Calcul matriciel avec Octave

Isar Stubbe

version de 6-11-2018

Table de matières.

1. Introduction au logiciel

Exercice 1. Pour le calcul matriciel sur ordinateur, nous utiliserons un logiciel libre adapté: Octave (`www.octave.org`). On peut utiliser la Command Window comme une calculatrice capable de faire du calcul matriciel. Executer les exemples suivants:

```
>> % ceci est un commentaire
>> clc % vider la fenêtre
>> clear % supprimer les variables éventuellement définies auparavant
>> close all % fermer toutes les figures éventuellement faites auparavant
>> disp('Hello World!')
>> 1+1 % on peut calculer
>> A=[1,2,3,4;5,6,7,8;9,10,11,12] % une matrice
>> A=A+A % redéfinir la matrice
>> B=[8,7;6,5;4,3;2,1] % une autre matrice
>> B(1:2,1:2) % construction d'une sous-matrice
>> [A(:,1:2),B(1:3,1)] % assemblage de matrices
>> [A(1:2,1:2);B]
>> C=A*B % le produit matriciel
>> A.*A % le produit élément-par-élément
>> D=2*A % le double d'une matrice
>> E=size(A) % le genre d'une matrice (lignes x colonnes)
>> F=ones(E)
>> G=zeros(E)
>> A'
>> A+F
>> (A*A')^2
>> A(2,3) % afficher l'élément à la place (i,j) de A
>> H=A(:,2)
```

```
>> I=A(4,:) % aide quand on fait une erreur
>> I=A(3,:)
>> max(A)
>> help max % aide pour une commande
>> J=ones(100,100)
>> J=ones(100000,100000); % effectuer une commande sans en afficher le résultat
>> K=[4:0.1:8]
>> K=[8:0.1:4] % ça pose un problème
>> K=linspace(4,8,41) % attention au nombre de pas...
>> K=linspace(8,4,41)
>> L=eye(3)
```

Au lieu de travailler dans la Command Window, il est bien plus utile de faire un script de toutes les commandes que l'on souhaite executer: ainsi on peut sauvegarder (dans un fichier avec extension ".m") et modifier le script au besoin. Cela se fait facilement dans l'Editor intégré, qui permet aussi de lancer le script avec le bouton Run. Copier le script suivant dans un fichier `exo1.m` pour calculer une factorisation LU avec la fonction `lu()` intégrée:

```
A=[1,2,3,4;6,7,8,9;11,12,13,14]
[L,U,P]=lu(A)
```

Noter le résultat:

$$\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \cdot \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \cdot \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}$$

Exercice 2. Parmis les structures de contrôle et boucles il y a:

```
if (condition)          for i=1:n              while (condition)
% instructions         % instructions         % instructions
else                    endfor                 endwhile
% instructions
endif
```

Les conditions et opérateurs logiques usuelles sont:

```
(i==j) % i égal à j
(i<j)  % i strictement plus petit que j
(i>j)  % i strictement plus grand que j
P & Q  % P et Q
P | Q  % P ou Q
~ P    % non P
```

On peut aussi utiliser `~=`, `>=` et `<=`.

Ecrire un script `exo2.m` pour calculer la somme $\sum_{i=0}^{100} \frac{3^i - 2i}{i+2}$.

La somme vaut _____ .

Exercice 3. (Bonus:) On sait que la suite $u_n = \sum_{i=0}^{n} x^i = 1 + x + x^2 + ... + x^n$ converge vers $(1-x)^{-1}$ pour $-1 < x < 1$. Autrement dit, pour tout $-1 < x < 1$ et tout $\varepsilon > 0$ on peut trouver une (suffisamment grande) valeur pour n telle que la valeur absolue de la différence entre u_n et $(1-x)^{-1}$ est inférieur à ε. Trouver la plus petite valeur de n telle que $|u_n - (1-x)^{-1}| < 0.01$ pour $x = 0.79453$ (pour la valeur absolue d'un nombre on pourra utiliser la commande `abs()`), et calculer le nombre u_n correspondant. Sauvegarder les commandes dans un script `exo3.m`.

La valeur cherchée est $n =$ _____ ; et dans ce cas $u_n =$ _____ .

Exercice 4. Le logiciel Octave permet la création de nouvelles fonctions (en outre des fonctions déjà définies comme `det()`, `rank()`, `sin()`, etc.). Pour cela, on crée (dans Editor) un fichier, toujours avec extension `.m`, *dont le nom est égal au nom de la fonction que l'on souhaite créer.* Par exemple, pour créer la fonction "tangent de x", on sauvegarde le contenu suivant sous `tangent.m`:

```
function y=tangent(x)
y=sin(x)/cos(x);
end
```

Ainsi, dans le Command Window et dans les scripts on peut s'en servir en écrivant:

```
>> x=pi;
>> tangent(x)
```

On notera que $\tan(\pi)$ n'est pas tout à fait 0: c'est dû aux erreurs d'approximation numérique commises par le logiciel. On peut changer le format d'affichage des nombres pour "cacher" ces erreurs:

```
>> format bank % afficher deux décimaux
>> x=pi;
>> tangent(x)
>> format short % afficher 4 décimaux et puissance de 10 (affichage par défaut)
```

Une fonction peut avoir plusieurs entrées et plusieurs sorties, qui peuvent être des matrices:

```
function [y1,y2,...]=ma_fonction(x1,x2,...)
% instructions
end
```

Une fonction peut en appeler d'autres; il suffit que toutes les fonctions (tous les fichiers `.m`) se trouvent dans le même répertoire de travail.

