PROJET DU PÔLE

Algèbre – Algorithmique
Programmation – Analyse Numérique

THÈME DU PROJET

Résolution d'un système d'équations linéaires (SEL)

1 Contexte et problématique

On s'intéresse à la résolution d'un SEL $\mathbf{Ax} = \mathbf{b}$ sous un triple aspect :

- ***Analyse Numérique.-*** Il faut étudier l'influence sur la qualité de la solution :
  - de la valeur du conditionnement de la matrice (calculé à l'aide de la norme matricielle euclidienne) ;
  - de la dimension de la matrice, et
  - de la méthode de la résolution utilisée (méthode de la bibliothèque Scilab et Jacobi).

- ***Algorithmique et Programmation.-*** Il faut établir les algorithmes de résolution d'un SEL selon la méthode de Gauss-Jordan et la méthode itérative de Jacobi et ensuite écrire les programmes correspondants.
  En particulier, on étudiera l'influence sur la qualité de la solution
  - du codage utilisé pour les nombres (nombre de bits pour l'exposant et la mantisse), et
  - de la méthode de la résolution utilisée (Gauss et Jacobi).

- ***Algèbre linéaire.-*** Il faut retrouver les résultats essentiels concernant la résolution d'un SEL, les normes matricielles et les valeurs propres.

Dans la suite vous trouverez, pour chacune de trois parties ci-dessus, les informations nécessaires pour la réalisation du projet ainsi que les livrables que vous devez fournir avec leurs dates de livraison (à respecter scrupuleusement).

2 Partie algèbre linéaire

L'objectif de cette partie est de compléter l'approche numérique de la résolution d'un SEL par une étude mathématique qui permettra d'une part de mieux appréhender les problèmes posés et, d'autre part, de fournir quelques raccourcis intéressants pour certains calculs.

2.1 Étude théorique A -Valeurs Singulières ou "Diagonalisation" des matrices "Rectangulaires"

Le but de cette étude est de bien comprendre, approfondir, et illustrer le rôle des transformations unitaires par les deux étapes suivantes :

- a) Donner une démonstration du théorème de diagonalisation d'une matrice rectangulaire (voir rappel) et ensuite
- b) Appliquer dans le cas de deux matrices que vous choisirez $A \in \mathcal{M}_{\mathbb{C}}(3,2)$ et $B \in \mathcal{M}_{\mathbb{R}}(2,3)$.

Définition 1 ***RAPPEL***

Pour une matrice carrée $A \in \mathcal{M}_n(\mathbb{K})$, on appelle ***valeurs singulières*** *de A, les racines carrées positives des valeurs propres de l'opérateur autoadjoint A^*A*
(resp.matrice carrée hermitienne).

On fait une extension de cette notion pour une matrice rectangulaire $A \in \mathcal{M}_{\mathbb{C}}(m,n)$ par le théorème suivant, qui fournit simultanément la diagonalisation de la matrice rectangulaire A.

Théorème 2 ***RAPPEL***

Soit $A \in \mathcal{M}_{\mathbb{C}}(m,n)$

$\Rightarrow \exists$ une matrice unitaire $U \in \mathcal{M}_m(\mathbb{C})$ et une autre matrice unitaire $V \in \mathcal{M}_n(\mathbb{C})$ telles que :

$$U^*AV = \begin{pmatrix} \mu_1 & & 0 & 0 & \vdots & 0 \\ 0 & \mu_2 & 0 & 0 & \vdots & 0 \\ 0 & 0 & \ddots & 0 & \vdots & 0 \\ \cdots & \cdots & \cdots & \mu_r & \vdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{avec } \mu_i > 0 \ \ \forall\ 1 \leq i \leq r \quad \textit{et } r = rg\ A$$

Les nombres réels positifs μ_i s'appellent les ***valeurs singulières*** *de la matrice A, et sont les racines carrées positives des valeurs propres de la matrice carrée autoadjointe A^*A.*

2.2 Étude théorique B - Norme matricielle Rayon spectral et applications

- a) Par application du résultat précédent montrer le théorème suivant qui donne une relation importante entre la norme matricielle et le rayon spectral d'un opérateur.
- b) Appliquer ce théorème (dans des cas précis des exemples que vous trouverez à 3 ou plus dimensions) d'une matrice **Unitaire** d'une matrice **Hermitienne**, d'une matrice **Orthogonale** ou d'une matrice **Réelle Symétrique**.

Théorème 3

Si $A \in \mathcal{M}_n(\mathbb{K})$, alors :

i) $\|A\|_2 \overset{déf}{=} \sup_{u\neq 0} \dfrac{\|Au\|_2}{\|u\|_2} = \sqrt{\rho(A^*A)} = \sqrt{\rho(AA^*)}$

ii) Si $\|\cdot\|$ *est une norme matricielle (subordonnée ou pas), alors :*

$$\rho(A) \leq \|A\|$$

iii) La norme $\|A\|_2$ *est invariante par transformation unitaire.*

iv) Si la matrice A *est normale alors* $\|A\|_2 = \rho(A)$.

