Devoir 1. Décomposition QR par la méthode de Householder.

NOM :
PRÉNOM :

Lancer `xmaple` *et charger la bibliothèque linalg* : `with(linalg);`
Si vous êtes perdu, utiliser l'aide de maple : ? suivi du nom de la commande.
Écriver vos réponses dans les encadrés de cette feuille.

Théorème : Soit A une matrice réelle de taille $n \times n$, inversible. Alors il existe une matrice orthogonale Q, *i.e.* $Q^TQ = {}^TQQ = I$, et une matrice triangulaire supérieure R telles que $A = QR$.

On cherche à écrire un algorithme donnant la décomposition QR d'une matrice inversible A, par la méthode de Householder. Commençons par décrire la méthode de Householder.

Définition : Soit $u \in \mathbb{R}^n$. On appelle matrice de Householder, la matrice H_u de taille $n \times n$ donnée par

$$H_u = I - 2\frac{u^Tu}{||u||^2}, \quad \text{si } u \text{ est non-nul,}$$

et $H_u = I$ si u est le vecteur nul, avec ${}^Tu$ le transposé de u et $||u||^2 = {}^Tuu$ la norme de u.

On remarque que H_u est une matrice de réflexion par rapport au vecteur u, et en particulier est une matrice orthogonale. On a alors la propositon suivante.

Proposition : Pour tout vecteur $v \in \mathbb{R}^n$, il existe $u \in \mathbb{R}^n$ tel que $H_uv = ||v||e_1$, où $e_1 = (1, 0, \dots, 0) \in \mathbb{R}^n$.

En effet, il suffit de prendre $u = 0$ si v est colinéaire à e_1, et $u = v - ||v||e_1$ sinon.

L'idée de la méthode de Householder est de multiplier A par des matrices de Householder pour faire apparaître une matrice triangulaire. Soit a_1 le premier vecteur colonne de A, et u_1 le vecteur de Householder associé de la proposition précédente. En notant $H_1 = H_{u_1}$, on a

$$A^{(2)} = H_1A = \begin{pmatrix} ||a_1|| & * \cdots * \\ 0_{\mathbb{R}^{n-1}} & B \end{pmatrix},$$

où B est une matrice de taille $n-1 \times n-1$. On répète alors l'opération précédente sur la matrice B. Soit $b_1 \in \mathbb{R}^{n-1}$ la premier vecteur colonne de B, et u_2 le vecteur de Householder associé. On pose

$$H_2 = \begin{pmatrix} 1 & 0 \\ 0 & H_{u_2} \end{pmatrix},$$

et on a

$$A^{(3)} = H_2A^{(2)} = H_2H_1A = \begin{pmatrix} ||a_1|| & * & * \\ 0 & ||b_1|| & * \\ 0 & 0_{\mathbb{R}^{n-2}} & C \end{pmatrix},$$

où C est une matrice de taille $n-2 \times n-2$. On itère alors l'opération jusqu'à obtenir une matrice triangulaire supérieure $A^{(n)}$ donnée par

$$H_{n-1} \dots H_1A = A^{(n)},$$

avec H_k de la forme $\begin{pmatrix} I_{k-1} & 0 \\ 0 & H_{u_k} \end{pmatrix}$. La décomposition QR de A est alors donnée par $Q = H_1 \dots H_n$, et $R = A^{(n)}$.

Pour décrire cette méthode, on va écrire à l'aide de maple des algorithmes sous la forme de procédure du type

```
algo:=proc(variables)
...
end proc;
```

On rappelle que pour définir des variables locales à l'intérieur d'une procédure, on utilise la commande `local`.

a) Définir la matrice identité $n \times n$. Pour cela, utiliser les commandes `diag` et `seq`.
Exemple :
`diag(1,2,3);` renvoie la matrice

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix},$$

et `seq(i,i=1..4);` renvoie la séquence $1, 2, 3, 4$.

```
identite:= n->
```

b) Tester l'égalité de deux vecteurs à l'aide de la commande
`equal(u1,u2);`
où u_1 et u_2 sont deux vecteurs définis par

```
u1:=vector([3,-2,1,6]);
u2:=vector([1,seq(0,i=1..3)]);
```

c) Définir le vecteur de $\mathbb{R}^n$, qu'on notera `e1`, égal à $(1, 0, \ldots, 0)$ à l'aide des commandes `vector` et `seq`.

```
e1:= n->
```

d) Tester la commande `sum` sur par exemple `sum(i^2,i=1..9)`. Écrire une fonction définissant la norme d'un vecteur. On utilisera la commande `vectdim(x)` donnant la dimension du vecteur `x` et `x[i]` pour extraire la i^{e} coordonnée de `x`.

```
norme:=x->
```

e) Boucle `if...then...else`. Pour effectuer une certaine opération lorsque on a une certaine condition, on utilise une boucle `if`, de la forme

```
if condition then
opération1 else
opération2
end if;
```

c'est-à-dire que si la condition est vraie, opération1 sera effectuée sinon opération2 le sera.
Par exemple, tester la procédure

```
carre:=proc(x)
local y;
y:=copy(x);
if y>25 then y:=y^2; else
y:=sqrt(y);
end if;
end proc;
```

sur les valeurs $5, 12, 34$. Interpréter (en français) la procédure précédente.