Ecrire une fonction `H=matricehilbert(n)` pour construire la *matrice de Hilbert d'ordre* $n \in \mathbb{N}_0$:

$$H_n = \begin{pmatrix} 1 & 1/2 & 1/3 & ... & 1/n \\ 1/2 & 1/3 & 1/4 & ... & \vdots \\ 1/3 & 1/4 & 1/5 & & \\ \vdots & & & \ddots & \\ 1/n & ... & & & 1/(2n-1) \end{pmatrix}$$

puis calculer son déterminant et son rang avec les fonctions intégrées `det()` et `rank()` dans les cas suivants:

dét(H_{10}) = _____ et rang(H_{20}) = _____

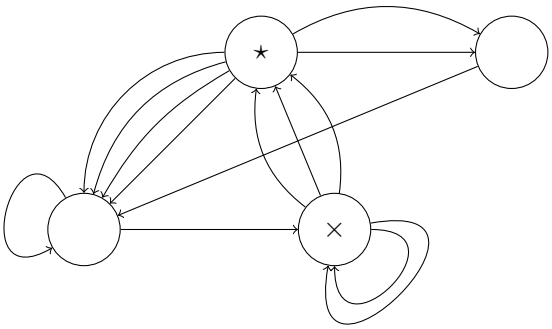
2. Multigraphe dirigé et matrice d'adjacence

Exercice 5. Un *multigraphe dirigé* est la donnée d'un nombre fini de sommets reliés par un nombre fini de flèches. Pour des raisons pratiques, on écarte le graphe avec 0 sommets. Si on numérote les sommets du graphe de 1 à n, alors la *matrice d'adjacence* $A = (a_{ij})_{ij} \in \mathbb{R}^{n\times n}$ du graphe est définie par

$$a_{ij} = \text{ nombre de flèches du sommet } j \text{ au sommet } i.$$

Au CM/TD nous avons vu que l'élément en position (i, j) dans la matrice A^k (pour $k \in \mathbb{N}$) donne le nombre de *chemins* de longueur k du sommet j au sommet i.

Soit maintenant le multigraphe dirigé



La matrice d'adjacence est

$$A = \begin{pmatrix} & \\ & \end{pmatrix}.$$

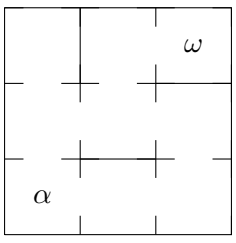
Faire un script `exo5.m` pour calculer:

Le nombre de chemins de longueur 7 du sommet '×' au sommet '⋆' est _____ .

Le nombre de chemins de longueur au plus 5 du sommet '×' au sommet '⋆' est _____ .

Le nombre de chemins de longueur entre 7 et 13 du sommet '×' au sommet '⋆' est _____ .

Exercice 6. Dans le labyrinthe suivant, partant de la case α, on se déplace toujours à une case voisine, et une fois qu'on est à la case ω, on y reste.



En faisant au plus 13 pas, le nombre de chemins de α à ω est: _____ .

Exercice 7. (Bonus:) Un multigraphe dirigé est dit *irréductible* (aussi *fortement connexe*) si, pour toute paire de sommets, il existe au moins un chemin de l'un à l'autre.

(a) Dessiner quelques exemples de graphes irréductibles et de graphes non-irréductibles.

(b) Montrer que, s'il existe un chemin entre deux sommets dans un graphe à n sommets, alors il existe toujours un chemin de longueur au plus $n-1$.

(c) Soit A la matrice d'adjacence d'un graphe donné. Montrer que ce graphe est irréductible si et seulement si tout élément de $A^0 + A^1 + ... + A^{n-1}$ est strictement positif. De manière générale, on dit qu'une matrice carrée A à éléments positifs est *irréductible* si tout élément de $A^0+A^1+...+A^{n-1}$ est *strictement* positif.

(d) Calculer $(I + A)^k$ pour k quelconque. (Indication: faire l'analogie avec $(1 + a)^k$ pour un nombre a; commencer par $k = 2$, $k = 3$, etc.) Observer que, pour tout $k > 0$ et $l > 0$, kA^l est à éléments strictement positifs si et seulement si A^l est à éléments strictement positifs. Conclure qu'une matrice A à éléments positifs est irréductible si et seulement si $(I + A)^{n-1}$ est une matrice à éléments *strictement* positifs.

(e) Ecrire une fonction `reponse=irreductible(A)` pour vérifier si une matrice d'adjacence est irréductible ou pas. (La variable `reponse` aura la valeur 1 si A est irréductible et 0 sinon.)

Exercice 8. (Bonus:) Un multigraphe dirigé est dit *primitif* s'il existe un nombre $k > 0$ tel que, pour toute paire de sommets, il existe un chemin de longueur k de l'un à l'autre.

(a) Dessiner quelques exemples de graphes primitifs et de graphes non-primitifs.

(b) Comment peut-on vérifier la primitivité d'un graphe par un calcul avec sa matrice d'adjacence?

(c) Formuler une définition mathématique pour la notion de *matrice primitive*. Montrer qu'une matrice primitive est toujours irréductible, mais que l'implication réciproque est fausse.

(d) Ecrire une fonction `reponse=primitive(A)` pour vérifier si une matrice d'adjacence est primitive ou pas. (La variable `reponse` aura la valeur 1 si A est primitive et 0 sinon.)

