

第一版

数值计算方法与实验 II



张 强

2013 年 8 月，南京

南京大学数学系
中国南京



前言

此文档是做为南京大学数学系计算与信息专业《数值计算方法 II》的配套讲义。内容主要是一些典型的线性方程以及非线性方程的数值求解方法。它们是数值计算的基础问题之一，很多实际问题的数值模拟最后都转化为这些问题的数值求解。通过本学期的学习，希望学生深入了解数值计算的本质目标，以及一些常用的数值技巧与数值分析工具。

本学期的内容基本是封闭的，仅需要学生具备数学分析和高等代数的基础知识。对一些过于复杂的结论和证明，我们会给出相应的描述和补充。主要参考文献有

- ☒ 林成森, 数值计算方法 (第二版), 科学出版社, 2005
- ☒ 李大明, 数值线性代数, 清华大学出版社, 2010
- ☒ 蔡大用, 数值代数, 清华大学出版社, 2005
- ☒ 徐树方, 高立, 张平文, 数值线性代数, 北京大学出版社, 2010
- ☒ 威尔金森, 代数特征值问题, 石钟慈, 邓建新译, 科学出版社, 2001
- ☒ 李庆扬, 王能超, 易大义, 数值分析, 清华大学出版社, 2010
- ☒ 李庆扬, 关治, 白峰杉, 数值计算原理, 清华大学出版社, 2009

目录

第一章	线性方程组的直接解法	1
1.1	高斯消去法	1
1.1.1	基本过程	1
1.1.2	基本技巧	2
1.1.3	Gauss 变换阵	3
1.1.4	逆矩阵的计算	4
1.2	直接三角解法	5
1.2.1	矩阵三角分解	5
1.2.2	矩阵三角分解的应用	5
1.3	向量和矩阵范数	8
1.3.1	向量和矩阵范数的定义	9
1.3.2	矩阵范数不等式	10
1.4	摄动分析	10
1.4.1	条件数	10
1.4.2	摄动理论	11
1.4.3	精度分析	11
1.4.4	列主元高斯消去法的数值稳定性分析	12
1.5	数值实验	13
第二章	线性方程组的迭代解法	15
2.1	迭代法的基本理论	15
2.1.1	不动点迭代的基本结构	15
2.1.2	不动点迭代的收敛性分析	15
2.1.3	停机标准	17
2.2	两个古典迭代算法	17
2.3	逐次超松弛迭代法	19
2.4	迭代加速	21
2.4.1	外推方法	21
2.4.2	半迭代方法	21
2.5	共轭斜量法	23
2.5.1	等价的极值问题与求解	24
2.5.2	共轭斜量系的构造	24
2.5.3	收敛性分析	26
2.5.4	预处理共轭斜量方法	26
2.6	数值实验	27

第三章 线性最小二乘问题	28
3.1 基本概念	28
3.1.1 最小二乘解	28
3.1.2 满秩分解表示	28
3.1.3 矩阵的广义逆	28
3.1.4 数值算法	29
3.2 矩阵的直交分解	30
3.2.1 Gram-Schmidt 直交化方法	30
3.2.2 Householder 变换	31
3.2.3 Givens 变换	33
3.2.4 小结	33
3.3 最小二乘解的各种表示	34
3.3.1 Gram-Schmidt 直交化方法	34
3.3.2 直交矩阵变换	35
3.4 奇异值分解	35
3.5 离散数据拟合	37
3.6 数值实验	37
第四章 矩阵特征值问题	38
4.1 特征值问题的估计及其敏感度分析	38
4.1.1 预备知识	38
4.1.2 特征值界定与排除定理	39
4.1.3 敏感度分析	39
4.2 幂法	40
4.2.1 正幂法	40
4.2.2 加速技巧	41
4.2.3 反幂法	41
4.2.4 其他特征值的求解	42
4.3 实对称矩阵的 Jacobi 方法	43
4.3.1 基本思想	43
4.3.2 古典 Jacobi 方法	44
4.3.3 循环 Jacobi 方法	45
4.3.4 特征向量的计算	45
4.4 对称矩阵的 Givens-Householder 方法	45
4.4.1 三对角化策略	45
4.4.2 三对角对称矩阵的二分法	46
4.4.3 特征向量的确定	47
4.5 QR 方法	47
4.5.1 基本思想	47
4.5.2 数值实现	47
4.5.3 隐式对称 QR 方法	48
4.5.4 双重位移的 QR 方法	49
4.6 数值实验	49

第五章	非线性方程（组）的数值方法	51
5.1	基本概念	51
5.2	标量方程的数值求解	51
5.2.1	区间分半法	52
5.2.2	不动点迭代	52
5.2.3	加速迭代收敛	52
5.2.4	Newton 方法	53
5.2.5	割线法	53
5.2.6	实多项式的实根计算	54
5.3	方程组的数值求解	54
5.3.1	预备知识	55
5.3.2	Newton 法	55
5.3.3	Newton 法的改进	56
5.3.4	极值算法	58
5.4	数值实验	59

第一章 线性方程组的直接解法


线性方程组的直接解法也称为准确解法，在没有舍入误差的情况下，经有限次数的四则运算（可能包括少量的开方运算），就可以得到问题的准确解。这方面的内容与利用初等变换求解线性方程组的内容有密切的联系。本章的数值方法主要集中在高斯消去方法及其各种等价变形在数字计算机上的实现过程，包括：(a) 程序设计流程和技巧；(b) 数据存储和计算复杂度的下降；以及 (c) 舍入误差的有效控制。

1.1 高斯消去法

高斯消去法是目前求解中小规模（阶数约 1000 左右）线性方程组的常用方法，特别当系数矩阵稠密且没有任何特殊结构的情形。该思想最早出现在中国秦汉时代的《九章算术》；到了 19 世纪初 (1809 年发表于 *Theoria Motus*)，西方也有了高斯消去法。然而，具有大量未知数的线性方程组求解则是在 20 世纪中叶计算机问世后才成为可能。

1.1.1 基本过程

高斯消去法是线性代数课程中的一个基本算法。其基本思想是逐次消去未知数，将要求解的方程组转化为同解的三角形方程组。这个过程等价于对增广矩阵 $[A|b]$ 做所谓的初等行变换，即左乘一系列的（特别是第三种）初等变换矩阵，将增广矩阵转化为上梯形矩阵。

 论题 1.1. 顺序高斯消去算法。

数值关注点是如何在计算机上自动实现这个精确算法，并得到理想的数值结果。高斯消去法在 Matlab 软件中的命令是 $A \setminus b$ 。

```
1. For  $k = 1, 2, \dots, n$ , Do
2.   For  $i = k + 1, \dots, n$ , Do
3.      $a_{ik} := a_{ik} / a_{kk}$ ;
4.   Enddo
5.   For  $j = k + 1, \dots, n$ , Do
6.     For  $i = k + 1, \dots, n$ , Do
7.        $a_{ij} := a_{ij} - a_{ik} a_{kj}$ ;
8.     Enddo
9.   Enddo
10. Enddo
```

在左侧的图文框，我们给出高斯消去法的一个拟程序代码。其中符号“ $:=$ ”表示所谓的赋值运算。这样的代码蕴含着数据覆盖技术，即对角线下方存储的数据是所有的乘子，而其余元素是高斯消元后的上三角矩阵。数据存储单元的元素变化是数值方法中的一个重点。

实际计算中，我们可以把右端项 b 放在矩阵 A 的右侧，形成所谓的增广矩阵，并在算法中的第 5 步中扩充 j 的取值范围到 $n + 1$ 。欲得到最后的答案，我们还需求解对应右上角矩阵的三角形方程组。这需要约 $O(n^2)$ 的

回代计算过程，具体代码此处省略。

★ 说明 1.1. 算法的代码编写对算法的实现效率有很大的影响，譬如多重循环的嵌套方式以及数据的读写效率。它们的改进强烈依赖所使用的计算环境。

不同语言的代码编写应稍有改变。上述代码中的三重循环次序是 $k-j-i$ ，它特别适用于 Fortran 语言，因其表示矩阵的二维数组是按列连续存放的。而 C++ 语言中表示矩阵的二维数组是按行连

续存放，最内层循环在内存中不连续第读取，从而影响算法的性能，此时有必要交换最内两层的循环次序。

上述代码还未考虑计算机的硬件结构，处理大规模数据的能力和效率很低。其基本操作是数与数之间的运算，每次运算都到进行一次数据读写。称这样的代码处于 *BLAS-1* 的运算级别ⁱ。实际上，数值算法还可以采用更高的 *BLAS* 运算级别来实现，譬如 *BLAS-2* 主要基于向量与向量乘积（含矩阵与向量乘积）的块数据操作。借用 *Matlab* 语言，顺序高斯消去法的 *BLAS-2* 版本为：

```

1. For  $k = 1, 2, \dots, n$ , Do
2.    $A(k+1:n, k) := A(k+1:n, k)/A(k, k)$ ;
3.    $A(k+1:n, k+1:n) := A(k+1:n, k+1:n) - A(k+1:n, k)A(k, k+1:n)$ ;
4. Enddo

```

最高的级别是 *BLAS-3*，它基于矩阵与矩阵乘积的块数据操作，更多地涉及到所谓并行计算的内容。因其超出本课程的内容，故在此不做讨论。

★ 说明 1.2. 计算复杂度可以通过整个计算过程的乘除法运算次数来衡量。在高斯消去法中，总次数为 $\sum_{k=1}^{n-1} (n-k)(n-k+1) = O(n^3/3)$ 。

注意到算法中的除法运算，需讨论算法是否可执行到底？

定理 1.1. 若顺序高斯消元法过程中的对角线元素 $a_{kk}^{(k)}, k = 1, 2, \dots, n-1$ ，均不为零，则消元过程可执行到底。其充分必要条件是各阶顺序主子阵都是非奇异的。若所有对角线元素均不为零，则回代过程也可执行。

对某些特殊类型的线性方程组，我们有明确的结论。

定理 1.2. 若系数矩阵 A 是对称正定的，则顺序高斯消元法可执行到底，且每一步的中间矩阵的元素绝对值不超过原矩阵元素的最大值。

证明：见后面的矩阵 Cholesky 分解部分。 □

以上的讨论与高等代数中所讲的初等变换求解线性方程组是一致的，还没有涉及到计算机近似计算的问题。舍入误差的逐渐积累会使最终计算结果偏离了问题的真解ⁱⁱ。

1.1.2 基本技巧

由于舍入误差，顺序高斯消元法会导致计算结果出现严重的偏差或错误。这通常与机器位长（机器精度）有关。譬如，在三位有效数字十进制计算机，用顺序高斯消元法求解如下的线性方程组。数据流的变化是

$$\begin{bmatrix} 0.001 & 1.00 & 1.00 \\ 1.000 & 2.00 & 3.00 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.001 & 1.00 & 1.00 \\ 1000 & -1000 & -1000 \end{bmatrix}.$$

回代可得数值解 $x_{\text{num}} = (0.00, 1.00)^\top$ ，与精确解 $x_\star = (1.002 \cdots, 0.998 \cdots)^\top$ 相去甚远。但是，若计算机的有效位长增加，问题可以得到很好的求解。

因此说，在现有的计算机条件下，对舍入误差进行精细的控制，尽可能地减小舍入误差的积累是一个非常必要的工作。在高斯消去法中，一个简单易行的有效方法是选取所谓的高斯消元主元，即当前对角线位置右下方所有元素中按绝对值最大的某个元素。常用的策略有列主元/全主元策略 (Wilkinson, 1961) 和 车型主元策略 (Neal 和 Poole, 1992)。出于搜索时间代价的考虑，我们更多地采用列主元策略。

ⁱBLAS=Basic Linear Algebraic Subroutine.

ⁱⁱ设计解各种问题的计算机算法，对所设计的算法做数值稳定性分析，说明它们的适用范围，以及使用这些算法的过程可能出现的各种问题等，均是计算数学区别于纯粹数学的重要特点与任务。在学习、使用和设计计算方法时应随时加以注意。

🔗 论题 1.2. 列主元高斯消去算法的实现不需要付出太大的代价。针对顺序高斯消去法的上述两个版本中，我们仅需在第 2 行代码前填入一小段补丁代码

- 选择 l 使得 $|a_{lk}| = \max_{k \leq i \leq n} |a_{ik}|$ ，交换 \mathbf{A} 的第 l 行和第 k 行；

这使得高斯消元过程中的所有消元乘子按绝对值均不超过 1，消元过程中的乘法运算所带来的舍入误差可被有效地控制。

★ 说明 1.3. 教课书还给出另一种选列主元策略，即按比例选取列主元。在每次寻求高斯消元主元前，首先对右下角矩阵的所有行向量按最大模单位化。这样的工作等价于所谓的预处理过程。

★ 说明 1.4. 计算数据的行交换过程要花费很多无真正价值的计算机数据读写时间。实际上，我们可通过一个简单的处理便可省掉此部分时间消耗。关键之处是引进一个 n 维指标向量 $\mathbf{p} = (p_1, p_2, \dots, p_n)$ 来记录整个消元过程中的行交换信息，其初始状态为自然序列。若高斯消去过程中有第 k 行与第 r 行的交换，则指标向量 \mathbf{p} 中的第 k 分量与第 r 个分量进行轮换即可。

★ 说明 1.5. 实际计算经验表明：列主元高斯消去法与全主元高斯消去法在数值稳定性方面完全可以媲美，但主元搜索时间却大大减少。因此，列主元高斯消去法已成为目前求解中小型稠密线性方程组最受欢迎的方法之一。

1.1.3 Gauss 变换阵

为便于数值算法的理论研究和分析，我们可将高斯消去过程用矩阵语言来描述，即矩阵的三角化过程。这是数值代数中非常重要的研究内容，其核心问题是：

已知 m 维非零向量 $\mathbf{a} = (a_1, a_2, \dots, a_m)$ ，如何构造一个简单矩阵左乘向量 \mathbf{a} 后可将其转化为仅首个位置非零的向量？

这里，我们假设 $a_1 \neq 0$ 。所谓的简单矩阵是指某种初等矩阵，通常是单位矩阵的某种秩一修正矩阵。实现该方法有很多中。

🔗 论题 1.3. 记 $\hat{\mathbf{e}} = (1, 0, 0, \dots, 0)^\top$ 为首个位置为 1 的 m 维单位向量。定义 Gauss 消元阵

$$\mathbf{S}_{m \times m} = \mathbf{I}_{m \times m} - \hat{\mathbf{p}} \hat{\mathbf{e}}^\top, \quad (1.1.1)$$

其中 $\hat{\mathbf{p}} = (0, a_2/a_1, a_3/a_1, \dots, a_m/a_1)^\top$ 是一个已知的 m 维列向量。

上述目标可扩展到 n 维向量 $\mathbf{a} = (a_{1k}, a_{2k}, \dots, a_{kk}, \dots, a_{nk})^\top$ 的高斯消元，即利用非零分量 a_{kk} 将其下方所有元素全部清零。

🔗 定义 1.1. 令 $\ell_k = (0, 0, \dots, 0, \ell_{k+1,k}, \ell_{k+2,k}, \dots, \ell_{n,k})^\top$ ，其中 $\ell_{ik} = a_{ik}/a_{kk}$ 。对应的高斯消元矩阵可描述为

$$\mathbf{L}_k^{-1} = \mathbf{I} - \ell_k \mathbf{e}_k^\top. \quad (1.1.2)$$

这里的 \mathbf{e}_k 是第 k 个标准单位向量。更方便的描述是所谓的单位矩阵扩张，即

$$\mathbf{L}_k^{-1} = \begin{bmatrix} \mathbf{I}_{(k-1) \times (k-1)} & \mathbf{O} \\ \mathbf{O} & \mathbf{S}_{(n-k+1) \times (n-k+1)} \end{bmatrix}.$$

这种默认的扩张方式将在本课程中多次使用，以后不再赘述。

🔗 论题 1.4. 注意到定义 1.1 中的 ℓ_k 恰好由第 k 步高斯消元过程中的对应乘子组成, 高斯消去法的第 k 步操作可描述为这个高斯消元阵的左乘。因此, 顺序高斯消去可以用如下矩阵语言描述: 记 $\mathbb{A}^{(1)} = \mathbb{A}, \mathbf{b}^{(1)} = \mathbf{b}$.

$$\mathbb{L}_{n-1}^{-1} \cdots \mathbb{L}_2^{-1} \mathbb{L}_1^{-1} \mathbb{A}^{(1)} = \mathbb{A}^{(n)}, \quad \mathbb{L}_{n-1}^{-1} \cdots \mathbb{L}_2^{-1} \mathbb{L}_1^{-1} \mathbf{b}^{(1)} = \mathbf{b}^{(n)}.$$

这种方式可以便于算法研究与理论分析。

利用高斯消元矩阵的基本性质: $\mathbb{L}_k = \mathbb{I} + \ell_k \mathbf{e}_k^\top$ 和 $\mathbb{L}_i \mathbb{L}_j = \mathbb{L}_i + \mathbb{L}_j - \mathbb{I}, (i < j)$, 这个过程衍生出一个重要的矩阵分解, 即 $\mathbb{A} = \mathbb{L}\mathbb{U}$, 其中 $\mathbb{U} = \mathbb{A}^{(n)}$ 是上三角矩阵,

$$\mathbb{L} = \mathbb{L}_1 \cdots \mathbb{L}_{n-1} = \begin{bmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n,n-1} & 1 \end{bmatrix}, \quad \text{其中 } \ell_{ij} = a_{ij}^{(j)} / a_{jj}^{(j)},$$

是单位下三角矩阵。这个矩阵分解表示在矩阵理论中非常基础和重要。类似工作将在下节做进一步的深入研究。

🔗 论题 1.5. 列主元的选取可描述为初等排列阵的左乘运算。从而, 列主元高斯消去算法可做如下矩阵描述:

$$\begin{aligned} \mathbb{U} = \mathbb{A}^{(n)} &= \mathbb{L}_{n-1}^{-1} \mathbb{I}_{n-1, r_{n-1}} \cdots \mathbb{L}_2^{-1} \mathbb{I}_{2, r_2} \mathbb{L}_1^{-1} \mathbb{I}_{1, r_1} \mathbb{A}^{(1)} \\ &= \underbrace{\tilde{\mathbb{L}}_{n-1}^{-1} \tilde{\mathbb{L}}_{n-2}^{-1} \cdots \tilde{\mathbb{L}}_2^{-1} \tilde{\mathbb{L}}_1^{-1}}_{\mathbb{L}^{-1}} \underbrace{\mathbb{I}_{n-1, r_{n-1}} \cdots \mathbb{I}_{2, r_2} \mathbb{I}_{1, r_1}}_{\mathbb{P}} \mathbb{A}^{(1)}. \end{aligned}$$

其中 \mathbb{L}_k^{-1} 是列主元交换后对应的高斯消元矩阵, \mathbb{P} 是所谓的置换阵。这里

$$\tilde{\mathbb{L}}_k^{-1} = \mathbb{I}_{n-1, r_{n-1}} \cdots \mathbb{I}_{k+1, r_{k+1}} \mathbb{L}_k^{-1} \mathbb{I}_{k+1, r_{k+1}} \cdots \mathbb{I}_{n-1, r_{n-1}}. \quad (1.1.3)$$

具有与 \mathbb{L}_k^{-1} 几乎相同的非零元素分布, 仅仅是高斯消元乘子的位置不同。因此, 矩阵 \mathbb{L} 仍具有单位下三角结构。

1.1.4 逆矩阵的计算

利用高斯消去法求解一系列的线性方程组也可以求解矩阵的逆矩阵, 但需要的乘除法次数较高。为减少计算复杂度, 我们可采用所谓的 Gauss-Jordan 消元算法。这是一个古老的算法, 它由测量技师 Wilhelm Jordan(1842-1899) 和 B.I. Clasen(1987) 分别独立提出。

在 Matlab 软件可用命令 `inv()` 计算逆矩阵。

🔗 论题 1.6. Gauss-Jordan 算法: 用对角元素消去同列的其他所有元素。针对不同的数据存储方式, 它的程序实现可以有两种方式。若节省存储空间, 我们可采用数据覆盖技术。相应的算法如下:

```

1. For  $k = 1, 2, \dots, n$ , Do
2.   交换  $\mathbb{A}$  的第  $k$  行和第  $p_k$  行, 其中  $p_k$  为相应的列主元;
3.    $a_{kk} = 1/a_{kk}$ ;
4.   For  $i = 1, \dots, n$  且  $i \neq k$ , Do  $a_{ik} := -a_{ik}a_{kk}$ ; Enddo
5.   For  $i = 1, \dots, n$  且  $i \neq k$ , Do
6.     For  $j = 1, \dots, n$  且  $j \neq k$ , Do  $a_{ij} := a_{ij} + a_{ik}a_{kj}$ ; Enndo
7.   Enddo
8.   For  $j = 1, \dots, n$  且  $j \neq k$ , Do  $a_{kj} := a_{kj}a_{kk}$ ; Enndo
9. Enddo
10. For  $k = n, n-1, \dots, 1$ , Do
11.   交换  $\mathbb{A}$  的第  $k$  列和第  $p_k$  列;
12. Enddo

```

该算法总共需 $O(n^3/2)$ 次乘除法运算。

逆矩阵的求解方法还有很多, 如 Newton 迭代法 $\mathbb{X}_{k+1} = 2\mathbb{X}_k(\mathbb{I} - \mathbb{A}\mathbb{X}_k)$ 。因课程时间限制, 此处不做赘述。

1.2 直接三角解法

本节介绍高斯消去法的不同实现方式。若所有计算均是准确的, 它们与高斯消去法是完全等价的。下面我们主要关注系数矩阵的处理, 其设计出发点是各种版本的矩阵三角分解。

1.2.1 矩阵三角分解

🔗 定义 1.2. 若存在某个上三角矩阵 \mathbb{U} 和某个下三角矩阵 \mathbb{L} 使得 $\mathbb{A} = \mathbb{L}\mathbb{U}$, 则称矩阵 \mathbb{A} 有三角分解。若 \mathbb{L} 为单位下三角阵, 相应的三角分解称为 Doolittle 分解; 若 \mathbb{U} 为单位上三角阵, 相应的三角分解称为 Crout 分解。

★ 说明 1.6. 不是所有的矩阵都有 LU 分解, 如 $\mathbb{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ 。

定理 1.3. 若 n 阶矩阵 \mathbb{A} 的顺序主子式 $\mathbb{A}_1, \mathbb{A}_2, \dots, \mathbb{A}_{n-1}$ 均非奇异, 则顺序 Gauss 消去法将给出矩阵 \mathbb{A} 的 Doolittle 分解。

🔗 定义 1.3. 有时, 我们还需考虑另一种三角分解 $\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{R}$, 其中 \mathbb{D} 为对角阵, \mathbb{R} 和 \mathbb{L} 分别为上下单位三角矩阵。进而 Doolittle 分解和 Crout 分解唯一。

定理 1.4. n 阶矩阵 \mathbb{A} 有唯一的 LDR 三角分解, 当且仅当顺序主子式 $\mathbb{A}_1, \mathbb{A}_2, \dots, \mathbb{A}_{n-1}$ 均非奇异。

证明: 数学归纳法与矩阵分块。 □

★ 说明 1.7. 定理 1.4 中的条件仅是一个充分条件。即使顺序主子式的条件不成立, 矩阵也可以有 LU 分解。譬如,

$$\begin{bmatrix} 0 & 0 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \quad (1.2.4)$$

请问这个矩阵的 LU 分解是唯一的吗?

