Alignés par Frobenius ? (Denise Vella-Chemla, 3.10.2020)

Il s'agit ici de donner témoignage de propriétés dont on ne comprend pas la cause, qui sont peut-être triviales, mais qui interrogent.

Poursuivant les expérimentations informatiques du dernier mois, on continue de représenter les nombres par des nombres complexes particuliers qui sont leur image par certaines fonctions. Ici, on utilise des variations du programme Python suivant :

```
import math
from math import *
import matplotlib
from matplotlib import *
import matplotlib.pyplot as plt
from matplotlib.pyplot import *
import mpmath
from mpmath import *

def prime(atester):
  pastrouve = True ; k = 2 ;
  if (atester in [0,1]): return False ;
  if (atester in [2,3,5,7]): return True ;
  while (pastrouve):
    if ((k * k) > atester): return True
    else:
      if ((atester % k) == 0): return False
      else: k=k+1

fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')

x = 1 ; n = 100 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
  x0, y0 = 0, 0
  t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))                  *****
  tprime=(1.0-(1.0/t)**50)/(1.0-(1.0/t))                   *****
  x0 = x0 + tprime.real
  y0 = y0 + tprime.imag
  #print(str(x0)+'    '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  #if (x != 1):
    #ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
  xprec = x0
  yprec = y0
  x = x+1

xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
print(ymax)
ax.set_xlim(xmin-10, xmax+10) ;
ax.set_ylim(ymin-10, ymax+10)
plt.show()
sys.exit(0)
```

Comme on le voit dans les lignes de code marquées par cinq étoiles, il s'agit de calculer soit $t = e^{\frac{2i\pi\ x}{n}}$, soit sa racine complexe $t^{1/2}$, puis d'appliquer à t la fonction

$$f(t) = \sum_{k=1}^{v} \left(\frac{1}{t}\right)^k = \frac{1-\left(\frac{1}{t}\right)^{v+1}}{1-\left(\frac{1}{t}\right)}$$

et de "plotter" le complexe correspondant.

La source de l'incompréhension est la suivante : selon les valeurs choisies pour n et v, on parvient à faire s'aligner les points ou pas dans le plan complexe, mais on ne comprend vraiment pas pour quelle raison ils s'alignent ou pas.

Dans la mesure où il est très lourd de fournir tous les programmes et leurs exécutions, on fournit ci-dessous quelques exemples illustrant :[1]