3. Interpolation et régression

Exercice 9. Avec Octave on peut faire des graphes de fonctions réelles:

```
>> x1=linspace(-10,10,200); % une liste d'abscisses
>> y1=sin(x1);              % le calcul des ordonnées
>> plot(x1,y1)              % le graphe
>> axis equal               % pour forcer un repère orthonormé
```

On peut noter l'importance de choisir "suffisamment" d'abscisses...

```
>> x2=linspace(-10,10,10);
>> y2=sin(x2);
>> plot(x2,y2)
>> axis equal
```

Pour marquer des points particuliers on peut utiliser:

```
>> x3=[-10, -8, -3, 6, 8];  % abscisses des points
>> y3=[0.2, 0.5, 1, -1, 0]; % ordonnées des points
```

```
>> plot(x3,y3,'o')          % les couples (x3,y3) seront indiqués par un petit rond
>> axis equal
```

Et il y a moyen de mettre plusieurs graphes dans un même repère:

```
>> plot(x1,y1,x2,y2,x3,y3,'o')
>> axis equal
```

On peut aussi faire des courbes paramétrées:

```
>> t=[0:0.01:2*pi];                        % paramètre
>> x4=cos(t)+0.5*cos(7*t)+(1/3)*sin(17*t); % abscisses des points
>> y4=sin(t)+0.5*sin(7*t)+(1/3)*cos(17*t); % ordonnées des points
>> plot(x4,y4)
>> axis equal
```

Un polynôme comme par exemple $P(x) = 3x^4+5x^2-2x+4$ est encodé par la matrice ligne de ses coefficients, le premier élément étant le coefficient du terme de plus haut degré:

```
>> P=[3, 0, 5, -2, 4];
```

Pour calculer la valeur de ce polynôme en $a = 8$ on fait:

```
>> a=8;
>> b=polyval(P,a)
```

Etant donné une matrice ligne d'abscisses, on peut également calculer une matrice ligne de valeurs du polynôme:

```
>> x5=linspace(-3,3,60);
>> y5=polyval(P,x5);
>> plot(x5,y5)
>> axis equal
```

Finalement, on peut mettre plusieurs repères dans une même figure: la commande `subplot(m,n,k)` crée un repère à la k-ième place dans une figure contenant $m \times n$ repères. Copier (sous le nom `exo9.m`) toutes les définitions nécessaires données ci-dessus, puis ajouter les commandes ci-dessous, et tester le script obtenu:

```
figure
subplot(2,2,1)
plot(x1,y1)
axis equal
subplot(2,2,2)
plot(x2,y2)
axis equal
subplot(2,2,3)
plot(x1,y1,x2,y2,x3,y3,'o')
axis equal
subplot(2,2,4)
plot(x4,y4)
axis equal
```

Exercice 10. Nous avons vu au CM/TD que, étant donné $n+1$ points de $\mathbb{R}^2$, disons (x_0, y_0), ..., (x_n, y_n), avec les x_i tous distincts, il existe un unique *polynôme d'interpolation* de degré n, $f(x) = a_n x^n + ... + a_1 x + a_0$, dont le graphe contient tous les (x_i, y_i). Pour déterminer les coefficients de ce polynôme, il faudra résoudre un système linéaire $AX = B$; on pourra le faire avec la commande `X=A\B`. (Attention: c'est `A` "sous" `B`!) Créer un script (nommé `exo10.m`) pour afficher, dans un seul repère, les points

$$(-3,4) \quad (-2,1) \quad (-1,2) \quad (0,0) \quad (1,2) \quad (2,-1) \quad (3,7)$$

ainsi que le graphe de la fonction polynomiale d'interpolation qu'ils déterminent (et qu'il faudra donc calculer).

On a $f(x) =$ _____ $x^6 +$ _____ $x^5 +$ _____ $x^4 +$ _____

_____ $x^3 +$ _____ $x^2 +$ _____ $x +$ _____ .

Sa valeur en $x = 1.7$ est $y =$ _____ .

Exercice 11. Nous avons vu au CM/TD comment on fait de la *régression polynomiale*: on peut déterminer un polynôme de degré k, soit $f(x) = a_k x^x + ... + a_1 x + a_0$, dont le graphe passe "le plus près possible" de n points (x_1, y_1), ..., (x_n, y_n) donnés de $\mathbb{R}^2$. Etant donné des matrices lignes `x=[x1, ... ,xn]` et `y=[y1, ... , yn]`, créer une fonction `P=regresspoly(x,y,k)` qui calcule les coefficients `P=[a_k,...,a_0]` du polynôme de régression $f(x) = a_k x^k + ... + a_1 x + a_0$ ainsi déterminé. Utiliser ensuite cette fonction dans le script suivant (à sauvegarder sous le nom `exo11.m`):

```
x=[1,2,3,4,5,6,7,8];
y=100*sin(x)+(1/10)*exp(x/100);
P=regresspoly(x,y,4);
x1=linspace(0,9,100);
y1=polyval(P,x1);
plot(x,y,'o',x1,y1)
```

Le polynôme en question est $f(x) =$ _____

et sa valeur en $x = 7$ est _____ .

Exercice 12. Par le lien

`http://www-lmpa.univ-littoral.fr/~stubbe/L2Info/DataLifeExpectancy.txt`

on trouvera les données officielles d'Eurostat concernant l'espérance de vie de nouveaux né(e)s en France, en Norvège et dans l'Union Européenne. Développer une stratégie et faire les calculs (dans un script `exo12.m`) pour estimer:

L'espérence de vie d'une fille née en Norvège en 2030: _____ .

