Informatique tronc commun – TP n° 13
Décomposition LU

Mercredi 5 avril 2017

1. **Lisez attentivement tout l'énoncé avant de commencer.**
2. Commencez la séance en créant un dossier au nom du TP dans le répertoire dédié à l'informatique de votre compte.
3. Ayez toujours un crayon et un papier sous la main. Quand vous réfléchissez à une question, utilisez les !
4. Vous devez être autonome. Ainsi, avant de poser une question à l'enseignant, merci de commencer par :
   - relire l'énoncé du TP (beaucoup de réponses se trouvent dedans) ;
   - relire les passages du cours [1] relatifs à votre problème ;
   - effectuer une recherche dans l'aide disponible sur votre ordinateur (ou sur internet) concernant votre question.

   Il est alors raisonnable d'appeler votre enseignant pour lui demander des explications ou une confirmation !

On s'intéresse dans ce TP à la mise en œuvre de différentes méthodes de résolution de systèmes linéaires, ainsi qu'à l'étude de leur efficacité.

Instructions de rendu

Attention : suivez précisément ces instructions. Vous enverrez à votre enseignant un fichier d'extension `.py` ainsi qu'un graphique au format PNG (extension `.png`).

Votre fichier portera un nom du type `tpXX_berne_zannad.py`, où `XX` est à remplacer par le numéro du TP et les noms de vos enseignants par ceux des membres du binôme. Le nom de ce fichier ne devra comporter ni espace, ni accent, ni apostrophe, ni majuscule. Dans ce fichier, vous respecterez les consignes suivantes.

- Écrivez d'abord en commentaires (ligne débutant par #), le titre du TP, les noms et prénoms des étudiants du groupe.
- Commencez chaque question par son numéro écrit en commentaires.
- Les questions demandant une réponse écrite seront rédigées en commentaires.

1. Dans le cas fort improbable où vous ne vous en souviendriez pas.

— Les questions demandant une réponse sous forme de fonction ou de script respecteront pointilleusement les noms de variables et de fonctions demandés.

La figure demandée à la dernière question (facultative) portera, comme d'habitude, un nom du type `tp13_berne_zannad_q10.png`, où les noms de vos enseignants sont à remplacer par ceux des membres du binôme.

1 Introduction

Lisez cette partie *chez vous* avant de commencer le TP.

1.1 Décomposition LU

Soit A une matrice carrée telle que toute sous-matrice principale (sous-matrice carrée calée dans le coin supérieur gauche) soit inversible.

On rappelle que :

— l'opération $L_i \leftarrow L_i + \alpha L_j$ effectuée sur les lignes d'une matrice A, revient à effectuer le produit $T(i, j, \alpha) \times A$, où $T(i, j, \alpha)$ est la matrice de transvection égale à l'identité, à l'exception du coefficient (i, j) qui vaut α ;

— l'opération $C_i \leftarrow C_i + \alpha C_j$ effectuée sur les colonnes d'une matrice A, revient à effectuer le produit $A \times T(j, i, \alpha)$.

On peut alors montrer que la matrice A admet une décomposition $A = LU$, où L est une matrice triangulaire inférieure et U une matrice triangulaire supérieure. La matrice L est obtenue en appliquant à la matrice identité les opérations sur les colonnes correspondant aux inverses des matrices d'opérations sur les lignes permettant de trianguler A par la méthode de Gauss (sans échange de ligne : toute sous-matrice de A étant principale, on peut montrer que si la matrice A est rendue triangulaire pour ses k premières colonnes, alors le coefficient diagonal de la colonne suivante n'est pas nul et peut être choisi comme pivot).

Plus précisément : si T_1, ... , T_p sont les p transvections successives à appliquer à A pour la rendre triangulaire supérieure, nous obtenons : $(T_p \times \ldots \times T_1) \times A = U$. Il faut alors poser $L = (T_p \times \ldots \times T_1)^{-1} = I_n \times T_1^{-1} \times \ldots \times T_p^{-1}$.

L'intérêt réside dans le fait qu'on ramène la résolution de $AX = B$ à la résolution de 2 systèmes triangulaires. Ceci s'avère notamment intéressant lorsque l'on a plusieurs systèmes à résoudre associés à la même matrice A.

1.2 Matrices dans Python avec `numpy`

Nous allons écrire un programme permettant de calculer cette décomposition LU, et le comparer à l'algorithme du pivot de Gauss classique.

Pour cela, nous écrirons les matrices sous forme de tableaux bi-dimensionnels de type `array`, en utilisant le module `numpy`, dont voici quelques exemples d'utilisations :

```
>>> import numpy as np
>>> A=np.array([[1, 2], [3, 4]])
>>> A
array([[1, 2],
```

```
       [3, 4]])
>>> B=np.eye(2)
>>> B[1,0]=2
>>> B
array([[ 1.,  0.],
       [ 2.,  1.]])
>>> C = A.dot(A) + 2*B.dot(A)
>>> C
array([[  9.,  14.],
       [ 25.,  38.]])
```

On a bien calculé $C = A^2 + 2BA$, avec $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ et $B = T(1, 0, 2)$.

