Els fractals

La geometria del món

Fabio Guevara

Ins Francesc Macià

Barcelona 2024

Contents

Per què els fractals?

Per què els fractals?

El motiu pel qual vaig triar aquest tema per a mi TR va ser:

- Curiositat per aquestes formes
- Interès per les matemàtiques
- Interès per la programació

Objectius

Objectius

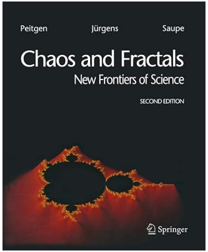
- A nivell teòric conèixer sobre el concepte de fractal i les teories matemàtiques que engloben.
- A nivell pràctic es buscava respondre les següents questions:
  - Es possible modelar en 3D un paisatge natural a partir d'un sol fractal?
  - Es podran obtenir més d'un tipus de paisatge?

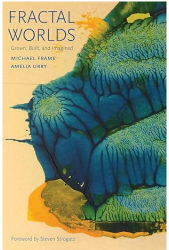
Metodologia

Metodologia

Recerca i investigació sobre els fractals i tots els conceptes que engloben. Ús de diverses fonts:

- Llibres de divulgació
- Vídeos
- Pàgines web





Metodologia

- Cerca d'un llenguatge de programació on desenvolupar (Python) i un suport de programació (Jupyter) per la pràctica.

- Elaboració de les imatges mostrades al treball en Jupyter. D'aquesta manera es va aprenent com funcionen els codis i llibreries emprats en els paisatges posteriorment.
- Després de comprendre el funcionament de l'eina, es comença l'elaboració dels paisatges naturals en 3D.

Exposició del concepte

Definició i origen

- La paraula fractal ve del llatí *fractus*, que significa trencat, fracturat o irregular
- Els fractals es defineixen com a formes geomètriques complexes, que tenen un patró que es repeteix a diferents escales i mesures.
- El primers patrons fractals coneguts datant de l'edat antiga, on eren utilitzats per decoració.
- Durant els segles XIX i XX es comença a elaborar les primeres idees i formes fractals.
- Finalment, Mandelbrot estableix el concepte i formula la geometria fractal.

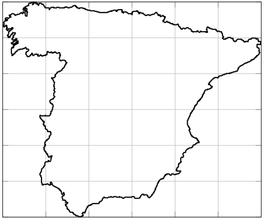
Característiques

Característiques:

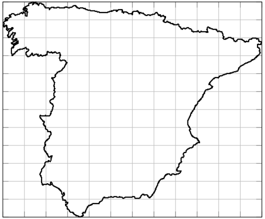
Els fractals presenten qualitats o característiques úniques, aquestes són:

- **Autosimilitud:** és la propietat de mantenir-se semblant a diverses escales.
  - Exacta
  - Aproximada
  - Estadística
- **Recursivitat:** es poden construir a partir de la repetició d'un mateix de manera indefinida.
- **Dimensió fractal:** els fractals són formes geomètriques de dimensió irracional.

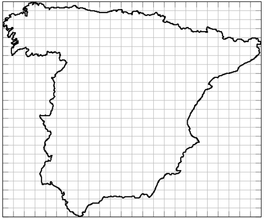
Comptatge de caixes



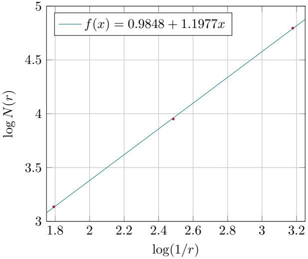
$r_1 = 1/6 \qquad N(r_1) = 23$



$r_2 = 1/12 \qquad N(r_2) = 52$



$r_3 = 1/24 \qquad N(r_3) = 121$



Dimensió d'autosimilitud

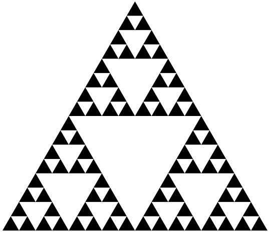
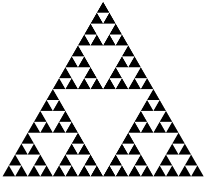
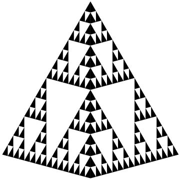
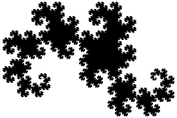
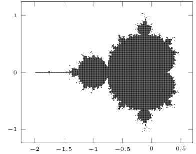
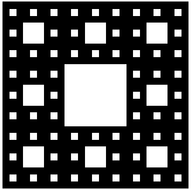
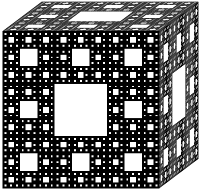
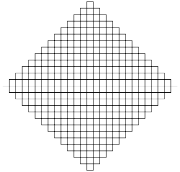
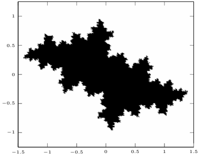


