《计算物理》第 4 次作业

1900017812 高乐耘*

2022 年 11 月 25 日

I. 迭代法求解线性方程组

我于源文件 solve/gd.c 和 solve/cg.c 中分别实现了 GD（梯度下降）和 CG（共轭梯度）迭代法求解实对称正定矩阵线性方程组的函数，并分别使用简单的测试程序 solve/gd-test.c 和 solve/cg-test.c 来测试它们编写的正确性。程序 solve/solve-gd.c 和 solve/solve-cg.c 分别使用这两种算法代入本题给出的矩阵进行求解，**以零向量为初始值，以残差向量的欧几里得范数不超过机器精度为判停标准**。后面我们将从它们不同的迭代过程中看出共轭梯度法在加快计算速度和节省计算量上的优越性。显然，本题的精确解为：

```
1.000000000000000000   0x1.00000000000000+0
1.000000000000000000   0x1.00000000000000+0
1.000000000000000000   0x1.00000000000000+0
1.000000000000000000   0x1.00000000000000+0
```

梯度下降算法的求解结果为：

```
0.999999999998341549   0x1.fffffffffc5a6p-1
1.000000000000977662   0x1.0000000001133p+0
1.000000000000403899   0x1.000000000071bp+0
0.999999999999764078   0x1.fffffffff7b3p-1
```

共轭梯度算法的求解结果为：

```
0.999999999999947042   0x1.fffffffffe23p-1
1.000000000000028422   0x1.0000000000080p+0
1.000000000000015321   0x1.0000000000045p+0
0.999999999999993561   0x1.ffffffffffc6p-1
```

以上结果的十六进制浮点数表示中所有的十六进制位在 64 位机器精度下均为连续变化的有效数字。

直观上我们看到，共轭梯度法的求根结果在十六进制表示下比梯度下降法大约高一个数量级。为了更加细致地比较两种算法，图 1 和 2 分别作出了误差向量欧几里得范数随迭代次数的变化情况。**注意，误差向量与残差向量不同，但分析精度时我们更关心误差**。图 1 中显示，梯度下降法经历了两万多次漫长的迭代才终于将残差范数降至机器精度以下，在整个过程中，误差范数缓慢而均匀地指数衰减。图 2 中显示，共轭梯度法在迭代次数达到未知元个

