – Intersection de deux droites presque parallèles

Sensibilité et distance à la singularité

Le système pécédent est sensible aux perturbations. Comme on obtient un système équivalent en multipliant chaque ligne par un scalaire non nul, on peut le remplacer par le système obtenu en remplaçant chaque équation $(Eq : i)$ par $\frac{1}{a_{i,1}}(Eq : i)$

$$\begin{pmatrix} 1 & 0.7217948 \\ 1 & 0.7217962 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0.2782051 \\ 0.2782037 \end{pmatrix} \tag{1.2}$$

On voit ici que les lignes `A(1, :)` et `A(2, :)` sont presque dépendantes puisque

$$\texttt{A}(1,:) - \texttt{A}(2,:) = [\texttt{0},\ \texttt{1.4}\,\texttt{10}^{-\texttt{6}}].$$

A vrai dire, (1.2) est une autre façon d'écrire l'équation des deux droites mentionnées précédemment. On voit mieux que ces droites sont presque parallèles.

Lorsque les lignes d'une matrice ne sont pas indépendantes, cette matrice est non inversible. On peut donc penser, que d'une certaine façon, c'est parce que la matrice A est proche d'une matrice singulière que le système est très sensible aux perturbations.

C'est effectivement une matrice proche de la singularité que l'on a utilisé. La plus petite valeur propre de A est à peu près 0.000000694. La matrice $A + \Delta A$ ci-dessous est singulière ; la perturbation ΔA qui la rend singulière est très petite.

$$A + \Delta A = \begin{pmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{pmatrix} + \begin{pmatrix} 0 & 0.00000109529025 \\ 0 & 0 \end{pmatrix}$$

Remarque 1 *Dans $\mathbb{R}$, le seul élément singulier est 0, de sorte que $|x|$ mesure la distance du réel x à la singularité. Dans $M_n(\mathbb{R})$, les matrices de la forme $\epsilon\, I$, où I désigne la matrice identité, sont d'inverse $\dfrac{1}{\epsilon}\, I$, et restent facilement inversibles tant que $\epsilon \neq 0$. Par contre des matrices, comme la matrice $A + \delta A$ ci-dessus, qui ne semblent pas particulièrement proches de 0 peuvent être non inversibles.*

1.2.3 Mesurer la distance à la singularité

Une matrice est singulière lorsqu'elle n'est pas inversible. Il existe plusieurs façons de caractériser cela. $A \in M_n(\mathbb{R})$ est singulière si :

- son determinant est nul ; cela ne peut pas nous être d'une grande utilité, puisque par exemple, la matrice $0.5\, I_{100}$ qui est loin d'être singulière a un determinant à peu près égal à 7.8886×10^{-31},
- une de ses valeurs propres est nulle ; le calcul des valeurs propres n'est pas un problème très simple, et cette caractérisation n'est pas utilisable en pratique
- son rang est strictement inférieur à n ; dans ce cas, son noyau n'est pas réduit à 0, et il existe un vecteur x non nul tel que $A\, x = 0$.

Nous allons chercher à nous appuyer sur cette dernière caractérisation. Mais, il convient de réfléchir un peu si on veut le faire. Pour un vecteur x d'une grandeur donnée, mesurée par sa norme $\|x\|$, le vecteur $A\, x$ pourra être d'une grandeur très variable. La remarque 1 nous laisse penser que lorsque $\dfrac{\|A\, x\|}{\|x\|}$ reste constant, l'inversion de la matrice n'est pas toujours un problème numériquement difficile.

Nous pouvons alors associer à une matrice A donnée les deux nombres

$$\begin{array}{rcccl} \mathrm{a}(A) & = & \max\limits_{\|x\|=1} \|A\, x\| & = & \max\limits_{x \neq 0} \dfrac{\|A\, x\|}{\|x\|} \\ \mathrm{r}(A) & = & \min\limits_{\|x\|=1} \|A\, x\| & = & \min\limits_{x \neq 0} \dfrac{\|A\, x\|}{\|x\|} \end{array} \tag{1.3}$$

Ces deux nombres s'appelleront coefficients d'agrandissement et de réduction de la matrice A relativement à la norme considérée. Leurs deux expressions sont égales par définition des normes..

L'ensemble des $x \in \mathbb{R}^2$ tels que $\|x\|_2 = 1$ est le cercle unité. Si $A \in M_2(\mathbb{R})$, l'image du cercle unité par A est une ellipse. La figure 1.2 montre ces images dans le cas de la matrice $M = \begin{pmatrix} 0.5 & 1 \\ 0.75 & 0.5 \end{pmatrix}$ et dans le cas de la matrice A de (1.1).

La partie droite montre 2 vecteurs qui réalisent l'agrandissement et la réduction de la matrice M pour la norme euclidienne. Cette matrice n'est pas particulièrement proche d'une matrice singulière, et l'ellipse n'a rien de vraiment particulier.

Par contre, on constate sur la partie gauche de cette figure que l'ellispe associée à la matrice A est très aplatie. On a vu que cette matrice est assez proche d'une matrice singulière.

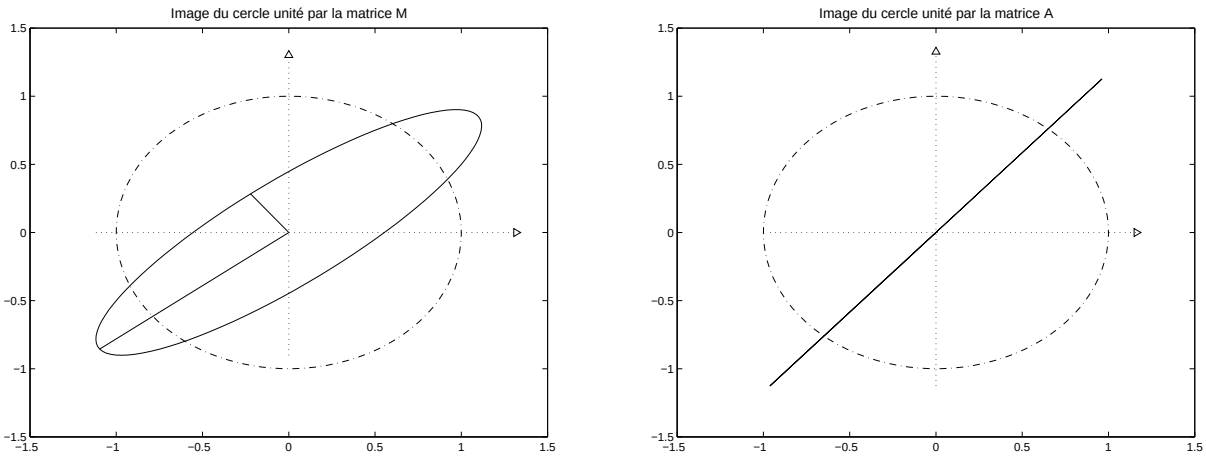


FIG. 1.2 – Images du cercle unité par les matrices M et A

Si une matrice n'est pas inversible, , on aura $A\,u = 0$ pour au moins 2 vecteurs u et $-u$ du cercle et l'ellipse sera complètement aplatie. Mais, plus que l'agrandissement ou la réduction, c'est le rapport de ces deux nombres qui caractérise l'aplatissement de l'ellipse, et on peut penser à utiliser ce rapport $\dfrac{\mathrm{r}(M)}{\mathrm{a}(M)}$ pour évaluer la distance d'une matrice à la singularité. Ce rapport indique un changement de nature géométrique qui illustre la perte de rang caractéristique de la singularité ; on passe d'une surface à un segment. Lorsque ϵ devient très petit, l'image du cercle unité par la matrice $\epsilon\, I$ reste un cercle, et il suffit de changer d'échelle pour retrouver la surface initiale.

Remarque 2 *L'utilisation de la norme euclidienne n'est pas indispensable. On peut aussi bien utiliser une autre norme. La figure 1.3 montre les images de la boule unité pour la norme $\|.\|_\infty$, qui est un carré, par les matrices M et A.*

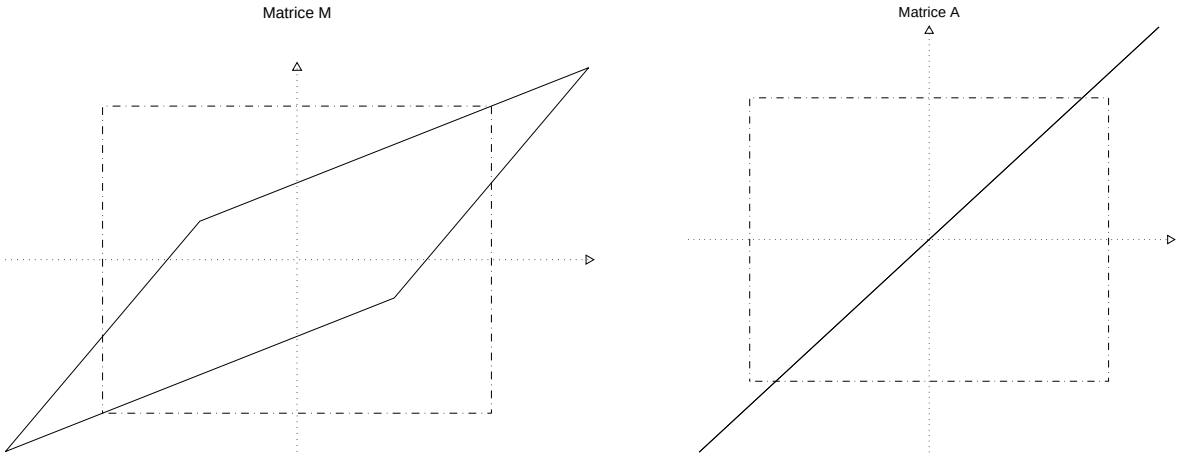


FIG. 1.3 – Images du carré unité par les matrices M et A

1.2.4 Conditionnement

Conditionnement d'une matrice

C'est plus précisément le conditionnement d'une matrice pour la résolution d'un système linéaire que l'on veut étudier. Soit à résoudre le système linéaire $Ax = b$. On suppose que la calcul

de la solution est fait par un algorithme quelconque implanté sur un ordinateur. Notons $\tilde{x} = x + \delta x$ la solution obtenue.

On va supposer que $x+\delta x$ est la solution exacte dans $\mathbb{R}^n$ d'un problème perturbé $A\,(x + \delta x) = b + \delta b$. On a ainsi choisi l'ensemble des perturbations admissibles, qui ne portent que sur le second membre ; d'autres choix sont possibles.

Comme $A\,x = b$ et $A\,\delta x = \delta b$, les définitions (1.3) nous fournissent :

$$\begin{array}{rcl} \|b\| & \leqslant & \mathrm{a}(A)\,\|x\| \\ \|\delta b\| & \geqslant & \mathrm{r}(A)\,\|\delta x\| \end{array}$$

d'où l'on tire, si A est inversible

$$\frac{\|\delta x\|}{\|x\|} \leqslant \frac{\mathrm{a}(A)}{\mathrm{r}(A)}\,\frac{\|\delta b\|}{\|b\|} \tag{1.4}$$

On peut donc d'éfinir $\mathcal{K}(A) = \dfrac{\mathrm{a}(A)}{\mathrm{r}(A)}$, avec $\mathcal{K}(A) = \infty$ si A est singulière. C'est l'inverse de la distance à la singularité de la matrice A.

Conditionnement et normes matricielles

La notion de norme définie pour les vecteurs de $\mathbb{R}^N$ s'étend aux matrices, en posant,

$$\|A\| = \max_{\|x\|=1} \{\|Ax\|\} \tag{1.5}$$

ce qui permet de vérifier, pour tout $x \in \mathbb{R}^N$:

$$\|Ax\| \leqslant \|A\| \times \|x\| \tag{1.6}$$

L'agrandissement $\mathrm{a}(A)$ n'est autre que $\|A\|$. Par ailleurs, si la matrice A est inversible, elle définit une bijection sur $\mathbb{R}^n$, de sorte que

$$\mathrm{r}(A) = \min_{x\neq 0} \frac{\|A\,x\|}{\|x\|} = \min_{y\neq 0} \frac{\|y\|}{\|A^{-1}\,y\|} = \frac{1}{\displaystyle\max_{y\neq 0} \frac{\|A^{-1}\,y\|}{\|y\|}} = \frac{1}{\|A^{-1}\|}.$$

On obtient alors

$$\mathcal{K}(A) = \|A\,\|\,\left\|A^{-1}\right\| \tag{1.7}$$

$\dfrac{\|\delta b\|}{\|b\|}$ mesure ici l'erreur inverse relative. $\mathcal{K}\,(A) = \|A\|\,\left\|A^{-1}\right\|$ s'appelle conditionnement de la matrice A, relativement à la norme choisie, pour la résolution des systèmes linéaires.

Remarque 3 *Les matrices A et A^{-1} jouent un rôle analogue dans (1.6). Le nombre $\mathcal{K}\,(A)$ est en fait aussi le nombre de conditionnement de A pour les produits matrice-vecteur !*

On a considéré que la matrice A était fixée, et que seul le second membre b pouvait être perturbé. On peut montrer que le conditionnement en fonction de perturbations de la matrice A est le même.

De façon générale, ce nombre $\mathcal{K}(A)$ sera tel que

$$\frac{\|\Delta x\|}{\|x\|} \leqslant \mathcal{K}(A) \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|} \right) \tag{1.8}$$

Essayons de vérifier la pertinence de ce nombre de conditionnement sur notre exemple. Nous pouvons utiliser la norme du maximum.

Pour $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, on aura $A\,x = \begin{pmatrix} 0.780\,x_1 + 0.563\,x_2 \\ 0.913\,x_2 + 0.659\,x_2 \end{pmatrix}$, de sorte

$$\|A\,x\|_\infty = \max\left\{|0.780\,x_1 + 0.563\,x_2|,\ |0.913\,x_2 + 0.659\,x_2|\right\}.$$

Si on impose $\max\{|x_1|,\ |x_2|\} = 1$, le maximum de $\|A\,x\|_\infty$ sera atteint pour le vecteur x tel que $x_1 = x_2 = 1$, et il vaut $\|A\|_\infty = 1.572$.

Par contre la matrice A^{-1} est donnée par $A^{-1} = 10^5 \times \begin{pmatrix} 6.59 & -5.63 \\ -9.13 & 7.80 \end{pmatrix}$, et par un raisonnement analogue, $\|A^{-1}\|_\infty = 16.93 \times 10^5$. On a donc un conditionnement de la matrice donné par $K_\infty = 2.661396 \times 10^6$.

Appelons u la solution du système $A\,u = \begin{pmatrix} 1.343 \\ 1.572 \end{pmatrix}$, et observons

$$A\,(u + \Delta u) = \begin{pmatrix} 2.565 \\ -0.121 \end{pmatrix}.$$

Ce second membre est tel que $\dfrac{\|\Delta b\|}{\|b\|} = \dfrac{1.6930}{1.572} \simeq 1.077$.

La solution du premier système est $u \simeq \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, mais $u + \Delta u \simeq 10^6 \times \begin{pmatrix} 1.7585 \\ -2.4362 \end{pmatrix}$.

L'inégalité (1.8) est bien vérifiée :

$$10^6 \times 2.4362 \leqslant 2.661396 \times 10^6 \times 1.077.$$

Remarque 4 *Comme on l'a déjà vu, on préfère souvent travailler avec la norme euclidienne. On peut montrer que la norme matricielle associée est définie par*

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T\,A)} \tag{1.9}$$

où $\lambda_{\max}(A^T\,A)$ désigne la plus grande valeur propre de la matrice $A^T\,A$.

1.3 Factorisation LU

1.3.1 Comment résoudre les systèmes linéaires ?

Les systèmes faciles à résoudre

Il existe des systèmes linéaires qui sont de résolution facile. Commençons par en examiner la résolution.

Systèmes diagonaux Si A est une matrice diagonale, $A = (a_{i,j})_{1 \leqslant i,j \leqslant n}$ avec $a_{i,j} = 0$ si $i \neq j$, le système $Ax = b$ est immédiatement résolu en posant pour tout i, $1 \leqslant i \leqslant n$, $x_i = \dfrac{b_i}{a_{i,i}}$.

Le coût de cette résolution est $c(n) = n$ opérations élémentaires.

Systèmes triangulaires Les systèmes de matrice triangulaire sont également de résolution facile. Examinons le cas de systèmes triangulaires supérieurs $T\,x = b$. La matrice T est telle que $t_{i,j} = 0$ si $i > j$.

Prenons l'exemple du système

$$\begin{pmatrix} 4 & 5 & 1 \\ 0 & 3 & 4 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 4 \\ 1 \\ 2 \end{pmatrix}. \tag{1.10}$$

Ce genre de système se résoud par une méthode dite de **substitution rétrograde** ou de **remontée**. On commence par calculer la dernière composante, on utlise sa valeur pour calculer l'avant dernière, et on recommence selon le même principe jusqu'a la première.

On calcule successivement

$$\begin{aligned} x_3 &= \frac{2}{2} &&= 1 \\ x_2 &= \frac{1 - 4 \times 1}{3} &&= -1 \\ x_1 &= \frac{4 - 5 \times (-1) - 1 \times 1}{4} &&= 2 \end{aligned}$$

Cet algorithme "remonte" la matrice T ligne par ligne. Les instructions MATLAB ci-dessous mettent en oeuvre cet algorithme

```
x(n) = b(n)/T(n,n);
for k = n-1:-1:1
  x(k) = (b(k) - T(k,k+1:n)*x(k+1:n))/T(k,k);
end;
```

On peut évaluer le nombre d'opérations nécessaires à la résolution d'un système triangulaire en fonction de la taille n du système :

- on effectue 1 division pour `x(n) = b(n)/T(n,n);`
- pour chaque valeur de k comprise entre 1 et N-1, on effectue
  $n - k$ multiplications et $n - k - 1$ additions pour le produit
  `T(k,k+1 :n)*x(k+1 :n)`

1 soustratcion et 1 division pour

```
x(k) = (b(k) - T(k,k+1 :n)*x(k+1 :n))/T(k,k);
```

Cela fait donc $2\,(n-k)+1$ opérations, et comme k varie de 1 à $n-1$, $n-k$ varie également de 1 à $n-1$: on obtient $2\,\dfrac{(n-1)\,n}{2}+n-1=n^2-1$ opérations auxquelles on ajoute la première division pour obtenir le coût $c(n)=n^2$ opérations élémentaires.

1.3.2 L'idée des méthodes directes

Transformer le système en un système diagonal, c'est diagonaliser A. Quand c'est possible, c'est le problème de recherche des éléments propres de la matrice A, et c'est beaucoup plus difficile que la résolution du système linéaire ! On ne peut pas espérer s'en sortir de cette façon.

Par contre on peut espèrer transformer notre système en un système triangulaire. En fait, on va remplacer la résolution de notre système par la résolution de deux systèmes triangulaires, à l'aide d'une factorisation de la matrice A de la forme :

$$A = L\,U$$

où L est une matrice triangulaire inférieure (**L**ower) et U une matrice triangulaire supérieure (**U**pper).

1.3.3 De l'élimination de Gauss à la factorisation LU

Elimination de Gauss

Soit à résoudre le système $A\,x=b$, où la matrice A appartient à $M_n(\mathbb{R})$. C'est un système de n équations, notées $(Eq.1),\ldots,(Eq.n)$, à n inconnues notées $x_1,\ldots,x_n$, qui sont les composantes du vecteur x.

Le principe de la méthode est d'éliminer les inconnues de proche en proche, en faisant disparaitre :

- l'inconnue x_1 des équations $(Eq.2)$ à $(Eq.n)$,
- l'inconnue x_2 des équations $(Eq.3)$ à $(Eq.n)$,
- ... et pour k variant de 1 à n, l'inconnue x_k des équations $(Eq.k+1)$ à $(Eq.n)$.

Pour ce faire, on transforme à chaque étape le système en un système équivalent, c'est-à-dire admettant exactement le même ensemble de solutions, au moyen d'une **transformation élémentaire** :

- ajout d'un multiple d'une équation à une équation donnée : cela consiste par exemple à remplacer l'équation $(Eq.i)$ par l'équation $(Eq.i)+\alpha\,(Eq.k)$
- permutation de deux équations,

A l'étape k, le coefficient $a_{k,k}^{(k)}$ de l'inconnue x_k s'appelle le **pivot**. On le note p_k. Il permet l'élimination de l'inconnue x_k des équations $(Eq.k+1)$ à $(Eq.n)$, en remplaçant l'équation $(Eq.i)$ par $(Eq.i)-\dfrac{a_{i,k}^{(k)}}{p_k}\,(Eq.k)$ pour $k+1\leqslant i\leqslant n$.

On va d'abord s'intéresser à la méthode de Gauss sans permutation des équations. Le pivot intervient en dénominateur d'une fraction : il doit être non nul.

Exemple

On considère le système

$$\left\{\begin{array}{rrrcl} 2x_1 & +x_2 & +x_3 & = & 1 \\ 6x_1 & +2x_2 & +x_3 & = & -1 \\ -2x_1 & +2x_2 & +x_3 & = & 7 \end{array}\right. \iff \left(\begin{array}{ccc} 2 & 1 & 1 \\ 6 & 2 & 1 \\ -2 & 2 & 1 \end{array}\right)\left(\begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array}\right) = \left(\begin{array}{c} 1 \\ -1 \\ 7 \end{array}\right) \qquad (1.11)$$

On note $A^{(1)} = A$ et $b^{(1)} = b$ ces données initiales : ce système s'écrit donc $A^{(1)}\,x = b^{(1)}$. On peut choisir comme pivot le coefficient $p_1 = 2$ de x_1 dans la première équation. On élimine x_1 des équations $(Eq.2)$ et $(Eq.3)$ par la transformation élémentaire :

- $(Eq.2)$ est remplacée par $(Eq.2) - \dfrac{6}{p_1}\,(Eq.1) = (Eq.2) - 3\,(Eq.1)$,
- $(Eq.3)$ est remplacée par $(Eq.3) - \dfrac{-2}{p_1}\,(Eq.1) = (Eq.3) + (Eq.1)$.