1.2.2 矩阵三角分解的应用

基于矩阵三角分解的算法称为直接三角解法。算法的推导很简单, 均基于单位上(下)三角阵的特点以及基本的矩阵元素计算公式

$$a_{ij} = \sum_{r=1}^{\min(i,j)} \ell_{ir} u_{rj}.$$

在本质上，直接三角解法仍属于高斯消去法，但计算的流程或次序是不同的。它不必计算消元过程中的中间结果，在数据读写方面具有更大的优势。

在 Gauss 消去法中，我们每次要依次计算出 \mathbb{L} 的相应列和 \mathbb{U} 的相应行，同时更新右下角的块矩阵。不难发现，处于 (n, n) 位置的数据将被更新约 n 次，大量的数据需要使用，会耗费很多的数据读写时间。与高斯消去法不同，直接三角分解算法只在需要更新某个位置才进入这个位置。所以说，直接三角分解是一种“需求驱动”的算法。

Crout 分解

Doolittle 算法就是顺序（或列主元）高斯消去法的不同实现过程而已。若计算过程是精确的，它们计算所得的结果是完全一致的。在 Matlab 软件中的命令是 `lu()`。

```

1. For  $k = 1, 2, \dots, n$ , Do
2.   For  $i = k, k + 1, \dots, n$ , Do
3.      $a_{ik} := a_{ik} - \sum_{r=1}^{k-1} a_{ir} a_{rk}$ ;
4.   Enddo
5.   For  $j = k + 1, k + 2, \dots, n$ , Do
6.      $a_{kj} := (a_{kj} - \sum_{r=1}^{k-1} a_{kr} a_{rj}) / a_{kk}$ ;
7.   Enddo
8. Enddo

```

下面，我们以稍有区别的 Crout 方法为例，介绍矩阵的直接分解算法。在 Crout 算法中，目标数据被先列后行地逐一更新，整体以瀑布型方式向右下角流动。

这里的伪代码依旧采用了数据覆盖技术。若求和符号中的上标小于下标，则对应的操作（第 3 行和第 6 行）默认为空操作，其值默认为零。这个默认准则将在本教程中一直使用，以后将不再提及。

若在 Crout 算法中引进列主元策略，只需在上述伪代码的第 5 行前添加相应的补丁。

Cholesky 分解

若 \mathbb{A} 是实对称正定矩阵，数值的计算复杂度可进一步降低。著名的 Cholesky 分解ⁱⁱⁱ 或 $\mathbb{L}\mathbb{L}^\top$ 算法是直接三角解法的典型代表，具有重要的理论和应用价值。该算法可用来判断一个对称矩阵的正定性。

定理 1.5. 设 \mathbb{A} 是实对称正定矩阵，则有三角分解 $\mathbb{A} = \mathbb{L}\mathbb{L}^\top$ ，其中 \mathbb{L} 是下三角矩阵。若要求 \mathbb{L} 的对角线元素均为正数时，这个三角分解是唯一的。

🔗 论题 1.7. 基本 Cholesky 分解算法的实现基于矩阵元素计算的基本公式

$$a_{ij} = \sum_{k=1}^j \ell_{ik} \ell_{kj}, \quad i \geq j.$$

对角线元素的计算需要开根号运算，因此 Cholesky 算法也称为平方根法。

Cholesky 算法可以 (a) 按列方向进行，或者 (b) 按行方向进行。我们重点讨论前者，即按列方向进行。若按行方向进行分解，其并行实现不容易，故很少使用。

```

1. For  $j = 1, 2, \dots, n$ , Do
2.    $a_{jj} := \left( a_{jj} - \sum_{k=1}^{j-1} a_{jk}^2 \right)^{1/2}$ ;
3.   For  $i = j + 1, j + 2, \dots, n$ , Do
4.      $a_{ij} := (a_{ij} - \sum_{k=1}^{j-1} a_{ik} a_{jk}) / a_{jj}$ ;
5.   Enddo
6. Enddo

```

按列方向进行的算法伪代码在左侧的图文框中给出。这里，我们依旧采用了数据覆盖技术，即 ℓ_{ij} 存放在 a_{ij} 的位置。

这种按列方向分解的实现方法也称为“向左看”算法。直到第 j 步时，第 j 列数据才被更新，所

ⁱⁱⁱ André-Louis Cholesky 是一个法国军官，潜心于测地学研究，勘测过希腊克里特岛和北非。

以该算法也称为“需求驱动”的算法或“延迟更新”的算法。

📌 论题 1.8. 为避免 Cholesky 算法中的开根号运算消耗太多的运算时间，我们可采用修正的平方根算法，即 LDL^T 算法。它基于三角分解 $A = LDL^T$ ，其中 L 是单位下三角阵， D 是正数构成的对角阵。基本计算公式为

$$a_{ij} = \sum_{k=1}^j \ell_{ik} d_k \ell_{kj}, \quad i \geq j.$$

下面给出的伪代码采用了按行方向进行的修正平方根算法，我们可依旧使用数据覆盖技术。左侧的基本算法虽然有效的避免了开根号运算，但乘除法的总次数却比原始的平方根法增加了一倍。

```

1. For  $i = 1, 2, \dots, n$ , Do
2.   For  $j = 1, 2, \dots, i-1$ , Do
3.      $a_{ij} := (a_{ij} - \sum_{k=1}^{j-1} a_{ik} a_{kk} a_{jk}) / a_{jj}$ ;
4.   Enddo
5.    $a_{ii} := a_{ii} - \sum_{k=1}^{i-1} a_{ik} a_{kk} a_{ik}$ .
6. Enddo

```

```

1. For  $i = 1, 2, \dots, n$ , Do
2.   For  $j = 1, 2, \dots, i-1$ , Do
3.      $a_{ij} := a_{ij} - \sum_{k=1}^{j-1} a_{ik} a_{jk}$ ;
4.   Enddo
5.   For  $j = 1, 2, \dots, i-1$ , Do
6.      $c := a_{ij}$ ;  $a_{ij} := a_{ij} / a_{kk}$ ;
7.      $a_{ii} := a_{ii} - c a_{ij}$ ;
8.   Enddo
9. Enddo

```

为减少这个算法中的重复运算，我们可引进中间辅助变量 $g_{ij} = a_{ij} d_{jj}$ ，它对应右侧修正算法中的变量 c 。这些工作是数值方法中减少计算复杂度研究的典型代表。在 Matlab 软件中的命令分别是 `chol()` 和 `ldl()`。

定理 1.6. 若矩阵 A 是对称正定的，则 $\ell_{ii} \leq \sqrt{a_{ii}}$ 。因此说，上述三个版本的算法均是数值稳定的，我们可以不进行列主元的选取。

★ 说明 1.8. 矩阵的对称正定性要求是一个非常重要的条件，它可以保证 LDL^T 算法具有良好的数值稳定性。若对称正定性不成立，问题变得有些不同。考虑

$$A = \begin{bmatrix} \varepsilon & 1 \\ 1 & \varepsilon \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \varepsilon^{-1} & 1 \end{bmatrix} \begin{bmatrix} \varepsilon & 0 \\ 0 & \varepsilon - \varepsilon^{-1} \end{bmatrix} \begin{bmatrix} 1 & \varepsilon \\ 0 & 1 \end{bmatrix}, \quad A^{-1} = \frac{1}{\varepsilon^2 - 1} \begin{bmatrix} \varepsilon & 1 \\ 1 & \varepsilon \end{bmatrix}.$$

这表明：当 $|\varepsilon| \ll 1$ 很小时，矩阵 A 是不定的， LDL^T 分解会引起很大的舍入误差，但是逆矩阵的直接计算结果却不会出现很大的舍入误差影响。因此，对于不定对称矩阵，我们常把列主元高斯消去法和二阶块高斯消去法相结合，来保证数值上的稳定性。

带状矩阵的分解

在实际的大规模数值计算中，线性方程组的系数矩阵通常包含很多的零元素，并且非零元素的分布具有一定的结构。无用的零元素占用大量的存储空间和消耗大量的零运算时间。因此，我们应充分发掘系数矩阵的稀疏特性，从存储技术和算法优化两方面进行改进。

★ 说明 1.9. 稀疏矩阵的数据存储技术是门繁杂的计算机技术。因篇幅有限，我们仅仅简要介绍一些基本技术，如三元组结构体 (i, j, a_{ij}) 表示，以及为解决数据关联搜索而引进的数据结构（如链表）等。在 Matlab 软件中的基本命令有 `sparse()`, `speye()`, `spones()`, `spdiags()`, `full()`。

★ 说明 1.10. 利用高斯消去法求解稀疏方程组，最大的困难是零元素经消元后可能会变成非零，甚至稀疏矩阵变成稠密矩阵。换言之，非零元素的填充控制是算法的关键之处，著名的方法有不完全三角（ ILU ）分解。

本教程主要讨论简单的带状矩阵，即远离对角线固定距离后的矩阵元素均为零。利用数学归纳法，不难证明带状矩阵的 LU 分解依旧具有相同的带状结构。

🔗 论题 1.9. 设矩阵的半带宽为 d ，我们可按斜线（或按行）存储技术存储矩阵，即仅仅用 $(2d-1)n$ 个存储单位替代普通的 n^2 个存储单位。请针对你的数据存储方式重写高斯消去法省略无用的零操作，并估算最终所需的乘除法次数是多少？

★ 说明 1.11. 我们需强调指出：带状矩阵的逆矩阵通常不再是带状矩阵。但是，带状矩阵的逆矩阵可能具有漂亮的结构。考虑不可约的上 Hessenberg 矩阵 \mathbb{H} ，即一个上三角矩阵加上一个所有元素非零的副对角线。Ikebe (1979) 指出逆矩阵 \mathbb{H}^{-1} 具有漂亮的结构，即存在两个列向量 $\mathbf{p} = (p_i)_{i=1}^n$ 和 $\mathbf{q} = (q_j)_{j=1}^n$ ，使得逆矩阵下三角部分的元素可表示为

$$(\mathbb{H}^{-1})_{ij} = p_i q_j, \quad i \geq j.$$

🔗 思考 1.1. 利用 Ikebe 的结果可以给出对称三对角矩阵的逆矩阵算法，其中 $q_1 = 1$ 预先被取定。

追赶法

🔗 论题 1.10. 考虑三对角矩阵 $\mathbb{A} = \text{tridiag}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ ，其中 \mathbf{a}, \mathbf{b} 和 \mathbf{c} 分别为从下至上的三条对角线向量。见下面的左侧图文框。

$$\mathbb{A} = \begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \dots & \dots & \dots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix}.$$

```

1.  $c_1 := c_1/b_1$ ;
2. For  $i = 2, 3, \dots, n$ , Do
3.      $b_i := b_i - a_i c_{i-1}$ ;
4.      $c_i := c_i/b_i$ ;
5. Enddo

```

所谓的追赶法就是 Crout 算法及其相应的两个双对角线矩阵的求解过程。右侧的图文框给出了相应的伪代码这里，其中我们依旧采用了数据覆盖策略。矩阵分解所需的乘除法次数共计 $O(2n)$ ，追赶和赶的过程需要 $O(3n)$ 次乘除法。

定理 1.7. 若三对角矩阵是严格对角占优（定义见下一章）的，则追赶法可以顺利进行到底，无需进行主元选取。

求解三对角方程组还有很多方法，如变参数追赶法（基于分解 $\mathbb{A} = \mathbb{D}\mathbb{L}\mathbb{R}$ ）和线性插值法（考虑右上角的 $n-1$ 阶子三角矩阵块）。限于篇幅，此处不再赘述。

🔗 思考 1.2. 循环三对角方程组具有重要的应用价值。所谓的循环三对角阵就是在原有三对角阵的右上角和左下角分别补充上相应的元素。请利用矩阵分解技术，给出循环三对角方程组的追赶法实现过程。

1.3 向量和矩阵范数

从本节开始，我们转向数值算法的理论分析。向量和矩阵范数是基本的度量工具；相应的定义和结论可以很容易地从实数域推广到复数域。在本课程中，我们将更多地集中探讨实数域上的代数问题。

1.3.1 向量和矩阵范数的定义

📌 定义 1.4. 向量范数的定义有三条规则: (a) 非负性; (b) 齐次性; (c) 三角不等式。相应的线性空间称为赋范空间, 其中的距离可定义为 $\text{dist}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$ 。

常用的三个 l_p 向量范数:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|, \quad \|\mathbf{x}\|_2 = \left(\sum_{i=1}^n |x_i|^2 \right)^{1/2}, \quad \|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

其中 l_2 范数也称为 Euclid (欧几里得) 范数。向量的内积满足著名的 Hölder 不等式: $|\mathbf{x}^\top \mathbf{y}| \leq \|\mathbf{x}\|_p \|\mathbf{y}\|_q$, 其中 $1/p + 1/q = 1$ 。

📌 定义 1.5. 矩阵范数的定义有四条规则: (a) 非负性; (b) 齐次性; (c) 三角不等式; 以及 (d) 相容性。

在 Matlab 软件中的命令是 `norm()`。

🔗 论题 1.11. 矩阵范数与向量范数具有密切的联系。若成立

$$\|\mathbb{A}\mathbf{x}\|_\alpha \leq \|\mathbb{A}\|_\beta \|\mathbf{x}\|_\alpha, \quad \forall \mathbf{x},$$

则称矩阵范数 $\|\cdot\|_\beta$ 相容于向量范数 $\|\cdot\|_\alpha$ 。特别地, 若能够找到某个向量使等号成立, 则称矩阵范数 $\|\cdot\|_\beta$ 从属于向量范数 $\|\cdot\|_\alpha$ 。

定理 1.8. 矩阵范数 $\|\cdot\|_\beta$ 从属于向量范数 $\|\cdot\|_\alpha$ 的必要条件是 $\|\mathbb{I}_{n \times n}\|_\beta = 1$ 。

Frobenius 范数 (又称为 Suchur 范数) 与 l_2 向量范数相容, 但它不从属于任何向量范数。

$$\|\mathbb{A}\|_F = \left(\sum_{i,j=1}^n |a_{ij}|^2 \right)^{1/2} = \left(\text{trac}(\mathbb{A}^\top \mathbb{A}) \right)^{1/2}.$$

🔗 思考 1.3. 证明 $\|\mathbb{A}\mathbb{B}\|_F \leq \|\mathbb{A}\|_F \|\mathbb{B}\|_F$ 。

定理 1.9. 对任意的矩阵范数, 均可找到某个向量范数使得两者是相容的; 反之, 对任意的向量范数, 均可导出相容 (甚至从属) 的矩阵范数, 即所谓的算子范数

$$\|\mathbb{A}\|_\alpha = \sup_{\|\mathbf{x}\|_\alpha \neq 0} \frac{\|\mathbb{A}\mathbf{x}\|_\alpha}{\|\mathbf{x}\|_\alpha} = \max_{\|\mathbf{x}\|_\alpha = 1} \|\mathbb{A}\mathbf{x}\|_\alpha.$$

这个定义可自然地推广到任意形状的矩阵。

🔗 论题 1.12. 给出常用的三个 (相容且从属的) 矩阵范数定义:

$$\|\mathbb{A}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}|, \quad \|\mathbb{A}\|_2 = \left[\varrho(\mathbb{A}^\top \mathbb{A}) \right]^{1/2}, \quad \|\mathbb{A}\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|.$$

其中 $\varrho(\mathbb{A}^\top \mathbb{A})$ 是矩阵 $\mathbb{A}^\top \mathbb{A}$ 的谱半径。

定理 1.10. 任意的两个向量 (或矩阵) 范数均是彼此等价的, 且它们关于其元素的变化都是连续的。

1.3.2 矩阵范数不等式

定理 1.11. 任意 (相容) 矩阵范数均满足 $\varrho(\mathbb{A}) \leq \|\mathbb{A}\|$; 反之, 对任意给定的正常数 ε , 至少存在一个矩阵范数 $\|\cdot\|$, 使得 $\|\mathbb{A}\| \leq \varrho(\mathbb{A}) + \varepsilon$.

定理 1.12. (Banach 引理) 若 $\|\mathbb{I}\| = 1$, 则当 $\|\mathbb{A}\| < 1$ 时, 有

$$\frac{1}{1 + \|\mathbb{A}\|} \leq \|(\mathbb{I} \pm \mathbb{A})^{-1}\| \leq \frac{1}{1 - \|\mathbb{A}\|}. \quad (1.3.5)$$

定理 1.13. 谱范数和 Frobenius 范数在 (左右) 酉变换下均保持不变。

定理 1.14. $\|\mathbb{A}\|_1 = \|\mathbb{A}^\top\|_\infty, \|\mathbb{A}^\top\|_2 = \|\mathbb{A}\|_2$.

1.4 摄动分析

1.4.1 条件数

条件数是衡量问题固有不可靠性的一个估计量。任何计算机解法要事先保证其实际可靠性超过解的固有可靠性都是无望的。

定义 1.6. 可逆矩阵 \mathbb{A} 的条件数定义为 $\kappa(\mathbb{A}) = \|\mathbb{A}\| \|\mathbb{A}^{-1}\|$. 若条件数非常大 (这是一个相对概念下的陈述), 则称矩阵 \mathbb{A} 是病态的; 否则, 称矩阵 \mathbb{A} 是良态的。

论题 1.13. 基本性质:

1. $\kappa(\mathbb{A}) \geq 1$;
2. $\kappa(c\mathbb{A}) = \kappa(\mathbb{A}), 0 \neq c = \text{const}$;
3. $\kappa(\mathbb{A}^{-1}) = \kappa(\mathbb{A})$;
4. $\kappa(\mathbb{A}\mathbb{B}) \leq \kappa(\mathbb{A})\kappa(\mathbb{B})$.

若矩阵 \mathbb{A} 是对称正定, 其谱条件数为

$$\kappa_2(\mathbb{A}) = \frac{\lambda_{\max}}{\lambda_{\min}},$$

其中 λ_{\max} 和 λ_{\min} 为最大最小特征值。

任意的两个条件数都是等价的。

说明 1.12. Hilbert 矩阵 $\mathbb{H}_n = (h_{ij})$ 是一个著名的病态矩阵, 其中

$$h_{ij} = \frac{1}{i+j-1}, \quad b_{ij} = \frac{(-1)^{i+j}(n+i-1)!(n+j-1)!}{(i+j-1)![(i-1)!(j-1)!]^2(n-i)!(n-j)!}.$$

逆矩阵 \mathbb{H}_n^{-1} 可表述为 $\mathbb{H}_n^{-1} = (b_{ij})$. 在 Matlab 软件中的命令是 `hilb()` 和 `invhilb()`.

定理 1.15. 可逆矩阵 \mathbb{A} 的病态程度可描述为与一个奇异矩阵的接近程度, 即

$$\min_{\delta\mathbb{A}} \left\{ \frac{\|\delta\mathbb{A}\|_2}{\|\mathbb{A}\|_2} : \mathbb{A} + \delta\mathbb{A} \text{ 奇异} \right\} = \kappa_2^{-1}(\mathbb{A}).$$

证明: 显然左端大于或等于右端, 下面我们证明等号可以取到。令

$$\mathbf{y} = \frac{\mathbb{A}^{-1}\mathbf{x}}{\|\mathbb{A}^{-1}\mathbf{x}\|_2}, \quad \delta\mathbb{A} = -\frac{\mathbf{x}\mathbf{y}^\top}{\|\mathbb{A}^{-1}\|_2},$$

其中 \mathbf{x} 为单位长度向量, 使得 $\|\mathbb{A}^{-1}\mathbf{x}\|_2 = \|\mathbb{A}^{-1}\|_2$. 定理结论不难验证。□

下面, 我们观察一个简单的 2 阶矩阵 \mathbb{A} , 其中的 ε 为很小的正数。不难计算可得

$$\mathbb{A} = \begin{bmatrix} 1 & 1 \\ 1 & 1+\varepsilon \end{bmatrix}, \quad \mathbb{A}^{-1} = \varepsilon^{-1} \begin{bmatrix} 1+\varepsilon & -1 \\ -1 & 1 \end{bmatrix}, \quad 0 < \varepsilon \ll 1.$$

显然, 矩阵 \mathbf{A} 的谱条件数 $\kappa_2(\mathbf{A}) = (2 + \varepsilon + \sqrt{4 + \varepsilon^2})^2 / (4\varepsilon) \approx 4/\varepsilon$. 当 ε 靠近零时, 逆矩阵 \mathbf{A}^{-1} 对 ε 的变化非常敏感, 矩阵 \mathbf{A} 变得很病态。在高斯列主元消去过程中, 病态矩阵是怎样变化的呢? 其 LU 分解为

$$\mathbb{L}_\varepsilon = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad \mathbb{U}_\varepsilon = \begin{bmatrix} 1 & 1 \\ 0 & \varepsilon \end{bmatrix}.$$

显见矩阵 \mathbb{L}_ε 是良态的, 而矩阵 \mathbb{U}_ε 是病态的。可证: \mathbb{U} 的条件数与 \mathbf{A} 的条件数可互相控制。在高斯列主元消去过程中, 几乎总是保持 $\|\mathbb{L}\|\|\mathbb{U}\| \approx \|\mathbf{A}\|$. 大量的数值试验支持这个断言。

1.4.2 摄动理论

若线性方程组 $\mathbf{A}\mathbf{x} = \mathbf{b}$ 的已知数据发生扰动 $\delta\mathbf{A}$ 和 $\delta\mathbf{b}$, 问题的唯一解也会发生相应的改变。摄动理论给出相应的扰动量 $\delta\mathbf{x}$ 可被有效控制, 它们均与系数矩阵的条件数 $\kappa(\mathbf{A})$ 有关:

1. 右端项有扰动 $\delta\mathbf{b}$: $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}.$
2. 系数矩阵有扰动 $\delta\mathbf{A}$: $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\kappa(\mathbf{A}) \frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}}{1 - \kappa(\mathbf{A}) \frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|}}.$

粗略地讲, 矩阵条件数是误差的放大率。这是问题的固有信息, 与数值方法无关。

上述估计虽然很有理论价值, 但是它们却是非常保守的估计。它们给出的理论上界常常与真实误差相距甚远。譬如, 我们考虑简单的二阶线性方程组, 其系数矩阵为 $\mathbf{A} = \text{diag}(\gamma, 1)$, 右端项为 $\mathbf{b} = (\gamma, 1)^\top$. 显见, 真实误差的放大率应该为 1。但是, 上述保守估计中的条件数与 γ 有关的。若 $\gamma \gg 1$, 这个保守估计的缺陷是明显的。

尽管如此, 上述估计在理论上却很难被改进, 因为这个上界的确可以取到。庆幸的是, 大量的数值经验表明实际计算中遇到最坏情形的概率并不大, 我们可以用其他方式给出扰动误差的上界估计。

1.4.3 精度分析

从数值计算的实用角度出发, 一个重要的问题是我们如何判断最终计算结果的可靠性。常用的方法是利用相同系数矩阵的线性方程组进行试算, 认为它们具有相同的数值精度而确认可信的有效数字。

另一种方法是类似于摄动理论建立所谓的事后误差估计

$$\frac{\|\mathbf{x}_* - \mathbf{x}_{\text{num}}\|}{\|\mathbf{x}_*\|} \leq \kappa(\mathbf{A}) \frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}, \quad (1.4.6)$$

其中的 $\mathbf{r} = \mathbf{A}\mathbf{x}_{\text{num}} - \mathbf{b}$ 是一个可计算的 (较为可信的) 数值残量。在上述估计中, 我们通常取无穷模范数。

余下的关键问题是如何给出条件数的合理估计。下面我们不做什么的理论分析, 而直接给出矩阵 \mathbb{B} 的列范数估算方法:

1. 取任意初始向量 \mathbf{x} , 满足 $\|\mathbf{x}\|_1 = 1$;
2. $\mathbf{w} = \mathbb{B}\mathbf{x}; \mathbf{v} = \text{sign}(\mathbf{w}); \mathbf{z} = \mathbb{B}^T \mathbf{v}$;
3. 若 $\|\mathbf{z}\|_\infty \leq \mathbf{z}^\top \mathbf{x}$, 则输出估算值 $\|\mathbf{w}\|_1$;
4. 否则, 令 $\mathbf{x} = \mathbf{e}_j$ 为 j 个标准单位向量, 其中 $|z_j| = \|\mathbf{z}\|_\infty$. 返回到第 2 步;

它是著名的最优化方法“盲人下山法”的一个应用，已被 LAPACK 中所采用。

回到事后误差估计(1.4.6)，为估算条件数 $\kappa_\infty(\mathbb{A})$ ，我们需计算 $\|\mathbb{A}\|_\infty$ 和 $\|\mathbb{A}^{-1}\|_\infty$ 。后者可通过令 $\mathbb{B} = \mathbb{A}^{-T}$ ，调用上述模块即可，其中第 2 步可利用高斯消元得到的 LU 分解快速求解相应的线性方程组，所付出的额外计算代价不大。在 Matlab 软件中的命令是 `rcond()`。

★ 说明 1.13. 数值解的精度可通过迭代进行改进，其基本思想就是对残量进行同类型的线性方程组修正。

1.4.4 列主元高斯消去法的数值稳定性分析

本节简略介绍高斯列主元消去法的数值稳定性结果，即数值结果的相对误差与计算机的关系。详细内容可请见教科书。

浮点运算的误差估计

由于计算机有限位长的原因，舍入误差在计算的过程中是不可避免的。那么，舍入误差积累会产生怎样的最终效果？

计算机上的浮点数可表示为 $f = \pm 0.d_1d_2 \cdots d_t \times 2^J$ ，其中 $d_1 \neq 0$ 。这里的 t 称为计算机字长。所有的浮点数仅仅是实数域上的一个离散子集，对四则运算不再是封闭的。令 \bullet 表示任意的一种四则运算，简单的分析可知浮点数运算满足估计

$$fl(a \bullet b) = (a \bullet b)(1 + \delta), \quad \text{其中 } |\delta| \leq \vartheta = \begin{cases} \frac{1}{2}\beta^{1-t}, & \text{舍入法,} \\ \beta^{1-t}, & \text{截断法.} \end{cases}$$

是计算机的机器精度。简单的应用这些估计可得 $|fl(\mathbf{x}^\top \mathbf{y}) - \mathbf{x}^\top \mathbf{y}| \leq 1.01n\vartheta|\mathbf{x}|^\top |\mathbf{y}|$ ，以及

$$fl(\alpha \mathbb{A}) = \alpha \mathbb{A} + \mathbb{E}, \quad |\mathbb{E}| \leq \vartheta|\alpha \mathbb{A}|, \quad (1.4.7a)$$

$$fl(\mathbb{A} + \mathbb{B}) = \mathbb{A} + \mathbb{B} + \mathbb{E}, \quad |\mathbb{E}| \leq \vartheta|\mathbb{A} + \mathbb{B}|, \quad (1.4.7b)$$

$$fl(\mathbb{A}\mathbb{B}) = \mathbb{A}\mathbb{B} + \mathbb{E}, \quad |\mathbb{E}| \leq 1.01n\vartheta|\mathbb{A}||\mathbb{B}|, \quad (1.4.7c)$$

其中的向量或矩阵的阶数为 n 。

上述分析是所谓的向前误差分析技术，过程显得非常繁琐。实际工作中，我们更多地采用所谓的向后误差分析，即将最终的计算误差归结为初始数据的误差产生的，而后的所有运算均是精确无误差的。如(1.4.7a)可表示为

$$fl(\alpha \mathbb{A}) = \alpha(\mathbb{A} + \mathbb{E}), \quad |\mathbb{E}| \leq \vartheta|\mathbb{A}|.$$

向后误差分析的优点是，它将浮点数的运算转化为实数的精确运算。

列主元高斯消去法的舍入误差分析

利用向后误差分析技术，我们认为高斯列主元消去法的舍入误差可视为扰动问题

$$(\mathbb{A} + \delta \mathbb{A})(\mathbf{x} + \delta \mathbf{x}) = \mathbf{b},$$

所造成的解偏差。问题的关键是利用向前误差分析技巧，给出摄动矩阵的一个合理估计。而后，我们可以利用前面的摄动理论给出误差的一个估计。

简而言之，列主元高斯消去法中的消去过程可以用如下的三角分解 $\mathbb{P}\mathbb{A} + \mathbb{E} = \mathbb{L}\mathbb{U}$ 表示，其中 \mathbb{E} 是扰动部分， \mathbb{P} 是置换矩阵。 \mathbb{L} 和 \mathbb{U} 是计算所得的两个三角矩阵。而后的回代过程可描述为两

个三角形方程组 $(\mathbb{L} + \mathbb{F})\mathbf{y} = \mathbb{P}\mathbf{b}$ 和 $(\mathbb{U} + \mathbb{G})(\mathbf{x} + \delta\mathbf{x}) = \mathbf{y}$ 的求解, 其中 \mathbb{F} 和 \mathbb{G} 是扰动矩阵。利用数学归纳法和向前误差分析技巧, 可知上述三个扰动矩阵的元素上界为

$$|e_{ij}| \leq 2n\vartheta \max_{ij} |a_{ij}|, \quad |f_{ij}| \leq \frac{6}{5}(n+1)\vartheta|\ell_{ij}|, \quad |g_{ij}| \leq \frac{6}{5}(n+1)\vartheta|u_{ij}|.$$

综上所述, 最终的扰动方程组为 $(\mathbb{A} + \delta\mathbb{A})(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}$, 其中 $\delta\mathbb{A} = \mathbb{E} + \mathbb{P}(\mathbb{F}\mathbb{U} + \mathbb{L}\mathbb{G} + \mathbb{F}\mathbb{G})$. 在高斯列主元消去过程中, \mathbb{L} 的元素均以 1 为界, 取行范数可得 $\|\delta\mathbb{A}\|_\infty \leq Cn^3\vartheta\eta(\mathbb{A})\|\mathbb{A}\|_\infty$, 其中 $C \approx 10$ 为绝对常数, $\eta(\mathbb{A})$ 为主元增长因子。它是一个非常关键的量, 定义如下:

$$\eta(\mathbb{A}) = \frac{\max_{i,j,k} |a_{ij}^{(k)}|}{\max_{ij} |a_{ij}|},$$

其中 $\mathbb{A}^{(k)} = (a_{ij}^{(k)})$ 是 k 步消元后得到的矩阵在计算机中的表示。若矩阵的相对扰动很小的时候, 利用摄动理论可知最终的舍入误差满足

$$\frac{\|\delta\mathbf{x}\|_\infty}{\|\mathbf{x}\|_\infty} \leq Cn^3\vartheta\eta(\mathbb{A})\kappa_\infty(\mathbb{A}). \quad (1.4.8)$$

在高斯列主元消去过程中, \mathbb{L} 的元素均以 1 为界, 故而每次消元右上角矩阵元素至多被放大 2 倍。因此, 主元增长因子不会超过 2^{n-1} 。大量的数值经验告诉我们, 在高斯列主元消去过程中, 这个主元增长因子 $\eta(\mathbb{A})$ 几乎总是 n 或稍小一些, 通常处于 $n^{2/3}$ 或 $n^{1/2}$ 的状态。这就使得对大多数问题选择列主元算法是可以的。但令人遗憾的是, 在个别算例中这个指标的确可达 2^{n-1} 。

1.5 数值实验

❖ 练习 1.1. 考虑单位正方形 $[0, 1] \times [0, 1]$ 上满足零边界条件的 Laplace 方程 $-\Delta u = f$, 我们可以采用五点差分格式离散导数近似求解。令 $h = 1/(n+1)$, 我们在 (ih, jh) 点上求解 u_{ij} , 其中指标 i, j 从 1 遍历到 n 。这样可得线性方程组 $\mathbb{A}\mathbf{x} = \mathbf{b}$, 其中 \mathbf{x} 由逐行依次编号的 u_{ij} 构成。系数矩阵 \mathbb{A} 为 n^2 阶对称正定矩阵

$$\mathbb{A}_{n^2 \times n^2} = \begin{bmatrix} \mathbb{T}_{n \times n} + 2\mathbb{I}_{n \times n} & -\mathbb{I}_{n \times n} & & & \\ -\mathbb{I}_{n \times n} & \mathbb{T}_{n \times n} + 2\mathbb{I}_{n \times n} & & & \\ & & \ddots & \ddots & -\mathbb{I}_{n \times n} \\ & & & -\mathbb{I}_{n \times n} & \mathbb{T}_{n \times n} + 2\mathbb{I}_{n \times n} \end{bmatrix}, \quad (1.5.9)$$

其中 $\mathbb{I}_{n \times n}$ 为 n 阶单位矩阵, 而 $\mathbb{T}_{n \times n}$ 为 n 阶方阵

$$\mathbb{T}_{n \times n} = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}. \quad (1.5.10)$$

这两个矩阵将会在本课程中经常使用。设问题的真解为 $\mathbf{x}_* = (1, 1, 1, \dots, 1)^\top$, 右端向量由 $\mathbf{b} = \mathbb{A}\mathbf{x}_*$ 给出。令 n 从 5 变化到 30, 进行如下数值观测:

1. 编制高斯消去法求解线性方程组 $\mathbb{A}\mathbf{x} = \mathbf{b}$, 分别绘制误差和 CPU 时间与 n 的关系图。误差的纵坐标取对数坐标。

2. 编写 LDL^T 算法求解线性方程组 $\mathbf{A}\mathbf{x} = \mathbf{b}$, 分别绘制误差和 CPU 时间与 n 的关系图。

3. 计算系数矩阵的条件数, 然后绘制矩阵条件数与矩阵阶数 n 的关系。摄动理论给出的舍入误差估计(1.4.8)是否完美地刻画了相对误差大小?

