**《数值代数》第五次上机作业 实验报告**

匡亚明学院 211240021 田铭扬

**摘要**

笔者使用C++语言（利用OpenBLAS和lapacke库）编程，对使用Newton法求解非线性方程(组)的零点的问题，进行了初步的探索。

**正文**

**前言**

实际生活是复杂的，在应用中遇到的问题往往不是线性的。在非线性方程（组）的零点（或不动点）问题的求解中，Newton法是十分经典且行之有效的一类方法，不少更“高级”的算法都是从Newton法衍生而来的。

本次数值实验的目的， 即是对这一经典方法进行一定的实践探索。

**问题**

用真实误差作为停机标准，用户指标设定为 ε = 10−6。

**·第1题** 用Newton方法，计算多项式的最大实根。绘制误差曲线，计算其数值收敛阶。

**·第2题** 已知是的重根。采用Newton方法求解，观察其误差曲线和数值收敛阶。修改算法，改善收敛阶，并给出相应实验数据。

**·第3题** 用割线法，计算前两题的问题；比较其与切线法的区别（例如收敛阶和 CPU 时间），并给出相应的数值观察。

**·第4题(上)** 考虑非线性方程组 （不进行任何手工化简）。取初始位置(−0.15, 1.4)，观察Newton方法的误差曲线和迭代次数；尝试其它初始位置，看看其具体情况是什么？

**·第7题** 设，其特征值问题可陈述为。阶数取n=5和n=8，任取单位向量x0，令λ0 = x0⊤Tnx0，执行Newton方法，观察其数值结果同幂法有何区别，并给出相应的解释。

**数学推导与程序设计**

Newton算法在讲义[1]或教材[2]中较详细的实现流程，故在此不再讨论。

题目要求以停机准则为实际误差，但其表现并不理想。因而前三题均改为与讲义一致：同时使用迭代点的相邻误差和函数值的实际误差作为停机准则。

此外，经过简单的数学推导可得以下结论：1.第1题的方程有单根x = -3和二重根x = 2；2.第2题的方程有五重根x = 0；3.第4题的方程组的Fréchet微分为；4.第7题的方程组的Fréchet微分为。以上结果会在程序编写中用到。

而在前三题中要求计算数值（后验）收敛阶，可以通过如下公式给出：

**实验环境**

部分代码使用C++编写。在虚拟机软件VMWare 17中运行deepin 20.9操作系统，设置8GB内存和8个CPU核心，开启Intel VT-x/EPT和IOMMU选项。使用了OpenBLAS、lapacke等数值代数库，对向量—向量与矩阵—向量运算进行并行加速。使用GCC 8.3.0-1编译器，未开启优化选项。

