DiagonalisationSimultanee

```
A=1/4*matrix(QQ,[[-1,1,-4,4,-3],[1,-1,4,-4,3],[10,-2,12,-
8,6],[7,1,4,0,3],[6,-6,8,-8,10]])
```

```
A
```

```
[-1/4  1/4   -1    1 -3/4]
[ 1/4 -1/4    1   -1  3/4]
[ 5/2 -1/2    3   -2  3/2]
[ 7/4  1/4    1    0  3/4]
[ 3/2 -3/2    2   -2  5/2]
```

```
B=1/4*matrix(QQ,[[12,-4,4,-4,2],[-4,12,-4,4,-2],[17,-25,28,-
24,13],[17,-25,20,-16,13],[-8,8,-8,8,4]])
```

```
B
```

```
[    3    -1     1    -1   1/2]
[   -1     3    -1     1  -1/2]
[ 17/4 -25/4     7    -6  13/4]
[ 17/4 -25/4     5    -4  13/4]
[   -2     2    -2     2     1]
```

```
A*B-B*A
```

```
[0 0 0 0 0]
[0 0 0 0 0]
[0 0 0 0 0]
[0 0 0 0 0]
[0 0 0 0 0]
```

```
#on peut utiliser eigenspaces_right(), ou bien
A.eigenmatrix_right()
```

```
DiagA=A.eigenspaces_right();DiagA
```

```
[
(2, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[ 1 -1 -2 -1 -2]),
(0, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[ 1  1 -2 -2  0]),
(1, Vector space of degree 5 and dimension 3 over Rational Fi
User basis matrix:
[   1   -1    0    0   -2]
[   0    0    1    0 -4/3]
[   0    0    0    1  4/3])
```

```
]
```

```
DiagB=B.eigenspaces_right();DiagB
```

```
[
(3, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[ 1 -1  2  2 -2]),
(1, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[0 0 1 1 0]),
(2, Vector space of degree 5 and dimension 3 over Rational Fi
User basis matrix:
[   1    0    0  9/2    7]
[   0    1    0 -1/2    1]
[   0    0    1    3    4])
]
```

```
DA=A.eigenmatrix_right()
```

```
DB=B.eigenmatrix_right()
```

```
DA
```

```
(
[2 0 0 0 0]  [   1    1    1    0    0]
[0 0 0 0 0]  [  -1    1   -1    0    0]
[0 0 1 0 0]  [  -2   -2    0    1    0]
[0 0 0 1 0]  [  -1   -2    0    0    1]
[0 0 0 0 1], [  -2    0   -2 -4/3  4/3]
)
```

```
DB
```

```
(
[3 0 0 0 0]  [   1    0    1    0    0]
[0 1 0 0 0]  [  -1    0    0    1    0]
[0 0 2 0 0]  [   2    1    0    0    1]
[0 0 0 2 0]  [   2    1  9/2 -1/2    3]
[0 0 0 0 2], [  -2    0    7    1    4]
)
```

```
# A et B sont diagonalisables et AB=BA, donc A et B sont
simultanément diagonalisables.
```

```
# On va faire les calculs de deux façons : d'abord en
utilisant DA et DB, puis en utilisant DiagA et DiagB.
```

```
DA
```

```
(
[2 0 0 0 0]  [   1    1    1    0    0]
```

```
[0 0 0 0 0]  [  -1    1   -1    0    0]
[0 0 1 0 0]  [  -2   -2    0    1    0]
[0 0 0 1 0]  [  -1   -2    0    0    1]
[0 0 0 0 1], [  -2    0   -2 -4/3  4/3]
)
```

```
u1=DA[1].column(0)
```

```
u1
```

```
    (1, -1, -2, -1, -2)
```

```
A*u1
```

```
    (2, -2, -4, -2, -4)
```

```
# u1 est vecteur propre pour A pour la valeur propre 2. Comme
Ker(A-2I) est de dimension 1 et comme cet espace est stable
par B, on en déduit que u1 est propre pour B. On le vérifie :
```

```
B*u1
```

```
    (2, -2, -4, -2, -4)
```

```
# Oui, u1 est dans Ker(B-2I).
```

```
u2=DA[1].column(1);u2 # u2 est vecteur propre pour la valeur
propre 0.
```

```
    (1, 1, -2, -2, 0)
```

```
# Tout comme u1, le vecteur u2 doit être propre pour B.
```

```
B*u2
```

```
    (2, 2, -4, -4, 0)
```

```
# u2 est bien dans Ker(B-2I).
```

```
# B a aussi deux espaces propres de dimension 1.
```

```
u3=DB[1].column(0);u3 #u3 engendre Ker(B-3I)
```

```
    (1, -1, 2, 2, -2)
```

```
B*u3
```

```
    (3, -3, 6, 6, -6)
```

```
u4=DB[1].column(1);u4 #u4 engendre Ker(B-I)
```

```
    (0, 0, 1, 1, 0)
```

```
B*u4
```

```
    (0, 0, 1, 1, 0)
```

```
# En raisonnant comme ci-dessus, on voit que u3 et u4 sont
propres pour A.