2.3 Étude Théorique C : Th. de Gerschgorin - Hadamard et Applications

Quand on étudie la stabilité de certains systèmes et souvent en Analyse Numérique (en particulier quand la diagonalisation de certaines matrices est impossible ou très difficile) on se contente d'explorer certains voisinages D_ℓ (dans $\mathbb{C}$) des valeurs propres λ_ℓ ; ainsi au lieu de les déterminer complétement on essaie de les localiser d'une certaine manière. Le résultat suivant fournit précisément la "localisation" des valeurs propres de A. On utilise la notation $Sp(A)$ pour l'ensemble des valeurs propres (spectre) de A.

Théorème 4 *(Gerschgorin - Hadamard)*

Soit $A = \{a_{ij} \in \mathcal{M}_n(\mathbb{C})$. *On définit les "****disques de Hadamard****" sur le plan* $\mathbb{C}$ *par :*

$$D_\ell = \left\{ z \in \mathbb{C} \ ; \ |z - a_{\ell\ell}| \leq \sum_{j\neq\ell} |a_{\ell j}| \right\}$$

$$\Rightarrow Sp(A) \subset \bigcup_{\ell=1}^{n} D_\ell$$

Exemple 5 Soit la matrice :

$$A = \begin{pmatrix} 7 & -2 & -1 & 0 \\ -2 & 7 & -1 & -1 \\ -1 & -1 & 7 & 2 \\ 0 & -1 & 2 & 7 \end{pmatrix}$$

Quel est le centre et le rayon du "cercle" (intervalle) où se trouve d'après le théorème 4 toutes les valeurs propres (réelles car A symétrique réelle) ?

Par application du th. de Gerschgorin résoudre les excercices (applications) suivants :

(1) Pour les exemples que vous avez construits dans la partie B (matrices Orthogonale-Hermitienne Réelle Symetrique ou Unitaire) est-ce que vos estimations concernant les domaines où se trouvent les valeurs propres d'après "Gerschgorin" sont en accord avec le rayon spectral correspondant (norme matricielle) des matrices associées ?

2) Quelle est l'information qu'on puisse obtenir d'après "Gerschgorin" pour les valeurs propres d'une matrice de "permutation" qui sur chaque ligne et chaque colonne a des zéros partout sauf un seul élément qui est égal à 1 ?

Exemple 6 *Pour $n = 4$:*

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

Montrer le résultat en généralisant pour n lignes et n colonnes.

3) Quel sera le disque de Hadamard pour une matrice symétrique à n^2 dimensions avec une diagonale "dominante", du type :

$$A = \begin{pmatrix} \alpha & \beta & \dots & \beta & \dots & \beta \\ \beta & \alpha & \beta & \beta & \dots & \beta \\ \beta & \beta & \ddots & \beta & \dots & \beta \\ \dots & \dots & \dots & \alpha & \dots & \beta \\ \beta & \beta & \beta & \beta \cdots & \alpha & \beta \\ \beta & \beta & \beta & \beta & \cdots & \alpha \end{pmatrix}$$

Que se passe-t-il quand $\beta = 0$?

4)

Pour les matrices suivantes trouver les valeurs propres et comparer vos résultats avec ceux que vous obtenez quand vous appliquez le théorème de Gerschgorin - Hadamard.

$$B_1 = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 1 & 0 \\ 5 & 1 & 3 \end{pmatrix} \qquad B_2 = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & -5 \\ 0 & 1 & -2 \end{pmatrix}$$

2.4 Applications de l'algébre linéaire à l'analyse numérique

Les questions suivantes ont pour but de vous permettre, lors d'une résolution d'un SEL, par des calculs préalables d'avoir une idée de la qualité numérique de la solution cherchée.

1. Pour tout réel κ et toute matrice diagonale

$$\mathbf{D} = \operatorname{diag}\left(1, \kappa^{-\frac{1}{n-1}}, \kappa^{-\frac{1}{n-2}}, \dots, \kappa^{-1}\right)$$

de taille n, on a $\kappa_2(\mathbf{D}) = \kappa$, où le conditionnement $\kappa_2(\mathbf{D})$ est calculé à partir de la norme 2.

2. Le calcul de la norme 2 d'une matrice étant difficile, on voudrait substituer au conditionnement κ_2 celui calculé avec une autre norme matricielle.

