Trabajos de Fractales Grupo AM 51


```
Autores:
    - Diégo Stéphan Jeandon Rodríguez
    - Alejandro Lopez Gomez
    - Marcos Laso Garcia
    - Jorge Cela Fernandez
    - Óscar Marqués Juan
```

Indice:

¿Que es un fractal?

Un fractal es un patrón geométrico que se repite a diferentes escalas y que tiene una estructura similar a la del todo. Los fractales suelen tener formas complejas y detalladas, y a menudo se encuentran en la naturaleza.

Por ejemplo, un árbol es un ejemplo de un fractal, ya que sus ramas se dividen en ramitas más pequeñas, que a su vez se dividen en ramillas todavía más pequeñas, y así sucesivamente. La estructura del árbol se repite a diferentes escalas, desde las ramas más grandes hasta las ramillas más pequeñas.

En matemáticas, los fractales se pueden definir y generar mediante fórmulas matemáticas y algoritmos. También se pueden dibujar y visualizar mediante el uso de ordenadores y gráficos. Los fractales tienen aplicaciones en diversas áreas, incluyendo la ciencia, la tecnología, el arte y la estadística.

¿Cuáles son las principales aplicaciones de los fractales en el día a día?

Los fractales tienen muchas aplicaciones en la vida diaria, aunque a menudo no nos damos cuenta de ello. Algunas de las principales aplicaciones de los fractales son:

1. Diseño gráfico: Los fractales se utilizan a menudo en el diseño gráfico y el arte para crear imágenes y patrones complejos y atractivos.

2. Análisis de datos: Los fractales se utilizan en el análisis de datos y la estadística para entender la estructura y el comportamiento de grandes conjuntos de datos.

3. Telecomunicaciones: Los fractales se utilizan en el diseño de antenas y sistemas de comunicación para mejorar la eficiencia y la capacidad de transmisión de señales.

4. Medicina: Los fractales se utilizan en la medicina para analizar la estructura de los tejidos y los órganos del cuerpo humano y para entender mejor cómo funcionan.

5. Ingeniería: Los fractales se utilizan en la ingeniería para diseñar estructuras y sistemas más eficientes y resistentes.

6. Ecología: Los fractales se utilizan en la ecología para entender cómo los ecosistemas funcionan y cómo se relacionan las diferentes partes de un ecosistema.

7. Geografía: Los fractales se utilizan en la geografía para entender la estructura y la forma de la Tierra y para analizar la distribución de los recursos naturales.

8. Meteorología: Los fractales se utilizan en la meteorología para analizar y predecir el tiempo y para entender cómo se forman los fenómenos meteorológicos.

¿Cuáles son los principales métodos matemáticos para calcular fractales?

Existen varios métodos matemáticos para calcular y generar fractales. Algunos de los principales métodos son:

1. Método de la iteración: Este método consiste en aplicar una fórmula matemática a un valor inicial y luego aplicar la misma fórmula al resultado obtenido, y así sucesivamente. Esto permite generar una serie de valores que se pueden dibujar para visualizar el fractal.

2. Método de la recursión: Este método se basa en la repetición de un proceso a diferentes niveles de detalle. Por ejemplo, para dibujar un árbol fractal, se pueden dibujar primero las ramas más grandes y luego dibujar ramas más pequeñas en cada una de ellas, y así sucesivamente.

3. Método de la transformación geométrica: Este método se basa en la aplicación de transformaciones geométricas, como traslaciones, rotaciones y escalamientos, a una figura básica

para generar un fractal.

4. Método de la serie de números complejos: Este método se basa en la generación de una serie de números complejos y en la representación gráfica de los mismos. Los números complejos se pueden usar para representar puntos en el plano complejo, y dibujar los puntos de esta manera permite visualizar el fractal.

Estos son algunos de los principales métodos matemáticos para calcular y generar fractales, aunque hay muchos otros métodos y algoritmos disponibles.

Métodos de calculo de fractales

Método de Newton

El método de Newton es un método matemático que se utiliza para calcular las raíces de una función matemática. Este método se basa en la idea de que, si conocemos una aproximación inicial de una raíz de una función, podemos calcular una aproximación mejor utilizando la derivada de la función.

El método de Newton se puede utilizar para calcular fractales de la siguiente manera:

1. Se elige una función matemática para generar el fractal. Por ejemplo, se puede utilizar una función polinómica o una función trigonométrica.

2. Se elige un punto inicial en el plano complejo, que será la primera aproximación de una raíz de la función.

3. Se calcula la derivada de la función en el punto inicial.

4. Se utiliza la fórmula del método de Newton para calcular una nueva aproximación de la raíz de la función.

5. Se repite el proceso utilizando la nueva aproximación como punto inicial, hasta que se alcance la precisión deseada.

Al dibujar los puntos en el plano complejo en los que se encuentran las raíces de la función, se puede visualizar el fractal generado por el método de Newton. Este método se puede utilizar para generar una gran variedad de fractales, incluyendo los conocidos como "conjuntos de Julia" y "conjuntos de Mandelbrot".

