Fibonacci

April 14, 2020

1 Autour de la suite de Fibonacci

[1]:
```
import numpy as np

import numpy.linalg as alg

import matplotlib.pyplot as plt
```

1.1 Etude de la suite de Fibonacci

$u_{n+2} = u_{n+1} + u_n$ avec $u_0 = 0$ et $u_1 = 1$.

On pose $X_n = (u_n, u_{n+1})$. On a $X_{n+1} = AX_n$ avec $A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$.

[2]:
```
A= np.array([[0,1],[1,1]]) # matrice de la suite de Fibonacci
A
```

[2]:
```
array([[0, 1],
       [1, 1]])
```

[3]:
```
def orbite(A,V,n): # fonction calculant l'orbite par A de la donnée initiale V
    X=np.zeros((n,2))
    X[0]=V
    for i in np.arange(1,n):
        X[i]=A.dot(X[i-1])
    return X
```

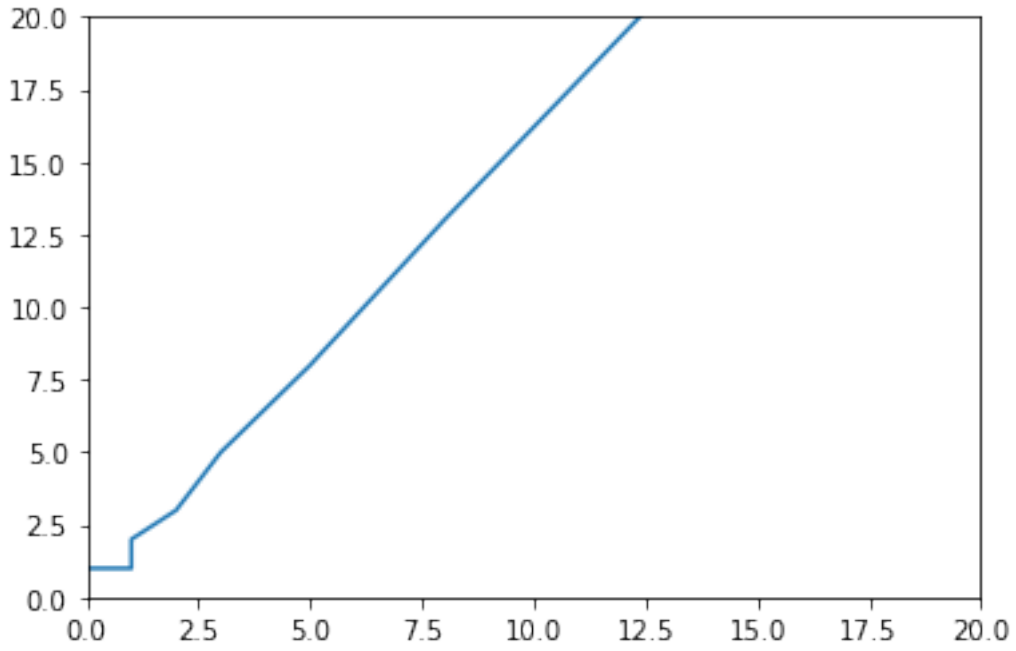
Calcul de la suite de Fibonacci pour $u_0 = 0$ et $u_1 = 1$, c'est-à-dire $X_0 = (0, 1)$.

[4]:
```
n=15
X=orbite(A,np.array([0,1]),n)
X
```

[4]:
```
array([[  0.,   1.],
       [  1.,   1.],
       [  1.,   2.],
       [  2.,   3.],
```

```
[  3.,   5.],
[  5.,   8.],
[  8.,  13.],
[ 13.,  21.],
[ 21.,  34.],
[ 34.,  55.],
[ 55.,  89.],
[ 89., 144.],
[144., 233.],
[233., 377.],
[377., 610.]])
```

[5]:
```
plt.plot(X[:,0],X[:,1],'-')
plt.axis([0,20,0,20])
#plt.axis('equal')
plt.show()
```



Croissance de la suite u_n : calcul de u_{n+1}/u_n qui est aussi la pente de X_n.