On obtient le système équivalent :

$$\left\{\begin{array}{rrrrrcl} 2x_1 & + & x_2 & + & x_3 & = & 1 \\ & - & x_2 & - & 2x_3 & = & -4 \\ & + & 3x_2 & + & 2x_3 & = & 8 \end{array}\right. \iff \left(\begin{array}{ccc} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 3 & 2 \end{array}\right)\left(\begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array}\right) = \left(\begin{array}{c} 1 \\ -4 \\ 8 \end{array}\right)$$

On note $A^{(2)}\,x = b^{(2)}$ ce deuxième système, équivalent au premier. On peut choisir comme pivot le coefficient $p_2 = -1$ de x_2 dans la deuxième équation. On élimine x_2 de l'équation $(Eq.3)$:

- $(Eq.3)$ est remplacée par $(Eq.3) - \dfrac{3}{p_2}\,(Eq.2) = (Eq.2) + 3\,(Eq.2)$.

On obtient le système équivalent :

$$\left\{\begin{array}{rrrrrcl} 2x_1 & + & x_2 & + & x_3 & = & 1 \\ & - & x_2 & - & 2x_3 & = & -4 \\ & & & - & 4x_3 & = & -4 \end{array}\right. \iff \left(\begin{array}{ccc} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & -4 \end{array}\right)\left(\begin{array}{c} x_1 \\ x_2 \\ x_3 \end{array}\right) = \left(\begin{array}{c} 1 \\ -4 \\ -4 \end{array}\right)$$

Ce dernier système, $A^{(3)}\,x = b^{(3)}$ est triangulaire supérieur. Il est inversible puisque les éléments diagonaux de $A^{(3)}$, $p_1,\ p_2$ et $p_3 = -4$ sont tous non nuls. On le résoud par substitution rétrograde pour obtenir $x = \left(\begin{array}{c} -1 \\ 2 \\ 1 \end{array}\right)$.

Ecriture matricielle des transformations élémentaires

A la première étape de l'exemple (1.11), la matrice $A^{(1)} = \left(\begin{array}{ccc} 2 & 1 & 1 \\ 6 & 2 & 1 \\ -2 & 2 & 1 \end{array}\right)$ est remplacée par la matrice $A^{(2)} = \left(\begin{array}{ccc} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 3 & 2 \end{array}\right)$. En termes de matrice, le fait de modifier une équation correspond a la modification d'une ligne. On constate en effet, en utilisant la notation MATLAB, que :

$$\begin{array}{rcl} \mathtt{A}^{(2)}(1,:) & = & \mathtt{A}^{(1)}(1,:), \\ \mathtt{A}^{(2)}(2,:) & = & \mathtt{A}^{(1)}(2,:) - 3\,\mathtt{A}^{(1)}(1,:), \\ \mathtt{A}^{(2)}(3,:) & = & \mathtt{A}^{(1)}(3,:) - (-1)\,\mathtt{A}^{(1)}(1,:). \end{array}$$

Regroupons ces lignes pour écrire les matrices correspondantes :

$$\begin{bmatrix} \mathtt{A}^{(2)}(1,:) \\ \mathtt{A}^{(2)}(2,:) \\ \mathtt{A}^{(2)}(3,:) \end{bmatrix} = \begin{bmatrix} \mathtt{A}^{(1)}(1,:) \\ \mathtt{A}^{(1)}(2,:) \\ \mathtt{A}^{(1)}(3,:) \end{bmatrix} - \begin{bmatrix} \mathtt{0} \\ \mathtt{3\,A}^{(1)}(1,:) \\ -\,\mathtt{A}^{(1)}(1,:) \end{bmatrix} \tag{1.12}$$

Le second terme du membre de droite de (1.12) s'écrit :

$$\begin{bmatrix} \mathtt{0} \\ \mathtt{3\,A}^{(1)}(1,:) \\ -\,\mathtt{A}^{(1)}(1,:) \end{bmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 3a_{1,1}^{(1)} & 3a_{1,2}^{(1)} & 3a_{1,3}^{(1)} \\ -a_{1,1}^{(1)} & -a_{1,2}^{(1)} & -a_{1,3}^{(1)} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 3 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} A^{(1)}. \tag{1.13}$$

de sorte que finalement, en notant I la matrice identité :

$$A^{(2)} = \left(I - \begin{pmatrix} 0 & 0 & 0 \\ 3 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \right) A^{(1)} \tag{1.14}$$

Cette écriture se simplifie en introduisant les vecteurs $l^1 = (0,\ 3,\ -1)^T$ et $e_1 = (1,\ 0,\ 0)^T$, premier vecteur de la base canonique :

$$A^{(2)} = (I - l^1\, e_1^T)\, A^{(1)} = E^{(1)}\, A^{(1)}. \tag{1.15}$$

De façon analogue, le passage de $A^{(2)}$ à $A^{(3)} = \begin{pmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & -4 \end{pmatrix}$ se fait selon

$$A^{(3)} = \left(I - \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -3 & 0 \end{pmatrix} \right) A^{(2)} = (I - l^2 e_2^T)\, A^{(2)} = E^{(2)}\, A^{(2)},$$

où $l^2 = (0,\ 0,\ -3)^T$ et e_2 désigne le second vecteur de la base canonique.

Notons $U = A^{(3)}$ la matrice triangulaire supérieure obtenue à l'issue de cette étape d'élimination. Elle vérifie :

$$U = E^{(2)}\, E^{(1)}\, A. \tag{1.16}$$

Les matrices $E^{(1)}$ et $E^{(2)}$ sont inversibles, selon

- $\left(E^{(1)}\right)^{-1} = L^{(1)} = (I + l^1\, e_1^T) = \begin{pmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$; on le vérifie en écrivant $(I - l^1\, e_1^T)\,(I + l^1\, e_1^T) = I - l^1\, \left(e_1^T\, l^1\right)\, e_1^T = I$, puisque

$$e_1^T\, l^1 = 1 \times 0 + 0 \times 3 + 0 \times (-1) = 0\,;$$

- $\left(E^{(2)}\right)^{-1} = L^{(2)} = (I + l^2\, e_2^T)$; on le vérifie de la même façon en écrivant $(I - l^2\, e_2^T)\,(I + l^2\, e_2^T) = I - l^2\, (e_2^T\, l^2)\, e_2^T = I$, puisque

$$e_2^T\, l^2 = 0 \times 0 + 1 \times 0 + 0 \times (-3) = 0.$$

En multipliant à gauche (1.16) par $L^{(1)}\, L^{(2)}$, on obtient :

$$L^{(1)}\, L^{(2)}\, U = L\, U = A. \tag{1.17}$$

La matrice L “s'assemble” sans calcul, selon

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ -1 & -3 & 1 \end{pmatrix} \tag{1.18}$$

Pour résoudre le système $A\,x = f$, on procède ensuite en deux étapes :
– l'étape de descente consiste à résoudre $L\,y = f$, c'est-à-dire

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ -1 & -3 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 7 \end{pmatrix},$$

dont la solution se calcule par substitutions progressives : $y = (1, -4, -4)^T$;
– l'étape de remontée consiste à résoudre $U\,x = y$; elle est identique à celle que l'on a décrite dans le cadre de l'élimination puisque $y = b^{(3)}$.

Cas général

Dans l'élimination de Gauss pour le système $A\,x = f$, avec $A = A^{(1)} \in M_n(\mathbb{R})$, la première étape s'écrit :

$$A^{(1)}\,x = f^{(1)} \Leftrightarrow A^{(2)}\,x = f^{(2)}$$

de telle façon que la matrice $A^{(2)}$ n'ait que des 0 sous l'élément diagonal $a^{(2)}_{1,1}$ de sa première colonne.

En supposant que $a_{1,1} = a^{(1)}_{1,1} \neq 0$, la matrice $A^{(2)}$ se déduit de $A^{(1)}$ par les relations suivantes :

$$\begin{array}{rcll} a^{(2)}_{1,j} & = & a^{(1)}_{1,j}, & \forall j,\ 1 \leqslant j \leqslant n, \\ a^{(2)}_{i,1} & = & 0, & \forall i,\ 2 \leqslant i \leqslant n, \\ a^{(2)}_{i,j} & = & a^{(1)}_{i,j} - l_{i,1}\, a^{(1)}_{1,j}, & \forall i, \forall j,\ 2 \leqslant i, j \leqslant n. \end{array}$$

où les coefficients $l_{i,1}$ sont définis par $l_{i,1} = \dfrac{a^{(1)}_{i,1}}{a^{(1)}_{1,1}}$. La matrice $A^{(2)}$ s'interprète donc ici aussi comme le produit $A^{(2)} = E^{(1)}\,A^{(1)}$, avec $E^{(1)}$ définie de la façon suivante :

$$E^{(1)} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -l_{2,1} & 1 & \ddots & & \vdots \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & 0 \\ -l_{n,1} & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

On vérifie immédiatement que $E^{(1)} = (I - l^1\,e_1^T)$, e_1 désignant le premier vecteur de la base canonique de $\mathbb{R}^n$ et l^1 le vecteur $(0,\, l_{2,1}, \ldots,\, l_{n,1})^T$.

La matrice s'écrit, avec la notation de MATLAB :

$$\mathtt{A^{(2)}} = \begin{bmatrix} \mathtt{a}^{(1)}_{1,1} & , & \mathtt{A^{(1)}(1,\,2:n)} \\ \\ \mathtt{0} & , & \mathtt{A^{(2)}(2:n,\,2:n)} \end{bmatrix}$$

Si le pivot $p_2 = a^{(2)}_{2,2} \neq 0$, on peut renouveler l'opération pour la matrice $\mathtt{A^{(2)}(2:n,\ 2:n)}$ qui appartient à $M_{n-1}(\mathbb{R})$ au moyen d'une matrice $\mathtt{E^{(2)}(2:n,\ 2:n)}$ construite de façon analogue à $E^{(1)}$. On complète cette matrice $E^{(2)}$ pour en faire une matrice de $M_n(\mathbb{R})$ telle que le produit à gauche par $E^{(2)}$ laisse invariantes le première ligne et et la première colonne :

$$\mathtt{E^{(2)}} = \begin{bmatrix} \mathtt{1} & , & \mathtt{0} \\ \mathtt{0} & , & \mathtt{E^{(2)}(2:n,\ 2:n)} \end{bmatrix} = \left(I - l^2\, e_2^T\right),$$

avec e_2 le second vecteur de la base canonique de $\mathbb{R}^n$ et $l^2 = (0,\ 0,\ l_{3,2},\ \ldots,\ l_{n,2})^T$ tel que pour $3 \leqslant i \leqslant n$, $l_{i,2} = \dfrac{a^{(2)}_{i,2}}{a^{(2)}_{2,2}}$. On obtient :

$$E^{(2)}\, A^{(2)} = E^{(2)}\ E^{(1)}\ A^{(1)} = \begin{bmatrix} \mathtt{a^{(1)}_{1,1}} & , & \mathtt{a^{(1)}_{1,2}} & , & \mathtt{A^{(1)}(1,\ 3:n)} \\ \mathtt{0} & , & \mathtt{a^{(2)}_{2,2}} & , & \mathtt{A^{(2)}(2,\ 3:n)} \\ \mathtt{0} & , & \mathtt{0} & , & \mathtt{A^{(3)}(3:n,\ 3:n)} \end{bmatrix}$$

avec $\mathtt{A^{(3)}(3:n,\ 3:n)} \in M_{n-2}(\mathbb{R})$: Si au-delà, les $a^{(k)}_{k,k}$ sont tous non nuls, on pourra réitérer l'opération pour obtenir finalement :

$$E^{(n-1)} \cdots E^{(2)}\ E^{(1)}\ A^{(1)} = U.$$

La factorisation $A = L\,U$ cherchée se déduit alors du lemme suivant :

Lemme 1 *Soit $E^{(k)}$ la matrice définie pour $1 \leqslant k \leqslant n-1$ par $E^{(k)} = I - l^k\, e_k^T$, avec $l^k = (l_{1,k}, \ldots,\ l_{n,k})^T$ telle que $l_{i,k} = 0$ si $i \leqslant k$, et e_k le k-ième vecteur de la base canonique de $\mathbb{R}^n$. Alors,*

(i) la matrice $E^{(k)}$ est inversible d'inverse $L^{(k)} = I + l^k\, e_k^T$,

(ii) le produit $L = L^{(1)} \ldots L^{(n-1)}$ "s'assemble" sans calculs de telle façon que

$$L(i,j) = \begin{cases} 0 & \text{si } i < j, \\ 1 & \text{si } i = j, \\ l_{i,j} & \text{si } i > j \end{cases}$$

En effet, $(I - l^k\, e_k^T)\ (I + l^k\, e_k^T) = I - l^k\ (e_k^T\, l^k)\ e_k^T$ et $e_k^T\, l^k = \displaystyle\sum_{i=k+1}^{n} \delta_{k,i}\, l_{i,k} = 0$.

D'autre part, $(I + l^1\, e_1^T) \ldots (I + l^{n-1}\, e_{n-1}^T) = I + l^1\, e_1^T + \ldots + l^{n-1}\, e_{n-1}^T$ puisque les produits $e_i^T\, l^j$ sont nuls dès que $j \geqslant i$.

Partant de $E^{(n-1)} \ldots E^{(1)}\, A = U$, on obtient donc :

$$A = L^{(1)} \ldots L^{(n-1)}\, U = L\, U.$$

C'est la factorisation cherchée, avec L triangulaire inférieure d'éléments diagonaux égaux à 1, et dont les élements sous-diagonaux sont les $l_{i,j} = \dfrac{a^{(j)}_{j,j}}{a^{(j)}_{i,j}}$, $1 \leqslant j < i \leqslant n$, et U triangulaire supérieure inversible puisque sa diagonale est constituée des pivots qui sont tous différents de 0.

Une condition nécessaire et suffisante :

On admettera le théorème suivant :

Théorème 2 *Soit $A = (a_{i,j})_{1 \leqslant i,j \leqslant n} \in M_n(\mathbb{R})$; si pour tout k, $1 \leqslant k \leqslant n$, la sous-matrice principale $A_k = (a_{i,j})_{1 \leqslant i,j \leqslant k}$ de A est telle que $det(A_k) \neq 0$, il existe une matrice triangulaire inférieure L, dont les éléments diagonaux sont égaux à 1, et une matrice triangulaire supérieure inversible U telles que $A = L\,U$. Cette factorisation est unique.*

Cette condition nécessaire et suffisante n'est pas très satisfaisante : on ne peut pas calculer les déterminants mineurs principaux de la matrice d'un système linéaire avant de le résoudre !

Implantation (version élémentaire) :

Nous allons proposer un algorithme de cette factorisation sous la forme d'une fonction MATLAB . Comme il s'agit d'une fonction, on peut "écraser" la matrice A qui lui est passée en argument, car elle alors considérée comme une **variable locale** de la fonction.

Observons d'abord la première étape de la factorisation : la matrice $A = A^{(1)}$ est transformée en $A^{(2)} = (I - l^1\, e_1^T)\, A^{(1)} = A^{(1)} - l^1\, (e_1^T\, A^{(1)})$.

Le produit $e_1^T\, A^{(1)})$ est une matrice à une ligne (vecteur ligne) qui est égale à la première ligne de $A^{(1)}$: on a donc

$$A^{(2)} = A^{(1)} - l^1\, A^{(1)}(1, :) \tag{1.19}$$

Le vecteur $l^1 = (0,\, l_{2,1},\, \ldots,\, l_{n,1})^T$ est tel que :
- la première ligne de $A^{(2)}$ sera égale à la première ligne de $A^{(1)}$,
- les coefficients des lignes 2 à N de la première colonne de $A^{(2)}$ seront nuls. On a donc besoin de ne modifier que la sous-matrice $A^{(1)}(2 : n,\, 2 : n)$; on peut disposer des positions 2 à n de la première colonne de la matrice $A^{(1)}$ pour stocker les composantes non nulles de l^1, c'est-à-dire les $l_{j,1}$, pour $2 \leqslant j \leqslant n$.
- dès que les composantes du vecteur l^1 sont calculées, on n'a donc plus à calculer que

$$A^{(2)}(2 : n,\, 2 : n) = A^{(1)}(2 : n,\, 2 : n) - l^1(2 : n)\; A^{(1)}(1,\, 2 : n) \tag{1.20}$$

- si l'on stocke les $l_{j,1}$ non nuls en utilisant les positions $A^{(1)}(2 : n,\, 1)$ inutiles pour la suite, (1.20) devient

$$A^{(2)}(2 : n,\, 2 : n) = A^{(1)}(2 : n,\, 2 : n) - A^{(1)}(2 : n,\, 1)\, A^{(1)}(1,\, 2 : n) \tag{1.21}$$

Le procédé que l'on à décrit va se répéter pour le passage de $A^{(2)}$ à $A^{(3)}$, et cette fois ce sont les positions $A^{(2)}(3 : n,\, 2)$ qui seront utilisées pour stocker les composantes non nulles de l^2 ; seule la sous-matrice $A^{(2)}(3 : n,\, 3 : n)$ sera modifiée par soustraction d'une **matrice de rang 1**, c'est-à-dire une matrice de la forme $M = u\, v^T$ où u et v sont des vecteurs. Et ce même principe sera mis en oeuvre jusqu'à obtention de la factorisation.

A l'issue de cette factorisation, la partie triangulaire supérieure de la matrice A sera occupée par la matrice U, et sa partie triangulaire strictement inférieure par celle de L.

La fonction MATLAB ci-dessous réalise cette factorisation. Les commentaires qui suivent immédiatement la ligne de déclaration de la fonction en indiquent l'objet, le format d'appel et la façon de l'utiliser.

```
 function [L, U] = Gauss(A);
%       Gauss calcule la factorisation LU de la matrice A
%       par la methode d'elimination de Gauss.
%       L'appel se fait selon :
%               >> [L, U] = Gauss(A);
%       La solution du systeme Ax = f est donnee par :
%               >> x = U\(L\f);

[n, n] = size(A);
for k = 1:n-1
    if A(k,k) == 0
      error('Matrice non factorisable ...')
    end
    Indices = k+1:n;

    % Colonne k de la matrice L) :
    A(Indices,k) = A(Indices,k)/A(k,k) ;

    % Mise a jour de la matrice A^(k) :
    A(Indices,Indices) = A(Indices,Indices)-A(Indices,k)*A(k,Indices) ;
end
L = tril(A,-1) + eye(n);
U = triu(A);
```

Coût de la factorisation

Pour chaque valeur de k, la variable `Indices` contient $n-k$ valeurs ; on doit donc effectuer :
- $n-k$ divisions pour le calcul de la colonne k de L,
- $(n-k)^2$ multiplications pour le calcul de la matrice `A(Indices,k)*A(k,Indices)`, et $(n-k)^2$ soustractions pour la mise à jour de `A(Indices,Indices)`.

Le coût $C_F(n)$ de la factorisation s'obtient en sommant pour k variant de 1 à $n-1$:

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

On a donc $C_F(n) \sim \dfrac{2\,n^3}{3}$ opérations élémentaires pour l'étape de factorisation.

Pour la résolution d'un système linéaire, on fait suivre cette étape de factorisation par deux étapes de résolution, respectivement d'un système triangulaire inférieur de matrice $L = \texttt{tril(A)} + \texttt{eye(n)}$ et supérieur de matrice $U = \texttt{triu(A)}$:

$$\left\{ \begin{array}{rcl} L\,y & = & f, \\ U\,x & = & y. \end{array} \right.$$

Le coût de la résolution du système reste donc $\mathcal{O}(\dfrac{2\,n^3}{3})$ opérations.

Retour à l'exemple

On peut observer sur la matrice $A = \begin{pmatrix} 2 & 1 & 1 \\ 6 & 2 & 1 \\ -2 & 2 & 1 \end{pmatrix}$ les différentes étapes de la fonction ci-desssus ; par souci de lisibilité, nous imprimons en caractères gras les valeurs calculées de la matrice L et leur positionnement dans la matrice A :

- pour $k = 1$, la matrice A est remplacée par $A = \begin{pmatrix} 2 & 1 & 1 \\ \mathbf{3} & -1 & -2 \\ \mathbf{-1} & -3 & 2 \end{pmatrix}$,
- pour $k = 2$, la matrice A est remplacée par $A = \begin{pmatrix} 2 & 1 & 1 \\ \mathbf{3} & -1 & -2 \\ \mathbf{-1} & \mathbf{-3} & -4 \end{pmatrix}$.

Les instructions MATLAB

```
L = tril(A,-1) + eye(n);
U = triu(A);
```

permettent bien de retrouver L et U.

1.3.4 Factorisation $P\,A = L\,U$

Principe

Sous réserve d'une permutation de ses lignes, toute matrice inversible est factorisable. Formellement, cette permutation des lignes peut s'écrire comme le produit à gauche de cette matrice par une matrice de permutation.

Théorème 3 *Si* $\det(A) \neq 0$, *il existe une matrice* P *de permutation telle que* $P\,A$ *soit factorisable selon* $P\,A = L\,U$.

Ce théorème exprime le fait que si A est inversible, on trouvera à chaque étape k de la factorisation, $1 \leqslant k \leqslant n-1$, au moins un coefficient $a_{i,k}^{(k)},\ k \leqslant i \leqslant n$, non nul.

Un exemple

Pour résoudre le système $A\,x = f$ suivant :

$$\left\{\begin{array}{rrrrcl} x_1 & +x_2 & +x_3 & +x_4 & = & 1 \\ x_1 & +x_2 & +3x_3 & +3x_4 & = & 3 \\ x_1 & +x_2 & +2x_3 & +3x_4 & = & 3 \\ x_1 & +3x_2 & +3x_3 & +3x_4 & = & 4 \end{array}\right. \iff \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 3 & 3 \\ 1 & 1 & 2 & 3 \\ 1 & 3 & 3 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 3 \\ 4 \end{pmatrix}$$

La première étape de l'élimination conduit au système équivalent :

$$\left\{\begin{array}{rrrrrrrcl} x_1 & + & x_2 & + & x_3 & + & x_4 & = & 1 \\ & & & & 2x_3 & + & 2x_4 & = & 2 \\ & & & & x_3 & + & 2x_4 & = & 2 \\ & & 2x_2 & + & 2x_3 & + & 2x_4 & = & 3 \end{array}\right.$$

On constate que dans ce système, l'inconnue x_2 est déjà éliminée des équations $(Eq.2)$ et $(Eq.3)$, c'est-à-dire que $a_{2,2}^{(2)} = a_{3,2}^{(2)} = 0$. La poursuite du processus d'élimination nécessite une

permutation des équations $(Eq.2)$ et $(Eq.4)$:

$$A^{(2)}\,x = f^{(2)} \Longleftrightarrow P_2\,A^{(2)}\,x = P_2\,f^{(2)} \Longleftrightarrow \left\{ \begin{array}{rcrcrcrcl} x_1 & + & x_2 & + & x_3 & + & x_4 & = & 1 \\ & & 2x_2 & + & 2x_3 & + & 2x_4 & = & 3 \\ & & & & x_3 & + & 2x_4 & = & 2 \\ & & & & 2x_3 & + & 2x_4 & = & 2 \end{array} \right.$$

La matrice P_2 qui multiplie à gauche chaque membre du système d'équations est la matrice d'une permutation élémentaire. On peut l'écrire

$$P_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

et on vérifie immédiatement que

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 2 & 2 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & 2 \end{pmatrix}$$

On poursuit l'élimination sans problème pour obtenir

$$\left\{ \begin{array}{rcrcrcrcl} x_1 & + & x_2 & + & x_3 & + & x_4 & = & 1 \\ & & 2x_2 & + & 2x_3 & + & 2x_4 & = & 3 \\ & & & & x_3 & + & 2x_4 & = & 2 \\ & & & & & - & 2x_4 & = & -2 \end{array} \right.$$

On a obtenu un système triangulaire inversible.

