迴圈與判斷式應用範例

蔡尚融

2018-03-19

二分法求根（Bisection method）

- 求方程式 $f(x) = 0$的根（root），藉由反覆使用中間值定理的前提與結果。
- 中間值定理（Intermediate value theorem）
  Let $f : [a, b] \to \mathbb{R}$ be a continuous function. If u is between $f(a)$ and $f(b)$, then there is a $c \in (a, b)$ such that $f(c) = u$.
- 考慮函數 $f(x)$ 在 $[a, b]$ 是連續的，且 $f(a)$ 與 $f(b)$ 是異號，根據中間值定理存在根 c 使得 $f(c) = 0$。
- 將區間 $[a, b]$ 由中點 c 分割為兩子區間 $[a, c]$ 與 $[c, b]$，考慮滿足中間值定理前提的子區間，作為尋找根的區間。

範例：

令 $f(x) = x^3 - 8x^2 + 5x + 1$；用二分法求方程式 $f(x) = 0$ 在 $[0, 1]$ 區間的近似根，其殘值絕對值小於 10^{-5}，並限制疊代次數不超過 20 次。（殘值（residual）的為 $r(x) = 0 - f(x) = f(x)$。）

1. 實作函式，求函數 $f(x)$ 值，初始區間與限制條件付值。
2. 檢查區間是否滿足中間值定理。
3. 重覆計算中點，並尋找符合兩端點異號的子區間，直到殘值絕對值小於要求，或達到最大疊代次數。

```
#!/usr/bin/env python
import math

def fn(x):
    return x ** 3 - 8.0 * x ** 2 + 5.0 * x + 1.0

# Initial interval [ak,bk] = [0, 1.0]
ak = 0
bk = 1.0

tol_iter = 10
print("Maximum iterations:", tol_iter)
tol_nres = 1.0E-5
print("Nonlinear residual:", tol_nres)
```

檢查起始區間是否中間值定理要求，須使用數學函式庫中 `copysign()` 函式，取得 $f(a_0)$ 與 $f(b_0)$ 正負號並比較。

起始區間：$[a_0, b_0] = [0, 1]$；

$$f(a_0) = f(0) = 1$$
$$f(b_0) = f(1) = -1$$

```
sign_a = math.copysign(1.0, fn(ak))
sign_b = math.copysign(1.0, fn(bk))
if sign_a == sign_b:
    print("Bisection mothed can not do.")
    exit()
```

計算中點：$c_0 = \frac{a_0+b_0}{2} = \frac{0+1}{2} = 0.5$；$f(c_0) = f(0.5) = 1.625$

```
ck = ak + delta / 2.0
```

迴圈疊代

1. 若函數絕對值在中點小於要求，或疊代達到最大次數，則停止二分法疊代。
2. 尋找符合中間值定理前提子區間：$[0, 0.5]$ 或 $[0.5, 1]$

x	$a_0 = 0$	$c_0 = 0.5$	$b_0 = 1$
$f(x)$	1	1.625	-1

用子區間代換原區間：$[a_1, b_1] = [0.5, 1]$。

```
i = 0
while math.fabs(fn(ck)) > tol_nres and i < tol_iter:
    i = i + 1
    ck = ak + (bk - ak) / 2.0
    print(i, "Nonlinear residual:", fn(ck))
    sign_a = math.copysign(1.0, fn(ak))
    sign_c = math.copysign(1.0, fn(ck))
    if sign_c == sign_a:
        ak = ck
    else:
        bk = ck

print("Approximate root:", ck)
print("Number of iterations:", i)
```

牛頓法求根（Newton method）

- 假設函數 $f(x)$ 是連續，且二階導函數存在。
- 使用函數 $f(x)$ 的泰勒級數（Taylor series）至一次微分項來尋找方程 $f(x) = 0$ 的根。

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \dots$$

- 假設 x^* 為根，考慮 $0 = f(x^*) = f(x_0) + f'(x_0)(x^* - x_0)$ 求 x^*。

$$x^* = x_0 - \frac{f(x_0)}{f'(x_0)}$$

- 牛頓疊代公式為：$x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$，其中 $f'(x_n) \neq 0$。

範例：

令 $f(x) = x^3 - 8x^2 + 5x + 1$；用牛頓法求方程式 $f(x) = 0$ 在 $[0, 1]$ 區間的近似根，其殘值絕對值小於 10^{-5}，並限制疊代次數不超過 10 次；嘗試初始值為 0 或 1。

1. 實作函式，求函數 $f(x)$ 值與一階導函數 $f'(x)$ 值。
2. 用迴圈執行牛頓疊代公式，直到殘值絕對值小於要求，或達到最大疊代次數。

```python
#!/usr/bin/env python
import math

def fn(x):
    return x ** 3 - 8.0 * x ** 2 + 5.0 * x + 1.0

def df(x):
    return 3.0 * x ** 2 - 16.0 * x + 5.0

tol_iter = 10
print("Maximum iterations:", tol_iter)
tol_nres = 1.0E-5
print("Nonlinear residual:", tol_nres)
```

起始值

```
xn = 0  # Or set xn = 1
```

牛頓疊代迴圈

```
i = 0
while math.fabs(fn(xn)) > tol_nres and i < tol_iter:
    i = i + 1
    xn = xn - fn(xn) / df(xn)
    print(i, "Nonlinear residual:", fn(xn))

print("Approximate root:", xn)
print("Number of iterations:", i)
```

隨堂練習：

求 $\sqrt{\frac{117}{123}} \approx 0.9753$ 的值，考慮 $f(x) = x^2 - \frac{117}{123}$；使用下列數值方法。

1. 二分法。
2. 牛頓法。