Figure 1: $D = \frac{\log(3^4)}{\log(2^4)} = 1.584962501$

Formes

Formes

Presència al món

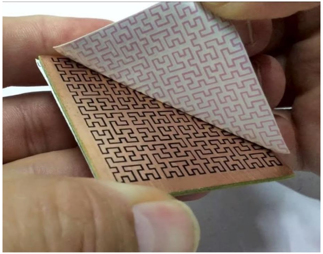
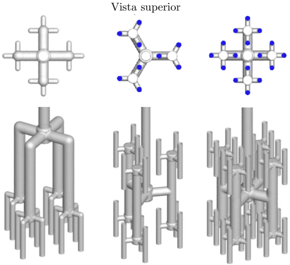
Presència al món

Els fractals es manifesten al món de dues formes:

- A la natura:



- A la ciència i tecnologia:

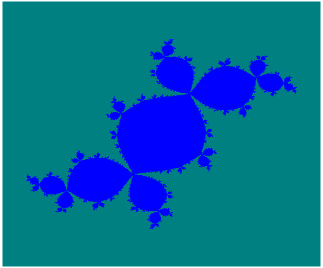
Pràctica

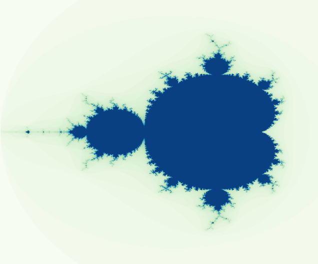
Procés

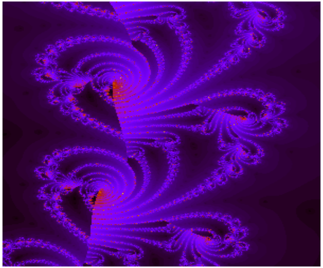
La pràctica d'aquest treball consisteix en la programació de paisatges naturals usant fractals. Per a dur a terme aquest comunicat es van seguir els següents passos:

- Cerca del llenguatge i suport de programació
- Aprenentatge. Programació de fractals bidimensionals com a assaig previ.
- Cerca de les fractals en els quals s'usessin en els paisatges.
- Construcció dels paisatges i resolució de les qüestions.

Fractals escollits







$$f_c(z) = z^2 + c.$$

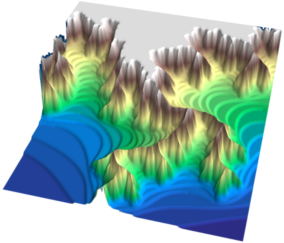
$$f(z) = z^2 + c$$

$$z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)}$$

On, $z, c \in \mathbb{C}$ i $n = 0, 1, \cdots$.

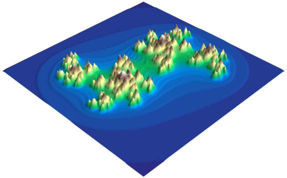
Resultats

Exposició dels paisatges obtinguts



```
def mandelbrot(c,ite):
    # Usem com a punt d'inici el nombre complex z=0+0i
    z=complex(0,0)
    # Amb el següent bucle es crea una successió de nombres complexos c(0), c(1), c(2), etc. Aquest procés s'
          executarà fins que es trobi la iteració c(k). Tal que
    # |c(k)|>2.
    for k in range(ite):
        z=z*z+c
        if abs(z)>2:
            break
    return k
# Definim una altra funció, que servirà per dibuixar regions del conjunt de Mandelbrot, utilitzant un nombre
        complex 'z' concret.
# z = nombres complexos
# dx = rang de capacitat pels valors reals, per obtenir la regió a graficar.
# dy = rang de capacitat pels valors imaginaris, per obtenir la regió a graficar.
# n = nombre de punts en la regió en els eixos x i y.
def plot_mandelbrot3D(z,dx,dy,n):
    # Construïm la regió de [real(z)-dx,real(z)+dx]X[imag(z)-dy,imag(z)+dy].
    x=np.linspace(np.real(z)-dx,np.real(z)+dx,n)
    y=np.linspace(np.imag(z)-dy,np.imag(z)+dy,n)
    # Fem una matriu de zeros de dimensió nxn.
    a=np.zeros((n,n))
    # Amb dos bucles for, construïm una malla de nombres complexos 'c' amb els valors 'x' i 'y'.
    for k in range(n):
        for j in range(n):
            c=complex(x[k],y[j])
            # Actualitzem la matriu, substituint cada zero pel valor de 'k' que retorna la funció mandelbrot
                    ().
            # c = tots el nombres complexos.
            a[k,j]=mandelbrot(c,40)
            # La matriu s'actualitzarà amb els valors de 'a' obtinguts en cada nova iteració.
    mayavi.mlab.figure(bgcolor=(1, 1, 1))
    smoothed_atlas = scipy.ndimage.gaussian_filter(a.T, 2)
    mayavi.mlab.surf(smoothed_atlas, warp_scale=5.5,colormap = 'terrain')
    mayavi.mlab.show()
```