Ecriture matricielle de la factorisation avec permutation

Le système triangulaire obtenu ci-dessus s'écrit $U\,x = f^{(4)}$, avec $f^{(4)} = (1,\,3,\,2,\,-2)^T$ et $U = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & -2 \end{pmatrix}$.

La matrice U de ce système est définie par :

$$E^{(3)}\,E^{(2)}\,P_2\,E^{(1)}\,A = U, \tag{1.22}$$

avec, pour chaque k, $1 \leqslant k \leqslant 3$, $E^{(k)} = I - l^k\,e_k^T$, et

- $l^1 = (0,\,-1,\,-1,\,-1)^T$,
- $l^2 = (0,\,0,\,0,\,0)^T$, puisque les coefficients $a_{3,2}^{(2)}$ et $a_{4,2}^{(2)}$ sont tous deux nuls,
- $l^3 = (0,\,0,\,0,\,-2)^T$.

La matrice P_2 s'écrit également sous une forme $P_2 = I - u\,u^T$, avec $u = e_2 - e_4$ (c'est la différence des deux vecteurs de la base canonique correspondant aux indices des lignes à permuter) ! Elle est telle que $P_2^2 = I$ (vérification immédiate). On peut alors glisser P_2^2 à droite de $E^{(1)}$ dans les deux membres de (1.22) :

$$E^{(3)}\,E^{(2)}\,P_2\,E^{(1)}\,P_2\,P_2\,A = U.$$

Le produit matriciel est associatif : on peut isoler les facteurs

$$P_2\, E^{(1)}\, P_2 = P_2\, (I - l^1\, e_1^T)\, P_2 = P_2^2 - P_2\, l^1\, e_1^T\, P_2 = I - (P_2\, l^1)\, (e_1^T\, P_2),$$

et calculer :

- $e_1^T\, P_2 = (1,\, 0,\, 0,\, 0) \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} = e_1^T,$

- $P_2\, l^1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ l_{2,1} \\ l_{3,1} \\ l_{4,1} \end{pmatrix} = \begin{pmatrix} 0 \\ l_{4,1} \\ l_{3,1} \\ l_{2,1} \end{pmatrix} = \widetilde{l^1}$

de sorte que finalement

$$E^{(3)}\, E^{(2)}\, \widetilde{E^{(1)}}\, P_2\, A = U,$$

d'où l'on tire, de façon analogue à ce qu'on a vu précédemment, en notant P la matrice P_2 :

$$P\, A = \widetilde{L^{(1)}}\, L^{(2)}\, L^{(3)}\, U = L\, U \tag{1.23}$$

avec $\mathtt{L} = [\widetilde{\mathtt{l}^1},\, \mathtt{l}^2,\, \mathtt{l}^3,\, \mathtt{l}^4]$.

Pour représenter cette factorisation, il est inutile de construire la matrice P. Il suffit d'adjoindre à la matrice A un compteur de permutation sous forme d'une colonne supplémentaire, p, initialisée à l'ordre naturel :

$$(A\,|\,p) \quad = \quad \left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 3 & 3 & 2 \\ 1 & 1 & 2 & 3 & 3 \\ 1 & 3 & 3 & 3 & 4 \end{array}\right) \quad \Longrightarrow \quad \left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 1 \\ \mathbf{1} & 0 & 2 & 2 & 2 \\ \mathbf{1} & 0 & 1 & 2 & 3 \\ \mathbf{1} & 2 & 2 & 2 & 4 \end{array}\right)$$

$$\Longrightarrow \quad \left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 1 \\ \mathbf{1} & 2 & 2 & 2 & 4 \\ \mathbf{1} & \mathbf{0} & 1 & 2 & 3 \\ \mathbf{1} & \mathbf{0} & 2 & 2 & 2 \end{array}\right) \quad \Longrightarrow \quad \left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 1 \\ \mathbf{1} & 2 & 2 & 2 & 4 \\ \mathbf{1} & \mathbf{0} & 1 & 2 & 3 \\ \mathbf{1} & \mathbf{0} & \mathbf{2} & -2 & 2 \end{array}\right)$$

On vérifie en effet que

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & -2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 3 & 3 & 3 \\ 1 & 1 & 2 & 3 \\ 1 & 1 & 3 & 3 \end{pmatrix} = P\, A.$$

Le compteur de permutation $p = (1,\, 4,\, 3,\, 2)^T$ permet de retrouver la permutation des lignes que l'on doit effectuer pour résoudre le système $P\, A = L\, U = P\, f$.

Généralisation

Considérons à présent une matrice inversible $A \in M_n(\mathbb{R})$. De même que ci-dessus à la seconde étape de la factorisation, si on rencontre un pivot nul à l'étape k, la permutation de la ligne k avec une ligne $i > k$ telle que $a_{i,k}^{(k)} \neq 0$ s'écrira $P_k = I - u\, u^T$, avec $u = e_k - e_i$.

Pour tout $j < k$, on aura

$$P_k\, E^{(j)}\, P_k = I - P_k\, l^j\, e_j^T\, P_k == I - \widetilde{l^j}\, e_j^T = \widetilde{E^{(j)}},$$

puis

$$\begin{aligned} P_k\, E^{(k-1)} \ldots E^{(1)} &= (P_k\, E^{(k-1)}\, P_k)\, (P_k \ldots Pk)\, (P_k\, E^{(1)}\, P_k)\, P_k \\ &= \widetilde{E^{(k-1)}} \ldots \widetilde{E^{(1)}}\, P_k \end{aligned} .$$

En introduisant des matrices P_k à chaque étape, avec éventuellement $P_k = I$ lorsqu'aucune permutation n'est nécessaire, on obtiendra

$$\left(E^{(n-1)}\, \widetilde{E^{(n-2)}} \ldots \widetilde{E^{(1)}} \right)\, (P_{n-1} \ldots P_1)\; A = U,$$

puis

$$P\, A = L\, U.$$

Stratégie de pivot partiel

La factorisation $P\, A = L\, U$ est nécessaire lorsqu'on rencontre un pivot nul. Il se peut que l'on rencontre un pivot qui sans être nul est très petit. Dans ce cas, l'effet des erreurs d'arrondi peut-être catastrophique.

Voici par exemple quelques instructions MATLAB qui utlisent la fonction `Gauss` listée précédement, pour calculer la factorisation $A = L\, U$ d'une matrice pour laquelle $a_{1,1}$ est très petit, sans être nul :

```
n = 5 ;
A = rand(n) + eye(n) ;
A(1,1) = 2*eps ;
[L, U] = Gauss(A) ;
B = L*U ;
erreur = norm(A-B)
```

La fonction `eps` de MATLAB fournit la précision relative de l'arithmétique dans laquelle est implanté MATLAB. Ici, `eps = 2.2204e-16`. Pour une réalisation j'ai obtenu :

```
>> erreur = 0.1940
```

En divisant par un pivot très petit, on augmente le risque d'erreurs dues à l'élimination de chiffres significatifs des mantisses.

En effet, si à l'étape k de la factorisation, le pivot $a_{k,k}^{(k)}$ est très petit, les coefficients $l_{i,k}$, $k \leqslant i \leqslant n$ seront en général très grand.

Les lignes de la matrice $A^{(k+1)}$ sont définies par l'expression MATLAB

$$\texttt{A(i, k + 1 : n)} - \texttt{l(i, k)} * \texttt{A(k, k + 1 : n)};$$

Le premier terme sera négligeable, et ces lignes seront presque dépendantes, puisqu'à peu près proportionnelles à `A(k,k+1 :n)` ! En terme de méthode d'élimination, le système $A^{(k+1)}\, x = f^{(k+1)}$ sera très sensible aux perturbations, puisque proche d'un système non inversible.

Pour limiter ce risque, on utilise en général une stratégie de recherche du pivot maximal dans chaque colonne. A l'étape k, plutôt que de rechercher le premier coefficient non nul parmi les $a_{i,k}^{(k)}$, $i \geqslant k$, on recherche l'indice i_m tel que

$$a_{i_m,k}^{(k)} = \max_{k \leqslant i \leqslant n} \left\{ \left| a_{i,k}^{(k)} \right| \right\}.$$

On permute ensuite les lignes k et i_m.

La fonction MATLAB ci-dessous met en oeuvre cette stratégie :

```
function [L, U, permute] = Gausspiv(A);
%       Gauss calcule la factorisation LU de la matrice
%       A avec strategie de pivot partiel.
%       L'appel se fait selon :
%               >> [L, U, permute] = Gauss(A);
%       Elle est telle que A(permute, :) = L*U.
%       La solution du systeme  Ax = f est donnee par :
%              >> x = U \(L \(f(permute, :)) ;

[n, n] = size(A);
permute = 1:n;
for k = 1:n-1

  % Recherche du pivot, permutation des lignes correspondantes
    [maxi, ligne] = max(abs(A(k:n,k)));
    ligne = ligne+k-1;
    A([k, ligne],:) = A([ligne, k],:);
    permute([k, ligne]) = permute([ligne, k]);

    if A(k,k) == 0
      error('Matrice non inversible ...')
    end

    % Colonne k de la matrice L) :
    A(k+1:n,k) = A(k+1:n,k)/A(k,k) ;

    % Mise a jour de la matrice A^(k) :
    A(k+1:n,k+1:n) = A(k+1:n,k+1:n)-A(k+1:n,k)*A(k,k+1:n) ;
end
L = tril(A,-1) + eye(n);
U = triu(A);
```

Cette stratégie est considérée comme suffisament stable par la plupart des bibliothèques scientifiques ; c'est celle qui est généralement implantée, en particulier dans MATLAB par la fonction `lu`.

Un exemple

On peut suivre les étapes de l'exécution de la fonction précédente sur la matrice

$$[\,\mathtt{A}\,|\,\mathtt{p}\,] = \left(\begin{array}{cccc|c} 1 & 2 & 4 & 17 & 1 \\ 3 & 6 & -12 & 3 & 2 \\ 2 & 3 & -3 & 2 & 3 \\ 0 & 2 & -2 & 6 & 4 \end{array}\right)$$

$$\mathtt{k}=1: \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ 1 & 2 & 4 & 17 & 1 \\ 2 & 3 & -3 & 2 & 3 \\ 0 & 2 & -2 & 6 & 4 \end{array}\right) \quad \Longrightarrow \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ \mathbf{1/3} & 0 & 8 & 16 & 1 \\ \mathbf{2/3} & -1 & 5 & 0 & 3 \\ \mathbf{0} & 2 & -2 & 6 & 4 \end{array}\right)$$

$$\mathtt{k}=2: \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ \mathbf{0} & 2 & -2 & 6 & 4 \\ \mathbf{2/3} & -1 & 5 & 0 & 3 \\ \mathbf{1/3} & 0 & 8 & 16 & 1 \end{array}\right) \quad \Longrightarrow \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ \mathbf{0} & 2 & -2 & 6 & 4 \\ \mathbf{2/3} & \mathbf{-1/2} & 4 & 3 & 3 \\ \mathbf{1/3} & \mathbf{0} & 8 & 16 & 1 \end{array}\right)$$

$$\mathtt{k}=3: \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ \mathbf{0} & 2 & -2 & 6 & 4 \\ \mathbf{1/3} & \mathbf{0} & 8 & 16 & 1 \\ \mathbf{2/3} & \mathbf{-1/2} & 4 & 3 & 3 \end{array}\right) \quad \Longrightarrow \quad \left(\begin{array}{cccc|c} 3 & 6 & -12 & 3 & 2 \\ \mathbf{0} & 2 & -2 & 6 & 4 \\ \mathbf{1/3} & \mathbf{0} & 8 & 16 & 1 \\ \mathbf{2/3} & \mathbf{-1/2} & \mathbf{1/2} & -5 & 3 \end{array}\right)$$

On vérifie que

$$\left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1/3 & 0 & 1 & 0 \\ 2/3 & -1/2 & 1/2 & 1 \end{array}\right) \left(\begin{array}{cccc} 3 & 6 & -12 & 3 \\ 0 & 2 & -2 & 6 \\ 0 & 0 & 8 & 16 \\ 0 & 0 & 0 & -5 \end{array}\right) = \left(\begin{array}{cccc} 3 & 6 & -12 & 3 \\ 0 & 2 & -2 & 6 \\ 1 & 2 & 4 & 17 \\ 2 & 3 & -3 & 2 \end{array}\right) = P\,A.$$

La matrice $P\,A$ ne se calcule pas par un produit matriciel. A l'issue de l'éxécution de cet algorithme, on a $p = (2,\ 4,\ 1,\ 3)^T$ et $P\,A = A(p,\ :)$.

Stratégie de pivot total

Les systèmes qui la rende indispensable sont très rares , et elle n'est que rarement implantée. La factorisation peut alors s'écrire :

$$P\,A\,Q = L\,U$$

La matrice Q est ici également une matrice de permutation ; son effet est de permuter les colonnes de A. Il faut alors conserver la mémoire de cette permutation pour replacer les inconnues du système dans le bon ordre.

Unicité de la factorisation

On suppose que la matrice A est "factorisable" selon $P\,A = L\,U$, les éléments diagonaux de L étant tous égaux à 1. Supposant que l'on puisse former deux factorisations différentes, on écrit :

$$P\,A = L_1\,U_1 = L_2\,U_2.$$

On peut alors former :

$$M = L_2^{-1}\, L_1 = U_2 U_1^{-1}$$

$L_2^{-1}\, L_1$ est triangulaire inférieure de diagonale unité, et $U_2 U_1^{-1}$ est triangulaire supérieure : toutes deux sont égales à l'identité.

Factorisation $L\, D\, R$

Il suffit d'écrire :

$$\begin{pmatrix} u_{1,1} & & & \\ & \ddots & u_{i,j} & \\ & & \ddots & \\ & 0 & & u_{n,n} \end{pmatrix} = \begin{pmatrix} u_{1,1} & & & \\ & \ddots & 0 & \\ & & \ddots & \\ & 0 & & u_{n,n} \end{pmatrix} \begin{pmatrix} 1 & & & \\ & \ddots & \frac{u_{i,j}}{u_{i,i}} & \\ & & \ddots & \\ & 0 & & 1 \end{pmatrix}$$

pour obtenir $U = D\, R$

1.4 Matrices symétriques définies positives

1.4.1 Matrices symétriques définies positives

Parmi les problèmes qui conduisent à un système linéaire où la matrice est symétrique, les plus intéressants sont ceux pour lesquels on n'a pas besoin de stratégie de pivot ; c'est le cas lorsque la matrice est symétrique définie positive.

1.4.2 Factorisation de Choleski

Théorème 4 *Une condition nécessaire et suffisante pour que A soit symétrique définie positive est qu'il existe B, triangulaire inférieure inversible, unique si ses éléments diagonaux sont choisis positifs, telle que $A = B\, B^T$.*

On vérifie sans problème la condition suffisante : si $A = B\, B^T$, alors $x^T\, A\, x = x^T\, B\, B^T\, x = \left\| B^T\, x \right\|^2 > 0$ si $x \neq 0$.

Pour la condition nécessaire, on raisonne par récurrence en vérifiant que le théorème est vrai pour $n = 1$, puisque $a \in \mathbb{R}^n$ n'est strictement positive que si et seulement si il existe $b > 0$ tel que $b^2 = a$, et en l'occurence, $b = \sqrt{a}$.

On suppose alors la condition vérifiée jusqu'à l'ordre $n-1$, et on écrit par blocs la matrice A selon :

$$A = \begin{pmatrix} A_{n-1} & a \\ a^T & m^2 \end{pmatrix}$$

où $A_{n-1} \in M_{n-1}(\mathbb{R})$ (sous-matrice principale de A) est également symétrique définie positive, $a \in \mathbb{R}^{n-1}$ et $a_{n,n} = m^2$ est strictement positif.

On cherche B de la forme :

$$B = \begin{pmatrix} B_{n-1} & 0 \\ b^T & b_{n,n} \end{pmatrix}$$

telle que :

$$B\,B^T = \begin{pmatrix} B_{n-1} & 0 \\ b^T & b_{n,n} \end{pmatrix} \begin{pmatrix} B_{n-1}^T & b \\ 0 & b_{n,n} \end{pmatrix} = A$$

c'est-à-dire :

$$\begin{pmatrix} B_{n-1}\,B_{n-1}^T & B_{n-1}b \\ b^T B_{n-1}^T & b^T\,b + b_{n,n}^2 \end{pmatrix} = \begin{pmatrix} A_{n-1} & a \\ a^T & m^2 \end{pmatrix}$$

Par hypothèse de récurrence, on peut trouver B_{n-1} triangulaire inférieure inversible telle que $A_{n-1} = B_{n-1}\,B_{n-1}^T$, et le vecteur $b \in \mathbb{R}^{n-1}$ est solution du système inversible $B_{n-1}b = a$.

Le seul problème est donc le calcul de $b_{n,n}$, qui n'est possible que si $m^2 - b^T\,b > 0$.

$$m^2 - b^T\,b = m^2 - \left(B_{n-1}^{-1}\,a\right)^T\,\left(B_{n-1}^{-1}\,a\right) = m^2 - a^T\,A_{n-1}^{-1}\,a > 0$$

En effet, A est symétrique définie positive : en choisissant le vecteur X qui s'écrit par blocs $X = \begin{pmatrix} A_{n-1}^{-1}\,a \\ -1 \end{pmatrix}$, avec $A_{n-1}^{-1} \in M_{n-1}(\mathbb{R})$ on vérifie que $X^T\,A\,X = m^2 - a^T\,A_{n-1}^{-1}\,a$.

1.4.3 Algorithme

Nous pouvons déduire cet algorithme de la démonstration du théorème 4 : le raisonnement par récurrence, dans cette démonstration, permet d'affirmer que si $A_k = (a_{i,j})_{1 \leqslant i,j \leqslant k}$ désigne la sous-matrice principale de A, et B_k celle de B, alors $A_k = B_k\,B_k^T$.

Le passage de B_{k-1} à B_k se fait en deux étapes :
- calcul de $b^T = \mathtt{B(k,\ 1:k-1)}$ en résolvant le système triangulaire inférieur $B_{k-1}\,b = a$, avec $a = \mathtt{A(1:k-1,\ k)}$;
- calcul de $b_{k,\,k} = \sqrt{a_{k,k} - b^T\,b}$.

L'algorithme de factorisation, peut donc s'écrire, en langage MATLAB, et dans une forme qui n'écrase pas la matrice A :

```
N = size(A,1);
B = zeros(size(A));
B(1,1) = sqrt(A(1,1));
for k = 2:N
   % Resolution du systeme triangulaire inférieur :
   for l = 1:k-1
       B(k,l) = (A(k,l)-B(l,1:l-1)*B(k,1:l-1)')/B(l,l);
   end;
   % Calcul de l'element diagonal
   B(k,k) = sqrt(A(k,k)-B(k,1:k-1)*B(k,1:k-1)');
end;
```

Remarque 5 *On peut tout aussi bien énoncer la factorisation de Choleski de la matrice symétrique définie positive A selon $A = H^T H$ avec H triangulaire supérieure. C'est généralement le cas dans la littérature anglo-saxonne, et c'est en particulier l'implantation de la fonction* `chol` *de* MATLAB.

Complexité

On peut s'aider du programme MATLAB ci-dessus pour calculer le coût de la factorisation. Pour chaque valeur de k comprise entre 2 et n, on effectue, sans compter les extractions de racines carrées :

- une résolution de système triangulaire de taille $k-1$, soit $(k-1)^2$ opérations élémentaires,
- $2\,(k-1)$ opérations élémentaires pour le calcul de l'élément diagonal.

Le coût de la factorisation est donc

$$\begin{aligned} C(n) &= \sum_{l=1}^{n-1} l^2 + 2\sum_{l=1}^{n-1} l, \\ &= \frac{n\,(n-1)\,(2n-1)}{6} + n\,(n-1), \\ &= \frac{n\,(n-1)\,(2n+5)}{6}, \end{aligned}$$

auquel s'ajoutent n calculs de racines.

Chapitre 2

Equations et Systèmes Non Linéaires

Dans le cas général, on ne peut pas espérer résoudre une équation ou un système d'équations non linéaires par une formule directe : ça n'est possible que pour quelques équations algébriques, comme par exemple les équations du second degré.

La formule qui donnne les deux racines des équations du second degré est connue depuis très longtemps ; elle a été retrouvée sur des tablettes babyloniennes datées des environs de 1500 avant notre ère. Pour les équations de degré 3 la solution a été établie par l'italien Tartaglia en 1535, mais publiée en 1545 par Jérôme Cardan, qui par ailleurs a laissé son nom à un système de transmission cher aux automobilistes. Pour le degré 4, c'est encore un italien, dont le nom sonne de façon plaisante aux oreilles des amateurs d'automobiles qui a établi la méthode (Ludovico Ferrari en 1565). Ces formules sont exactes si l'on considère que l'extraction d'une racine est une opération exacte. Et on sait depuis Niels Abel et Evariste Galois que l'on ne peut pas étendre ce type de formules au calcul des racines d'un polynôme quelconque de degré supérieur.

Mais si les calculettes nous fournissent immédiatement la valeur $\alpha = \sqrt{a}$ pour un réel a positif, cette opération est en fait la résolution de l'équation non linèaire $x^2 - a = 0$, et s'obtient en calculant de proche en proche les itérés d'une suite qui tend vers α.

De façon plus générale, on peut être amené à rechercher une racine réelle d'une équation $F(x) = 0$, où F est une fonction quelconque. On devra construire une suite $(x_k)_{k\in\mathbb{N}}$ qui converge vers une solution du problème. Il est important de remarquer que l'on parle ici d'une solution : on n'a généralement pas unicité de solution, et il arrive que l'on en cherche plusieurs, ou qu'on les cherche toutes.