f) On va maintenant construire le vecteur u et la matrice de Houselholder H_u à l'aide d'une procédure. Soit x un vecteur de $\mathbb{R}^n$.
- si x est colinéaire à $e_1 = (1, 0, \ldots, 0)$, *i.e.* $x = ||x||e_1$, prendre u le vecteur nul, et H_u la matrice identité $n \times n$,
- si x n'est pas colinéaire à e_1, prendre $u = x - ||x|| \cdot e_1$, et $H_u = I - 2\frac{u \cdot {}^t u}{||u||^2}$.

On va donc pour cela utiliser une boucle `if...then...else`.

Écrire alors votre procédure donnant l'algorithme suivant dans l'encadré ci-dessous :

```
matriceHouse=procédure(vecteur x)
variables locales : n, ee, u, H
n=dimension de x
ee=(1,0,...0) de dimension n
Initialisation de u : u=vecteur nul(n)
Initialisation de H : H=matrice(n,n)
Si x=||x||ee alors
u=vecteur nul(n) et H=matrice identité(n,n)
sinon
u=x-||x||ee et H=matrice identité-2(u*transpose(u))/||u||^2
Fin de Si
Renvoyer u,H
Fin de la procédure
```

On utilisera les commandes `vectdim(x)` donnant la dimension du vecteur `x`, `vector(n,0)` donnant le vecteur nul de $\mathbb{R}^n$, `matrix(n,n)` pour initialiser une matrice quelconque. On rappelle que la multiplication des matrices (et aussi des vecteurs) est donnée par la commande `&*` et que pour effectuer des opérations sur les matrices (addition, multiplication,...) on utilise `evalm`.

```
matriceHouse:=proc(x)
local u,ee,n,H;









RETURN(evalm(u),evalm(H));
end proc;
```

g) Boucle `for...do`. Pour répéter une opération un nombre fixé de fois, on utilise une boucle `for...do` de la forme

```
for i from début to fin do
...
end do;
```

Par exemple, tester le programme suivant sur différentes valeurs.

```
repetition:=proc(t)
local y,s,i;
s:=NULL;
for i from 1 to t do
  y:=i^2;
  s:=s,y;
end do;
s;
end proc;
```

Avec quelle fonction de Maple peut-on obtenir le même résultat ? L'écrire ci-dessous.

h) Définir, à l'aide de la commande `matrix`, les matrices

$$U = \begin{pmatrix} 3 & 5 & -1 & 3 \\ 1 & 1 & -2 & 3 \\ 2 & 3 & 1 & 1 \\ -2 & 2 & 3 & 2 \end{pmatrix}, \quad V = \begin{pmatrix} 2 & -17 \\ -6 & 56 \end{pmatrix}.$$

Effectuer la commande `copyinto(V,U,2,2)` suivi de `evalm(A)`. Qu'effectue la commande `copyinto` ?

Utiliser l'aide de maple des commandes `submatrix` et `subvector` pour extraire la sous-matrice 3×3 correspondant au coin inférieur droit de U, et le sous-vecteur composé des coordonnées des lignes 2 à 4 de la 2^{e} colonne de U. Écrire les commandes utilisées ci-dessous.

i) Écrire votre procédure dans l'encadré ci-dessous donnant l'algorithme de la décomposition QR via la méthode de Householder.

```
House=procédure(matrice A)
variables locales : Q, R, n, i, u, H, K, RR, KK
n=dimension de A
Initialiser R : R=copie de A
Initialiser Q : Q=matrice identité(n,n)
Initialiser KK : KK=matrice identité(n,n)
Pour i allant de 1 à n-1 faire
  H=matrice de Householder du sous-vecteur des coordonnées de i à n
    du i-ième vecteur colonne de R
  K=copier H dans la matrice identité(n,n) à la position (i,i)
  RR=multiplier H et la sous-matrice du coin inférieur droit de R
     à partir de la coordonnée (i,i)
  R=copier RR dans R à la position (i,i)
  KK=multiplier K et KK
Fin de faire
Renvoyer transposée de KK, R
Fin de la procédure
```

On testera ensuite la procédure sur la matrice

$$A = \begin{pmatrix} 12 & -51 & 4 \\ 6 & 167 & -68 \\ -4 & 24 & -41 \end{pmatrix}.$$

(On pourra d'ailleurs vérifier la procédure en effectuant le produit QR.)

```
House:=proc(A)
local Q,R,n,i,u,H,K,RR,KK;









RETURN(evalm(transpose(KK)),evalm(R));
end proc;
```

Donner alors la décomposition QR de A.