Travaux pratiques

1 Listes, modules numpy, scipy, . . .

1. Liste, array

```
import numpy
L=range(7,10)
S= numpy.array(L)          # transforme la liste L en array
print(S)
S= numpy.zeros(7)          # S contient 7 fois le nombre 0
print(S)
S= numpy.ones(7)           # que contient S ?
print(S)
S= 24*numpy.ones(7)        # que contient S ?
print(S)
S= numpy.eye(7)            # que contient S ?
print(S)

import numpy as np
X=np.linspace(10,30)        # que contient X ?
print(X)
X=np.linspace(10,30,10)     # que contient X ?
print(X)
u = np.array(range(4,9))
type(u)
np.size(u)
np.shape(u)
A = np.array([[1,2,3],[4,5,6]])
type(A)
np.size(A)
np.shape(A)
```

2. Produit de matrices

```
A = np.array([[1,2], [3,4]])
B = np.array([[1,0], [-1,1]])
C=A*B                  #   C=?
D=np.dot(A,B)          #   D=?
```

3. Exemple de fonction

```
def ExempleA(n):
    A=np.zeros((n,n))
    for i in range(1,n-1):
        A[i,i-1], A[i,i], A[i,i+1] = -1,2,-1
    i=0
    A[i,i], A[i,i+1] = 2,-1
    i=n-1
    A[i,i-1], A[i,i] = -1,2
    return A

n=6
X=ExempleA(n)
print(X)
```

4. Graphiques

```
import matplotlib.pyplot as plt
from math import sin, exp
x=[z/20. for z in range(21)]
y=[sin(3*xk) for xk in x]
z=[exp(-xk) for xk in x]
plt.xlabel('x') ; plt.ylabel('sin(3x),exp(-x)') ;    # 'd' comme 'diamon
plt.plot(x,y, '-r',label='sin(3x)', linewidth=2, marker='d',markersize=
plt.plot(x,z,'-b',label='exp(-x)', linewidth=2, marker='o', markersize=
plt.legend(title='Legende', loc=8)
plt.grid()
plt.show()
```

5. Algèbre matricielle

   (a) Inverse (déconseillé pour des matrices de grande taille !), norme, rang, déterminant, spectre, . . .

```
from numpy.linalg import inv
n=7
A=ExempleA(n)
B=inv(A)
C=np.dot(A,B)
print("||AB-I|| = ",np.linalg.norm(C-np.eye(n)))
```

```
A=ExempleA(5)
r=np.linalg.matrix_rank(A)

from scipy.linalg import expm
A=np.zeros((2,2))
print('np.exp(A)  = \n', np.exp(A))      # Comparer
print('expm(A)    = \n', expm(A))        # np.exp(A) et expm(A)


from numpy.linalg import eig  # pour v.p
n=3
A=ExempleA(n)
D, V = eig(A)
k=2
d,u=D[k],V[:,k]
z=np.dot(A,u)-d*u
print(np.linalg.norm(z))
```

(b) Systèmes linéaires

```
import numpy as np
from numpy import linalg   # sous librairie algebre lineaire
n=5
A=np.random.rand(n,n)
s=np.random.rand(n)
b=np.dot(A,s)
x=linalg.solve(A,b)
normex=linalg.norm(s-x)
```

- Équations non linéaires

  (a) Cas scalaire

```
# resoudre exp(-x)=sin(x)
import matplotlib.pyplot as plt
x=np.linspace(0,10)
plt.plot(x,np.exp(-x),'b+',x,np.sin(x),'or')
from scipy.optimize import fsolve
from math import exp, sin
def f(x):
    return exp(-x)-sin(x)
```

```
x=fsolve(f, 3)
print('    x = ', x)
print('f(x) = ', f(x))
```

(b) Cas vectoriel

```
# resoudre le systeme :
# exp(x) + x y^2 =1,
# sin(x+y) + x^2 =1

def f(p):
    x, y = p[0], p[1]
    q=[exp(x)+x*y*y-1,sin(x+y)+x*x-1]
    return q

x=  fsolve(f, (0,0))
print('    x = ', x)
print('f(x) = ', f(x))
```