Ejemplo de código Python que genera fractales según el Método de Newton

```
import matplotlib.pyplot as plt

def fractal_newton(f, df, x0, n_iter=100):
  """
```

```
    Genera un fractal utilizando el método de Newton.

    Parámetros:
    - f: La función polinómica para la cual se quieren calcular las raíces.
    - df: La derivada de la función f.
    - x0: Punto inicial para el cálculo de la raíz.
    - n_iter: Número de iteraciones para el cálculo de la raíz.

    Devuelve:
    - Una lista con las raíces encontradas por el método de Newton.
    """
    roots = []
    for i in range(n_iter):
        x = x0 - f(x0) / df(x0)
        if abs(x - x0) < 1e-10:  # Criterio de convergencia
            roots.append(x)
            break
        x0 = x
    return roots

# Ejemplo de uso de la función:

# Definimos la función y su derivada
def f(x):
    return x**3 - 1

def df(x):
    return 3*x**2

# Calculamos las raíces de la función con el método de Newton
roots = fractal_newton(f, df, x0=1.0)

# Dibujamos los puntos en el plano complejo en los que se encuentran las raíces
plt.scatter([x.real for x in roots], [x.imag for x in roots])
plt.show()
```

En este ejemplo, la función `fractal_newton` toma como parámetros la función polinómica `f`, su derivada `df`, el punto inicial `x0` y el número de iteraciones `n_iter`. La función devuelve una lista con las raíces encontradas por el método de Newton.

Para visualizar los puntos en el plano complejo en los que se encuentran las raíces, utilizamos la función `scatter` de Matplotlib.

Método de Halley

El método de Halley es un método matemático que se utiliza para calcular las raíces de una función matemática. Este método se basa en la idea de que, si conocemos una aproximación inicial de una raíz de una función, podemos calcular una aproximación mejor utilizando tanto la derivada de la función como la segunda derivada de la función.

El método de Halley se puede utilizar para calcular fractales de la siguiente manera:

1. Se elige una función matemática para generar el fractal. Por ejemplo, se puede utilizar una función polinómica o una función trigonométrica.
2. Se elige un punto inicial en el plano complejo, que será la primera aproximación de una raíz de la función.
3. Se calculan la derivada y la segunda derivada de la función en el punto inicial.
4. Se utiliza la fórmula del método de Halley para calcular una nueva aproximación de la raíz de la función.
5. Se repite el proceso utilizando la nueva aproximación como punto inicial, hasta que se alcance la precisión deseada.

Al dibujar los puntos en el plano complejo en los que se encuentran las raíces de la función, se puede visualizar el fractal generado por el método de Halley. Este método se puede utilizar para generar una gran variedad de fractales, aunque es menos común que el método de Newton.

Ejemplo de código Python que genera fractales según el Método de Halley

```
import matplotlib.pyplot as plt

def fractal_halley(f, df, ddf, x0, n_iter=100):
  """
  Genera un fractal utilizando el método de Halley.

  Parámetros:
  - f: La función polinómica para la cual se quieren calcular las raíces.
  - df: La derivada de la función f.
  - ddf: La segunda derivada de la función f.
  - x0: Punto inicial para el cálculo de la raíz.
  - n_iter: Número de iteraciones para el cálculo de la raíz.

  Devuelve:
  - Una lista con las raíces encontradas por el método de Halley.
  """
  roots = []
  for i in range(n_iter):
    x = x0 - (2*f(x0)*df(x0)) / (2*df(x0)**2 - f(x0)*ddf(x0))
    if abs(x - x0) < 1e-10:  # Criterio de convergencia
      roots.append(x)
      break
    x0 = x
  return roots

# Ejemplo de uso de la función:

# Definimos la función y sus derivadas
```

```python
def f(x):
  return x**3 - 1

def df(x):
  return 3*x**2

def ddf(x):
  return 6*x

# Calculamos las raíces de la función con el método de Halley
roots = fractal_halley(f, df, ddf, x0=1.0)

# Dibujamos los puntos en el plano complejo en los que se encuentran las raíces
plt.scatter([x.real for x in roots], [x.imag for x in roots])
plt.show()
```

En este ejemplo, la función `fractal_halley` toma como parámetros la función polinómica `f`, su derivada `df` y su segunda derivada `ddf`, el punto inicial `x0` y el número de iteraciones `n_iter`. La función devuelve una lista con las raíces encontradas por el método de Halley.