**实验结果分析**

**第一题**

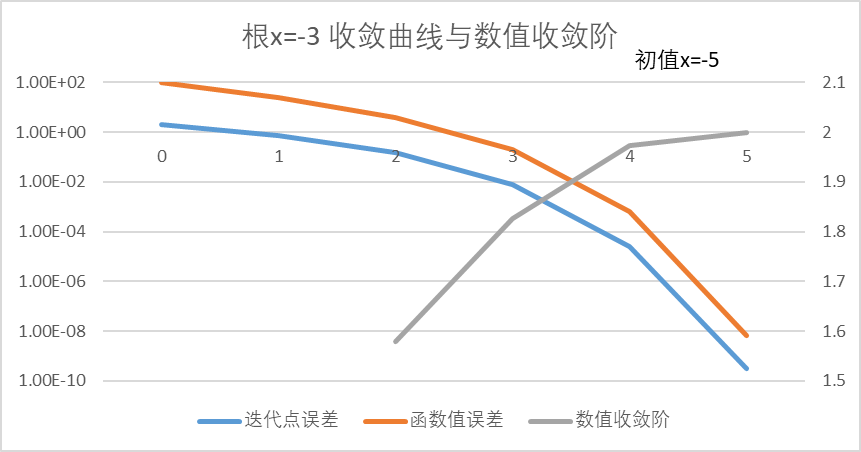
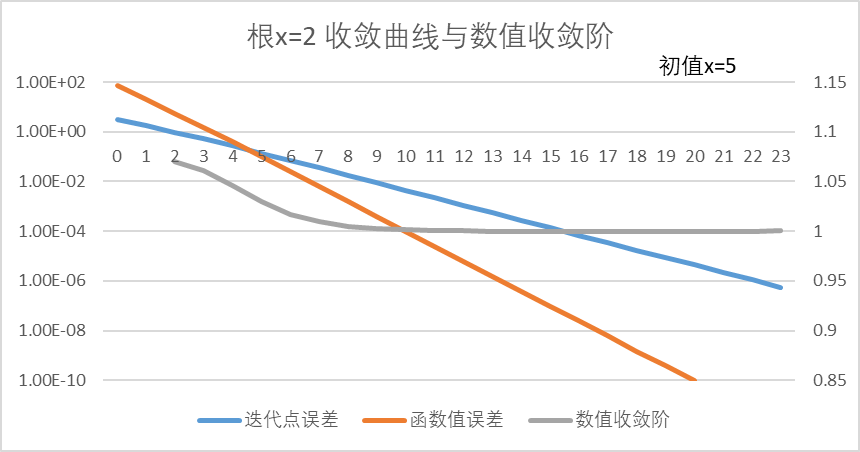
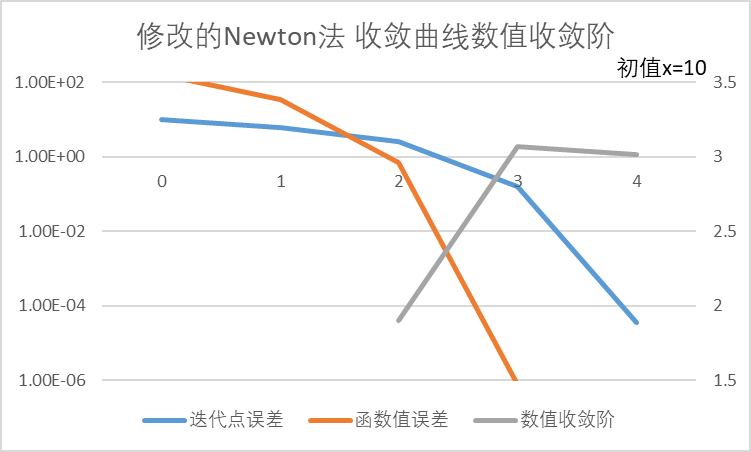
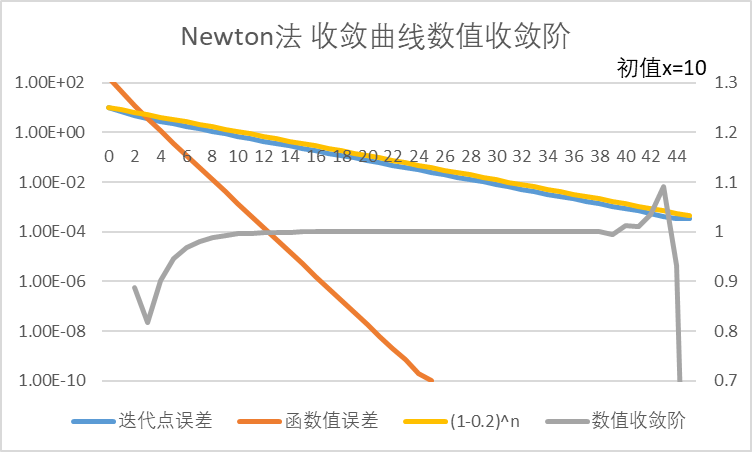
****

