數值分析

Numerical Analysis

108 學年度　第一次學習成果報告

學 號 ： S07240018

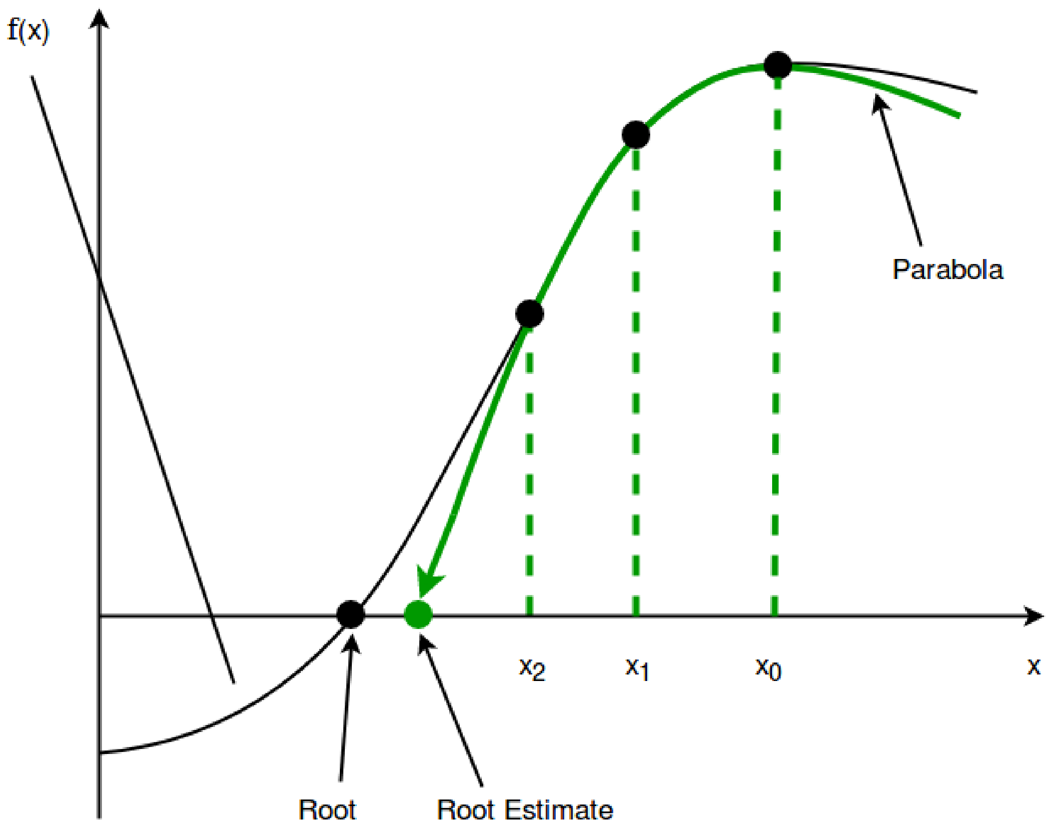
姓 名 ： 溫宏岳

主題	使用Müller Method處理Omar Khayyam古典問題之研究與探討
講評	
評分	
	☐ 需大幅修改　☐ 少許修正　☐ 無需修改

I. 前提：Müller Method 是什麼？

Müller Method 是一種求根的算法，是一種求解形式為 f (x) = 0 的方程的數值方法。它由 David E. Muller 於 1956 年首次提出。