Il faudra alors :

1. localiser la ou les racines que l'on veut calculer,
2. s'assurer que la suite (x_k) converge vers une limite r telle que $F(r) = 0$,
3. prévoir un test d'arrêt des calculs, et estimer la précision qu'il garantit,
4. éventuellement, savoir comparer différentes méthodes.

2.1 Heron d'Alexandrie et les racines carrées

Egalement appelé Heron l'ancien, il vécut au premier siècle de notre ère et fut un des grands mécaniciens de l'antiquité ; il inventa diverses machines et instruments de mesure comme l'odomètre permettant de mesurer les distances parcourues en comptant le nombres de tours d'une roue, et dont le principe est encore utilisé par les compteurs kilomètriques des bicyclettes.

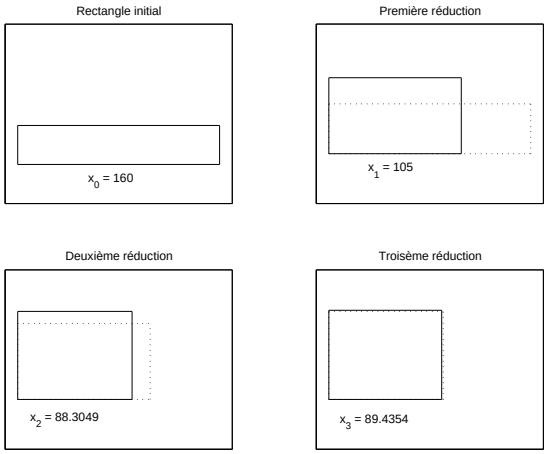
L'algorithme de calcul des racines carrées qu'on lui atribue était en fait déja connu des Babyloniens.

2.1.1 La methode de Héron :

Approche pragmatique :

Soit a un nombre positif dont on veut calculer la racine carrée $\alpha = \sqrt{a}$; α est la longueur du côté d'un carré dont l'aire est a. On va chercher à construire un tel carré en partant d'un rectangle, qu'on va peu à peu transformer pour le faire ressembler à un carré. Si l'on considère une valeur x_0 donnée comme une estimation grossière de α, on construit d'abord un rectangle $\mathcal{R}$ d'aire a ayant un côté de longueur $L = x_0$ et un second côté égal à $l = \dfrac{a}{L} = \dfrac{a}{x_0}$.

Pour construire un nouveau rectangle qui ressemble plus à un carré, il est raisonnable de choisir pour longueur d'un côté la moyenne des longueurs des côtés du rectangle précédent. Cette longueur sera $L = x_1 = \dfrac{1}{2}\,(x_0 + \dfrac{a}{x_0})$. Le second côté aura pour longueur $l = \dfrac{a}{L} = \dfrac{a}{x_1}$. Et l'on va continuer jusqu'à ce que notre appareil de mesure ne nous permette plus de distinguer les longueurs des deux côtés. La figure ci-dessous nous montre trois étapes de ce procédé :



Sur cet exemple, nous cherchons à calculer $\alpha = \sqrt{8000}$. La valeur initiale est $x_0 = 160$. Le rectangle initial a pour côtés 160 et 50.

Pour construire un rectangle qui ressemble plus à un carré, on le remplace par le rectangle dont un côté est la moyenne des deux précédents, c'est-à-dire $x_1 = \dfrac{1}{2}\,(x_0 + \dfrac{a}{x_0}) = 105$ de sorte que l'autre côté a pour longueur $\dfrac{8000}{105} = 76.1905$.

On va ensuite réitérer cette démarche en appliquant la formule

$$x_{k+1} = \frac{1}{2}\,(x_k + \frac{a}{x_k}) \tag{2.1}$$

x_k	$\frac{a}{x_k}$
160.00000000000000	50.00000000000000
105.00000000000000	76.19047619047619
90.59523809523810	88.30486202365309
89.45005005944560	89.43538874135297
89.44271940039928	89.44271879958389
89.44271909999159	89.44271909999158

On a obtenu la racine cherchée “à la précision machine”. Les chiffres corrects sont soulignés, et l'on constate que dès que l'on a obtenu 3 chiffres corrects (pour x_3), la convergence devient très rapide : 8 chiffres pour x_4 et 15 pour x_5 !

Le principe des approximations successives

On peut regarder cet algorithme d'un point de vue légèrement différent. Partant de l'équation $F(x) = x^2 - a = 0$, qu'on écrit de façon équivalente $x^2 = a$ ou $x = \frac{a}{x}$, on obtient successivement :

$$\begin{array}{lcl} x^2 & = & a \\ x^2 + a & = & 2\,a \\ \frac{1}{2}\,(x^2 + a) & = & a \\ \frac{1}{2}\,(x + \frac{a}{x}) & = & \frac{a}{x} \quad = \quad x \end{array}$$

On a remplacé l'équation $F(x) = 0$ par une équation équivallente $f(x) = x$, avec ici $f(x) = \frac{1}{2}\left(x + \frac{a}{x}\right)$, qui permet de retrouver la définition (2.1) de la méthode de Heron :

$$\left\{ \begin{array}{ll} x_0 \in \mathbb{R}, & \text{point de départ des itérations} \\ x_{k+1} = f(x_k), & k \geqslant 0. \end{array} \right.$$

Ces deux relations définissent les itérations de la méthode. C'est un exemple de méthode de point fixe : la limite $\alpha = \sqrt{a}$ de la suite ainsi définie par récurrence est telle que $\alpha = f(\alpha)$.

Ces deux relations restent imprécises sur deux points fondamentaux :
- le choix de x_0,
- le critère à vérifier pour arrêter les itérations : il n'est pas question de continuer indéfiniment les calculs.

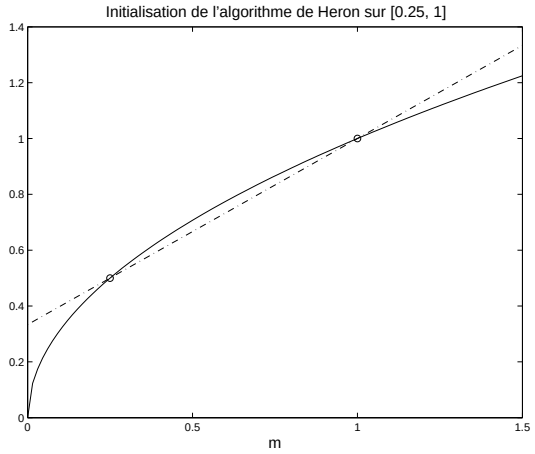
On va voir que pour notre algorithme, ces deux points peuvent être élucidés facilement.

Choix d'une valeur initiale

Notre choix de x_0 était assez empirique, et pour tout dire pas très heureux : le premier rectangle est un peu trop allongé. On peut essayer de rechercher une meilleure valeur pour x_0

Pour cela, on va écrire a sous la forme $a = m \times 4^p$, avec $\frac{1}{4} \leqslant m \leqslant 1$. Attention, m n'est pas tout à fait une mantisse car il est écrit en base 10. On aura alors $\alpha = \sqrt{m} \times 2^p$, et $0.5 \leqslant \sqrt{m} \leqslant 1$.

Il nous suffit maintenant de savoir initialiser les itérations pour rechercher $\sqrt{m}$. La figure suivante montre que le courbes représentant $s(x) = \sqrt{x}$ et la droite d'équation $y = \frac{1+2x}{3}$ qui passe par les points $\left(\frac{1}{4},\ \frac{1}{2}\right)$ et $(1,\ 1)$ sont assez proches.



On va donc choisir d'initialiser la recherche de $\sqrt{m}$ en utilisant la valeur $x_0 = \frac{1+2m}{3}$. Auparvant, il vaut trouver la valeur de p.

On remarque, mais l'ordinateur le fera très facilement pour nous, que

$$\frac{1}{4} \leqslant \frac{8000}{4^7} = 0.48828125 \leqslant 1,$$

de sorte que l'on choisira cette valeur pour m, avec $p = 7$. On obtient alors :

x_i	$\frac{m}{x_i}$
$\underline{0}.65885416666667$	$\underline{0}.74110671936759$
$\underline{0.69}998044301713$	$\underline{0.69}756413178540$
$\underline{0.69877}228740126$	$\underline{0.69877}019853767$
$\underline{0.6987712429}6946$	$\underline{0.6987712429}6790$
$\underline{0.69877124296868}$	$\underline{0.69877124296868}$

On retrouve $\sqrt{a} \simeq 0.69877124296868 \times 2^7 = 89.44271909999159$.

Arrêt des itérations

Supposons calculé un itéré x_i de notre suite, et évaluons :

$$x_{k+1} - \sqrt{m} = \frac{1}{2}\left(x_k + \frac{m}{x_k}\right) - \sqrt{m} = \frac{1}{2}\,\frac{x_k^2 - 2\,x_k\,\sqrt{m} + (\sqrt{m})^2}{(\sqrt{x_k})^2} = \frac{1}{2}\left(\frac{x_k - \sqrt{m}}{\sqrt{x_k}}\right)^2$$

de sorte que si $e_k = x_k - \sqrt{m}$ désigne l'erreur après k itérations, on aura :

$$e_{k+1} = \frac{1}{2\,x_k}\,e_k^2$$

Mais si $x_{k-1} \in [0.5,\ 1]$, alors $x_k \in [0.5,\ 1]$, de sorte que $2\,x_k \geqslant 1$ et :

$$e_{k+1} \leqslant e_k^2.$$

Par ailleurs, la fonction $e(x) = \sqrt{x} - \dfrac{1+2x}{3}$ a sa dérivée $e'(x) = \dfrac{1}{2\sqrt{x}} - \dfrac{2}{3}$ qui s'annule pour $x = \dfrac{9}{16}$. C'est donc pour $m = \dfrac{9}{16}$ que l'erreur e_0 sera maximale. Ce maximum de l'erreur est d'environ $0.041666 < 0.05$

On en déduit que $e_4 \leqslant e_3^2 \leqslant e_2^4 \leqslant e_1^8 \leqslant (0.05)^{16} < 2 \times 10^{-21}$, de sorte que 4 itérations suffisent à atteindre une précision meilleure que l'unité d'arrondi u.

2.2 Résolution d'une équation non linéaire

Au long de cette section, nous considérons la fonction

$$F(x) = 0.01\ \exp x + 10\ \cos x - 3\,x.$$

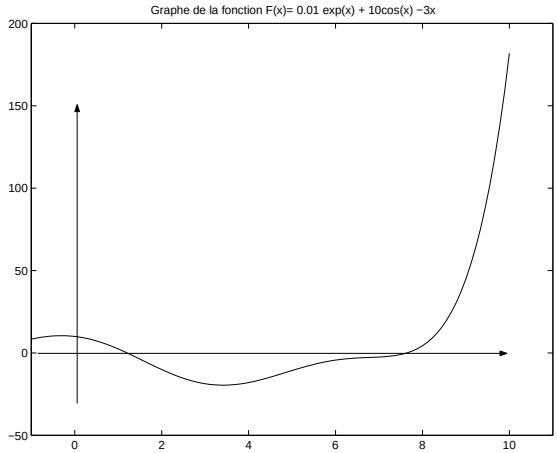
2.2.1 Localisation des racines

La recherche d'une solution de l'équation $F(x) = 0$ ne peut se faire que par une méthode itérative. On verra que l'on peut utiliser deux types de méthodes. Certaines travaillent sur des intervalles contenant la racine cherchée, d'autres avec une seule valeur (la méthode de Heron pour la recherche d'une racine carrée).

La méthode doit donc être alimentée par une ou deux valeurs initiales et à priori, ces valeurs doivent être assez proche de la racine cherchée

Représentation graphique

Une première façon de choisir une valeur initiale consiste à faire une représentation graphique. Voici une représentation graphique de F sur l'intervalle $[-1,\ 10]$.



On constate au vu de cette figure que notre fonction possède deux racines :
- la première est dans l'intervalle $[0,\ 2]$, plutôt vers le milieu,
- la seconde est dans l'intervalle $[7,\ 8]$, plus près de 8.

Le tracé d'une telle figure est particulièrement facile avec le logiciel MATLAB . Voici les commandes qu'il faut entrer pour obtenir le graphe de la fonction F

```
x = [-1:0.01:10];
y = 0.01*exp(x) + 10*cos(x) - 3*x;
z = zeros(size(x));
plot(x,y,x,z);
```

La figure sera à peu près la courbe représentée ci-dessus (sans le titre, l'axe des y et les flèches pour marquer les axes)

On peut expliciter et commenter ces instructions :
- `x = [-1 :0.01 :10];` construit un vecteur ligne contenant tous les points $x_i = -1 + i \times 0.01$ compris entre -1 et 10.
- `y =0.01*exp(x) + 10*cos(x) + 3*x;` construit un vecteur ligne contenant les valeurs $y_i = F(x_i)$; on remarquera que MATLAB calcule les valeurs des fonctions usuelles pour chaque élément d'un tableau qui lui est passé en argument.
- `z = zeros(size(x));` crée un vecteur de même taille que `x` contenant des zéros ; ce vecteur sera utilisé pour tracer l'axe des absisses.
- `plot(x,y,x,z);` crée une figure contenant deux courbes ; la première relie les points $(x_i,\ y_i)$ et la seconde les points $(x_i,\ 0)$ (axe des x).

Balayage systèmatique

Faute de possibilité de tracer facilement un graphe, on peut balayer systématiquement l'intervalle qui nous intèresse pour rechercher les éventuels changements de signe de F.

Pour notre exemple, on peut calculer les valeurs de $F(x)$ pour les valeurs entières de x. Les valeurs sont, dans l'ordre 8.40, 10.01, 2.43, -10.08, -18.69, -17.99, -10.67, -4.36, -2.49, 4.35, 44.91, 181.87.

On observe un changement de signe entre 1 et 2 ainsi qu'entre 7 et 8.

Arguments physiques

Pour des problèmes réels, certaines considérations physiques vous permettront de situer grossièrement les racines qui vous intéressent. C'est d'ailleurs la première reflexion qu'il faut faire ; elle permet de délimiter l'intervalle d'étude de la fonction, avant même la représentation graphique.

2.2.2 Méthode de dichotomie

Principe

Si la fonction continue F change de signe sur un intervalle $[a,\ b]$, c'est-à-dire si $F(a)\,F(b) < 0$, alors F s'annule au moins une fois sur cet intervalle. Cette remarque peut être utilisée pour former une suite d'intervalles de plus en plus petits qui contiennent une racine F.

En notant a_0, b_0 les bornes de cet intervalle et $m = \dfrac{a_0 + b_0}{2}$ le milieu de cet intervalle, on aura deux possibilités :

- soit $F(a_0)\, F(m) \leqslant 0$, et dans ce cas nous aurons une racine dans $[a_0,\ m]$; on pose $a_1 = a_0$ et $b_1 = m$;
- soit $F(b_0)\, F(m) \leqslant 0$, et dans ce cas nous aurons une racine dans $[m,\ b_0]$; on pose $a_1 = m$ et $b_1 = b_0$.

Dans les deux cas, le nouvel intervalle $[a_1,\ b_1]$ est de longueur $b_1 - a_1 = \dfrac{b_0 - a_0}{2}$. On recommence le même processus avec ce nouvel intervalle.

On construit ainsi une suite d'intervalles $[a_k,\ b_k]$ qui sont emboîtés et contiennent une racine de F. On arrêtera ces itérations lorsque la longueur $|a_k - b_k|$ de l'intervalle sera inférieure à une tolérance choisie à l'avance. On prendra pour racine la valeur $\tilde{r} = \dfrac{a_k + b_k}{2}$.

Mise en oeuvre

Voici les résultats obtenus pour notre fonction F en partant des intervalles $[1,\ 2]$ et $[7,\ 8]$.

k	a_k	b_k	a_k	b_k
0	1.000000000	2.000000000	7.000000000	8.000000000
1	1.000000000	1.500000000	7.500000000	8.000000000
2	1.000000000	1.250000000	7.500000000	7.750000000
3	1.125000000	1.250000000	7.625000000	7.750000000
4	1.187500000	1.250000000	7.625000000	7.687500000
5	1.187500000	1.218750000	7.625000000	7.656250000
6	1.203125000	1.218750000	7.625000000	7.640625000
7	1.203125000	1.210937500	7.632812500	7.640625000
8	1.203125000	1.207031250	7.636718750	7.640625000
9	1.203125000	1.205078125	7.638671875	7.640625000
10	1.204101562	1.205078125	7.639648437	7.640625000
11	1.204589843	1.205078125	7.639648437	7.640136718
12	1.204589843	1.204833984	7.639648437	7.639892578
13	1.204589843	1.204711914	7.639770507	7.639892578
14	1.204589843	1.204650878	7.639831542	7.639892578

Les racines calculées sont alors respectivement $\tilde{r_1} = 1.204620361$ et $\tilde{r_2} = 7.639862060$.

Convergence et arrêt des itérations

Sur les deux racines calculées précédement, la précision garantie est la même : en effet la longueur des derniers intervalles calculés est, dans un cas 6.1035×10^{-5} et dans l'autre 6.1036×10^{-5}.

Cette précision peut être prévue à l'avance. A l'initialisation, l'intervalle $[a_0,\ b_0]$ a pour longueur 1 ; à la première itération, sa longueur sera $\dfrac{1}{2}$, et à la k-ième, elle sera $\dfrac{1}{2^k}$.

Si l'on souhaite une précision de 10^{-4}, on cherchera k tel que $\dfrac{1}{2^k} \leqslant 10^{-4}$, soit encore $k \ln 2 \geqslant 4 \ln 10$, c'est à dire $k \geqslant 4\, \dfrac{\ln 10}{\ln 2} \simeq 13.2877$. On retrouve la valeur $k = 14$.

En fait, la précision effectivement atteinte est 0.5×10^{-4}, puisque la racine est choisie au milieu du dernier intervalle.

Ces remarques se généralisent, et peuvent s'énoncer de la façon suivante :

Théorème 5 *Si r désigne une racine de l'équation $F(x) = 0$ contenue dans l'intervalle $[a_0,\ b_0]$. La méthode de dichotomie appliquée à la recherche de cette racine est telle que :*

i) la racine approchée r_k calculée à l'itération k vérifie $|r_k - r| \leqslant \dfrac{|a_0 - b_0|}{2^{k+1}}$.

ii) le nombre k_ϵ d'itérations nécessaire pour atteindre une précision ϵ est tel que $k_\epsilon \geqslant \ln \dfrac{|a_0 - b_0|}{\epsilon} - 1$.

La borne de l'erreur est divisée par 2 à chaque itération. Lorsqu'une méthode itérative est telle qu'il existe une constante $c < 1$ pour laquelle $|r_{k+1} - r| \leqslant c\ |r_k - r|$, on dit qu'elle converge linéairement.

2.2.3 La méthode de Newton

Principe

Supposons maintenant que comme précédement, une représentation graphique nous ait indiqué que la valeur $x_0 = 8$ est proche d'une racine. Un développement de Taylor à l'ordre 1 en h au point x_0 s'écrit :

$$F(x_0 + h) = F(x_0) + h\,F'(x_0) + h\,\varepsilon(h) \tag{2.2}$$

La racine cherchée peut s'écrire $r = x_0 + h$ pour un h inconnu qui sera tel que :

$$F(x_0) + h\,F'(x_0) + h\,\varepsilon(h) = 0 \tag{2.3}$$

Comme h est assez petit, $\varepsilon(h)$ sera encore plus petit ; on va négliger le terme $h\,\varepsilon(h)$ et forcer l'égalité à 0 en posant

$$F(x_0) + h_1\,F'(x_0) = 0 \tag{2.4}$$

qui fournit $h_1 = -\dfrac{F(x_0)}{F'(x_0)}$.

On aura ainsi défini le premier itéré d'une nouvelle méthode en posant

$$x_1 = x_0 - \frac{F(x_0)}{F'(x_0)}.$$

Par ailleurs, on sait que la tangente à la courbe représentant $y = F(x)$ au point x_0 a pour équation

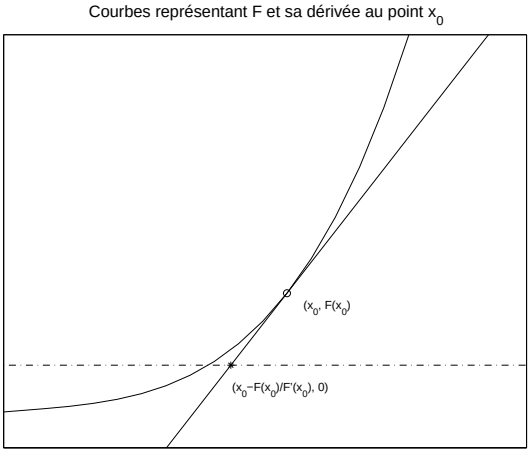
$$y = F(x_0) + (x - x_0)\,F'(x_0) \tag{2.5}$$

On remarque alors que cette tangente coupe l'axe des x au point x_1 : le choix de x_1 est donc la solution d'un modèle linéaire de notre problème au voisinage de x_0 !

Mise en oeuvre

La méthode de Newton va être définie en itérant cette démarche. L'algorithme consiste donc, partant de x_0 voisin d'une racine, à calculer pour $k \geqslant 0$:

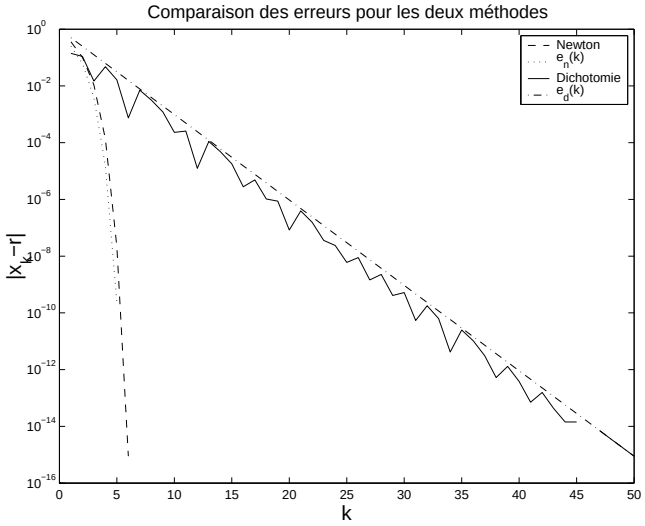
$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)} \tag{2.6}$$



Partant respectivement des valeurs $x_0 = 2$ et $x_0 = 8$, nous obtenons :

k	x_k	x_k
0	2.0000000000000000	8.0000000000000000
1	1.1607032574053444	7.7425762473069293
2	1.2049212788797627	7.6507460430283869
3	1.2046178784694039	7.6400156469715865
4	1.2046178652072419	7.6398801183259391
5	1.2046178652072419	7.6398800969514733
6	1.2046178652072419	7.6398800969514733

On constate que la suite a convergé, dans un cas à la quatrième itération, et dans l'autre à la cinquième ! La méthode de Newton est beaucoup plus rapide que la précédente. On peut aussi le voir en regardant les courbes représentant l'évolution de l'erreur pour chaque méthode dans la recherche de la racine.