   (a) Énoncer et démontrer les majorations permettant d'établir les équivalences de normes, pour les normes $1, 2$ et ∞ ;

   (b) En déduire des encadrements de $\kappa_2(\mathbf{A})$ par $\kappa_1(\mathbf{A})$ et $\kappa_\infty(\mathbf{A})$;

   (c) La matrice $\mathbf{D}$ peut-elle être utilisée pour étudier la sensibilité des méthodes de résolution au conditionnement, en considérant κ_1 et κ_∞ ? ;

3. Dans quelle mesure les résultats d'une étude de sensibilité au conditionnement effectuée en considérant κ_2 peuvent-ils être indicatifs de ceux avec κ_1 et κ_∞? ;
4. Le critère de l'erreur résiduelle (cf. infra, "Annexe B – e") n'est pas toujours un critère infaillible. Voici un exemple simple. Soit le système

$$\begin{bmatrix} 0.835 & 0.667 \\ 0.333 & 0.266 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.168 \\ 0.067 \end{bmatrix}$$

dont la solution exacte est $[1, -1]^\top$. Mais la solution approchée $[0.333, -0.165]^\top$ fournit une erreur résiduelle égale à 0.0000055.

Existe-t-il un indicateur, à part le conditionnement qui est toujours difficile à calculer, qui permet d'être circonspect quant à la valeur de l'erreur résiduelle ?

5. En utilisant les disques de Gershgorin, élaborer :
   (a) une démarche qui permet d'avoir une indication si la matrice A pourrait éventuellement être singulière, et
   (b) une démarche qui permet d'évaluer le conditionnement très approximativement et de manière qualitative (c'est-à-dire conditionnement petit, moyen, grand ou très grand).

2.5 Livrable

Un rapport dans lequel vous présenterez vos réponses aux questions posées.
Date de livraison : 12.12.2008

3 Partie algorithmique – programmation

Les objectifs de cette partie sont :
- l'élaboration des algorithmes avec utilisation des tableaux à une et deux dimensions ;
- l'écriture des programmes en ADA, issus des algorithmes ci-dessus, et
- l'utilisation des plusieurs codages pour les nombres flottants.

Pour réaliser ces objectifs, on se propose d'effectuer la résolution d'un SEL selon deux méthodes :

1. la méthode du pivot de Gauss, qui est une méthode directe, et
2. la méthode itérative de Jacobi.

Pour chacune de ces deux méthodes

1. Décomposer en sous problèmes.
2. Écrire l'algorithme de chacun des différents sous problèmes.
3. Écrire l'algorithme principal qui appelle les algorithmes des sous problèmes identifiés.

Nous donnons dans la suite une présentation détaillée pour chacun de ces objectifs, ainsi que du livrable.

3.1 Algorithmes pour la résolution d'un SEL

L'objectif de cette partie est d'élaborer les algorithmes qui permettent de faire la résolution d'un SEL. Ces algorithmes seront utilisés par la suite pour écrire les programmes correspondants en ADA. Il s'agit de faire les algorithmes pour deux méthodes de résolution, à savoir

1. méthode de Gauss-Jordan, dont nous donnons un résumé en annexe C, et
2. méthode itérative de Jacobi, dont le formulaire se trouve en annexe B.

3.2 Programmes pour la résolution d'un SEL

Il s'agit d'écrire en ADA le code correspondant aux deux algorithmes de résolution.

Les programmes seront testés en utilisant les mêmes matrices que celles utilisées dans la partie Analyse Numérique (cf. infra "Partie analyse numérique"). L'échange des données entre Ada et Scilab peut se faire de deux façons, à savoir :

- Soit par l'intermédiaire des fichiers. En effet, Scilab permet l'export de matrice dans un fichier texte. Si on utilise cette fonctionnalité, un export doit être fait depuis Scilab pour stocker la matrice **A** dans un fichier texte qui, par la suite, sera lu par le programme ADA correspondant.
- Soit par des appels croisés. En effet nous pouvons, à partir d'un programme Scilab, appeler un programme écrit en ADA et compilé, et récuperer ses résultats [1]. Le programme en Ada peut aussi lire des fichiers écrits par Scilab. Cette technique a été étudiée lors du 3e TP de Scilab et elle est présentée brièvement en annexe D.

Pour chaque matrice utilisée, il faut construire un fichier qui a la structure suivante :

- 1ère ligne : nombre de lignes, nombre de colonnes de la matrice ;
- 2ème ligne et suivantes : 1ère ligne et lignes suivantes de la matrice.

Ce format vous permettra de :

1. Ouvrir un fichier, lire la taille de la matrice ($L \times C$), faire de l'allocation dynamique de tableau, et prévoir un traitement d'exception si les données du fichier ne correspondent pas à la taille signalée au début du fichier.
2. Faire un traitement d'exception au niveau de la procédure d'ouverture et de lecture du fichier, puis propager l'exception pour faire un traitement généralisé au niveau du programme principal.
3. Organiser vos fichiers source dans de paquetages, à savoir
   (a) paquetage de lecture des données
   (b) paquetage de la résolution d'un SEL par Gauss-Jordan.