L'espérence de vie d'un garçon né en France en 2030: _____ .

L'espérence de vie d'une fille née en Europe en 2030: _____ .

4. Gram-Schmidt, QR

Exercice 13. Au CM/TD nous avons défini la *projection orthogonale* de $P \in \mathbb{R}^{n\times 1}$ sur (la droite déterminée par) $A \in \mathbb{R}^{n\times 1}$ ainsi que la *normalisation* de $P \in \mathbb{R}^{n\times 1}$. Ecrire deux fonctions, `Pprime=proj(A,P)` et `Pprime=normal(P)`, pour effectuer ces calculs.

La projection orthogonale de $P = \begin{pmatrix} 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}$ sur les colonnes de $A = \begin{pmatrix} 5 & -3 & 3 & -3 \\ 3 & -2 & -2 & 0 \\ 8 & 7 & -3 & 1 \\ 5 & 3 & -2 & 1 \\ 2 & 0 & -1 & 3 \end{pmatrix}$ est $\begin{pmatrix} \_\_\_ \\ \_\_\_ \\ \_\_\_ \\ \_\_\_ \\ \_\_\_ \end{pmatrix}$.

Exercice 14. Le *procédé classique de Gram-Schmidt (Classical Gram-Schmidt, CGS)* permet de transformer une matrice $A \in \mathbb{R}^{n\times k}$ de rang k en une matrice $Q \in \mathbb{R}^{n\times k}$ ayant la même image que A mais dont les colonnes forment une suite orthonormale. Comme nous avons vu au CM/TD, si on note $A_1, ..., A_k$ les colonnes de A, on calcule d'abord

- $B_1 = A_1$,
- pour $i \geq 2$: $B_i = A_i - \sum_{k=1}^{i-1} \texttt{proj}(B_k, A_i)$,

et puis on pose

- pour tout i: $Q_i = \texttt{normal}(B_i)$.

Ecrire une fonction `Q=classGS(A)` qui implémente ce procédé, et tester votre fonction avec les commandes suivantes:

```
>> A=[1,2;3,4;5,6]
>> classGS(A)
>> B=randi([-10 10],4,8)
>> classGS(B)
```

Exercice 15. Soit la matrice

$$A_\varepsilon = \begin{pmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{pmatrix}$$

où $\varepsilon > 0$; le procédé de Gram-Schmidt s'applique donc à A_ε. Ecrire un script, sauvegardé dans un fichier `exo15.m`, dans lequel, pour chaque $\varepsilon \in \{0.1, 0.01, ..., 10^{-10}\}$, on calcule avec la fonction `classGS` la matrice Q correspondant à A_ε, puis on affiche le produit scalaire de la deuxième et la troisième colonne de Q.

ε	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
$Q_2^t Q_3$						

Constat?

Exercice 16. Le procédé classique de Gram-Schmidt pose donc quelques problèmes quant à son implémentation sur ordinateur, essentiellement dû aux erreurs d'arrondi; on dit que le procédé n'est pas *numériquement stable*. Le *procédé modifié de Gram-Schmidt (Modified Gram-Schmidt, MGS)* est une variante du procédé classique de Gram-Schmidt, qui résout ce problème de stabilité. Explicitement, pour les colonnes $A_1, ..., A_k$ d'une matrice $A \in \mathbb{R}^{n\times k}$ de rang k:

- on pose $B_1 = A_1$,
- pour $i \geq 2$ on a l'itération suivante pour calculer B_i:

$$\begin{aligned}
&B_i^{(1)} = A_i \\
&B_i^{(2)} = B_i^{(1)} - \texttt{proj}(B_1, B_i^{(1)}) \\
&B_i^{(3)} = B_i^{(2)} - \texttt{proj}(B_2, B_i^{(2)}) \\
&\vdots \\
&B_i^{(j+1)} = B_i^{(j)} - \texttt{proj}(B_j, B_i^{(j)}) \\
&\vdots \\
&B_i := B_i^{(i)} = B_i^{(i-1)} - \texttt{proj}(B_{i-1}, B_i^{(i-1)})
\end{aligned}$$

et ensuite on normalise pour obtenir le résultat:

- pour $i \geq 1$: $Q_i = \texttt{normal}(B_i)$.

Ecrire une fonction `Q=modifGS(A)` pour implémenter cet algoritme, puis reprendre l'exercice précédent mais avec la fonction `modifGS`, dans un script appelé `exo16.m`.

ε	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
$Q_2^tQ_3$						

Exercice 17. Utiliser le MGS pour écire un fonction `[Q,R]=modifQR(A)` donnant la factorisation QR d'une matrice (sous les conditions habituelles). Calculer la factorisation QR de la matrice suivante à deux décimaux près (donc en `format bank`):

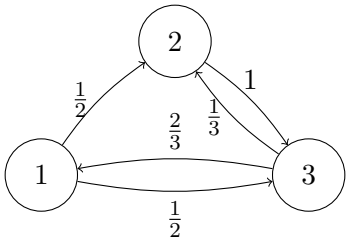
$$\begin{pmatrix} -8 & 4 & -4 & 7 \\ -8 & 4 & -3 & 6 \\ 8 & -7 & 4 & 2 \\ -6 & -10 & 0 & -7 \end{pmatrix} = \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

5. Matrice stochastique, vecteur stationnaire

Exercice 18. On appelle $X \in \mathbb{R}^n$ un *vecteur stochastique* si tous ses éléments sont positifs et leur somme vaut 1. Une *matrice stochastique* (aussi appelée *matrice de Markov*) est une matrice carrée dont chaque colonne est un vecteur stochastique. A toute matrice stochastique $M = (m_{ij})_{ij} \in \mathbb{R}^{n\times n}$ on peut associer un graphe pondéré (et *vice versa*): ce graphe a n sommets, et de tout sommet j à tout sommet i une flèche

"pondérée" par le poids m_{ij}. Un tel graphe visualise un *système probabiliste discret* (aussi appelé *chaîne de Markov*): chaque sommet est un état du système, chaque flèche indique la probabilité de transition d'un état à un autre.

