Calcul scientifique et numériques : Ensembles Fractales (Newton, Julia, Mandelbrot)

PC/PC* - Lycée Thiers

2014/2015

Fractales

- Méthode de Newton et Fractale de Newton
- Les ensembles de Julia
- L'ensemble de Mandelbrot

Fractale : convergence de la méthode de Newton

• Soit f une application réelle de classe au moins C^1. Soit une suite $(u_n)_n$ construite par la méthode de Newton :

$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}$$

Soit u un zéro de f (c.à d. $f(u) = 0$) en lequel $f'(u) \neq 0$.

• La suite (u_n) converge vers u lorsque u_0 est suffisamment proche de u.

• Réciproquement, si la suite (u_n) converge vers u, alors u est un zéro de f ; en effet par passage à la limite :

$$(u_{n+1} - u_n)f'(u_n) = -f(u_n) \underset{(u_n \to u)}{\Longrightarrow} f(u) = (u-u)f'(u) = 0$$

• La convergence d'une suite définie par la méthode Newton est un problème très difficile. L'ensemble des points u_0 pour laquelle la suite (u_n) converge est un ensemble très complexe.

• On peut s'en convaincre en étendant la suite au plan complexe (c'est possible notamment lorsque f est un polynôme et f' son polynôme dérivée) :

$$(u_n) : \begin{cases} u_0 \in \mathbb{C} \\ u_{n+1} = u_n - \dfrac{f(u_n)}{f'(u_n)} \end{cases}$$

• Illustrons le phénomène en prenant pour f le polynôme $z \mapsto z^3 - 1$ (et donc $f' : z \mapsto 3z^2$) :

$$u_0 \in \mathbb{C} \qquad \forall\, n \in \mathbb{N} \quad u_{n+1} = u_n - \frac{u_n^3 - 1}{3u_n^2} = \boxed{\frac{2u_n^3 + 1}{3u_n^2}} = u_{n+1}$$

• La fonction f a 3 racines dans $\mathbb{C}$: 1, $\exp(2i\pi/3)$, $\exp(4i\pi/3)$.

Fractale : convergence de la méthode de Newton

• Il faut choisir un critère approximatif de convergence. Prenons par exemple :
Donné $u_0 \in \mathbb{C}$, u_{30} est à distance $< tol = 0.01$ de l'une des 3 racines.
C'est à dire, en notant les racines $z_0 = 1$, $z_1 = \frac{-1+i.\sqrt{3}}{2}$, $z_2 = \frac{-1-i.\sqrt{3}}{2}$:

$$\boxed{\min(|u_{30} - z_0|, |u_{30} - z_1|, |u_{30} - z_2|) < 0.01}$$

• Bien sûr en prenant un rang n plus grand (> 30) et une tolérance tol plus fine (< 0.01) on obtiendrait un tracé plus fidèle. A tester empiriquement pour obtenir un bon compromis entre précision du tracé et rapidité d'éxécution.

• Remarque : pour manipuler un polynôme arbitraire :

```
P = np.poly1d([a, b, c, d])
```

avec pour paramètre une liste de coefficients `[a,b,c,d]` définit le polynôme :

$$P(X) = a.X^3 + b.X^2 + c.X + d \quad \in \mathbb{C}[X]$$

(et ainsi de suite pour toute liste de coefficients).
– `P(x)` évalue `P` au nombre (complexe) `x`.
– `np.roots(P)` retourne le tableau `numpy` des racines de `P`.
– `P.order` retourne le degré de `P`.
– `np.poly1d.deriv(P)` retourne la liste des coefficients du polynôme dérivée de `P`.
– Toutes les opérations `+,-,*,**` sont correctement implémentées sur les polynômes.