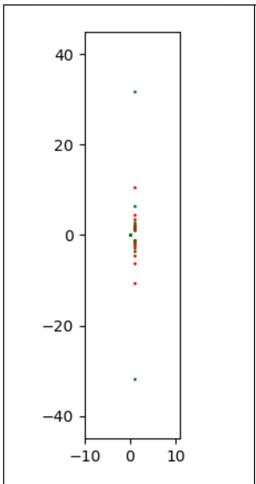

Fig. 1

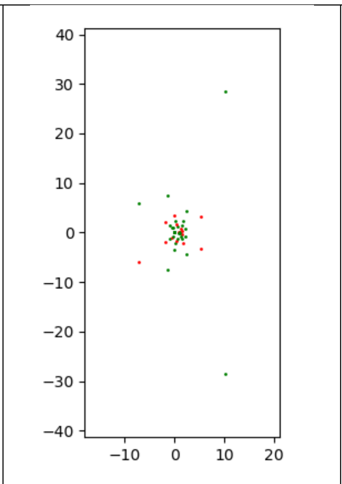

Fig. 2

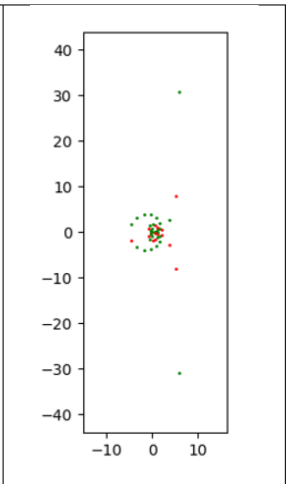

Fig. 3

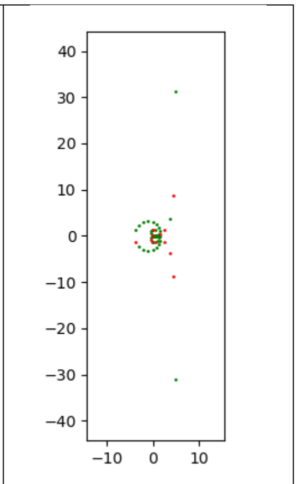

Fig. 4

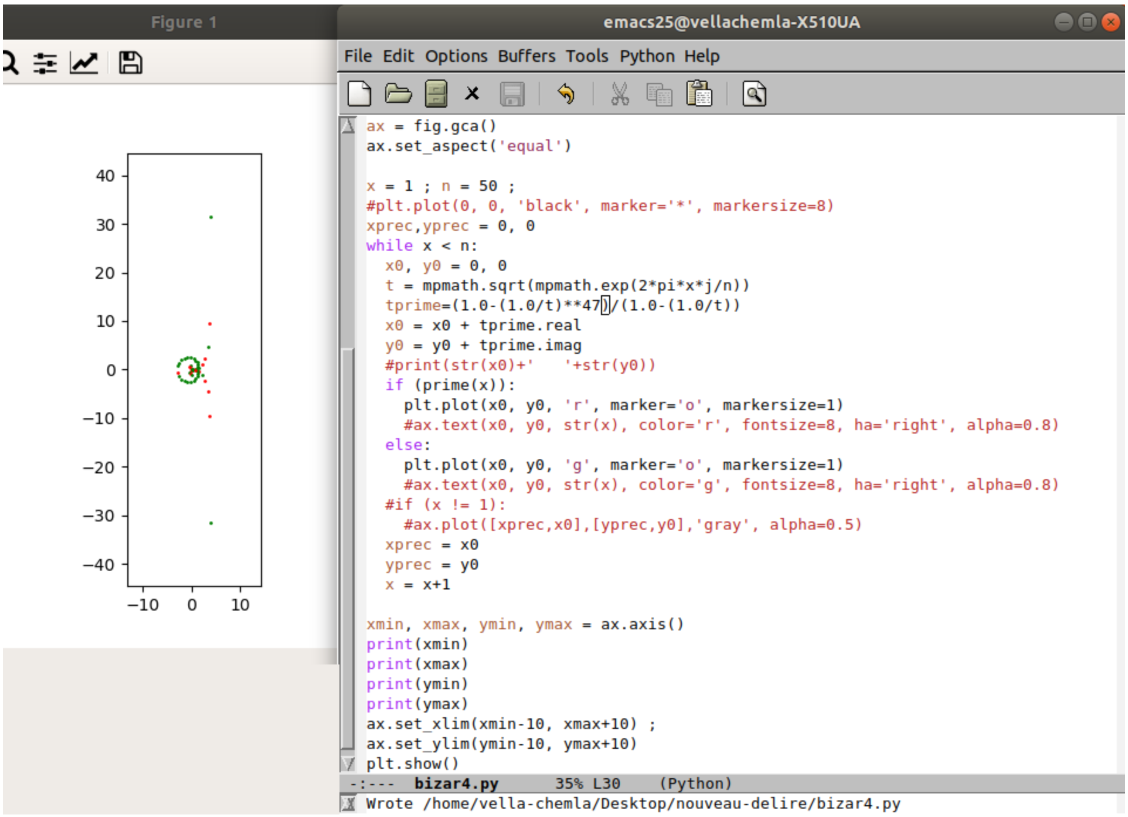

Fig. 5

1. On ne comprend également pas pourquoi les résultats des programmes Python ne sont pas identiques lorsqu'on écrit mpmath.sqrt(mpmath.exp(2*j*pi*x/n)) et mpmath.exp(j*pi*x/n) (*attention* : le complexe $i = \sqrt{-1}$ s'écrit j en Python.)

