CompPhys Assignment 02

李尚坤 20307130215

1. **求解方程的两个正根**
   1. **题目描述**

Sketch the function :

(1) Determine the two positive roots to 4 decimal places using the bisection method.

(2) Take the two roots that you found in the previous question (accurate to 4 decimals) and “polish them up” to 14 decimal places using the Newton-Raphson method.

(3) Determine the two positive roots to 14 decimal places using the hybrid method.

* 1. **解决方案描述**

(1) Bisection method

·首先以0.5为步长，找出使得函数符号相反的两个区间，即为两个正根所在的区间（这一步骤也可以通过观察函数图像代替，但C/C++中绘图不便，因此用此方法）。

·在这两个区间内分别用二分法求出函数的根。

·二分法误差小于最后一次区间长度，因此当区间长度小于0.0001时，即精确到四位小数。

(2) Newton-Raphson method

·将之前求得的两个根作为初始值，采用牛顿迭代公式进行求解；

·当迭代中对根的修正值小于时，即精确到十四位小数。

(3) Hybrid method

·将(1)中所求区间作为求根的初始区间[a,b]，以区间中点作为初始的根

·如果函数在处切线与x轴不平行，则用牛顿迭代法进行下一次迭代。牛顿法所得结果如果在当前区间[a,b]内，则根据迭代后对应函数值的正负取新区间；若不在区间[a,b]内，舍弃牛顿迭代的结果，并采用二分法求解。

·如果函数在处切线与x轴平行，则用二分法进行下一次迭代。

·当满足时，即可认为满足精度（注：此处不能使用区间长度作为精度判断标准）

* 1. **伪代码**

|  |  |
| --- | --- |
| Algorithm 1.1 Bisection method | |
| ***Input***: the range of the two positive roots [0,10], | |
| ***Output***: two positive roots (4 decimal places) of the equation | |
| 1: | **Def** |
| 2: | **End Def** |
| 3: | **Def** FindBracket() //用于找到两个正根所在区间 |
| 4: | **For** by step **Do** |
| 5: | **If** |
| 6: | **Then** //为一个的二维数组 |
| 7: | **End If** |
| 8: | **End for** |
| 9: | **End Def** |
| 10: | **Def** FindRoot\_bis(, ) |
| 11: | **While** abs() |
| 12: | **If** |
| 13: | **Then** |
| 14: | **Else** |
| 15: |  |
| 16: | **End If** |
| 17: | **End While** |
| 18: | **End Def** |
| 19: | FindBracket() |
| 20: |  |
| 21: | FindRoot\_bis(), FindRoot\_bis() |
| 22: | **Return** |
| 23: | **End** |

|  |  |
| --- | --- |
| Algorithm 1.2 Newton-Raphson method | |
| ***Input***: two roots obtained by the former method | |
| ***Output***: two positive roots (14 decimal places) of the equation | |
| 1: | **Def** |
| 2: | **End Def** |
| 3: | **Def** FindRoot\_new() |
| 4: | **While** abs() |
| 5: |  |
| 6: | **End While** |
| 7: | **Return** |
| 8: | **End Def** |
| 9: | FindRoot\_new(), FindRoot\_new() |
| 10: | **Return** |
| 11: | **End** |

|  |  |
| --- | --- |
| Algorithm 1.3 Hybrid method | |
| ***Input***: the intervals obtained in former method of the two roots: | |
| ***Output***: two positive roots (14 decimal places) of the equation | |
| 1: | **Def** |
| 2: | **End Def** |
| 3: | **Def** FindRoot\_hyb() |
| 4: |  |
| 5: | **While** abs |
| 6: | **If** |
| 7: | **Then** |
| 8: | **If** |
| 9: | **Then If** |
| 10: |  |
| 11: | **Else** |
| 12: |  |
| 13: | **End If** |
| 14: | **Else** |
| 15: |  |
| 16: | **If** |
| 17: | **Then** |
| 18: | **Else** |
| 19: |  |
| 20: | **End If** |
| 21: |  |
| 22: | **End If** |
| 23: | **Else** |
| 24: |  |
| 25: | **If** |
| 26: | **Then** |
| 27: | **Else** |
| 28: |  |
| 29: | **End If** |
| 30: |  |
| 31: | **Return** |
| 32: | **End While** |
| 33: | **End Def** |
| 34: | FindRoot\_hyb(), FindRoot\_hyb() |
| 35: | **Return** |
| 36: | **End** |