On considère maintenant le jeu suivant: on choisit une de trois positions initiales, et ensuite le hasard décide quel mouvement on fait, selon les probabilités données dans le graphe suivant:



La matrice stochastique associée au graphe est $M = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}$.

Calculer:

$$M \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}, \quad M^2 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}, \quad M^3 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}.$$

Que représentent ces résultats?

Calculer:

$$M^{100} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}, \quad M^{100} \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}, \quad M^{100} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \\ \\ \end{pmatrix}.$$

Constat?

Calculer aussi:

$$M \cdot \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \quad \end{pmatrix} \qquad M^3 \cdot \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \quad \end{pmatrix} \qquad M^5 \cdot \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \quad \end{pmatrix}.$$

Constat?

Exercice 19. (Bonus:) Montrer les assertions suivantes.

(a) Si M est une matrice stochastique et S est une colonne stochastique, alors aussi la colonne $M \cdot S$ est stochastique (lorsque le produit matriciel est bien défini).

(b) Si M et N sont deux matrices stochastiques de même taille, alors pour tout $0 \leq \alpha \leq 1$ aussi $K = \alpha \cdot M + (1-\alpha) \cdot N$ est stochastique. On dit que K est une *combinaison convexe* de M et N.

(c) Pour tout $n > 0$, la matrice $n \times n$ dont chaque élément vaut n^{-1} est stochastique. On va la noter U_n dans la suite.

(d) Soit M une matrice stochastique carrée, de taille $n \times n$. Pour tout $0 < \alpha < 1$, la matrice $\alpha \cdot M + (1-\alpha) \cdot U_n$ est stochastique et à éléments strictement positifs; il suit donc que cette combinaison convexe est une matrice primitive.

Exercice 20. On dit qu'une colonne stochastique $S \in \mathbb{R}^{n \times 1}$ est un *vecteur stationnaire* d'une matrice stochastique $M \in \mathbb{R}^{n \times n}$ si $MS = S$; autrement dit, il s'agit d'un *vecteur propre stochastique pour la valeur propre* $\lambda = 1$. Le *Théorème de Perron-Frobenius*[1] assure que toute matrice stochastique irréductible $M \in \mathbb{R}^{n \times n}$ a un unique vecteur stationnaire $S = (s_1, ..., s_n)$. Ce vecteur S est donc la *seule* solution au système linéaire

$$\begin{cases} M \cdot \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix} &= \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix} \\ s_1 + ... + s_n &= 1 \end{cases}$$

qui s'écrite de manière équivalente comme

$$\left(\begin{array}{c} M - I_n \\ \hline 1 \quad \dots \quad 1 \end{array}\right) \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

Coder une fonction `S=vectstat(M)` pour effectuer ce calcul. Reprendre ensuite l'Exercice 18: calculer le vecteur stationnaire de la matrice stochastique, et comparer le résultat avec les calculs déjà faits.

[1] D'après les mathématiciens allemands Oskar Perron (1880-1975) et Ferdinand Georg Frobenius (1849-1917). Pour l'énoncé complet du théorème général, consulter une bonne référence. Ici nous n'avons besoin que du cas particulier des matrices stochastiques, où l'énoncé se simplifie considérablement.

Constat?

C'est encore le *Théorème de Perron-Frobenius* qui explique ce résultat: pour une matrice stochastique primitive M, on peut approcher son vecteur stationnaire S par

$$S \approx M^k \cdot U \quad \text{avec } U \text{ un distribution quelconque et } k \text{ "assez grand".}$$

Ce calcul s'appelle la *Méthode de la Puissance Itérée ('Power Method')*. Pour la plus grande efficacité on prend souvent la distribution uniforme pour U.

Exercice 21. Pour leurs vols européens, une compagnie aérienne propose deux formules: un billet moins cher mais une seule pièce de bagage à main est autorisée, puis un billet plus cher mais avec la possibilité de prendre jusque 35kg de bagage. Peu après le lancement de ces deux formules, on a constaté que 40% des voyageurs ayant pris la formule la moins chère en sont contents et reprennent un tel billet pour leur prochain voyage; mais 60% change donc de formule. D'autre part, parmi ceux qui ont pris la formule avec plus de bagage, 80% en sont contents alors que 20% changent de formule.

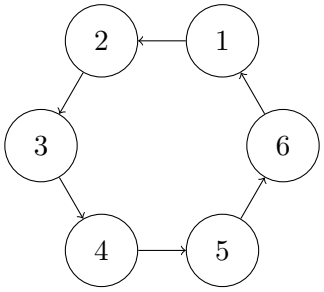
Dessiner le graphe pondéré mettant en situation ces données:

Donner la matrice stochastique $M = \begin{pmatrix} & \\ & \end{pmatrix}$.

Calculer le vecteur stationnaire $S = \texttt{vectstat(M)} = \begin{pmatrix} \\ \end{pmatrix}$.