*电子邮件地址: seeson@pku.edu.cn　手机号: 13759115414

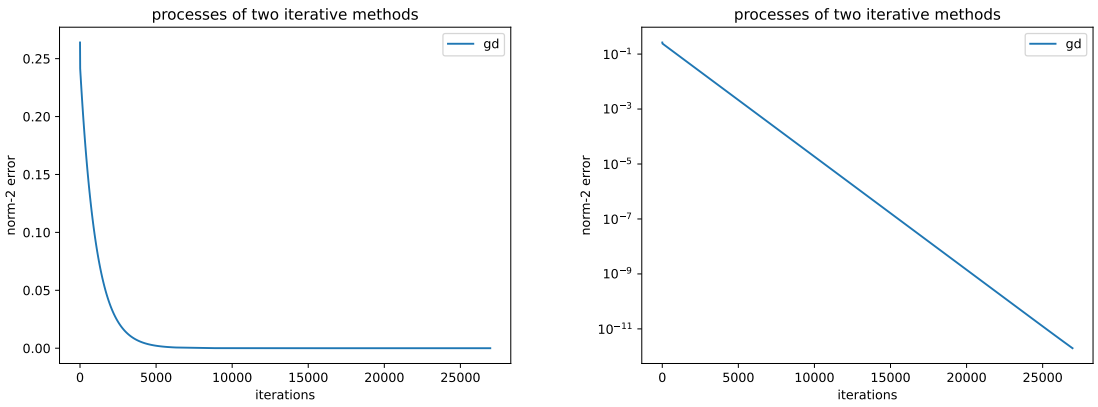

图 1: 梯度下降法迭代过程中误差向量欧几里得范数的变化。左：纵坐标线性，右：纵坐标对数线性。

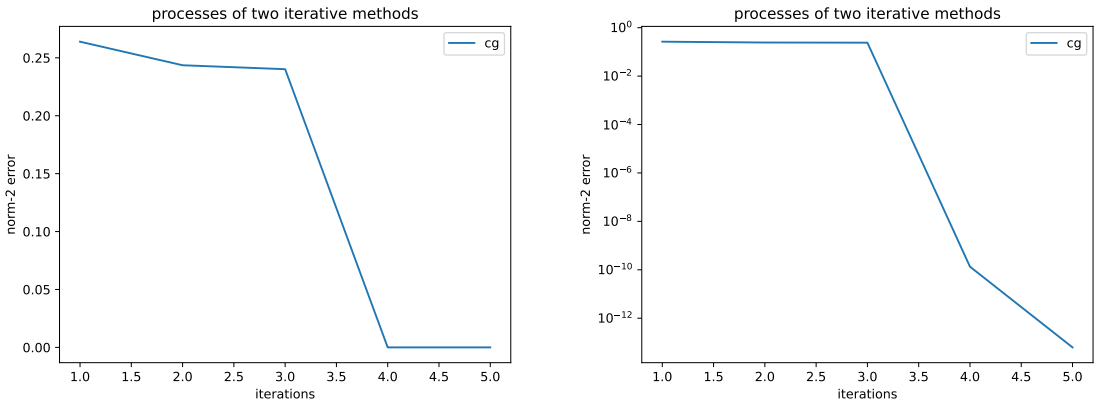

图 2: 共轭梯度法迭代过程中误差向量欧几里得范数的变化。左：纵坐标线性，右：纵坐标对数线性。

数时，与理论分析一致地将误差几乎降低至零，但由于浮点计算的有限精度，额外进行了第五次迭代才将残差范数降低至浮点精度下，与此同时误差也随之降低若干个数量级。

在这个例子中，共轭梯度法相比梯度下降法节约了数千倍的计算量达到了甚至超过梯度下降法的求解精度，充分体现出共轭梯度法基于克莱洛夫子空间迭代算法的优越性。

II. 迭代法求解实对称正定三对角矩阵本征值

我于源文件 eigen/qr.c，eigen/qrh.c，eigen/qri.c，eigen/qrih.c，eigen/hhr.c，eigen/qrig.c，eigen/jcbi.c，eigen/sgi.c 中分别实现了基于 Gram-Schmidt 正交化的 QR 分解函数，基于 Householder 变换的 QR 分解函数，基于 Gram-Schmidt 正交化的 QR 迭代求本征值函数，基于 Householder 变换的 QR 迭代求本征值函数，Hessenberg-Householder 变换函数，基于 Givens 变换的针对上 Hessenberg 矩阵的 QR 迭代求本征值函数，实对称矩阵的 Jacobi 迭代求本征值函数，和实对称三对角矩阵的 Sturm-Gershgorin 二分法求本征值函数。eigen/qr-test.c，eigen/qrh-test.c，eigen/qri-test.c，eigen/qrih-test.c，eigen/hhr-test.c，eigen/qrig-test.c，eigen/jcbi-test.c，eigen/sgi-test.c 为它们对应的测试代码。源程序 eigen/eigen.c 实现了本题的求解流程。所有迭代方法的判停标准为至少满足以下任意条件之一：

- 迭代次数达到设定的上限
- 二分区间长度小于设定的上限（Sturm-Gershgorin 二分法）或矩阵的非对角元绝对值之和小于设定的上限（其它方法）

源程序 eigen/eigen.c 中，**我们将迭代次数设定得足够大，并将精度设定得足够小，观察迭代时本征值结果变化的图像，确保迭代收敛到稳定值，以比较不同算法对求解这一问题而言的优劣**。

手算容易得到题中矩阵的特征方程为 $x^4 - 10x^3 + 32x^2 - 35x + 7 = 0$，**使用二分法容易在机器精度下求出其四个实根**（eigen/plot#L35-L37），**作为本征值的精确值**，以计算上述各迭代算法得到的本征值的误差。

由于本题中的矩阵为实对称正定三对角矩阵，以上所有的 QR 分解迭代都可以直接使用，且使用基于 Givens 变换的 QR 迭代前无需进行 Hessenberg-Householder 变换。所有方法迭代过程中四个本征值的绝对误差变化情况绘制于图 3 中。注意，**除 Sturm-Gershgorin 方法直接获得各本征值二分区间外，其余方法迭代过程中矩阵被近似对角化，因此我们可以认为其对角元就是当前迭代步骤给出的本征值近似值**。由于图 3 中各曲线存在遮挡，图 4 在图 3 的基础上去除了其中一条曲线，以方便观察。

图 3 中基于 Householder 变换的 QR 方法迭代曲线被完全遮挡，作为补充，图 4 中显示了它。对于后三个本征值而言，这条曲线终止于误差较高的点，说明我对这一算法的实现不够稳定。仔细观察程序输出的这个部分：

```
engin_qrih-12: 0.130721
       4.74304      -0.0592612    -1.06363e-15     1.24792e-12
    -0.0592612          3.1795    -0.00609912    -1.25784e-10
             0    -0.00609912         1.82274     1.18215e-08
             0               0               0        0.254719
```

可以看到，基于 Householder 变换的 QR 迭代终止于第 12 次迭代，此时矩阵的非对角元绝对值之和为 0.130721，并没有实现较好的对角化。

从图 3 中我们看到，基于 Gram-Schmidt 正交化和 Givens 变换的 QR 迭代过程大致接近。忽略因 QR 分解细节造成的差别，以更为简单的基于 Givens 变换的 QR 迭代算法为例，第 5，10，15……次迭代后的矩阵分别为：