它是從對根的三個初始假設開始，然後通過這三個點構造一個拋物線，接著將 x 軸與拋物線的交點作為下一個近似值。這個過程一直持續到找到具有所需精度水平的根為止。



II. 過程：算法及其運作方式(Python)

```
import matplotlib.pyplot as plt
import numpy as np
import math

MAX_ITERATIONS = 10000

# 設 f(x)
def f(x):

    #取 f(x) = x ** 3 + 2*(x ** 2) + 10*x - 20
 return (1 * pow(x, 3) + 2 * x * x + 10 * x - 20)

def Muller(a, b, c):

 res = 0
 i = 1

 while True:

       #計算各種常數
         #計算 x^3 所需
       f1 = f(a);f2 = f(b);f3 = f(c);
       print('\n',i ,". f(c) = ",f(c))
       d1 = f1 - f3
       d2 = f2 - f3

       h1 = a - c
       h2 = b - c

       a0 = f3
       a1 = (((d2 * pow(h1, 2)) -
```

```
            (d1 * pow(h2, 2))) /
            ((h1 * h2) * (h1 - h2)))
        a2 = (((d1 * h2) - (d2 * h1)) /
            ((h1 * h2) * (h1 - h2)))

        #abs 是返回()裡的絕對值的方法
        #math.sqrt 是返回()裡的平方根，對於()裡>0 的方法
        x = ((-2 * a0) / (a1 + abs(math.sqrt(a1 * a1 - 4 *
a0 * a2))))
        y = ((-2 * a0) / (a1 - abs(math.sqrt(a1 * a1 - 4 *
a0 * a2))))

        #取更接近 x^2 的根
        if x >= y:
            res = x + c
        else:
            res = y + c

        #檢查 x^3 與 x^2 的相似度直到小數點後兩位
        m = res * 100
        n = c * 100

        #math.floor 是返回不大於 m,n 的最大整數的方法
        m = math.floor(m)
        n = math.floor(n)


        if abs(res-c)<0.0001:
            break
        else:
            a = b
            b = c
            c = res
```

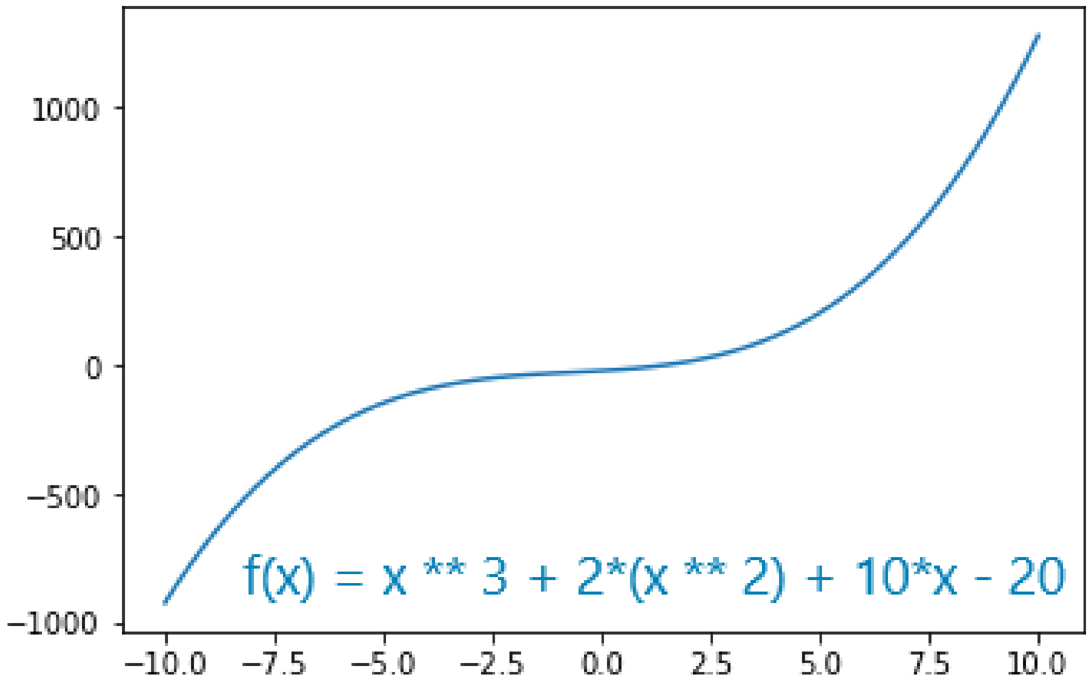
```
            print("Output : ",c)

        if i > MAX_ITERATIONS:
            print("Root cannot be found using", "Muller's method")
            break
        i += 1

    #round 是返回 res 四捨五入到小數點後 4 位的方法
    if i <= MAX_ITERATIONS:
        print('\n 第',i,"次, the value of the root is", round(res, 4))

#主程式
a = 0
b = 1
c = 2
Muller(a, b, c)

#作圖
x = np.linspace(-10,10,100)
y = x ** 3 + 2*(x ** 2) + 10*x - 20
plt.plot(x,y)
plt.show()
```