❖ 练习 1.2. 矩阵非零元素的分布对数值计算的影响, 这个问题最优答案的确定是个 NP 问题。考虑行列重排后相等的两个矩阵

$$\mathbb{B}_1 = \begin{bmatrix} 1 & & & a \\ & 1 & & a \\ & & \ddots & \vdots \\ & & & 1 & a \\ a & a & \cdots & a & 1 \end{bmatrix}, \quad \mathbb{B}_2 = \begin{bmatrix} 1 & a & \cdots & a & a \\ a & 1 & & & \\ \vdots & & \ddots & & \\ a & & & 1 & \\ a & & & & 1 \end{bmatrix}, \quad (1.5.11)$$

比较三角分解后的非零元素分布。在 Matlab 中观测矩阵结构图的命令是 `spy()`。利用矩阵的元素分布特点修改 Crout 算法, 努力尝试省掉那些无用的零运算时间, 观察是否有 CPU 时间的节省。

❖ 练习 1.3. 设 $\mathbb{C} = \mathbb{D}_{n \times n} \mathbb{T}_{n \times n}$, 其中 $\mathbb{D} = \text{diag}(2^{-i})$ 或单位矩阵, $\mathbb{T}_{n \times n}$ 的定义见(1.5.10)。考虑线性方程组 $\mathbb{C}_{n \times n} \mathbf{x} = \mathbf{b}$ 的追赶法, 取真解为 $\mathbf{x}_* = (1, 1, 1, \dots, 1)^T$ 。令 n 从 500 变换到 2000, 请观测误差与矩阵阶数 n 的关系。考虑等价的问题 $\mathbb{T}_{n \times n} \mathbf{x} = \mathbb{D}^{-1} \mathbf{b}$, 追赶法的计算结果能否得到改善?

❖ 练习 1.4. 随机构造 n 阶可逆矩阵 \mathbb{B} , 进行列主元高斯消去分解。观测主元增长因子 $\eta(\mathbb{B})$ 是否大多处于 $n^{2/3}$ 或 $n^{1/2}$ 的状态。

第二章 线性方程组的迭代解法

求解线性方程组的另一种方法是迭代算法，它通过生成一个简单的解序列给出真解的一个合理近似。它具有不改变矩阵元素和编程实现容易等优势。它更多地基于内积运算（含矩阵与向量的乘法，向量与向量内积），具有本质上的 BLAS-2 机制。迭代法的存储量较小，是目前求解大规模稀疏矩阵的常用算法。

2.1 迭代法的基本理论

迭代法的基本思想是建立与原问题等价的不动点方程，从而诱导出相应的迭代格式。这种方法也称为不动点迭代方法。

2.1.1 不动点迭代的基本结构

④ 定义 2.1. 构造向量序列 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ ，其中 $\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \dots, \mathbf{x}_{k-r})$ 。

④ 定义 2.2. 若线性方程组 $\mathbb{A}\mathbf{x} = \mathbf{b}$ 的真解 \mathbf{x}_* 是 r 阶迭代公式的稳态解，则称迭代公式是完全相容的。以下均默认迭代是完全相容的。

我们从简单的一阶迭代格式，它通常可写成如此两种形式：

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbb{H}_k(\mathbf{b} - \mathbb{A}\mathbf{x}_{k-1}) = \mathbf{x}_{k-1} - \mathbb{H}_k\mathbf{r}_{k-1}, \quad (2.1.1a)$$

$$\mathbf{x}_k = \mathbb{G}_k\mathbf{x}_{k-1} + \mathbf{g}_k, \quad (2.1.1b)$$

其中 $\mathbf{r}_k = \mathbb{A}\mathbf{x}_k - \mathbf{b}$ 称为第 k 步迭代的残量；残量为零对应到精确解。通常，我们称 \mathbb{H}_k 为预处理矩阵， \mathbb{G}_k 为迭代矩阵。迭代算法研究的重点内容是如何构造合适的迭代矩阵或预处理矩阵，使得向量序列快速收敛到问题的真解。

👉 论题 2.1. 若 $\mathbb{G}_k = \mathbb{I} - \mathbb{H}_k\mathbb{A}$ 和 $\mathbf{g}_k = \mathbb{H}_k\mathbf{b}$ ，则上述两个迭代公式是等价的。

2.1.2 不动点迭代的收敛性分析

准备知识

为刻画解向量序列的收敛性，我们需要向量与矩阵序列的一些基础知识。此时，范数与谱半径是重要的度量工具。

④ 定义 2.3. $\lim_{k \rightarrow \infty} \mathbf{x}_k = \mathbf{x} \Leftrightarrow \lim_{k \rightarrow \infty} \|\mathbf{x}_k - \mathbf{x}\| = 0$.

④ 定义 2.4. $\lim_{k \rightarrow \infty} \mathbb{A}_k = \mathbb{A} \Leftrightarrow \lim_{k \rightarrow \infty} \|\mathbb{A}_k - \mathbb{A}\| = 0$.

主要的结论有

定理 2.1. $\lim_{k \rightarrow \infty} \mathbb{A}^k = \mathbb{O} \Leftrightarrow \rho(\mathbb{A}) < 1$.

定理 2.2. 矩阵级数 $\sum_{k=0}^{\infty} \mathbb{B}^k$ 收敛的充要条件是 $\varrho(\mathbb{B}) < 1$, 其和为 $(\mathbb{I} - \mathbb{B})^{-1}$.

定理 2.3. 若 $\|\mathbb{B}\| < 1$, 则矩阵级数 $\sum_{k=0}^{\infty} \mathbb{B}^k$ 收敛。矩阵级数的余项满足

$$\left\| \sum_{k=m+1}^{\infty} \mathbb{B}^k \right\| \leq \frac{\|\mathbb{B}\|^{m+1}}{1 - \|\mathbb{B}\|}, \quad (2.1.2)$$

与收敛幂级数的性质保持一致。其可视为有限项的三角不等式的推广。

收敛性判定

记 $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}_*$ 为迭代法第 k 步的误差 (向量), 其满足误差方程

$$\mathbf{e}_k = \mathbb{G}_k \mathbf{e}_{k-1}, \quad \text{或} \quad \mathbf{e}_k = (\mathbb{I} - \mathbb{H}_k \mathbb{A}) \mathbf{e}_{k-1}. \quad (2.1.3)$$

若对任意的初始向量, 均有 $\lim_{k \rightarrow \infty} \mathbf{e}_k = \mathbf{0}$, 则称迭代算法是收敛的。否则称迭代法是发散的。

定理 2.4. 迭代算法收敛等价于迭代矩阵乘积极限为 $\mathbf{0}$, 即

$$\lim_{k \rightarrow \infty} \prod_{m=1}^k \mathbb{G}_m = \mathbf{0}, \quad \text{或者} \quad \lim_{k \rightarrow \infty} \prod_{m=1}^k (\mathbb{I} - \mathbb{H}_m \mathbb{A}) = \mathbf{0}.$$

若 $\mathbb{G}_k \equiv \mathbb{G}$ 或 $\mathbb{H}_k \equiv \mathbb{H}$, 则称上述一阶迭代方法是定常的, 即 $\mathbf{x}_k = \mathbb{G} \mathbf{x}_{k-1} + \mathbf{g}$. 利用问题及算法简单的线性结构可知 $\mathbf{e}_k = \mathbb{G}^k \mathbf{e}_0$, 从而有结果:


定理 2.5. 一阶定常迭代法收敛的充要条件是 $\varrho(\mathbb{G}) < 1$. 因此, $\|\mathbb{G}\| < 1$ 是一阶定常迭代法收敛的一个充分条件。

收敛速度的刻画与估计

我们可以利用误差向量 \mathbf{e}_k 趋于零的快慢来评价迭代法的优劣。以一阶定常迭代方法 $\mathbf{x}_k = \mathbb{G} \mathbf{x}_{k-1} + \mathbf{g}$ 为例, 我们有误差估计:

$$\|\mathbf{e}_k\| \leq \|\mathbb{G}^k\| \|\mathbf{e}_0\|. \quad (2.1.4)$$


这个结果在某种意义下是不可改善的, 因为等号是可以取到的。注意真实的误差下降与初始向量的选取也是有关系的。为摆脱此影响, 我们对一个算法的收敛快慢评价应该是以它在最坏意义下的表现。

 论题 2.2. 利用保守上界估计(2.1.4), 我们通常按误差下降的平均效应刻画迭代误差趋零的快慢程度。

1. 平均收敛速度 $R_k(\mathbb{G}) = -\frac{1}{k} \ln \|\mathbb{G}^k\|$.
2. 渐进收敛速度 $R_{\infty}(\mathbb{G}) = \lim_{k \rightarrow \infty} R_k(\mathbb{G}) = -\ln \varrho(\mathbb{G})$.

这些概念是上世纪五六十年代提出的, 譬如渐进收敛速度是 Young (1954) 给出的。达到指定的误差要求所需的最小迭代次数与收敛速度的倒数成正比。

虽然这些概念与迭代误差在每一步的真实变化还有一定的差距, 但是它们的确可以在一定程度上, 用于比较不同迭代算法的收敛快慢。

 思考 2.1. 仔细比较如下两个矩阵的谱范数 $\|\mathbb{A}^m\|_2$ 和 $\|\mathbb{B}^m\|_2$ 的发展过程, 其中

$$\mathbb{A} = \begin{bmatrix} \alpha & 4 \\ 0 & \alpha \end{bmatrix}, \quad \mathbb{B} = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}, \quad 0 < \alpha < \beta < 1.$$

当 α 靠近 1 的时候, 是否会在初始阶段发生 $\|\mathbb{A}^m\|_2 > \|\mathbb{B}^m\|_2$ 呢?

✿ 思考 2.2. 证明 $\lim_{k \rightarrow \infty} \|\mathbb{G}^k\|^{1/k} = \varrho(\mathbb{G})$.

但是上述结果很难应用于实际计算, 因为 $\|\mathbb{G}^k\|$ 和 $\varrho(\mathbb{G})$ 都是很难计算和分析的。故而, 我们更多地考虑如下的误差估计。

定理 2.6. 若 $\|\mathbb{G}\| < 1$, 则一阶定常迭代法 $\mathbf{x}_k = \mathbb{G}\mathbf{x}_{k-1} + \mathbf{g}$ 是收敛的, 且满足

1. 先验误差估计: $\|\mathbf{e}_k\| \leq \|\mathbb{G}\|^k \|\mathbf{e}_0\|$;
2. 后验误差估计 (I): $\|\mathbf{e}_k\| \leq \frac{\|\mathbb{G}\|}{1-\|\mathbb{G}\|} \|\mathbf{x}_k - \mathbf{x}_{k-1}\|$;
3. 后验误差估计 (II): $\|\mathbf{e}_k\| \leq \frac{\|\mathbb{G}\|^k}{1-\|\mathbb{G}\|} \|\mathbf{x}_1 - \mathbf{x}_0\|$.

证明: 误差关系式的应用。 □

2.1.3 停机标准

何时停止迭代过程是个很重要的问题。真实误差仅仅可作为理论上的停机标准, 即当 $\|\mathbf{e}_k\| \leq \mathcal{E}$ 时停机, 其中 \mathcal{E} 是用户指标。但是, 真实误差是无法实际计算的量, 因此我们更多地利用如下停机准则:

1. 残量准则: $\|\mathbf{r}_k\| \leq \mathcal{E}$;
2. 相邻误差准则: $\delta_k \equiv \|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leq \mathcal{E}$;
3. 后验误差停机准则: $\frac{\delta_k^2}{\delta_{k-1} - \delta_k} \leq \mathcal{E}$.

范数通常取为最大模范数或者欧几里得范数。第三种方法来源与后验误差估计 (I) 和幂法估算。这里使用的是绝对误差, 我们也可使用相对误差。

★ 说明 2.1. 请注意舍入误差对格式的收敛有影响。譬如, 应用上面的最后两个停机准则时, 用户要小心出现所谓的“假停机”现象。考虑

$$\mathbf{x}_k = \begin{bmatrix} 0 & 1 - 10^{-6} \\ 1 - 10^{-6} & 0 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} 10^{-6} \\ 10^{-6} \end{bmatrix},$$

其迭代矩阵的范数接近 1. 若用 6 位有效数字 10 进制计算机求解, 取初值为 $\mathbf{x}_0 = (0.1, 0.1)^\top$, 则迭代仅对每个分量产生 10^{-6} 的增量。因此, $\|\mathbf{x}_k - \mathbf{x}_{k-1}\|_\infty = 10^{-6} < \mathcal{E}$, 出现所谓的假停机。对迭代法的舍入误差分析超出本课程要求, 略。

2.2 两个古典迭代算法

本节介绍两个古典迭代算法。其一是 Jacobi 方法。它最初是由 Jacobi 在 1845 年提出的。而后的相关工作还包括: Geiringer(1945) 的同步位移法, 以及 Killer(1958) 的 Richardson 迭代法。物理上, 该方法也称为阻尼法。其二是著名的 Gauss-Seidel 方法。据传言, 它最初是由 Gauss(1822) 提出, 用于解决由测地问题而产生的最小二乘线性方程组的分析和计算。而后, Seidel 在 1874 年再次提出此方法, 但却不主张使用它。理论分析最早出现于 Von. Misers 和 Pollaczek-Geiringer(1949) 的工作中。

一阶定常迭代方法通常基于所谓的“矩阵分裂”方式给出:

$$\mathbf{A} = \mathbf{Q} - \mathbf{R}, \text{ 其中 } \mathbf{Q} \text{ 是逼近 } \mathbf{A} \text{ 的容易求逆的预处理矩阵。}$$

假设矩阵 \mathbf{A} 按其対角线部分, 严格下三角部分和严格上三角部分划分, 有如下形式:

$$\mathbf{A} = \mathbf{D} - \mathbf{DL} - \mathbf{DU}.$$

选取対角线部分或下三角部分可形成上述两个古典算法。

🔍 论题 2.3. 两个算法的描述。Jacobi 方法的迭代矩阵是 $\mathbf{B} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$, 而 GS 方法的迭代矩阵是 $\mathbf{T}_1 = (\mathbf{I} - \mathbf{L})^{-1}\mathbf{U}$ 。第 i 个分量 $\mathbf{x}^{(i)}$ 的更新方式是:

$$\begin{aligned} 1. J \text{ 方法: } \quad \mathbf{x}_k^{(i)} &= \frac{1}{a_{ii}} \left[\mathbf{b}^{(i)} - \sum_{j \neq i} a_{ij} \mathbf{x}_{k-1}^{(j)} \right]; \\ 2. GS \text{ 方法: } \quad \mathbf{x}_k^{(i)} &= \frac{1}{a_{ii}} \left[\mathbf{b}^{(i)} - \sum_{j < i} a_{ij} \mathbf{x}_k^{(j)} - \sum_{j > i} a_{ij} \mathbf{x}_{k-1}^{(j)} \right]; \end{aligned}$$

这两个算法的主要思想与区别是所谓的同步更新策略与异步更新策略。 J 方法具有很好的本质并行结构, 但缺点是需要存放两组工作单元记录解向量。

★ 说明 2.2. 迭代矩阵的谱与线性方程组的行排序有关, 譬如, 我们用 J 方法求解如下两个同解的线性方程组

$$\begin{bmatrix} 3 & -10 \\ 9 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -7 \\ 5 \end{bmatrix}, \quad \begin{bmatrix} 9 & -4 \\ 3 & -10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ -7 \end{bmatrix}.$$

请计算 J 迭代矩阵的谱半径, 并指出 J 迭代是否收敛? 这隐含地说明预处理技术的必要性。详细的预处理思想容后介绍。

🔍 论题 2.4. 事实上, 上述两种迭代方法的收敛性没有任何的关联。但是, GS 方法通常会比 J 方法收敛得更快一些, 特别是对于某些特殊的情形。

定理 2.7. 若 $\|\mathbf{B}\|_{\infty} < 1$, 则 GS 方法也收敛, 并比 J 方法更快。

定理 2.8. 若 $\|\mathbf{B}\|_1 < 1$, 则 GS 方法也收敛。

定理 2.9. 若系数矩阵 \mathbf{A} 是对角占优的, 则 J 方法和 GS 方法均收敛。

1. 严格对角占优矩阵: $\forall i, |a_{ii}| > \sum_{j \neq i} |a_{ij}|$;
2. 弱对角占优不可约矩阵: $\forall i, |a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$, 且至少有一个不等式是严格成立的; 所谓的不可约是指问题不能分解为更小规模的问题。

定理 2.10. 若系数矩阵 \mathbf{A} 实对称正定, 则 GS 方法必然收敛。但是, J 方法收敛还需补充一个充要条件: $2\mathbf{D} - \mathbf{A}$ 也正定。

★ 说明 2.3. 上面四个定理的证明是迭代法收敛性分析的基本技巧: 范数估计或特征值分析。


★ 说明 2.4. 矩阵对角占优的强度可用上述不等式两端的最小差距来衡量, 如

$$\mathbf{A}_1 = \begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 1 & -\frac{3}{4} \\ -\frac{1}{4} & 1 \end{bmatrix}.$$

简单计算可知 J 方法的迭代矩阵 $\rho(\mathbf{B}_1) > \rho(\mathbf{B}_2)$ 。虽然 \mathbf{A}_1 更加对角占优一些, 但其对应 J 方法的收敛速度略慢一些。


2.3 逐次超松弛迭代法

逐次超松弛 (SOR) 方法在历史上具有重要的地位, 它开辟了线性方程组迭代算法研究的新领域。其基本思想是将 GS 迭代方法给出的新解与旧解做适当的加权平均。通过权重选取, 算法的收敛速度较原算法有明显的加速。

 论题 2.5. 算法描述: 其迭代矩阵是 $\mathbb{T}_\omega = (\mathbb{I} - \omega\mathbb{L})^{-1}[(1 - \omega)\mathbb{I} + \omega\mathbb{U}]$. 分量表达方式是:

$$\mathbf{x}_k^{(i)} = (1 - \omega)\mathbf{x}_k^{(i)} + \frac{\omega}{a_{ii}} \left[\mathbf{b}^{(i)} - \sum_{j < i} a_{ij}\mathbf{x}_k^{(j)} - \sum_{j > i} a_{ij}\mathbf{x}_{k-1}^{(j)} \right];$$

其中 ω 称为松弛因子。当 $\omega = 1$, 该算法就是 GS 算法。


 思考 2.3. 请指出 SOR 迭代方法的矩阵分裂形式。

定理 2.11. SOR 方法收敛的必要条件是 $0 < \omega < 2$.

定理 2.12. 若系数矩阵 \mathbf{A} 对称正定, 则当 $0 < \omega < 2$ 时 SOR 方法收敛。

证明: 这是迭代矩阵特征值分析的典型例子。 □

数值试验表明松弛因子 ω 的不同取值对 SOR 迭代方法的收敛速度有明显的影响, 特别是存在最佳松弛因子使收敛速度得到显著的提升。这些研究曾是 80 年代之前的主流工作之一, 形成 SOR 算法的亮点。

 论题 2.6. 最佳松弛因子的研究依赖线性方程组的系数矩阵的非零元素分布结构, 相关概念有“相容次序”和“性质 \mathbf{A} ”等。详细的内容请见教科书。常用的结论有

1. 三对角阵或块三对角阵都是具有相容次序的。
2. 若矩阵具有相容次序, 则它必具有性质 \mathbf{A} 。
3. 若矩阵具有性质 \mathbf{A} , 它不一定具有相容次序, 但经过适当的行列重排后便可具有相容次序。

为简化讨论而突出这部分研究的思想, 我们考虑一种典型的系数矩阵结构。在椭圆方程的差分离散过程中, 我们经常遇到如下的线性方程组。所有的未知量分属两个不同的集合, 每个未知量仅与另一个集合中的未知量发生关联。对应这种现象, 相应的系数矩阵 \mathbf{A} 称具有性质 \mathbf{A} , 即存在置换阵 \mathbb{P} 使得

$$\mathbb{P}\mathbf{A}\mathbb{P}^T = \begin{bmatrix} \mathbb{D}_1 & \mathbb{H} \\ \mathbb{K} & \mathbb{D}_2 \end{bmatrix}, \text{ 其中 } \mathbb{D}_1 \text{ 和 } \mathbb{D}_2 \text{ 是两个对角线矩阵.}$$

右侧的矩阵对应为未知量采用所谓的红黑原则重新编号和方程组中方程次序的重新排列。下面, 我们直接对等式右侧的分块矩阵进行讨论。不失一般性, 进一步假设 \mathbb{D}_1 和 \mathbb{D}_2 都是单位阵, 即

$$\mathbf{A} = \mathbb{I} - \mathbb{B} = \mathbb{I} - (\mathbb{L} + \mathbb{U}). \quad (2.3.5)$$

定理 2.13. 设 λ 是 SOR 迭代矩阵 \mathbb{T}_ω 的特征值, 而 μ 是 J 迭代矩阵 \mathbb{B} 的特征值, 则有

$$(\lambda + \omega - 1)^2 = \lambda\omega^2\mu^2. \quad (2.3.6)$$

证明：利用分块矩阵表述，

$$\begin{bmatrix} \mathbb{I} & \\ & \alpha^{-1}\mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbb{D}_1 & \alpha^{-1}\mathbb{H} \\ \alpha\mathbb{K} & \mathbb{D}_2 \end{bmatrix} \begin{bmatrix} \mathbb{I} & \\ & \alpha\mathbb{I} \end{bmatrix} = \begin{bmatrix} \mathbb{D}_1 & \mathbb{H} \\ \mathbb{K} & \mathbb{D}_2 \end{bmatrix}, \quad (2.3.7)$$

分别计算两种方法的迭代矩阵特征值，即可得到结论。 \square

定理 2.14. 设 n 阶矩阵 \mathbb{B} 的所有特征值均可表达为 $\mu_j = \alpha_j + \sqrt{-1}\beta_j$ ，其中 α_j 和 β_j 均为实数。若存在正数 D ，使得

$$\alpha_j^2 + \beta_j^2/D < 1, \quad j = 1, 2, \dots, n,$$

则当 $0 < \omega < 2/(1+D)$ 时 SOR 迭代收敛。因此，若矩阵 \mathbb{B} 的特征值 μ_j 均为实数，则 SOR 迭代收敛的充分必要条件是

$$0 < \omega < 2, \text{ 且 } \rho(\mathbb{B}) < 1.$$

证明：利用二次方程的根与系数关系，以及(2.3.6)。 \square

论题 2.7. 下面讨论最佳松弛因子如何选取。本课程仅讨论一个简单的情形，即迭代矩阵 \mathbb{B} 的所有特征值均为实数，且谱半径小于 1。固定 \mathbb{B} 的某个特征值 μ ，考虑

$$\text{直线 } y = g_\omega(\lambda) = \frac{\lambda + \omega - 1}{\omega}, \quad \text{抛物线 } y = m(\lambda) = \pm \lambda^{1/2}|\mu|,$$