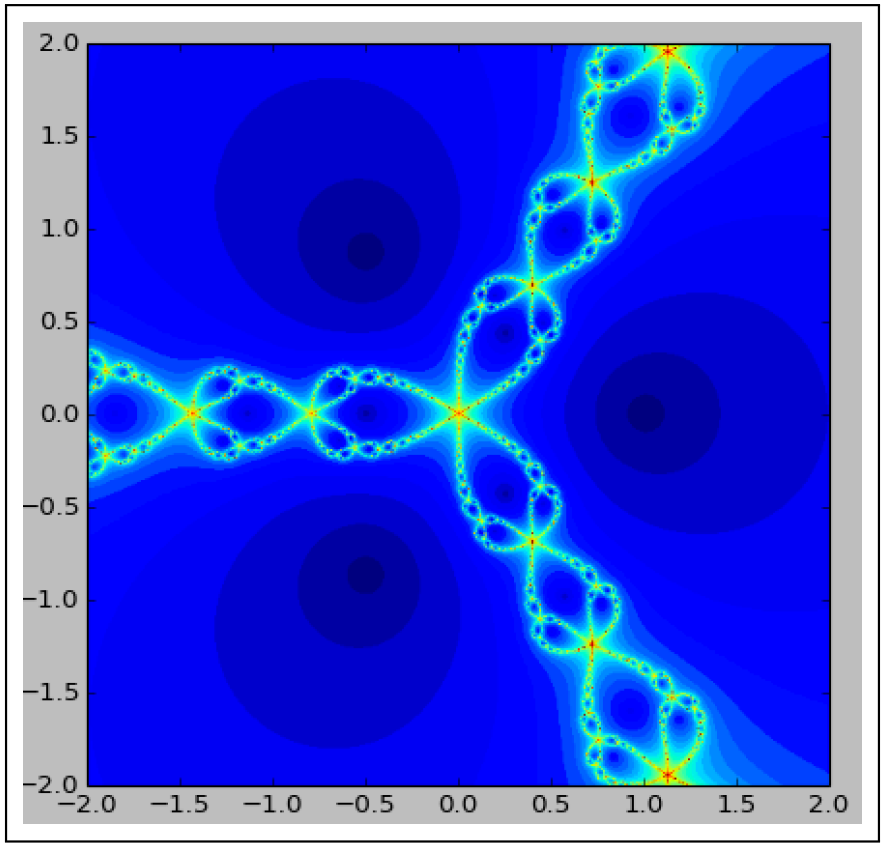
Fractale : convergence de la méthode de Newton

```
# Importation des modules nécessaires :
import numpy as np
import matplotlib.pyplot as plt

# Les 3 racines complexes de f :
z0 = 1
z1 = -0.5+((3**0.5) / 2)*1j
z2 = -0.5-((3**0.5) / 2)*1j

# Fonction F de récurrence : u(n+1) = F(u(n)) :
F = lambda z: (2*z**3 +1)/(3*z**2)

# Fonction estimant la convergence de (un) et sa rapidité pour u0 = z :
N = 30    # Profondeur d'itération
tol = 0.01    # Tolérance
def g(z):
    for i in range(N):
        z = F(z)
        m = min(abs(z-z0),abs(z-z1),abs(z-z2))
        if m < tol:
            return i
    return N

# Vectorisation de g pour l'appliquer à un tableau :
gv = np.vectorize(g)
```

Fractale : convergence de la méthode de Newton

```
# Constitution de l'image

# Constantes pour la précision du tracé
K = 500

# Création de la grille pour le tracé :
x = np.linspace(-2,2,K)
y = np.linspace(-2,2,K)
X,Y = np.meshgrid(x,y)

# Tableau à convertir en image :
Tab = gv(X+Y*1j)

# Tracé :
plt.clf()    # Effacement de la fenêtre graphique (optionnel)
plt.imshow(Tab, extent=[-2,2,-2,2])    # Création de l'image
# l'option extent définit les plages de valeurs des 2 axes
plt.show()
```

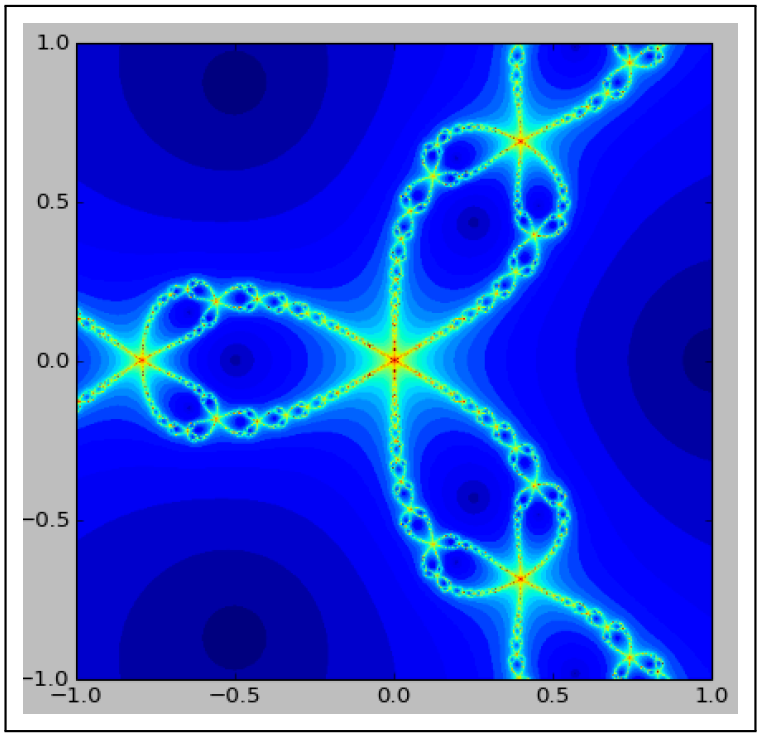
Fractale : convergence de la méthode de Newton



Les points "de divergence" sont en rouge foncé. Tous les autres points convergent vers l'une des 3 racines (en bleu foncé), à une vitesse d'autant plus grande qu'il apparaît plus bleuté.