On y remarque que l'erreur pour chacune de ces méthodes est comparable à une courbe :

- pour la méthode de Newton, c'est la courbe représentant $e_n(k) = (\frac{1}{2})^{2^k}$,
- pour la méthode de dichotomie, c'est la courbe représentant $e_d(k) = (\frac{1}{2})^k$.

Pour la méthode de dichotomie, on retrouve le principe de la convergence linéaire : à chaque itération, l'erreur est divisée par 2. Pour la méthode de Newton, c'est le principe observé pour la méthode de Heron que l'on retrouve : en effet, ici $|x_0 - r| \simeq \frac{1}{2}$, et au-delà, pour $k \geqslant 0$, la relation $|x_{k+1} - r| \simeq |x_k - r|^2$ entraine donc

$$|x_{k+1} - r| \simeq \left(|x_{k-1} - r|^2\right)^2,$$

qui permet, en remontant jusqu'à $k = 0$, de retrouver

$$|x_k - r| \simeq (\frac{1}{2})^{2^k}.$$

On dit qu'il y a convergence quadratique lorsque la méthode itérative pour calculer la racine r est telle que pour tout k et pour un $c < 1$, $|x_{k+1} - r| \leqslant c\,|x_k - r|^2$.

Remarque 6 *Il ne faut pas s'étonner de retrouver ici une propriété de la méthode de Heron. En effet, si l'on veut appliquer la méthode de Newton à la fonction $F(x) = x^2 - 2$, dont la dérivée est $F'(x) = 2\,x$, les itérations s'écrivent*

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2\,x_k} = \frac{x_k}{2} + \frac{1}{x_k} = \frac{1}{2}\,\left(x_k + \frac{2}{x_k}\right).$$

La méthode de Heron est une application particulière de la méthode de Newton.

On peut penser que la méthode de Newton est préférable à la méthode de dichotomie, cependant pour notre exemple,

- partant de $x_0 = 3$, on converge, assez vite d'ailleurs puisque c'est en 6 itérations, mais vers une racine indésirable $r = -2.35581727259318$ qui est en dehors de l'intervalle qui nous intéresse ;
- partant de $x_0 = 30$, on converge bien vers $r = 7.63988009695147$, mais la convergence est très lente : il faut 29 itérations. Il est vrai que ce choix d'initialisation est un peu absurde puisque $F(30) \simeq 10^{11}$.

Combiner deux méthodes

La méthode de Newton est incontestablement la meilleure lorsqu'on est assez proche de la racine recherchée. "Assez proche" dépend de la fonction F. On peut le caractériser lorsqu'on connait bien les dérivées première et seconde de F au voisinage de la racine. En général, cette caractérisation reste théorique et n'a que peu d'intérèt dans les cas réels.

On peut combiner les deux méthodes de facon pragmatique. A l'initialisation, on dispose d'un intervalle $[a_0,\ b_0]$ tel que $F(a_0)\,F(b_0) < 0$. On choisit une des bornes a_0 ou b_0 pour x_0. On calcule x_1 selon la méthode de Newton.

- Si $x_1 \notin [a_0,\ b_0]$, on le rejette et on effectue une itération de dichotomie pour trouver $[a_1,\ b_1]$.
- Si $x_1 \in [a_0,\ b_0]$, alors :
  - ř Si $F(x_1)\,F(a_0) < 0$, on pose $a_1 = a_0$ et $b_1 = x_1$.
  - ř Si $F(x_1)\,F(b_0) < 0$, on pose $a_1 = x_1$ et $b_1 = b_0$.
- On itère en partant de l'intervalle $[a_1,\ b_1]$.

Quand on veut éviter les dérivées

Un autre inconvénient de la méthode de Newton est que parfois les dérivées de F sont difficiles à calculer.

On va alors se rappeler que l'on peut approcher une dérivée par une formule de différences finies.

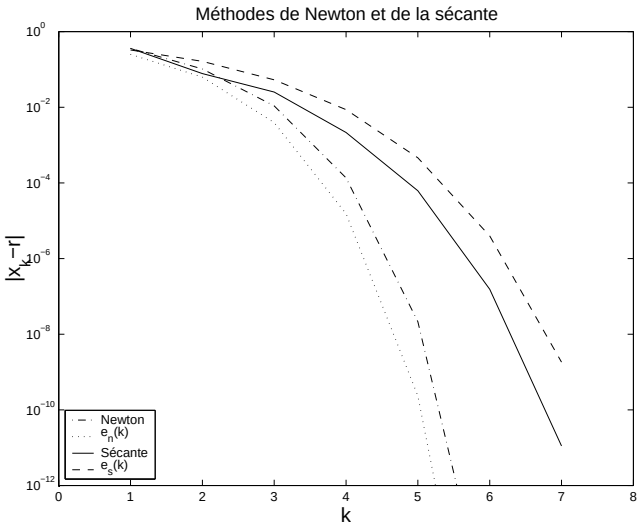
Supposons calculés les itérés jusqu'à l'étape k. La méthode de Newton demande de calculer $x_{k+1} = x_k - \dfrac{F(x_k)}{F'(x_k)}$.

L'idée de la méthode de la sécante consiste à remplacer $F'(x_k)$ par

$$d_k = \frac{F(x_k) - F(x_{k-1})}{x_k - x_{k-1}}.$$

Cette méthode nécessite deux valeurs pour être initialisée. Partant de x_0, on calculera un d_0 en posant, pour un h convenablement choisi $d_0 = \dfrac{F(x_0 + h) - F(x_0)}{h}$.

La courbe ci-dessous compare la méthode de Newton à celle de la sécante, initialisée en choisissant $h = -0.1$ pour $x_0 = 8$.



Sur cette figure, la courbe représentant $e_s(k)$ donne une idée de la convergence de la méthode de la sécante.Les valeurs $e_s(k)$ sont définies par $e_s(k) = (\dfrac{1}{2})^{d^k}$, avec $d = \dfrac{1+\sqrt{5}}{2}$.

Cette convergence n'est pas trop mauvaise par rapport à celle de la méthode de Newton.

Chapitre 3

Equations Différentielles Ordinaires

Les équations différentielles, et plus généralement les équations aux dérivées partielles, constituent l'outil de base pour la description des modèles mathémathiques de phénomènes rencontrés dans la nature. Le terme d'**équations différentielles ordinaires** est employé pour désigner les équations différentielles qui peuvent se mettre sous la forme

$$y'(t) = f(t, y). \tag{3.1}$$

par opposition à celles qui s'expriment seulement de façon implicite :

$$f(t, y, y'(t)) = 0. \tag{3.2}$$

3.1 Différents types de problèmes

3.1.1 Equations différentielles et conditions pour une solution

Au-delà de l'ordre de l'équation, c'est le type de problèmes posés qui différencie, et les équations différentielles, et leur analyse numérique.

- Les **problèmes à valeur initiale**, pour des équations du premier ordre :

$$\left\{ \begin{array}{lcll} y'(t) & = & f(t, y(t)) \, ; \quad t \in [a,\, b], & y \in \mathbb{R}^n, \\ y(a) & = & \alpha \in \mathbb{R}^n & . \end{array} \right. \tag{3.3}$$

- Les problèmes du second ordre peuvent être des **problèmes aux limites** comme par exemple :

$$\left\{ \begin{array}{l} y''(x) = f(x) \ ; \ x \in [a,\, b], \\ y(a) = \alpha \ ; \ y(b) = \beta. \end{array} \right.$$

- Les problèmes du second ordre, ou d'ordre supérieur peuvent être également des problèmes à valeur initiale ; on les transforme alors en problèmes d'ordre 1 de taille supérieure. Partant d'une équation différentielle, on forme un **système différentiel** ; c'est le cas pour le problème :

$$\left\{ \begin{array}{l} y''(x) = f(x) \, ; \, x \in [a,\, b], \\ y(a) = \alpha \, ; \, y'(a) = \beta. \end{array} \right.$$

que l'on écrira sous la forme :

$$\left\{ \begin{array}{lcl} Y'(x) & = & F(x,\, Y(x)) \ ; \ x \in [a,\, b], \\ Y(a) & = & Y_a, \end{array} \right.$$

avec $Y = \begin{pmatrix} y(x) \\ y'(x) \end{pmatrix}$ et $Y_a = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ sous la forme :

$$Y'(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} Y(x) + \begin{pmatrix} 0 \\ f(x) \end{pmatrix}.$$

Ce chapitre est consacré aux problèmes à valeurs initiales.

3.1.2 Le problème du pendule

Le mouvement d'un pendule de masse m, suspendu à un point O par un fil non pesant de longueur l, en rotation d'angle $\theta(t)$ autour de O est gouverné par l'équation :

$$\theta''(t) = -\frac{g}{l} \sin(\theta(t)). \tag{3.4}$$

L'angle θ est mesuré par rapport à une verticale passant par O. On s'intéresse au mouvement entre les instants $t_0 = 0$ et t_f. Les conditions initiales peuvent être :

$$\theta_0 = \frac{\pi}{3} \text{ rad}, \quad \theta'(0) = 0 \text{ rad/s}. \tag{3.5}$$

En général, on introduit $\omega^2 = \dfrac{g}{l}$. Ce problème est un problème non linéaire.

L'équation (3.4) est d'ordre 2, mais les conditions (3.5) en font un problème à valeurs initiales ; il doit donc être transformé en un système de deux équations différentielles du premier ordre. On pose $y_1(t) = \theta(t)$ et $y_2 = \theta'(t)$. On obtient :

$$\begin{array}{rcl} y_1'(t) & = & y_2(t) \\ y_2'(t) & = & -\omega^2 \sin(y_1(t)) \end{array}$$

Ce système entre dans le cadre de l'équation (3.3) avec $Y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ et

$$F = \begin{pmatrix} F_1(t,\, Y) \\ F_2(t,\, Y) \end{pmatrix} = \begin{pmatrix} y_2 \\ -\omega^2 \sin y_1 \end{pmatrix} \tag{3.6}$$

3.1.3 Problémes de Cauchy

Les problèmes du type (3.3) s'appellent problèmes de Cauchy lorsque la fonction f est continue sur $[a,\, b] \times \mathbb{R}^n$, la condition n'étant pas donnée nécessairement à gauche de l'intervalle. Nous allons simplement donner l'énoncé d'un théorème qui assure l'existence et l'unicité d'une solution à un problème à valeurs initiales. C'est le théorème de Cauchy-Lipschitz.

Théorème 6 *Si f est continue dans $[a,\, b] \times \mathbb{R}^n$, et s'il existe une constante $L \geqslant 0$ telle que $\|f(t,y) - f(t,z)\| \leqslant L \|y - z\|$ $\forall t \in [a,\, b]$, $\forall (y,\, z) \in \mathbb{R}^{2n}$ pour une norme de $\mathbb{R}^n$, le problème (3.3) admet une solution unique pour toute donnée initiale α.*

La seconde condition s'appelle condition de Lipschitz par rapport à la variable y. Bien que ce théorème nous assure de l'existence d'une solution, il est souvent illusoire de vouloir la calculer explicitement, ou l'exprimer à l'aide de fonctions usuelles.

3.1.4 Exemples

Le problème du pendule satisfait les hypothèses du théorème 6. La fonction F de (3.6) vérifie en effet, pour tous Y et Z de $\mathbb{R}^2$:

- $|F_1(Y) - F_1(Z)| = |y_2 - z_2| \leqslant \|Y - Z\|$,
- $|F_2(Y) - F_2(Z)| = \omega^2 \, |\sin(y_1) - \sin(z_1)| \leqslant \omega^2 \, \|Y - Z\|$, en utilisant $\sin a - \sin b = 2 \, \sin(\frac{a-b}{2}) \, \cos(\frac{a+b}{2})$.

On ne peut cependant pas expliciter sa solution à l'aide de fonctions usuelles. Cette solution peut s'exprimer à l'aide d'une fonction spéciale appelée fonction "sinus amplitude" de Jacobi, mais c'est beaucoup plus compliqué que de rechercher une approximation numérique de la solution en utilisant une des méthodes que nous allons exposer dans ce cours.

Le probléme :

$$\left\{ \begin{array}{lcl} y'(t) & = & y(t)^{1/2} \, ; \, t \in [0, \, 1], \\ y(0) & = & 0. \end{array} \right.$$

ne satisfait pas les hypothèses du théorème 6. Il admet au moins deux solutions distinctes $y_1(t) = 0$ et $y_2(t) = \left(\frac{t}{2}\right)^2$.

Le problème :

$$\left\{ \begin{array}{ll} y''' - 3y'' - y' \, y = 0 \, ; & t \geqslant 0 \\ y(0) = 0, \; y'(0) = 1, \; y''(0) = -1. & \end{array} \right.$$

est un problème de Cauchy : en introduisant $y_1 = y$, $y_2 = y'$ et $y_3 = y''$ et en posant $Y = (y_1, \, y_2, \, y_3)^T$, il s'écrit :

$$\left\{ \begin{array}{lcl} Y' & = & F(t, \, Y) \, ; \, t \geqslant 0 \\ Y(0) & = & Y_0. \end{array} \right.$$

avec $F(t, \, Y) = (y_2, \, y_3, \, 3 \, y_3 + y_2 \, y_1)^T$; elle ne satisfait pas la condition de Lipschitz globale $\|F(t, \, Y) - F(t, \, Z)\| \leqslant L \, \|Y - Z\|$ pour tous Y et Z dans $\mathbb{R}^3$, mais elle satisfait une condition de Lipschitz locale, c'est à dire lorsque Y et Z restent dans un domaine borné contenant Y_0 : c'est le cas lorsque F est continuement dérivable par rapport à Y, et ici, la matrice jacobienne de F (relativement à Y) s'écrit :

$$J_Y = \left(\begin{array}{ccc} \dfrac{\partial F_1}{\partial y_1} & \cdots & \dfrac{\partial F_1}{\partial y_3} \\ \cdots & & \cdots \\ \dfrac{\partial F_3}{\partial y_1} & \cdots & \dfrac{\partial F_3}{\partial y_k} \end{array} \right) = \left(\begin{array}{ccc} 0 & 1 & 0 \\ 0 & 0 & 1 \\ y_2 & y_1 & 3 \end{array} \right). \tag{3.7}$$

3.2 Approximation numérique des solutions

3.2.1 Principes généraux

Pour simplifier les notations, on supposera désormais que $a = 0$, et on notera $b = T$. On cherche des valeurs approchées de la solution $y(t)$ sur $[0, \, T]$. Pour cela, on se donne une partition $0 = t_0 < t_1 < ... < t_N = T$ et on cherche des y_i aussi voisin que possible de $y(t_i)$.

Les **pas de discrétisation** sont les réels $h_i = t_{i+1} - t_i$, et lorsque cela ne nuit pas à la qualité de la solution, n'implique pas de coûts de calculs trop importants ou plus généralement pour alléger l'exposé, on supposera la pas constant : on le notera alors $h = \dfrac{T}{N}$.

3.2.2 Les méthodes d'Euler

En intégrant l'équation (3.1) sur l'intervalle $[t_n, t_{n+1}]$, on obtient :

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} y'(s)\, ds = y(t_n) + \int_{t_n}^{t_{n+1}} f(s, y(s))\, ds. \tag{3.8}$$

On peut remplacer l'intégrale par une formule approchée, et si on choisit la formule des rectangles à gauche :

$$\int_{t_n}^{t_{n+1}} f(s, y(s))\, ds \simeq h\, f(t_n,\, y(t_n)),$$

on obtient la formule d'Euler explicite :

$$y_{n+1} = y_n + h\, f(t_n, y_n)\ ;\ 0 \leqslant n \leqslant N-1.$$

On peut choisir une formule des rectangles à droite, et dans ce cas, c'est la méthode d'Euler implicite que l'on obtient :

$$y_{n+1} = y_n + h\, f(t_{n+1}, y_{n+1})\ ;\ 0 \leqslant n \leqslant N-1.$$

Cette méthode est dite implicite car y_{n+1} ne s'obtient qu'après la résolution d'une équation.

Ces deux méthodes sont dites à un pas, car le calcul de y_{n+1} se fait seulement en fonctionde l'étape n (y_n, t_n) et du pas h.

3.2.3 Une mise en oeuvre réussie

Le problème :

$$\left\{ \begin{array}{lcl} y'(t) & = & y(t)\ \ 0 \leqslant t \leqslant 1, \\ y(0) & = & 1, \end{array} \right.$$

admet comme unique solution $y(t) = e^t$.

La méthode d'Euler explicite consiste à calculer, pour $n \geqslant 0$,

$$\begin{array}{rcl} y_{n+1} & = & y_n + h\, y_n \\ y_{n+1} & = & (1+h)\, y_n \end{array}$$

de sorte partant de $y_0 = 1$, on obtient $y_{n+1} = (1+h)^{n+1}$. Avec $h = \dfrac{1}{N}$, on retrouve pour y_N la valeur supposée approchée de $y(1) = e$, et on sait qu'en effet $\lim\limits_{N\to\infty} y_N = \lim\limits_{N\to\infty} (1 + \dfrac{1}{N})^N = e$.

La méthode implicite fournit quant à elle :

$$y_N = \frac{1}{(1 - \frac{1}{N})^N},$$

qui converge également vers e lorsque N tend vers l'infini.

3.2.4 Une mise en oeuvre défaillante

Un exemple surprenant Considérons le problème suivant :

$$\begin{cases} y'(t) &= 3y(t) - 3t \ ; \ 0 \leqslant t \leqslant 20, \\ y(0) &= \dfrac{1}{3}, \end{cases} \tag{3.9}$$

dont la solution est $y(t) = \dfrac{1}{3} + t$ et appliquons la méthode d'Euler pour en chercher la solution approchée en $t = 20$, à priori voisine de 20.33333. Pour cela, nous exécutons le code MATLAB suivant :

```
T = 20;
yini = 1/3;
Res= zeros(4,2);
N = [20 ; 40 ; 200 ; 400];% Nombre de pas de temps;
Res(:,1) = T./N;          % pas de temps;

for l = 1:4
   h = Res(l,1);
   t = 0;
   y = yini;
   for k = 1:N(l)
     y = y+3*h*(y-t);
     t = t+h;
   end;
   Res(l,2) = abs(1/3+t-y);% Erreur sur y(T);
end;
```

Les résultats obtenues peuvent surprendre. Ils sont résumés dans le tableau suivant :

h	1	0.5	0.1	0.05
erreur	$2.0345\ 10^{-5}$	0.4300	$6.5252\ 10^{3}$	$2.5143\ 10^{7}$

Pour $T = 10$, on obtient en double précision IEEE :

h	1	0.5	0.1	0.05
erreur	$1.9403\ 10^{-11}$	$4.7278\ 10^{-9}$	$2.6318\ 10^{-8}$	$1.8232\ 10^{-5}$

L'erreur est plus raisonnable mais, paradoxalement, elle est croissante avec N. Enfin, en choisissant $T = 5$, on obtient des résultats tout à fait raisonnables, bien que l'erreur soit encore croissante avec N :

h	1	0.5	0.1	0.05
erreur	$1.8651\ 10^{-14}$	$4.9560\ 10^{-13}$	$5.24026\ 10^{-13}$	$0.15525\ 10^{-11}$

Analyse de l'exemple Calculons la solution générale de l'équation différentielle :

$$y'(t) = 3y(t) - 3t$$

La solution de l'équation homogène est $y(t) = Ke^{3t}$. La méthode de variation de la constante fournit pour notre équation

$$\begin{aligned} K'(t) &= -3t\, e^{-3t}, \\ K(t) &= (t + \frac{1}{3})\, e^{-3t} + k, \\ y(t) &= t + \frac{1}{3} + k\, e^{3t}. \end{aligned}$$

La constante k peut être évaluée à l'aide de la valeur de y en 0, avec $k = 0$ si $y(0) = \frac{1}{3}$. Mais si $y(0) = \frac{1}{3} + \epsilon$, on obtient $k = \epsilon$ de sorte que la solution réelle du problème devient $y_\epsilon(t) = \frac{1}{3} + t + \epsilon\, e^{3t}$.

Ainsi, pour $t = 5$, on aura $y_\epsilon(5) = \frac{16}{3} + (y_0 - \frac{1}{3})\, e^{15}$. Pour une unité d'arrondi $u = 1.1102\ 10^{-16}$, en majorant $|y_0 - \frac{1}{3}|$ par $\frac{u}{3}$, les erreurs que l'on peut attendre sur l'estimation de $y(T)$ sont :

- pour $T = 5$, $e = 2.4196\ 10^{-10}$,
- pour $T = 10$, $e = 7.9096\ 10^{-4}$,
- pour $T = 20$, $e = 8.4526\ 10^{9}$.

On retrouve l'ordre de grandeur des erreurs constatées !

3.2.5 Comportement des solutions

La solution générale de l'équation linéaire homogène du premier ordre,

$$y' = a\, y. \tag{3.10}$$

est $y(t) = k\, e^{a\, t}$, la constante k étant déterminée par la condition initiale. On constate que toute solution tendra vers l'infini avec t si $a > 0$, vers 0 si $a < 0$ et restera constante si $a = 0$. L'exemple traité en 3.2.4 montre que ce comportement sera difficilement modifié par la solution particulière correspondant à un second membre ne dépendant que de t.