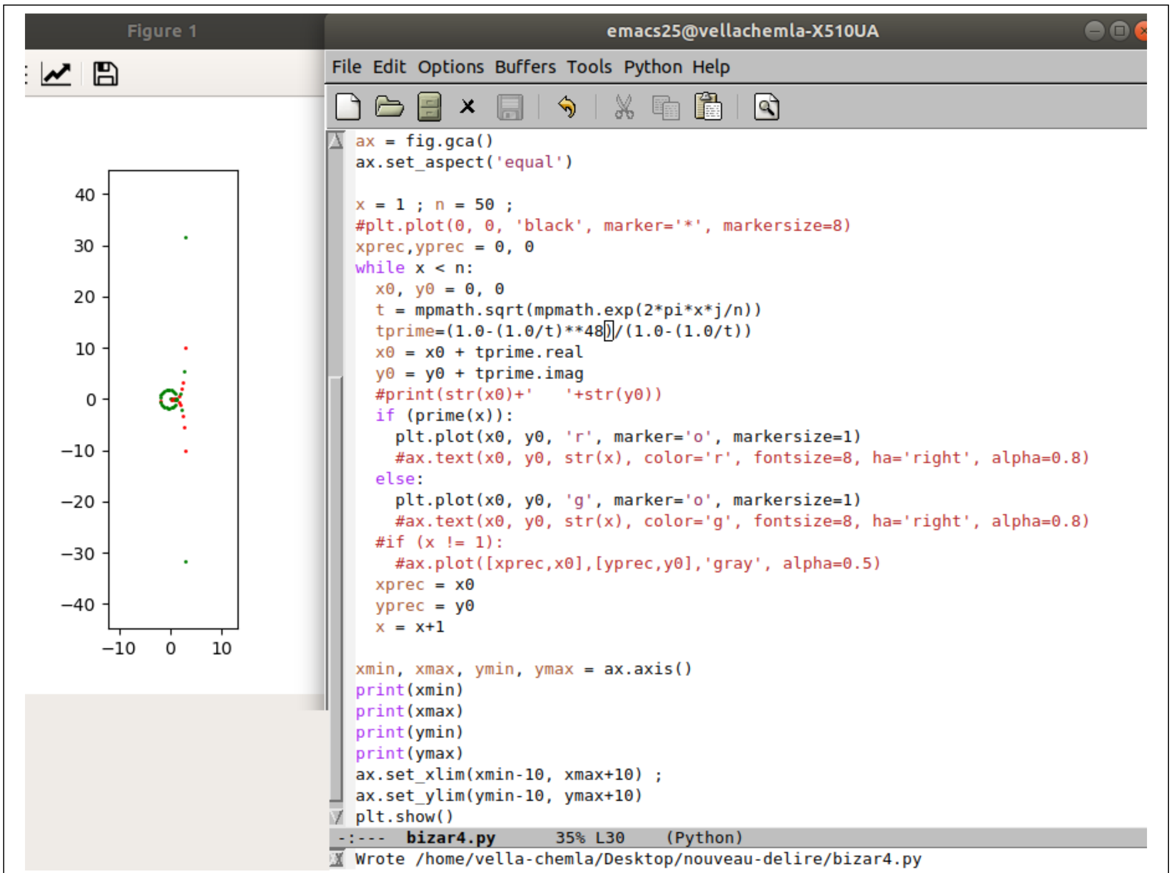

Fig. 6

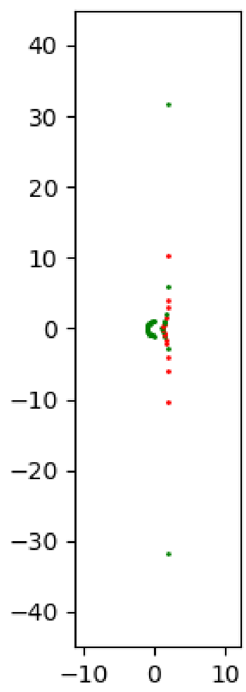

Fig. 7

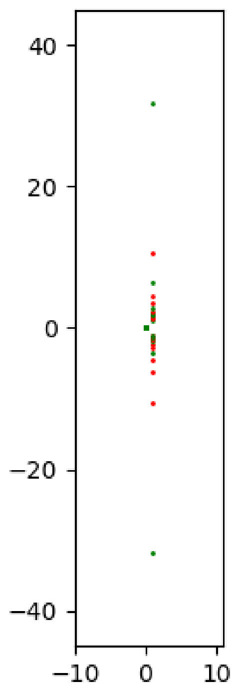

Fig. 8

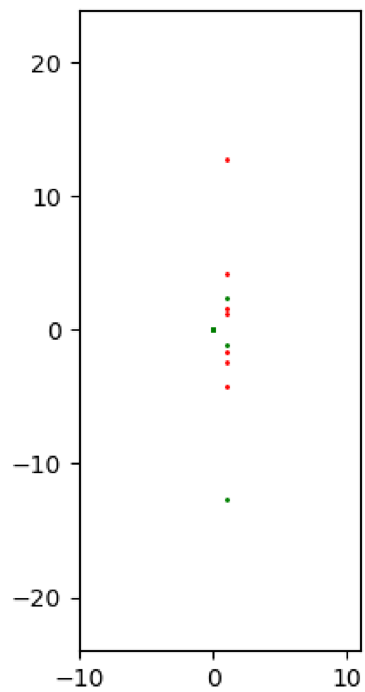

Fig. 9

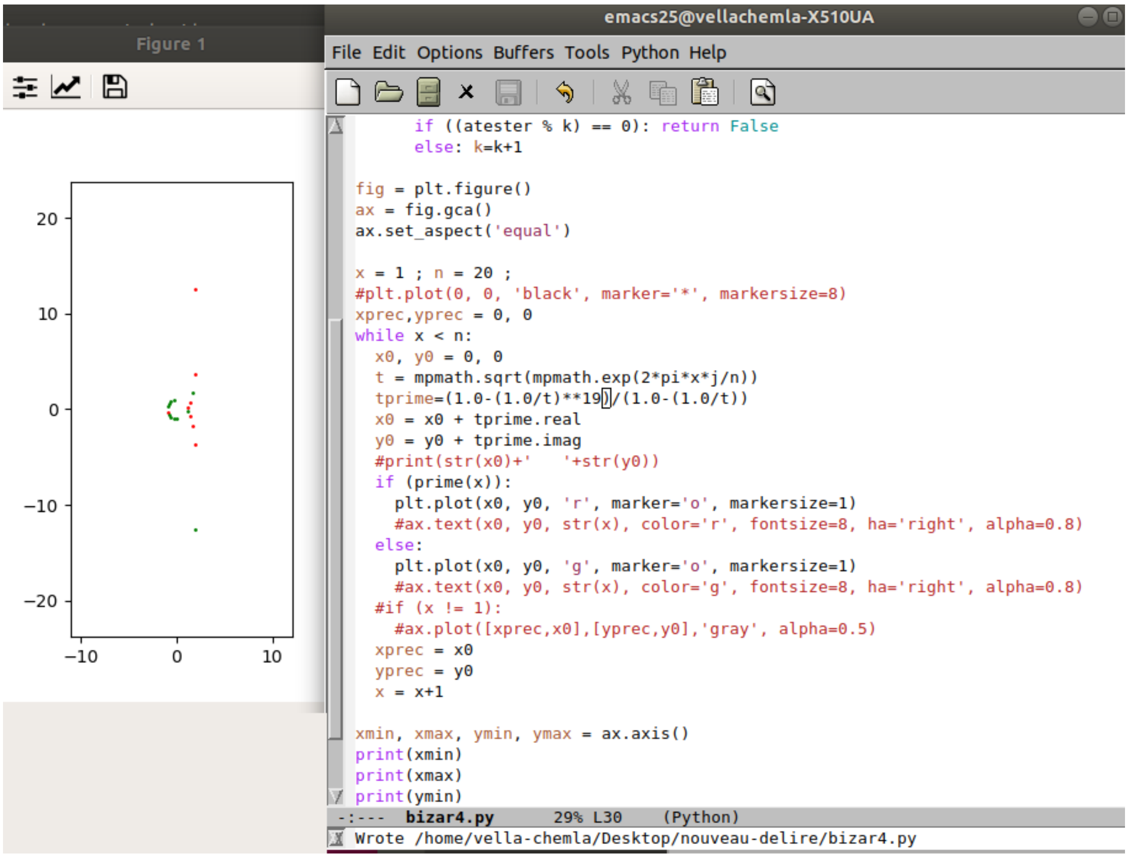
Figure 1
20
10
0
−10
−20
−10
0
10
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
    if ((atester % k) == 0): return False
    else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
x = 1 ; n = 20 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
  x0, y0 = 0, 0
  t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))
  tprime=(1.0-(1.0/t)**19)/(1.0-(1.0/t))
  x0 = x0 + tprime.real
  y0 = y0 + tprime.imag
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  #if (x != 1):
    #ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
  xprec = x0
  yprec = y0
  x = x+1
xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
-:--- bizar4.py   29% L30   (Python)
Wrote /home/vella-chemla/Desktop/nouveau-delire/bizar4.py