* 1. **输入/输出示例**

本题在程序中设定两个正根的存在范围为0～10，在此范围内查找两个正根，结果如表1所示。终端的输出结果如图所示。

表1 输出结果

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Outputs** | | **Note** |
|  |  |
| Bisection | 0.6566 | 1.8342 | 4 decimals places |
| Newton-Raphson | 0.65662043104711 | 1.83424318431392 | 14 decimals places |
| Hybrid | 0.65662043104711 | 1.83424318431392 | 14 decimals places |

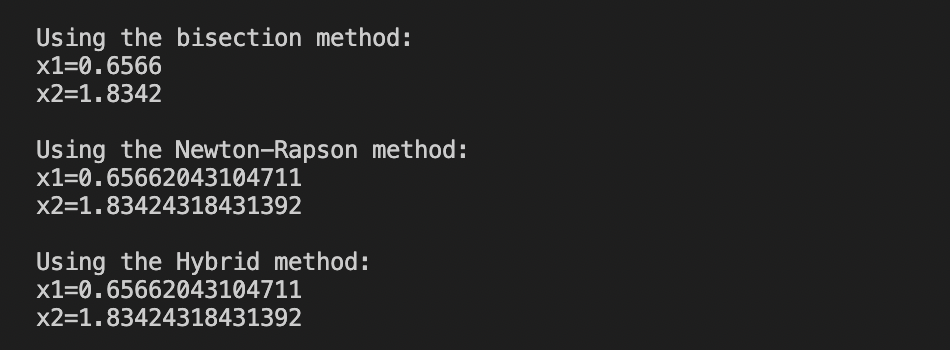
****

图1 终端输出结果

* 1. **用户手册**
* 本程序源文件为SketchFuntion.cpp，可执行文件为SketchFuntion.exe
* 为了防止直接运行.exe文件时cmd在运行结束后直接关闭，在源文件的末尾增加了一行命令system(“pause”)
* 本程序使用了三种方法求解方程，直接运行可执行文件，可以得到三种方法计算得到的结果
* 可通过修改FindBracket函数中start和end的值修改两个根存在的大致区间，本程序已设置为0和10.

1. **寻找函数的全局最小值**
   1. **题目描述**

Search for the minimum of the function in the whole place

* 1. **解决方案描述**

本题目采用梯度下降法求得函数的全局最小值

(1)为一个周期函数，周期为

(2)因此我们只需要在一个的平面区域内找到最小值点，就可将其作为全局最小值点

(3)给定一个初始点,我们在的平面内随机取30个点

(4)从这30个点出发，分别沿负梯度方向移动，分别到达对应的极小值点；

(5)从这30个局部极小值点中找到最小值，将其作为全局最小值点

(6)本算法中隐含有一个假设：就是从30个随机取点出发，一定可以找到函数所有的局部极小值

该假设的合理性在于：平面的大小仅为，并且函数是连续的，因此选取30个初始点就可以很好地代表这个局部平面内的点

* 1. **伪代码**

|  |  |
| --- | --- |
| Algorithm 2 寻找函数全局最小值 | |
| ***Input***: initial point , steps | |
| ***Output***: minimum of the function and the point | |
| 1: | **Def** Find\_min() |
| 2: | **For** **to** **by** **Do** |
| 3: | random point in plane |
| 4: | **While**  **or** |
| 5: |  |
| 6: |  |
| 7: | **End While** |
| 8: |  |
| 9: | **End For** |
| 10: | **For** each in **Do** |
| 11: | **If** |
| 12: | **Then** |
| 13: | **End If** |
| 14: | **End For** |
| 15: | **Return** |
| 16: | **End Def** |
| 17: | Find\_min() |
| 18: | **End** |