Comment cette information peut-elle être utile pour la compagnie aérienne?

Exercice 22. (Bonus:) Soit le graphe suivant:



Si on accorde à chaque flèche le poids 1, alors la matrice stochastique est

$$M = \begin{pmatrix} & & \\ & & \\ & & \\ & & \\ & & \\ & & \end{pmatrix}.$$

Est-elle irréductible? primitive?

Son vecteur stationnaire est

$$S = \texttt{vectstat(M)} = \begin{pmatrix} \\ \\ \\ \\ \\ \\ \end{pmatrix}.$$

Si on tente d'approximer le vecteur stationnaire par la Méthode des Puissances Itérées, on observe en particulier que

$$M \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} = \begin{pmatrix} \\ \\ \\ \\ \\ \end{pmatrix}, \quad M^2 \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} = \begin{pmatrix} \\ \\ \\ \\ \\ \end{pmatrix}, \quad M^3 \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{pmatrix} = \begin{pmatrix} \\ \\ \\ \\ \\ \end{pmatrix}, \quad \text{etc.}$$

et par conséquent...

Exercice 23. Créer une fonction `M=stoch(A)` qui transforme une matrice $A \in \mathbb{R}^{n \times n}$ d'éléments positifs en une matrice stochastique en divisant chaque colonne par la somme de ses éléments; si une colonne de A est nulle, elle sera remplacée par la colonne $(1/n, ..., 1/n)$. Tester la fonction avec les commandes suivantes:

```
>> A=randi([1,99],10,10); % une matrice aléatoire 10x10 d'entiers entre 1 et 99
>> M=stoch(A);
>> k=randi(10);           % un entier aléatore entre 1 et 10
>> sum(M(:,k))
```

Si un multigraph dirigé est donné (p.e. celui de Exercice 5), et A est sa matrice d'adjacence, comment interpréter alors la matrice stochastique $M = \text{stoch}(A)$?

Répéter ensuite quelques fois les commandes suivantes:

```
>> A=randi([1,99],10,10)   % une matrice aléatoire 10x10 d'entiers entre 1 et 99
>> M=stoch(A)              % on la transforme en matrice stochastique
>> E=eig(M)                % calcul des valeurs propres de M
>> max(E)                  % afficher la plus grande valeur propre
```

Constat?

C'est encore le *Théorème de Perron-Frobenius* qui assure qu'une matrice stochastique irréductible M admet 1 comme valeur propre simple et dominante (en valeur absolue); et si M est primitive, alors la valeur propre 1 est même strictement dominante.

Exercice 24. *Qui est le meilleur de la classe?* Voici un tableau avec les résultats des examens (sur 20):

	Math	Phys	Chim	Info	Econ
Anne	14	12	15	11	9
Blaise	9	16	11	14	11
Christelle	17	19	15	16	11
Diego	13	9	12	12	15

Si on calcule les moyennes des notes, on obtient le classement suivant:

1. _____
2. _____
3. _____
4. _____

Si on fait un multigraphe dont les sommets sont les étudiants, et dans lequel on ajoute une flèche de x à y pour chaque examen que x a fait mieux que y, alors selon le vecteur stationnaire de la matrice stochastique de ce multigraphe, on obtient le classement suivant:

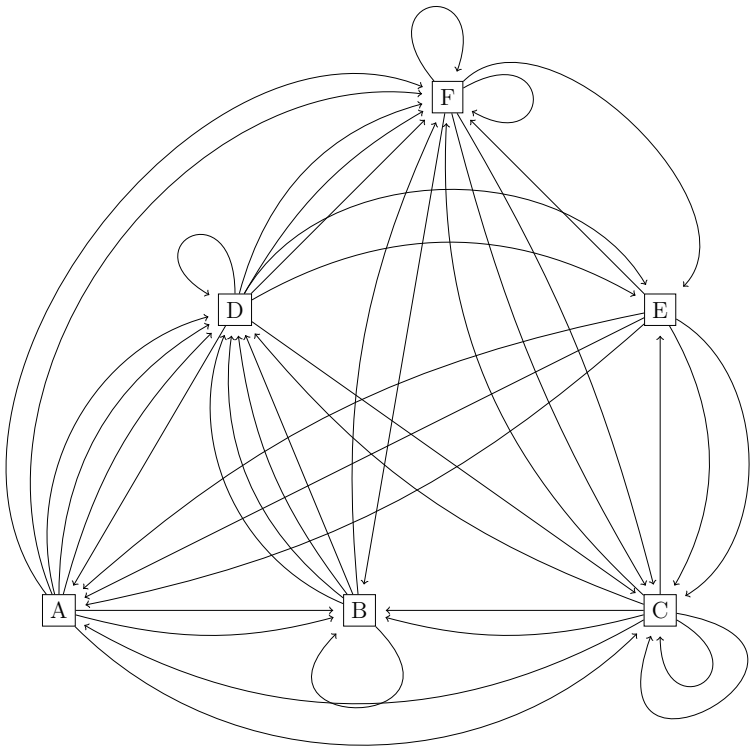
1. _____
2. _____
3. _____
4. _____

On peut aussi faire un multigraphe dont les sommets sont toujours les étudiants, mais on ajoute une flèche de x à y pour chaque point que x a plus que y. Selon le vecteur stationnaire de la matrice stochastique de ce deuxième multigraphe, on obtient le classement suivant:

1. _____
2. _____
3. _____
4. _____

6. Matrice de Google

Exercice 25. On considère le multigraphe suivant:



Numéroter ses sommets par ordre alphabétique et donner la matrice de multi-adjacence A, la matrice stochastique M, puis le vecteur stationnaire S de M:

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

$$M = \texttt{stoch(A)} = \begin{pmatrix} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{pmatrix} \qquad S = \texttt{vectstat(M)} = \begin{pmatrix} \\ \\ \\ \\ \\ \\ \end{pmatrix}$$

Désormais on interprète ce multigraphe comme un réseau de pages web liées entre elles. On s'imagine une personne qui visite une première page web (choisie au hasard), puis suit un des hyperliens de cette page vers

une autre page, etc. Si on répète ce procédé *ad infinitum*, alors la page la plus souvent visitée (donc "la page la plus importante") est donnée par le score le plus élevé dans le vecteur stationnaire du graphe.

