Función-atan2

December 17, 2024

Autor: David Alejandro Puga Novoa

Materia: Métodos Numéricos

Fecha de entrega: 17/12/2024

1 Introducción breve sobre la función atan2

La función `atan2(y, x)` calcula el arctangente de dos variables (y, x) de forma que devuelve el ángulo en radianes entre el eje X positivo y el punto (x, y).

A diferencia de la función atan, `atan2` tiene en cuenta los signos de y y x para determinar el cuadrante correcto del ángulo.

1.1 Ejemplo básico de atan2

[4]:
```
import math
x1, y1 = 1, 1
angulo_1 = math.atan2(y1, x1)
print(f"Punto (1, 1): Ángulo = {angulo_1} radianes ({math.degrees(angulo_1)}␣
↪grados)")
```

```
Punto (1, 1): Ángulo = 0.7853981633974483 radianes (45.0 grados)
```

2 ¿Por qué se recomienda usar la función atan2?

La función atan2 se recomienda porque evita la ambigüedad que puede surgir cuando se usan los signos de x e y por separado. Al calcular la tangente inversa usando atan, solo se obtiene un ángulo en el primer o cuarto cuadrante.

Sin embargo, atan2 devuelve el ángulo correcto en los cuatro cuadrantes.

Además: - atan2 maneja casos en los que x es igual a 0 sin errores, retornando valores adecuados como pi/2 o -pi/2.

2.1 Ejemplo práctico

[5]:
```
# Caso con x = 0
x2, y2 = 0, 1
angulo_2 = math.atan2(y2, x2)
print(f"Punto (0, 1): Ángulo = {angulo_2} radianes ({math.degrees(angulo_2)}␣
 ↪grados)")
```

```
Punto (0, 1): Ángulo = 1.5707963267948966 radianes (90.0 grados)
```

3 Diferencias entre atan y atan2

- `atan(y/x)`: Solo considera la relación entre y y x, lo que puede llevar a ángulos incorrectos si x o y son negativos.
- `atan2(y, x)`: Tiene en cuenta los signos de x y y para devolver el ángulo exacto en los cuatro cuadrantes.

Ejemplo: Si x=-1 y y=1: - atan(y/x) = atan(-1) -> devuelve un valor en el cuarto cuadrante. - atan2(y, x) = atan2(1, -1) -> devuelve un valor en el segundo cuadrante, que es correcto.

3.1 Ejemplo comparativo

[6]:
```
# Comparación entre atan y atan2
x3, y3 = -1, 1
atan_result = math.atan(y3 / x3)
atan2_result = math.atan2(y3, x3)
print(f"atan(y/x): {atan_result} radianes ({math.degrees(atan_result)} grados)")
print(f"atan2(y, x): {atan2_result} radianes ({math.degrees(atan2_result)}␣
 ↪grados)")
```

```
atan(y/x): -0.7853981633974483 radianes (-45.0 grados)
atan2(y, x): 2.356194490192345 radianes (135.0 grados)
```

4 Ejemplos prácticos adicionales de la función atan2 en Python

A continuación, se presentan varios ejemplos que ilustran el uso de atan2 en diferentes cuadrantes:

4.0.1 Ejemplo 1: Segundo cuadrante

[8]:
```
x4, y4 = -1, 1
angulo_4 = math.atan2(y4, x4)
print(f"Punto (-1, 1): Ángulo = {angulo_4} radianes ({math.degrees(angulo_4)}␣
 ↪grados)")
```

```
Punto (-1, 1): Ángulo = 2.356194490192345 radianes (135.0 grados)
```

4.0.2 Ejemplo 2: Tercer cuadrante

[9]:
```
x5, y5 = -1, -1
angulo_5 = math.atan2(y5, x5)
print(f"Punto (-1, -1): Ángulo = {angulo_5} radianes ({math.degrees(angulo_5)}␣
 ↪grados)")
```

```
Punto (-1, -1): Ángulo = -2.356194490192345 radianes (-135.0 grados)
```

4.0.3 Ejemplo 3: Cuarto cuadrante

[10]:
```
x6, y6 = 1, -1
angulo_6 = math.atan2(y6, x6)
print(f"Punto (1, -1): Ángulo = {angulo_6} radianes ({math.degrees(angulo_6)}␣
 ↪grados)")
```

```
Punto (1, -1): Ángulo = -0.7853981633974483 radianes (-45.0 grados)
```

5 Visualización de los ángulos generados

Podemos visualizar los puntos y sus ángulos en un plano cartesiano para comprender mejor cómo atan2 calcula los ángulos.