Fig. 10

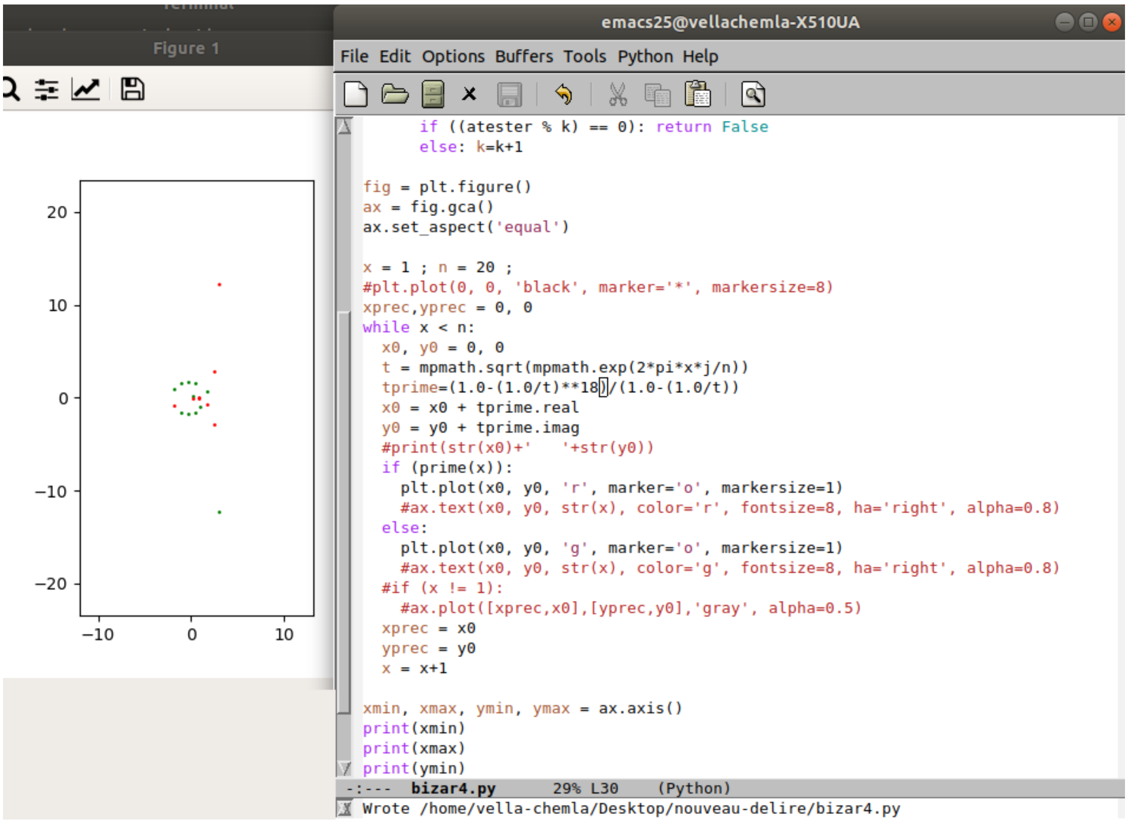
Figure 1
20
10
0
−10
−20
−10
0
10
emacs25@vellachemla-X510UA
File Edit Options Buffers Tools Python Help
    if ((atester % k) == 0): return False
    else: k=k+1
fig = plt.figure()
ax = fig.gca()
ax.set_aspect('equal')
x = 1 ; n = 20 ;
#plt.plot(0, 0, 'black', marker='*', markersize=8)
xprec,yprec = 0, 0
while x < n:
  x0, y0 = 0, 0
  t = mpmath.sqrt(mpmath.exp(2*pi*x*j/n))
  tprime=(1.0-(1.0/t)**18)/(1.0-(1.0/t))
  x0 = x0 + tprime.real
  y0 = y0 + tprime.imag
  #print(str(x0)+'   '+str(y0))
  if (prime(x)):
    plt.plot(x0, y0, 'r', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='r', fontsize=8, ha='right', alpha=0.8)
  else:
    plt.plot(x0, y0, 'g', marker='o', markersize=1)
    #ax.text(x0, y0, str(x), color='g', fontsize=8, ha='right', alpha=0.8)
  #if (x != 1):
    #ax.plot([xprec,x0],[yprec,y0],'gray', alpha=0.5)
  xprec = x0
  yprec = y0
  x = x+1
xmin, xmax, ymin, ymax = ax.axis()
print(xmin)
print(xmax)
print(ymin)
-:--- bizar4.py   29% L30   (Python)
Wrote /home/vella-chemla/Desktop/nouveau-delire/bizar4.py