Il y a, pourtant, un double problème avec ce modèle: (1) la matrice stochastique M n'admet peut-être pas un (unique) vecteur stationnaire, et (2) c'est un peu simple de penser que tout surfeur sur internet suit éternellement les hyperliens qui lui sont présentés. Une légère modification de la matrice stochastique du graphe résout ces deux problèmes—et c'est ça l'invention de Larry Page, cofondateur de Google en 1998.

On introduit ainsi la *matrice de Google* du graphe:

$$G = \alpha M + (1-\alpha)\begin{pmatrix} 1/n & \dots & 1/n \\ \vdots & & \vdots \\ 1/n & \dots & 1/n \end{pmatrix}$$

où M est la matrice stochastique, n est le nombre total de pages (ici $n = 6$), et $\alpha \in [0, 1]$ est un paramètre. D'un côté, ce paramètre α exprime la "fidélité" du surfeur: il y a une probabilité α que le visiteur suit les hyperliens proposés, et une probabilité $1 - \alpha$ qu'il choisira au hasard une nouvelle page de départ. De l'autre côté, cette nouvelle matrice stochastique est *primitive* pour tout $\alpha \neq 1$, et cela implique que l'on peut calculer son unique vecteur stationnaire efficacement par la Méthode des Puissances Itérées.

Coder une fonction `G=google(M,alpha)` pour effectuer ce calcul.

Pour $\alpha = 0.85$, calculer:

$G =$ `google(M,0.85)` $=$ $\begin{pmatrix} \\ \\ \\ \\ \\ \\ \end{pmatrix}$ et $S =$ `vectstat(G)` $=$ $\begin{pmatrix} \\ \\ \\ \\ \\ \\ \end{pmatrix}$

Selon ce modèle, quelle est donc le classement des pages web de "la plus importante" à "la moins importante"?

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____

C'est exactement avec un tel procédé que Google classe les pages web lors d'une recherche sur internet; cet algorithme est désormais connu comme le 'PageRank' algorithme. On estime que la matrice que Google utilise, est de taille $3 \cdot 10^{10} \times 3 \cdot 10^{10}$; pour le calcul de son vecteur stationnaire, Google utilise la Méthode de la Puissance Itérée, et on estime que Google fait entre 50 et 100 itérations. On pense que la valeur de α est autour de 0.85 (mais c'est un secret industriel). Le calcul prend plusieurs jours.

Exercice 26. Deux candidats à la Présidentielle, Bernard et Cathérine, s'affrontent lors d'un débat. Au lendemain du débat, une enquête montre que, parmis les gens qui comptaient auparavant voter pour Bernard, 85% confirme son choix alors que 7% va finalement voter pour Cathérine; le reste décide de s'abstenir. Parmis

les gens qui comptaient voter pour Cathérine, 23% change de candidat et 19% s'abstiendra, alors que le reste confirme son choix pour Cathérine. Finalement, de ceux qui voulaient s'abstenir, 40% va finalement quand-même voter pour Bernard et 17% pour Cathérine, et le reste ne votera donc pas.

Le graphe représentant cette situation est:

La matrice stochastique de ce graphe, et son vecteur stationnaire, sont donc

$$M = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \quad \text{et} \quad S = \begin{pmatrix} \\ \\ \\ \end{pmatrix}.$$

Mais l'institut de sondage estime que seul 92% des électeurs intérrogés tiendra parole. Pour tenir compte de cette volatilité, l'institut décide d'utiliser une matrice de google.

La matrice utilisée sera donc $G =$ `google(M,alpha)` avec $\alpha =$ _____.

Ainsi on trouve la matrice et son vecteur stationnaire suivant:

$$G = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \quad \text{et} \quad S' = \begin{pmatrix} \\ \\ \\ \end{pmatrix}.$$

Formuler à l'aide de ces calculs une prévision du résultat de l'élection:

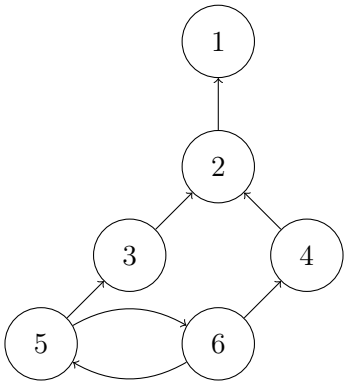
_____

_____

_____

_____

Exercice 27. (Bonus:) Soit le graphe



Selon le vecteur stationnaire de la matrice de Google du graphe, prenant $\alpha = 0.6$, le classement des sommets de ce graphe est le suivant:

Selon le vecteur stationnaire de la matrice de Google du graphe, prenant $\alpha = 0.7$, le classement des sommets de ce graphe est le suivant:

Morale?