的交点 $\lambda_1(\omega)$ 和 $\lambda_2(\omega)$ 。这里假设交点存在，即 λ 为实数。否则，由定理 2.13 可知，存在的共轭复根满足 $|\lambda_i(\omega)| \equiv |\omega - 1|$ 。当 ω 变化时，要使得 $\max_{i=1,2} |\lambda_i(\omega)|$ 达到最小，就应要求直线 $g_\omega(\lambda)$ 与抛物线 $m(\lambda)$ 相切才能达到目的。即二次方程判别式为零，对应

$$\omega = \frac{2}{1 + \sqrt{1 - \mu^2}}, \quad \max_{i=1,2} |\lambda_i(\omega)| \leq |\omega - 1|. \quad (2.3.8)$$

显然，当 $|\mu|$ 增加时切点位置会随之增加。因此，最佳松弛因子应取为

$$\omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - \rho^2(\mathbb{B})}}.$$

它对应谱半径函数 $\rho(\mathbb{T}_\omega)$ 的不可微点；其具体表达式见教科书。

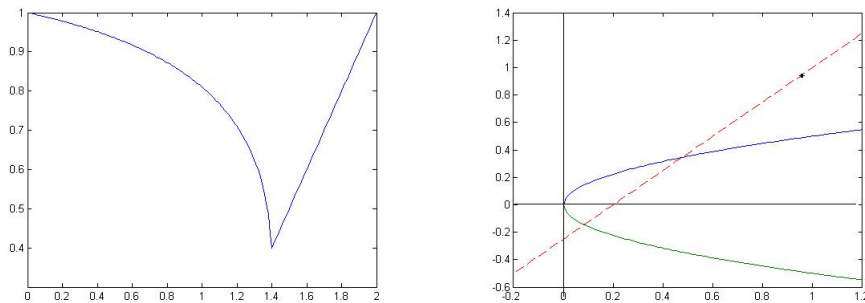


图 2.3.1: 左: 谱半径函数 $\rho(\mathbb{T}_\omega)$. 右: 直线与抛物线的交点。

★ 说明 2.5. 注意到谱半径函数在不可微点的左导数为无穷大, 实际计算时通常会偏大选取松弛因子。

👉 论题 2.8. 最佳松弛因子时超松弛算法的迭代速度提升。

回顾 1.5 中的线性方程组 $\mathbb{A}_{n^2 \times n^2} \mathbf{x} = \mathbf{b}$, 它对应正方形区域上 Laplace 方程的离散。其系数矩阵是块三对角矩阵, 经简单的行列重排, 不难发现其满足性质 A。容易验证三对角矩阵 $\mathbb{T}_{n \times n}$ 的特征值为 $\lambda_\kappa = 2(1 - \cos \frac{\kappa\pi}{n+1})$, 对应的单位特征向量为

$$\mathbf{v}_\kappa = \sqrt{\frac{2}{n+1}} \left(\sin \frac{\kappa\pi}{n+1}, \sin \frac{2\kappa\pi}{n+1}, \dots, \sin \frac{n\kappa\pi}{n+1} \right)^\top, \quad \kappa = 1, 2, \dots, n.$$

由 $\mathbb{A}_{n^2 \times n^2} = \mathbb{T}_{n \times n} \otimes \mathbb{I}_{n \times n} + \mathbb{I}_{n \times n} \otimes \mathbb{T}_{n \times n}$ 可知 $\mathbb{A}_{n^2 \times n^2}$ 的特征值为 $\lambda_{pq} = \lambda_p + \lambda_q$, 其中的指标 p 和 q 均遍历 1 到 n . 这蕴含着

$$R_\infty(\mathbb{B}) = -\ln \varrho(\mathbb{B}) = -\ln \cos h\pi \sim \frac{1}{2}\pi^2 h^2, \quad (2.3.9a)$$

$$R_\infty(\mathbb{L}_1) = -2 \ln \varrho(\mathbb{B}) \sim \pi^2 h^2, \quad (2.3.9b)$$

$$R_\infty(\mathbb{L}_{rmopt}) = -\ln \frac{1 - \sqrt{1 - \varrho(\mathbb{B})^2}}{1 + \sqrt{1 - \varrho(\mathbb{B})^2}} = -\ln \frac{1 - \sin(h\pi)}{1 + \sin(h\pi)} \sim 2h\pi. \quad (2.3.9c)$$

其中 $h = 1/(n+1)$. 从这些结果看出, 具最佳松弛因子的 SOR 收敛速度要比 J 方法和 GS 方法的收敛速度有显著意义上的提升, 特别是 n 越来越大时。

上面针对模型方程的推导是 Franke 给出的, 而后由 Young 进行了推广。详细的内容可参见教科书。

★ 说明 2.6. 还有所谓的块 SOR 方法和对称 SOR 方法。

2.4 迭代加速

SOR 方法的研究告诉我们在迭代法的构造过程中, 我们可以充分地挖掘和利用已有的历史计算信息得到更快速的收敛效果。本节将给出一些相关研究的介绍。

2.4.1 外推方法

首先我们讨论最简单的外推思想。其加速技巧与 SOR 方法类似, 对相邻的两个数值近似做所谓的 γ -加权平均。

$$\mathbf{x}_k = \gamma(\mathbb{G}\mathbf{x}_{k-1} + \mathbf{g}) + (1 - \gamma)\mathbf{x}_{k-1}.$$

✿ 思考 2.4. 假设原始迭代矩阵 \mathbb{G} 是对称的, 其最大最小特征值是已知的。计算新的迭代矩阵 $\gamma\mathbb{G} + (1 - \gamma)\mathbb{I}$ 的谱半径, 并找到最优的参数 γ 使收敛速度达到最快。

2.4.2 半迭代方法

它是外推思想的一种贪婪的推广。我们想要充分地利用手中的所有信息, 希望通过适当的加权平均给出“更好”的近似解, 即

$$\mathbf{y}_m = \sum_{k=0}^m \alpha_{m,k} \mathbf{x}_k, \quad (2.4.10)$$

其中 \mathbf{x}_k 由某个基础迭代 $\mathbf{x}_k = \mathbb{G}\mathbf{x}_{k-1} + \mathbf{g}$ 给出, 参数组 $\alpha_{m,k}$ 满足相容性条件 $\sum_{k=0}^m \alpha_{m,k} = 1$. 记相应的误差为 $\boldsymbol{\eta}_m = \mathbf{y}_m - \mathbf{x}_*$, 其迭代误差满足

$$\boldsymbol{\eta}_m = \sum_{k=0}^m \alpha_{m,k} \mathbb{G}^k \mathbf{e}_0 = \mathbb{P}_m(\mathbb{G}) \mathbf{e}_0, \quad (2.4.11)$$

其中 $\mathbb{P}_m(\mathbb{G})$ 是一个矩阵多项式。这衍生出新的概念“多项式迭代算法”。

这个朴素的思想在真正的数值计算中没法直接应用, 因为历史数据和参数组的记录会导致巨大的存储量需求, 而参数组的计算和确定也会引起严重的舍入误差积累。


变系数 Richardson 方法

事实上, 半迭代加速的思想在一些古典迭代算法中已经得到隐含地体现。经典的例子是变系数 Richardson (1910 年) 迭代法

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \tau_k(\mathbf{b} - \mathbb{A}\mathbf{x}_{k-1}), \quad (2.4.12)$$

其中的迭代参数 τ_k 是随 k 变化的。若参数 τ_k 一直保持不变, 则称其为定常 Richardson 迭代法, 它是一种简单的残量松弛方法。

为讨论方便, 我们考虑一个特殊情形: \mathbb{A} 是实对称正定矩阵, 相应的最大最小特征值分别为 λ_{\max} 和 λ_{\min} 。

 论题 2.9. 变系数 Richardson 迭代法可视为定常 Richardson 迭代法的一种半迭代加速。若想在指定的 m 步迭代中得到最佳的收敛速度, 对应的参数组 $\{\tau_k\}_{k=1}^m$ 为

$$\tau_k = \left[\frac{\lambda_{\max} - \lambda_{\min}}{2} \cos \theta_k + \frac{\lambda_{\max} + \lambda_{\min}}{2} \right]^{-1}, \quad \text{其中 } \theta_k = \frac{2k-1}{2m} \pi.$$

相应的收敛速度估计在如下定理给出。

定理 2.15. 设 \mathbb{A} 是实对称正定矩阵, m 为指定的正整数。变系数 Richardson 迭代在最佳变系数的设置下, 有收敛速度估计

$$\frac{\|\mathbf{e}_m\|_2}{\|\mathbf{e}_0\|_2} \leq 2 \left(\frac{\sqrt{\kappa(\mathbb{A})} - 1}{\sqrt{\kappa(\mathbb{A})} + 1} \right)^m,$$


其中 $\kappa(\mathbb{A}) = \lambda_{\max}/\lambda_{\min}$ 为谱条件数。

★ 说明 2.7. 不难发现, 为达到指定的误差要求所需的最小迭代步数有了明显的下降。在定常 Richardson 迭代法中, 这个量与 $\kappa(\mathbb{A})$ 成正比, 而在变系数 Richardson 方法中, 这个量仅与 $\sqrt{\kappa(\mathbb{A})}$ 成正比。

★ 说明 2.8. 上述最优参数组的计算与 m 相关。若 m 很大, 参数组的计算舍入误差会导致迭代收敛速度的严重破坏。为克服这个困难, 常用的办法是采用较小的 m 和所谓的循环 (或重新启动) 策略。

Chebyshev 加速方法

从变系数 Richardson 方法中可以看到 Chebyshev 多项式起着重要的作用。注意到这是一个正交多项式, 它具有所谓的三项递推关系式, 我们可以得到真正可数值实现的半迭代算法。

 论题 2.10. 基本迭代的半迭代加速算法。

假设基础迭代法 $\mathbf{x}_k = \mathbb{G}\mathbf{x}_{k-1} + \mathbf{g}$ 是可对称化的, 即迭代矩阵 \mathbb{G} 具有实特征值 $\{\lambda_i\}_{i=1}^n$ 和完备的向量系 $\{\boldsymbol{\xi}_i\}_{i=1}^n$. 设初始误差展开为 $\mathbf{e}_0 = \sum_{1 \leq i \leq n} \beta_i \boldsymbol{\xi}_i$, 由(2.4.11)可知半迭代方法的误差可表示

$$\mathbf{e}_k = \sum_{i=1}^n \beta_i Q_k(\lambda_i) \xi_i, \quad \text{其中 } Q_k(\lambda) = \sum_{\ell=0}^k \alpha_{k,\ell} \lambda^\ell.$$

利用谱范数作为度量，如何选取合适的参数使得算法获得最快的收敛速度？设基础迭代矩阵 \mathbb{G} 的最大最小特征值分别是 λ_{\max} 和 λ_{\min} ，上述目标可转化为一个实 Chebyshev 问题：

$$\max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} |Q_k^*(\lambda)| = \min_{\substack{Q_k \in \mathbb{P}_k \\ Q_k(1)=1}} \max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} |Q_k(\lambda)|.$$

此处假设了 $1 \notin [\lambda_{\min}, \lambda_{\max}]$ ，以避免归一化的困难。答案是归一化的 Chebyshev 多项式

$$Q_k^*(\lambda) = \frac{T_k(\ell(\lambda))}{T_k(\ell(1))}, \quad \text{其中 } \ell(\lambda) = \frac{2\lambda - \lambda_{\max} - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}.$$

这里的 $T_k(x)$ 为 k 次 Chebyshev 多项式，满足三项递推关系式 $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$ ，其中 $T_0(x) = 1$ 和 $T_1(x) = x$ 。

显见参数组 $\{\alpha_{k,\ell}\}$ 是这个最佳多项式 $Q_k^*(\lambda)$ 的系数。真正计算和存储这些参数和所有的历史数据是一个不可完成的任务。我们必须给出等价的实用公式。

注意 Chebyshev 多项式的正交性质，利用误差方程和 $(\mathbb{I} - \mathbb{G})\mathbf{x}_* = \mathbf{g}$ ，我们可以反推到如下的变系数二步迭代公式

$$\begin{aligned} \mathbf{x}_{k+1} &= 2\ell(\mathbb{G}) \frac{T_k(\xi)}{T_{k+1}(\xi)} \mathbf{x}_k - \frac{T_{k-1}(\xi)}{T_{k+1}(\xi)} \mathbf{x}_{k-1} + \frac{4}{\lambda_{\max} - \lambda_{\min}} \frac{T_k(\xi)}{T_{k+1}(\xi)} \mathbf{g} \\ &= \rho_{k+1} \{\nu(\mathbb{G}\mathbf{x}_k + \mathbf{g}) + (1 - \nu)\mathbf{x}_k\} + (1 - \rho_{k+1})\mathbf{x}_{k-1}, \end{aligned}$$

其中 $\xi = \ell(1)$, $\nu = 2/(2 - \lambda_{\max} - \lambda_{\min})$, $\rho_{k+1} = 2\xi T_k(\xi)/T_{k+1}(\xi)$ 。最后一个参数可递推求解：

$$\rho_{k+1} = \left[1 - \frac{1}{4\xi^2} \rho_k\right]^{-1}, \quad \rho_1 = 2.$$

在半迭代算法的第一步，我们取 $\mathbf{x}_1 = \mathbb{G}\mathbf{x}_0 + \mathbf{g}$ ，其中 \mathbf{x}_0 是任意的初值。

✿ 思考 2.5. 设线性方程组的系数矩阵具有性质 A ，或者由 (2.3.6) 定义。考虑 Jacobi 迭代方法的半迭代加速。请观察 ρ_k 的极限是什么？尝试度量一下其平均收敛速度是否具有本质上的提升？

★ 说明 2.9. 从上述两个例子不难发现，半迭代方法参数组的设定均与系数矩阵的谱分布区间有关。但是，最大最小特征值的确定是比线性方程组的求解更加困难的问题。这个缺陷限制了半迭代方法的应用。

★ 说明 2.10. 上述讨论也可以推广到基础迭代矩阵 \mathbb{G} 具有复特征值的情形，其参数组的设定与复特征值所属的椭圆区域有关。具体内容超出本课程的要求，略。

2.5 共轭斜量法

本节介绍求解对称正定线性方程组 $\mathbf{Ax} = \mathbf{b}$ 的一种新型方法，其基于一个等价的极值问题的快速求解过程。它是上世纪 50 年代由 Hestenes 和 Stiefel 首先提出的。共轭斜量法实际上是一种直接法，但在更多的情况下，它被视为一种迭代法。它无需对系数矩阵的特征值做事前估计，具有无参数、收敛快等优势。这种算法的研究及其推广形成了 Galerkin 方法或者 Krylov 子空间投影方法。

2.5.1 等价的极值问题与求解

🔍 论题 2.11. 设 \mathbf{x}_* 是线性方程组的真解。在共轭斜量法的多种引进途径中, 较为直观的是如下二次方程 (椭圆抛物面) 的最优化问题求解:

$$f(\mathbf{x}_*) = \min_{\mathbf{x}} f(\mathbf{x}), \quad \text{其中 } f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^\top \mathbb{A} \mathbf{x} - \mathbf{b}^\top \mathbf{x}.$$

定理 2.16. 两种表述是彼此等价的。

关于这个二次优化问题的快速求解有很多方法, 最常用的核心技术是一维搜索策略。即, 依次由当前位置 \mathbf{x}_k 出发, 沿搜索方向 \mathbf{p}_k , 寻找最优位置 \mathbf{x}_{k+1} 使目标函数达到极小:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{其中 } \alpha_k = -\frac{\mathbf{r}_k^\top \mathbf{p}_k}{\mathbf{p}_k^\top \mathbb{A} \mathbf{p}_k}, \quad \mathbf{r}_k = \mathbb{A} \mathbf{x}_k - \mathbf{b}.$$

这个算法的一个重要特征是搜索空间的逐维扩张, 即

$$\mathbf{x}_k \in \pi_k \equiv \mathbf{x}_0 + \mathcal{L}_k, \quad \mathcal{L}_k = \text{span}\{\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{k-1}\}. \quad (2.5.13)$$

若 $f(\mathbf{x}_k) = \min_{\mathbf{x} \in \pi_k} f(\mathbf{x})$, 则称 \mathbf{x}_k 关于搜索空间 \mathcal{L}_k 最优。若维数扩张到全空间, 所得极值即为线性方程组的真解。

显然, 每次一维搜索给出的新位置关于当前搜索方向是最优的, 那么它关于历史搜索方向是否也是最优的呢? 为回答此问题, 我们需要如下的引理。

引理 2.1. \mathbf{x}_k 关于搜索空间 \mathcal{L}_k 最优的充要条件是残量 $\mathbf{r}_k \perp \mathcal{L}_k$, 即

$$\mathbf{r}_k^\top \mathbf{p}_r = 0, \quad r = 0, 1, \dots, k-1,$$

其中 $\mathbf{r}_k = \mathbb{A} \mathbf{x}_k - \mathbf{b}$ 是当前位置 \mathbf{x}_k 的残量。

2.5.2 共轭斜量系的构造

下面, 我们利用引理 2.1 讨论搜索方向的设定策略。

🔍 论题 2.12. 最自然的搜索方向 \mathbf{p}_k 设置是当前位置的最速下降方向, 即

$$-\text{grad}f(\mathbf{x}_k) = \mathbf{b} - \mathbb{A} \mathbf{x}_k = -\mathbf{r}_k.$$

这种方法称为最速下降法。显然, $\mathbf{x}_{k+2} \in \mathbf{x}_k + \text{span}(\mathbf{r}_k, \mathbf{r}_{k+1})$, 其关于当前搜索方向 \mathbf{r}_{k+1} 是最优的。但是简单的分析表明: 其关于前一搜索方向 \mathbf{r}_k 却不是最优的, 因为

$$\begin{aligned} \mathbf{r}_{k+2}^\top \mathbf{r}_k &= (\mathbf{r}_{k+1} + \alpha_{k+1} \mathbb{A} \mathbf{r}_{k+1})^\top \mathbf{r}_k = \alpha_{k+1} \mathbf{r}_{k+1}^\top \mathbb{A} \mathbf{r}_k = \frac{\alpha_{k+1}}{\alpha_k} \mathbf{r}_{k+1}^\top (\mathbf{r}_{k+1} - \mathbf{r}_k) \\ &= \frac{\alpha_{k+1}}{\alpha_k} \mathbf{r}_{k+1}^\top \mathbf{r}_{k+1} \neq 0. \end{aligned}$$

🌀 思考 2.6. 上述事实造成迭代的收敛速度越来越慢, 产生所谓的“盘旋收敛”现象。请给出相应的收敛估计。

要得到关于整个历史搜索空间的最优条件，我们先从一个简单的目标出发：怎样使新的迭代位置关于临近的局部二维搜索空间是最优呢？设 $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{q}$ 是从 \mathbf{x}_k 出发沿当前搜索方向 \mathbf{q} 的最优位置，若还想要其关于历史搜索方向 \mathbf{p} 也是最优，我们有

$$0 = \mathbf{r}_{k+1}^\top \mathbf{p} = (\mathbf{r}_k - \mathbb{A}\mathbf{q})^\top \mathbf{p} = \mathbf{r}_k^\top \mathbf{p} - \mathbf{q}^\top \mathbb{A}\mathbf{p} = -\mathbf{q}^\top \mathbb{A}\mathbf{p},$$

即 \mathbf{p} 与 \mathbf{q} 是关于 \mathbb{A} 共轭正交的。出于此目标，我们诱导出如下的重要定义。

🔗 定义 2.5. 若对任意两个不同的指标 i 和 j ，均有 $\mathbf{p}_i^\top \mathbb{A}\mathbf{p}_j = 0$ ，则称 $\{\mathbf{p}_k\}_{k=0}^m$ 构成一个共轭向量系。以此共轭向量系为搜索方向的一维搜索算法称为共轭斜量法。

定理 2.17. 由共轭斜量系得到的位置序列关于历史搜索方向是整体最优的。

★ 说明 2.11. 若所有的计算都是精确的，共轭斜量法至多进行 n 步便可得到问题的真解。因此说，共轭斜量法应是一种直接算法。

🔗 论题 2.13. 共轭斜量方向的局部构造过程：注意到 \mathbf{r}_{k+1} 与 \mathbf{p}_k 的垂直性，它们局部张成一个二维平面。我们可在这个平面上寻找一个新的共轭方向 \mathbf{p}_{k+1} 。共轭斜量法的具体计算过程如下：

取初始方向 $\mathbf{p}_0 = -\mathbf{r}_0 = \mathbf{b} - \mathbb{A}\mathbf{x}_0$ ，做迭代

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \alpha_k = -\frac{\mathbf{r}_k^\top \mathbf{p}_k}{\mathbf{p}_k^\top \mathbb{A}\mathbf{p}_k}, \quad (2.5.14a)$$

$$\mathbf{r}_{k+1} = \mathbf{r}_k + \alpha_k \mathbb{A}\mathbf{p}_k, \quad (2.5.14b)$$

$$\mathbf{p}_{k+1} = -\mathbf{r}_{k+1} + \beta_k \mathbf{p}_k, \quad \beta_k = \frac{\mathbf{r}_{k+1}^\top \mathbb{A}\mathbf{p}_k}{\mathbf{p}_k^\top \mathbb{A}\mathbf{p}_k}. \quad (2.5.14c)$$

在 Matlab 软件中的基本命令是 `cg()`。

🔗 思考 2.7. 忽视 α_k 和 β_k 的计算过程，不妨将其认为是事先设定的参数。请将共轭斜量法表示为一个二阶非定常迭代方法的形式。

🔗 论题 2.14. 可以证明，上述局部共轭构造过程可以给出整体的共轭斜量系。事实上，在共轭斜量法中存在着三套向量组。若 $\mathbf{r}_k \neq 0$ ，有如下重要结论：

$$\text{span}\{\mathbf{r}_0, \mathbf{r}_1, \dots, \mathbf{r}_k\} = \text{span}\{\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_k\} = \text{span}\{\mathbf{r}_0, \mathbb{A}\mathbf{r}_0, \dots, \mathbb{A}^k \mathbf{r}_0\}.$$

这个结果是共轭斜量法的核心。从不同的角度出发，我们可以将算法推广到非对称正定的情形。

🔗 论题 2.15. (非零) 共轭方向与 (非零) 残量方向之间存在着如下关系：

$$\mathbf{r}_i^\top \mathbf{r}_j = 0, \forall i \neq j, \quad \mathbf{p}_k^\top \mathbf{r}_i = \begin{cases} -\mathbf{r}_k^\top \mathbf{r}_k, & i \leq k; \\ 0 & i \geq k+1. \end{cases}$$

换言之，残量 \mathbf{r}_i 是椭圆抛物面在当前位置的法方向，而共轭方向 \mathbf{p}_i 是更快指向椭圆抛物面顶点的斜方向。从而，共轭斜量法中参数的计算可简化为

$$\alpha_k = \frac{\mathbf{r}_k^\top \mathbf{r}_k}{\mathbf{p}_k^\top \mathbb{A}\mathbf{p}_k}, \quad \beta_k = \frac{\mathbf{r}_{k+1}^\top \mathbf{r}_{k+1}}{\mathbf{r}_k^\top \mathbf{r}_k}.$$

在每步迭代中，参数的确定仅需三次内积运算。

2.5.3 收敛性分析

👉 论题 2.16. 共轭斜量法的误差按 l_2 模单调下降。

👉 论题 2.17. 共轭斜量法的误差按能量模可表示为某个泛函的极小，即

$$\|e_{k+1}\|_{\mathbb{A}}^2 = \min_{Q_k \in \mathbb{P}_k} e_0^\top \mathbb{A} \left[\mathbb{I} + \mathbb{A} Q_k(\mathbb{A}) \right]^2 e_0. \quad (2.5.15)$$

其中能量模的定义为 $\|x\|_{\mathbb{A}} = (x^\top \mathbb{A} x)^{1/2}$. 这个结论有两个重要的推论。

定理 2.18. 若矩阵 \mathbb{A} 仅有 m 个相异的特征值，则共轭斜量法至多 m 步可得真解。

定理 2.19. 记 $\kappa(\mathbb{A})$ 是矩阵 \mathbb{A} 的谱条件数。共轭斜量法满足如下的估计：

$$\frac{\|e_m\|_{\mathbb{A}}}{\|e_0\|_{\mathbb{A}}} \leq 2 \left(\frac{\sqrt{\kappa(\mathbb{A})} - 1}{\sqrt{\kappa(\mathbb{A})} + 1} \right)^m.$$

综合上述两个结论的推导过程，简单的修正可知共轭斜量法的收敛速度不仅依赖于系数矩阵的条件数，还依赖于系数矩阵的特征值聚集状态。共轭斜量法的实际迭代次数常常少于理论迭代次数，具有所谓的“超线性收敛”现象。

2.5.4 预处理共轭斜量方法

预处理技术是数值线性代数的重要技术，它可以改善问题的条件数，降低数值敏感度和提高收敛速度。

👉 论题 2.18. 算法描述。令 $\mathbb{Q} = \mathbb{C}\mathbb{C}^\top$ ，考虑等价的线性方程组

$$\mathbb{C}^{-1} \mathbb{A} \mathbb{C}^{-\top} \mathbb{C}^\top x = \mathbb{C}^{-1} b.$$

我们可以用 CG 算法求解这个等价问题，它在基础 CG 算法中增加了 $\mathbb{Q}z = g$ 的求解。

取初始方向 $r_0 = \mathbb{A}x_0 - b, z_0 = \mathbb{Q}^{-1}r_0, p_0 = -z_0$ ，做迭代

$$x_{k+1} = x_k + \alpha_k p_k, \quad \alpha_k = -\frac{r_k^\top z_k}{p_k^\top \mathbb{A} p_k}, \quad (2.5.16a)$$

$$r_{k+1} = r_k + \alpha_k \mathbb{A} p_k, \quad (2.5.16b)$$

$$z_{k+1} = \mathbb{Q}^{-1} r_{k+1}, \quad (2.5.16c)$$

$$p_{k+1} = -z_{k+1} + \beta_k p_k, \quad \beta_k = \frac{r_{k+1}^\top z_{k+1}}{r_k^\top z_k}. \quad (2.5.16d)$$

每步迭代仅增加很少的额外工作，却由于同解方程组谱条件数的明显降低，算法的收敛速度得到很大的改善。在 *Matlab* 软件中的基本命令是 `pcg()`。

因此， \mathbb{Q} 的逆矩阵求解是预处理策略的关键。这方面的研究称为预处理技术的核心内容。通常它可由基础迭代中的矩阵分裂思想给出，譬如 SSOR 方法中的

$$\mathbb{Q} = (\mathbb{D} + \omega \mathbb{L}) \mathbb{D}^{-1} (\mathbb{D} + \omega \mathbb{L})^\top. \quad (2.5.17)$$