```

```
A*u3
```

```
(1, -1, 2, 2, -2)
```

```
A*u4
```

```
(0, 0, 1, 1, 0)
```

```
# Donc u3 et u4 sont dans Ker(A-I)
```

```
# Il ne reste plus qu'un vecteur propre pour A et B à trouver.
Comme on a deux vecteurs dans Ker(A-I) (u3 et u4) et deux
vecteurs dans Ker(B-2I) (u1 et u2), et comme ces espaces sont
de dimension 3, le vecteur manquant est dans Ker(A-I) et
Ker(B-I). On peut donc utiliser la commande "intersection".
Pour définir les espaces, on peut utiliser la commande "span",
ou bien utiliser le résultat de la commande
"eigenspaces_right", c'est-à-dire DiagA et DiagB calculés plus
haut.
```

```
EA1=span([DA[1].column(2),DA[1].column(3),DA[1].column(4)]);
EA1
```

```
Vector space of degree 5 and dimension 3 over Rational Field
Basis matrix:
[   1   -1    0    0   -2]
[   0    0    1    0 -4/3]
[   0    0    0    1  4/3]
```

```
EB2=span([DB[1].column(2),DB[1].column(3),DB[1].column(4)]);
EB2
```

```
Vector space of degree 5 and dimension 3 over Rational Field
Basis matrix:
[   1    0    0  9/2    7]
[   0    1    0 -1/2    1]
[   0    0    1    3    4]
```

```
Inter=EA1.intersection(EB2);Inter
```

```
Vector space of degree 5 and dimension 1 over Rational Field
Basis matrix:
[ 1 -1 -1  2  2]
```

```
u5=Inter.basis()[0];u5
```

```
(1, -1, -1, 2, 2)
```

```
#Construisons la matrice de passage.
```

```
P=matrix([u1,u2,u3,u4,u5]);P
```

```
[ 1 -1 -2 -1 -2]
[ 1  1 -2 -2  0]
[ 1 -1  2  2 -2]
[ 0  0  1  1  0]
[ 1 -1 -1  2  2]
```

```
P=transpose(P);P  #Il faut transposer P pour que les vecteurs
u1,u2,u3,u4,u5 soient les vecteurs colonnes de P.
```

```
[ 1  1  1  0  1]
[-1  1 -1  0 -1]
[-2 -2  2  1 -1]
[-1 -2  2  1  2]
[-2  0 -2  0  2]
```

```
P^(-1)*A*P
```

```
[2 0 0 0 0]
[0 0 0 0 0]
[0 0 1 0 0]
[0 0 0 1 0]
[0 0 0 0 1]
```

```
P^(-1)*B*P
```

```
[2 0 0 0 0]
[0 2 0 0 0]
[0 0 3 0 0]
[0 0 0 1 0]
[0 0 0 0 2]
```

```
# Hourra !
# On aurait pu tout faire à partir de DiagA et DiagB (donnés
par la commande eigenspaces_right())
```

```
DiagA=A.eigenspaces_right();DiagA
```

```
[
(2, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[ 1 -1 -2 -1 -2]),
(0, Vector space of degree 5 and dimension 1 over Rational Fi
User basis matrix:
[ 1  1 -2 -2  0]),
(1, Vector space of degree 5 and dimension 3 over Rational Fi
User basis matrix:
[   1   -1    0    0   -2]
[   0    0    1    0 -4/3]
[   0    0    0    1  4/3])
]
```

```
DiagB=B.eigenspaces_right();DiagB
```

```
[
(3, Vector space of degree 5 and dimension 1 over Rational Fi
```

```
    User basis matrix:
    [ 1 -1  2  2 -2]),
    (1, Vector space of degree 5 and dimension 1 over Rational Fi
    User basis matrix:
    [0 0 1 1 0]),
    (2, Vector space of degree 5 and dimension 3 over Rational Fi
    User basis matrix:
    [   1    0    0  9/2    7]
    [   0    1    0 -1/2    1]
    [   0    0    1    3    4])
    ]
```

```
v1=DiagA[0][1].basis()[0];v1
```

```
    (1, -1, -2, -1, -2)
```

```
v2=DiagA[1][1].basis()[0];v2
```

```
    (1, 1, -2, -2, 0)
```

```
v3=DiagB[0][1].basis()[0];v3
```

```
    (1, -1, 2, 2, -2)
```

```
v4=DiagB[1][1].basis()[0];v4
```

```
    (0, 0, 1, 1, 0)
```

```
EA1=DiagA[2][1];EA1
```

```
    Vector space of degree 5 and dimension 3 over Rational Field
    User basis matrix:
    [   1   -1    0    0   -2]
    [   0    0    1    0 -4/3]
    [   0    0    0    1  4/3]
```

```
EB2=DiagB[2][1];EB2
```

```
    Vector space of degree 5 and dimension 3 over Rational Field
    User basis matrix:
    [   1    0    0  9/2    7]
    [   0    1    0 -1/2    1]
    [   0    0    1    3    4]
```

```
Inter=EA1.intersection(EB2);Inter
```

```
    Vector space of degree 5 and dimension 1 over Rational Field
    Basis matrix:
    [ 1 -1 -1  2  2]
```

```
v5=Inter.basis()[0];v5
```

```
    (1, -1, -1, 2, 2)
```

```
P=matrix([v1,v2,v3,v4,v5]);P
```

```
    [ 1 -1 -2 -1 -2]
    [ 1  1 -2 -2  0]
    [ 1 -1  2  2 -2]
    [ 0  0  1  1  0]
    [ 1 -1 -1  2  2]
```

```
P=P.transpose();P
```

```
[ 1  1  1  0  1]
[-1  1 -1  0 -1]
[-2 -2  2  1 -1]
[-1 -2  2  1  2]
[-2  0 -2  0  2]
```

```
P^(-1)*A*P
```

```
[2 0 0 0 0]
[0 0 0 0 0]
[0 0 1 0 0]
[0 0 0 1 0]
[0 0 0 0 1]
```

```
P^(-1)*B*P
```

```
[2 0 0 0 0]
[0 2 0 0 0]
[0 0 3 0 0]
[0 0 0 1 0]
[0 0 0 0 2]
```

```
#Ci-dessous, on calcule u5 sans utiliser la commande
"intersection".
