目录

[第一次作业 1](#_Toc129520792)

[第二次作业 2](#_Toc129520793)

[第三次作业 5](#_Toc129520794)

[二分法程序改进 5](#_Toc129520795)

[不动点迭代法程序实现 5](#_Toc129520796)

## 第一次作业

1.1.设a=0.937关于精确数x有三位有效数字，估计a的相对误差.对于 f(x)= ,估计 f(a) 对于 f(x)的误差和相对误差。

解：a的相对误差.由于

|e(x)||x-a|， ，

因此，

f(a) 对于 f(x) 的误差和相对误差分别为

1.2.要使的近似值的相对误差限小于0.1%，要取几位有效数字？

解：设取n位有效数字，由定理1可知，，由于=4.4，∴=4，故只要取n=4，就有

，

故只要对的近似值取4位有效数字，其误差限就小于0.1%。

1.3.已测得操场长l的值为=110m,宽d的值为=80m，已知m，m.求面积s=ld的绝对误差限与先对误差限.[1]

解：∵s = ld,

∴

其中==80m，==110m，

=0.2m， =0.1m，

∴绝对误差限27；

相对误差限。

参考文献：

1. 黄云清,舒适,陈艳萍,金继承,文立平,石钟慈,等.数值计算方法[M].北京:科学出版社，2010.24-27

## 第二次作业

问题一：

1. 对于特定的问题，可能不存在Python模块可以直接使用的算法，这时候需要对算法进行自己的实现
2. 当使用Python模块进行数值计算和模拟时，需要了解算法的局限性以及其适用范围，才能正确地应用这些模块
3. 理解算法还可以更好地优化程序的性能，找到更高效的实现方式
4. 了解算法的工作原理可以更好地调试程序，当程序出现问题时，可以更快地定位问题所在
5. 学习数值计算与模拟的算法可以提高编程能力，更好地掌握编程技巧和编程思维

2.2避免有效位丢失的求解方法

问题二：利用平方完成后配方法求解，公式如下：

1. ，

原理：平方完成后配方法求解时只需要计算平方根，而不需要进行减法运算，避免因计算小数点后的差值而造成的有效位丢失。

1. 利用公式  求解，其中sign(b)为b的符号函数。

原理：避免计算平方根时可能出现的有效位丢失问题，同时也避免了计算中的减法运算。

1. 利用公式，求解。

原理：将求根公式进行了变形。[1] 附录

附录：

[三种代码的python实现]

方法一：

import math

def solve\_quad\_equation(a, b, c):

delta = b\*\*2 - 4\*a\*c

if delta >= 0:

x1 = (-b + math.sqrt(delta))/(2\*a)

x2 = (-b - math.sqrt(delta))/(2\*a)

return x1, x2

else:

return None

方法二：

def solve\_quad\_equation2(a, b, c):

if b >= 0:

sign\_b = 1

else:

sign\_b = -1

delta = b\*\*2 - 4\*a\*c

if delta >= 0:

x1 = (-b/(2\*a)) - (sign\_b\*math.sqrt(delta)/(2\*a))

x2 = (2\*c)/(-b - sign\_b\*math.sqrt(delta))

return x1, x2

else:

return None

方法三：

def solve\_quad\_equation3(a, b, c):

delta = b\*\*2 - 4\*a\*c

if delta >= 0:

x1 = (-2\*c)/(b + math.sqrt(b\*\*2 - 4\*a\*c))

x2 = (-2\*c)/(b - math.sqrt(b\*\*2 - 4\*a\*c))

return x1, x2

else:

return None

参考文献：

1. 黄云清,舒适,陈艳萍,金继承,文立平,石钟慈,等.数值计算方法[M].北京:科学出版社，2010.24-27

## 第三次作业

### 二分法程序改进

### 不动点迭代法程序实现

代码

import math

x = 1.5

iter = 0

LIMIT = 1e-5

func=int(input('Please select your function:(1~5)'))

if func==1:

def f(x):

return 7\*x\*\*5-13\*x\*\*4-21\*x\*\*3-12\*x\*\*2+59\*x+3

elif func==2:

def f(x):

return math.pow((13\*x\*\*4+21\*x\*\*3+12\*x\*\*2-58\*x-3)/7, 1/5)

elif func==3:

def f(x):

return (13+21/x+12/(x\*\*2)-58/(x\*\*3)-3/(x\*\*4))/7

elif func==4:

def f(x):

return math.pow(((12\*x\*\*2-58\*x-3)/(7\*x\*\*3-13\*x-21)), 1/3)

elif func==5:

def f(x):

return math.pow((-58\*x-3)/(7\*x\*\*3-13\*x\*\*2-21\*x-12), 1/2)

while 1:

iter += 1

if abs(f(x) - x) < LIMIT:

x =f(x)

print("{:.6g},{:.6g}".format(iter, x))

break

else:

x = f(x)

print("{:.6g},{:.6g}".format(iter, x))

if iter==10:

break