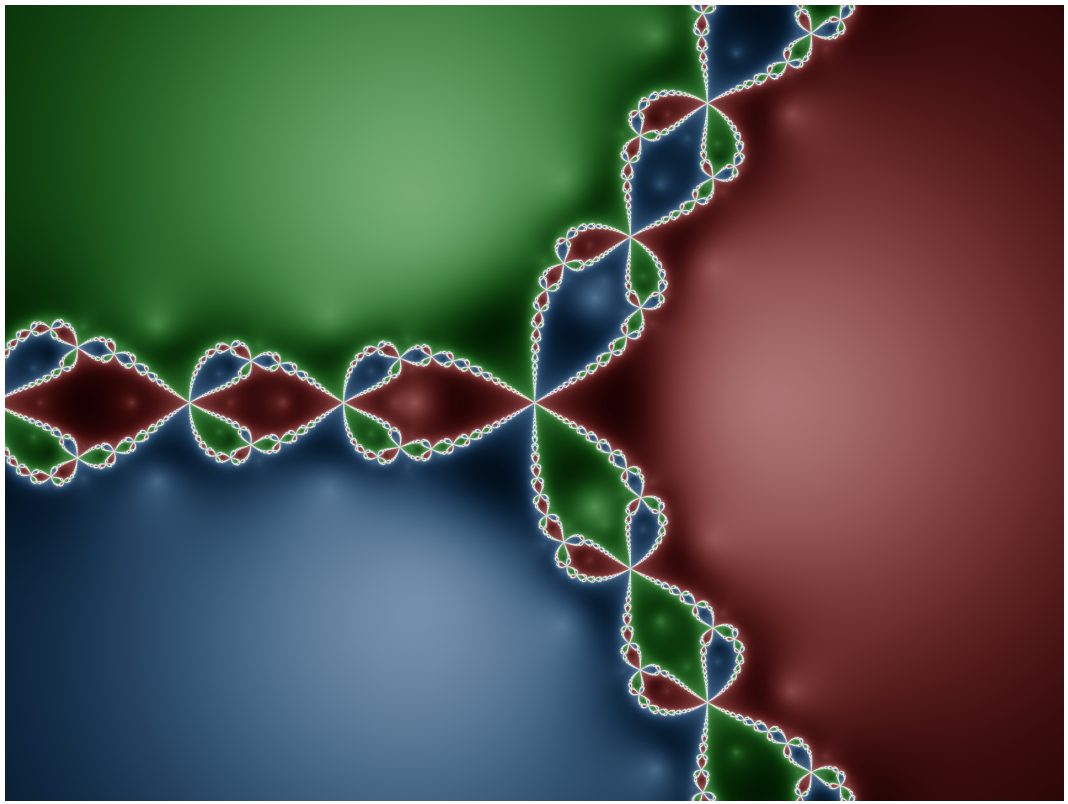
En fait les frontières des bassins d'attraction (en bleu) sont des points divergents.

Fractale : convergence de la méthode de Newton




```
x = np.linspace(-1,1,K)
y = np.linspace(-1,1,K)
X,Y = np.meshgrid(x,y)
Tab = gv(X+Y*1j)
plt.figure(2)
plt.imshow(Tab, extent=[-1,1,-1,1])
plt.show()
```

Fractale : convergence de la méthode de Newton



Sur cette image en haute résolution apparaît la fractale de Newton avec :

– les 3 bassins d'attraction dont la couleur (rouge, vert, bleu) code la limite $(1, e^{2i\pi/3}, e^{4i\pi/3})$,

– la clarté code la rapidité de convergence.

– les points de divergence en blanc sont les frontières des bassins d'attraction.

Fractale : Ensembles de Julia

• Etant donnés deux nombres complexes z_0 et c et la suite complexe (z_n) définie par la relation de récurrence :

$$z_{n+1} = z_n^2 + c$$

l'ensemble de Julia est la frontière (au sens topologique sur l'espace vectoriel normé $(\mathbb{C}, |.|)$) de l'ensemble des points $z_0 \in \mathbb{C}$ pour lesquels la suite (z_n) est bornée (i.e. $\exists\, M > 0$, tel que $\forall\, n \in \mathbb{N}$, $|z_n| < M$).