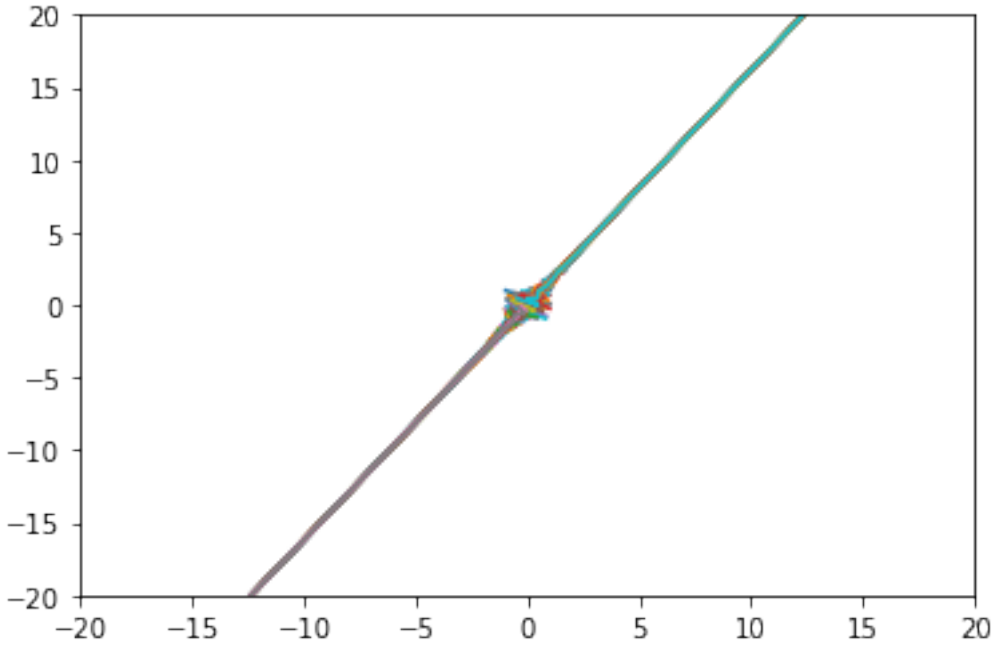
[6]:
```
X[1:n,1]/X[1:n,0]
```

[6]:
```
array([1.        , 2.        , 1.5       , 1.66666667, 1.6       ,
       1.625     , 1.61538462, 1.61904762, 1.61764706, 1.61818182,
       1.61797753, 1.61805556, 1.61802575, 1.61803714])
```

1.2 Etude suivant une condition initiale aléatoire

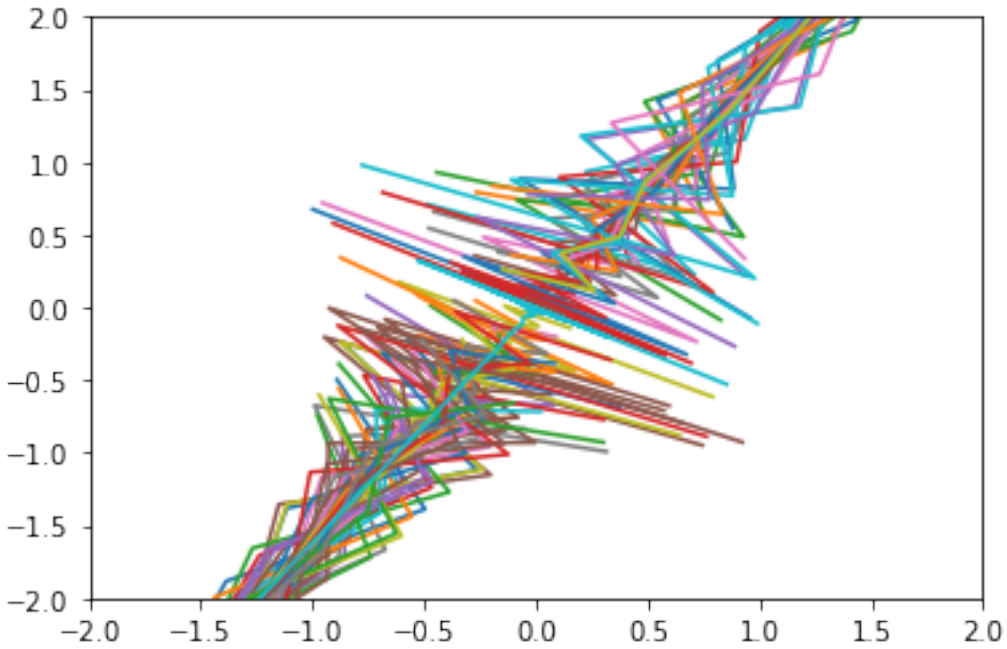
1.2.1 Cent orbites à grande échelle!

[7]:
```
for i in np.arange(100): # 100 oribtes
    V=2*np.random.rand(2)-np.array([1,1]) # vecteur aléatoire dans [-1,1]x[-1,1]
    X=orbite(A,V,20)
    plt.plot(X[:,0],X[:,1], '-')
    plt.axis([-20,20,-20,20])#plt.axis('equal')
    #plt.ylim(-10,10)
```



1.2.2 Cent orbites en zoomant un peu en (0,0).

[8]:
```
for i in np.arange(100):
    V=2*np.random.rand(2)-np.array([1,1]) # vecteur aléatoire dans [-1,1]x[-1,1]
    X=orbite(A,V,20)
    plt.plot(X[:,0],X[:,1], '-')
    plt.axis([-2,2,-2,2])#plt.axis('equal')
    #plt.ylim(-10,10)
```



1.3 Valeurs propres et vecteurs propres de A

[9]:
```
alg.eig(A)
```

[9]:
```
(array([-0.61803399,  1.61803399]),
 array([[-0.85065081, -0.52573111],
        [ 0.52573111, -0.85065081]]))
```

La valeur propre la plus grande $\lambda_2 = 1.6183399$ donne la croissance de la suite de Fibonacci.

En fait $\lambda_2 = \frac{1+\sqrt{5}}{2}$ (nombre d'or) d'après le cours.

[10]:
```
V=alg.eig(A)[1] # extraction des vecteurs propres de A
V
```

[10]:
```
array([[-0.85065081, -0.52573111],
       [ 0.52573111, -0.85065081]])
```

[11]:
```
V/V[0,:] # pentes des vecteurs propres = lambda_1 et lambda_2 !
```

[11]:
```
array([[ 1.        ,  1.        ],
       [-0.61803399,  1.61803399]])
```

Un nuage de vecteurs aléatoires dans $[-1, 1]^2$.

[12]:
```
for i in np.arange(100):
    U=2*np.random.rand(2)-np.array([1,1])
    plt.plot(U[0],U[1],'*')
```