ISIMA 1 - Programmation Numérique - 1ère session 3 février 2012
Documents autorisés : cours, notes de cours & TP
Durée : 2 heures

Les deux problèmes doivent être rédigés sur des feuilles séparées.

1 Méthode de Givens-Householder

Soient $A \in \mathcal{M}_n(\mathbb{R})$, une matrice réelle symétrique et $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ ses valeurs propres.
On souhaite utiliser la réduction de Householder pour construire une matrice orthogonale O telle que $B = O^T AO$ soit tridiagonale :

$$B = O^T AO = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \dots & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \ddots & \vdots \\ 0 & \beta_2 & \ddots & \ddots & 0 \\ \vdots & \ddots & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_n \end{bmatrix} \tag{1}$$

L'algorithme de tridiagonalisation est le suivant :

Pour k=1..n-2 :
  $H \leftarrow$ *matrice de Householder qui annule les éléments sous la sous-diagonale de la colonne k de A*
  $O \leftarrow OH$
  $A \leftarrow HAH^T$

Question 1.1.
a. Ecrivez un sous-programme `householder` qui détermine, pour $j < n$, la matrice de Householder annulant les j dernières composantes d'un vecteur $x \in \mathbb{R}^n$ (cf. TP3 sur la méthode QR).
b. Ecrivez le sous-programme `tridiagonalisation` qui triagonalise une matrice A supposée symétrique.

Soit $B \in \mathcal{M}_n(\mathbb{R})$ la matrice tridiagonale symétrique définit par l'équation (1).
Soit p_i la suite de polynômes définie par la formule de récurrence suivante :

$$\begin{cases} p_0(\lambda) & = 1 \\ p_1(\lambda) & = \alpha_1 - \lambda \\ p_i(\lambda) & = (\alpha_i - \lambda)p_{i-1}(\lambda) - \beta_{i-1}^2 p_{i-2}(\lambda) \quad \text{pour } 2 \leq i \leq n \end{cases}$$

Alors :
– p_n est le polynôme caractéristique de B,
– le nombre de racines de p_n inférieures à λ est le nombre $N(\lambda)$ de changements de signe dans la suite $\big(1, p_1(\lambda), ..., p_n(\lambda)\big)$.

Question 1.2. Ecrivez une fonction `nbracines` qui retourne, pour $\lambda \in \mathbb{R}$, le nombre de racines $N(\lambda)$ inférieures à λ du polynôme caractéristique de B.

Cela nous permet d'établir un algorithme de dichotomie (*bissection de Givens*) pour le calcul d'une valeur propre λ_i de B (et donc de A par similitude). En effet pour un intervalle $[t_0, t_1]$ tels que $\lambda_i \in [t_0, t_1]$, et $m = (t_0 + t_1)/2$:

$$\begin{cases} \lambda_i \in [t_0, m[& \text{si } N(m) \geq i \\ \lambda_i \in [m, t_1] & \text{sinon.} \end{cases}$$

On peut initialiser la dichotomie en choisissant l'intervalle initial $\big[-\|B\|, \|B\|\big]$ avec une norme subordonnée.

Question 1.3. Ecrivez l'expression qui permet de calculer efficacement la norme $\|.\|_\infty$ ou $\|.\|_1$ d'une matrice tridiagonale. (Rappel : $\|A\|_\infty = \max_i \sum_j |a_{ij}|$ et $\|A\|_1 = \max_j \sum_i |a_{ij}|$)

Question 1.4. Ecrivez le sous-programme `givens_householder` qui détermine les valeurs propres d'une matrice A symétrique réelle.

II- Problème de transport à 2 indices (*à rédiger sur feuilles séparées*)

Il s'agit de transporter au moindre coût des produits depuis M sources vers N destinations. Le coût unitaire de transport de la source i vers la destination j de X_{ij} produits est C_{ij}. La quantité de produits fournie (offre) par la source i est S_i et celle reçue (demande) par la destination j est D_j (établie à partir des commandes) avec :

$$S_i=\sum_{j=1}^{N} X_{ij}, 1\le i\le M \quad \text{et} \quad D_j=\sum_{i=1}^{M} X_{ij}, 1\le j\le N \qquad (1)$$

Pour que le problème ait une solution il faut que l'offre globale soit supérieure ou égale à la demande globale mais on ajuste l'offre pour avoir l'égalité $SG = DG$:

$$SG=\sum_{i=1}^{M} S_i \quad \text{et} \quad DG=\sum_{j=1}^{N} D_j$$

Le coût du transport est $CG=\sum_{i=1}^{M}\sum_{j=1}^{N} C_{ij}X_{ij}$

avec la contrainte $SG = DG$.

1- Quel est le nombre de variables X_{ij} ? Justifier le fait que le rang du système linéaire (1) est inférieur ou égal à $M + N - 1$.

L'algorithme de recherche d'une solution optimale est le suivant :

a) Vérifier $SG = DG$ (arrêt sinon).
b) Générer une **solution initiale.**
c) **Méthode itérative de recherche** limitée à ***ItMax*** itérations :
 - c.1) Afficher le coût de la solution X.
 - c.2) Si la **condition d'optimalité** est vérifiée alors arrêter la recherche.
 - c.3) **Améliorer** la solution X.

d) Afficher la solution X, son coût CG et le nombre d'itérations utilisées ***Iter***.

Programmation **FORTRAN90.**
On considère que les **tableaux** ont une **allocation statique.** Les **booléens** seront déclarés avec le type adéquat, jamais comme entier ! Les **sous-programmes** seront placés dans un **module.**

2- Ecrire la fonction réelle **Cout** qui calcule CG et dont vous expliciterez les paramètres.
3- Ecrire l'en-tête de la fonction booléenne **Optimale** qui retourne la valeur équivalente à vrai quand la solution X est optimale. Ecrire l'en-tête de la procédure **Ameliorer** qui améliore X.

La procédure **CoinNordOuest** génère une solution initiale X. Au départ X est initialisé à zéro, on choisit d'abord $l = c = 1$ et on pose $X_{11} = \min(S_1, D_1)$.
Puis on se déplace :
à droite si $S_l \ge D_c$ ($c = c + 1$) ou en bas sinon ($l = l + 1$)
arrivé en (l, c) on détermine X_{lc} le plus grand possible tel que $\sum_{j=1}^{N} X_{lj}\le S_l$ et $\sum_{i=1}^{M} X_{ic}\le D_c$.
Si la somme en ligne a été égalée on descend sinon on va à droite et on détermine le nouvel X_{lc}...
La marche s'arrête quand on ne peut plus descendre ou aller à droite.

4- Précisez la **condition d'arrêt** de la procédure précédente. Combien de **variables** détermine-t-on (raisonner sur un schéma de la marche) ? Ecrire la procédure **CoinNordOuest.**
5- Ecrire le **programme principal** conformément à l'algorithme de **recherche d'une solution optimale.** Les données M, N, C, S et D sont lues dans le fichier « Don.txt » handle n°10. Vous n'utiliserez que les sous-programmes définis précédemment et **aucun autre.** Les sous-programmes **Optimale** et **Ameliorer** sont supposés déjà définis.