D'autre part il faut, pour l'interaction homme-machine, élaborer des menus. Dans le menu principal on doit retrouver

1. Méthode de Gauss

1. Cette possibilité est intéressante si on veut centraliser les résultats pour faire un traitement informatisé des différents résultats de vos expériences. Par exemple pour faire des tableaux ou tracer des courbes en utilisant des résultats en provenance aussi bien de Scilab que d'ADA. Nous vous rappelons que vous pouvez, en Scilab, écrire des textes compilables directement par LaTeX et aussi exporter des figures en format compatible avec LaTeX.

2. Méthode de Jacobi
3. Quitter l'application

Dans le menu secondaire après avoir choisi dans le menu principal (1 ou 2), on propose à l'utilisateur trois choix de codage des réels en fonction du nombre de bits de la mantisse :

1. Nombre de bits de la mantisse 23.
2. Nombre de bits de la mantisse 35.
3. Nombre de bits de la mantisse 52.

3.3 Codage des réels

Pour évaluer l'influence du codage d'un réel en machine, c'est-à-dire du nombre de bits reservés pour la mantisse et du nombre de bits reservés pour l'exposant, on se souviendra d'abord que la précision ne dépend que de la taille de la mantisse.

On étudiera trois cas : nombre de bits de la mantisse = $23, 35$ et 52 bits respectivement, en utilisant la contrainte de préision `digits`.

3.4 Livrable

Les consignes pour l'utilisation des codes devront être fournies dans une section séparée et clairement identifiée.

Date de livraison :
– pour la partie algorithmique le 12.12.2008 ;
– pour la partie programmation en ADA le 09.01.2009

4 Partie analyse numérique

L'objectif de cette partie est l'étude de la solution d'un SEL du point de vue numérique. Cette étude doit prendre en considération les points suivants :
– le conditionnement de la matrice ;
– la dimension de la matrice, et
– la méthode de résolution utilisée.

Nous donnons ci-après une présentation détaillée pour chacun de ces points, ainsi que du livrable.

4.1 Matrice à conditionnement donné

On définit le conditionnement d'une matrice $\mathbf{A} \in \mathbb{R}^{n\times n}$ relativement à une norme matricielle $\|.\|_\alpha$ par

$$\kappa_\alpha(\mathbf{A}) = \|\mathbf{A}\|_\alpha \left\|\mathbf{A}^{-1}\right\|_\alpha$$

Dans la suite on prendra $\alpha = 2$.

Soit le SEL

$$\mathbf{Ax} = \mathbf{b}$$

On mettra en évidence ce résultat, en effectuant une étude pour le même type de matrice en prenant $\kappa = 1$, 10, 1000, 10^4, 10^8 et 10^{16} respectivement.

Pour ce faire, on engendrera des matrices carrées régulières avec la valeur du conditionnement fixée d'avance. La méthode pour créer de telles matrices est la suivante :

1. On crée la matrice diagonale

$$\mathbf{D} = \operatorname{diag}\left(1, \kappa^{-\frac{1}{n-1}}, \kappa^{-\frac{1}{n-2}}, \ldots, \kappa^{-1}\right)$$

   où n est la dimension de la matrice et κ le conditionnement défini.
   On peut vérifier que $\kappa_2(\mathbf{D}) = \kappa$.
2. On construit deux matrices carrées orthogonales $\mathbf{U}$ et $\mathbf{V}$ de même dimension que la matrice $\mathbf{A} \in \mathbb{R}^{n \times n}$.
3. On calcule la matrice $\mathbf{A}$ selon la formule

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{D} \cdot \mathbf{V}^\top$$

   Nous avons

$$\kappa_2(\mathbf{A}) = \|\mathbf{A}\|_2 \left\|\mathbf{A}^{-1}\right\|_2 = \left\|\mathbf{U}\mathbf{D}\mathbf{V}^\top\right\|_2 \left\|\mathbf{V}\mathbf{D}^{-1}\mathbf{U}^\top\right\|_2 = \|\mathbf{D}\|_2 \left\|\mathbf{D}^{-1}\right\|_2 = \kappa_2(\mathbf{D}) = \kappa$$

Les détails pratiques pour la création de la matrice $\mathbf{A}$ en Scilab sont donnés à l'annexe A.

4.2 Dimension de la matrice

La dimension de la matrice a une influence sur la qualité des résultats dans la mesure où le nombre d'opérations est différent pour des matrices de dimension différente.

On se propose d'étudier cette influence, en utilisant des matrices du même type avec des dimensions $n \times n$ différentes.

On prendra $n = 10$, 50, 100 et 150.