Output

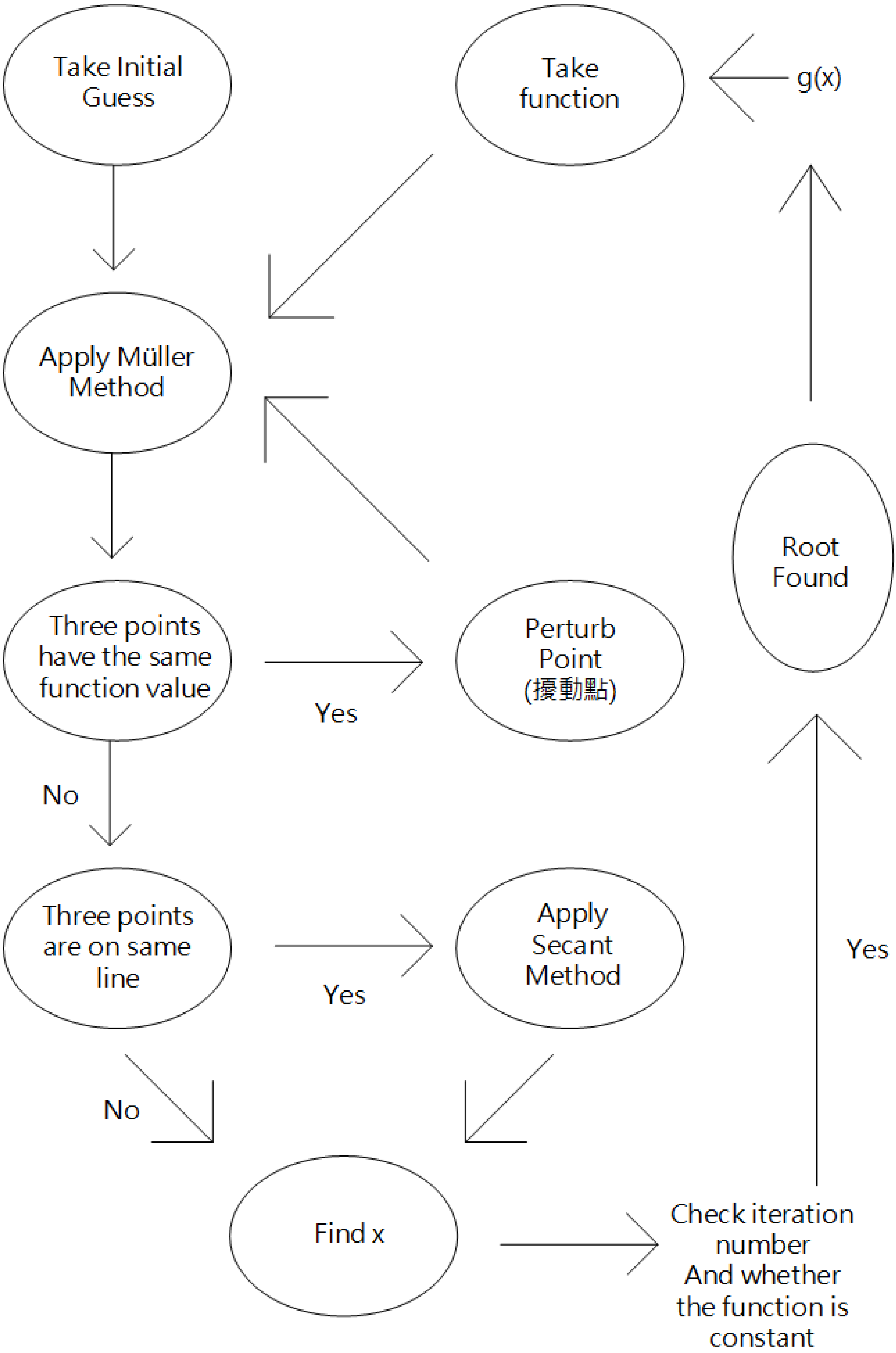
```
In [1]: runfile('D:/willy/Desktop/Numerical Analysis_ 1.py', wdir='D:/willy/Desktop')

 1 . f(c) =  16
Output :  1.3540659228538017

 2 . f(c) =  -0.30967927067320744
Output :  1.368647229785477

 3 . f(c) =  -0.0033937474211640506
Output :  1.3688080368924294

 4 . f(c) =  -1.4963268384349249e-06

第 4 次, the value of the root is 1.3688081078213805
```

