UNIVERSITÉ DE GENÈVE
FACULTÉ DES SCIENCES
Section de mathématiques

Analyse Numérique
Corrigé Série 13

19 décembre 2019

1. (**Elimination de Gauss – version de Doolittle**) On considère l'algorithme de Doolittle suivant pour calculer la décomposition $A = LU$ d'une matrice $A \in \mathbb{R}^{n\times n}$ à l'aide de la méthode de Gauss, ici formulé sans recherche de pivot :

   (Algorithme de Doolittle)

   $U \leftarrow A,\ L \leftarrow I$
   for $k = 1,\dots,n$ **do**
       for $j = k,\dots,n$ **do**
           $u_{kj} \leftarrow a_{kj} - \boldsymbol{\ell}_{k,1:k-1}^T \boldsymbol{u}_{1:k-1,j}$
       for $i = k+1,\dots,n$ **do**
           $\ell_{ik} \leftarrow (a_{ik} - \boldsymbol{\ell}_{i,1:k-1}^T \boldsymbol{u}_{1:k-1,k})/u_{kk}$

   (a) Justifier que l'on peut remplacer les termes a_{kj}, a_{ik} qui apparaissent dans l'algorithme ci-dessus par respectivement u_{kj}, u_{ik}, sans modifier le résultat de l'algorithme.

   Sol.: *En effet, au début de l'algorithme on a $U = A$, et on remarque qu'à l'étape k de l'algorithme, u_{kj}, u_{ik} n'on encore jamais été modifiés, et valent donc a_{kj}, a_{ik}, respectivement.*

   (b) Justifier que cet algorithme met en oeuvre l'algorithme de la décomposition $A = LU$, où L est une matrice triangulaire inférieure avec des 1 sur la diagonale.

   Indication. Pour les parties triangulaires supérieure et inférieure de la matrice A, montrer :

   $$\begin{aligned} a_{kj} &= \boldsymbol{\ell}_{k,1:k-1}^T \boldsymbol{u}_{1:k-1,j} + \ell_{kk} u_{kj}, \quad j = k,\dots,n, \\ a_{ik} &= \boldsymbol{\ell}_{i,1:k-1}^T \boldsymbol{u}_{1:k-1,k} + \ell_{ik} u_{kk}, \quad i = k+1,\dots,n. \end{aligned}$$

   Sol.: *L'égalité $A = LU$ s'écrit*

   $$a_{ij} = (LU)_{ij} = \sum_{r=1}^{\min(i,j)} \ell_{ir} u_{rj}, \quad i,j = 1,\dots n.$$

   car L est triangulaire inférieure et A est triangulaire supérieure. Ceci correspond aux formules de l'indication. En remarquant $\ell_{kk} = 1$, l'algorithme calcule donc bien les coefficients ℓ_{ik} et u_{ki} par récurrence sur $k = 1,\dots n, i = k,\dots n$.

   (c) (0.5 pts) ($\star$) Modifier l'algorithme pour ajouter la recherche partielle de pivot.

   Sol.:

   $U \leftarrow A,\ L \leftarrow I,\ P \leftarrow I$
   for *$k = 1,\dots,n$* ***do***
       Chercher $i \geq k$ tel que $|u_{ik}|$ soit maximal
       if *$u_{ik} \neq 0$* ***then***
           $\boldsymbol{u}_{k,k:n} \leftrightarrow \boldsymbol{u}_{i,k:n}$ (échanger deux lignes)
           $\ell_{k,1:k-1} \leftrightarrow \ell_{i,1:k-1}$
           $\boldsymbol{p}_{k,1:n} \leftrightarrow \boldsymbol{p}_{i,1:n}$
           for *$j = k,\dots,n$* ***do***
               $u_{kj} \leftarrow u_{kj} - \boldsymbol{\ell}_{k,1:k-1}^T \boldsymbol{u}_{1:k-1,j}$
           for *$i = k+1,\dots,n$* ***do***
               $\ell_{ik} \leftarrow (u_{ik} - \boldsymbol{\ell}_{i,1:k-1}^T \boldsymbol{u}_{1:k-1,k})/u_{kk}$

2. ($\star$, tout l'exercice) **(Décomposition LU et élimination de Gauss)**
   Soit $A = (a_{ij})$ une matrice carrée à diagonale dominante par colonne, c.-à-d.,

   $$|a_{jj}| > \sum_{i\neq j} |a_{ij}|,\ \forall j. \tag{1}$$

(a) (0.25 pts) Montrer que A est inversible.