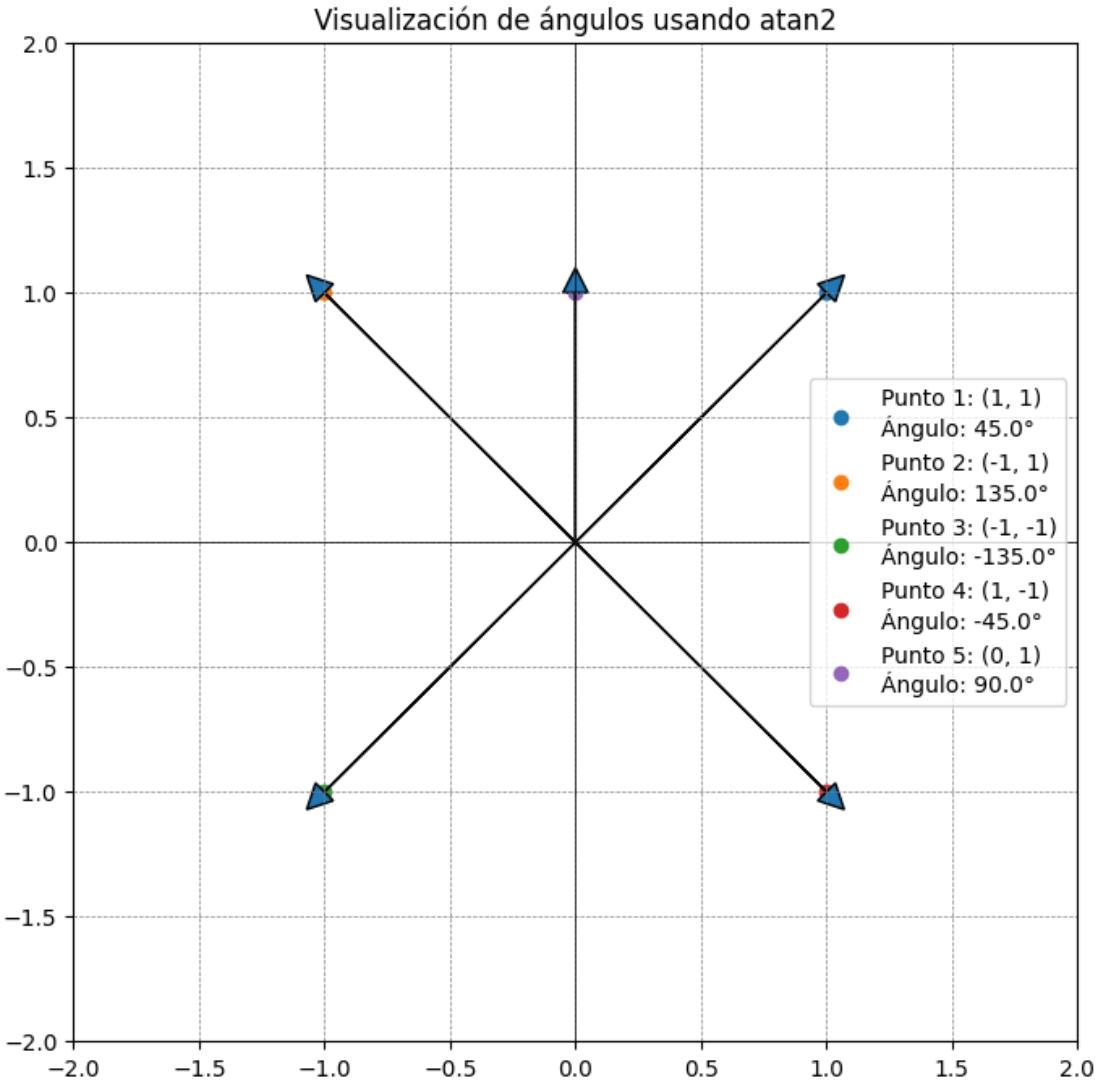
[11]:
```
import matplotlib.pyplot as plt

# Crear puntos de prueba
ejes_x = [1, -1, -1, 1, 0]
ejes_y = [1, 1, -1, -1, 1]
angulos = [math.atan2(y, x) for x, y in zip(ejes_x, ejes_y)]

# Graficar los puntos en un plano cartesiano
plt.figure(figsize=(8, 8))
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)
plt.grid(color = 'gray', linestyle = '--', linewidth = 0.5)

for i, (x, y, ang) in enumerate(zip(ejes_x, ejes_y, angulos)):
    plt.scatter(x, y, label=f"Punto {i+1}: {x, y}\nÁngulo: {round(math.
 ↪degrees(ang), 2)}°")
    plt.arrow(0, 0, x, y, head_width=0.1, head_length=0.1)

plt.xlim(-2, 2)
plt.ylim(-2, 2)
plt.title("Visualización de ángulos usando atan2")
plt.legend()
plt.show()
```



6 Conclusión

La función atan2 es preferible a atan porque proporciona resultados correctos en los cuatro cuadrantes y evita errores cuando x = 0.

Y como se pudo reflejar en los ejemplos y visualizaciones el cómo atan2 calcula los ángulos con precisión.