```

```
v5=DA[1][[0..4],4]
```

```
v5
```

```
[  0]
[  0]
[  0]
[  1]
[4/3]
```

```
B*v5
```

```
[-1/3]
[ 1/3]
[-5/3]
[ 1/3]
[10/3]
```

```
u3
```

```
(1, -1, 2, 2, -2)
```

```
u3=matrix(u3).transpose();u3
```

```
[ 1]
[-1]
[ 2]
[ 2]
[-2]
```

```
C=u3.augment(u4);C
```

```
[ 1  0]
[-1  0]
[ 2  1]
[ 2  1]
[-2  0]
```

```
C=C.augment(v5);C
```

```
[  1   0   0]
[ -1   0   0]
[  2   1   0]
[  2   1   1]
[ -2   0 4/3]
```

```
C
```

```
[  1   0   0]
[ -1   0   0]
[  2   1   0]
[  2   1   1]
[ -2   0 4/3]
```

```
C.rank()
```

```
3
```

```
M=B-2*identity_matrix(5)
```

```
M
```

```
[    1    -1     1    -1   1/2]
[   -1     1    -1     1  -1/2]
[ 17/4 -25/4     5    -6  13/4]
[ 17/4 -25/4     5    -6  13/4]
[   -2     2    -2     2    -1]
```

```
N=M*C
```

```
N
```

```
[   1    0 -1/3]
[  -1    0  1/3]
[   2   -1 -5/3]
[   2   -1 -5/3]
[  -2    0  2/3]
```

```
N.right_kernel()
```

```
Vector space of degree 3 and dimension 1 over Rational Field
Basis matrix:
[ 1 -3  3]
```

```
_.basis()
```

```
[
(1, -3, 3)
```

```
    ]
```

```
C*_[0]
```

```
    (1, -1, -1, 2, 2)
```

```
u5=matrix(_)
```

```
u5
```

```
    [ 1 -1 -1  2  2]
```

```
u1=matrix(u1).transpose();u1
```

```
    [ 1]
    [-1]
    [-2]
    [-1]
    [-2]
```

```
P=u1
```

```
P
```

```
    [ 1]
    [-1]
    [-2]
    [-1]
    [-2]
```

```
P=P.augment(u2)
```

```
P
```

```
    [ 1  1]
    [-1  1]
    [-2 -2]
    [-1 -2]
    [-2  0]
```

```
P=P.augment(u3)
```

```
P
```

```
    [ 1  1  1]
    [-1  1 -1]
    [-2 -2  2]
    [-1 -2  2]
    [-2  0 -2]
```

```
P=P.augment(u4)
```

```
P
```

```
    [ 1  1  1  0]
```

```
[-1  1 -1  0]
[-2 -2  2  1]
[-1 -2  2  1]
[-2  0 -2  0]
```

```
P=P.augment(u5.transpose())
```

```
P
```

```
[ 1  1  1  0  1]
[-1  1 -1  0 -1]
[-2 -2  2  1 -1]
[-1 -2  2  1  2]
[-2  0 -2  0  2]
```

```
P^(-1)*A*P
```

```
[2 0 0 0 0]
[0 0 0 0 0]
[0 0 1 0 0]
[0 0 0 1 0]
[0 0 0 0 1]
```

```
P^(-1)*B*P
```

```
[2 0 0 0 0]
[0 2 0 0 0]
[0 0 3 0 0]
[0 0 0 1 0]
[0 0 0 0 2]
```