* 1. **输入输出示例**

本题的通过输入不同的初始点的位置和步长，所得的结果如表2所示。程序在终端执行的输入输出图如图2所示。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Inputs** | | | **Outputs** | | **Note** |
|  |  |  |  |  |
| 0.1 | 1 | 5 | (0.000007,10.995570) | -2.000000 | 所得的全局最小值都相同 |
| 0.01 | -5 | -6 | (-6.283178,-1.570801) | -2.000000 |
| 0.001 | 100 | -200 | (100.530972,-196.349545) | -2.000000 |

表2 输入输出结果

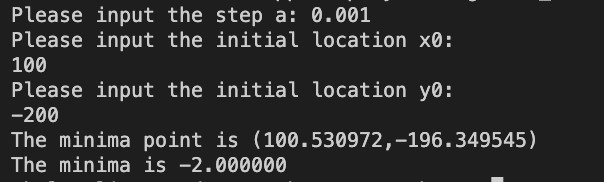
****

图2 终端输入输出结果

* 1. **用户手册**
* 本程序源文件为FindMin.cpp，可执行文件为FindMin.exe
* 为了防止直接运行.exe文件时cmd在运行结束后直接关闭，在源文件的末尾增加了一行命令system(“pause”)
* 本程序使用梯度下降法求得了周期函数在全平面的最小值
* 运行程序后，首先会跳出提示语“Please input the step a:”，提示输入梯度下降的步长，用键盘键入一个数字之后按回车键结束
* 随后先后跳出提示语“Please input the initial location x0:”与“Please input the initial location y0:”，用键盘分别键入后按回车键结束，随后程序输出最小值以及取得最小值所在的点

1. **求解隐函数**
   1. **题目描述**

Determine the reduced magnetization as a function of reduced temperature for simple materials.

For a given , solve , plot as a function of .

* 1. **解决方案描述**

(1)首先研究方程解的情况，即以0.5为步长取定分别为中的数，作出的图像

(2)由图可以知道，当与的时候，函数都只有这一个零点；当时，函数有三个零点，除这一平凡解之外，还有关于原点对称的两个零点，并且这两个零点的绝对值总在0到1之间

(3)考虑真实的物理场景：约化温度，的两个非零解大小相等，正负代表磁化方向，的平凡解则不予考虑

(3)因此我们只需要着重分析当时的解,这一部分的解可以由二分法求得

* 1. **伪代码**

|  |  |
| --- | --- |
| Algorithm 3 Temperature dependence of magnetization | |
| ***Input***: ­the interval may exsit | |
| ***Output***: the graph of | |
| 1: | **For** *t*  **to** **by** **Do** |
| 2: | **For** **to** **by** **Do** |
| 3: | **Plot** |
| 4: | **End For** |
| 5: | **End For** |
| 6: | **Def** FindRoot() |
| 7: | **If** |
| 8: | **Then Return** 0 |
| 9: | **Else** |
| 10: | **While** abs() |
| 11: | **If** |
| 12: | **Then** |
| 13: | **Else** |
| 14: |  |
| 15: | **End If** |
| 16: | **Return** |
| 17: | **End Def** |
| 18: | **For Do** |
| 19: | FindRoot(0.000001,1,) |
| 20: | **End For** |
| 21: | **Plot** |
| 22: | **Return** the graph of |
| 23: | **End** |