图1-2：Newton法收敛曲线与数值收敛阶

如图，算法结果与理论相符：在重根处表现出线性收敛，而在单根处最终为平方收敛。且注意到单根处的收敛步数（5步）远少于重根（23步）。

**第二题**

****

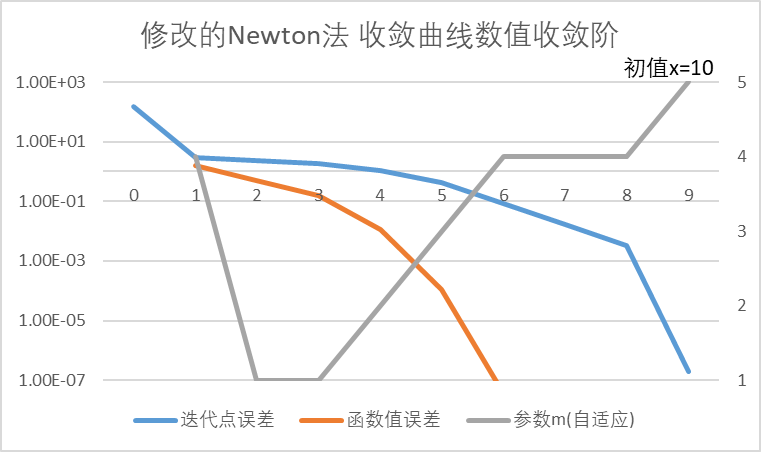
****

图3-5：Newton法及其改进 收敛曲线与数值收敛阶

如左上图，Newton法线性收敛至五重根x=0处。图中黄线以渐进下降速度1 - 5-1变化，由此看到，算法表现能与理论(讲义[1]P131定理5.4)较好地吻合。

设置参数m=5应用修正的Newton法，得到右上图，其中可以见到收敛速度的极大改善。算法的数值收敛解最终达到3阶（而非2阶），可能是由于问题较为特殊，或算法收敛过快、数值收敛阶还未稳定。另外需要注意，算法实际上是在第5步停机，但第5步的计算结果为“nan”（推测可能是突破了下溢值），只能采用第4步的结果，这是在实际应用中需要注意的。

下图是参照讲义[1]P132进行“自适应”Newton法的结果。其收敛速度显著好于原方法，虽相比已知m的方法略差，但优点正在于不需预先知道根重数。

**第三题**

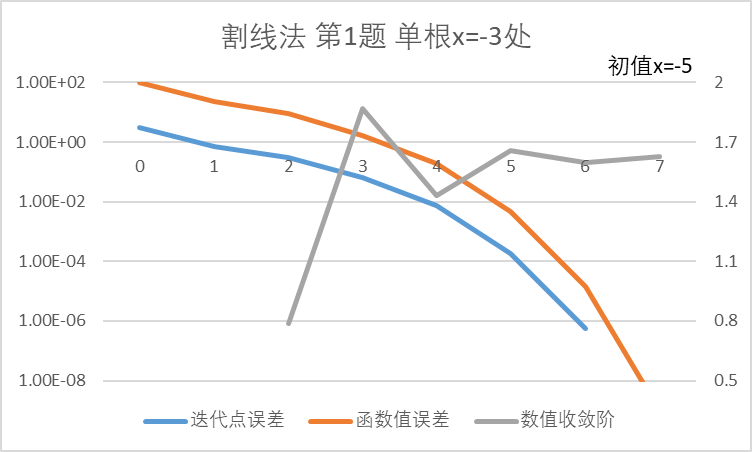
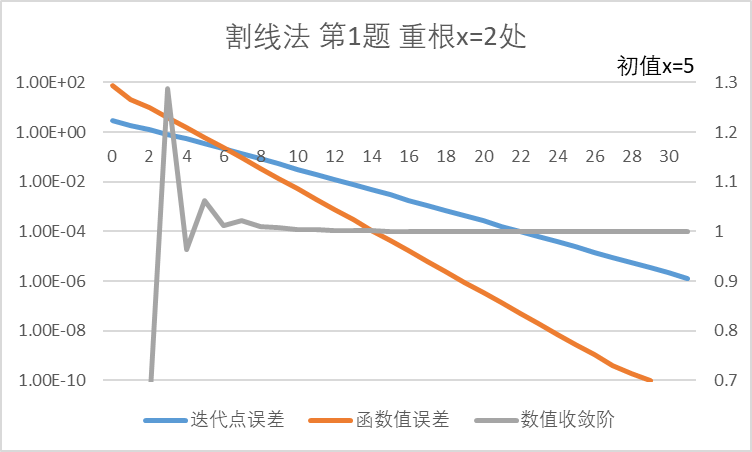
****