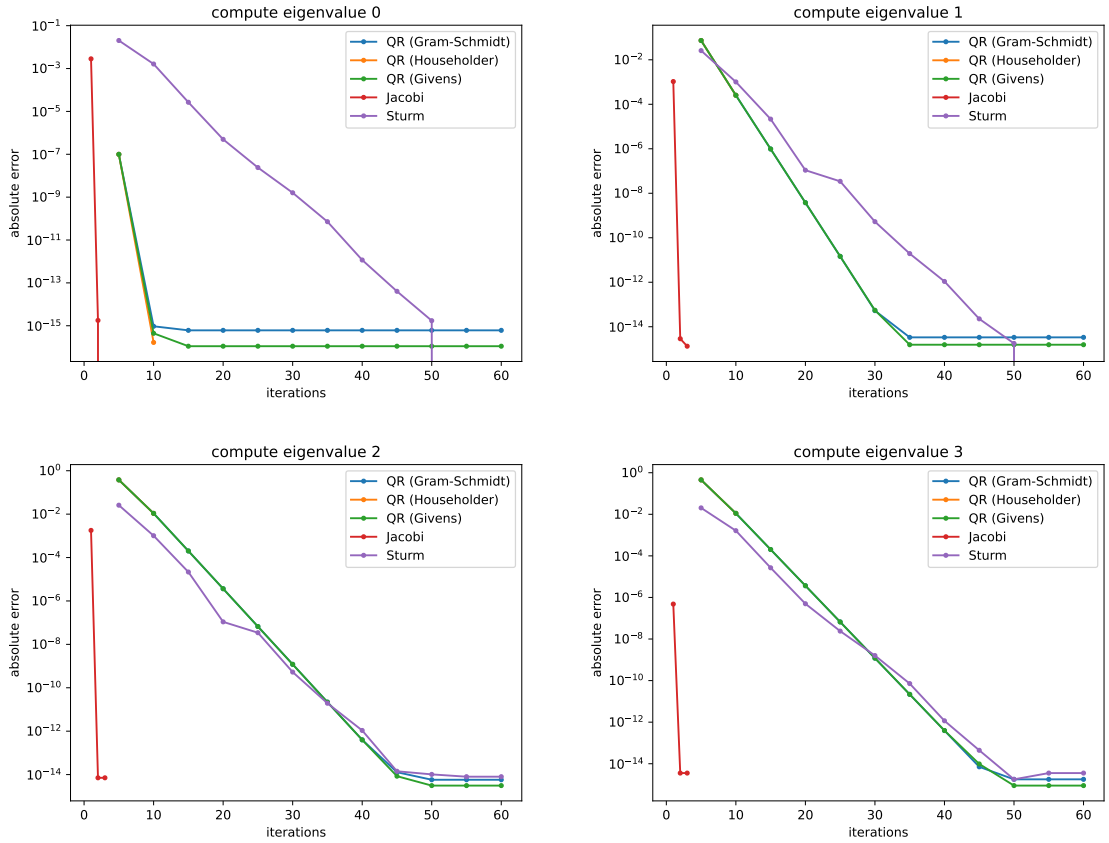

图 3: 各迭代方法求解本征值过程中本征值绝对误差的变化情况。四个本征值均为正数，按照升序大小排列。

```
qrig:
5:
[[ 4.29276628e+00 -7.21313977e-01 -1.79875969e-16 -3.41297430e-16]
 [-7.21313977e-01  3.55611356e+00 -3.34967464e-01  9.47592699e-17]
 [ 0.00000000e+00 -3.34967464e-01  1.89640130e+00 -3.99652715e-04]
 [ 0.00000000e+00  0.00000000e+00 -3.99652715e-04  2.54718859e-01]]
10:
[[ 4.73418406e+00 -1.31448547e-01 -2.55004323e-16 -3.47107253e-16]
 [-1.31448547e-01  3.18812610e+00 -1.85822775e-02 -7.09717417e-17]
 [ 0.00000000e+00 -1.85822775e-02  1.82297108e+00 -2.07643133e-08]
 [ 0.00000000e+00  0.00000000e+00 -2.07643133e-08  2.54718760e-01]]
15:
[[ 4.74507887e+00 -1.78120256e-02 -2.68943125e-16 -3.41025780e-16]
 [-1.78120256e-02  3.17748431e+00 -1.15075434e-03 -9.58405935e-17]
 [ 0.00000000e+00 -1.15075434e-03  1.82271806e+00 -1.10660375e-12]
 [ 0.00000000e+00  0.00000000e+00 -1.10658664e-12  2.54718760e-01]]
20:
[[ 4.74527757e+00 -2.39738535e-03 -2.71037573e-16 -3.40067087e-16]
 [-2.39738535e-03  3.17728658e+00 -7.14944440e-05 -9.91751186e-17]
 [ 0.00000000e+00 -7.14944440e-05  1.82271708e+00 -7.61738699e-17]
 [ 0.00000000e+00  0.00000000e+00 -5.89784840e-17  2.54718760e-01]]
```