4.3 Méthodes de résolution de système

Il est possible que la méthode utilisée pour la résolution d'un SEL influe sur les résultats de cette résolution. Pour étudier cette question, on utilisera deux méthodes :

- Une méthode de résolution directe, faisant partie de la bibliothèque de Scilab. Il s'agit de la fonction `linsolve` qu'on pourra utiliser comme suit :
  `[x] = linsolve(A,b) ;`
  où `A` est la matrice carrée des coefficients du système, `b` est le vecteur du second membre et `x` est le vecteur solution. Pour plus de détails, se reporter à l'aide de Scilab.
- Une méthode itérative, à savoir la méthode Jacobi, dont l'exposé se trouve dans le poly du cours et que vous devez programmer.
  Les détails pour la mise en œuvre de ces méthodes sont donnés à l'annexe B.

4.4 Livrable

Un rapport écrit impérativement en LaTeX, dans lequel on présente :
- les expériences effectuées avec leurs résultats ;
- les conclusions à tirer en ce qui concerne l'utilisation des méthodes de résolution en fonction du conditionnement de la matrice et de sa dimension.

Le rapport doit être accompagné des fichiers des programmes Scilab associés à une notice d'utilisation de ces programmes.

Il y aura une note supérieure à 0 (zéro) si et seulement si le livrable est complet.

Date de livraison : 26.01.2009

5 Rapport de synthèse

Il s'agit d'établir une conclusion générale concernant le projet. Cette conclusion doit prendre en cosidération de façon synthétique tous les éléments qui ont été examinés lors des expériences, à savoir la méthode, le conditionnement, la dimension de la matrice, le codage, la prcision, le temps d'excution et le langage. Il faut penser à appuyer les conclusions qui s'y prêtent aux résultats mathématiques.

Sur le fond, le rapport attendu doit :
- permettre à un néophyte de comprendre les méthodes utilisées,
- donner les arguments conduisant aux conclusions énoncées : aucune affirmation non étayée n'est admissible,
- proposer des éléments de choix pour aider le lecteur dans une démarche de résolution de système linéaire.

5.1 Livrable

Un rapport rédigé impérativement en LaTeX.

Date de livraison : 26.01.2009

6 Recommandations

Sur la forme, il est vivement conseillé d'utiliser des tableaux et des graphiques, afin d'améliorer la lisibilité des résultats.

Le rapport doit être écrit selon les règles de l'art, dont un bref rappel est donné par le document "Écriture d'un rapport en Analyse Numérique" qui se trouve sur Arel dans XXXX et sur le site *http ://sifoci.eisti.fr*, rubrique *Analyse Numérique*.

NB.- Si l'un des quatre livrables est absent ou incomplet, la note sera de 0 (zéro) pour l'*ensemble du projet*.
Nous vous rappelons aussi que le plagiat sera sanctionné.

APPENDICES

A Construction des matrices avec Scilab

Pour faire en Scilab, les programmes nécessaires pour construire la matrice **A** il faut suivre la démarche suivante :

1. Construire la matrice diagonale

$$\mathbf{D} = \text{diag}\left(1, \kappa^{-\frac{1}{n-1}}, \kappa^{-\frac{1}{n-2}}, \ldots, \kappa^{-1}\right)$$

pour $\kappa = 1$, 10, 1000, 10^4, 10^8 et 10^{16} et $n = 10$, 50, 100 et 150.

2. Pour chacune des matrices **D**, construites précédement, calculer la matrice $\mathbf{A} = \mathbf{UDV}^\top$.

   (a) Pour l'élaboration des matrices orthogonales **U** et **V**, on procède comme suit :

   (b) On construit la matrice tridiagonale

$$\mathbf{H} = \begin{bmatrix} 5 & -4 & 1 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ -4 & 6 & -4 & 1 & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & -4 & 6 & -4 & 1 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 1 & -4 & 6 & -4 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 1 & -4 & 6 & -4 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 1 & -4 & 5 \end{bmatrix}$$

   de dimension $(n \times n)$.

      i. Remarquons que cette matrice se construit facilement, car les lignes 3 à $n-2$ se forment par un décalage circulaire droit du vecteur $\begin{bmatrix} 1 & -4 & 6 & -4 & 1 & 0 & 0 & \cdots \end{bmatrix}$.

   (c) On utilise la fonction de Scilab `svd` pour faire une décomposition en valeurs singulières de cette matrice, en utilisant l'appel

      i. `[U,S,V] = svd(H)`
      Les matrices **U** et **V** sont orthogonales.

   (d) On construit la matrice $\mathbf{A} = \mathbf{UDV}^\top$.

B Calculs à effectuer avec Scilab

Nous donnons ici à la démarche à suivre pour obtenir la solution d'un SEL avec les deux méthodes : librairie de Scilab et Jacobi.