• Critère de majoration : on prend comme critère, que pour des constantes M et m à choisir que :

$$\boxed{\forall\, n \in [[0, m]], \quad |z_n| \leqslant M}$$

• Les ensembles de Julia peuvent se définir pour n'importe quel polynôme complexe. Ils donnent des ensembles fractales importants en *dynamique holomorphe*. Lorsque z_0 appartient à l'ensemble de Julia, une variation infinitésimale de z_0 peut entraîner de grandes différences dans le comportement de la suite (z_n). Son complémentaire est un ouvert dans $\mathbb{C}$ appelé l'ensemble de Fatou : domaine où une faible variation de z_0 entraîne une faible variation du comportement de (z_n) (stabilité).

Leur noms sont en hommage aux mathématiciens français Pierre Fatou et Gaston Julia dont les travaux au début du XXe siècle sont à l'origine de la discipline mathématiques de grand essor depuis pour comprendre les phénomènes 'chaotiques'.

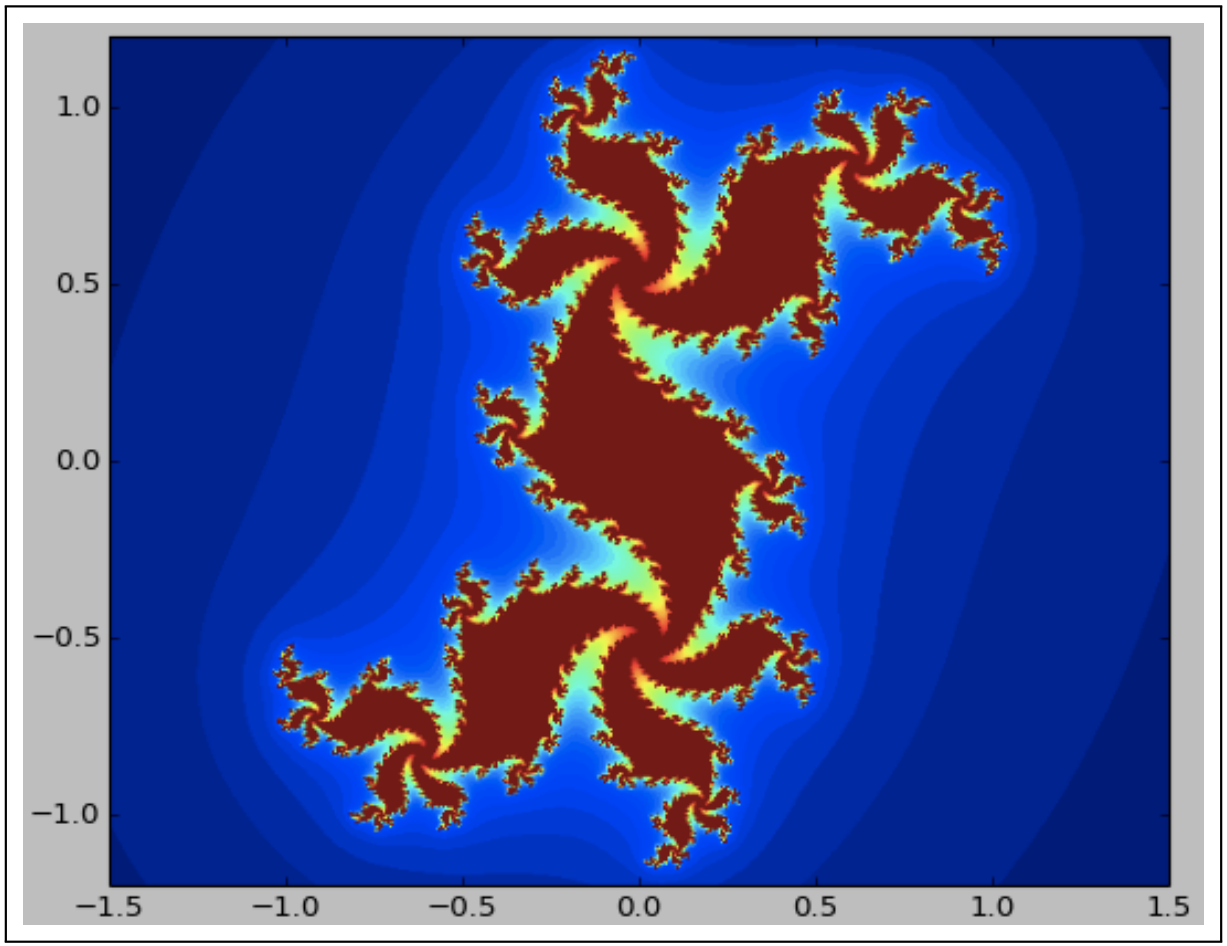
Fractale : Ensembles de Julia

```
import numpy as np
import matplotlib.pyplot as plt

M=70 ; m=50    # Constantes à choisir

def g(u,c):    # Critère de rapidité de non-majoration
    result = m
    for k in range(m+1):
        u = u**2 + c
        if abs(u) > M:
            result = k
            break
    return result

g_v = np.vectorize(g)    # Vectorisation de g

K=500    # Précision du tracé

def julia(c=1):    # Tracé de l'ensemble de Julia
    x = np.linspace(-1.5,1.5,K)
    y = np.linspace(-1.2,1.2,K)
    X, Y = np.meshgrid(x,y)
    Tab = g_v(X+1j*Y,c)
    plt.imshow(Tab, extent = [-1.5, 1.5, -1.2, 1.2])
    plt.show()
```

