ENSIM
École d'ingénieurs
Le Mans Université

Le Mans Université

$3^{\text{éme}}$ **Année**

Spécialité Acoustique et Instrumentation

Mathématiques de l'ingénieur 2

Responsable : Y.SERRESTOU

ENSEIGNANTS :
S.HASSAD, M.BENDAROUICHE, D.CACITTI-HOLLAND,Y.ESSTAFA, Y.SERRESTOU

ENSIM - Université du Maine - Rue Aristote - 72085 LE MANS Cedex 9 - France
Tél : 02 43 83 35 93 - Télécopie : 02 43 83 37 94

1 TP1 - Pivot de Gauss

1.1 Méthode de la remontée

1. Écrire une fonction *remontee* qui prend en entrée une matrice triangulaire supérieure A inversible de taille $n \times n$ et un vecteur colonne b de taille n et qui retourne le vecteur colonne x de taille n tel que $Ax = b$.
2. Vérifier le programme avec des exemples simples.
3. Pour évaluer la complexité de cette méthode, compléter votre script en générant des matrices triangulaires supérieures A_n aléatoires de taille $n \times n$ et des vecteurs colonnes b_n aléatoires de taille n pour $n \in \{10, ..., 10N\}, N \in \mathbb{N}^*$. Évaluer les temps de calcul nécessaires à la fonction *remontee* pour résoudre les systèmes $A_n x = b_n$.

1.2 Pivot de Gauss

1. Écrire une fonction *choixpivot* qui prend en entrée une matrice A de taille $n \times n$, un vecteur colonne b de taille n et un entier $k \in 1, n$ et qui échange éventuellement la première ligne de la matrice A avec une autre pour avoir un pivot non nul en position (k, k) et échange également les lignes correspondantes du vecteur colonne b.
2. Écrire une fonction *choixpivotmax* comme à la question 1 mais qui utilise le choix du pivot partiel ou total.
3. Écrire une fonction *Gauss* qui prend en entrée une matrice A de taille $n \times n$ et un vecteur colonne b de taille n et qui renvoie, grâce à la méthode du pivot de Gauss, le vecteur colonne x de taille b tel que $Ax = b$. On pourra se servir des fonctions *choixpivot* et *remontee*.
4. Évaluer la complexité de la fonction précédente comme à la question 3 de l'exercice 1.

1.3 Inversion de matrice

1. Ecrire une fonction *GaussJordon* qui prend en entrée une matrice A inversible de taille n et qui renvoie son inverse A^{-1} grâce à la méthode de Gauss-Jordan.
2. Comparer les temps de calcul avec ceux de la fonction *inv*() de Matlab.

2 TP2 - Factorisation LU et Cholesky

2.1 Factorisation LU

1. Programmer la factorisation LU d'une matrice A. Programmer aussi une fonction `descente` qui permet de résoudre un système linéaire pour une matrice triangulaire inférieure.
2. Utiliser les algorithmes précédents, ainsi que la fonction `remontee`, pour résoudre un système linéaire $Ax = b$.
3. Évaluer la complexité de cette méthode.

2.2 Factorisation de Cholesky

Le but de cette partie est d'inverser une matrice par la méthode de Cholesky.

1. Écrire une procédure `Cholesky` qui prend en paramètre une matrice symétrique définie positive A et renvoit une matrice triangulaire inférieure B telle que

$$A = B{}^tB.$$

2. Écrire une procédure `ResolutionCholesky` qui prend en paramètres une matrice symétrique définie positive A et une matrice Y, et qui renvoit la solution du système $AX = Y$ en la calculant par la méthode de Cholesky.
3. Comparer numériquement les temps de calculs nécessaires pour les méthodes de Gauss et de Cholesky ; on prendra bien garde à n'utiliser que des matrices symériques définies positives !
4. Sachant que
   (a) la méthode de Gauss a un temps de calcul de l'ordre de $2n^3/3$ opérations élémentaires ;
   (b) l'algorithme de Cholesky demande un temps de calcul de l'ordre de $2n^3/6$ opérations élémentaires.

   Estimer le rapport des temps de calcul pour les deux méthodes. Commenter vos résultats.
5. Estimer le temps de calcul d'une opération élémentaire avec la méthode qui vous semble la plus adaptée, justifier votre choix.

3 TP3 - Résolutions numériques d'équations différentielles

Introduction

Les équations différentielles ordinaires (EDO) modélisent les phénomènes d'évolution en temps continu et interviennent donc dans une très grande variété de problèmes physiques : mécanique des solides, mécaniques des fluides, dynamique moléculaire, réactions chimiques, économie, dynamiques des populations, etc ...

Les problèmes que modélisent les edos sont du type :

$$\begin{cases} y'(t) &= f(t, y(t)) \quad \forall t \in [a, b] \\ y(a) &= y_a \end{cases} \tag{1}$$

Note : à la place de la notation $y'(t)$ on trouve souvent la notation $\frac{dy}{dt}$.
Si l'application f est continue et telle que $f(t, .)$ est lipschitzienne pour tout t, alors il existe une seule solution à ce problème différentiel. Cette solution est de plus continûment différentiable et le problème théorique est bien posé (une faible perturbation dans les conditions initiales ou dans l'évaluation de f entraîne une faible perturbation de la solution).