1. Effectuer, pour les mêmes matrices que précédement, la résolution du système linéaire. La démarche à suivre est la suivante :

   (a) Créer le vecteur du second membre $\mathbf{b} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}^\top$ de dimension $(n \times 1)$.

(b) Utiliser la fonction de Scilab `linsolve` pour calculer la solution $\mathbf{x}$ du système linéaire $\mathbf{Ax} = \mathbf{b}$:
`[x] = linsolve(A, b)`

(c) Évaluer la précision de la solution obtenue, en utilisant la norme.

(d) Programmer la méthode itérative de Jacobi, dont nous donnons le formulaire :

$$\begin{aligned} \mathbf{x}(k+1) = &\quad -\mathbf{D}^{-1}(\mathbf{L}+\mathbf{R})\,\mathbf{x}(k-1) + \mathbf{D}^{-1}\mathbf{b}; k = 1, 2, \ldots ; \text{ où } k = \text{no d'itération} \\ \mathbf{x}(0) = &\quad \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}^{\top} \end{aligned}$$

avec
- $\mathbf{D}$ la diagonale de $\mathbf{A}$;
- $\mathbf{L}$ la partie de $\mathbf{A}$ strictement inférieure à la diagonale ;
- $\mathbf{R}$ la partie de $\mathbf{A}$ strictement supérieure à la diagonale.

Nous avons ainsi $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{R}$.

(e) Contrairement à la méthode directe, la précision de la solution obtenue sera fixée par vous-même. Le critère d'arrêt doit être l'erreur résiduelle relative, à savoir la valeur

$$r(k) = \frac{\|\mathbf{Ax}(k) - \mathbf{b}\|_2}{\|\mathbf{b}\|_2}$$

avec k numéro d'itération.

Pour que les résultats soient valides, il faut que $r(k) \leq 10^{-8}$ pour la valeur qui a k à l'arrêt des itérations.

(f) Le calcul itératif doit s'arrêter
- soit lorsque la valeur voulue de l'erreur résiduelle est atteinte ;
- soit lorsque le nombre d'itérations, fixé d'avance, est dépassé. Il est évident que, dans ce cas, l'erreur résiduelle ne sera pas inférieure au seuil fixé.

(g) Vous devez fixer un nombre d'itérations assez grand, par exemple 100 000.

(h) À la fin des calculs, le programme doit afficher la précision atteinte par les deux méthodes. (Remarquons que la précision du résultat n'est pas l'erreur résiduelle.) Au vu de ces résultats et si l'arrêt des itérations se fait sur le critère du nombre d'itérations, l'utilisateur doit avoir la possibilité de poursuivre les calculs, à partir du point d'arrêt, afin d'obtenir une meilleure erreur résiduelle (plus proche du seuil, voire inférieure au seuil).

N.B. Si, pendant les itérations, le critère commence augmenter, il faut arrêter les calculs en indiquant qu'il y ait eu divergence.

Pour les calculs et les différentes comparaisons, nous vous rappelons que Scilab fonctionne avec la norme IEEE-754 en double précision ce qui signifie qu'il a une mantisse de 52 bits et un exposant de 11 bits. Ainsi un nombre réel x en décimal sera compris par l'ordinateur comme le nombre binaire $(-1)^S \times (1+Y) \times 10^{E-B}$, où S est le signe, Y est la mantisse, E est l'exposant et B le biais (ici égal à 1023). Nous vous rappelons aussi que nous avons $1 + Y$ pour la mantisse à cause du bit caché.

C Méthode de Gauss-Jordan

La méthode de Gauss-Jordan est une amélioration de la méthode d'élimination de Gauss. Son objectif est de rendre la matrice $\mathbf{A}$ diagonale.

À chaque étape de l'algorithme, on élimine les valeurs de la matrice $\mathbf{A}$ correspondant à la variable x_k, non seulement des $n-k$ dernières lignes (méthode de Gauss) mais également des $k-1$ premières (variante de Jordan) et on remplace la valeur a_{kk} par 1.

Nous allons établir les formules du passage de la matrice de la $k-$ième itération $\mathbf{A}^{(k)}$ à la matrice $\mathbf{A}^{(k+1)}$.

On suppose que tous les déterminants principaux de $\mathbf{A}$ sont non nuls [2]. Ce qui implique que le terme $a_{kk}^{(k)}$ est non nul.