Fractale : Ensembles de Julia

- Exemples :

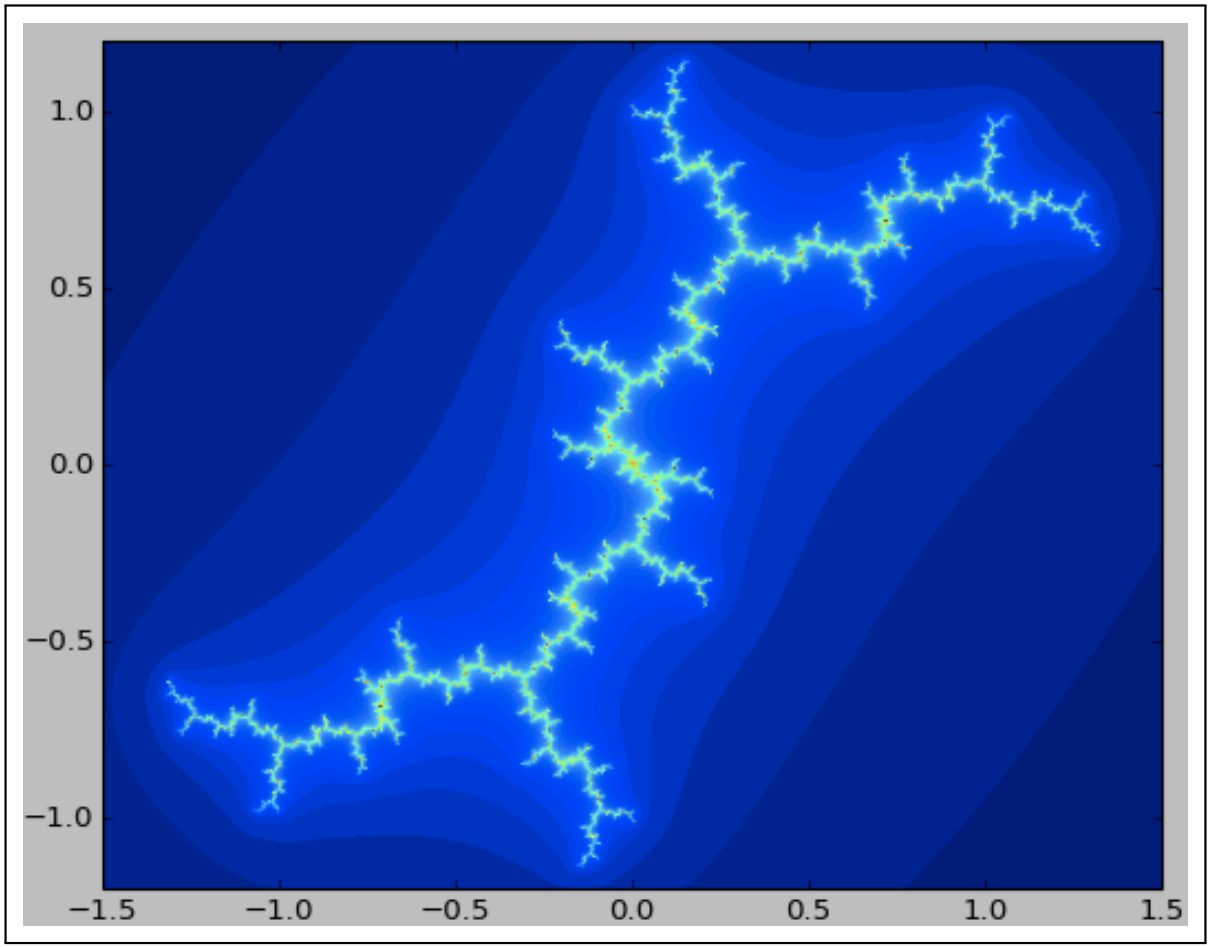
```
>>> julia(0.3+0.5j)
```



Fractale : Ensembles de Julia

- Exemples :