因此，**Output：The value of the root is <u>1.3688081078213805</u>**

Image



Process chart


Take Initial Guess
Take function
g(x)
Apply Müller Method
Three points have the same function value
Yes
Perturb Point (擾動點)
No
Three points are on same line
Yes
Apply Secant Method
No
Find x
Check iteration number And whether the function is constant
Yes
Root Found

III. 結論：為什麼要使用 Müller Method？

Müller Method 是尋根的方法之一，同時也包括 Bisection method、Secant method 和 Newton – Raphson method。

但是，與這些方法相比，Müller Method 具有某些優點，如下所示:

1. 在 Müller Method 中，收斂速度（即，我們每一步離根的距離更近）約為 1.84，而對於 Secant method 則為 1.62，對於線性而言，即 Bisection method 為 1。

   因此，**Müller Method** 比 **Bisection method** 和 **Secant method** 還要**快**。

2. **儘管它比收斂速度為 2 的 Newton – Raphson method 要慢**，但它克服了 Newton – Raphson method 的最大缺點之一，也就是說，Müller Method 在每個步驟中不必去計算導數。

3. 可以找到假想的根。

因此，這說明 Müller Method 是計算函數根的有效方法。

但缺點也有:

1. 手算的時間會非常漫長，會有更多的錯誤空間。
2. 會發現無關的根。

接下來是介紹 Omar Khayyam 的方法。

作為數學家，他最著名的是他在三次方程的分類和求解的作法，他通過圓錐曲線的交點提供了幾何求解。Omar Khayyam 也促進了對平行公理的理解。

Khayyam 在數學界非常有名。他倖存的數學著作包括：*評述關於歐幾里得基本原理*，*關於圓象限的劃分*，以及*關於代數問題的證明*。他還寫了關於提取二項式定理的論文和自然數 n 階根。

Question:

Omar Khayyam in the base-60number system as

f(x) = 1 + 22*(x^1) + 7*(x^2) + 42*(x^3) + 33*(x^4) + 4*(x^5) + 40*(x^6), x=(1/60)

Output

```
In [1]: x=(1/60)

In [2]: 1 + 22*(pow(x,1)) + 7*pow(x,2) + 42*pow(x,3) + 33*pow(x,4) + 4*pow(x,5) + 40*pow(x,6)
Out[2]: 1.3688081078532237
```

因此，**Output2：The value of the root is <u>1.3688081078532237</u>**

由上述：

Output1：The value of the root is 1.3688081078213805 (Müller Method)

Output2：The value of the root is 1.3688081078532237 (Omar Khayyam)

由以上我們可以得知小數點後 11 位才不同，也就代表 Output1≠Output2，一定會有誤差產生。

因此估計的值一定不等於實際的值，但會在使用 Müller Method 第 4 次時，會與 Omar Khayyam 非常之接近。

參考文獻:

https://en.wikipedia.org/wiki/Muller%27s_method

https://www.codewithc.com/c-program-for-mullers-method/

https://www.codewithc.com/mullers-method-algorithm-flowchart/

http://kilyos.ee.bilkent.edu.tr/~microwave/programs/utilities/numeric1/infoMuller.htm

https://github.com/apauley/numerical-analysis/blob/master/Chapter1/Python/muller.py