

云南大学数学与统计学院 上机实践报告

课程名称：数值计算实验	年级：2015 级	上机实践成绩：
指导教师：朱娟萍	姓名：刘鹏	
上机实践名称：非线性方程求根	学号：20151910042	上机实践日期：2017-12-25
上机实践编号：No.06	组号：	最后修改时间：01:53

一、实验目的

1. 通过对所学的非线性方程求根法的理论方法进行编程，提升程序编写水平；
2. 通过对理论方法的编程实验，进一步掌握理论方法的每一个细节；

二、实验内容

1. 编制求线性方程根的程序；
2. 编程实现用埃特金法求方程的根。

三、实验平台

Windows 10 1709 Enterprise 中文版；
Python 3.6.0；
Wing IDE Professional 6.0.5-1 集成开发环境；
MATLAB R2017b win64；
AxMath 公式编辑器；
EndNote X8 文献管理。

四、实验记录与实验结果分析

1 题

用二分法求方程 $x^2 - x - 1 = 0$ 的正根，要求精确到小数点后一位。^[1]
(略)

2 题

请用埃特金方法编程求出 $x = \tan x$ 在 $x = 4.5$ 附近的根。

解答：

由于程序比较简单，所以不铺张开写，两题合并写，而且可以对比，同一道题的迭代深度。

程序代码

```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun Dec 24 21:28:46 2017
4
5  @author: Newton
6  """
7  """filename: 2.1 get root.py"""
8
```

```

9  import math
10
11  class root:
12      """This class provides some ways to find roots"""
13
14      def __init__(self, fun_name, x_left, x_right = None):
15          """fun_name represents the name of the function if the equation.
16
17          Both left and right ends will be given by x_left and x_right.
18          """
19          if x_right != None:
20              """Only support binary method."""
21
22              if fun(x_left) * fun(x_right) < 0:
23                  self.x_l = x_left
24                  self.x_r = x_right
25                  self.fun = fun_name
26                  self.method = 'binary'
27              else:
28                  raise ValueError("values on x_right and x_left should have opposite sign.")
29          else:
30              """Only support Aitken method."""
31              self.x = x_left
32              self.fun_after_convert = fun_name
33              self.method = 'Aitken'
34
35      def binary(self, e):
36          """e = (b - a) / 2"""
37          if self.method != 'binary':
38              raise ValueError("Method does not support!")
39          a = self.x_l
40          b = self.x_r
41          times = 0
42
43          while (abs(b-a)/2) > e:
44              if self.fun((a + b)/2) == 0:
45                  return (a + b)/2
46              elif self.fun(a) * self.fun((a + b)/2) < 0:
47                  b = (a + b)/2
48              else:
49                  a = (a + b)/2
50              times += 1
51          ans = ((a + b)/2, times)
52          return ans
53
54      def aitken(self, e):
55          """e = x - x0"""
56          if self.method != 'Aitken':
57              raise ValueError("Method does not support!")

```

```

58     x0 = self.x
59     x1 = self.fun_after_convert(x0)
60     x2 = self.fun_after_convert(x1)
61
62     x = x0 - (x1-x0)*(x1-x0)/(x2-2*x1+x0)
63     times = 1
64
65     while abs(x-x0) > e:
66         x0 = x
67         x1 = self.fun_after_convert(x0)
68         x2 = self.fun_after_convert(x1)
69
70         x = x0 - (x1-x0)*(x1-x0)/(x2-2*x1+x0)
71
72         times += 1
73     ans = (x, times)
74     return ans
75
76 if __name__ == "__main__":
77     def fun(x):
78         return x - math.tan(x)
79
80     def fun_after_convert(x):
81         return math.tan(x)
82
83     c = root(fun, 4.4, 4.6)
84     e = 0.00001
85     tmp = c.binary(e)
86     print('root.binary(x - tan(x) == 0) is ', tmp[0], 'iterations depth:', tmp[1])
87     d = root(fun_after_convert, 4.5)
88     tmp = d.aitken(e)
89     print('root.aitken(x - tan(x) == 0) is ', tmp[0], 'iterations depth:', tmp[1])

```

Code Box 1

运行结果

```

2.1 get root.py (pid 671) Debug process terminated
root.binary(x - tan(x) == 0) is 4.493414306640625 iterations depth: 14
root.aitken(x - tan(x) == 0) is 4.493409457909101 iterations depth: 6

```

代码分析

由于二分法与埃特金方法的函数并不是一样的，前者是原函数，后者是迭代函数，所以很难写一个通用算法解决这个迭代函数的生成问题。所以这个 class 意义不是很大，不过可以通过对属性进行赋值，重复进行计算，也算有一定的灵活性。

五、实验体会

通过编程，复习了简单迭代法及其改进。明白了二分法与埃特金法的斯坦弗森过程之间的区别。

六、参考文献

[1] 金一庆, 陈越, 王冬梅. 数值方法[M]. 北京: 机械工业出版社; 2000.2.