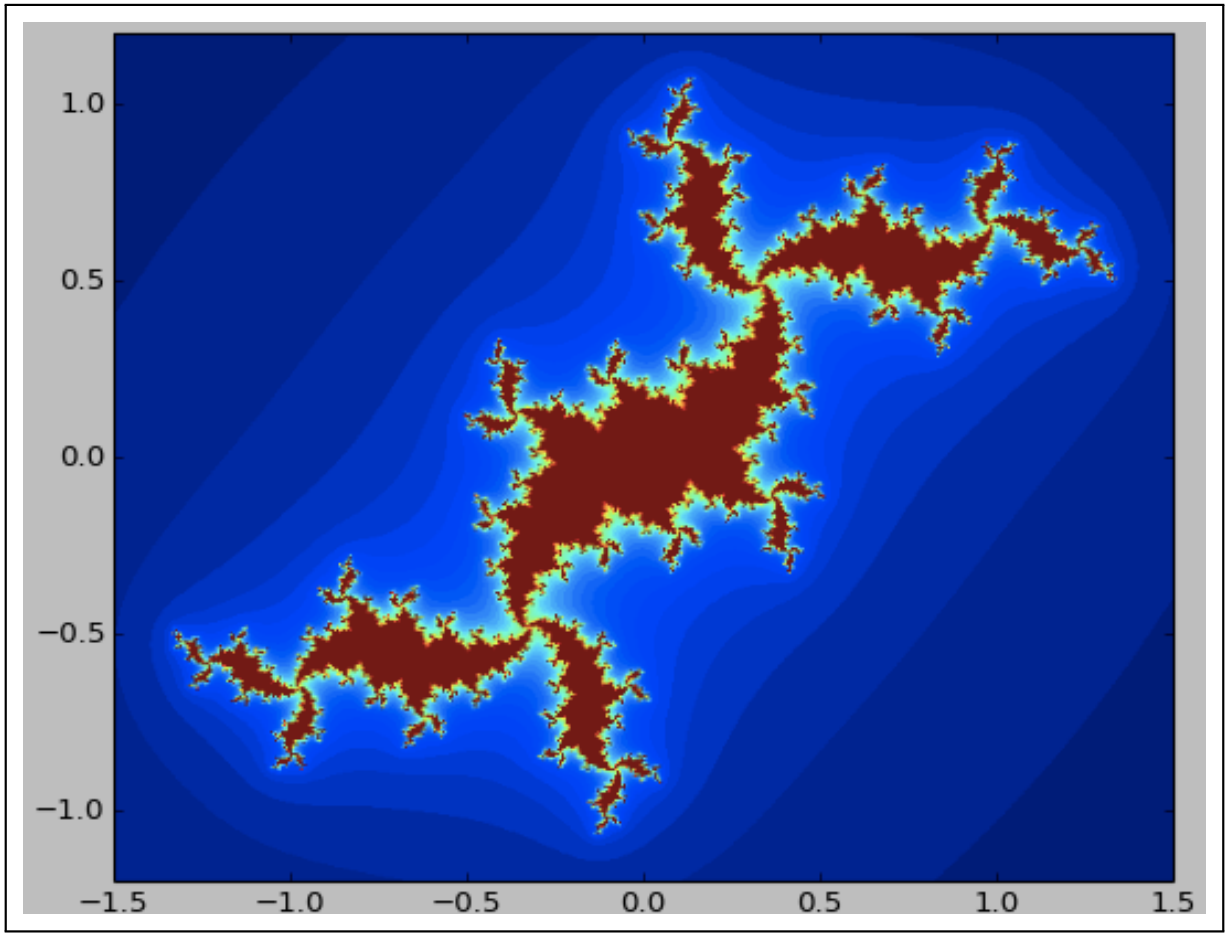
```
>>> julia(-0.038088+0.9754633j)
```



Fractale : Ensembles de Julia

- Exemples :