* 1. **输入/输出示例**

直接执行程序，我们将得到在时的函数图像，如图3所示。

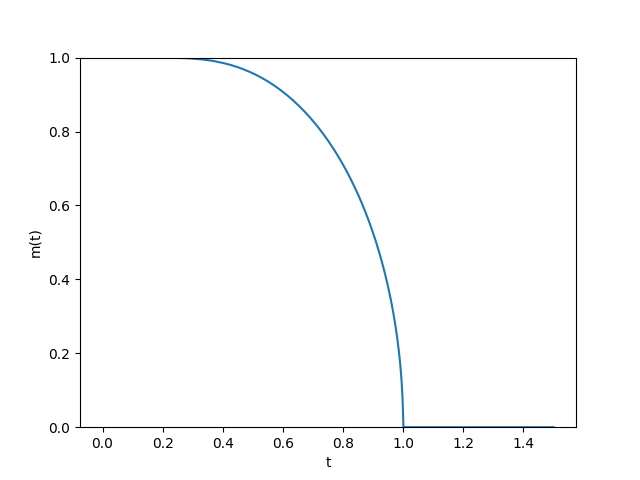


图3 程序输出结果

我们同时给出在分析过程中绘制的，的值取定时，的函数图像。如图4、图5、图6、图7所示，分别为时的图像。

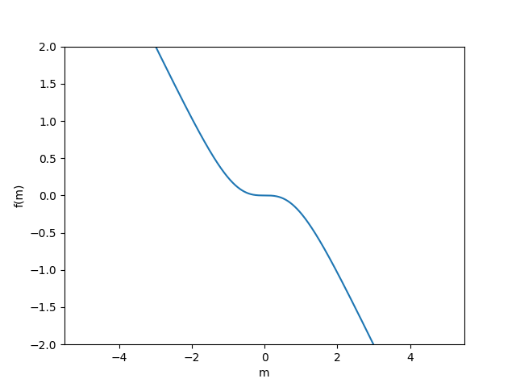


图6 t=1 一个平凡解

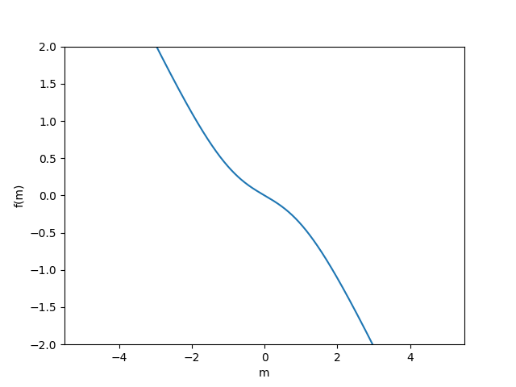


图7 t=1.4 一个平凡解

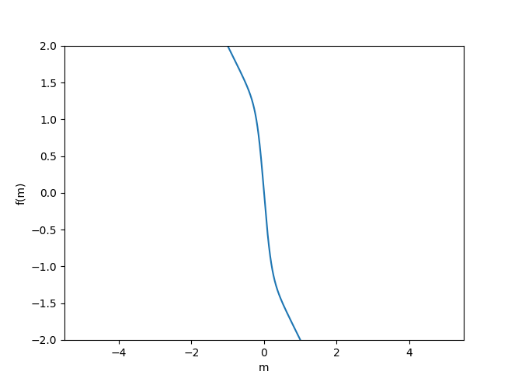


图4 t=-0.2 一个平凡解

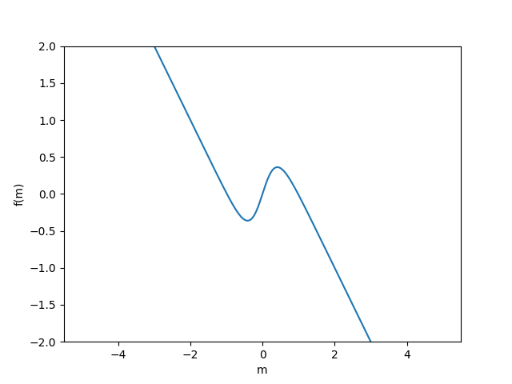


图5 t=0.4 三个解

* 1. **用户手册**
* 本程序源文件为Plotm\_t.py，可执行程序为Plotm\_t.exe
* 在执行源程序之前，应当先安装numpy以及matplotlib库
* 本程序用于求解隐函数，绘制约化磁化强度与约化温度之间的关系()
* 运行程序后，将直接输出函数图像