Resultats

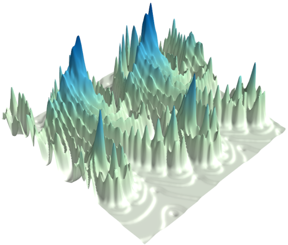


```
def julia(c,z,ite):
    for k in range(ite):
        z=z*z+c
        if abs(z)>4:
            break
            pass
        pass
    return k

# Valors de la malla.
x_min = -1.5
x_max = 1.5
y_min = -1.5
y_max = 1.5

# A continuació definim una altra funció, la qual, anomenarem plot_julia3D(). Aquesta funció ens servirà per
        crear el paisatge 3D.
def plot_julia3D():
    # Construïm dos eixos de [x_min, x_max]X[y_min, y_max]. També farem una matriu de zeros de dimensió nxn.
    x=np.linspace(x_min,x_max,n)
    y=np.linspace(y_min,y_max,n)
    a=np.zeros((n,n))
    # Amb dos bucles for, construïm una malla de nombres complexos 'z' amb els valors 'x' i 'y'.
    for i in range(n):
        for j in range(n):
            z=complex(x[i],y[j])
            a[i,j]=julia(c,z,40)
            #print(a,c)
            pass
        pass
mayavi.mlab.figure(bgcolor=(1, 1, 1))
mayavi.mlab.surf(a.T, warp_scale=2.5,colormap = 'YlGn')
smoothed_atlas = scipy.ndimage.gaussian_filter(a.T, 2)
mayavi.mlab.surf(smoothed_atlas, warp_scale=2.5,colormap = 'terrain')
mayavi.mlab.show()
```

Resultats

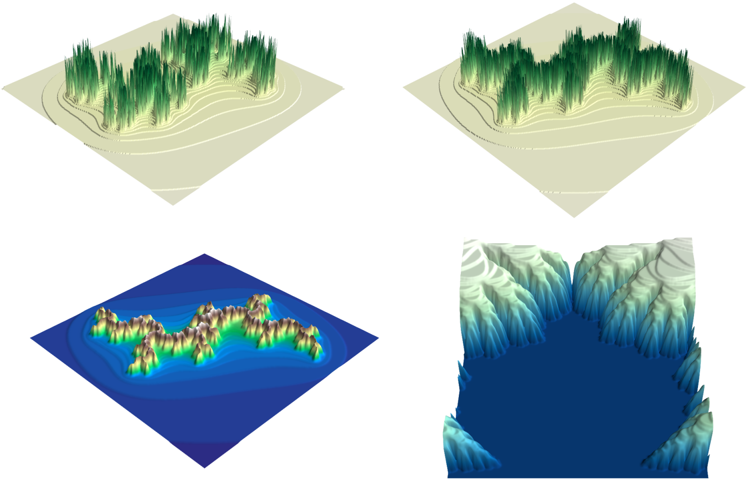


```
def newton_fractal(f,df,z0,ite):
    # tol = La tolerància: un petit valor que serveix com a valor límit, és a dir, només agafem els valors
            que siguin majors a la tolerància.
    tol = 1e-8
    z=z0
    for k in range(ite):
        dz = f(z)/df(z)
        if abs(dz)<tol:
            return k
        z = z - 1.25*dz

f=lambda z : z**(4 + 3j) - 1
df=lambda z : (4 + 3j)*z**(3 + 3j)
x_min = 0.4
x_max = 0.9
y_min = -0.9
y_max = -0.3

n_space = 300
x_values=np.linspace(x_min,x_max,n_space)
y_values=np.linspace(y_min,y_max,n_space)
n=len(x_values)
m=len(y_values)
a=np.zeros((n,m))
for k in range(n):
    for j in range(m):
        z=complex(x_values[k],y_values[j])
        # Actualitzem la matriu, substituint cada zero pel valor del mòdul de 'z' que retorna la funció
                newton_fractal().
        # z = tots el nombres complexos.
        a[k,j]=newton_fractal(f,df,z,200)
        # La matriu s'actualitzarà amb els valors de 'a' obtinguts en cada nova iteració.
        pass
    pass
mayavi.mlab.figure(bgcolor=(1, 1, 1))
smoothed_atlas = scipy.ndimage.gaussian_filter(a.T, 1.5)
mayavi.mlab.surf(smoothed_atlas, warp_scale=2.5,colormap = 'GnBu')
mayavi.mlab.show()
```

Resultats



Conclusions:

1. S'ha obtingut un ampli coneixement sobre els fractals.
2. S'ha après a utilitzar un llenguatge de programació (Python).
3. Respecte a les qüestions inicials:
    1. Es poden construir paisatges a partir d'un conjunt de julia. També a partir de seccions del conjunt de Mandelbrot i la fractal de Newton.
    2. Amb els conjunts de Julia i Mandelbrot es poden construir diversos paisatges. La fractal de Newton només permet muntanyes nevades.