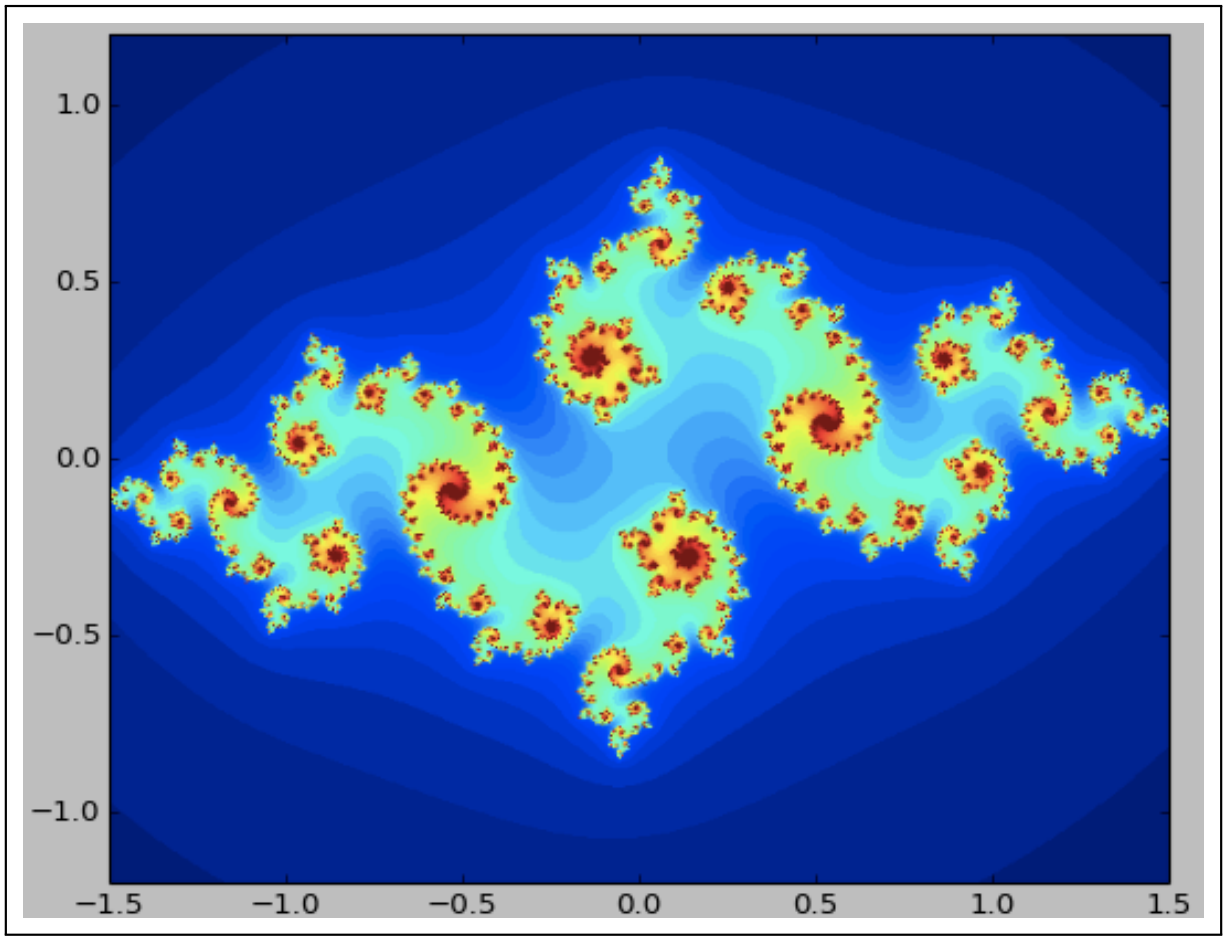
```
>>> julia(-0.2+0.8j)
```



Fractale : Ensembles de Julia

- Exemples :