我们还可以采用不完全 LU 分解给出预处理矩阵。对于其他类型的预处理矩阵，限于篇幅，此处不再展开介绍。

2.6 数值实验

依旧考虑 1.5 中的线性方程组 $\mathbb{T}_{n \times n} \mathbf{x} = \mathbf{b}$ 或 $\mathbb{A}_{n^2 \times n^2} \mathbf{x} = \mathbf{b}$, 真解为 $\mathbf{x}_* = (1, 1, \dots, 1)^\top$ 。在如下数值实验中均取初始向量为零向量, 误差指标为 $\mathcal{E} = 10^{-6}$ 。我们可采用 2 范数或无穷范数做度量。

❖ 练习 2.1. 编制程序实现 *Jacobi* 迭代方法和 *Gauss-Seidel* 方法。比较迭代次数与停机标准 (残量, 相邻差量, 下界估计方式) 的关系, 以及停机时对应的真实误差。

❖ 练习 2.2. 编写程序实现 *SOR* 迭代方法。观测 *SOR* 迭代方法中松弛因子 ω 的不同取值对迭代次数的影响, 并确定最佳迭代因子的取值。

❖ 练习 2.3. 对上述三种方法, 分别绘制误差下降曲线以及残量的下降曲线 (采用对数坐标系), 绘制 (按真实误差) 迭代次数与矩阵阶数倒数的关系;

❖ 练习 2.4. 编制变系数 *Richardson* 迭代方法, 绘制误差下降曲线以及残量的下降曲线。观测循环指标 m 对迭代的影响。

❖ 练习 2.5. 对 *Jacobi* 迭代进行半迭代加速 (考虑 $m=5$ 的循环迭代), 绘制误差下降曲线以及残量的下降曲线。

❖ 练习 2.6. 共轭梯度算法的研究。比较 $n = 100, 101$ 时 *CG* 算法与 *SOR* 算法的迭代次数; 绘制误差下降曲线以及残量的下降曲线; 绘制迭代次数与矩阵阶数的关系。

❖ 练习 2.7. 编制预处理 *CG* 算法, 采用 *SSOR* 做为预处理因子。取定矩阵阶数, 考察迭代次数与 ω 的关系。

第三章 线性最小二乘问题

很多的实际问题（如数据分析、线性回归）最终得到的未知数个数与约束条件个数可能是不相等的。当约束条件过多时，问题通常会转化为一个线性矛盾方程组 $\mathbf{A}\mathbf{x} = \mathbf{b}$ ，其中 $\mathbf{A} = \mathbf{A}_{m \times n}$ 为 $m \times n$ 阶矩阵ⁱ。无论是数学概念还是数值计算，矛盾方程组引起很多需要深入研究的问题。

3.1 基本概念

3.1.1 最小二乘解

📌 定义 3.1. 若 $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2$ 当 $\mathbf{x} = \tilde{\mathbf{x}}$ 达到极小，则称 $\tilde{\mathbf{x}}$ 为线性方程组 $\mathbf{A}\mathbf{x} = \mathbf{b}$ 的一个最小二乘解。

👉 论题 3.1. 若 $\mathbf{b} \in \text{Range}(\mathbf{A})$ ，由线性方程组的基本理论可知解的存在性。这个解显然是最小二乘解。否则，利用 \mathbf{b} 在 $\text{Range}(\mathbf{A})$ 及其正交补空间上的分解，可知最小二乘解是存在的。

定理 3.1. 最小二乘问题的解集与法方程组 $\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$ 的解集相同。

★ 说明 3.1. 若矩阵 \mathbf{A} 不是列满秩的，则最小二乘解是不唯一的。

3.1.2 满秩分解表示

最小二乘解可基于矩阵的满秩分解给出显式表达。

定理 3.2. 矩阵的满秩分解 $\mathbf{A}_{m \times n} = \mathbf{G}_{m \times r} \mathbf{F}_{r \times n}$ ，其中 $r = \text{rank}(\mathbf{A}) > 0$ 。

定理 3.3. 若矩阵有满秩分解 $\mathbf{A} = \mathbf{G}\mathbf{F}$ ，则有一个最小二乘解如下：

$$\mathbf{x}_{\text{LS}} = \mathbf{G}^\top (\mathbf{G}\mathbf{G}^\top)^{-1} (\mathbf{F}^\top \mathbf{F})^{-1} \mathbf{F}^\top \mathbf{b}.$$

这个解的表达式具有重要意义，它是最小二乘解集合中向量长度最小的那个 唯一解，称为极小最小二乘解。

3.1.3 矩阵的广义逆

📌 定义 3.2. 设 \mathbf{A} 是实数域上的 $m \times n$ 阶矩阵。若存在 $n \times m$ 阶实矩阵 \mathbf{X} ，满足

$$\mathbf{A}\mathbf{X}\mathbf{A} = \mathbf{A}, \quad \mathbf{X}\mathbf{A}\mathbf{X} = \mathbf{X}, \quad (\mathbf{A}\mathbf{X})^\top = \mathbf{A}\mathbf{X}, \quad (\mathbf{X}\mathbf{A})^\top = \mathbf{X}\mathbf{A},$$

则称 \mathbf{X} 是 \mathbf{A} 的 (Moore-Penrose) 广义逆，记为 $\mathbf{X} = \mathbf{A}^\dagger$ 。在复数域上的推广是简单的。

定理 3.4. 广义逆矩阵是存在唯一的。

ⁱ通常假定 $m \geq n$ ；否则问题称为不定方程组。

定理 3.5. 若有满秩分解 $A = GF$, 则 $A^\dagger = G^\top(GG^\top)^{-1}(F^\top F)^{-1}F^\top$. 换言之, 极小最小二乘解可表示为 $x_{LS} = A^\dagger b$.

🔍 论题 3.2. 若矩阵本身可逆, 广义逆与古典逆是相同的。但是, 广义逆与古典逆两个概念有着明显的区别。即使局限于方阵, 很多适用于古典逆的运算规则和性质不再成立, 如:

1. $(AB)^\dagger \neq B^\dagger A^\dagger$, $AA^\dagger \neq A^\dagger A$, $(A^k)^\dagger \neq (A^\dagger)^k$;
2. A 与 A^\dagger 的非零特征值不是互为倒数。
3. 特别要注意的一个重要性质是: 广义逆矩阵可能不再连续地依赖于原矩阵元素的变化而变化。

若矩阵的秩发生变化, 矩阵元素的微小变化会引起广义逆矩阵的元素相距甚远, 如三阶方阵 $A_\varepsilon = \text{diag}(1, \varepsilon, 0)$ 在 $\varepsilon = 0$ 附近的表現。若秩保持不变, 广义逆矩阵的元素是连续依赖的。因此说, 矩阵秩的保持在广义逆的计算中特别重要。

🌸 思考 3.1. 针对上述性质给出反例。

3.1.4 数值算法

最小二乘解的直接求解主要有两种方法, 但其数值上的表现要格外的小心处理。若矩阵亏秩而造成列向量组是线性相关 (或数值线性相关), 某些算法将无法顺利进行。但是, 如何在数值上确定和保持矩阵秩不变是个很困难的问题, 特别当矩阵本身就是亏秩的时候。故而, 在本课程中, 我们将更多地讨论列满秩矩阵的最小二乘问题数值方法。亏秩情形下的数值处理方法, 限于篇幅不做详细介绍。

正规化方法

当系数矩阵 A 是列满秩的, 此时最小二乘解是唯一的, 我们可用法方程组 $A^\top A x = A^\top b$ 求解, 或者通过扩展方程组

$$\begin{bmatrix} I_m & A \\ A^\top & O_n \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

来求解, 其中 r 为对应残量。我们可以用直接法或迭代法求解上述两个方程组, 但第二个线性方程组的数值求解比较困难。若使用直接法, 我们需要做如下变形:

$$\begin{bmatrix} A_1 & 0 & I_n \\ A_2 & I_{m-n} & 0 \\ 0 & A_2^\top & A_1^\top \end{bmatrix} \begin{bmatrix} x \\ r_2 \\ r_1 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ 0 \end{bmatrix}, \quad \text{其中} \quad \begin{bmatrix} A & r & b \end{bmatrix} = \begin{bmatrix} A_1 & r_1 & b_1 \\ A_2 & r_2 & b_2 \end{bmatrix}.$$

其中 A_1 是可逆的方阵。这种方法表面上回避了 $A^\top A$ 的直接计算, 并同时计算出残量, 但与方法方程组方法具有相同的数值困难。

正规化方法有很大的数值困难, 主要原因有二。其一是数值精度依赖于矩阵 $A^\top A$ 的条件数 $\kappa_2(A^\top A)$, 它是原矩阵条件数 $\kappa_2(A) = \|A\|_2 \|A^\dagger\|_2$ 的平方ⁱⁱ。从而, 对病态最小二乘问题, 正规化算法关于数值舍入误差的健壮性明显下降, 舍入误差对计算结果产生非常严重的影响。其二, 由于

ⁱⁱ任意矩阵的谱范数均可定义为 $\|A\|_2 = [\varrho(A^\top A)]^{1/2}$, 或者为矩阵 A 的最大奇异值。事实上, 列满秩的最小二乘问题关于解的灵敏度将主要由 $\kappa_2(A) + \|r\|_2 \kappa_2^2(A)$ 来度量, 而关于残量的敏感度只是线性的依赖于 $\kappa_2(A)$ 。

上下数值溢出会造成法方程组系数矩阵的秩发生变化，使得法方程组超级病态甚至奇异。譬如，考虑矩阵乘积：

$$\mathbb{A}_\varepsilon = \begin{bmatrix} 1 & 1 \\ \varepsilon & 0 \\ 0 & \varepsilon \end{bmatrix}, \quad \mathbb{A}_\varepsilon^\top \mathbb{A}_\varepsilon = \begin{bmatrix} 1 + \varepsilon^2 & 1 \\ 1 & 1 \end{bmatrix} \approx \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

可见：当 $\varepsilon < \sqrt{\vartheta}$ 时，法方程组已经是奇异的。其中 ϑ 为所谓的机器精度。

★ 说明 3.2. 注意到共轭斜量法的理论起点，不难发现它很适用于最小二乘问题的计算。即使系数矩阵不是列满秩的，共轭斜量法也是可用的。此时的最小二乘解不一定是极小最小二乘解。

直交化方法

为避免大规模数值计算中（谱）条件数的增大造成舍入误差的严重影响，直交化方法求解最小二乘问题是非常好的选择。直交化方法等同于矩阵的直交分解，具体方法在后续章节详细介绍。

3.2 矩阵的直交分解

通常有三种方法可以实现矩阵的直交分解。第一种是对矩阵 \mathbb{A} 的线性无关列向量组进行 Gram-Schmidt 正交化，将线性无关向量组转化为直交向量组；第二种是 Householder 镜像变换法，第三种是 Givens 平面旋转变换法，将矩阵转化为上梯形矩阵。用矩阵语言来描述，第一种方法是用三角（或梯形）矩阵右乘来实现，后两种方法是用一系列直交阵左乘来实现。

3.2.1 Gram-Schmidt 直交化方法

由于历史的原因，Gram-Schmidt (GS) 直交化方法是一个重要的直交化方法。但其数值稳定性较差，不适合大规模的数值计算。

👉 论题 3.3. 利用 Gram-Schmidt 直交化方法可得矩阵 \mathbb{A} 的直交分解：

$$(A): \mathbb{A}_{m \times n} \mathbb{P}_{n \times n} = \mathbb{Q}_{m \times r} \mathbb{U}_{r \times n}, \text{ 其中 } \mathbb{P} \text{ 为 } n \text{ 阶置换阵, } r = \text{rank}(\mathbb{A}).$$

这里的 \mathbb{Q} 是列直交阵， \mathbb{U} 是对角线元素为正的上梯形矩阵。这种表述也称为 QR 分解。

★ 说明 3.3. GS 直交化方法可通过两种不同的执行次序来实现：传统的 CGS 方法是利用当前列向量与历史列向量组的正交性，逐列计算矩阵 \mathbb{U} 的信息；而修正的 MGS 方法是利用当前列向量与未来列向量组的正交性，逐行计算矩阵 \mathbb{U} 的信息。

我们可以采用数据覆盖技术，将列直交阵 \mathbb{Q} 仍存储在矩阵 \mathbb{A} 的原有位置；只需额外开辟空间存储 \mathbb{U} 的信息。

★ 说明 3.4. 与传统的 CGS 方法相比，修正的 MGS 方法数值健壮性更好。设 $\mathbb{A}_{m \times n}$ 是列满秩矩阵，修正的 MGS 直交化过程等价于 $\mathbb{A} + \delta \mathbb{A} = \mathbb{Q}_{MGS} \mathbb{R}_{MGS}$ ，其中 $\delta \mathbb{A}$ 是扰动矩阵。舍入误差分析表明：

$$\|\delta \mathbb{A}\|_2 \leq c_{m,n} \vartheta \|\mathbb{A}\|_2, \quad \|\mathbb{Q}_{MGS}^\top \mathbb{Q}_{MGS} - \mathbb{I}\|_2 \leq c_{m,n} \vartheta \kappa_2(\mathbb{A}) + O((\vartheta \kappa_2(\mathbb{A}))^2),$$

其中 ϑ 是机器精度， $C_{m,n}$ 是绝对常数。但是当 $n > 2$ 时，传统的 CGS 方法给出的直交阵不再满足第二条性质。以 25×15 范德蒙矩阵 $\mathbb{A} = (p_i^{j-1})$ 为例，其中 p_i 是以 0, 1 为端点的等距分布点

列ⁱⁱⁱ。我们有

$$\begin{aligned}\|\mathbf{A} - \mathbf{Q}_{CGS}\mathbf{R}_{CGS}\|_2 &= 5.0 \times 10^{-16}, & \|\mathbf{Q}_{CGS}^\top \mathbf{Q}_{CGS} - \mathbf{I}\|_2 &= 5.2, \\ \|\mathbf{A} - \mathbf{Q}_{MGS}\mathbf{R}_{MGS}\|_2 &= 1.0 \times 10^{-15}, & \|\mathbf{Q}_{MGS}^\top \mathbf{Q}_{MGS} - \mathbf{I}\|_2 &= 9.5 \times 10^{-9}.\end{aligned}$$

不难看出，两种方法在列向量的直交性方面有明显的差异。但是，它们给出的 QR 乘积均很好地近似 \mathbf{A} ，是矩阵 QR 分解的一种向后稳定的算法。这是直交分解的一个优势之一。

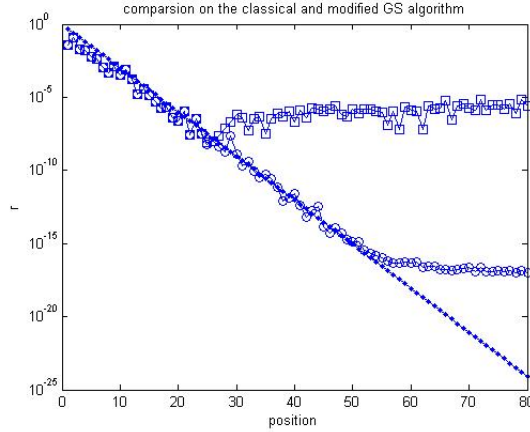


图 3.2.1: 矩阵 $\mathbf{U}\text{diag}\{2^{-k}\}_{k=1}^{80}\mathbf{V}^\top$ 的对角线元素比较，其中 \mathbf{U} 和 \mathbf{V} 是直交阵。

当矩阵的条件数非常恶劣的时候，上述两种 GS 直交化方法给出的对角线元素也存在明显的差异。请见图 3.2.1，其中目标矩阵是由一个元素快速变化的对角阵相继左乘和右乘直交阵而得到的。因此，我们不再使用传统的 CGS 方法；若再提及 GS 方法，均指修正的 MGS 方法。

★ 说明 3.5. GS 直交化过程非常适于处理列满秩矩阵。为避免除法异常停机， GS 直交化过程能够顺利执行的关键是前 r 列向量是线性无关的。若前 r 列向量线性相关，需要执行所谓的列交换或最大线性无关向量组的选取。对于列亏秩矩阵或非常病态的问题，这将引起非常困难的数值问题：舍入误差的有效控制以及矩阵秩（离散量）的确定。这个目标的数值处理方法超出本课程的范围，此处不做展开介绍。

👉 论题 3.4. 利用双 *Gram-Schmidt* 直交化过程可得矩阵的不完全直交分解：

$$(B): \mathbf{A}_{m \times n} = \mathbf{Q}_{m \times r} \mathbf{R}_{r \times r} \mathbf{V}_{n \times r}^\top, \text{ 其中 } \mathbf{R} \text{ 为 } r \text{ 阶上三角阵.}$$

这里的 $\mathbf{Q}_{m \times r}$ 和 $\mathbf{V}_{n \times r}$ 均是列直交阵。简单的列向量正交扩充可给出矩阵的完全直交分解：

$$(C): \mathbf{A}_{m \times n} = \mathbf{H}_{m \times m} \tilde{\mathbf{R}}_{m \times n} \mathbf{K}_{n \times n}^\top, \text{ 其中 } \tilde{\mathbf{R}} \text{ 是 } r \text{ 阶上三角阵 } \mathbf{R} \text{ 的零扩充阵.}$$

这里的 $\mathbf{H}_{m \times m}$ 和 $\mathbf{K}_{n \times n}$ 均是直交方阵。矩阵的完全直交分解是一个非常有用的分析工具。

3.2.2 Householder 变换

Gram-Schmidt 正交化过程对于列向量组的线性无关性有很强的依赖性，受舍入误差的积累影响比较明显，特别是列向量的正交性较差^{iv}。为此，数值计算更多地采用正交矩阵（如 *Householder*

ⁱⁱⁱ 此实验摘录于 N.J.Higham 的 "Accuracy and Stability of Numerical Algorithms" 第二版 373 页。

^{iv} 事实上，修正的 MGS 方法仅仅在这个指标上的表现明显弱于 *Householder* 方法。

变换阵) 来实现矩阵的 QR 分解。Householder 变换阵最初出现在 Turnbull 和 Aitken 的书中 (1932 年), 来证明 Schur 分解的存在性, 而后因为 Householder 将其成功地系统应用于矩阵特征值计算 (1958) 而得名。它对舍入误差的表现非常完美和稳定, 并且可以有效地用于亏秩矩阵的直交化过程。

👤 定义 3.3. 设 \mathbf{u}_n 是 n 维非零实向量, 则称

$$\mathbb{H}_{n \times n} = \mathbb{I}_{n \times n} - b^{-1} \mathbf{u}_n \mathbf{u}_n^\top, \quad \text{其中 } b = \frac{1}{2} \|\mathbf{u}_n\|_2^2, \quad (3.2.1)$$

为 Householder 变换阵。它也是对单位矩阵做秩一修正而来的初等变换阵, 被镜面法向量 \mathbf{u}_n 唯一确定。Householder 变换阵是对称正交矩阵, 它具有重要的“镜像”效应:

$$\mathbb{H}\mathbf{u} = -\mathbf{u}, \quad \mathbb{H}\mathbf{g} = 0, \forall \mathbf{g} \perp \mathbf{u}.$$

🌸 思考 3.2. 如何计算 b 或向量长度的值? 要小心数据的上溢与下溢。

👉 论题 3.5. 任意 n 阶矩阵与向量的乘积需要乘法次数共计 n^2 。但是, 对 Householder 变换阵, 所需的乘法次数可下降到 $2n + 1$, 因为

$$\mathbb{H}_{n \times n} \mathbf{g}_n = \mathbf{g}_n - b^{-1} (\mathbf{u}_n^\top \mathbf{g}_n) \mathbf{u}_n.$$

这展现出秩一修正矩阵在计算复杂度方面所具有的优势。

Householder 变换阵是一个重要的数值工具, 其核心价值体现在如下问题。

👉 论题 3.6. 已知向量 $\mathbf{a} = (a_1, a_2, \dots, a_n)^\top$, 如何构造 Householder 变换阵 $\mathbb{H}_{n \times n}$, 将 \mathbf{a} 变换到仅首个分量非零的向量, 即 $\mathbb{H}_{n \times n} \mathbf{a} = (\alpha, 0, 0, \dots, 0)^\top$? 这是数值代数的基本问题。

利用镜面反射功能, 我们很容易得到如下的算法 $[\alpha, b] = \text{householder}(\mathbf{a})$ 。见左侧的图文框。这里, 我们使用了数据覆盖技术, 镜面法向量 \mathbf{u} 保存在 \mathbf{a} 的位置上。在输出列表中, 我们用 2 个额外的浮点数存储空间记录 α 和 b ; 其中 b 的信息保留是为了减少后续的计算量。具体的 Householder 变换阵 $\mathbb{H}_{n \times n}$ 无需直接保存, 它可由 b 和 \mathbf{u} 快速给出。

1. $\alpha := -\text{sgn}(a_1) \|\mathbf{a}\|_2$;
2. $b := \alpha^2 - \alpha a_1$;
3. $a_1 := a_1 - \alpha$.

上述算法的主要技巧是通过 α 的符号选取, 我们使用了绝对值更大的 b 值, 以便减小舍入误差的影响。

🌸 思考 3.3. 如何将上述目标推广到复数域?

👉 论题 3.7. 利用前面两个基本流程, 我们可以将矩阵 $\mathbb{A}_{m \times n}$ 上三角 (梯形) 化。

1. For $k = 1, 2, \dots, n$, Do
2. 计算 $m - k + 1$ 阶矩阵 \mathbb{H}_k : $[\alpha, b] = \text{householder}(\mathbb{A}(k : m, k))$;
3. 计算矩阵乘积: $\mathbb{H}_k \mathbb{A}(k : m, k + 1 : n)$;
4. Enddo

利用数据覆盖技术, Householder 矩阵的镜面法向量 \mathbf{u} 可保存在原有位置。变换后的对角线元素记录在 α 中。

🌸 思考 3.4. 如何得到最终的直交阵?

★ 说明 3.6. Householder 方法也是矩阵 QR 分解的一种向后稳定的算法。虽然数值结果 \mathbb{Q}_{num} 与 \mathbb{R}_{num} 的计算误差可能很大, 但它们的乘积 $\mathbb{Q}_{\text{num}} \mathbb{R}_{\text{num}}$ 却非常准确地接近 \mathbb{A} 。此外, \mathbb{Q}_{num} 列向量之间的正交性得到更好的保持。Wilkinson 指出上述算法有很好的数值稳定性, 即 $\|\mathbb{H}_{\text{num}} - \mathbb{H}\|_2 \leq C\vartheta$, 其中 C 是绝对常数, ϑ 是机器精度。

3.2.3 Givens 变换

若我们仅想对某些指定的位置进行清零, 我们还可采用 Givens 平面旋转阵进行变换。它也是一种正交矩阵, 但与 Householder 变换阵不同, 其特征值均为 1。

📍 定义 3.4. 记 $c = \cos \theta$ 和 $s = \sin \theta$. 称由单位矩阵做秩一修正而来的初等变换阵

$$\mathbb{G}(i, j; \theta) = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & c & \cdots & s \\ & & \vdots & \ddots & \vdots \\ & & -s & \cdots & c \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \quad (3.2.2)$$

为 Givens 平面旋转阵。它的特征值模均为一。它不是一个对称矩阵, 也不是单位矩阵的秩一修正; 这是与 Householder 矩阵的一个区别。

📍 论题 3.8. 已知向量 $\mathbf{a} = (\cdots, x_i, \cdots, x_j, \cdots)^\top$ 。如何构造 Givens 平面旋转阵 $\mathbb{G}(i, j; \theta)$, 利用第 i 个分量将第 j 个分量旋转为零?

局限在 (i, j) 平面, 我们可得相应的算法 $[c, s] = \text{givens}(i, j, \mathbf{a})$:

1. 若 $x_j = 0$, 则 $c = 1, s = 0$;
2. 若 $|x_j| \geq |x_i|$, 通常取 $s > 0$, 即 $t = \frac{x_i}{x_j}, s = \frac{1}{\sqrt{1+t^2}}, c = st$;
3. 若 $|x_j| < |x_i|$, 通常取 $c > 0$, 即 $t = \frac{x_j}{x_i}, c = \frac{1}{\sqrt{1+t^2}}, s = ct$;

这样的处理对舍入误差的控制要好一些, 因为 $|t| \leq 1$ 。Wilkinson 指出上述算法有很好的数值稳定性, 即 $|c_{num} - c| + |s_{num} - s| \leq C\vartheta$, 其中 C 是绝对常数, ϑ 是机器精度。

★ 说明 3.7. 平面旋转阵的信息记录, 除位置信息 i 和 j 之外, 还有角度信息 c 和 s 两个数。利用 Stewart(1976) 提出的技术, 我们只需存储一个浮点数 ρ , 它可存储在原有数据的位置。互逆的两个基本过程操作如下:

1. 若 $c = 0$, 令 $\rho = 1$;
2. 若 $|s| < |c|$, 令 $\rho = \text{sgn}(c)s/2$;
3. 若 $|s| \geq |c|$, 令 $\rho = 2\text{sgn}(s)c$.

1. 若 $\rho = 1$, 令 $c = 0, s = 1$;
2. 若 $|\rho| < 1$, 令 $s = 2\rho, c = \sqrt{1-s^2}$;
3. 若 $|\rho| > 1$, 令 $c = \rho/2, s = \sqrt{1-c^2}$.

把 c 和 s 中的较小数存储起来的根本原因是, 如果 x 接近 1 时公式 $\sqrt{1-x^2}$ 的精度就低得可怜。

📍 思考 3.5. 可构造一系列 Givens 平面旋转阵 $\mathbb{G}(i_1, j_1), \dots, \mathbb{G}(i_r, j_r)$, 将 n 维向量 \mathbf{a} 变换到仅首个分量非零的向量。换言之, Householder 变换可通过一系列的 Givens 变换来实现。反之呢?