Fig. 11

Ci-dessous, le même graphique est fourni que sur la Figure 11 mais en écrivant les nombres dont sont plottées les images.

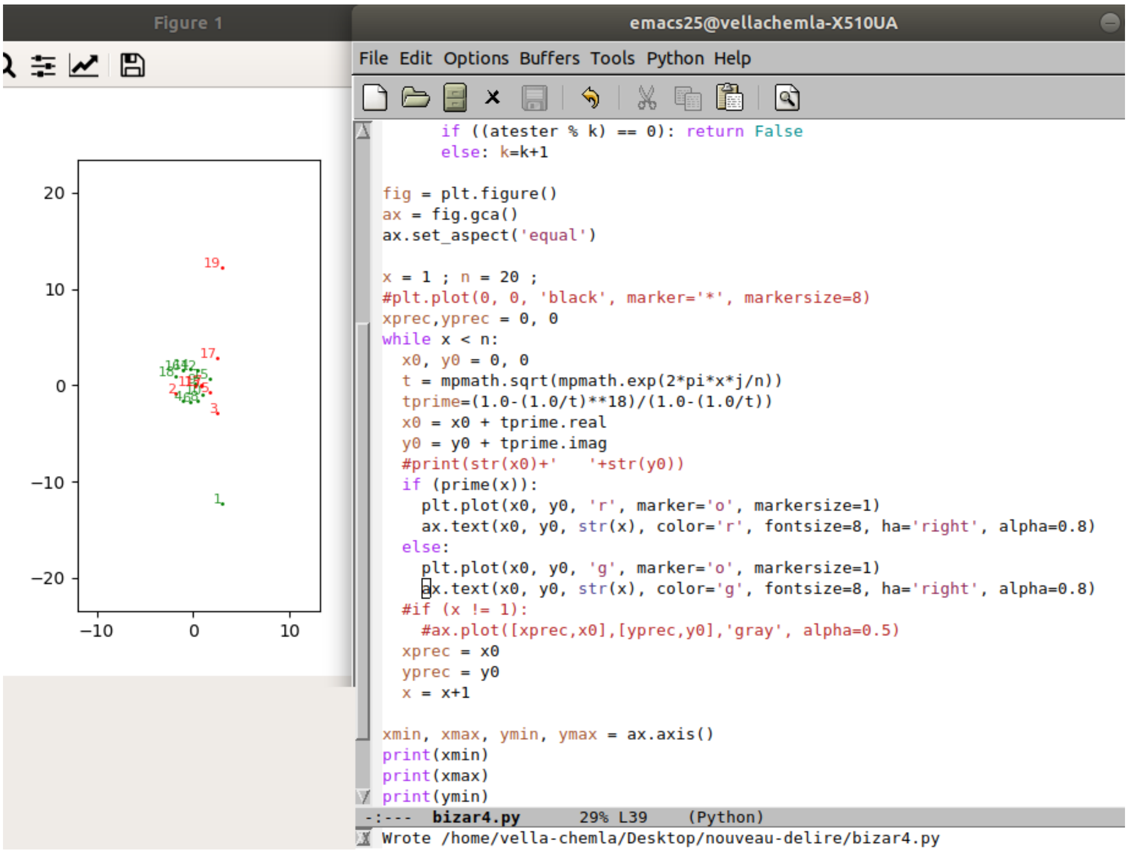

Fig. 12

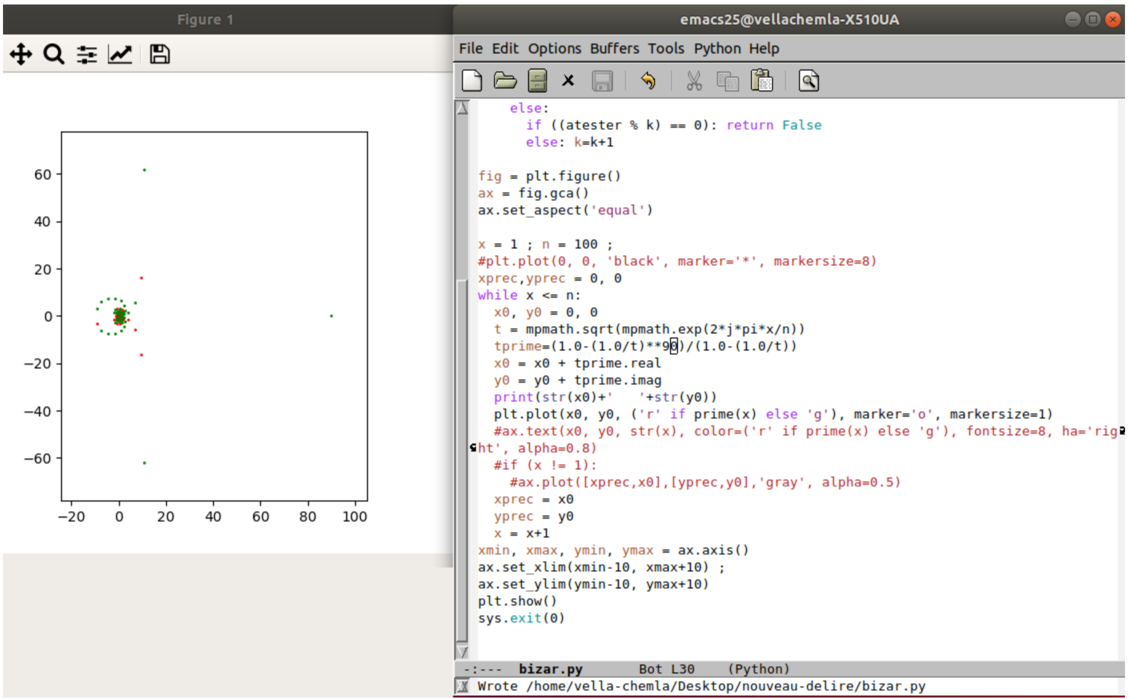

Fig. 13

Dans les figures 1, 8 et 9, pour lesquelles on constate l'alignement des nombres complexes images, ont systématiquement été sommées les puissances jusqu'à la n-ième au numérateur.

Ci-après, sur les Figures 14 et 15, en sommant les puissances jusqu'à la 50ème, pour x compris entre 1 et 100 (donc jusqu'à la $n/2$-ième puissance seulement, et non jusqu'à la n-ième comme précédemment), les nombres complexes images sont alignés sur deux droites, selon que ce sont les images de nombres premiers de la forme $4u+1$ ou $4u+3$ [2].