Indication : Montrer que $A^\top \boldsymbol{x} = \mathbf{0}$ implique $\boldsymbol{x} = \mathbf{0}$.

Sol.: *Soit un vecteur $\boldsymbol{x}$ tel que $A^\top \boldsymbol{x} = \mathbf{0}$. Il suffit de montrer $\boldsymbol{x} = \mathbf{0}$ (en effet noyau nul équivaut à inversible pour une matrice carrée, et A inversible équivaut à $A^\top$ inversible). Par l'absurde, supposons $\boldsymbol{x} \neq \mathbf{0}$ et soit $x_j \neq 0$ la plus grande composante de $\boldsymbol{x}$ en valeur absolue. On a $\sum_i a_{ij} x_i = 0$, d'où $a_{jj} x_j = -\sum_{i \neq j} a_{ij} x_i$. On déduit*

$$|a_{jj} x_j| \leq \sum_{i \neq j} |a_{ij}||x_i| \leq \sum_{i \neq j} |a_{ij}||x_j|.$$

En divisant par $x_j \neq 0$, on obtient $|a_{jj}| \leq \sum_{i \neq j} |a_{ij}|$ ce qui contredit l'hypothèse.

(b) (0.25 pts) Démontrer qu'au premier pas d'une décomposition LU de A, il n'y a pas d'échanges de lignes, même si on utilise la recherche de pivot partielle (c'est-à-dire une recherche de pivots dans la même colonne à chaque étape de l'algorithme).

Sol.: *Comme A est une matrice à diagonale dominante on a $|\ell_{i1}| = |\frac{a_{i1}}{a_{11}}| < 1$. Donc une recherche de pivot n'est pas nécessaire.*

(c) (0.25 pts) Démontrer $\displaystyle\sum_{i=2}^{n} |\ell_{i1}| < 1$ où $\ell_{i1} = a_{i1}/a_{11}$.

Sol.: *A partir de l'équation (1) on a pour $j = 1$, $|a_{11}| > \sum_{i=2}^{n} |a_{i1}|$, d'où*

$$\sum_{i=2}^{n} |\ell_{i1}| = \sum_{i=2}^{n} |\frac{a_{i1}}{a_{11}}| < 1. \tag{2}$$

(d) (0.75 pts) En utilisant les points précédents et un raisonnement par récurrence, démontrer que pendant la décomposition LU de A, il n'y a pas d'échanges de lignes, même si on utilise la recherche de pivot partielle.

Sol.: *On rappelle que pour la première itération on a*

$$a_{ij}^{(1)} = a_{ij} - \ell_{i1} a_{1j},$$

On va faire deux estimations de $a_{ij}^{(1)}$, pour $i \neq j$

$$|a_{ij}^{(1)}| = |a_{ij} - \ell_{i1} a_{1j}| \leq |a_{ij}| + |\ell_{i1}||a_{1j}| \tag{3}$$

et pour $i = j$

$$|a_{jj}^{(1)}| = |a_{jj} - \ell_{j1} a_{1j}| \geq |a_{jj}| - |\ell_{j1}||a_{1j}|. \tag{4}$$

Donc on peut sommer (3) sur toute la colonne et utiliser (2)

$$\begin{aligned}
\sum_{\substack{i \neq j \\ 2 \leq i \leq n}} |a_{ij}^{(1)}| &\leq \sum_{\substack{i \neq j \\ 2 \leq i \leq n}} |a_{ij}| + \sum_{\substack{i \neq j \\ 2 \leq i \leq n}} |\ell_{i1}||a_{1j}| \\
&\leq \sum_{\substack{i \neq j \\ 2 \leq i \leq n}} |a_{ij}| + (1 - |\ell_{j1}|)|a_{1j}| \\
&= \sum_{\substack{i \neq j \\ 1 \leq i \leq n}} |a_{ij}| - |\ell_{j1}||a_{1j}|
\end{aligned}$$