3.2.4 小结

★ 说明 3.8. 假设 $m \geq n$, 让我们比较一下上述三种方法的计算复杂度。Householder 变换与修正的 MGS 直交化过程所需的乘除法次数分别为

$$N_{opt}^{\text{House}} \approx \sum_{k=1}^n 2(n+1-k)(m+1-k) \approx mn^2 - \frac{1}{3}n^3, \quad N_{opt}^{\text{GS}} \approx \sum_{k=1}^n 2(k-1)m \approx mn^2.$$

Hoseholder 变换法的计算复杂度稍微低于 *MGS* 法。若矩阵是稠密的, *Householder* 变换所需的乘除法次数是 *Givens* 变换的一半。事实上, 前者就是为了减少后者的计算量而提出的。但是, 当矩阵 A 足够稀疏时, *Givens* 变换相比 *Householder* 变换, 在计算复杂度上占有一定的优势。因此说, 在大多数情况下, *Householder* 变换都是很好的首选方法。

★ 说明 3.9. 上述三种方法都可以给出列满秩矩阵的 *QR* 分解, 那么它们给出的 *QR* 分解是否相同呢? 实际上, 若上三角阵 U 的对角线元素符号被锁定, *QR* 分解是唯一的。这是一个非常重要的结论, 详见教科书习题 7.11。

★ 说明 3.10. 当 $m = n$ 时, *Householder* 变换也可用于线性方程组的求解, 但其运算次数是高斯消去法的两倍。

3.3 最小二乘解的各种表示

本节返回到线性最小二乘问题的直接解法。对应不同的直交化方法, 我们可以得到最小二乘解的不同显式表达。它们在计算复杂度和舍入误差的控制方面各有优缺点。

3.3.1 Gram-Schmidt 直交化方法

求解最小二乘问题的基本思想是充分利用酉变换不改变向量长度的特性, 将问题转化为某些具有特殊结构的最小二乘问题。

👉 论题 3.9. 利用矩阵的完全 *QR* 分解 (C) , 可以建立最小二乘解的一般结构

$$\mathbf{x}_{LS} = \mathbb{K} \begin{bmatrix} \mathbb{R}^{-1} \mathbf{g} \\ \mathbf{y} \end{bmatrix}, \quad \text{其中 } \mathbb{H}^T \mathbf{b} = \begin{bmatrix} \mathbf{g} \\ \mathbf{h} \end{bmatrix},$$

其中 \mathbf{y} 是任意的 $n - r$ 维向量, $\|\mathbf{h}\|_2$ 等于最小残量大小。当 $\mathbf{y} = 0$ 时, \mathbf{x}_{LS} 给出极小最小二乘解。

这种表达仅具有重要的理论意义。实际上, 完全直交分解中的直交列扩张过程没有任何的数值计算价值。我们将更多地采用如下方式表述最小二乘解。

👉 论题 3.10. 利用不完全直交分解 (B) , 可以给出极小最小二乘解

$$\mathbf{x}_{LS} = \mathbb{V}_{r \times n} \mathbb{R}_{r \times r}^{-1} \mathbb{Q}_{m \times r}^T \mathbf{b} = \mathbb{A}^\dagger \mathbf{b}.$$

不完全直交分解需要两次 *GS* 正交化过程。

👉 论题 3.11. 直接利用 *Gram-SchmidtS* 直交化过程 (A) , 可以给出极小最小二乘解

$$\mathbf{x}_{LS} = \mathbb{U}_{r \times n}^T (\mathbb{U}_{r \times n} \mathbb{U}_{r \times n}^T)^{-1} \mathbb{Q}_{m \times r}^T \mathbf{b} = \mathbb{A}^\dagger \mathbf{b}.$$

若矩阵 A 是列满秩的, 我们有更简洁的答案 $\mathbf{x}_{LS} = \mathbb{U}^{-1} \mathbb{Q}^T \mathbf{b}$ 。

★ 说明 3.11. 上述算法均要计算 $\mathbb{H}^T \mathbf{b}$ 或者 $\mathbb{Q}^T \mathbf{b}$, 这个目标均可在对增广矩阵 $[A|\mathbf{b}]$ 的列向量施行 *GS* 过程中一同实现。相比事后的计算, 这样的处理在舍入误差方面的表现要好一些。计算次序会影响舍入误差的积累。

3.3.2 直交矩阵变换

鉴于 Householder 直交变换方法具有非常好的数值稳定性, 该方法被广泛用于最小二乘问题的求解。

👉 论题 3.12. 对增广矩阵 $[A|b]$ 采用 Householder 直交变换方法进行消元 (操作流程和数据流类似于高斯消去法), 我们可得如下结果

$$\underbrace{H_r \cdots H_2 H_1}_{Q^T} [A|b] = \begin{bmatrix} R & R_1 & Q_1^T b \\ 0 & 0 & Q_2^T b \end{bmatrix} = \begin{bmatrix} U_{r \times n} & Q_1^T b \\ 0 & Q_2^T b \end{bmatrix}.$$

其中 r 为矩阵 A 的秩, R 为 r 阶可逆上三角矩阵, 正交阵 $Q = [Q_1 \ Q_2]$, 其中 Q_1 为 Q 的前 r 列。最小二乘解如何计算?

1. 若矩阵 A 是列满秩的, 则 U 是可逆的上三角阵, 此时的最小二乘解唯一, 可由 $x_{LS} = U^{-1}Q_1^T b$ 给出。
2. 若矩阵 A 是列亏秩的, 则 $x_{LS} = R^{-1}Q_1^T b$ 仅仅是一个最小二乘解。它不一定是极小最小二乘解。若要得到极小最小二乘解, 还需施行右侧的 Householder 变换。
3. 右下角 $Q_2^T b$ 的长度为最小二乘解的对应残量大小。

★ 说明 3.12. 为增强 Householder 直交变换方法的数值稳定性, 实际计算时我们还可以选取主列并进行列交换。选取原则是以具有最大长度的列向量为主列。该策略实现简单, 对矩阵亏秩的情形也非常有效。

★ 说明 3.13. Givens 平面旋转方法与 Householder 直交变换方法的实现是类似的。

3.4 奇异值分解

奇异值分解在矩阵理论中是一种非常重要的分析工具, 它是矩阵 Schur 分解和特征值分解的推广; 若矩阵是实对称正定的, 奇异值分解就是利用正交矩阵对角化的特征值分解。在 Matlab 中可通过命令 `svd()` 来实现。

定理 3.6. 设 $A \in \mathbb{R}^{m \times n}$, 则存在直交阵 $U \in \mathbb{R}^{m \times m}$ 和 $V \in \mathbb{R}^{n \times n}$, 使得

$$A = U_{m \times m} D_{m \times n} V_{n \times n}^T, \quad D = \text{generaldiag}(\sigma_1, \sigma_2, \dots, \sigma_p), \quad p = \min(m, n),$$

广义对角阵中的奇异值 σ_i 通常是按降序排列, 即 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$, 其中 $r = \text{rank}(A)$; 余下的奇异值均为零, 即 $\sigma_{r+1} = \sigma_{r+2} = \dots = \sigma_p = 0$ 。对应奇异值 σ_i , 称 U 中的第 i 列向量 u_i 为左奇异向量, 而称 V 中第 i 列 v_i 为右奇异向量, 因为

$$u_i^T A = \sigma_i v_i^T, \quad A v_i = \sigma_i u_i.$$

👉 论题 3.13. 几何含义: 矩阵可视为线性变换, 描述刚体的旋转与拉伸。奇异值分解实际上回答了如何建立从一个正交坐标系到另一个正交坐标系下的线性变换描述。

👉 论题 3.14. 矩阵秩, 值域、核空间, 以及秩一展开等。

$$\text{range}(\mathbb{A}) = \text{span}\{\mathbf{u}_i\}_{i=1}^r, \quad \text{ker}(\mathbb{A}) = \text{span}\{\mathbf{v}_i\}_{i=r+1}^n, \quad \mathbb{A} = \sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T.$$

👉 论题 3.15. 奇异值刻画了一个矩阵到低秩矩阵集合之间的距离。设 \mathbb{A} 的秩为 r ，则

$$\min_{\text{rank}(\mathbb{B})=k} \|\mathbb{A} - \mathbb{B}\|_2 = \|\mathbb{A} - \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^T\|_2 = \sigma_{k+1}, \quad \text{若 } k < r.$$

若数值计算得到的奇异值 σ_k, σ_{k+1} 分别处于机器精度两侧，则称该矩阵具有数值秩 k 。通常，我们认为数值秩与真实秩很接近或相等的。这个性质可以保证最小二乘计算的可靠性。

最后我们回答最小二乘问题的求解。

👉 论题 3.16. 设 $\mathbb{A} \in \mathbb{R}^{m \times n}$ 有奇异值分解 $\mathbb{A} = \mathbf{U} \mathbf{D} \mathbf{V}^T$ ，则极小最小二乘解可表示为

$$\mathbf{x}_{LS} = \mathbf{V} \mathbf{D}^\dagger \mathbf{U}^T \mathbf{b}.$$

这种方法具有数值稳定性强、可有效处理亏秩矩阵等优点。但是，求得矩阵的奇异值分解却需要耗费更多的机时。因此，只有当问题的病态非常严重时，我们才会采用奇异值分解方法求解最小二乘解。

★ 说明 3.14. 矩阵的奇异值分解虽然理论很漂亮，但其数值计算却很困难。类似于特征值的计算，即使没有误差，我们也不能在有限步数内得到精确的奇异值分解。常用的方法是首先采用 *Householder* 变换法，将矩阵变换为双对角线上三角阵，然后再通过一个迭代过程（类似于求解特征值的 *QR* 方法）将其转化为对角矩阵。具体内容可参阅 *Golub* 和 *Kahan* 在 20 世纪 60 年代的工作；因其超出本课程要求，故详略。在 *Matlab* 中可通过命令 `svd()` 来实现。

★ 说明 3.15. 奇异值分解在图像压缩中有很好的应用。很多现实中的图像可用矩阵表示，具有非常有限的主要结构，即有价值的主奇异值个数 k 相当很小。利用前 k 个主奇异值对矩阵 \mathbb{A} 做截断的秩一展开，即我们只需记录主奇异值 σ_i 和奇异向量 \mathbf{u}_i 和 \mathbf{v}_i ，从而数据存储量从 mn 下降到 $(m+n+1)k$ 。

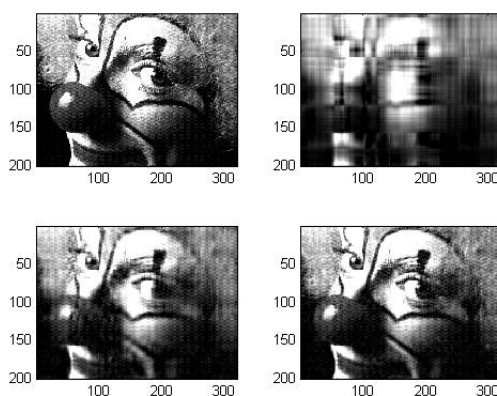


图 3.4.2: 小丑图的压缩与恢复：左上角为原图，余下三个为 $k = 5, 10, 15$ 。

在图 3.4.2 中，我们应用该策略对小丑图进行压缩与恢复。在 *Matlab* 中的实现是 `load clown.mat; [U,S,V]=svd(X); colormap('gray'); image(U(:,1:k)*S(1:k,1:k)*V(:,1:k)').`

3.5 离散数据拟合

数据拟合或线性回归就是一个最小二乘问题。我们欲建立如下类型的数据拟合模型

$$y(x) = \sum_{j=0}^n \alpha_j \phi_j(x), \quad (3.5.3)$$

近似某个真实函数 $\phi(x)$ ，其中 $\{\phi_j(x)\}_{j=0}^n$ 是线性无关的基函数， α_j 为待定参数。很多实际问题中函数 $\phi(x)$ 通常是未知的，我们想从离散数据 $\{(x_i, y_i)\}_{i=1:m}$ 给出合理的参数信息，上述目标转化为一个最小二乘问题。当利用法方程组求解这个最小二乘问题时，我们希望法方程组的系数矩阵是非奇异。

🔍 论题 3.17. 函数的最佳平方逼近问题与离散数据的最小二乘问题具有密切的联系。若 $\phi(x)$ 是已知的函数，模型问题属于函数的最佳平方逼近问题。由 $\{\phi_j(x)\}_{j=0}^n$ 的线性无关性可知对应的法方程组是对称正定的，有唯一解。

那么，针对同一个模型，离散数据的最小二乘问题对应的法方程组是否也一定可逆，有唯一解呢？这个目标不是永远成立的，它需要所谓的 Haar 条件：

对不全为零的参数组 $\{\beta_j\}_{j=0:n}$ ，方程 $g(x) = \sum_{j=0}^n \beta_j \phi_j(x)$ 的根个数不超过 n 个。

以保障最小二乘解的唯一性。由 Haar 条件可知，多项式数据拟合总是唯一存在的。

3.6 数值实验

❖ 练习 3.1. 回忆 1.5 的矩阵定义，考虑列满秩的最小二乘问题 $A_{n \times (n-1)} \mathbf{x}_{n-1} = \mathbf{b}_n$ ，其中系数矩阵和右端项分别为

$$A_{n \times (n-1)} = T_{n \times n}(1:n, 1:n-1) = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \\ & & & & -1 \end{bmatrix}, \quad (3.6.4)$$

$$\mathbf{b}_n = \left(1 + \frac{1}{n}, \frac{2}{n}, \frac{3}{n}, \dots, \frac{n-2}{n}, 1 + \frac{n-1}{n}, 0 \right)^\top. \quad (3.6.5)$$

对应的最小二乘解为 $\mathbf{x}_{LS} = (1, 1, 1, \dots, 1, 1)^\top$ 。分别用

(a) 法方程组; (b) MGS 方法; (c) Householder 变换法; 和 (d) Givens 平面旋转法

四种方法求解这个最小二乘问题。令 n 从 5 增加到 30，绘图比较上述四种算法的计算工作量（可用 cpu 时间表示）和计算精度与 n 的关系。

❖ 练习 3.2. 随机构造一个可逆方阵，利用不同方法给出它的 QR 分解。观测所得列向量的正交性和所谓的向后稳定性。

❖ 练习 3.3. 实现本章的两张图。


第四章 矩阵特征值问题

很多实际问题可以归结为矩阵的特征值问题，如结构动力学，电力网络，量子化学和物理学中某些临界值的确定等。特征值问题是一个包含非线性现象的线性代数问题，有着非常有趣的数值表现。本章将主要讨论实矩阵的特征值问题求解方法。

4.1 特征值问题的估计及其敏感度分析

本节我们回顾特征值问题的一些基本概念与著名结论。因课时有限，我们略去具体的证明过程，而仅仅介绍相关概念与结论含义。

4.1.1 预备知识

 论题 4.1. 设 $\lambda_1, \lambda_2, \dots, \lambda_r$ 是 r 个互异的特征值，则矩阵 \mathbb{A} 的特征多项式可表示为


$$\det(\lambda \mathbb{I} - \mathbb{A}) = (\lambda - \lambda_1)^{n_1} (\lambda - \lambda_2)^{n_2} \cdots (\lambda - \lambda_r)^{n_r}, \quad (4.1.1)$$


其中 n_i 称为 λ_i 的代数重数。而称 $\gamma_i = n - \text{rank}(\lambda_i \mathbb{I} - \mathbb{A})$ 为 λ_i 的几何重数，它描述特征值 λ_i 所对应的独立特征向量个数。实矩阵的特征值和特征向量也可能是复数的。


\mathbb{A} 的最小多项式是首项系数为 1 且次数最低的多项式 $p(\lambda)$ ，它满足 $p(\mathbb{A}) = 0$ 。可证最小多项式具有形式

$$p(\lambda) = (\lambda - \lambda_1)^{\ell_1} (\lambda - \lambda_2)^{\ell_2} \cdots (\lambda - \lambda_r)^{\ell_r}, \quad 1 \leq \ell_i \leq n_i. \quad (4.1.2)$$

这可给出向量空间的直和分解 $\mathbb{C}^n = V_1 \oplus V_2 \oplus \cdots \oplus V_r$ ，其中 $V_i = \ker((\mathbb{A} - \lambda_i \mathbb{I})^{\ell_i})$ 是维数为代数重数 n_i 的 \mathbb{A} -不变子空间，它包含了 λ_i 所对应的所有特征向量。

 论题 4.2. 矩阵的 Jordan 分解是矩阵理论分析的重要工具之一，它可以给出特征值和特征向量的所有信息，特别是可以判定独立特征向量个数是否有亏损。非亏损矩阵就是可对角化矩阵，此时每个特征值的代数重数与几何重数相等。但是令人遗憾的是，求 Jordan 分解的几乎所有方法在数值上都不是十分稳定的。

 论题 4.3. 矩阵的 Schur 分解表明任意矩阵可通过酉相似变换到上三角阵。更常用的是实 Schur 分解定理，即任意矩阵可通过正交矩阵相似变换到块上三角矩阵，其中对角线的矩阵块阶数至多为 2。这种分解方式在数值上更容易实现。

 论题 4.4. 描述特征向量的更好语言是特征不变子空间。设 \mathcal{P} 和 \mathcal{Q} 是维数相同的两个子空间，其相应的子空间正交投影可分别表示为幂等矩阵 \mathbb{P} 和 \mathbb{Q} ，则两个子空间 \mathcal{P} 与 \mathcal{Q} 的距离可定义为

$$\text{dist}(\mathcal{P}, \mathcal{Q}) = \|\mathbb{P} - \mathbb{Q}\|_2 = \{1 - [\sigma_{\min}(\mathbb{P}^H \mathbb{Q})]^2\}^{1/2}, \quad (4.1.3)$$

其中 $\sigma_{\min}(\mathbb{P}^H \mathbb{Q})$ 是矩阵 $\mathbb{P}^H \mathbb{Q}$ 的最小非负奇异值。

设 \mathbf{x} 和 \mathbf{y} 是具有夹角 θ 的两个单位向量。考虑两个子空间 $\mathcal{P} = \text{span}(\mathbf{x})$ 和 $\mathcal{Q} = \text{span}(\mathbf{y})$ ，其相应的正交投影矩阵为 $\mathbf{x}\mathbf{x}^\top$ 和 $\mathbf{y}\mathbf{y}^\top$ 。由上述定义可知 $\text{dist}(\mathcal{P}, \mathcal{Q}) = \sin \theta$ 。

4.1.2 特征值界定与排除定理

定理 4.1. $\rho(\mathbb{A}) \leq \|\mathbb{A}\|$.

定理 4.2. 【Gerschgorin 第一圆盘定理】矩阵 $\mathbb{A} = (a_{ij})_{n \times n}$ 的任意特征值必落在某个圆盘 S_i 内, 其中 $S_i = \{\lambda: |\lambda - a_{ii}| \leq \sum_{j \neq i} |a_{ij}|\}$.

定理 4.3. 【Gerschgorin 第二圆盘定理】设 $\mathbb{A} = (a_{ij})_{n \times n}$ 的 n 个圆盘有 m 个构成了一个联通域, 并与其它 $n - m$ 个圆盘严格分离, 则这个联通域中恰好有 m 个特征值。

4.1.3 敏感度分析

尽管特征值的变化连续依赖于矩阵元素的变化ⁱ, 但其敏感度却有很大的区别。譬如, 设 ε 是个非常小的数。简单计算可知 n 阶矩阵

$$\begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & \dots & \dots & \dots & \\ & & & 0 & 1 \\ \varepsilon & & & & 0 \end{bmatrix}$$

具有特征值 $\sqrt[n]{\varepsilon}$; 微小的数据扰动造成了特征值发生很大的改变。但是, 若将这个 ε 移到对角线上方区域, 特征值不受任何影响。

定理 4.4. 【Bauer-Fike 定理】设 \mathbb{A} 和 \mathbb{B} 为两个已知的复矩阵, 其中 \mathbb{A} 可通过矩阵 \mathbb{Q} 相似对角化。对于 \mathbb{B} 的任意特征值 $\mu \in \lambda(\mathbb{B})$, 均存在 \mathbb{A} 的一个特征值 $\lambda \in \lambda(\mathbb{A})$, 使得

$$|\lambda - \mu| \leq \|\mathbb{Q}^{-1}\| \|\mathbb{Q}\| \|\mathbb{A} - \mathbb{B}\|.$$

⊙ 定义 4.1. 注意到 Bauer-Fike 定理, 关于特征值的整体条件数通常定义为

$$\nu(\mathbb{A}) = \inf_{\mathbb{Q} \in \mathcal{D}_{\mathbb{A}}} \|\mathbb{Q}\| \|\mathbb{Q}^{-1}\|, \quad (4.1.4)$$

其中 $\mathcal{D}_{\mathbb{A}}$ 是可使 \mathbb{A} 标准化的所有相似矩阵形成的集合。这里的范数通常取为谱范数。

★ 说明 4.1. Bauer-Fike 定理的结果可进一步推广到亏损矩阵, 它们与 Jordan 分解中所含的 Jordan 块阶数有关。此时, 上述条件数的概念依旧成立。

上述的整体敏感度刻画过于粗糙, 因为特征值的敏感程度还与其局部特征信息有关。为说明这个现象, 我们以单特征值值为研究对象, 并假设特征值及其相应的特征向量关于扰动是光滑变化的。

⊙ 定义 4.2. 设 \mathbf{x} 和 \mathbf{y} 分别是矩阵 \mathbb{A} 对应单特征值 λ 的单位左特征向量和单位右特征向量, 定义局部特征值条件数为

$$W(\lambda; \mathbb{A}) = \frac{1}{\mathbf{x}^\top \mathbf{y}}. \quad (4.1.5)$$

★ 说明 4.2. 上述两个条件数定义均为酉变换下的不变量。这个重要性质告诉我们, 对矩阵进行的酉相似约化不会使特征值问题的条件数变坏。因此, 在进行特征值计算时, 我们可以放心地先将矩阵经酉相似变换约化为尽可能简单的形式, 再进行下一步的特征值问题的数值计算。

★ 说明 4.3. 无论是整体刻画还是局部刻画, 对称矩阵的特征值问题都是良态的。矩阵特征值问题的病态与线性方程组问题的病态是两个完全不同的概念。

ⁱ这可以利用复变函数中的留数定理证明, 此处略。

★ 说明 4.4. 重特征值的敏感度很复杂。譬如，重特征值的左右特征向量可以正交，根据单特征值的条件数定义，它变成无穷大。对于亏损矩阵，特征值重复或者互相靠近的特征值问题是病态的。

有关特征向量的敏感性也比较复杂，其不仅与相应的特征值的条件数有关，也和这个特征值与其他特征值的分离程度有关。但是，将敏感的特征向量放在一起形成的特征子空间却可以是不敏感的。具体讨论超出课程设置，可参见 Wilkinson 的专著。

4.2 幂法

4.2.1 正幂法

🔍 论题 4.5. 基本思想：利用不同特征值的几何增长速度差异，我们可将初始向量中的不同特征成份进行筛选和分离。典型思想来源于如下公式

$$\mathbf{A}^k \mathbf{v}_0 = \sum_{1 \leq j \leq n} \alpha_j \lambda_j^n \mathbf{x}_j = \lambda_1^n \sum_{1 \leq j \leq n} \alpha_j \left(\frac{\lambda_j}{\lambda_1} \right)^n \mathbf{x}_j.$$

这里假设了矩阵 \mathbf{A} 具有完备的特征向量系（仅具有线性初等因子），并将所有特征值按模降序排列。其中， λ_1 是可分离的主（实）特征值，因其绝对值在所有特征值中最大。

上述想法很自然地使我们想要利用 $\mathbf{A}^k \mathbf{v}_0 / \lambda_1^k$ 来求解矩阵 \mathbf{A} 的近似特征向量。然而，在实际计算时，这个想法却是行不通的。因为要求的主特征值是未知的，而且 \mathbf{A}^k 的计算工作量太大。

🔍 论题 4.6. 数值操作的考量：主要技术是适当的向量单位化，以避免幂次乘积的数值上下溢出。算法基本结构为

$$\mathbf{u}_k = \mathbf{A} \mathbf{v}_{k-1}, \quad m_k = \max(\mathbf{u}_k), \quad \mathbf{v}_k = \frac{\mathbf{u}_k}{m_k},$$

其中 $\max(\mathbf{u}_k)$ 表示向量 \mathbf{u}_k 按模最大的首个分量值。

★ 说明 4.5. 要在上述迭代过程收集到主特征信息，理论上需要初始向量在主特征子空间 $\text{span}(\mathbf{x}_1)$ 上的分量必须非零。但是，我们不用过分担心这个条件是否满足。若这个分量非零或接近零时，其数值表现为一个非常缓慢的假收敛过程。此时，舍入误差的积累可能起到积极的作用。我们也可更换初始向量。

主要的收敛结果如下：

定理 4.5. 幂法中的 m_k 收敛到主特征值 λ_1 ，即 $m_k = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$ ，其中 λ_2 是按模次优的特征值。前两个主特征值的差异决定了收敛的速度。同时，幂法中的向量 \mathbf{v}_k 收敛ⁱⁱ到主特征向量 \mathbf{x}_1 。

🔍 论题 4.7. 上述结果可推广到更一般的情形，即主特征值是半单的（特征向量系无亏失，或几何重数等于代数重数）且与其他特征值是按模分离的。若主特征值不能有效地与其他特征值分离，幂法的推广不甚理想。

ⁱⁱ 其真正含义应是收敛到特征子空间，即 $\text{span}(\mathbf{v}_k) \rightarrow \text{span}(\mathbf{x}_1)$ ，或者等价于这两个空间的距离趋于零。

1. 特征向量系的完备很重要，等模的主特征值有如下三种情形。
 - (a) 主特征值是实重根：此时幂法可有效的简单推广，得到的特征向量与初始向量密切相关。
 - (b) 主特征值互为相反数；此时幂法本身不收敛，但我们可以考虑两步幂法得到相应的信息。
 - (c) 主特征值为等模的共轭复数：幂法虽可推广，但过程复杂，效率不高。要利用最小二乘技术，对小虚部的特征值计算数值稳定性很差。
2. 若主特征值是重根并造成特征向量系的亏失，幂法将不再具有所谓的几何收敛速度，而仅有极其缓慢的调和收敛速度。

✿ 思考 4.1. 设 $\mathbb{J}_{n \times n}$ 是具有单一特征值 λ 的单个 Jordan 阵。对任意的初始向量 \mathbf{v}_0 ，计算 $\mathbb{J}_{n \times n}^k \mathbf{v}_0$ ，并观察 $k \rightarrow \infty$ 的情形。

作为本节结束，我们指出幂法的计算公式和收敛情形依赖于特征值的分布情形，因此实际使用时很不方便，特别是不适合自动计算。只有在矩阵阶数非常高，无法利用其它高效算法，才用幂法求解少量的按模最大的几个特征值及其特征向量。然而，幂法的基本思想是非常重要的，由它可诱导出一些更有效的算法。

4.2.2 加速技巧

👉 论题 4.8. 由定理 4.5 可知 m_k 以线性收敛速度趋于主特征值。此时，Aitken 加速技巧是一种非常有效的加速方法。该方法也称为 Δ^2 方法，因为

$$\tilde{m}_k = m_k - \frac{(\Delta m_k)^2}{\Delta^2 m_k}, \text{ 其中 } \Delta m_k = m_{k+1} - m_k.$$