Les formules de passage de $\mathbf{A}^{(k)}$ à $\mathbf{A}^{(k+1)}$ et aussi de $\mathbf{b}^{(k)}$ à $\mathbf{b}^{(k+1)}$ sont

$$\begin{array}{lcll}
a_{kj}^{(k+1)} & = & \frac{a_{kj}^{(k)}}{a_{kk}^{(k)}}; & j=k+1,\ldots,n \\
a_{ij}^{(k+1)} & = & a_{ij}^{(k)} - a_{ik}^{(k)} \frac{a_{kj}^{(k)}}{a_{kk}^{(k)}}; & i=1,,\ldots,n, i\neq k, j=k+1,\ldots,n \\
a_{kk}^{(k+1)} & = & 1 & \\
b_{k}^{(k+1)} & = & \frac{b_{k}^{(k)}}{a_{kk}^{(k)}} & \\
b_{i}^{(k+1)} & = & b_{i}^{(k)} - a_{ik}^{(k)} \frac{b_{k}^{(k)}}{a_{kk}^{(k)}}; & i=1,,\ldots,n, i\neq k
\end{array}$$

À la $n-$ième étape de l'algorithme, nous avons la solution $\mathbf{x}=\mathbf{b}^{(n)}$.

Ce calcul peut aussi se faire en deux étapes, d'abord sur les lignes et ensuite sur les colonnes.

On note ℓ_k la k-ième ligne et $\mathbf{c}_k$ la k-ième colonne de la matrice $\mathbf{A}$. On suppose qu'on effectue la k-ième étape de l'algorithme. Les formules sont :

$\ell_k^{(k+1)} = \ell_k^{(k)}$
$\ell_i^{(k+1)} = -a_{ik}^{(k)}\ell_k^{(k)} + a_{kk}^{(k)}\ell_i^{(k)}$; $i \neq k$
$\mathbf{c}_k^{(k+1)} = \mathbf{c}_k^{(k)}$
$\mathbf{c}_j^{(k+1)} = -a_{kj}^{(k+1)}\mathbf{c}_k^{(k)} + a_{kk}^{(k+1)}\mathbf{c}_j^{(k)}$; $j \neq k$

D Passage des données entre Scilab et Ada

Scilab permet d'exécuter un programme écrit en Ada [3] qui :

- peut lire un fichier des données écrit en Scilab sous forme texte ou sous forme binaire ;
- modifier ce fichier ou écrire un nouveau fichier.

Au retour du programme Ada, le programme Scilab peut lire les fichiers créés ou modifiés par Ada.

De plus, nous pouvons afficher dans la console de Scilab toutes les sorties écran du programme Ada.

Notons que les donnes en Scilab et en Ada doivent tre du mme format. Comme Scilab a des réels double prcision, on travaillera en nombre réel, c'est–dire on codera les entiers en réels, et en Ada on aura le type `Float` pour les données.

2. Si cette hypothèse n'est pas vérifiée, il faut procéder à des permutations des lignes.
3. et aussi en Fortran ou en Java

Nous donnons ci-après les listings en Ada et en Scilab des programmes qui permettent la communication entre Scilab et Ada.

D.1 Échange des fichiers texte

Les programmes suivants permettent l'échange du fichier `Texte.TXT`, qui est un fichier texte, entre Scilab et Ada.

Le programme en Ada est le suivant

```
-----------------------------------------
--
--  Fichier squentiel texte
--  *  crit en Scilab;
--  *  lu en Ada;
--  *  modifi en ADA, et
--  *  rcupr par Scilab
--
-----------------------------------------

with Text_Io;
use  Text_Io;
WITH Ada.Float_Text_IO;
WITH Ada.Integer_Text_IO;
WITH Ada.Text_IO;

procedure Texte is

   F : File_Type;
   S : String (1..180);
   L : Integer;
   V : Float;

   package Fio is new Float_Io(Float);

BEGIN

   Ada.Text_IO.Put(''         ****  AFFICHAGE ADA ****'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;

   --  Lecture du fichier Texte.TXT cr par Scilab
   Ada.Text_IO.Put(''Affichage fichier Scilab'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.Put(''------------------------'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;
   Open (F,In_File,''Texte.TXT'');
   Get_Line (F,S,L);
   Ada.Text_IO.Put(''Premire ligne :  '');
   Ada.Text_IO.Put_Line (S(1..L));
```

```
   Fio.Get (F,V);
   Ada.Text_IO.Put(''Deuxime ligne :  '');
   Ada.Float_Text_IO.Put (Item =>V, Fore => 10);
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;
   Close (F);


-- Ecriture du fichier Texte.TXT
    L := 39;   -- Longueur du texte de la premire ligne du fichier
   S(1..L):=''Passage fichier texte d'ADA vers Scilab'';
   V := V + 10.0;
   Ada.Text_IO.Put(''Nouveau fichier crit par Ada'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.Put(''----------------------------'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;
   Ada.Text_IO.Put(''Premire ligne :  '');
   Ada.Text_IO.Put_Line (S(1..L));
   Ada.Text_IO.Put(''Deuxime ligne :  '');
   Ada.Float_Text_IO.Put (Item =>V, Fore => 10);
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;

   Open (F,Out_File,''Texte.TXT'');
   Put_Line (F,S(1..L));
   Fio.Put (f,V);
   Put_Line (f,'''');
   Close (F);

end Texte;
```