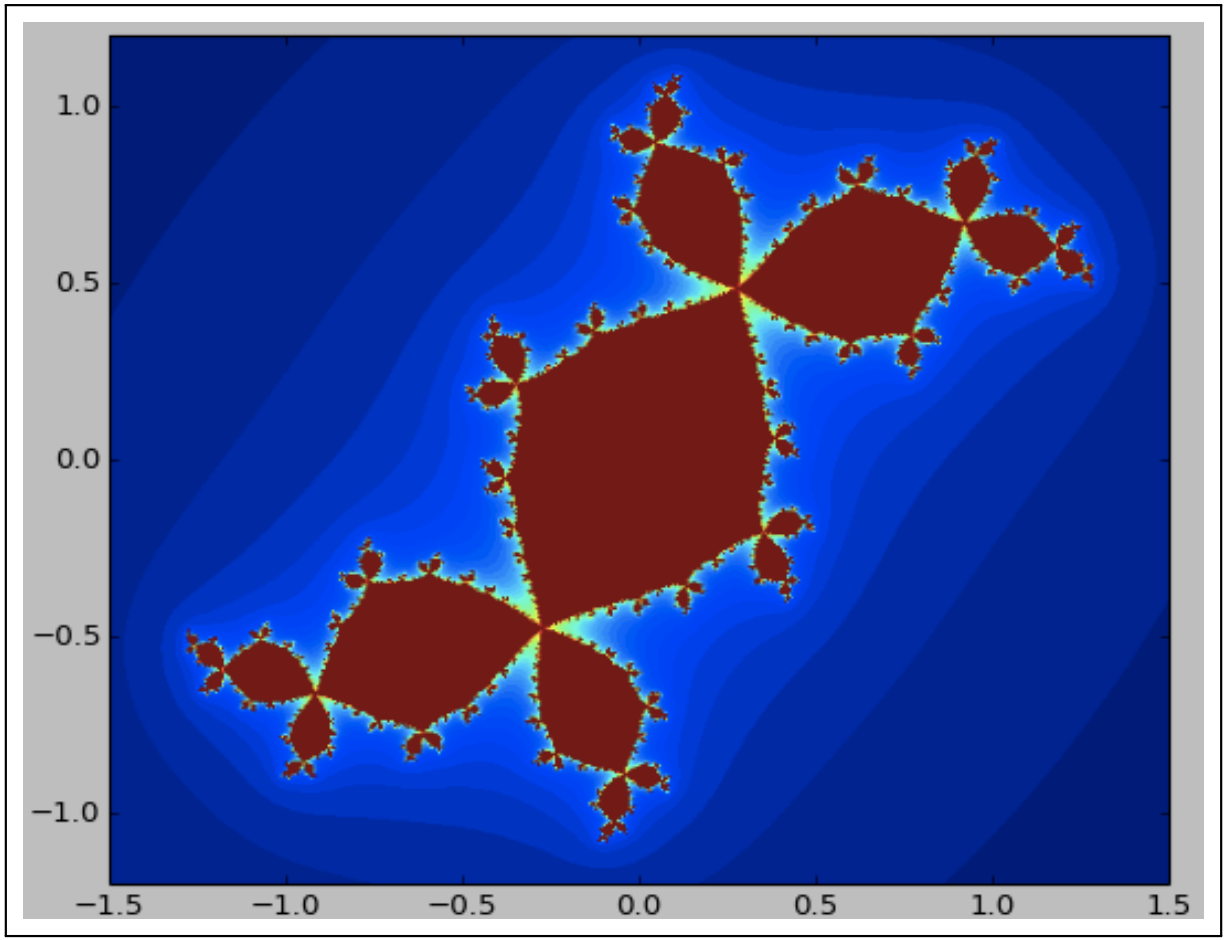
```
>>> julia(-0.8+0.2j)
```



Fractale : Ensembles de Julia

- Exemples : **Le lapin de Douady**

```
>>> julia(-0.123+0.745j)
```

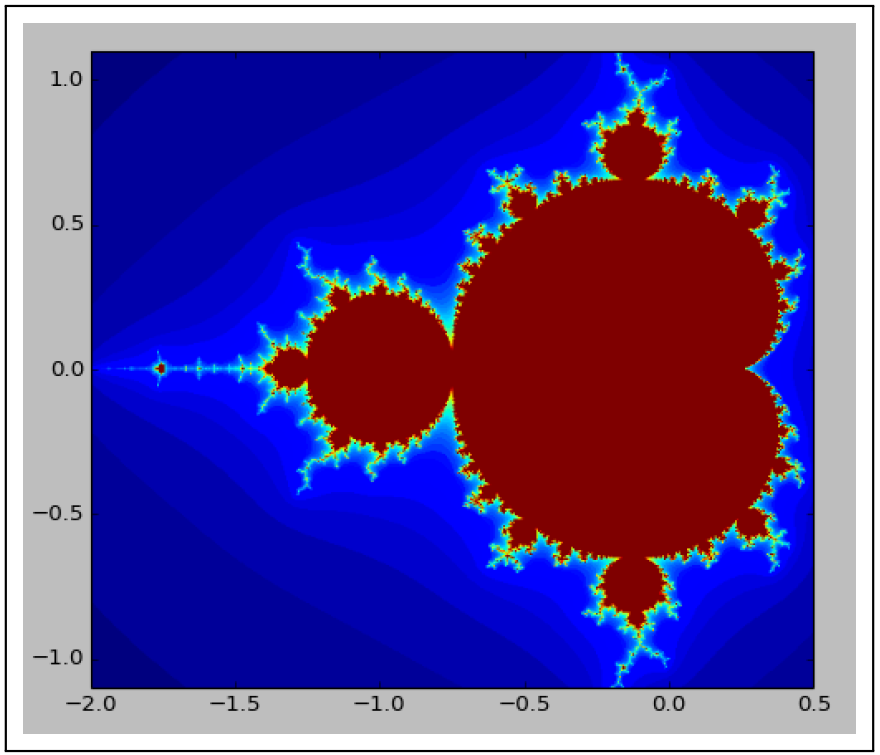


Fractale : Ensemble de Mandelbrot

• Etant donnés dun nombre complexe c et la suite complexe (z_n) définie par la relation de récurrence :

$$z_{n+1} = z_n^2 + c \qquad z_0 = 0$$

l'ensemble de Mandelbrot est la frontière (au sens topologique sur l'espace vectoriel normé $(\mathbb{C}, |.|)$) de l'ensemble des points $c \in \mathbb{C}$ pour lesquels la suite (z_n) est bornée (i.e. $\exists\, M > 0$, tel que $\forall\, n \in \mathbb{N}$, $|z_n| < M$).



Construction en TD...