更多的讨论在下一章给出。

👉 论题 4.9. 若 \mathbb{A} 是实对称矩阵，我们还可采用 Rayleigh 商加速法：

$$R(\mathbf{v}_k) = \frac{\mathbf{v}_k^\top \mathbb{A} \mathbf{v}_k}{\mathbf{v}_k^\top \mathbf{v}_k} = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right).$$

与定理 4.5 相比，线性收敛的速度提高至原速度的平方倍。此时的向量单位化可采用欧几里得范数，不用再计算 m_k 。

★ 说明 4.6. Rayleigh 商是一个非常重要的量，在对称矩阵的特征值刻画中占有重要地位。此外， $R(\mathbf{x})$ 是关于 μ 的矛盾方程组 $\mathbb{A}\mathbf{x} - \mu\mathbf{x} = 0$ 的最小二乘解，它表示离 \mathbf{x} 最近的特征子空间。

👉 论题 4.10. 原点平移方法是最简单易行的加速方法，但平移量在幂法中的有效确定是个很难的问题。这个技巧更多地用于其他方法，若其有很好的策略确定平移量。

4.2.3 反幂法

反幂法可给出按模最小的非零特征值及其特征向量。其基本思想就是对矩阵 \mathbb{A}^{-1} 施行正幂法，但却具有一些与幂法不同的性质。

★ 说明 4.7. 半次迭代: 方程组的求解建立于 \mathbb{A} 的 LU 分解, 在每步迭代中我们只需做两个三角形线性方程组的求解。因初始向量选取的任意性, 我们可略去第一个半步计算。

★ 说明 4.8. 利用原点位移方法, 反幂法可以求出按模意义下最接近某个指定数值 q 的特征值及其特征向量。事实上, 反幂法更多地被用于求特征向量。

★ 说明 4.9. 若我们用某种方法已经得到某个特征值的近似值 q 之后, 我们可应用反幂法于矩阵 $\mathbb{A} - q\mathbb{I}$, 求出对应这个特征值的近似特征向量。通常只需迭代一次就可得到足够好的近似特征向量。第二次迭代一般不会给出更好的近似特征向量。

在第一步迭代中, 所谓舍入误差的坏影响却在实际上起到了正面的效果。简单地说, 线性方程组的求解误差主要造成其解在特征子空间 $\text{span}\{\mathbf{x}_1\}$ 上的投影长度改变。误差越大, 在特征子空间上的投影越大。这对于我们计算近似特征向量而言是非常有利的, 因为此时关心的对象是方向, 而不是大小。影响逆迭代结果的主要因素是 \mathbb{A} 关于所求特征值的条件数。

★ 说明 4.10. 若 \mathbb{A} 是对称矩阵, 我们可同时采用原点平移技术和 Rayleigh 商技术。这形成著名的 Rayleigh 商算法: 任给单位向量 \mathbf{q}_0 , 计算 $\mu_0 = \mathbf{q}_0^H \mathbb{A} \mathbf{q}_0$ 。做迭代

$$(\mathbb{A} - \mu_k \mathbb{I}) \mathbf{v}_k = \mathbf{q}_{k-1}, \quad \mathbf{q}_k = \mathbf{v}_k / \|\mathbf{v}_k\|_2, \quad \mu_k = \mathbf{q}_k^H \mathbb{A} \mathbf{q}_k.$$

该算法也可用于非对称矩阵。通常, 我们用 $\rho_k = \|(\mathbb{A} - \mu_k \mathbb{I}) \mathbf{q}_k\|_2$ 刻画 μ_k 靠近某个特征值的程度。可以证明: 对于任意的矩阵 \mathbb{A} , 均有 ρ_{k+1} 被 ρ_k^2 所控制, 即算法具有所谓的平方收敛速度。当矩阵 \mathbb{A} 是对称的, 我们可获得三次方收敛速度。

4.2.4 其他特征值的求解

降维法

降维方法是数值计算的常用方法。就此处而言, 就是应用幂法求解某个低阶矩阵的主特征值, 其核心内容称之为矩阵收缩技术, 并建立两个矩阵的特征信息关联。

🔗 论题 4.11. 关键步骤是找到一个可逆矩阵 \mathbb{S}_1 , 将已知的主特征向量转化为仅首位置非零的向量, 即 $\mathbb{S}_1 \mathbf{x}_1 = t \mathbf{e}_1$ 。我们可采用 Gauss 消元阵、Householder 变换阵或 Givens 旋转阵。

降维方法的诟病是逐次求解造成舍入误差的显著积累, 以及收缩技术对矩阵稀疏结构的破坏。后者更加抵消了幂法的简洁性与优势。

Wielandt 收缩法

设我们已经有幂法得到主特征信息 $(\lambda_1, \mathbf{x}_1)$, 为求次特征信息, 我们对矩阵 \mathbb{A} 进行秩一修正, 考虑 $\mathbb{A}_1 = \mathbb{A} - \sigma \mathbf{x}_1 \mathbf{v}^H$, 其中 σ 和 \mathbf{v} 是待定的信息。通常, 取 $\sigma = \lambda_1$ 和 $\mathbf{v} = \mathbf{x}_1$ 。这种方法对对称矩阵是非常有效的, 但也可用于非对称情形。

上述处理的优势是我们不必存储 \mathbb{A}_1 , 而是只要存储 \mathbf{v} 和 σ 就行了; 从而, 原始矩阵的数据不用改变。

同时迭代法

为此, 我们希望在破坏矩阵元素的情形下, 建立一个算法能够同时求出矩阵的前几个主特征值及其特征向量空间。下面介绍的是子空间同时迭代法就可以用于这个目的。

🔗 论题 4.12. 原始的子空间法: 取 m 个列直交向量组成初始矩阵 \mathbb{V}_0 , 然后循环计算

$$\mathbb{U}_k = \mathbb{A}\mathbb{V}_{k-1}, \quad \mathbb{U}_k = \mathbb{V}_k\mathbb{R}_k.$$

其中第一步是基本的正幂法；第二步是对（斜）像空间重构新的列正交基底，以避免特征空间的数值坍塌。

为简便，我们假设矩阵 \mathbb{A} 是对称的；此时的特征值都是实数。此时的收敛分析可以清楚地给出，并且进行很好的数值加速。

🔍 论题 4.13. 子空间投影算法：这是一个非常有价值的数值计算思想，可以有效地克服维数危机。它可视为 Rayleigh 商技术的推广，我们试图在一个具有较低维数的子空间中求解原有特征问题的近似解。

设子空间基底由 \mathbb{V}_{k-1} 的 m 个列直交向量所构成。设想所求的特征向量被局限在这个子空间（或最小二乘思想），我们可得一个小规模的 m 阶特征值问题

$$\mathbb{V}_{k-1}^\top \mathbb{A} \mathbb{V}_{k-1} \mathbf{y}_{k-1} = \tilde{\lambda}_{k-1} \mathbf{y}_{k-1}.$$

显而易见，这个小规模特征问题的计算要比原有问题要容易很多。我们可以认为 $\tilde{\lambda}$ 是原有矩阵 \mathbb{A} 的一个近似特征值，对应的近似特征向量可以表示为 $\mathbb{V}_{k-1} \mathbf{y}_{k-1}$ 。

教科书中的子空间同时迭代法就是上述两个算法的合并。

定理 4.6. 设对称实矩阵 \mathbb{A} 的特征值互异，则子空间同时迭代法关于特征值和特征向量均具有很好的收敛性质。

证明：该算法的收敛性证明是典型的。 □

★ 说明 4.11. 初始正交列向量组的选取方式主要有两种。其一是 (a) 由一个随机向量构造 Householder 矩阵，然后再任取一些列向量；其二是 (b) 随机构造一组列向量，然后进行 Gram-Schmidt 正交化。大量的数值经验表明后者要好一些。

★ 说明 4.12. 算法中低阶方阵的特征值和特征向量可用 Jacobi 方法解决。当 k 适当大的时候，这个低阶矩阵非常接近对角阵，Jacobi 方法特别有效和快捷。

4.3 实对称矩阵的 Jacobi 方法

回忆矩阵论中的基本结论：若 \mathbb{A} 是实对称矩阵，则必存在实正交阵使其对角化。本节拟采用此策略，施行一系列简单的正交矩阵做相似变换，使 \mathbb{A} 本质上相似趋于某个对角阵。这个古老的算法是由 Jacobi 于 1846 年提出。由于其编程简单，并行效率高的特点，近年来又重新受到人们的重视。

4.3.1 基本思想

Jacobi 方法的基本思想可以在二阶对称矩阵的相似变换实现中得到体现，即寻找一个 Givens 平面旋转阵 $\mathbb{G}(p, q) \equiv \mathbb{G}(p, q; \theta)$ ，使得 $b_{pq} = 0$ ，其中 a_{pq} 称为旋转主元：

$$\begin{bmatrix} b_{pp} & b_{pq} \\ b_{pq} & b_{qq} \end{bmatrix} = \mathbb{G}(p, q) \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix} \mathbb{G}(p, q)^\top,$$

称这样的操作过程为一次 Jacobi 旋转变换。其中旋转角度 θ 可如下确定：

$$\xi = \cot 2\theta = \frac{a_{pp} - a_{qq}}{2a_{pq}}, \quad \theta \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right]. \quad (4.3.6)$$

👉 论题 4.14. 欲计算平面旋转矩阵 $\mathbb{G}(p, q)$ 中的参数 $c = \cos \theta$ 与 $s = \sin \theta$, 我们通常由(4.3.6)计算出 $t = \tan \theta$. 通常, 我们要求 $|t| \leq 1$. 相应的基本计算公式为

$$t = \operatorname{sgn}(\xi) \left[|\xi| + \sqrt{1 + \xi^2} \right]^{-1}, \quad c = \frac{1}{\sqrt{1 + t^2}}, \quad s = ct. \quad (4.3.7)$$

但是, 该公式不适用于 a_{pq} 很小而 $a_{pp} \neq a_{qq}$ 的时候. 此时, 非常大的 ξ 会导致有效位数的大量损失, 从而计算结果总是 $c = 1$ 和 $s = 0$, 迭代无法继续下去. 此时, 算法修正如下:

$$T = \tan \frac{\phi}{2} = \frac{a_{pq}}{2(a_{pp} - a_{qq})}, \quad \cos \phi = \frac{1 - T^2}{1 + T^2}, \quad \sin \phi = \frac{2T}{1 + T^2}. \quad (4.3.8)$$

后两个数关于 c 和 s 有很好的近似结果.

🌀 思考 4.2. 证明: 当 $\theta \rightarrow 0$ 时, 有 $\phi - \theta \approx \frac{5}{4}\theta^3$.

★ 说明 4.13. 当 $a_{pq} = 0$ 时, 我们不需要做 *Jacobi* 旋转. 若 $|a_{pq}| < \mathcal{E}\sqrt{a_{pp}a_{qq}}$, 我们可数值上认定 a_{pq} 为零, 其中 \mathcal{E} 是事先给定的一个关值, 如 $\mathcal{E} = 10^{-14}$.

4.3.2 古典 *Jacobi* 方法

当将上述思想推广到高阶矩阵时, 旋转矩阵的相似变换会造成井字线上的数据发生更替, 这可能将那些已经清零的非对角元素重新变得非零. 换言之, 相似对角化过程不再可以在有限步数内结束

👉 论题 4.15. 古典 *Jacobi* 方法: 首先选择主元 a_{pq} 为当前矩阵中按绝对值最大的非对角线元素, 然后做 (p, q) 平面上的 *Jacobi* 旋转.

定理 4.7. 古典 *Jacobi* 方法给出的矩阵序列是本质收敛到某个对角矩阵. 所谓的本质收敛是指几何意义下的收敛.

证明: 非对角矩阵的 Frobenius 范数趋于零. □

★ 说明 4.14. *Jacobi* 方法中要求 $|t| \leq 1$ 是有实际意义的, 它可以保证算法的真正收敛, 即对角线元素目标一致地趋向某个特征值, 不会发生位置上的跳转. 换言之, *Jacobi* 方法给出的矩阵序列收敛到某个固定的对角阵. 这要用到如下引理: 设 $\{\mathbf{u}_k\}_{k=0}^{\infty}$ 是有限维赋范空间的有界序列. 若它最多有有限个聚点且 $\lim_{k \rightarrow \infty} \|\mathbf{u}_{k+1} - \mathbf{u}_k\| = 0$, 则 \mathbf{u}_k 收敛到某个聚点.

★ 说明 4.15. 若 \mathbb{A} 的特征值互异, 我们可以证明 *Jacobi* 方法给出的特征向量也是收敛的. 若 \mathbb{A} 有重特征值, 虽不再保证特征向量的收敛性, 我们仍能得到非常好的特征向量的近似, 它是(多维)特征子空间中的某个特征向量.

★ 说明 4.16. *Jacobi* 方法是数值稳定的. Demmel 和 Veselić 指出: *Jacobi* 方法的相对误差被 $\vartheta \kappa_2(\mathbb{D}^{-1/2} \mathbb{A} \mathbb{D}^{-1/2})$ 所控制, 其中 $\mathbb{D} = \operatorname{diag}(\mathbb{A})$, ϑ 为机器精度.

👉 论题 4.16. Givens 平面旋转相似变换仅影响井字线上的元素. 利用旋转角度的确定公式, 交叉点处的元素可按如下简化公式计算:

$$a_{pp} := a_{pp} + ta_{pq}, \quad a_{qq} := a_{qq} - ta_{pq}.$$

注意到一次 *Jacobi* 旋转可分解为 Givens 平面旋转阵的左乘和右乘运算, 对那些非交叉点的井字线元素, 我们仅需做一次矩阵乘法. 考虑到矩阵的对称性, 不难发现每次迭代所需乘除法次数仅为 $O(4n)$.

4.3.3 循环 Jacobi 方法

🔍 论题 4.17. 循环 Jacobi 方法就是对矩阵的严格上三角元素依次进行 Jacobi 旋转, 不再消耗机时进行全局搜索旋转主元。通常, 称这样的 $n(n-1)/2$ 次 Jacobi 旋转过程为一次“扫描”。

由于不需要寻找最佳的旋转平面, 因此要比经典的 Jacobi 方法快得多, 并且非常适合并行计算。

★ 说明 4.17. Schonhage(1964) 和 Van Kempen(1966) 的工作指出: 若循环 Jacobi 方法收敛, 则该方法具有渐进平方收敛速度。此外, Brent 和 Luk(1985) 指出最终的扫描次数是与 $\log n$ 成正比。

🔍 论题 4.18. 循环 Jacobi 方法会遇到旋转主元 a_{pq} 接近于零的状态, 此时的 Jacobi 旋转在数值上没有任何有益的贡献。为此, 我们可采用所谓的阈值扫描策略: 设 $\sigma \geq n$ 是一个固定的常数, 逐步设置阈值直至达到机器精度为止。

$$\delta_0 = \mathbb{E}_0, \quad \delta_k = \delta_{k-1}/\sigma, \quad k = 1, 2, \dots$$

🌀 思考 4.3. 证明阈值扫描策略下的 Jacobi 方法给出的矩阵序列也是收敛的。

★ 说明 4.18. 在求解一般稠密矩阵的特征值问题时, Jacobi 方法不如 QR 方法的收敛速度快。然而由于 Jacobi 方法适合于并行化的特点, 它近年来又重新引起人们的重视。将指标集 $(1, 1), \dots, (1, n), \dots, (n-1, n)$ 轮换分组, 分别执行平面旋转矩阵左乘和右乘, 便可实现循环 Jacobi 方法的并行化。

4.3.4 特征向量的计算

🔍 论题 4.19. Jacobi 方法的优点之一就是计算特征向量特别方便。它可利用所有的旋转阵信息快速恢复, 而每个 Jacobi 迭代信息的存储仅需两个整数记录位置 (i, j) 和一个浮点数记录角度信息。

4.4 对称矩阵的 Givens-Householder 方法

该方法主要包括三个步骤。其中是经有限步相似约化为三对角化矩阵, 然后是二分法求解特征值, 最后利用反幂法求解特征向量。

4.4.1 三对角化策略

注意到 Givens 旋转相似变换不可能在有限步数内到达对角阵, 其主要原因是后续操作影响已经处理过的部分。那么, 我们在有限步数内能达到的最简单矩阵是什么呢? 它就是所谓的三对角矩阵。

🔍 论题 4.20. 我们可采用 Givens 平面旋转阵来实现矩阵的相似三对角化。主要思路是以副对角线元素及其下方某个元素构造旋转平面, 将副对角线下方的所有元素依次清零。这个目标与 Jacobi 方法截然不同, 所需的旋转角度计算公式也是完全不同的。相应的旋转信息可存储在原有位置。

🔍 论题 4.21. 我们也可采用 Householder 变换阵来实现矩阵三对角化。其基本设计思想是构造 Householder 变换阵 \mathbb{H}_{n-k} , 使对角线下方的列向量 $\mathbf{a}_k = (a_{k,k+1}, a_{k,k+2}, \dots, a_{kn})^\top$ 转化为仅首个位置非零。假设左上角矩阵 \mathbb{A}_k 已经被三对角化, 我们需进行变换

$$\begin{bmatrix} \mathbb{I}_k & 0 \\ 0 & \mathbb{H}_{n-k} \end{bmatrix} \begin{bmatrix} \mathbb{A}_k & 0 \\ 0 & \mathbf{a}_k^\top \end{bmatrix} = \begin{bmatrix} \mathbb{A}_k & 0 \\ 0 & \mathbb{H}_k \mathbf{a}_k \end{bmatrix} \begin{bmatrix} 0 & (\mathbb{H}_k \mathbf{a}_k)^\top \\ \mathbb{H}_k \mathbb{A}_{n-k} & \mathbb{H}_k \mathbb{A}_{n-k} \end{bmatrix}.$$

其中的表述采用了数据覆盖技术, *Householder* 变换矩阵的信息可保存在相应的位置。

★ 说明 4.19. 若矩阵 \mathbb{A} 稠密 (所有元素均非零), *Hoseholder* 方法与 *Givens* 方法相比, 其乘法次数由 $\frac{4}{3}n^3$ 下降到 $\frac{2}{3}n^3$, 开根号次数由 $\frac{1}{2}n^2$ 下降到 $n-2$ 。

✿ 思考 4.4. 能否用 *Gauss* 消元阵完成上述三对角化过程?

4.4.2 三对角对称矩阵的二分法

至此, 特征值问题的求解转化为一个对称三对角矩阵的特征值问题求解。设 $\mathbb{T}_{n \times n}$ 是一个不可约的三对角对称矩阵, 其对角元素记为 α_i , 非对角元素记为 $\beta_i \neq 0$ 。若某个非对角元素为零, 则问题可分割为两个小规模的特征值问题。

🔗 定义 4.3. 记矩阵 $\mathbb{T}_{n \times n} - \lambda \mathbb{I}_{n \times n}$ 的第 i 阶顺序主子式为 $p_i(\lambda)$ 。它是一个 i 次多项式, 满足递推关系式 (补充定义 $p_0(\lambda) = 1$):

$$p_i(\lambda) = (\alpha_i - \lambda)p_{i-1}(\lambda) - \beta_i^2 p_{i-2}(\lambda).$$

定理 4.8. 上述多项式满足如下性质:

1. $\text{sgn } p_i(-\infty) = 1, \text{sgn } p_i(+\infty) = (-1)^i$;
2. 相邻两个多项式没有公共根;
3. 若 $p_i(\mu) = 0$, 则 $p_{i-1}(\mu)p_{i+1}(\mu) < 0$;
4. $p_i(\lambda)$ 的根全是单根, 并且 $p_i(\lambda)$ 的根严格分隔 $p_{i+1}(\lambda)$ 的根。

🔗 定义 4.4. 记一个 *Sturm* 序列 $\{p_i(\mu)\}_{i=0}^k$ 中相邻两个数符号相同的数目为 $S_k(\mu)$, 其中 μ 为给定的实数。若 $p_i(\mu) = 0$, 则称 $p_i(\mu)$ 的符号与 $p_{i-1}(\mu)$ 的符号相反, 而称 $p_{i+1}(\mu)$ 的符号与 $p_i(\mu)$ 的符号相同。

对高阶多项式, *Sturm* 序列 $\{p_i(\mu)\}_{i=0}^k$ 的计算存在严重的舍入误差和上溢的风险。为避免此数值风险, 我们经常采用如下算法统计变号数目:

$$\text{令 } q_1(\mu) = \alpha_1 - \mu, \text{ 依次计算 } q_i(\mu) = \alpha_i - \mu - \frac{\beta_i^2}{q_{i-1}(\mu)}.$$

易知, $S_k(\mu)$ 正好是数列 $q_1(\mu), q_2(\mu), \dots, q_k(\mu)$ 中非负数的数目。注: 此处比值的含义应按极限概念来理解。若 $q_{i-1}(\mu) = 0$, 则定义 $q_i(\mu) = -\infty$ 。

定理 4.9. 对任意给定实数 μ , $p_r(\lambda)$ 恰有 $s_r(\mu)$ 个根严格大于 μ 。

🔗 论题 4.22. *Sturm* 序列的应用。

1. 界定特征值范围并隔离特征值在一个有限区间; $a \leq \lambda \leq b$.
2. 取中点位置 $c = (a + b)/2$, 计算符号一致数 $s_n(c)$.
3. 折半缩减区间 $[a, b]$ 到 $[a, c]$ 或 $[c, b]$.
4. 返回到第 2 步, 直到达到用户要求。

★ 说明 4.20. 显然二分法无条件收敛。尽管舍入误差会限制它可能达到的计算精度, 但从标准的浮点运算误差分析可证该方法是数值稳定的。

4.4.3 特征向量的确定

利用直接法 (令 $x_1 = 1$) 求解三对角矩阵的特征向量不是一个好的选择, 因为其数值稳定性很差。我们可采用反幂法改善特征值的近似程度, 并求出相应的特征向量。

最后, 利用相似约化过程中的信息, 回复原始矩阵的特征向量。

4.5 QR 方法

QR 方法是目前计算中小规模稠密矩阵的全部特征信息的最有效方法之一, 其有效实现包含了很多的数值技巧。它与 Schur 分解有密切关系, 利用正交相似变换将矩阵逐步约化为上三角阵或拟上三角阵。该算法具有平方收敛速度。若矩阵是实对称的, 则可达到三次方收敛速度。

4.5.1 基本思想

👉 论题 4.23. 算法: 记 $A_1 = A$. 在之后的每步迭代中, 依次做矩阵的 QR 分解和交换次序做矩阵乘积

$$A_k = Q_k R_k, \quad A_{k+1} = R_k Q_k,$$

其中 Q_k 是正交阵, R_k 是上三角阵。不难发现, 这个迭代序列的任意两个矩阵是正交相似的。

QR 方法的收敛性比较复杂。

定理 4.10. 若矩阵 A 的所有特征值是按模可严格分离, 且以 A 的左特征向量为行组成的矩阵有 LU 分解, 则上述 QR 方法给出的矩阵 A_k 本质上收敛到上三角阵。事实上, 特征值的按模分离是结论的关键; 此时特征值必为实数。

👉 论题 4.24. QR 方法与幂法有密切的联系, 因为

$$A^k = Q_1 Q_2 \cdots Q_k R_k \cdots R_2 R_1 = \tilde{Q}_k \tilde{R}_k.$$

我们可以利用正幂法阐述第一列的收敛情形。利用反幂法阐述最后一列的收敛情形。它与 Rayleigh 商迭代有密切的联系, 可证右下角最后一个元素的收敛速度是最快的。

4.5.2 数值实现

上述基本 QR 方法的计算复杂度很高, 在特征值问题的计算中缺乏足够的竞争力。一些计算细节的改进将大大提高了 QR 算法的生命力。

👉 论题 4.25. 为降低 QR 方法每步的计算量, 通常先采用 Householder 镜像阵将矩阵相似变换到一个上 Hessenberg 矩阵。这部分内容与对称矩阵的三对角化方法是一致的, 仅仅是非对称的上三角部分需要花费额外的计算来完成。整个计算过程需要乘除法次数约为 $O(5n^3/3)$ 。

👉 论题 4.26. 真正的 QR 方法是针对上 Hessenberg 矩阵实现的。此时的矩阵 QR 分解可采用一系列的 Givens 平面旋转阵来实现, 每个列向量可用对角线元素将副对角线上的元素旋转为零。一次完整 QR 迭代过程仅需乘除法次数 $O(2n^2)$ 。

但是, 交换相乘后形成的相似变换阵不再是上 Hessenberg 矩阵了。为保持计算过程中矩阵 Hessenberg 结构的不变性, 我们可采用 Givens 平面旋转阵的错时相乘方式。

👉 论题 4.27. 为加速 QR 方法的迭代收敛速度, 带原点位移是一个很好的技术。

$$A_k - t_k I = Q_k R_k, \quad A_{k+1} = R_k Q_k + t_k I.$$

此时, 成功的平移策略主要有如下两种。

第一种是直接以右下角元素为位移量, 即 $t_k = a_{nn}^{(k)}$ 。第二种是以右下角的二阶矩阵最靠近右下角元素的特征值为位移量, 即所谓的 Wilkinson 位移量。

Wilkinson 策略特别适合于对称矩阵。设对称三对角阵为 $\text{tridiag}(\beta_i, \alpha_i, \beta_i)$, 其中 α_i 是对角元素, β_i 是副对角线元素。取

$$t_m = \alpha_n + \delta - \text{sgn}(\delta) \sqrt{\delta^2 + \beta_{n-1}^2}, \quad \text{其中 } \delta = (\alpha_{n-1} - \alpha_n)/2.$$

★ 说明 4.21. 位移策略不保证迭代一定收敛。譬如, 二阶置换阵有等模的两个特征值, 带第一种位移的 QR 迭代序列会出现循环。

👉 论题 4.28. 副对角线元素为零的数值判断准则:

★ 说明 4.22. 目前 Matlab 用于求解多项式之根的函数 `roots()` 就是将著名的 QR 方法应用到多项式 $p_n(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$ 所对应的友矩阵

$$\begin{bmatrix} 0 & & & -a_0 \\ 1 & 0 & & -a_1 \\ & 1 & \ddots & \vdots \\ & & \ddots & 0 & -a_{n-2} \\ & & & 1 & -a_{n-1} \end{bmatrix}.$$

这一方法还未利用友矩阵的特点, 因此所需的计算量为 $O(n^3)$ 。最近的研究表明, 这个计算量可以下降到 $O(n^2)$, 具体细节略。

4.5.3 隐式对称 QR 方法

实对称矩阵的特征值均为实数, 我们可用一步位移方法求解全部的特征值。从一个上 Hessenberg 矩阵ⁱⁱⁱ 相似变换到下一个上 Hessenberg 矩阵的过程还可用隐含的方式来实现, 即所谓的“驱逐出境”策略。