Exercice 28. (Bonus:) Montrer que, pour toute matrice stochastique M et tout $0 < \alpha < 1$, la matrice de Google $G = \alpha M + (1 - \alpha)U_n$ est stochastique et primitive; cela implique que l'on peut calculer son vecteur stationnaire efficacement par la Méthode des Puissances Itérées. (On rapelle que U_n est la notation pour la matrice $n \times n$ dont chaque élément vaut n^{-1}.)

7. Valeurs singulières, traitement d'image

Exercice 29. Nous avons vu au CM/TD que, si $A = USV^t$ est la décomposition par ses valeurs singulières (notées $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_k > 0$) d'une matrice $A \in \mathbb{R}^{m \times n}$ de rang r, alors pour tout $k \leq r$ la matrice $A_k = \Sigma_{i=1}^{k} \sigma_i U_i V_i^t$ est la meilleure approximation de A par une matrice de rang k (où V_i, resp. U_i, est la i-ième colonne de V, resp. U). Pour calculer la décomposition par valeurs singulières de A, on pourra utiliser la fonction `[U,S,V]=svd(A)`. Ecrire une fonction `X=SVDapprox(A,k)` pour calculer la meilleure approximation de A par une matrice de rang k. Soit H la matrice de Hilbert 4×4 (cf. Exercice 4). Ecrire un script `exo29.m` pour calculer (H et) toutes ses approximations de rang 1 à rang$(H) = 4$ (en `format bank`):

$$H_1 = \begin{pmatrix} & \\ & \end{pmatrix}, \qquad H_2 = \begin{pmatrix} & \\ & \end{pmatrix},$$

$$H_3 = \begin{pmatrix} & \\ & \end{pmatrix}, \qquad H = H_4 = \begin{pmatrix} & \\ & \end{pmatrix}.$$

Exercice 30. Pour Octave, une image en niveaux de gris est une matrice dont chaque élément représente une valeur de luminosité d'un pixel (entre 0 = noir et 255 = blanc). On peut donc facilement faire du "calcul matriciel" pour modifier une telle image. Expérimenter dans la Command Window avec les instructions suivantes:

```
>> I=imread('http://www-lmpa.univ-littoral.fr/~stubbe/L2info/image2.jpg');
>> size(I)        % taille de la matrice
>> I              % montrer la matrice contenant les valeurs grises des pixels
>> imshow(I)      % montrer l'image
>> [L,U,P]=lu(I); % du calcul matriciel...
>> U
>> imshow(U)
>> J=I';
>> imshow(J)
>> K=imread('http://www-lmpa.univ-littoral.fr/~stubbe/L2info/image3.jpg');
>> imshow(K)
>> J=0.5*I+0.5*K;
>> imshow(J)
```

Exercice 31. Une image en `truecolor` de $m \times n$ pixel est la superposition de trois matrices, une pour la composante rouge, une pour la composante verte, une pour la composante bleue. L'élément à la position (i, j) d'une telle matrice est un des 2^8 nombres entre 0 et 255, et encode l'intensité de rouge, verte ou bleu du pixel correspondant; ainsi on peut donc former 2^{24} (soit environ 16 million) couleurs différentes. Dans la Command Window, essayer les commandes ci-dessous:

```
>> I=imread('http://www-lmpa.univ-littoral.fr/~stubbe/L2info/image1.jpg');
>> size(I)       % Que constate-t-on?
>> figure(1)
>> imshow(I);
>> R=I(:,:,1);   % Red
>> G=I(:,:,2);   % Green
>> B=I(:,:,3);   % Blue
>> figure(2)
>> imshow(R);
```

```
>> figure(3)
>> imshow(G);
>> figure(4)
>> imshow(B);
```

Puisqu'une image couleur est une superpostion de trois matrices, on peut la modifier avec les techniques de l'algèbre linéaire. Mais attention au format des éléments des matrices! Pour une matrice provenant d'une image, les éléments sont des entiers entre 0 et 255: c'est le format `uint8` (*unsigned 8 bit integer*). Cependant, pour certaines fonctions – et notamment le `svd` – Octave exige que les éléments soient en format `double` (*double-precision floating-point number*). Copier sous le nom `exo31.m`, puis exécuter, le script suivant:

```
I=imread('http://www-lmpa.univ-littoral.fr/~stubbe/L2info/image1.jpg');
                        % L'image-matrice ...
I=double(I);            % ... dont on change le format des éléments.
R=I(:,:,1);             % On sépare la composante rouge ...
R=SVDapprox(R,20);      % ... pour appliquer l'approximation par SVD.
G=I(:,:,2);             % ... composante 'verte' ...
G=SVDapprox(G,20);
B=I(:,:,3);             % ... composante 'bleue' ...
B=SVDapprox(B,20);
J=zeros(size(I));       % On recompose une matrice-image ...
J(:,:,1)=R;
J(:,:,2)=G;
J(:,:,3)=B;
J=uint8(J);              % ... et on remet le bon format des éléments.
imshow(J)                % Affichage du résultat.
```

Exercice 32. On souhaite maintenant comparer, dans une même figure, les approximations de rang 5, de rang 10, de rang 20 et de rang 50 de l'image `image1.jpg`. Pour cela, transformer d'abord la partie pertinante du script de l'exercice précédent en une fonction `J=imageapprox(I,k)` pour calculer l'image-matrice approchée J en fonction de l'image-matrice I donnée et le rang k souhaité. Ecrire ensuite un script (que l'on sauvegardera comme `exo32.m`) pour terminer cet exercice.