Plaçons nous à présent dans la cas d'un problème de Cauchy défini par une équation du second ordre, comme l'équation (1.2) du chapitre I :

$$y'' + \alpha\, y' + y = 0.$$

Cette équation sécrit comme un système du premier ordre,

$$Y' = \begin{pmatrix} y \\ y' \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ -1 & -\alpha \end{pmatrix} \begin{pmatrix} y \\ y' \end{pmatrix}. \tag{3.11}$$

Les valeurs propres $\lambda_{1,2}$ de la matrice $\begin{pmatrix} 0 & 1 \\ -1 & -\alpha \end{pmatrix}$ sont en fait les racines du polynôme caractéristique de l'équation (1.2), $r_{1,2} = \frac{-\alpha \pm \sqrt{\alpha^2 - 4}}{2}$. Dans ce cas, les deux composantes de la solution auront le comportement représenté sur la figure 3.1. La solution générale à valeurs réelles est telle que :

- si $\alpha^2 < 4$, les racines sont complexes et $y(t) = e^{-\frac{\alpha\, t}{2}}\,(a\,\cos\beta t + b\,\sin\beta t)$; le système a une composante périodique qui sera amortie si α est positif (cas stable) et amplifiée jusqu'à l'explosion sinon ;
- si $\alpha^2 > 4$, les racines sont réelles et le système n'a plus de comportement périodique. Sa solution est de la forme $y(t) = e^{-\frac{\alpha\, t}{2}}\,(a\,e^{\beta t} + b\,e^{-\beta t})$, avec $|\beta| < \left|\dfrac{\alpha}{2}\right|$, et le système reviendra ou non à son état d'équilibre selon que α sera négatif ou positif.

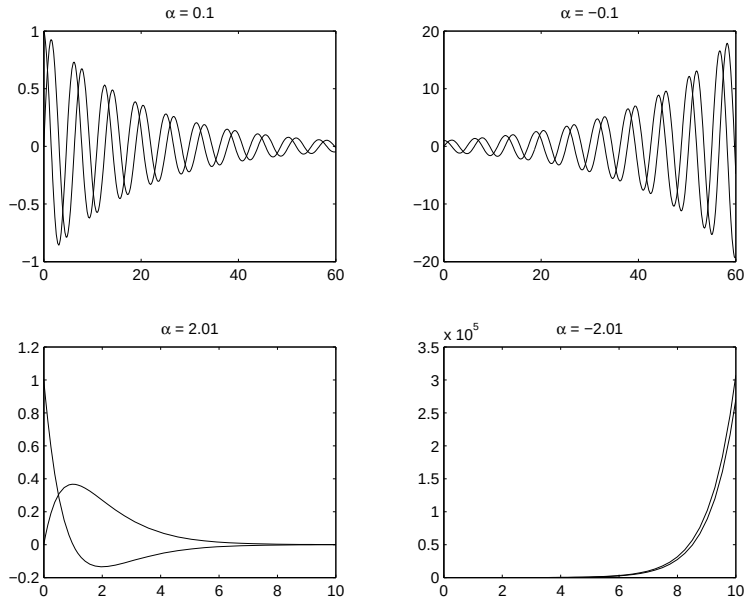


FIG. 3.1 – Comportement des solutions du système (3.11)

Lorsque le système différentiel est de dimension supérieure à 2, de la forme

$$Y'(t) = A\,Y(t), \tag{3.12}$$

le comportement des différentes composantes de la solution dépend encore des valeurs propres de la matrice A. Lorsque cette matrice est diagonalisable, c'est-à-dire lorsqu'il existe une matrice P inversible telle que $P^{-1}\,A\,P = \Lambda$ soit une matrice diagonale, dont les éléments diagonaux sont les valeurs propres de A, on obtient

$$P^{-1}\,Y'(t) = P^{-1}\,A\,P\,P^{-1}\,Y(t) = \Lambda\,P^{-1}\,Y(t),$$

de sorte que si $X(t) = P^{-1}\,Y(t)$ représente la fonction $Y(t)$ dans la base des colonnes de P, ses composantes $x_i(t)$ seront de la forme

$$x_i(t) = k_i e^{\lambda_i\, t}. \tag{3.13}$$

Remarque 7 *Le problème peut se compliquer lorsque le système différentiel n'est plus linéaire. On ne peut alors parler que de comportement local des solutions. Au voisinage d'un point donné $(Y_d,\, t_d)$, un développement de Taylor de la fonction F s'écrira :*

$$F(t,\, Y) = F(t_d,\, Y_d) + \partial_t F(t_d,\, Y_d)\,(t - t_d) + J_Y(t_d,\, Y_d)\,(Y - Y_d) + \ldots \tag{3.14}$$

où $\partial_t F$ a pour composantes les dérivées partielles $\partial_t F_i$ des composantes de F, et J_Y désigne la matrice jacobienne relativement à Y définie de façon analogue à (3.7).

Les dérivées partielles en t correspondent à des solutions particulières, de sorte que c'est la matrice jacobienne J_Y qui peut renseigner sur le comportement des solutions. Pour le problème du pendule, par exemple, (3.6) fournit

$$J_Y = \begin{pmatrix} 0 & 1 \\ -\omega^2 \cos y_1 & 0 \end{pmatrix}.$$

$y_1(t) = \theta(t)$ désigne l'angle du pendule par rapport à la verticale, et la physique le contraint à rester compris entre $-\frac{\pi}{2}$ et $\frac{\pi}{2}$. Les valeurs propres sont toujours imaginaires pures, ce qui correspond, en chaque instant, à une trajectoire périodique, mais ces valeurs propres ne sont pas constantes, et la solution très stable est quand même plus compliquée qu'une simple combinaison de sinus et de cosinus.

Cette situation est très favorable, car en général J_Y ne donne pas d'information utilisable pour analyser le comportement global de la solution.

3.3 Etude générale des méthodes à un pas

3.3.1 Schémas à un pas

On s'intéresse au problème :

$$\left\{ \begin{array}{rcl} y'(t) & = & f(t,\, y(t))\ ; \quad 0 \leqslant t \leqslant T, \\ y(0) & = & \alpha. \end{array} \right. \tag{3.15}$$

où l'on suppose que les hypothéses du théoréme 6 sont vérifiées. On parle de méthode à un pas lorsqu'à l'étape n, on n'utilise que la valeur approchée à cet étape pour évaluer y_{n+1}. Dans la cas d'un pas $h = \frac{T}{N}$ constant, la forme générale des schémas à un pas est donc la suivante :

$$\left\{ \begin{array}{rcl} y_0 & = & \alpha, \\ y_{n+1} & = & y_n + h\,\Phi(t_n,\, y_n,\, h), \quad 0 \leqslant n \leqslant N-1. \end{array} \right. \tag{3.16}$$

C'est la fonction Φ qui caractérise le schéma. Pour la méthode d'Euler explicite, on a $\Phi(t\,y\,,h\,) = f(t,\, y)$.

Ce schéma sera convergent si les valeurs y_n qu'il calcule se rapprochent des valeurs $y(t_n)$ de la solution de (3.15) aux instants t_n, pour n assez grand.

Pour caractériser cette convergence, on introduit les **erreurs de résolution** approchée,

$$e_n = y(t_n) - y_n, \tag{3.17}$$

qui sont dominées par l'erreur globale

$$E_N = \max_{0 \leqslant n \leqslant N} |e_n| \tag{3.18}$$

3.3.2 Erreur globale et erreurs locales de discrétisation

Comme la solution $y(t)$ est inconnue, les erreurs e_n ne seront pas accessibles. Ces erreurs sont en fait des erreurs cumulées qui correspondent à la propagation d'erreurs locales. La méthode donne accés aux seuls y_n. Une façon de définir des erreurs locales consiste à poser

$$\delta_{n+1} = z_n(t_{n+1}) - y_{n+1} \tag{3.19}$$

où z_n est la solution du problème

$$\left\{ \begin{array}{rcl} z_n'(t) & = & f(t,\, z_n(t))\ ; \quad t_n \leqslant t \leqslant T, \\ z_n(t_n) & = & y_n. \end{array} \right. \tag{3.20}$$

Les δ_n s'appellent **erreurs locales de discrétisation**.

Les $z_n(t)$ sont des solutions de la même équation différentielle que $y(t)$, chacune d'entre elles étant celle qui passe par y_n à l'instant t_n. On obtient alors

$$\delta_{n+1} = \begin{cases} z_n(t_{n+1}) - z_n(t_n) - h\,\Phi(t_n,\, z_n(t_n),\, h), \\ z_n(t_{n+1}) - y_n - h\,\Phi(t_n,\, y_n,\, h). \end{cases} \tag{3.21}$$

En fait, (3.21) montre que δ_{n+1} peut-être considérée comme une erreur locale de consistance : elle mesure avec quelle précision le schéma approche localement une solution de l'équation différentielle.

Sous la forme (3.19), δ_n mesure la contribution à l'erreur ajoutée par l'étape n. On pourrait alors penser que la convergence vers 0 avec h de $\displaystyle\sum_{n=1}^{N} |\delta_n|$ suffise à garantir la convergence de la méthode. Cela n'est pas le cas. Cette contribution pourra être amplifiée ou réduite selon que le problème aura tendance à amplifier ou à atténuer les perturbations. Cette tendance pourra varier dans le cas de problèmes non linèaires. Elle caractérise la stabitlité du problème.

Pour les problèmes linéaires, cette tendance ne varie pas. Intéressons nous au problème :

$$\begin{cases} y'(t) &= y - t^2 + \dfrac{7\,t - 3}{2}\;; \quad 0 \leqslant t \leqslant 1.6, \\ y(0) &= 0. \end{cases} \tag{3.22}$$

C'est un problème linéaire du premier ordre à coefficients constants dont la solution $y(t) = t^2 - \dfrac{3\,t}{2}$ se calcule facilement.

La figure 3.2 montre 4 étapes de la méthode d'Euler explicite appliquée à ce problème, avec un pas $h = 0.4$. Les valeurs y_n, $1 \leqslant n \leqslant 4$ sont pointées par les flèches qui illustrent la trajectoire de la méthode. Cette figure fait également apparaitre les trajectoires des différentes solutions exactes $z_n(t),\ 0 \leqslant n \leqslant 3$ figurant dans l'expression (3.19), avec bien sur $z_0(t) = y(t)$.

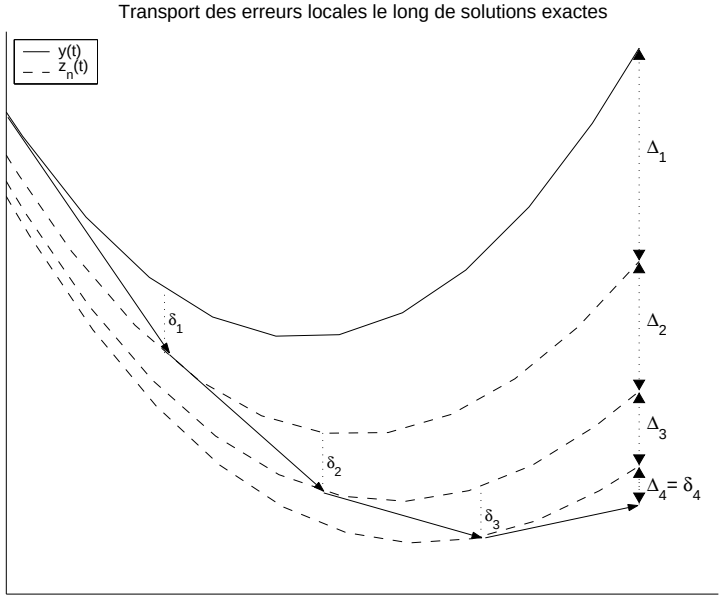


FIG. 3.2 – Les z_n transportent les erreurs locales de discrétisation

Les erreurs locales (3.19) ne sont pas simplement additionnées pour contribuer à l'erreur globale : elles sont d'abord "transportées" par les solutions z_n pour fournir des contributions Δ_n à

l'erreur finale e_4 qui vérifie :

$$e_4 = \sum_{n=1}^{4} \Delta_n.$$

Sur notre exemple, ce "transport" a plutôt enrichi les erreurs locales.

3.3.3 Stabilité du problème

Le transport des erreurs locales de discrétisation par les solutions $z_n(t)$ ne les enrichit pas nécessairement, et cela ne dépend pas de la solution $y(t)$.

On vérifiera en exercice que la fonction $y(t) = t^2 - \dfrac{3\,t}{2}$ est également solution des problèmes :

$$\left\{ \begin{array}{rcl} y'(t) & = & \dfrac{4}{3}(y - t^2) + 4t - \dfrac{3}{2}\ ; \\ y(0) & = & 0. \end{array} \right. \qquad \left\{ \begin{array}{rcl} y'(t) & = & \dfrac{4}{3}(t^2 - y) - \dfrac{3}{2}\ ; \\ y(0) & = & 0. \end{array} \right. \tag{3.23}$$

En appliquant à nouveau la méthode d'Euler implicite avec un pas $h = 0.4$ à ces deux problèmes, on obtient les résultats représentés par la figure 3.3. On constate que dans un cas les erreurs locales de discrétisation sont clairement amplifiées, et dans l'autre clairement atténuées.

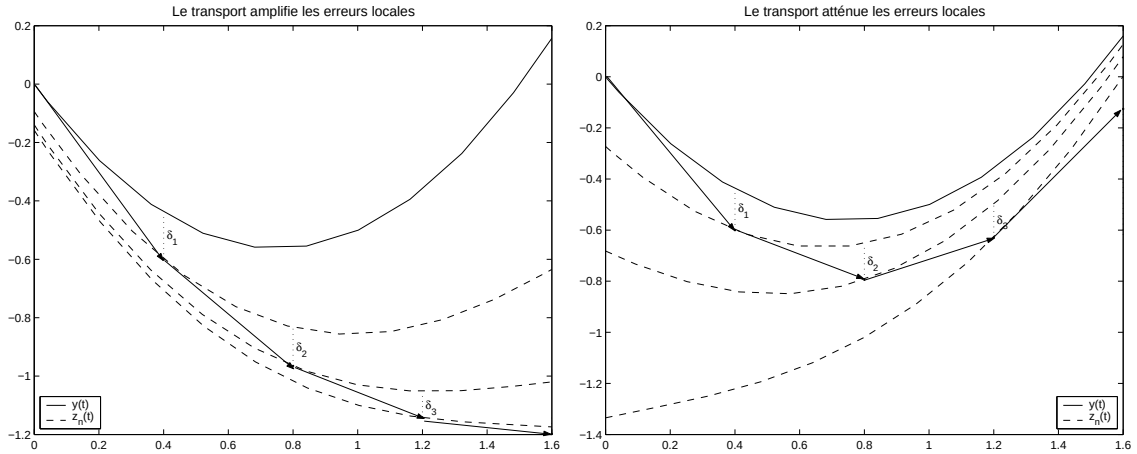


FIG. 3.3 – Les erreurs locales de discrétisation peuvent être amplifiées ou atténuées

Ces deux problèmes sont définis par des équations différentielles linéaires. La dérivée partielle $\partial_y f$ est constante, et c'est elle qui permet de savoir si les erreurs locales seront amplifiées ou atténuées. Dans le cas "stable", elle est ègale à $-\dfrac{4}{3}$, et dans le cas "instable" à $+\dfrac{4}{3}$.

Dans le cas général, ce comportement va varier au cours du calcul. On peut montrer qui si $\partial_y f(t,\, y) < 0$, alors on aura $\Delta_n \leqslant \delta_n$ pour tout n.

On ne peut pas se contenter de résoudre les seuls problèmes "stables".

La caractérisation (3.21) permet d'obtenir une estimation de l'erreur locale de discrétisation en cours de calcul, car elle s'appuie sur les valeurs y_n que l'on a calculées. Cette notion sera très utile pour mettre un place des stratégies de contrôle du pas : selon que le problème sera plus ou moins stable au point courant, on décidera de choisir un pas plus petit.

On choisira un pas assez petit pour que la contribution de cette erreur locale reste acceptable.

3.3.4 Majoration de l'erreur

Nous avons indiqué que (3.21) représente une forme d'erreur locale de consistance. Pour tenir compte du fait qu'il ne s'agit que d'erreurs locales, on dira que la méthode est **consistante** si

$$\lim_{h\to 0} \sum_{n=1}^{N} |\delta_n| = 0. \tag{3.24}$$

Comme cette somme fait intervenir $N = \dfrac{T}{h}$ termes, la condition de consistance sera qu'il existe une constante K telle que $|\delta_n| \leqslant K\, h^2$ pour tout n.

Cette convergence peut être plus ou moins rapide. La notion d'**ordre d'un schéma** va permettre de caractériser la consistance.

On dira que la méthode est d'ordre p s'il existe une constante K telle que :

$$|\delta_n| \leqslant K\, h^{p+1}. \tag{3.25}$$

Remarque 8 *En utilisant (3.20), (3.21) et un dévéloppement à l'ordre* 1 *de* $z_n(t_{n+1})$*, on obtient*

$$\delta_{n+1} = h\ (f(t_n,\, y_n) - \Phi(t_n,\, y_n,\, h)) + \mathcal{O}(h^2) \tag{3.26}$$

On en déduit que la méthode est consistante si et seulement si, pour tout $t \in [0, T]$ *et tout* $y \in \mathbb{R}$ *:*

$$\Phi(t, y, 0) = f(t, y). \tag{3.27}$$

Si les fonctions f *est* Φ *sont suffisament régulières, on peut obtenir des conditions d'ordre* p *en poussant plus loin le développement de Taylor (3.26).*

On a remarqué que les erreurs δ_n sont transportées par les solutions de (3.20). Nous allons essayer de donner une majoration des contributions Δ_n à l'erreur finale e_N.

A un instant $t > t_{n+1}$, Δ_{n+1} s'écrit comme une fonction de t :

$$\Delta_{n+1}(t) = |z_n(t) - z_{n+1}(t)| \tag{3.28}$$

$$\Delta_{n+1}(t) = \left| z_n(t_{n+1}) - z_{n+1}(t_{n+1}) + \int_{t_{n+1}}^{t} \left(z_n'(s) - z_{n+1}'(s)\right)\, ds \right|. \tag{3.29}$$

De la définition (3.19) et de (3.20), on tire :

$$\Delta_{n+1}(t) \leqslant \delta_{n+1} + \int_{t_{n+1}}^{t} |f(s,\, z_n(s)) - f(s,\, z_{n+1}(s))|\, ds. \tag{3.30}$$

En supposant que f satisfait les hypothèses du théoreme 6, on obtient

$$\Delta_{n+1}(t) \leqslant \delta_{n+1} + L \int_{t_{n+1}}^{t} |z_n(s) - z_{n+1}(s)|\, ds. \tag{3.31}$$

On pose alors $v_n(t) = \displaystyle\int_{t_{n+1}}^{t} |z_n(s) - z_{n+1}(s)|\, ds$. La fonction v_n est telle que $v_n(0) = 0$, et vérifie :

$$v_n'(t) = \Delta_{n+1}(t) \leqslant \delta_{n+1} + L\, v_n(t). \tag{3.32}$$

(3.32) s'écrit également $v_n'(t) - L\, v_n(t) \leqslant \delta_{n+1}$ qui fournit, en multipliant par e^{-Lt} :

$$\left(v_n'(t) - L\, v_n(t)\right)\, e^{-Lt} = \frac{d}{dt}\left(v_n(t)\, e^{-Lt}\right) \leqslant \delta_{n+1}\, e^{-Lt}. \tag{3.33}$$

En intégrant de t_{n+1} à t, et comme $v_n(t_{n+1}) = 0$:

$$v_n(t)\, e^{-Lt} \leqslant \frac{\delta_{n+1}}{L}\left(e^{-Lt_{n+1}} - e^{-Lt}\right). \tag{3.34}$$

$$v_n(t) \leqslant \frac{\delta_{n+1}}{L}\left(e^{L(t-t_{n+1})} - 1\right). \tag{3.35}$$

L'inégalité (3.31) fournit finalement :

$$\Delta_{n+1}(t) \leqslant \delta_{n+1} + L\,\frac{\delta_{n+1}}{L}\left(e^{L(t-t_{n+1})} - 1\right) = \delta_{n+1}\, e^{L(t-t_{n+1})}. \tag{3.36}$$

Remarque 9 *Ce résultat est une variante d'un lemme très célèbre connu sous le nom de lemme de Gronvall. Un lemme est à priori un résultat annexe qui sert à la démonstration d'un théorème. Comme quelques autres, celui-ci est beaucoup plus utile que bien des théorèmes !*

La relation (3.36) va nous permettre de majorer l'erreur $e_N = \displaystyle\sum_{n=1}^{N} \Delta_n(T)$ pour une méthode d'ordre p, avec $T = t_N = N\, h$. Dans ce cas, suivant (3.25),

$$e_N \leqslant K\, h^{p+1} \sum_{n=1}^{N} e^{L(T-t_{n+1})} = K\, h^p\, e^{LT} \sum_{n=1}^{N} h\, e^{-L\, t_{n+1}}. \tag{3.37}$$

En remarquant que la somme de droite est la formule composée des rectangles à droite pour l'intégrale $\displaystyle\int_0^T e^{-L\, s}\, ds = \frac{1 - e^{-LT}}{L}$, et que cette intégrale domine la somme, on obtient

$$e_N \leqslant \frac{K}{L}\, h^p\left(e^{LT} - 1\right). \tag{3.38}$$

Remarque 10 *La démonstration "classique" de la convergence des méthodes s'inspire des démonstration de Cauchy (1864) et Runge (1905).*

Dans ce cadre, on a coutume de définir une **erreur locale de consistance** *pour le schéma (3.16) appliqué au problème (3.15) en posant par exemple*

$$\tau_n(h) = y(t_{n+1}) - y(t_n) - h\,\Phi(t_n,\, y(t_n),\, h). \tag{3.39}$$

Contrairement aux erreurs locales de discrétisation, ces erreurs locales de consistances ne peuvent pas être estimées de façon fiable en utilisant les valeurs calculées. En effet, elles s'appuient sur les valeurs $y(t_n)$ de la solution $y(t)$ et non sur les y_n.

Elles fournissent pourtant elles aussi une contribution (que l'on notera encore Δ_n) à l'erreur finale. Mais cette fois, elles sont "transportées" par le schéma, suivant le principe illustré par la figure 3.4.