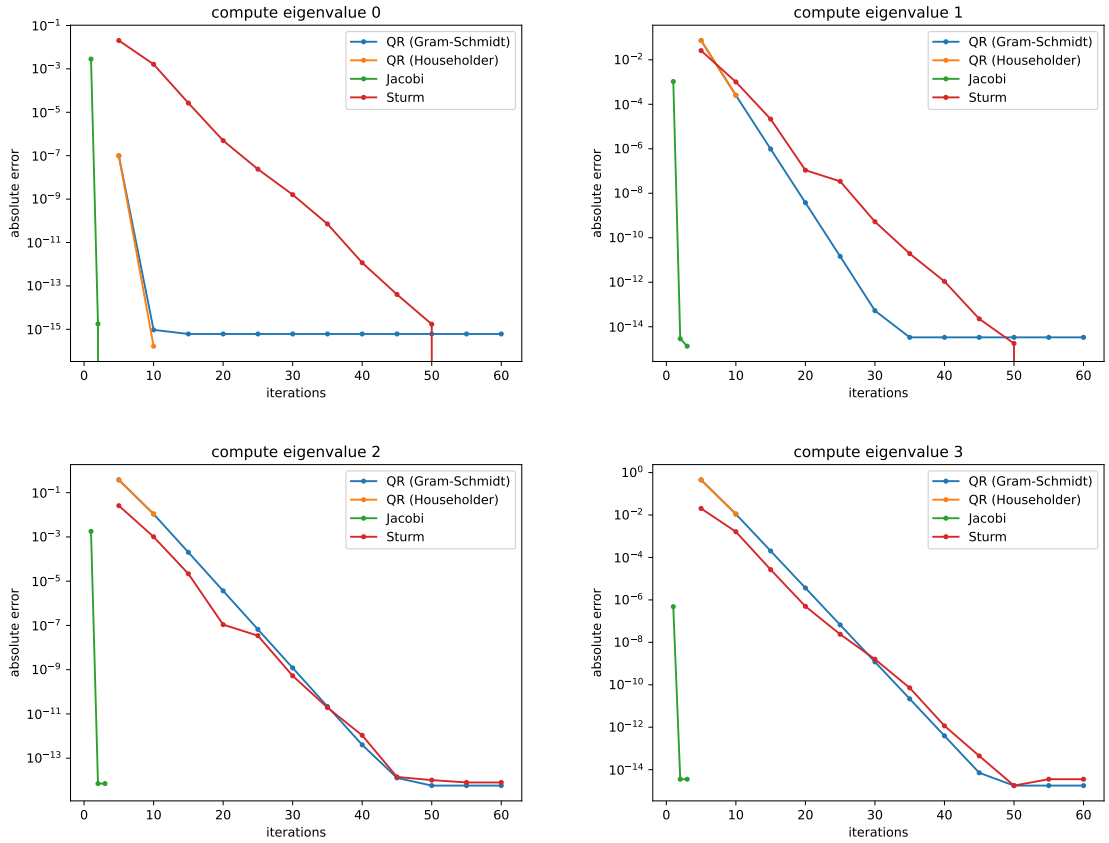

图 4: 各迭代方法求解本征值过程中本征值绝对误差的变化情况。为方便观察，去除了基于 Givens 变换的 QR 迭代方法。

25:
[[4.74528117e+00 −3.22632786e−04 −2.71340387e−16 −3.39935563e−16]
[−3.22632786e−04 3.17728299e+00 −4.44210350e−06 −9.96241529e−17]
[0.00000000e+00 −4.44210350e−06 1.82271708e+00 −1.72034517e−17]
[0.00000000e+00 0.00000000e+00 −3.14341662e−21 2.54718760e−01]]
30:
[[4.74528124e+00 −4.34188356e−05 −2.71382531e−16 −3.39917817e−16]
[−4.34188356e−05 3.17728292e+00 −2.75997749e−07 −9.96846309e−17]
[0.00000000e+00 −2.75997750e−07 1.82271708e+00 −1.72006149e−17]
[0.00000000e+00 0.00000000e+00 −1.67536827e−25 2.54718760e−01]]
35:
[[4.74528124e+00 −5.84316066e−06 −2.71388291e−16 −3.39915428e−16]
[−5.84316066e−06 3.17728292e+00 −1.71483529e−08 −9.96927734e−17]
[0.00000000e+00 −1.71483531e−08 1.82271708e+00 −1.72006338e−17]
[0.00000000e+00 0.00000000e+00 −8.92932493e−30 2.54718760e−01]]
40:
[[4.74528124e+00 −7.86352881e−07 −2.71389071e−16 −3.39915107e−16]
[−7.86352881e−07 3.17728292e+00 −1.06546502e−09 −9.96938694e−17]
[0.00000000e+00 −1.06546526e−09 1.82271708e+00 −1.72006350e−17]
[0.00000000e+00 0.00000000e+00 −4.75912342e−34 2.54718760e−01]]
45:

```
[[ 4.74528124e+00 -1.05824722e-07 -2.71389177e-16 -3.39915063e-16]
 [-1.05824722e-07  3.17728292e+00 -6.61994794e-11 -9.96940169e-17]
 [ 0.00000000e+00 -6.61997229e-11  1.82271708e+00 -1.72006351e-17]
 [ 0.00000000e+00  0.00000000e+00 -2.53650258e-38  2.54718760e-01]]
50:
[[ 4.74528124e+00 -1.42415343e-08 -2.71389191e-16 -3.39915058e-16]
 [-1.42415347e-08  3.17728292e+00 -4.11289228e-12 -9.96940368e-17]
 [ 0.00000000e+00 -4.11313580e-12  1.82271708e+00 -1.72006351e-17]
 [ 0.00000000e+00  0.00000000e+00 -1.35189714e-42  2.54718760e-01]]
55:
[[ 4.74528124e+00 -1.91657738e-09 -2.71389193e-16 -3.39915057e-16]
 [-1.91657778e-09  3.17728292e+00 -2.55314744e-13 -9.96940395e-17]
 [ 0.00000000e+00 -2.55558262e-13  1.82271708e+00 -1.72006351e-17]
 [ 0.00000000e+00  0.00000000e+00 -7.20529871e-47  2.54718760e-01]]
60:
[[ 4.74528124e+00 -2.57926184e-10 -2.71389193e-16 -3.39915057e-16]
 [-2.57926582e-10  3.17728292e+00 -1.56348841e-14 -9.96940398e-17]
 [ 0.00000000e+00 -1.58784024e-14  1.82271708e+00 -1.72006351e-17]
 [ 0.00000000e+00  0.00000000e+00 -3.84025737e-51  2.54718760e-01]]
```