On peut utiliser maintenant (1) et l'estimation (4)

$$\begin{aligned}
\sum_{i \neq j} |a_{ij}^{(1)}| &< |a_{jj}| - |\ell_{j1}||a_{1j}| \\
&\leq |a_{jj}^{(1)}|.
\end{aligned}$$

Donc après la première itération on a que la sous matrice de taille $(n-1) \times (n-1)$ est aussi diagonale dominante.

Finalement on peut conclure que si on réitère l'idée on peut faire une élimination de Gauss sans avoir besoin de chercher du pivot.

3. **(Algorithme de Thomas)**
L'*algorithme de Thomas* est une formulation simplifiée de la factorisation LU qui peut être utilisée pour résoudre des systèmes linéaires $A\boldsymbol{x} = \boldsymbol{b}$ où la matrice A est une *matrice tridiagonale*.
Soit A une matrice tridiagonale (de taille $n \times n$) :

$$A = \begin{pmatrix} d_1 & r_1 & & & \\ \ell_2 & d_2 & r_2 & & \\ & \ell_3 & \ddots & \ddots & \\ & & \ddots & \ddots & r_{n-1} \\ & & & \ell_n & d_n \end{pmatrix}.$$

(a) Calculer la décomposition LU de A.
Sol.: *En multipliant L et U on trouve que*

$$\begin{pmatrix} 1 & 0 & & & \\ \ell_{21} & 1 & 0 & & \\ \ell_{31} & \ell_{32} & 1 & & \\ & & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n(n-1)} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1(n-1)} & u_{1n} \\ 0 & u_{22} & \cdots & u_{2(n-1)} & u_{2n} \\ 0 & 0 & u_{32} & \cdots & u_{3n} \\ & & \ddots & \ddots & \\ 0 & 0 & \cdots & 0 & u_{nn} \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1(n-1)} & u_{1n} \\ * & * & \cdots & * & * \\ * & * & * & \cdots & * \\ & & \ddots & \ddots & \\ * & * & \cdots & * & * \end{pmatrix}$$

$$\Rightarrow u_{13} = u_{14} = \cdots = u_{1_n} = 0.$$

Puis

$$\begin{pmatrix} 1 & 0 & & & \\ \ell_{21} & 1 & 0 & & \\ \ell_{31} & \ell_{32} & 1 & & \\ & & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n(n-1)} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & 0 & \cdots & 0 \\ 0 & u_{22} & \cdots & \cdots & u_{2n} \\ 0 & 0 & u_{32} & \cdots & u_{3n} \\ & & \ddots & \ddots & \\ 0 & 0 & \cdots & 0 & u_{nn} \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} & 0 & \cdots & 0 \\ * & * & u_{23} & \cdots & u_{2n} \\ * & * & * & \cdots & * \\ & & \ddots & \ddots & \\ * & * & \cdots & * & * \end{pmatrix}$$

$$\Rightarrow u_{24} = u_{25} = \cdots = u_{2n} = 0,$$

etc. Ainsi

$$U = \begin{pmatrix} \delta_1 & \rho_1 & & & \\ & \delta_2 & \rho_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \rho_{n-1} \\ & & & & \delta_n \end{pmatrix},$$

Puis

$$\begin{pmatrix} 1 & 0 & & & \\ \ell_{21} & 1 & 0 & & \\ \ell_{31} & \ell_{32} & 1 & & \\ & & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n(n-1)} & 1 \end{pmatrix} \begin{pmatrix} \delta_1 & \rho_1 & & & \\ & \delta_2 & \rho_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \rho_{n-1} \\ & & & & \delta_n \end{pmatrix} = \begin{pmatrix} \delta_1 & * & 0 & \cdots & 0 \\ \ell_{21}\delta_1 & * & * & \cdots & 0 \\ \ell_{31}\delta_1 & * & * & \cdots & 0 \\ & & \ddots & \ddots & \\ \ell_{1n}\delta_1 & * & \cdots & * & * \end{pmatrix}$$