Il faut alors une hypothèse de stabilité de la méthode pour obtenir une majoration analogue à (3.38). Le schéma défini par Φ sera stable si il existe une constante $M > 0$ telle que pour tous t et h dans $[0, T]$:

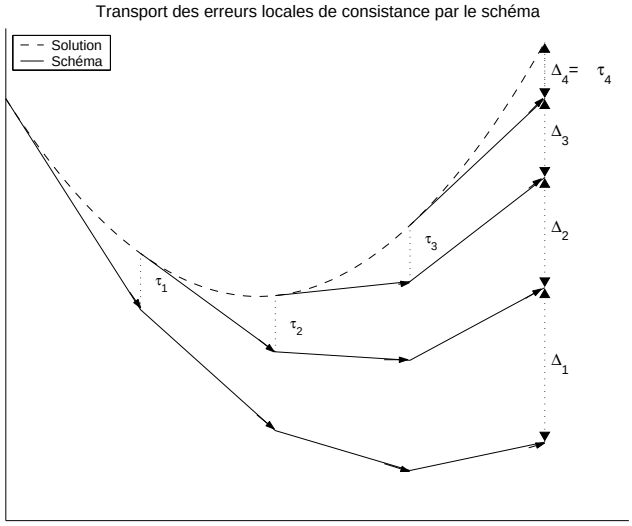


FIG. 3.4 – Le schéma transporte les erreurs locales de consistance

$$|\Phi(t,y,h) - \Phi(t,z,h)| \leqslant M\,|y-z|\ ;\ \forall (y,\,z) \in \mathbb{R}^2. \tag{3.40}$$

Cette condition est en fait un analogue des hypothèses du théorème 6, mais relativement à Φ plutôt qu'à f. Lorsque le schéma est consistant, (3.40) sera en général vérifiée lorsque f satisfait les hypothèses de ce théorème.

3.3.5 Calcul en arithmétique finie

On va supposer que les calculs sont perturbés par des erreurs d'arrondi, ce qui va introduire un terme supplémentaire aux erreurs locales de discrétisation. On aura à présent

$$\delta_{n+1} \leqslant K\,h^{p+1} + \epsilon_{n+1}. \tag{3.41}$$

On aura également une erreur initiale ϵ_0. Cette erreur correspond à l'écriture de la condition initiale en arithmétique fine : c'est par exemple l'écart entre $\frac{1}{3}$ et fl($\frac{1}{3}$) pour l'exemple (3.9). Cette erreur sera transportée par la solution du problème perturbé, pour fournir une contribution $\Delta_0 = \epsilon_0\,e^{L\,T}$ à l'erreur finale, selon le principe de la démonstration précédente ou de l'analyse de l'exemple (3.9).

En supposant que les erreurs d'arrondi des étapes 1 à N sont majorées par une valeur σ, on obtiendra alors une majoration de la forme

$$e_N \leqslant \epsilon_0\,e^{L\,T} + (K\,h^p + \frac{\sigma}{h})\,e^{L\,T}\sum_{n=1}^{N} h\,e^{-L\,t_{n+1}} \leqslant e^{L\,T}\left(\epsilon_0 + C\,h^p + \frac{\sigma}{h}\right). \tag{3.42}$$

On remarque un terme en $\frac{1}{h}$ qui tend vers l'infini lorsque h tend vers 0. En fait, l'exemple suivant va nous montrer que lorsque le problème est suffisament stable, cette nouvelle majoration ne peut dégrader que de façon anecdotique la convergence d'une méthode.

On considère le problème :

$$\left\{\begin{array}{lcl} y'(t) & = & y(t),\ \ 0<t<1, \\ y(0) & = & 1. \end{array}\right.$$

Sa solution est $y(t) = e^t$ et vérifie $y(1) = e$. On va appliquer à ce problème la méthode d'Euler modifiée, en posant $y_0 = 1$, et pour $0 \leqslant n \leqslant N-1$:

$$y_{n+1} = (1 + h + \frac{h^2}{2})\, y_n.$$

Dans ce cas particulier, l'erreur locale de discrétisation peut être calculée par un développement limité :

$$\delta_n \quad = \quad y_n\, e^h - y_n(1 + h + \frac{h^2}{2}) \simeq y_n\, \frac{h^3}{6}.$$

On réalise alors les calculs en utilisant MATLAB, c'est-à-dire en arithmétique IEEE pour laquelle l'unité d'arrondi est $u \simeq 1.11\,10^{-16}$. On convient de choisir $\epsilon_0 = \sigma = u$. La fonction qui donne la majoration empirique de l'erreur résultant de (3.42) est alors, avec ici $L = T = 1$:

$$e(h) = e * \left(u + \frac{e}{6}\, h^2 + \frac{u}{h} \right).$$

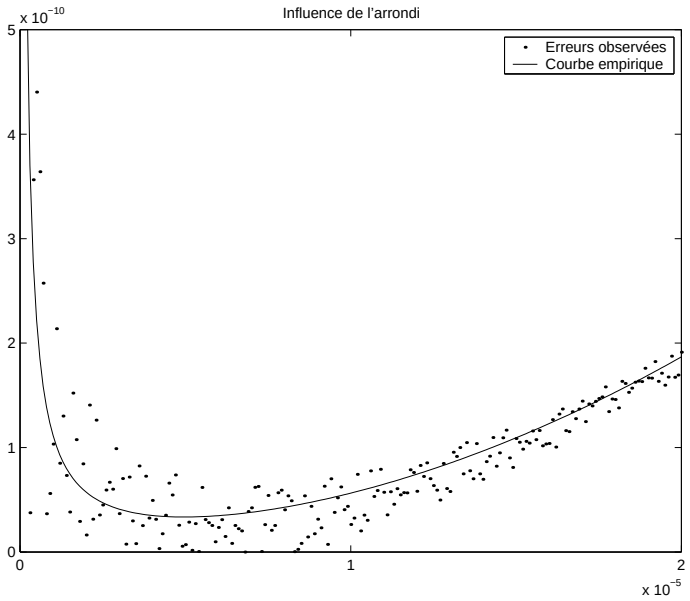


FIG. 3.5 – Estimation et observation des erreurs d'arrondi

La figure 3.5 représente les erreurs observées pour 200 valeurs du pas h régulièrement réparties entre 10^{-7} et $2\,10^{-5}$: elle fait clairement apparaitre que le comportement des erreurs observées est globalement bien représenté par la courbe empirique.

On constate aussi que l'influence des erreurs d'arrondi est négligeable : il est inutile de choisir un pas trop petit, mais les risques d'erreurs proviennent surtout de l'éventuelle instabilité du problème.

3.3.6 Les méthodes de type Runge-Kutta

La méthode d'Euler modifiée, ou méthode du point milieu

Dans la méthode d'Euler explicite, on remplace l'intégrale figurant dans (3.8) par $hf(t_n, y_n)$, c'est-à-dire en utilisant la valeur de f à gauche de l'intervalle d'intégration. La méthode d'Euler implicite utilise l'extrèmité droite.

On peut chercher un compromis en remplaçant cette intégrale par $h\, f(t_n + \frac{h}{2}, \bar{y}_n)$, où $\bar{y}_n$ doit être une valeur approchée de $y(t_n, \frac{h}{2})$, c'est-à-dire au centre de l'intervalle.

Pour obtenir une valeur $\bar{y}_n$, on peut utiliser la méthode d'Euler, et poser

$$\bar{y}_n = y_n + \frac{h}{2} f(t_n, y_n).$$

Le schéma s'écrit donc :

$$y_{n+1} = y_n + h\, f(t_n + \frac{h}{2}, y_n + \frac{h}{2} f(t_n, y_n)). \tag{3.43}$$

La fonction qui le définit est $\Phi(t, y, h) = f(t + \frac{h}{2},\, y + \frac{h}{2} f(t, y))$

La méthode est consistante, puisque $\Phi(t, y, 0) = f(t, y)$. On peut vérifier qu'elle est d'ordre 2.

On peut vérifier que cette méthode est stable si f satisfait les conditions du théorème 6, de sorte que le schéma proposé est convergent, d'ordre 2.

Forme générale des méthodes de Runge-Kutta

Introduction On peut reprendre la méthode (3.43), en l'écrivant sous la forme :

$$\left\{ \begin{array}{lcl} y_{n+\frac{1}{2}} & = & y_n + \frac{h}{2} f(t_n,\, y_n), \\ y_{n+1} & = & y_n + h\, f(t_n + \frac{h}{2}\, y_{n+\frac{1}{2}}). \end{array} \right. \tag{3.44}$$

Sous cette forme, elle rentre dans le cadre d'une famille de méthodes appelées méthodes de Runge-Kutta. Il s'agit ici d'une méthode à deux étages.

On peut également penser à utiliser la formule des trapèzes pour approcher l'intégrale figurant dans (3.8). On obtient alors un schéma implicite défini par :

$$y_{n+1} = y_n + \frac{h}{2} \left(f(t_n,\, y_n) + f(t_{n+1},\, y_{n+1}) \right).$$

Si l'on souhaite éviter le caractère implicite de ce schéma, on remplace $f(t_{n+1},\, y_{n+1})$ par l'estimation qu'en donne le schéma d'Euler. On obtient la méthode d'Euler améliorée :

$$y_{n+1} = y_n + \frac{h}{2} \left(f(t_n,\, y_n) + f(t_{n+1},\, y_n + h\, f(t_n,\, y_n)) \right). \tag{3.45}$$

Cette écriture est très compacte, cela risque d'être bien pire si l'on souhaite raffiner ce type de construction. On préfère décrire ce schéma sous une forme algorithmique :

$$\left\{ \begin{array}{lcl} k_1 & = & f(t_n, y_n), \\ k_2 & = & f(t_n + h, y_n + h\, k_1), \\ \Phi(t_n,\, y_n,\, h) & = & \frac{1}{2}(k1 + k_2) \\ y_{n+1} & = & y_n + h\, \Phi(t_n,\, y_n,\, h). \end{array} \right. \tag{3.46}$$

Forme générale Plus généralement, on définit un schéma de Runge-Kutta a q étages en se donnant :

- une partition $0 \leqslant c_1 \leqslant \ldots \leqslant c_q \leqslant 1$ de $[0,1]$,
- pour chaque c_i, une formule d'intégration approchée :

$$\int_0^{c_i} f(s)\,ds \simeq \sum_{j=1}^{q} a_{i,j}\, f(c_j),$$

- une formule d'intégration approchée sur $[0,\ 1]$:

$$\int_0^{1} f(s)\,ds \simeq \sum_{j=1}^{q} b_j\, f(c_j).$$

Le schéma s'écrit alors, avec des notations analogues à (3.46) :

$$\left\{\begin{array}{lcll} k_1 & = & f(t_n, y_n), & \text{si } c_1 = 0, \\ k_i & = & f(t_n + c_i\, h,\ y_n + h \displaystyle\sum_{j=1}^{q} a_{i,j}\, k_j), & 2 \leqslant i \leqslant q \\ \Phi(t_n,\ y_n,\ h) & = & \displaystyle\sum_{j=1}^{q} b_j\, k_j, & \\ y_{n+1} & = & y_n + h\,\Phi(t_n,\ y_n,\ h). & \end{array}\right. \tag{3.47}$$

Un tel schéma n'est explicite que lorsque pour chaque i, les $a_{i,j}$ sont nuls si $j \geqslant i$: la matrice $A = (a_{i,j})_{1 \leqslant i,\, j \leqslant q}$ est alors strictement triangulaire inférieure.

Pour une présentation synthétique de la méthode, on utilise souvent le diagramme suivant (cas d'une méthode explicite) :

$$\begin{array}{c|ccccccc} 0 & & & & & & & \\ c_2 & a_{21} & & & & & & \\ c_3 & a_{31} & a_{32} & & & & & \\ \vdots & \vdots & \vdots & \ddots & & & & \\ c_i & a_{i1} & a_{i2} & \ldots & a_{ii-1} & & & \\ \vdots & \vdots & \vdots & \ldots & \vdots & \ddots & & \\ c_q & a_{q1} & a_{q2} & \ldots & a_{qi-1} & \ldots & a_{qq-1} & \\ \hline & b_1 & b_2 & \ldots & b_{i-1} & \ldots & b_{q-1} & b_q \end{array}$$

On retrouve par exemple les méthodes d'Euler modifiée et améliorée à travers leurs diagrammes respectifs :

$$\begin{array}{c|cc} 0 & & \\ 1/2 & 1/2 & \\ \hline & 0 & 1 \end{array} \qquad\qquad \begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

Ecriture matricielle On a déja introduit la matrice A. L'étude des schémas de Runge-Kutta est facilitée par une présentation matricielle. On introduit en outre le vecteur $b = (b_1, \ldots, b_q)^T$ et la matrice $C \in M_q(\mathbb{R})$, diagonale, dont les éléments diagonaux sont les c_i.

On introduit également le vecteur $e \in \mathbb{R}^q$ défini par $e = (1, \ldots, 1)^T$. Ces éléments permettent d'énoncer le théorème suivant :

Théorème 7 *La méthode de Runge-Kutta caractérisée par les éléments A, b, C et e définis ci-dessus est :*

- *consistante si et seulement si* $b^T\, e = 1$,
- *d'ordre 2 si en outre* $b^T\, C\, e = b^T\, A\, e = \dfrac{1}{2}$,
- *d'ordre 3 si en outre* $\left\{ \begin{array}{rcl} b^T\, C^2\, e & = & \dfrac{1}{3} \\ b^T\, A\, C\, e & = & \dfrac{1}{6} \end{array} \right.$
- *d'ordre 4 si en outre* $\left\{ \begin{array}{rcl} b^T\, C^3\, e & = & \dfrac{1}{4} \\ b^T\, A\, C^2\, e & = & \dfrac{1}{12} \\ b^T\, A^2\, C\, e & = & \dfrac{1}{24} \\ b^T\, C\, A\, C\, e & = & \dfrac{1}{8} \end{array} \right.$

3.3.7 La méthode "classique" de Runge-Kutta d'ordre 4

C'est la méthode habituellement désignée par le nom de méthode de Runge-Kutta. C'est une méthode excellente pour la plupart des problèmes de Cauchy, en tous cas souvent la première à essayer. Elle est à 4 étages, définie par la fonction $\Phi(t, y, h) = \dfrac{1}{6}\,(k_1 + 2k_2 + 3k_3 + k_4)$ avec :

$$\begin{array}{rcl} k_1 & = & f(t, y) \\ k_2 & = & f(t + \dfrac{h}{2},\, y + \dfrac{h}{2}\, k_1) \\ k_3 & = & f(t + \dfrac{h}{2},\, y + \dfrac{h}{2}\, k_2) \\ k_4 & = & f(t + h,\, y + h\, k_3)) \end{array}$$

Cette méthode combine les formules de Simpson, du trapèze, et les évaluations de $y(s)$ par la méthode d'Euler. On vérifiera en exercice que son écriture matricielle est la suivante :

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

On vérifiera également qu'elle est bien d'ordre 4.

3.3.8 Seuil de stabilité

Les majorations (3.38) ou (3.42) ne sont pas toujours pessimistes, comme le montre l'exemple (3.9). Cependant la figure 3.3 nous montre que selon la stabilité du problème, les erreurs locales peuvent être amplifiées, comme dans le cas de (3.9), ou atténuées. Ces majorations ne prennent pas en compte cette seconde possibilité.

Dans ce cas, les estimations du pas qu'on peut en déduire seront beaucoup trop restrictives : pour les problèmes stables, on peut penser qu'un pas assez grand fournira une solution acceptable. La question que l'on peut se poser est alors de savoir s'il n'existe pas une limite supérieure à la taille du pas h que l'on peut choisir.

L'exemple qui suit montre au moins que cette question est pertinente. Intéressons nous au problème suivant :

$$\begin{cases} y'(t) &= -9\ y(t) + 5\ t + 4, \quad 0 < t < 10, \\ y(0) &= \dfrac{1}{3} \end{cases} \tag{3.48}$$

Sa solution est

$$y(t) = \frac{45\,t + 31 - 4\,\exp(-9t)}{81}.$$

La solution d'un probléme perturbé, de même équation différentielle et de condition initiales $y_\epsilon(0) = \dfrac{1}{3} + \epsilon$ sera $y_\epsilon(t) = y(t) + \epsilon\ \exp(-9t)$. L'influence d'une perturbation de la donnée initiale est donc négligeable. On pouvait s'en convaincre en remarquant qu'il s'agissait d'un problème linéaire à coefficient constant avec $\partial_y f\,(t, y) = -9 < 0$.

Nous appliquons à ce problème la méthode d'Euler explicite, avec les pas $h = 0.20,\ 0.21,\ 0.22$ et 0.23. Nous représentons sur la figure ci-dessous les courbes des solutions calculée (trait plein) et exacte (trait en pointillés) :

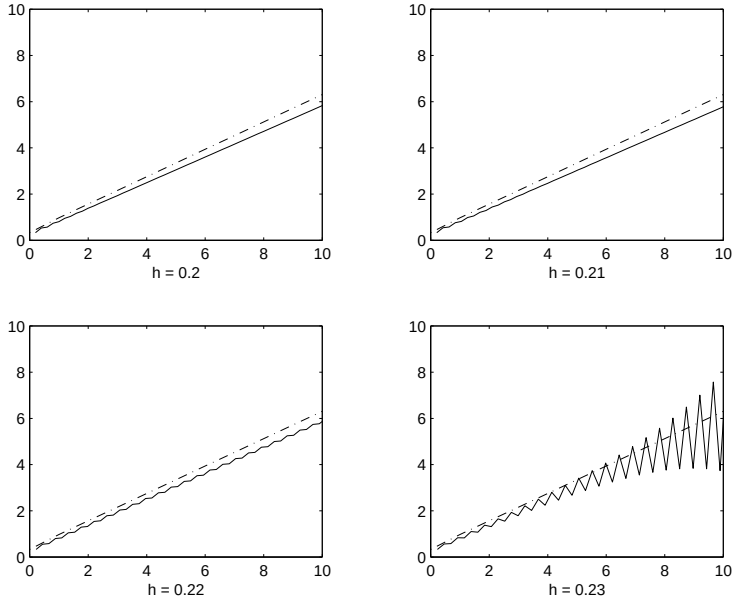


FIG. 3.6 – Apparition d'instabilités pour certaines valeurs du pas

On voit apparaitre, d'abord légèrement, puis franchement pour $h = 0.23$ une instablité des valeurs calculées par la méthode. On voit nettement se développer un phénomène d'instabilité autour de ces valeurs du pas.

Pour le comprendre, on étudie l'équation homogène associée à (3.48) :

$$y'(t) = -9\,y(t).$$

Sa solution est $y(t) = y(0)\,e^{-9t}$ et tend vers 0 lorsque t tend vers l'infini. On attend des valeurs y_n que l'on calcule qu'elles respectent au moins le comportement à l'infini de la solution $y(t)$. Le

schéma d'Euler calcule

$$y_1 = y_0 - 9\,h\,y_0 = (1 - 9\,h)\,y_0. \tag{3.49}$$

et au-delà, pour tout n,

$$y_n = (1 - 9\,h)^n\,y_0. \tag{3.50}$$

On constate que y_n ne tendra vers 0 lorsque n tend vers l'infini qu'à la condition que $|1 - 9\,h| < 1$, et comme le pas h est positif, il devra vérifier $9\,h < 2$, d'ou $h < \dfrac{2}{9} \simeq 0.2222$. Cette valeur constitue le seuil de stabilité pour le problème (3.48).

De façon générale, lorsque l'on applique une méthode à un pas à l'équation $y' = -a\,y$, la première étape fournit $y_1 = R(a\,h)y_0$, et pour $n \geqslant 1$, on aura

$$y_n = R(a\,h)^n y_0.$$

La fonction $R(x)$ va permettre de calculer le seuil de stabilité de la méthode pour un problème donné. On détermine d'abord r tel que $0 < x < r$ entraine $|R(x)| < 1$. Le seuil de stabilité est alors $h^* = \dfrac{r}{a}$.

Toute méthode à un pas d'ordre p sera telle que

$$R(x) = \sum_{k=0}^{p} (-1)^k\,\frac{x^k}{k!} + \mathcal{O}(x^{p+1}). \tag{3.51}$$

C'est une conséquence de la condition (3.25). En particulier, pour les méthodes de Runge-Kutta explicites d'ordre p égal au nombre q d'étages, la fonction $R(x)$ est donnée par

$$R(x) = \sum_{k=0}^{p} (-1)^k\,\frac{x^k}{k!} \tag{3.52}$$

Ainsi, on peut vérifier que pour la méthode **3.3.7**, $r \simeq 2.7853$. Pour la méthode d'Euler explicite, la fonction $R(x)$ est donnée par

$$R(x) = \frac{1}{1 + x}. \tag{3.53}$$

Cette fonction vérifie bien (3.51) puisque la méthode est d'ordre 1. Elle est telle que $|R(x)| < 1$ pour tout $x > 0$. Il n'y a donc pas de restriction à la stabilité de cette méthode : elle inconditionnellement stable.

Remarque 11 *G. Dahlquist a introduit la notion de A-stabilité pour caractériser les méthodes pour lesquelles il n'y a pas de restriction de stabilité pour l'équation $y' = -a\,y$, avec $a > 0$.*

En fait, aucune méthode explicite ne peut être A-stable, alors que les méthodes implicites peuvent l'être. C'est la raison pour laquelle elles suscitent beaucoup d'intérêt, et qui les rend indispensables pour les problèmes raides.

Remarque 12 *Pour des systèmes différentiels, le réel a est remplacé par une matrice A symétrique définie positive. La solution du problème $y' = -A\,y$ est donnée par $y(t) = \exp(-t\,A)\,y(0)$, où l'exponentielle est une exponentielle de matrice. Elle tend aussi vers 0 à l'infini.*

Le seuil de stabilité sera alors $s = \dfrac{r}{\lambda_M(A)}$, où $\lambda_M(A)$ désigne la plus grande valeur propre de A.

3.3.9 Méthodes emboitées et estimation de l'erreur

Les formules (3.42) et (3.38) donnent une idée du comportement de l'erreur pour un pas constant sur l'intervalle d'étude. Elle permettent d'avoir une idée à priori du pas qu'il convient de choisir, en fonction d'une tolérance donnée. Mais cette estimation risque d'être beaucoup trop pessimiste et pratiquement inefficace.

En général, les solutions peuvent varier lentement à certains endroits, et plus rapidement à d'autres. Un pas constant n'est donc pas toujours adapté. Il est souhaitable de pouvoir décider à chaque instant d'un pas qui permet de respecter une tolérance fixée à priori.

C'est le comportement local de la solution qui permettra de décider, et on va se servir des erreurs locales de discrétisation (3.19) :

$$\delta_n = z_n(t_{n+1}) - y_{n+1}.$$

Ces erreurs font intervenir une solution exacte du problème, qui est inaccessible. Supposons maintenant que l'on dispose de deux méthodes, une d'ordre p_1 et l'autre d'ordre p_2, avec $p_1 > p_2$. Partant de la valeur y_n calculée à l'étape n, elle fournissent suivant (3.19) et (3.25)

$$y_{n+1}^{(1)} = z_n(t_{n+1}) + \delta_n^{(1)} \simeq z_n(t_{n+1}) + C_1\, h^{p_1+1} \tag{3.54}$$

$$y_{n+1}^{(2)} = z_n(t_{n+1}) + \delta_n^{(2)} \simeq z_n(t_{n+1}) + C_2\, h^{p_2+1} \tag{3.55}$$

On déduit $\epsilon = |y_{n+1}^{(2)} - y_{n+1}^{(1)}| \simeq |C_2\, h^{p_2+1} - C_1\, h^{p_1+1}| \sim |C_2\, h^{p_2+1}|$ au voisinage de $h = 0$.