图6-7：第1题改用割线法 收敛曲线与数值收敛阶

如图，第1题改用割线法，两根处的迭代步数都比Newton法略多（重根处23步→31步/单根处5步→7步）。而关于重根处的后验收敛阶仍为1阶，单根处则只有1.6阶左右（与理论的1.618阶相符）。

由于时间有限，本次实验没有进行CPU时间的比较。但由于题目中的函数较为简单，可以“手工”求导，算法中的导数值是显示计算出的。因此猜测，即使比较CPU时间，割线法不会因无需计算导数产生较大的时间优势。

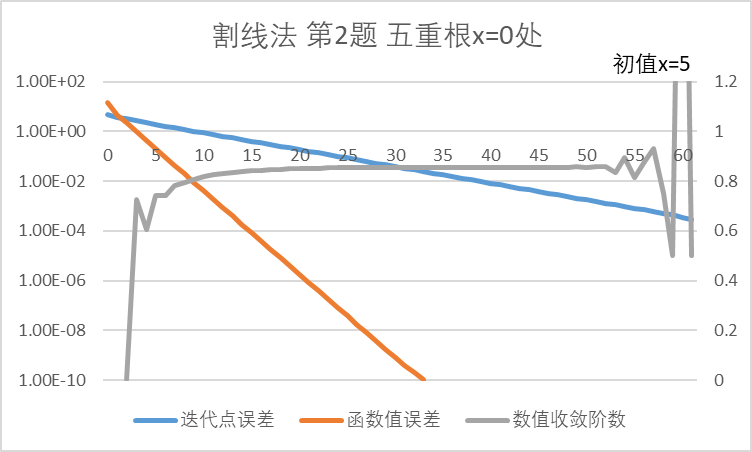


图8：第2题改用割线法 收敛曲线与数值收敛阶

如图，第2题割线法的迭代步数比Newton法多了50%（44步→62步），且收敛阶从1阶降为约0.85阶。可能是由于本题的零点很“坏”（重数为5）。

**第四题**

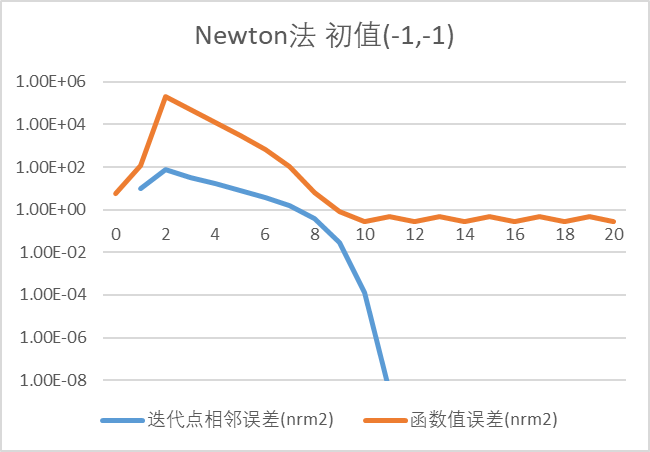
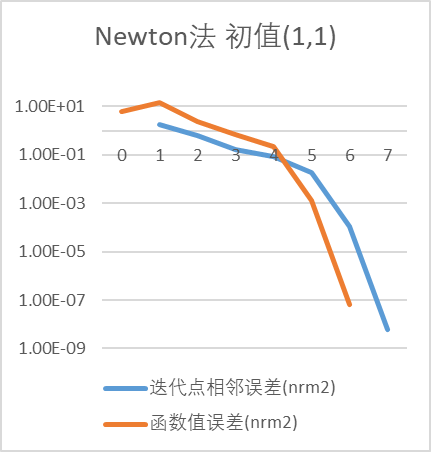
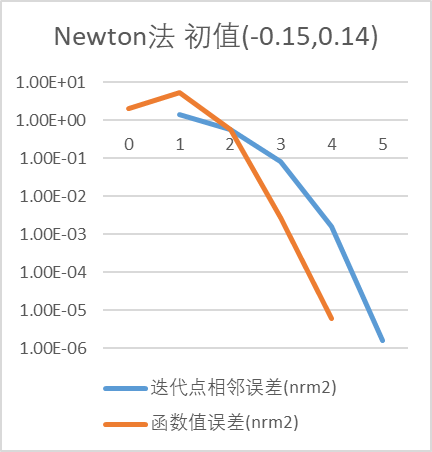
****