$$\Rightarrow \ell_{31} = \ell_{41} = \cdots = \ell_{n1} = 0.$$

Puis

$$\begin{pmatrix} 1 & 0 & & & \\ \ell_{21} & 1 & 0 & & \\ 0 & \ell_{32} & 1 & & \\ & & \ddots & \ddots & \\ 0 & \ell_{n2} & \cdots & \ell_{n(n-1)} & 1 \end{pmatrix} \begin{pmatrix} \delta_1 & \rho_1 & & & \\ & \delta_2 & \rho_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \rho_{n-1} \\ & & & & \delta_n \end{pmatrix} = \begin{pmatrix} * & * & 0 & \cdots & 0 \\ * & * & * & \cdots & 0 \\ * & * & * & \cdots & 0 \\ * & \ell_{42}\delta_2 & * & \cdots & * \\ & & \ddots & \ddots & \\ * & \ell_{n2}\delta_2 & \cdots & * & * \end{pmatrix}$$

$$\Rightarrow \ell_{42} = \ell_{43} = \cdots = \ell_{4n} = 0.$$

Ainsi

$$L = \begin{pmatrix} 1 & & & & \\ \lambda_2 & 1 & & & \\ & \lambda_3 & \ddots & & \\ & & \ddots & \ddots & \\ & & & \beta_n & 1 \end{pmatrix},$$

Calculons les coefficients λ_i, ρ_i *et* δ_i

$$(LU)_{i,j} = \sum_{k=1}^{n} L_{i,k}U_{k,j} = L_{i,i-1}U_{i-1,j} + L_{i,i}U_{i,j},$$

pour $j \in \{i-1, i, i+1\}$.
Ainsi on trouve que $(LU)_{i,i-1} = \ell_i$, $(LU)_{i,i} = d_i$, $(LU)_{i,i+1} = r_i$, *ce qui donne*

$$\lambda_i\delta_{i-1} = \ell_i, \qquad \lambda_i\rho_{i-1} + \delta_i = d_i, \qquad \rho_i = r_i,$$

d'où la solution

$$\lambda_i = \ell_i/\delta_{i-1}, \qquad \delta_i = d_i - \lambda_i r_{i-1}, \qquad \rho_i = r_i, \qquad \text{pour } i = 2, \ldots, n,$$

avec $\delta_1 = d_1$.

(b) Quel est le nombre approximatif d'opérations dont on a besoin pour construire L et U ? Donner la solution sous la forme Cn^k.

Sol.: *D'après le point précédent on voit que l'on a besoin d'environ* $2n$ *opérations pour construire les* δ_i *et* n *opérations pour construire les* λ_i. *Il faut donc environ* $3n$ *opérations pour construire* L *et* U.

(c) On suppose maintenant que L et U sont construits. Résoudre le système linéaire $A\boldsymbol{x} = \boldsymbol{b}$ en utilisant les résultats précédents. Quel est le nombre approximatif d'opérations, donné sous la forme Cn^k, de cet algorithme ?

Sol.: *Les matrices L, U étant calculées et vu que $A\boldsymbol{x} = \boldsymbol{b} \Leftrightarrow L\boldsymbol{y} = \boldsymbol{b}$, $U\boldsymbol{x} = \boldsymbol{y}$, on résout d'abord $L\boldsymbol{y} = \boldsymbol{b}$:*

$$\begin{aligned} y_1 &= b_1, \\ y_i &= b_i - \lambda_i y_{i-1} \qquad \textit{pour } i = 2, \dots, n, \end{aligned}$$

et après on résout $U\boldsymbol{x} = \boldsymbol{y}$:

$$\begin{aligned} x_n &= y_n/\delta_n, \\ x_i &= (y_i - r_i x_{i+1})/\delta_i \qquad \textit{pour } i = n-1, \dots, 1. \end{aligned}$$

Il faut environ $2n$ opérations pour calculer les y_i et $3n$ opérations pour construire les x_i. Le nombre d'opérations est donc environ $5n$.