On se fixe une tolérance tol, et on souhaite que ϵ soit aussi voisin que possible de cette tolérance, sans la dépasser. On choisira par exemple de calculer un pas nouveau h_{nouv} de façon que $tol \simeq C_2\, h_{nouv}^{p_2+1}$. On a donc :

$$\frac{tol}{\epsilon} = \frac{C_2\, h_{nouv}^{p_2+1}}{C_2\, h^{p_2+1}},$$

de sorte que

$$\frac{h_{nouv}}{h} = \left(\frac{tol}{\epsilon}\right)^{\frac{1}{p_2+1}} \tag{3.56}$$

On choisit alors, avec une petite marge de sécurité, le nouveau pas :

$$h_{nouv} = 0.8\, h \left(\frac{tol}{\epsilon}\right)^{\frac{1}{p_2+1}} \tag{3.57}$$

Le seul problème est que l'on doit mettre en oeuvre 2 méthodes pour mener à bien ces calculs.

L'idée des méthodes de Runge-Kutta emboitées est d'utiliser les mêmes calculs pour les deux méthodes. Pour illustrer ce principe, observons les diagrammes la méthode d'Euler améliorée et de la méthode de Runge-Kutta d'ordre 4 :

0		
1/2	1/2	
	0	1

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

Ces deux diagrammes peuvent se rassembler en un seul :

0				
$1/2$	$1/2$			
$1/2$	0	$1/2$		
1	0	0	1	
$y^{(1)}$	$1/6$	$1/3$	$1/3$	$1/6$
$y(2)$	0	1	0	0

La mise en oeuvre de la méthode d'ordre $p_1 = 4$ conduit à calculer, à l'étape n, pour le pas courant h :

$$\begin{aligned} k_1 &= f(t_n, y), \\ k_2 &= f(t_n + \frac{h}{2},\, y_n + \frac{h}{2}\, k_1), \\ k_3 &= f(t_n + \frac{h}{2},\, y_n + \frac{h}{2}\, k_2), \\ k_4 &= f(t_n + h,\, y_n + h\, k_3)), \end{aligned}$$

avant de poser

$$y_{n+1}^{(1)} = y_n + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right).$$

Ces mêmes calculs peuvent être utilisés pour calculer

$$y_{n+1}^{(2)} = y_n + h\, k_2.$$

Cette fois, la méthode est d'ordre $p_2 = 2$, et suivant (3.57), le nouveau pas sera donné par :

$$h_{nouv} = 0.8\, h\, \left(\frac{tol}{\epsilon}\right)^{\frac{1}{3}}$$

Remarque 13 *En fait, on travaille plutôt avec des erreurs relatives,*

$$\epsilon = \frac{|y_{n+1}^{(2)} - y_{n+1}^{(1)}|}{\max\{|y_{n+1}^{(2)}|,\, |y_{n+1}^{(1)}|\}}.$$

A l'étape n, on commence par comparer ϵ à tol, et le calcul n'est conservé que lorsque $\epsilon < tol$. La valeur choisie pour la suite sera $y_{n+1} = y_{n+1}^{(1)}$. Ainsi, la méthode restera d'ordre p_1.

3.3.10 La paire de Bogacki et Shampine

Les fonctions `ode23` et `ode45` emboîtent respectivement des méthodes d'ordres (2, 3) et (4, 5). `ode45` est à 7 étages, connue dans la littérature sous le nom de méthode de Dormand-Prince d'ordres (4, 5).

`ode23` est caractérisée par le diagramme suivant :

0				
$1/2$	$1/2$			
$3/4$	0	$3/4$		
1	$2/9$	$1/3$	$4/9$	
$y^{(1)}$	$2/9$	$1/3$	$4/9$	0
$y(2)$	$7/24$	$6/24$	$8/24$	$3/24$

Elle est connue sous le nom de méthode de Bogacki et Shampine. On vérifiera en exercice qu'elle est bien d'ordres (2, 3). C'est la méthode 1 qui est d'ordre 3, de sorte que l'on ne calcule pas $y_{n+1}^{(2)}$.

Avec $y_{n+1}^{(1)} = y_n + \dfrac{h}{9}\,(2\,k_1 + 3\,k_2 + 4\,k_3)$, on doit calculer $k_4 = f(t_{n+1},\, y_{n+1}^{(1)})$ pour estimer

$$y_{n+1}^{(2)} = y_n + \frac{h}{24}\,(7\,k_1 + 6\,k_2 + 8\,k_3 + 3k_4)\,,$$

puis faire la différence $\epsilon = y_{n+1}^{(1)} - y_{n+1}^{(2)}$. On préfère calculer directement

$$\epsilon = \frac{h}{72}\,(-5\,k_1 + 6\,k_2 + 8\,k_3 - 9\,k_4)\,.$$

Remarque 14 *Les méthodes décrites pour le contrôle du pas permettent de calculer un pas nouveau à partir du pas courant. Il faut bien commencer par un premier pas. Les techniques pour le choisir sont plus ou moins sophistiquées. Celle qui suit est assez rustique, et correspond à peu près à ce qui est mis en oeuvre par* `ode23`*.*

La dérivée à l'instant initial est donnée par $y'(t_0) = f(t_0,\, y_0) = f_0$. En assimilant la courbe à sa tangente, le taux d'accroissement relatif au premier pas sera de l'ordre de $r = \left|\dfrac{y_0}{f_0}\right|$.

En supposant que le choix d'un pas égal à r fournirait une erreur relative de l'ordre de 1*, le "bon" pas initial nouveau sera, suivant (3.57) :*

$$h_0 = 0.8\,r\;(tol)^{\frac{1}{p_2+1}} \tag{3.58}$$

3.4 Méthodes implicites

Les méthodes implicites nécessitent à chaque pas la résolution d'une équation (ou d'un système) non linéaire. Cette équation est définie par :

$$y_{n+1} = y_n + h\,\Phi(t_{n+1},\; y_{n+1},\; h). \tag{3.59}$$

3.4.1 Stratégie de prédiction-correction (PECE)

Plaçons nous dans le cas de la méthode d'Euler implicite. Il faut résoudre l'équation

$$x = y_n + h\,f(t_{n+1},\; x), \tag{3.60}$$

où x désigne l'inconnue y_{n+1}.

Considérons la fonction g définie par $g(x) = y_n + h\,f(t_{n+1},\; x)$. Si f satisfait les hypothèses du théorème 6, on aura

$$|g(y) - g(z)| = |h\,f(t_{n+1},\; y) - h\,f(t_{n+1},\; z)| \leqslant h\,L\,|y - z|.$$

Choisissons alors $x^{(0)}$, si possible voisin de la solution, et calculons

$$x^{(1)} = y_n + h\,f(t_{n+1},\; x^{(0)}) = g(x^{(0)}) \tag{3.61}$$

Comme $g(x) = x$, on aura

$$|x^{(1)} - x| \leqslant h\,L\,|x^{(0)} - x|. \tag{3.62}$$

Il faudra choisir un pas h tel que $h < \dfrac{1}{L}$ pour que $x^{(1)}$ soit meilleur que $x^{(0)}$. On peut recommencer, et définir une suite $x^{(k)}$ qui converge vers x, mais il faudra s'assurer que le pas est suffisament petit pour avoir une suite convergente. Il faudra ensuite gérer la convergence de cette suite, avec une boucle et un test d'arrêt. On préfère travailler uniquement avec $x^{(0)}$ et $x^{(1)}$ en mettant en place une stratégie de contrôle du pas.

La première étape est un bon choix de $x^{(0)}$: on le calcule en utilisant la méthode d'Euler explicite, et on calcule $x^{(1)}$ suivant (3.61).

Cette stratégie est connue sous le nom de stratégie de prédiction-correction.

En supposant y_n calculé, et en notant $f_n = f(t_n, y_n)$, le calcul de y_{n+1} se fait selon :

Prédiction	$y_{n+1}^{(P)} = y_n + h\,f_n$
Evaluation	$f_{n+1}^{(P)} = f(t_{n+1},\, y_{n+1}^{(P)})$
Correction	$y_{n+1}^{(C)} = y_n + h\,f_{n+1}^{(P)}$
Evaluation	$f_{n+1} = f(t_{n+1},\, y_{n+1}^{(C)})$

Remarque 15 *Cette stratégie est également utilisée dans le cadre de méthodes multipas. Les plus utilisées sont les méthodes d' Adams :*

- *les méthodes d'Adams-Basforth sont explicites,*
- *les méthodes d'Adams-Moulton sont implictes.*

On peut donc les associer dans une stratégie PECE avec un contrôle du pas. La fonction `ode113` *de* MATLAB *implante cette méthode, pour des ordres pouvant varier de* 1 *à* 13.

3.4.2 Contrôle du pas des méthodes PECE

Pour décider de garder $x^{(1)}$, il faut s'assurer que la majoration (3.62) est acceptable. On le fait de façon empirique en évaluant l'écart

$$\Delta = |y_{n+1}^{(P)} - y_{n+1}^{(C)}|$$

Pour tirer profit de cette évaluation, il faut un peu plus d'informations sur l'erreur locale de discrétisation δ_n (3.19). Pour les méthodes d'Euler, on a

- dans le cas explicite, $z_n(t_{n+1}) = y_n + h\,f(t_n,\, y_n)) + \dfrac{h^2}{2} z_n''(t_{n,1})$,
- dans le cas implicite, $z_n(t_{n+1}) = y_n + h\,f(t_n,\, z_n(t_{n+1})) - \dfrac{h^2}{2} z_n''(t_{n,2})$.

$t_{n,1}$ et $t_{n,2}$ sont tous deux dans l'intervalle $[t_n,\ t_{n+1}]$. On peut donc espérer que

$$\begin{aligned} z_n(t_{n+1})) &\simeq y_{n+1}^{(P)} + \frac{h^2}{2} y''(t_{n,1}), \\ z_n(t_{n+1})) &\simeq y_{n+1}^{(C)} - \frac{h^2}{2} y''(t_{n,2}). \end{aligned}$$

Par soustraction, on obtient $\Delta \simeq Kh^2$, de sorte que

$$|y_{n+1}^{(C)} - z_n(t_{n+1})| \simeq \frac{\Delta}{2}.$$

Si la tolérance qu'on s'est imposée est tol et si $\dfrac{\Delta}{2} < tol$, on continuera avec la valeur $y_{n+1} = \dfrac{1}{2}\,(y_{n+1}^{(P)} + y_{n+1}^{(C)})$.

Dans tous les cas, on choisira le nouveau pas h_{nouv} selon

$$h_{nouv} = 0.8\,h\sqrt{\frac{2\,tol}{\Delta}}, \tag{3.63}$$

le facteur 0.8 étant simplement une marge de sécurité.

Remarque 16 *On dispose ici à petit prix d'un bon moyen de choisir le premier pas. Partant d'abord de y_0, on peut calculer $f_0 = f(t_0,\, y_0)$ qui fournit une première estimation de $y'(t_0)$ permettent de choisir par exemple $\bar{h}_1 = 0.01 \left|\dfrac{y_0}{f_0}\right|$ de façon à limiter l'accroissement au premier pas à 1%. Cette limitation risquant de s'avérer trop stricte, on utilise ce pas pour calculer $\bar{y}_1$, puis $\bar{f}_1 = f(t + \bar{h}_1,\, \bar{y}_1)$ pour estimer $|y''(t_0)|$ par $d_2 = \left|\dfrac{f_0 - \bar{f}_1}{\bar{h}_1}\right|$.*

En considérant alors que l'erreur au premier pas est bien estimée par $\dfrac{h^2}{2}\,y''(t_0)$, on évalue $\tilde{h}_1 = \sqrt{\dfrac{2 * tol}{d_2}}$, et on choisit finalemement :

$$h_1 = \min\left\{100\,\bar{h}_1,\ \tilde{h}_1\right\}.$$

Remarque 17 *On peut aussi chercher à résoudre l'équation (3.59) en utilisant la méthode de Newton, ou une de ses variantes. C'est ce que font les méthodes connues sous le nom de méthodes de Rosenbrock qui sont construites à partir de méthodes de type Runge-Kutta semi-implicites. La fonction* `ode23s` *de* MATLAB *met en oeuvre deux méthodes de Rosenbrock d'ordre 2 et 3, emboitées sur 3 étages.*

3.5 Problèmes raides

3.5.1 La flamme de l'allumette

La notion de raideur d'un problème est un concept assez difficile, mais important du domaine de la résolution approchée des équations différentielles. Si l'idée intuitive de ce qu'est un problème raide est assez largement partagée, sa définition rigoureuse reste controversée.

De façon pragmatique, les problèmes raides sont des problèmes pour lesquels la solution, ou une des composantes de la solution, risque d'avoir une variation très brutale. Les problèmes raides sont des problèmes pour lesquels les méthodes explicites sont inefficaces.

L'exemple qui suit illustre cette idée. Il est dû à L. Shampine. Il s'agit d'un modèle simplifié de propagation de flamme : lorsque vous grattez une allumette, la boule de flamme va grossir très rapidement avant d'atteindre une taille critique, pour laquelle la quantité d'oxygène consommée à l'intérieur de cette boule est égale à celle qui est disponible en surface. La surface d'une sphère de

rayon y est $4\,\pi\,y^2$, et son volume est $\frac{4}{3}\,\pi\,y^2$. Cette loi d'équilibre est grossièrement décrite par le problème :

$$\begin{cases} y' = y^2 - y^3\,;\; 0 \leqslant t \leqslant \dfrac{2}{\rho} \\ y(0) = \rho. \end{cases} \tag{3.64}$$

Ce problème semble très anodin, et il l'est en effet lorsque le rayon initial ρ n'est pas trop petit. Mais si ρ est petit, il devient assez raide. La figure 3.7 nous montre les courbes calculées par MATLAB pour 2 valeurs du rayon.

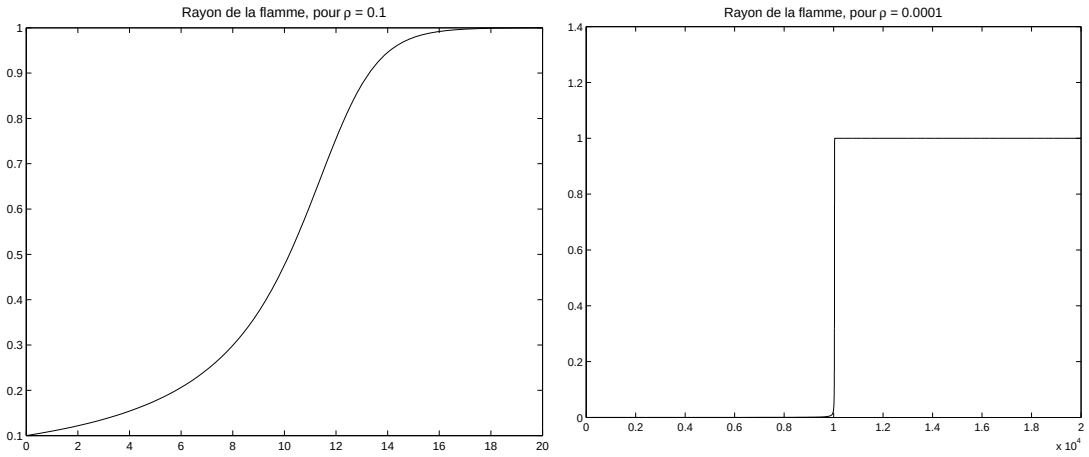


FIG. 3.7 – Le problème se raidit quand ρ devient petit

3.5.2 Les solveurs de MATLAB

La figure 3.7 semble indiquer que MATLAB a bien réussit à approcher la solution du problème raide. L'appel d'un solveur MATLAB pour obtenir la courbe de droite peut s'écrire :

```
>> fonction = inline('y.^2-y.^3','t','y');
>> delta = 0.0001;
>> intervalle =  [0, 2/delta];
>> [t, y] = ode-solveur(fonction, intervalle, delta);
>> plot(t, y);
```

Dans ces lignes, `ode-solveur` sera remplacé successivement par `ode113`, `ode23`, `ode45` et `ode23s`.

En fait, il convient de regarder les choses d'un peu plus près. Tous ces solveurs ne se comportent pas de la même façon, et la figure 3.8 nous montre les courbes obtenues avec une échelle définie par la commande

```
>> axis([0.98/delta 1.12/delta 1-30*delta 1+30*delta]);
```

On constate alors, que parmi les solveurs testés, seul `ode23s` semble se comporter correctement.

Les courbes associées à `ode23` et `ode45` présentent une zone très chaotique, mais les valeurs calculées respectent la tolérance relative par défaut qui est de 10^{-3}. Ces solveurs ont donc bien fonctionné. Ils ne sont simplement pas adaptés au problème.

`ode113` ne réussit pas à respecter cette tolérance. C'est normal car les méthodes multipas linéaires ne sont pas adaptées aux problèmes raides. Elles conviennent à des problèmes assez

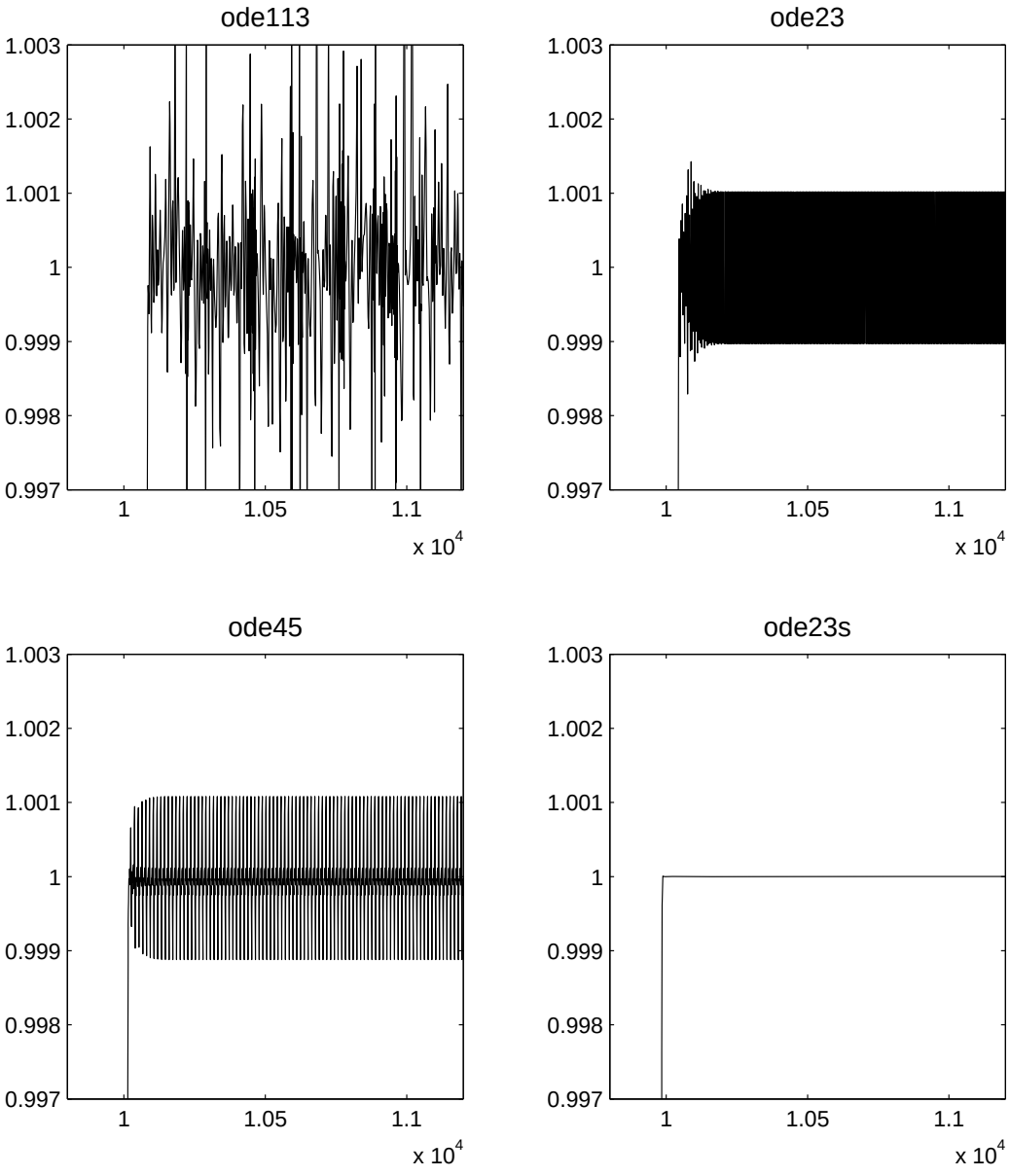


FIG. 3.8 – Le problème se raidit quand ρ devient petit

réguliers pour lesquels le coût d'évaluation de f est important. Elles sont aussi mieux adaptées que les méthodes à un pas pour les calculs de trajectoires à long terme.

Nous pouvons obtenir des informations sur le déroulement des calculs en déclarant :

```
>> options = odeset('Stats','on');
>> [t, y] = ode-solveur(fonction, intervalle, delta, options);
```

Voici deux des réponses obtenues :

- pour `ode45` :

```
3028 successful steps
762 failed attempts
22741 function evaluations
0 partial derivatives
0 LU decompositions
0 solutions of linear systems
```

- pour `ode23s` :

```
55 successful steps
14 failed attempts
250 function evaluations
55 partial derivatives
69 LU decompositions
207 solutions of linear systems
```

La statistique relative à `ode23s` mentionne des évaluations de dérivées partielles, qui sont faites par différences finies. Par contre, s'agissant ici d'une équation et non d'un système, les résolutions de systèmes linéaires sont de simples divisions. On constate qu'il faut environ 7 évaluations de fonctions par étape pour `ode45` qui est une méthode à 7 étages.

Le tableau suivant nous donne rappelle le nombre d'évaluations de la fonction f, nous donne la taille du vecteur `t` et le temps CPU (en secondes) utilisé par l'ordinateur sur lequel nous avons effectué le calcul.

Solveur	`ode113`	`ode23`	`ode45`	`ode23s`
Evaluations	19906	12070	22741	250
CPU	8.7667	1.5833	2.8167	0.0667
Taille	9248	4012	12113	56