从图 3 中可见，Jacobi 迭代算法在数次迭代后迅速收敛至机器精度附近，每次迭代后的矩阵分别为：

```
jcbi:
1:
[[ 2.57557408e-01 -6.66573810e-02  4.82826114e-03  1.46599391e-03]
 [-6.66573810e-02  1.82166341e+00 -4.89364070e-02 -3.32950569e-05]
 [ 4.82826114e-03 -4.89364070e-02  3.17549842e+00  1.14058069e-16]
 [ 1.46599391e-03 -3.32950569e-05 -1.14925430e-16  4.74528076e+00]]
2:
[[ 2.54718760e-01  5.23445637e-08 -8.60545124e-15 -1.29932100e-17]
 [ 5.23445639e-08  1.82271708e+00  1.66531587e-16 -5.41723252e-17]
 [-8.70144980e-15 -1.66528101e-16  3.17728292e+00  1.16510229e-16]
 [ 1.29931909e-17  5.41723252e-17 -1.16510229e-16  4.74528124e+00]]
3:
[[ 2.54718760e-01 -5.71548129e-17  4.79992741e-17 -1.29931986e-17]
 [ 5.71548129e-17  1.82271708e+00  1.66529846e-16 -5.41723256e-17]
 [-4.79992741e-17 -1.66529846e-16  3.17728292e+00  1.16510229e-16]
 [ 1.29931986e-17  5.41723256e-17 -1.16510229e-16  4.74528124e+00]]
```

从图 3 中可见，由于 Sturm-Gershgorin 方法的二分特性，对于各个本征值，其收敛速度大致相同。其二分区间为 $[0,5]$，各次迭代后四个本征值的近似值分别为：

```
sgi:
eigenvalue 0:
{
  5: 0.234375,              10: 0.25634765625,        15: 0.2547454833984375,
  20: 0.25471925735473633, 25: 0.25471873581409454, 30: 0.25471876142546535,
  35: 0.25471875989751425, 40: 0.2547187598270284,  45: 0.2547187598258205,
  50: 0.2547187598258627,  55: 0.25471875982586095
}
```

```
eigenvalue 1:
{
  5: 1.796875,              10: 1.82373046875,        15: 1.8227386474609375,
  20: 1.8227171897888184,   25: 1.8227171152830124,   30: 1.8227170803584158,
  35: 1.8227170808677329,   40: 1.8227170808881965,   45: 1.8227170808871307,
  50: 1.8227170808871063,   55: 1.822717080887108,    60: 1.822717080887108
}
eigenvalue 2:
{
  5: 3.203125,              10: 3.17626953125,        15: 3.1772613525390625,
  20: 3.1772828102111816,   25: 3.1772828847169876,   30: 3.177282919641584,
  35: 3.177282919132267,    40: 3.1772829191118035,   45: 3.1772829191128693,
  50: 3.1772829191128937,   55: 3.1772829191128915,   60: 3.1772829191128915
}
eigenvalue 3:
{
  5: 4.765625,              10: 4.74365234375,        15: 4.7452545166015625,
  20: 4.745280742645264,    25: 4.7452812641859055,   30: 4.745281238574535,
  35: 4.745281240102486,    40: 4.745281240172972,    45: 4.7452812401741795,
  50: 4.745281240174137,    55: 4.745281240174139,    60: 4.745281240174139
}
```