Nous utiliserons aussi les fonctions définies dans le fichier `random_matrix`, disponible sur le site de la classe, ainsi que la fonction `clock` du module `time` pour chronométrer les temps d'éxécution.

Après avoir enregistré le fichier `random_matrix` dans votre dossier de travail, vous pourrez utiliser les fonctions écrites dedans en l'utilisant comme une bibliothèque Python, par l'une des commandes suivantes.

```
import random_matrix as rm
from random_matrix import *
```

On peut mesurer une durée en faisant la différence entre deux appels de la fonction `clock` (attention, son fonctionnement diffère selon les systèmes d'exploitation, mais nous ne en soucierons pas ici), comme suit.

```
>>> from time import clock
>>> tic = clock()
>>> tic
0.176583
>>> x = [i**2 for i in range(10**5)]
>>> toc = clock()
>>> tic, toc
(0.176583, 0.264413)
>>> t = toc-tic
>>> t # temps écoulé en secondes
0.08783000000000002
```

2 Travail demandé

Q1 Écrire une fonction `trans_ligne(A,i,j,alpha)` prenant en argument un quadruplet `(A,i,j,alpha)`, où A est une matrice d'ordre n, $i, j \in [\![0, n[\![$, et $\alpha \in \mathbb{R}$, et qui multiplie A à gauche par la matrice $T(i, j, \alpha)$, c'est-à-dire qui applique à A la transvection $L_i \leftarrow L_i + \alpha L_j$. Ainsi cette fonction modifie A et ne renvoie rien.

Attention : par souci d'efficacité, il est préférable de voir cette opération comme une opération entre des lignes de A, plutôt que comme un produit matriciel. On s'arrangera

autant que possible pour éviter les calculs inutiles sur les coefficients qu'on sait être nuls. Ainsi cette fonction modifie A et ne renvoie rien.

Q2 Écrire une fonction `trans_colonne(A,i,j,alpha)` prenant en argument un quadruplet `(A,i,j,alpha)`, où A est une matrice d'ordre n, $i, j \in [\![0, n[\![$, et $\alpha \in \mathbb{R}$, et qui multiplie A à droite par **l'inverse** de la matrice $T(i, j, \alpha)$, c'est-à-dire qui applique à A la transvection $C_j \leftarrow C_j - \alpha C_i$.

Attention : les remarques de la question précédente sont toujours d'actualité.

Q3 Écrire une fonction `LU(A)` prenant en argument une matrice carrée `A` retournant les deux matrices L et U de la décomposition ci-dessus.

Q4 Écrire une fonction `resolution_sup(U,B)` prenant en argument un couple de matrices `(U,B)`, où U est une matrice triangulaire supérieure et B est une matrice ayant autant de lignes que U, et renvoyant la solution du système $UX = B$.

Q5 Écrire une fonction `resolution_inf(L,B)` prenant en argument un couple de matrices `(L,B)`, où L est une matrice triangulaire inférieure et B est une matrice ayant autant de lignes que L, et renvoyant la solution du système $LX = B$.

Q6 Écrire une fonction `temps_LU(A,B)` prenant en argument un couple de matrices `(A,B)` et qui renvoie le temps de résolution du système $AX = B$ par les opérations suivantes :

— calcul (une fois pour toutes) de la décomposition LU de A ;

— pour chaque colonne de B, notée b_i, résolution du système $AX = b_i$ à l'aide de deux systèmes triangulaires.

Q7 Écrire une fonction `temps_pivot(A,B)` prenant en argument un couple de matrices `(A,B)` et qui renvoie le temps de résolution du système $AX = B$ par les opérations suivantes :

— pour chaque colonne de B, notée b_i, résolution du système $AX = b_i$ à l'aide de l'algorithme du pivot de Gauss (vu en cours).

Q8 Écrire une fonction `temps_inverse(A,B)` prenant en argument un couple de matrices `(A,B)` et qui renvoie le temps de résolution du système $AX = B$ par les opérations suivantes :

— calcul (une fois pour toutes) de l'inverse de A par l'algorithme du pivot de Gauss ;

— pour chaque colonne de B, notée b_i, résolution du système $AX = b_i$ par le produit matriciel $A^{-1}b_i$.

Q9 Comparer les résultats obtenus aux trois questions précédentes sur des matrices $A \in \mathscr{M}_{100}(\mathbb{R})$ et $B \in \mathscr{M}_{100,25}(\mathbb{R})$, choisies aléatoirement (les coefficients étant pris uniformément dans $[\![0, 100]\!]$).

Vous affecterez les matrices utilisées respectivement aux variables `A` et `B` dans votre script, ainsi que les temps calculés respectivement aux variables `t_LU`, `t_pivot` et `t_inverse`.

Q10 *Question facultative :* plus généralement, tracer, pour $k \in [\![1, 25]\!]$, chaque liste de points correspondant au temps de réponse pour la résolution de k systèmes associés à $A \in \mathscr{M}_{100}(\mathbb{R})$, par chacune des méthodes précédentes (décomposition LU, pivot de Gauss et calcul de A^{-1}). Commenter.

Vous enverrez le graphe produit à votre enseignant.