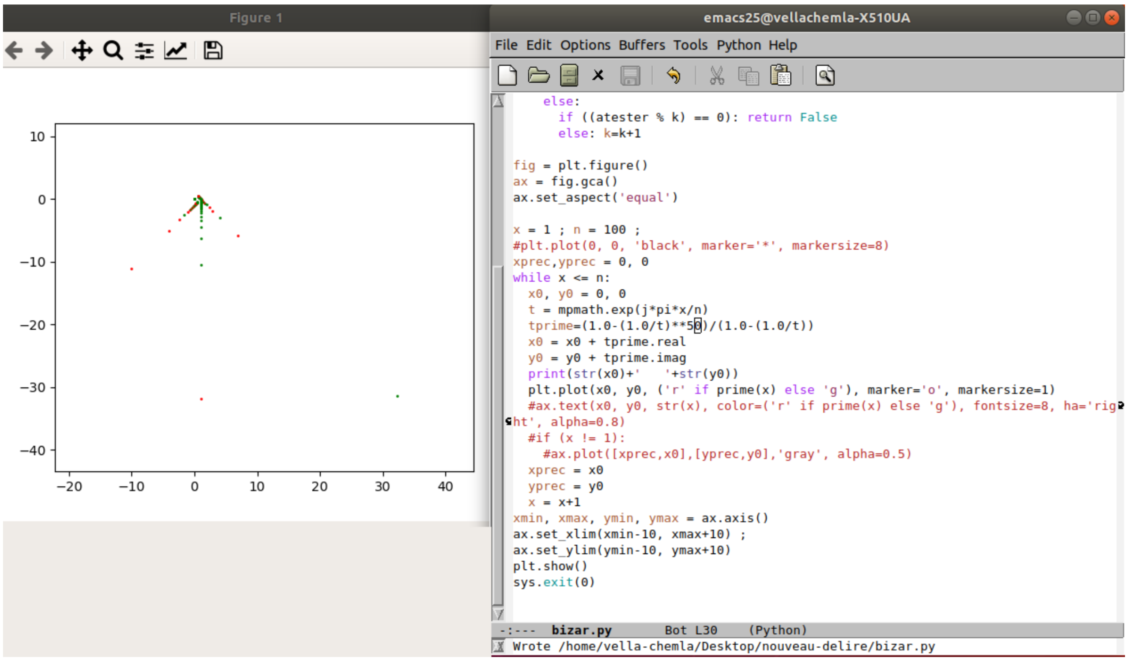

Fig. 14

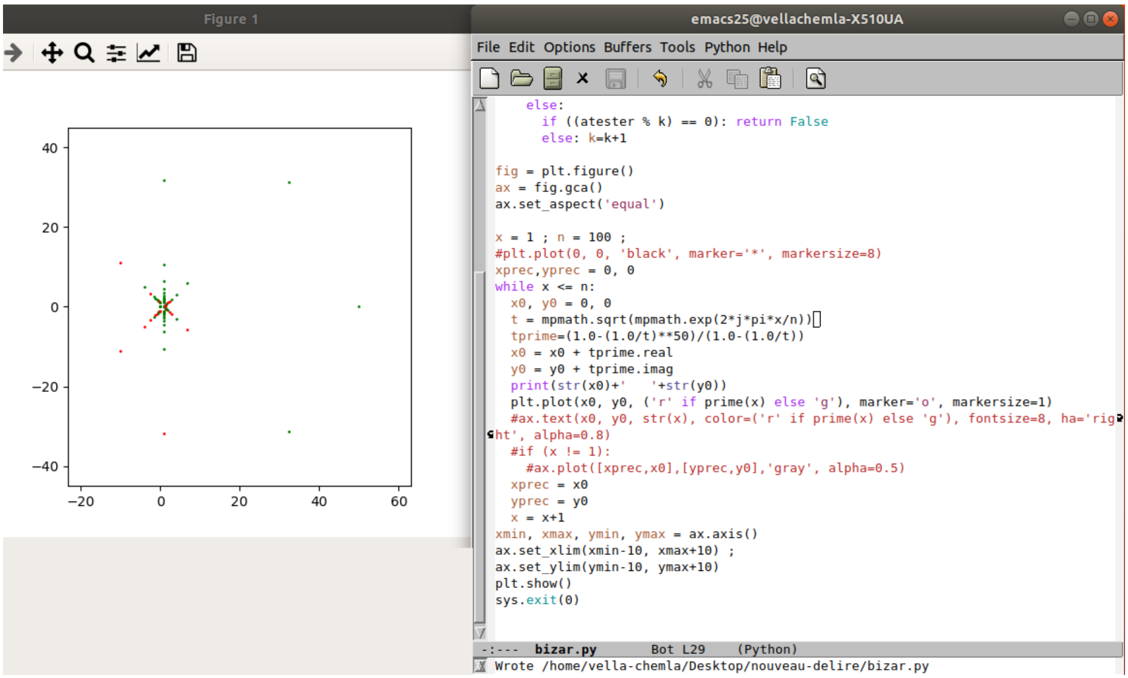

Fig. 15

2. C'est entre ces deux exécutions qu'on a cette incompréhension de la note de bas de page vue plus haut.

On aimerait peut-être relier nos découvertes à cette équation :

$$\left(\frac{1}{\sqrt{3}} + i\right)^3 - \left(\frac{1}{\sqrt{3}} - i\right)^3 = 0$$

et à ses variantes faisant intervenir des puissances autres que des cubes.