

ESCUELA SUPERIOR DE CÓMPUTO

Matematicas avanzadas para la ingenieria

Raices n-esimas de un complejo

 $por: \\ L\'{o}pez \; Manr\'iquez \; \'{A}ngel$

profesor Olvera Andana Miguel

1. Introduccion

Para hacer del programa que grafique las raices de un complejo z, se hara uso de los siguientes temas.

Teorema 1.1. Raices n-esimas de un complejo Sea z = (a, b), entonces $z^{\frac{1}{n}}$ tiene exactamente n raices, de la forma

$$w_k = |z|^{1/n} cis(\frac{\theta + 2k\pi}{n}) \quad k \in \{0, 1, ..., n - 1\}$$

2. Planteamiento del problema

Hacer un programa en cualquier lenguaje que grafique la raiz n-esima de un complejo.

3. Diseño e implementacion de la solucion

Para dar solucion al problema, usamos el teorema de la raiz n-esima de la unidad como se muestra en el siguiente codigo

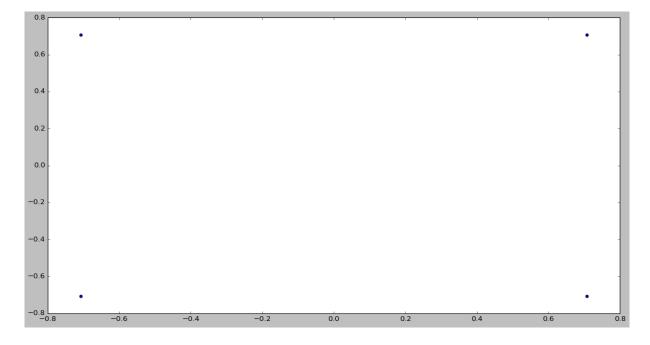
```
# Programa que grafica las raices n-esimas de un complejo
# y las muestra en el plano complejo
# Importamos las bibliotecas para graficar en el plano
import numpy as np
import matplotlib.pyplot as plt
# Para el obtener la arcotangente de un angulo
import math
# Obtenemos la parte real e imaginaria para z = a + bi
a = raw_input ("re {z} = ")
a = float(a)
b = raw_input ("im {z} = ")
b = float(b)
n = raw_{input} ("n (z ^ (1 / n)) = ")
# Obtenemos a n, para la raiz n-esima
n = int(n)
# Hallamos el modulo de z
r = math.sqrt (a ** 2 + b ** 2)
# Hallamos el angulo formado por el complejo con respecto
# al eje real
theta = math.atan2 (b, a)
# Le calculamos la raiz n-esima a |z|
r = r ** (1 / n)
# Declaramos dos listas para guardar las coordenadas de
# los puntos
x = []
y = []
```

```
# Obtenemos los n - 1 angulos, posteriormente agregamos
# a las listas las coordenadas rectangulares del complejo
for k in range (0, n):
    phi = float ((2 * k * math.pi + theta) / n)
        x.append (r * math.cos (phi))
        y.append (r * math.sin (phi))
# Graficamos los puntos
plt.scatter (x, y)
plt.show ()
```

4. Funcionamiento

```
Para (-1,0), n=4
```

```
angel@lambda: roots » make [22:03:40]
python root.py
re {z} = -1
im {z} = 0
n (z ^ (1 / n)) = 4
```



Para (3,4), n=32

```
angel@lambda: roots >> make [22:32:04]
python root.py
re {z} = 3
im {z} = 4
n (z ^ (1 / n)) = 32
```