对于本题中所给的求本征值的例子，仔细分析图 3，不难得到这样的结论：

- Jacobi 迭代法是高效且非常稳定的：尽管在我的实现中，Jacobi 迭代的每一步都要对所有的非对角元进行 Givens 变换，但仅仅经过三次迭代，我们就得到了非常精确的四个本征值。这与 Jacobi 迭代适用于较小的绸密实对称矩阵的经验理论是一致的。
- 对于与其它本征值相差较大的本征值（如本题中最小的本征值），QR 迭代法的收敛速度较快，这与 QR 迭代法收敛速度的理论分析是一致的。
- Sturm 算法是稳定且非常高效的，对求解各个本征值而言没有差异：Sturm 算法基于二分法，其收敛速度是线性的，求解结果也是稳定的。此外，这种方法虽然看上去迭代次数较多，但每次迭代所需的计算量都非常小，它其实是十分高效的。

III. 一维环形原子链的振动

本题中 n ($n \geq 3$) 个粒子的一维环形原子链振动方程为：

$$\ddot{x} + Ax = 0,\ A_{n\times n} = \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & 0 & -1 \\ -1 & 2 & -1 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2 & -1 & 0 \\ 0 & 0 & 0 & \cdots & -1 & 2 & -1 \\ -1 & 0 & 0 & \cdots & 0 & -1 & 2 \end{pmatrix} \tag{1}$$

约定 $x_0 := x_n,\ x_{n+1} := x_1$，则对于 $\forall i = 1, 2, \cdots, n,\ \forall j = 1, 2, \cdots, n$：

$$A(i;j) = -\delta_{i-1,j} + 2\delta_{i,j} - \delta_{i+1,j} \tag{2}$$

令 $x(t) = x(0)\mathrm{e}^{-\mathrm{i}\omega t} \Rightarrow \ddot{x} = -\omega^2 x \Rightarrow -\omega^2 x + Ax = 0 \Rightarrow Ax = \omega^2 x$，这说明 x 满足 A 的本征方程，本征值为 ω^2。由于 A 为实对称矩阵，$\omega^2 \in \mathbb{R}$，且 A 在数域 K ($K = \mathbb{R}, \mathbb{C}$) 上的线性空间 K^n 中的本征矢 $\xi_1, \xi_2, \ldots, \xi_n$ 构成 K^n 的一组基底，它们对应的本征值分别记为 $\lambda_1, \lambda_2, \ldots, \lambda_n$（题中已假设 $|\lambda_1| > |\lambda_2| \geq \ldots |\lambda_n|$）。对于任意单位矢量 $q_0 \in \mathbb{C}^N$，设

$$q_0 = \sum_{i=1}^{n} C_{i0}\xi_i,\ C_{i0} \neq 0 \tag{3}$$

按照迭代关系有

$$\begin{aligned} q_k &= \frac{z_k}{|z_k|} = \frac{Aq_{k-1}}{|z_k|} = \frac{Az_{k-1}}{|z_k||z_{k-1}|} = \frac{A^2 q_{k-2}}{|z_k||z_{k-1}|} = \cdots = \frac{A^k q_0}{|z_k||z_{k-1}|\cdots|z_1|} \\ &= \frac{A^k \sum_{i=1}^{n} C_{i0}\xi_i}{|z_k||z_{k-1}|\cdots|z_1|} = \frac{\sum_{i=1}^{n} C_{i0}A^k\xi_i}{|z_k||z_{k-1}|\cdots|z_1|} = \frac{\sum_{i=1}^{n} C_{i0}\lambda_i^k\xi_i}{|z_k||z_{k-1}|\cdots|z_1|} = \lambda_1^k \frac{\sum_{i=1}^{n} C_{i0}(\lambda_i/\lambda_1)^k\xi_i}{|z_k||z_{k-1}|\cdots|z_1|} \end{aligned} \tag{4}$$

所以

$$\begin{aligned} \lim_{k\to+\infty} q_k &= \lim_{k\to+\infty} \lambda_1^k \frac{\sum_{i=1}^{n} C_{i0}(\lambda_i/\lambda_1)^k\xi_i}{|z_k||z_{k-1}|\cdots|z_1|} = \left(\lim_{k\to+\infty} \frac{\lambda_1^k}{|z_k||z_{k-1}|\cdots|z_1|}\right) \sum_{i=1}^{n} C_{i0} \lim_{k\to+\infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k \xi_i \\ &= \lim_{k\to+\infty} \frac{\lambda_1^k}{|z_k||z_{k-1}|\cdots|z_1|} C_{10}\xi_1 \end{aligned} \tag{5}$$

当上式右边的极限存在，即迭代收敛时，q_k 最终将趋向于平行于 A 的模最大的本征值对应的本征矢 ξ_1，即也趋向于 A 的模最大的本征值对应的一个本征矢。继而我们有

$$\begin{aligned} \lim_{k\to+\infty} v_k &= \lim_{k\to+\infty} (q_k, Aq_k) = (\lim_{k\to+\infty} q_k, A \lim_{k\to+\infty} q_k) = (\lim_{k\to+\infty} q_k, \lambda_1 \lim_{k\to+\infty} q_k) \\ &= \lambda_1 (\lim_{k\to+\infty} q_k, \lim_{k\to+\infty} q_k) = \lambda_1 \lim_{k\to+\infty} (q_k, q_k) = \lambda_1 \lim_{k\to+\infty} 1 = \lambda_1 \end{aligned} \tag{6}$$