Le programme correspondant en Scilab est le suivant :

```
//  Echange d'un fichier texte entre Scilab et Ada

// Ecriture du fichier
  ft = mopen  (''Texte.TXT'',''w'');
  disp(''****  Affichage Scilab'');
  s='' -- Passage fichier texte de Scilab vers Ada --'';
  disp(''Premire ligne du fichier'');
  disp (s);
  mfprintf (ft,''%s\n'',s);
  val = %pi;
  disp(''Deuxire ligne du fichier'');
  disp (val);
  mfprintf (ft,''%f\n'',val);
  mclose (ft);

// Appel du programme Ada
//  v = host (''texte.exe'')  // Si =0 ==> opration correcte
```

```
  unix_w (''texte.exe'');

// Lecture du fichier modifi par Ada
  ft = mopen  (''Texte.TXT'',''r'');
  s1=mgetl (ft,1);
  disp(''****  Affichage Scilab du retour d ADA'');
  disp(''Premire ligne du fichier'');
  disp (s1);
  s2=mgetl (ft,1);
  val=sscanf (s2,''%f'');
  disp(''Deuxire ligne du fichier'');
  disp (val);
  mclose (ft);
```

Notons qu'il faut compiler le programme Ada avant de l'appeler par Scilab. En effet Scilab appelle l'exécutable de ce programme.

Si dans Scilab nous utilisons, pour l'appel du programme Ada, la commande `host(NomProgrammeEnAda.exe)` le programme sera exécuté mais les affichages d'Ada n'apparaîtront pas à la console de Scilab. Si on veut que ces affichages apparaissent, il faut utiliser la commande `unix_w(NomProgrammeEnAda.exe)`.

D.2 Échange des fichiers séquentiels binaires

Comme pour les fichiers texte, on peut aussi échanger des fichiers binaires entre Scilab et Ada. Nous donnons les listings des programmes qui font l'échange du fichier `Data.DAT` (qui est un fichier séquentiel). La démarche pour un fichier binaire en accès direct est la même.

Le programme en Ada est le suivant :

```
--------------------------------------------
--
--  Fichier squentiel binaire
--  *  \UNICODE{0xe9}crit en Scilab;
--  *  lu en Ada;
--  *  modifi en ADA, et
--  *  rcupr par Scilab
--
--------------------------------------------

WITH Ada.Float_Text_IO;
WITH Ada.Text_IO;
with Sequential_io;

procedure Data is

   package sio is new Sequential_Io(Float);

   F : sio.File_Type;
   T : array (1..10) of Float;

begin
```

```
   Ada.Text_IO.Put(''         ****  AFFICHAGE ADA ****'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;

   --  Lecture du fichier Data.DAT cr par Scilab
   Ada.Text_IO.Put(''Affichage fichier Scilab'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.Put(''------------------------'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;
   Sio.Open (F,Sio.In_File,''Data.DAT'');
   for I in 1..10 loop
      Sio.Read (F,T(I));
      Ada.Float_Text_IO.Put (Item =>T(I), Fore => 6, Aft => 0, Exp => 0);
   end loop;
   Sio.Close (F);
   Ada.Text_IO.New_Line;

-- Ecriture du fichier Data.DAT
   for I in 1..10 loop
      T(I):=10.0+T(I);
   end loop;

   Ada.Text_IO.Put(''Nouveau fichier crit par Ada'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.Put(''-----------------------------'');
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;

   Sio.Open (F,Sio.Out_File,''Data.DAT'');
   for I in 1..10 loop
      Sio.Write (F,T(I));
      Ada.Float_Text_IO.Put (Item =>T(I), Fore => 6, Aft => 0, Exp => 0);
   end loop;
   Sio.Close (F);
   Ada.Text_IO.New_Line;
   Ada.Text_IO.New_Line;

end Data;
```

Le programme Scilab est le suivant :

```
//  Aler-retour d'un fichier binaire squentiel entre Scilab et Ada

  t=[];
  for i=1:10 do
    t(i)=i;
  end;
  t=t';  // Ada renvoie un vecteur ligne
  disp(''****  Affichage Scilab'');
```

```
  disp (t);

// Ecriture du fichier
  fd = mopen  (''Data.DAT'',''wb'');
  mput (t,''f'',fd);
  mclose (fd);

// Appel du programme Ada
  unix_w (''data.exe'');

// Lecture du fichier modifi par Ada
  fd = mopen  (''Data.DAT'',''rb'');
  t=mget (10,''f'',fd);
  mclose (fd);

  disp(''****  Affichage Scilab du retour d ADA'');
  disp (t);
```