4 Méthodes de résolution

4.1 Principe général

Les méthodes d'intégration numérique qui permettent de calculer la valeur approchée de $y(t)$ reposent sur une discrétisation de l'intervalle $[a, t]$ (cas d'un pas de discrétisation constant) :

$$\begin{aligned} &t_i = a + ih \\ &i \in 0, 1, ...N \\ &h = (t - a)/N \end{aligned}$$

On obtient alors par intégration exacte de (1) l'itération :

$$y(t_n) = y(t_{n-1}) + \int_{t_{n-1}}^{t_n} f(t, y(t))dt \tag{2}$$

De cette itération sur les valeurs exactes de $y(t_n)$ on dérive l'expression des valeurs approchées :

$$y_n = y_{n-1} + h\Phi_n \tag{3}$$

où y_n est la valeur approchée de $y(t_n)$ et Φ_n, appelé accroissement relatif approché, est une approximation de l'intégrale apparaissant dans (2). On définit alors l'erreur de discrétisation par :

$$e_n = y_n - y(t_n)$$

La définition d'une méthode repose donc essentiellement sur le choix de Φ_n, c'est-à-dire d'une valeur approchée pour l'intégrale de (2).

4.2 Quelques méthodes de résolution

Les méthodes les plus répondues sont celles d'Euler, d'Euler modifiée , de Runge Kutta et d'Adams Basforth explicite à l'ordre 4. Le principe de ces méthodes repose sur la définition de la suite y_n. Ci-dessous les suites construites par ces différentes méthodes :

— Méthode de Euler

$$y_{n+1} = y_n + hf(t_n, y_n)$$

— Méthode de Euler modifiée (RK2)

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

— Méthode de Runge Kutta à l'ordre 4

$$y_{n+1} = y_n + h\frac{f_1 + 2f_2 + 2f_3 + f_4}{6}$$

$$\text{où } \begin{cases} f_1 = f(t_n, y_n) \\ f_2 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2}f_1) \\ f_3 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2}f_2) \\ f_4 = f(t_n + h, y_n + hf_3) \end{cases}$$

— La méthode d'Adams Basforth

$$y_{n+1} = y_n + \frac{h}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}) \text{ où } f_k = f(t_k, y_k)$$

h est le pas de la méthode.

4.3 Travail à réaliser

Exercice **1. Questions préliminaires**

1. Ecrire des fonctions qui permettent la construction de courbes intégrales sur $[t_0; t_{max}]$ vérifiant $y(t_0) = y_0$ avec n subdivisions de l'intervalle $[t_0; t_{max}]$ à l'aide des méthodes de Euler, de Euler modifié , de Runge Kutta à l'ordre 4, et d'Adams Basforth explicite à l'ordre 4.

$$[t, y] = euler(f, t_0, y_0, t_{max}, n]$$
$$[t, y] = RK2(f, t_0, y_0, t_{max}, n]$$
$$[t, y] = RK4(f, t_0, y_0, t_{max}, n]$$
$$[t, y] = adams(f, t_0, y_0, t_{max}, n]$$

   où t est le vecteur $[t_0, t_1, \dots t_{max}]$ et y est le vecteur $[y(t_0), y(t_1), \cdots, y(t_{max})]$

2. Calculer la solution exacte de

$$\begin{cases} y'(t) = -y(t) + t + 1 \\ y(0) = 1 \end{cases}$$

3. Sur $[0, 1]$, comparer avec les solutions trouvées numériquement à l'aide des méthodes ci-dessus pour $n = 10, 30$ et 50.

Exercice **2. La méthode d'Euler**

1. Résoudre à l'aide de la méthode d'Euler l'équation :

$$\begin{cases} y'(t) &= \lambda y(t) \quad \forall t \in [a,b] \\ y(a) &= y_a \\ \lambda &< 0 \end{cases} \tag{4}$$

2. Montrez sur des exemples que certaines valeurs de h conduisent à une instabilité numérique.
3. Expliquer pourquoi la condition de stabilité numérique dans ce cas est donnée par :

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

Exercice **3. La méthode d'Euler implicite**

La méthode d'Euler *implicite* repose aussi sur l'approximation de l'intégrale de l'équation (2) par la méthode dite *des rectangles*. La différence est qu'on n'approche pas la valeur de f sur l'intervalle par $f(t)$ mais par $f(t+h)$. Cela nous donne la même précision : $\int_t^{t+h} f(x)dx = hf(t+h) + O(h^2)$. L'itération (3) devient alors :

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}) \tag{5}$$

1. Comment calcule-t-on y_{n+1} par cette méthode ? Quelle difficulté cette méthode pose-t-elle (quelle opération doit-on savoir faire sur la fonction $y \to (Id - hf)(y)$) ?
2. Programmez la méthode d'Euler implicite pour résoudre l'équation 4.
3. Montrez que la méthode ainsi définie est numériquement stable quel que soit le pas h (on parle de méthode A-stable ou encore inconditionnellement stable).

Exercice **4. Proie Prédateur**

Dans cette partie, on va intégré le modèle mathématique du problème connu du prédateur et de la proie. On considère une population de sardines notée $x(t)$ et une population de requins notée $y(t)$ le modèle régissant l'évolution de ces deux populations est donné comme suit :

$$\begin{cases} x'(t) = ax(t) - bx(t)y(t) + F(t)x(t) \\ y'(t) = cx(t)y(t) - dy(t) \end{cases}$$

Avec $F(t)$ une fonction signifiant une action externe agissant sur la population de sardine et qu'on peut choisir arbitrairement.

1. Justifier ce modèle en donnant le sens de chacun des termes.
2. Intégrer les équations précédentes par un schéma de type Euler et par un autre schéma de type trapèze, avec $F(t) = 0$; $a = 2$; $b = 1$; $c = 0,3$; $d = 1$; $x(0) = 2$; $y(0) = 1$; $h = 0,001$