这一极限的存在性由式 (5) 中的极限存在保证。

现在考虑 $n = 10$ 的例子。在源文件 vibra/vibra.c 中，我将 q_0 初始化为各分量绝对值差别不太大的随机单位向量，在一次运行中，迭代至相邻两次本征值相差绝对值和本征矢各分量平均平方差别的平方根同时收敛到机器精度以下时，迭代次数，得到的模最大的本征值对应的本征矢和这个本征值分别为：

[329]

+0.316228	+0x1.43d13624848efp−2
−0.316228	−0x1.43d13624848dep−2
+0.316228	+0x1.43d13624848dfp−2
−0.316228	−0x1.43d13624848f3p−2
+0.316228	+0x1.43d1362484912p−2
−0.316228	−0x1.43d136248492fp−2
+0.316228	+0x1.43d1362484940p−2
−0.316228	−0x1.43d136248493fp−2
+0.316228	+0x1.43d136248492bp−2
−0.316228	−0x1.43d136248490cp−2
+4.000000	+0x1.0000000000000p+2

这与其对应的解析结果：$q_{+\infty} = (1, -1, 1, -1, \ldots, 1, -1)/\sqrt{10},\ v_{+\infty} = 4$ 非常一致地符合。

此外，为了验证该算法在统计上的相对稳定性，图 5 作出了 1000 次重复计算的迭代次数分布，可以看到，对于不同的随机初始值，收敛到机器精度所需的迭代次数基本上是稳定的。

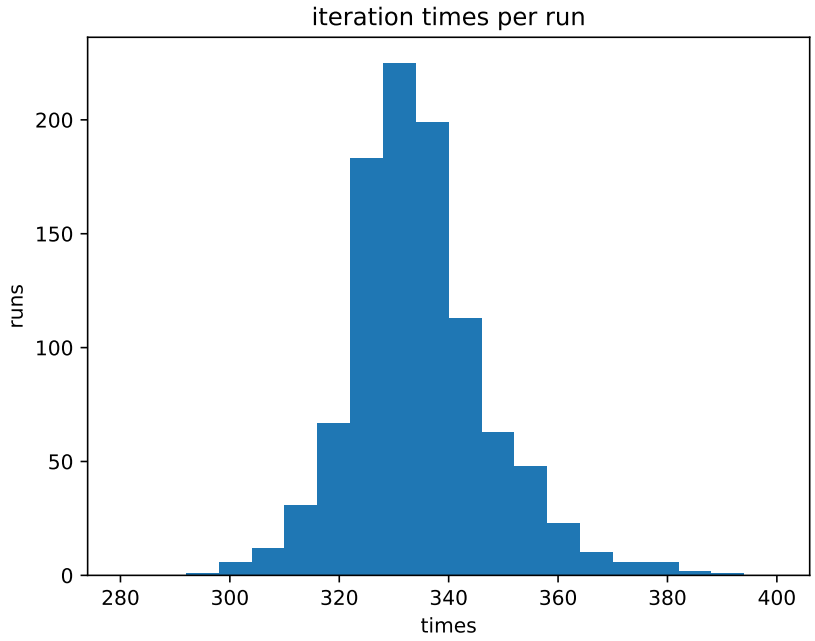

图 5: 1000 次不同随机初始值至收敛至机器精度所需的迭代次数统计图

IV. 洛仑兹吸引子

我在源文件 lorenz/lorenz.c 中实现了将自变量取值区间 $[t_0, t_n]$ 等分为 n 段后，对于给定的洛仑兹方程参数 (σ, ρ, β) 和初值条件 $y_0 = (y_{10}, y_{20}, y_{30})$，使用向前差分显式表达式，将各格点 t_0，t_2，…，t_n 上的函数值 $y(t)$ 和导数值 $y'(t)$ 求出的过程。我与课件中一样，令 $t_0 = 0$，$t_n = 10$，$n = 100000$，$y_0 = (12, 4, 0)$，改变 (σ, ρ, β)，得到图 6 中所示的结果。