Para visualizar los puntos en el plano complejo en los que se encuentran las raíces, utilizamos la función `scatter` de Matplotlib.

Método de Chebyshev

El método de Chebyshev es un método matemático que se utiliza para calcular las raíces de una función polinómica. Este método se basa en la idea de que, si se conocen los valores máximo y mínimo de una función en un intervalo cerrado, es posible calcular una aproximación de las raíces de la función en ese intervalo.

El método de Chebyshev se puede utilizar para calcular fractales de la siguiente manera:

1. Se elige una función polinómica para generar el fractal.

2. Se elige un intervalo cerrado en el que se sabe que existen raíces de la función.

3. Se calculan los valores máximo y mínimo de la función en el intervalo seleccionado.

4. Se utiliza la fórmula del método de Chebyshev para calcular una aproximación de las raíces de la función en el intervalo.

5. Se repite el proceso para otros intervalos, hasta que se alcance la precisión deseada.

Ejemplo de código Python que genera fractales según el Método de Chebyshev

```python
import numpy as np

def chebyshev_fractal(n, a, b, func):
  """
  Calcula un fractal utilizando el método de Chebyshev y una función polinómica dada.

  Parameters:
  n (int): grado del polinomio de Chebyshev.
  a (float): límite inferior del intervalo de aproximación.
  b (float): límite superior del intervalo de aproximación.
  func (function): función polinómica que se va a aproximar.

  Returns:
  np.array: valores del fractal calculado.
  """
  # Generamos los nodos de Chebyshev
  x = np.cos((2 * np.arange(n) + 1) / (2 * n) * np.pi)
  # Calculamos los coeficientes del polinomio de Chebyshev
  c = np.polynomial.chebyshev.chebfit(x, func(x), n)
  # Generamos una malla de puntos para evaluar el polinomio
  x_eval = np.linspace(a, b, 1000)
  # Evaluamos el polinomio en la malla de puntos
  y_eval = np.polynomial.chebyshev.chebval(x_eval, c)

  return y_eval
```

En este ejemplo, la función `chebyshev_fractal` toma como parámetros el grado `n` del polinomio de Chebyshev, los límites `a` y `b` del intervalo de aproximación y la función polinómica `func` que se va a aproximar. La función devuelve una matriz `numpy` con los valores del fractal calculado.

Para utilizar esta función, simplemente se necesita proporcionar los parámetros adecuados y luego llamar a la función. Por ejemplo, para calcular un fractal utilizando un polinomio de Chebyshev de grado 5 y una función polinómica dada, se podría hacer lo siguiente:

```python
import numpy as np

# Define la función polinómica que queremos aproximar
def func(x):
  return x**2 + 2*x + 1

# Calcula el fractal utilizando el método de Chebyshev
fractal = chebyshev_fractal(5, -1, 1, func)
```

¿Cual de los 3 métodos vistos se considera mejor, y porque?

Es difícil determinar qué método es el "mejor" para calcular fractales, ya que depende de muchos factores, como la complejidad de la función que se está evaluando, el intervalo de aproximación y la precisión deseada.

El método de Newton es un método iterativo que se utiliza para encontrar raíces de una función. Es un método muy eficiente para encontrar raíces cercanas a un punto inicial dado, pero puede tener problemas de convergencia si el punto inicial está muy lejos de la raíz real. Además, el método de Newton requiere la evaluación de la función y de su derivada, lo que puede ser costoso computacionalmente para algunas funciones complejas.

El método de Halley es un método iterativo que se utiliza para encontrar raíces de una función. Es una variante del método de Newton que utiliza la información de la segunda derivada de la función para mejorar la precisión y la velocidad de convergencia. Al igual que el método de Newton, el método de Halley puede tener problemas de convergencia si el punto inicial está muy lejos de la raíz real y requiere la evaluación de la función y de sus dos primeras derivadas, lo que puede ser costoso computacionalmente para algunas funciones complejas.

El método de Chebyshev es un enfoque que se utiliza para aproximar funciones matemáticas con una precisión determinada utilizando polinomios de Chebyshev. Estos polinomios son un conjunto especial de polinomios que se pueden usar para aproximar funciones en un intervalo determinado con una precisión deseada. El método de Chebyshev es útil para aproximar funciones en un intervalo dado y puede ser más eficiente computacionalmente que los métodos de Newton y Halley para algunas funciones complejas.

En general, es difícil decir cuál de estos métodos es el "mejor" para calcular fractales, ya que depende de muchos factores y de la aplicación específica. Es posible que uno de estos métodos sea más adecuado para algunas situaciones y otro método sea más adecuado para otras. Es importante evaluar cada método y elegir el que mejor se ajuste a las necesidades de la aplicación en cuestión.