图9-11：Newton在不同初值下的收敛曲线

如图，共选择了三种初值进行试验。使用题目要求的初值，算法收敛至方程的解(0,1)；使用初值(1,1)，算法收敛至方程的近似解(1.7598,1.7940)。

但使用初值(-1,-1)时，算法没有停机，而是在(46.588,2.5762)附近的两个点（相邻误差小于1e-8）之间震荡。使用Mathmatica画图（见下页）可以直观看到，原方程组在这附近的形态并不“好”（函数值有许多局部极小值），出现这样的现象——可能是因为迭代序列陷入了一个局部极小值的领域——就并不奇怪了。这说明，Newton法的效果受所求方程组的性质“好坏”影响较大。

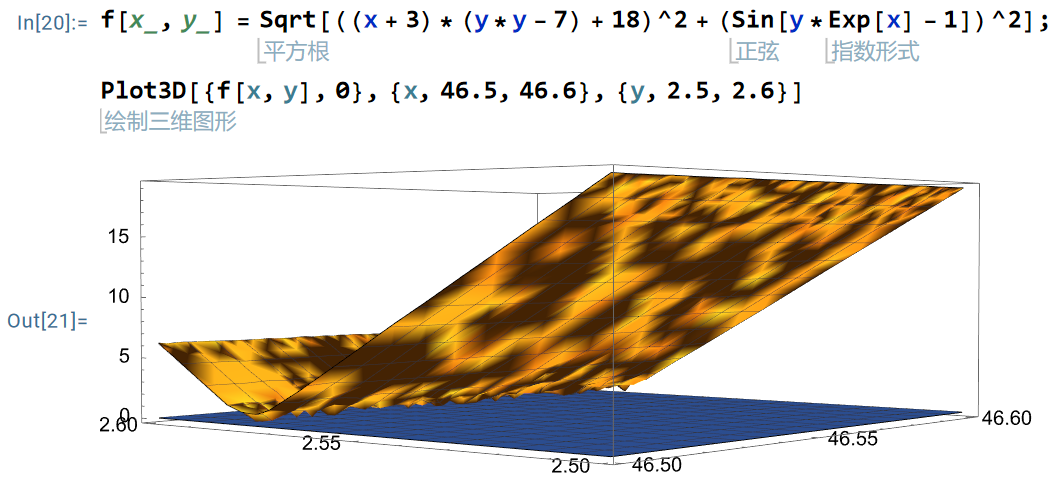


图12：方程组(函数值的2-范数)在(46.55,2.55)附近的表现

**第七题**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n=5** | **计算结果** | 0.2679491924 | 1.0000000000 | \ |
| **对应特征值** | 2-2\*cos(Pi/6) | 2-2\*cos(2Pi/6) | \ |
| **误差** | <1E-10 | <1E-10 | \ |
| **n=8** | **计算结果** | 0.1206147584 | 0.4679111138 | 1.0000000000 |
| **对应特征值** | 2-2\*cos(Pi/9) | 2-2\*cos(2Pi/9) | 2-2\*cos(3Pi/9) |
| **误差** | <1E-10 | <1E-10 | <1E-10 |

表1：Newton法计算特征值的结果

如表，随机选取初值进行多次尝试，能够得到原矩阵所有模小于1的特征值。但是函数值的计算结果（受篇幅限制未列出）会收敛至一个非0的值。猜测这是因为λ接近Tn的特征值时，方程组的Fréchet微分矩阵（见“数学推导与程序设计”部分）接近奇异，使得舍入误差的问题无法被忽略。

**结语**

这是本学期的最后一次数值实验。在实验中，笔者简要地探索了Newton方法等的实际表现，但还留有不少问题。这为今后学习更多计算、优化相关的课程埋下了伏笔。

**实验代码**

由于篇幅限制，代码及原始数据不在实验报告中列出，可以在笔者github仓库中查看。网址为：<https://github.com/lk758tmy/NA2-Codes>。

**参考文献**

[1] 《数值代数》讲义. 张强

[2] 数值计算方法-下册. 林成森. 科学出版社. 2005-1第二版