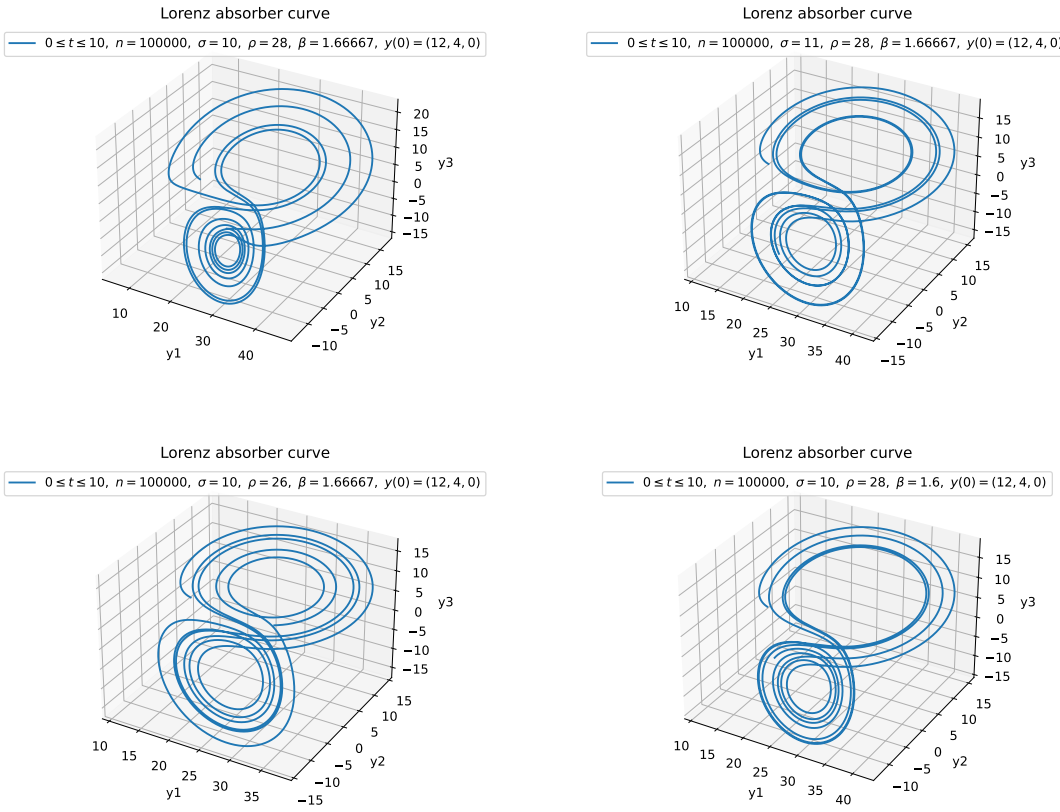

图 6: 洛仑兹吸引子曲线

图 6 中的四组曲线均呈现出有界、无周期、不收敛、不自交，轨道混乱地围绕两个不动点的特征，是典型的吸引子。四图中参数 (σ, ρ, β) 均有明显不同，但曲线整体特征仍然保持，这说明洛仑兹吸引子在本题中所取参数的邻域内对参数的变化不是特别敏感，这与洛仑兹吸引子高度敏感于初值变化的一般特征相反。当 $(\sigma, \rho, \beta) = (0, 0, 0)$ 时，曲线形状也非常特别，如图 7 所示。在这一情形下，不动点从两个退化到了一个。

A 项目源代码

参见 `https://github.com/lyazj/numphy-04`。

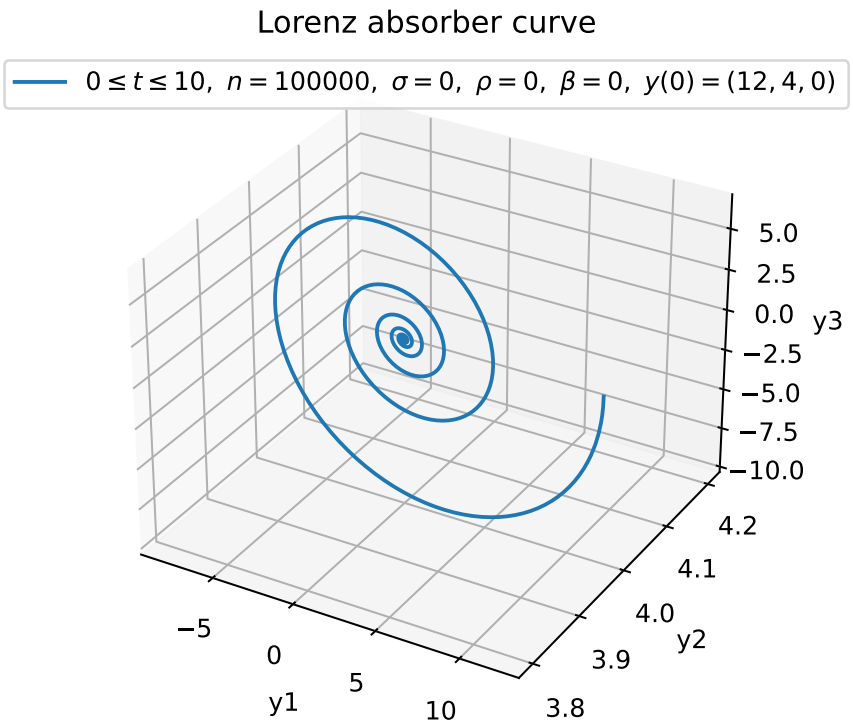

图 7: 零参数洛仑兹吸引子曲线