👉 论题 4.29. 其理论基础是任意矩阵上 Hessenberg 化的唯一性:

ⁱⁱⁱ 对称上 Hessenberg 矩阵就是对称三对角矩阵。

设 A 有两个相似分解, 即 $U^T A U = H$ 和 $V^T A V = G$, 其中 U, V 是正交阵, H, G 是不可约的上 Hessenberg 阵。若 U 和 V 的第一列相等, 则存在对角阵 $D = \text{diag}\{\pm 1\}$, 使得 $U = VD, H = DGD$ 。


假设在 QR 方法的过程矩阵一直保持不可约, 我们有如下的通用算法:


1. 计算 Wilkinson 位移 μ ; 令 $x = T(1, 1) - \mu, y = T(2, 1)$;
2. For $k = 1 : n - 1$, Do
3. 计算 $[c, s] = \text{GIVENS}(x, y)$ 将列向量 $(x, y)^T$ 中的 y 旋转消灭;
4. 计算 $T = G(k, k+1)TG(k, k+1)^T$;
5. 若 $k < n - 1$, 令 $x = T(k+1, k), y = T(k+2, k)$.
6. Enddo

若不可约结构被破坏, 则表明矩阵可以分块降阶。若对称三对角矩阵采用斜对角线方式存储, 第 4 步和第 5 步的伪代码可做相应修改。

4.5.4 双重位移的 QR 方法

若课程时间允许, 我们可考虑介绍本节的补充内容。

 论题 4.30. 若实矩阵有共轭复特征值, 其应含于 Schur 阵的二阶对角矩阵块中。连续两步的不同 (复数) 位移可以耦合在一起, 从而在实数域中完成操作。此时, 若使用使用矩阵平方运算, 会导致每步迭代需要 $O(n^3)$ 次乘法。这可用隐式 QR 算法来优化。

 论题 4.31. 隐式 QR 算法: 类似于对称情形的隐式操作, 完成所谓的双重步位移的 QR 迭代。具体细节略。

4.6 数值实验

考虑 1.5 中的三对角对称矩阵 T_n 的特征值问题, 尝试用不同的方法进行数值计算。矩阵阶数分别取为 $n = 100$ 和 $n = 101$, 要求精确计算到小数点后第 6 位。

❖ 练习 4.1. 取初始向量为 $v_0 = (1, 1, 1, \dots, 1)^T$, 用乘幂法计算主特征值及其相应的特征向量。请绘制主特征值误差的下降曲线, 以及特征子空间距离的下降曲线。然后, 请采用 Atiken 加速技巧和 Rayleigh 商技术分别对算法进行加速, 并完成类似的工作。

❖ 练习 4.2. 用反幂法求解最靠近 2 的特征值, 及其对应的特征向量。观察是否有所谓的“一次迭代”特性。

❖ 练习 4.3. 用幂法求解第二主特征值及其特征向量; 用同时迭代方法求解 T_n 的前两个主特征值。比较两者的计算效果。

❖ 练习 4.4. 分别用古典 Jacobi 方法、循环 Jacobi 和阈值 Jacobi 方法求解 T_n 的全部特征值; 绘制相应的收敛过程。

❖ 练习 4.5. 用二分法加原点位移反幂法求解在 $(1, 2)$ 间的特征值;

❖ 练习 4.6. 用对称隐式 QR 方法求解全部特征值。

❖ 练习 4.7. 阈值 *Jacobi* 方法具有求解小特征值的优势。考虑对称正定矩阵

$$\mathbf{A} = \begin{bmatrix} 10^{40} & 10^{29} & 10^{19} \\ 10^{29} & 10^{20} & 10^9 \\ 10^{19} & 10^9 & 1 \end{bmatrix}$$

直接计算可知其特征值为 $10^{40}, 9.9 \times 10^{19}, 9.81818 \times 10^{-1}$ 。请用阈值 *Jacobi* 方法求解三个特征值, 并与 *eig()* 计算结果进行比较。

第五章 非线性方程（组）的数值方法

5.1 基本概念

如同线性方程组的迭代法研究，非线性方程组 $f(\mathbf{x}) = \mathbf{0}$ 的迭代法研究也有类似的概念。设迭代序列 $\{\mathbf{x}_k\}$ 由 r 阶迭代公式 $\mathbf{x}_k = \mathbf{g}(\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \dots, \mathbf{x}_{k-r})$ 给出，我们要重新明确如下概念：

1. 迭代序列是确定的：即迭代公式保持所有的运算有效。
2. 收敛情况的刻画：非线性问题的收敛性很复杂，它包括所谓的全局收敛（或大范围收敛）以及局部收敛。线性方程组的迭代法中仅有收敛或不收敛两种状态，因为收敛必是全局收敛。传统的局部分析通常假定了问题真解的信息。若收敛性分析不依赖问题真解的信息，这种分析称为半局部收敛分析。
3. 收敛速度的刻画：记 $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}_*$ 为第 k 步迭代误差，其中 \mathbf{x}_* 是方程的某个真解。若存在非负常数 p 和 C ，使得

$$\|\mathbf{e}_k\| \leq C\|\mathbf{e}_k\|^p, \quad \forall k \geq k_0.$$

这里的 $\|\cdot\|$ 是向量范数。对单个标量方程而言，所谓的范数就是绝对值；通常绝对值符号被去掉。从实用的角度出发，通常要求 $p \geq 1$ 。为分析简便，我们更多地采用极限形式来定义。

- (a) 若 $C \neq 0$ ，则称算法是 p 阶收敛的。若 $C = 0$ ，则称算法是至少 p 阶收敛的。
 - (b) 当 $p = 1$ 时，界定常数还需要有额外限制。若 $0 < C < 1$ ，则称算法是线性收敛的；若 $C = 0$ ，则称算法是超线性收敛的。
4. 算法效率的刻画：算法的实用收敛效率（即收敛到同样精度所需的计算时间）是更重要的问题。特别是求解非线性方程组时，算法的效率成为应用的瓶颈。设 W 是每步迭代的计算复杂度（乘除法次数，或 CPU 时间），效率通常定义为

$$\eta = \frac{\ln p}{W}, \text{ 若 } p > 1; \quad \eta = \frac{\ln C}{W}, \text{ 若 } p = 1.$$

这表明计算效率不仅仅来源与收敛阶，它与计算复杂度也有关系。

5. 数值稳定性。

★ 说明 5.1. 停机标准的设置于线性方程组的迭代方法中的设置是类似的。最主要的度量方式是残量和相邻解差距。

5.2 标量方程的数值求解

我们从标量方程出发，讨论非线性方程求解的基本技术。这是本章的重点内容。在 Matlab 中，我们可用 `fzero()` 计算初值附近的一个根。

5.2.1 区间分半法

🔗 论题 5.1. 它是闭区间上连续函数介值定理的应用，仅适用于标量方程。这个算法非常简单，但给出的近似程度不是很高。

5.2.2 不动点迭代

更常用的迭代算法是基于与 $f(x) = 0$ 同解的不动点方程 $x = g(x)$ 建立的，其中 $g(x)$ 也称为迭代函数。相应的一阶不动点迭代算法可表述为 $x_{k+1} = g(x_k)$ 。不动点迭代也称为 Picard 迭代。

不是所有的不动点迭代都是收敛的。

定理 5.1. 【压缩映像】若定义在 $[a, b]$ 上的迭代函数 $g(x)$ 满足

1. $g(x) \in [a, b], \forall x \in [a, b]$;
2. 存在 Lip 常数 $L < 1$, 使得 $|g(x) - g(y)| \leq L|x - y|, \forall x, y \in [a, b]$;

任取初值 $x_0 \in [a, b]$, 不动点迭代序列均收敛到问题的真解 x_* , 且满足线性收敛误差估计

$$|e_k| \leq \frac{L^k}{1-L} |x_1 - x_0|.$$

定理 5.2. 若迭代函数 $g(x)$ 在 $[a, b]$ 上连续可微, 且满足

$$g^{(j)}(x_*) = 0, \quad j = 0, 1, \dots, m-1; \quad g^{(m)}(x_*) \neq 0,$$

则称不动点迭代是局部收敛的, 且是 m 阶收敛。

5.2.3 加速迭代收敛

🔗 论题 5.2. 若不动点迭代序列 $\{x_k\}_{k=0}^\infty$ 线性收敛到某个真解 x_* , 我们可采用 Aitken 加速技术得到新序列 $\{\tilde{x}_k\}_{k=1}^\infty$ 。若 x_k 不等于真解 x_* , 则有

$$\lim_{k \rightarrow \infty} \frac{x_{k+1} - x_*}{x_k - x_*} = C < 1 \quad \Rightarrow \quad \lim_{k \rightarrow \infty} \frac{\tilde{x}_{k+1} - x_*}{x_k - x_*} = 0.$$

最后的结论表明收敛速度得到质的提升。

★ 说明 5.2. 上述结论中的条件 $C < 1$ 是非常重要的。若 $C = 1$ 时, Aitken 方法可能没有明显的加速效率。反例可考虑序列 $x_k = 1/(ak)$ 的 Aitken 加速。尽管如此, 在通常情况下的 Aitken 方法对真解的近似程度仍会有一定的改善; 见教科书的例子。

🔗 论题 5.3. Aitken 加速的局部应用可形成 Steffensen 迭代法, 其迭代函数为

$$\psi(x) = x - \frac{[g(x) - x]^2}{g(g(x)) - 2g(x) + x}.$$

该算法的几何解释是对不动点迭代的残量 $(x_k, g(x_k) - x_k)^\top$ 进行线性外推。可以证明: 在适当的条件下, Steffensen 方法具有局部平方收敛。

5.2.4 Newton 方法

Newton-Raphson 方法是解非线性问题最著名和最有效的算法之一。Newton 最早讨论这一方法的时间据考证是 1669 年，他使用的一个例子是 $x^3 - 2x - 5 = 0$ 。1690 年 Raphson 以略有修改的方式发表了这一算法。

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Newton 迭代的收敛速度与真解信息有关。下面假设 $f(x)$ 是光滑函数。

定理 5.3. 若 x_* 是 $f(x) = 0$ 的单根，则 Newton 迭代是局部二阶收敛的。

定理 5.4. 若 x_* 是 $f(x) = 0$ 的 m 重根，则 Newton 迭代退化为线性收敛，且误差渐进下降速度为 $1 - m^{-1}$ 。

🔍 论题 5.4. 重根的出现会导致严重的舍入误差问题。此时，Newton 迭代法精度阶的提升策略：

1. 若重数 m 是已知的，可简单修正算法为 $x_{k+1} = x_k - \frac{mf(x_k)}{f'(x_k)}$ 。
2. 若重数 m 是未知的，我们可以考虑新的函数过滤掉根的重数，即考虑新问题 $F(x) = f(x)/f'(x)$ 的 Newton 迭代；

$$x_{k+1} = x_k - \frac{f(x_k)f'(x_k)}{[f'(x_k)]^2 - f(x_k)f''(x_k)}.$$

为避免二阶导数的计算，我们还可采用 Steffensen 加速算法。

3. 重根 m 的自动探测：利用标准 Newton 迭代，计算

$$h(x_k) = \frac{\ln |f(x_k)|}{\ln |f(x_k)| - \ln |f'(x_k)|}.$$

当这个值稳定时，跳转到算法 1。

🔍 论题 5.5. Newton 方法又称切线法，其几何含义就是用切线方程局部线性化非线性方程。局部线性化是一种常用的想法。

🔍 论题 5.6. 在适当的条件下，Newton 迭代是整体收敛的。如函数 $f(x)$ 在 $[a, b]$ 上单调保凸，在端点处函数取值异号，且从端点出发的迭代序列确定的。见教科书。

🌸 思考 5.1. 重要的应用：开根号运算。

5.2.5 割线法

利用历史数据进行 $f(x)$ 的局部线性插值逼近，然后对这个线性方程求根作为新的迭代值。这也可解释为利用历史数据做差商逼近 Newton 迭代中的导数。我们可得

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}.$$

一般来说，割线法的收敛速度稍慢于 Newton 法。在适当条件下，其收敛阶为 1.618。

★ 说明 5.3. 割线法中可能出现分母为零的情形，从而造成意外的停机。这时，我们需要迭代重启，即给出新的初值重新迭代。

5.2.6 实多项式的实根计算

👉 论题 5.7. 若 n 次多项式 $p(x)$ 的系数有微小扰动，即 $p_\varepsilon(x) = p(x) + \varepsilon q(x)$ ，其中 $q(x)$ 是一个次数不超过 n 的多项式。相应的根扰动可表示为

$$x_k(\varepsilon) = x_k - \frac{q(x_k)}{p'(x_k)}\varepsilon,$$

其中 x_k 是原多项式的根。这里的比值大小反映了多项式是否病态。通常，高次多项式是病态的。

我们可以用前面的各种算法求解多项式的根，但最好的方法是 Newton 方法。

👉 论题 5.8. 多项式取值及其导数的计算：Horner 算法（或秦九韶算法）。

$$f(x) = (x - \mu)g(x) + b_0 = (x - \mu) \sum_{i=1}^n b_i x^{i-1} + b_0.$$

显然 $f(\mu) = b_0$ ，而 $f'(\mu) = g(\mu)$ 。其中 $\{b_i\}_{i=0}^n$ 的计算公式为

$$b_n = a_n, \quad b_j = a_j + b_{j+1}\mu, \quad \text{for } j = n-1 : 0.$$

👉 论题 5.9. 简介抛物线法或 Muller 方法。其基本思想与弦截法（即割线法）类似。

★ 说明 5.4. 设 $f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$ 是一个实系数多项式。关于实根的取值范围或实根数目的估计，常用的结果有：

1. Lagrange 法：设 $a_n > 0$ ， a_{n-k} 为第一个负系数。再设 b 是负系数中的最大绝对值，则 $f(x)$ 的正根上限为 $1 + (b/a_n)^{1/k}$ 。
2. Sturm 序列法：称由 $f(x)$ 与 $f'(x)$ 辗转相除得到的序列为 Sturm 序列。记 $\sigma(\mu)$ 为这个 Sturm 序列在 μ 点的符号变化的次数（删除掉零值），它表示严格大于 μ 的实根数目。
3. Descartes 符号律：一元实系数多项式按降幂次方式排列，则多项式的正根数目等于相邻非零系数的符号改变个数减去一个非负偶数。

具体讨论超出本课程的要求，详略。

★ 说明 5.5. 多项式的根还可转化为矩阵的特征值问题。还有很多专门针对求多项式全部零点的特殊算法，本课程不做过多讨论。

5.3 方程组的数值求解

本节简要介绍非线性方程组 $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ 的数值求解方法。此时，我们将面临如下的困难：(a) 精确解的数学理论（存在性唯一性）不清楚；(b) 有些算法的原理不再成立，如二分法；(c) 即使适用于标量方程的算法能推广到方程组，推广的方式也是重要研究内容，因为 (d) 计算效率成为算法的瓶颈。

5.3.1 预备知识

这部分工作涉及到向量值函数的微积分，它们可简单理解为多元函数微积分（连续，导数，积分）的推广。这里，我们不纠缠于这些内容的细致讨论，而仅仅列出与后续讨论有紧密联系的一些结论。

📌 定义 5.1. 设向量值函数 $\mathbf{f}(\mathbf{x}) = \{f_i(\mathbf{x})\}_{i=1}^m$ 是 $\mathbb{R}^n \rightarrow \mathbb{R}^m$ 的映像，其中每个分量的偏导数均连续。其 Frechet 导数是 $m \times n$ 阶 Jacobi 矩阵

$$\mathbf{f}'(\mathbf{x}) = D\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} (\mathbf{x}). \quad (5.3.1)$$

若 $f(\mathbf{x})$ 是标量多元函数，有 $f'(\mathbf{x}) = [\nabla f(\mathbf{x})]^\top$ 。事实上，Frechet 导数可精确定义如下

$$\lim_{\|\Delta \mathbf{x}\| \rightarrow 0} \frac{\|\mathbf{f}(\mathbf{x} + \Delta \mathbf{x}) - \mathbf{f}(\mathbf{x}) - \mathbf{f}'(\mathbf{x})\Delta \mathbf{x}\|}{\|\Delta \mathbf{x}\|} = 0. \quad (5.3.2)$$

★ 说明 5.6. 向量值函数与多元函数的主要区别之处是微分中值定理。设 \mathbf{x} 和 \mathbf{y} 是给定的两个位置，即使函数充分光滑，也不存在一个局部位置 $\boldsymbol{\xi}$ ，使得

$$\mathbf{f}(\mathbf{y}) - \mathbf{f}(\mathbf{x}) = \mathbf{f}'(\boldsymbol{\xi}) \cdot (\mathbf{y} - \mathbf{x}).$$

此时，每个分量 f_i 的对应中值通常是不同的。但是，它们造成的实际困难却不大，因为有积分表示式

$$\mathbf{f}(\mathbf{y}) - \mathbf{f}(\mathbf{x}) = \int_0^1 \mathbf{f}'(\mathbf{x} + s(\mathbf{y} - \mathbf{x})) ds \cdot (\mathbf{y} - \mathbf{x}).$$

这个积分称为 Jacobi 矩阵 \mathbf{f}' 从 \mathbf{x} 到 \mathbf{y} 的 Riemann 积分，它对应于每个分量函数的积分。

📌 论题 5.10. 在后续算法研究中，我们会使用向量范数与矩阵范数作为度量方式。常用的不等式主要有

1. 积分不等式: $\|\int_0^1 \mathbf{f}(t) dt\| \leq \int_0^1 \|\mathbf{f}(t)\| dt$.
2. 设 \mathbf{f} 在凸集上处处 Frechet 可微，且导数 \mathbf{f}' 是 γ -Lip 连续，则有

$$\|\mathbf{f}(\mathbf{y}) - \mathbf{f}(\mathbf{x}) - \mathbf{f}'(\mathbf{x})(\mathbf{y} - \mathbf{x})\| \leq \frac{\gamma}{2} \|\mathbf{y} - \mathbf{x}\|^2.$$

此时，压缩映像定理依旧是重要的分析工具。

5.3.2 Newton 法

不动点迭代的概念与分析方法可以很容易地从标量方程推广到方程组。此处不再赘述。下面，我们仅关注著名的 Newton 方法。

$$\mathbf{f}'(\mathbf{x}_k)\Delta \mathbf{x}_k = -\mathbf{f}(\mathbf{x}_k), \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}_k.$$

它的理论分析有着悠久的历史。在 1829 年, Cauchy 研究了 $n = 1$ 时的局部收敛性, 直到 1899 年才由 Runge 推广到方程组。在 1916 年, Fine 讨论了半局部收敛分析。而后著名的工作还有 Ostrowski(1936), Willers(1938) 和 Kantovich(1948)。

定理 5.5. 局部收敛分析: 若 $\mathbf{f}'(\mathbf{x})$ 在真解 \mathbf{x}_* 附近是连续的非奇异矩阵, 则 Newton 方法是局部超线性收敛。若进一步假设 \mathbf{f}' 在真解附近是 Lipschitz 连续的, 则 Newton 方法至少是局部二阶收敛的。

★ 说明 5.7. 设序列 $\{\mathbf{x}_k\}$ 超线性收敛到 \mathbf{x}_* , 则其必满足 $\lim_{k \rightarrow \infty} \frac{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|}{\|\mathbf{x}_k - \mathbf{x}_*\|} = 1$ 。因此说, 我们可以用相邻误差作为迭代误差的估计。

请注意, 上述逆命题不成立, 如奇偶子列分别定义为 $\mathbf{x}_{2k+1} = \frac{2}{(2k)!}$ 和 $\mathbf{x}_{2k} = \frac{1}{(2k)!}$ 。

★ 说明 5.8. Newton 方法是自校正的, 即 \mathbf{x}_{k+1} 仅依赖于 \mathbf{x}_k 。

定理 5.6. 半收敛分析: 设 $\mathbf{f}(\mathbf{x})$ 在 D_0 上处处 Frechet 可微, 且具有如下性质:

1. $\|\mathbf{f}'(\mathbf{x}) - \mathbf{f}'(\mathbf{y})\| \leq \gamma \|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in D_0$;
2. $[\mathbf{f}'(\mathbf{x})]^{-1}$ 存在, 且 $\|[\mathbf{f}'(\mathbf{x})]^{-1}\| \leq \beta, \forall \mathbf{x} \in D_0$;
3. $\|[\mathbf{f}'(\mathbf{x}_0)]^{-1} \mathbf{f}(\mathbf{x}_0)\| \leq \alpha$;

其中 D_0 是一个开凸集。若 $\mathbf{x}_0 \in D_0$ 且 $h = \alpha\beta\gamma/2 < 1$, 则 Newton 迭代序列至少二阶收敛到问题的某个真解 $\mathbf{x}_* \in S_r(\mathbf{x}_0)$, 其中 $\gamma = \alpha/(1-h)$ 。

★ 说明 5.9. 由半收敛分析的证明过程可知, 收敛条件成立的一个必要条件是

$$\|\mathbf{x}_2 - \mathbf{x}_1\| < \|\mathbf{x}_1 - \mathbf{x}_0\|. \quad (5.3.3)$$

这通常会保证 $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|$ 的下降。因此, 在初始两步的计算中若发现该条件不成立, 则我们自动放弃这个没有前途的初值。但是, 初始解向量 \mathbf{x}_0 的选取是问题的难点。

★ 说明 5.10. 利用 Newton 位移的逐次缩减实现所谓的“盲人下山法”, 可以找到某个真解的大概位置。然后, 再利用 Newton 迭代给出快速的二次收敛。当迭代矩阵接近奇异时, 我们可采用基于 Tikhonov 正则化方法的修正 Newton 算法:

$$[\mathbf{f}'(\mathbf{x}_k) + \lambda_k \mathbb{I}] \Delta \mathbf{x}_k = -\mathbf{f}(\mathbf{x}_k), \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}_k.$$

其中 $\lambda_k > 0$ 也称为阻尼因子。

5.3.3 Newton 法的改进

Newton 方法虽具有所谓的局部二阶收敛, 但是其有两个缺点。其一是 Jacobi 矩阵的计算消耗大量的 CPU 时间, 每步需要计算 $n^2 + n$ 个函数值和导数值。其二是线性方程组的求解很耗时, 成为算法效率的主要障碍。下面我们对此做些改进。

修正 Newton 法

这也称为沙文基思方法, 它局部锁定 Jacobi 矩阵, 减少了线性方程组求解的困难。

$$\mathbf{x}_{k,0} = \mathbf{x}_k, \quad \mathbf{x}_{k,j} = \mathbf{x}_{k,j-1} - [\mathbf{f}'(\mathbf{x}_k)]^{-1} \mathbf{f}(\mathbf{x}_{k,j-1}), \quad \mathbf{x}_{k+1} = \mathbf{x}_{k,m}.$$

在一定的假设条件下, 该算法可达 $m+1$ 阶收敛速度。

割线法

割线法改进了 Jacobi 矩阵的计算困难。它用适当的近似代替了导数的精确计算。据不同的解释，它主要有两大类方法：

👉 论题 5.11. 离散的 Newton 方法：直接用差商矩阵 $\mathbb{J}(\mathbf{x}_k, \mathbf{h}_k)$ 代替 Newton 迭代矩阵。

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left[\mathbb{J}(\mathbf{x}_k, \mathbf{h}_k) \right]^{-1} \mathbf{f}(\mathbf{x}_k).$$

其中 $(\mathbf{h}_k)_{ij} = \bar{h}_{ij} \|\mathbf{f}(\mathbf{x}_k)\|$ 为趋于零的离散化参数。这里的 \bar{h}_{ij} 是事先给定的常数，通常为简便，令每行的元素相等。

👉 论题 5.12. 利用非线性方程的局部线性插值（或曲面的局部平面化）思想，将线性方程组的交点作为好的近似位置。我们可得如下算法：

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{A}_k^{-1} \mathbf{f}(\mathbf{x}_k), \quad \mathbf{A}_k^{-1} = \mathbf{H}_k \mathbf{\Gamma}_k^{-1}.$$

其中 $\mathbf{H}_k = [\mathbf{x}_{k,1} - \mathbf{x}_{k,0}, \dots, \mathbf{x}_{k,n} - \mathbf{x}_{k,0}]$ 和 $\mathbf{\Gamma}_k = [\mathbf{f}_{k,1} - \mathbf{f}_{k,0}, \dots, \mathbf{f}_{k,n} - \mathbf{f}_{k,0}]$ 是由 $n+1$ 个辅助点 $\{\mathbf{x}_{k,\ell}, \mathbf{f}_{k,\ell}\}_{\ell=0}^n$ 生成的矩阵。通常要求辅助点信息处于一般位置，即 \mathbf{H}_k 和 $\mathbf{\Gamma}_k$ 均可逆。

设 $\mathbf{x}_{k,0} = \mathbf{x}_k$ 。依据辅助点的选取策略，我们有两个著名的方法。

1. 两点序列割线法； $\mathbf{x}_{k,\ell} = \mathbf{x}_k + \{(\mathbf{x}_{k-1})_\ell - (\mathbf{x}_k)_\ell\} \mathbf{e}_j$.
2. $(n+1)$ 点序列割线法： $\mathbf{x}_{k,\ell} = \mathbf{x}_{k-\ell}$. 这里的 $\mathbf{\Gamma}_k^{-1}$ 具有便于计算的递推关系。

在每次迭代中，前者需计算 n^2 个函数取值，而后者仅需 n 次函数取值。

★ 说明 5.11. 无论哪种方法，可证 p 点序列割线法的收敛阶为 $\lambda^p = \lambda^{p-1} + 1$ 的最大正根。虽然 $(n+1)$ 点序列割线法收敛阶占优，但其容易产生数值不稳定。

拟 Newton 法

拟 Newton 法是六十年代发展起来的解非线性方程组的有效方法。它继承了 $(n+1)$ 点序列割线法的优点，克服了 Newton 法需要求导和求逆等缺点。回忆 $(n+1)$ 点序列割线法的性质

$$\begin{aligned} \mathbf{A}_k \mathbf{p}_j &= \mathbf{q}_j, \quad j = k-1, k-2, \dots, k-n; \\ \mathbf{A}_{k+1} \mathbf{p}_j &= \mathbf{q}_j, \quad j = k, k-1, \dots, k-n+1, \end{aligned}$$

其中 $\mathbf{p}_j = \mathbf{x}_{j+1} - \mathbf{x}_j$, $\mathbf{q}_j = \mathbf{f}_{j+1} - \mathbf{f}_j$. 这些蕴含着

$$(\mathbf{A}_{k+1} - \mathbf{A}_k) \underbrace{[\mathbf{p}_k, \mathbf{p}_{k-1}, \dots, \mathbf{p}_{k-n+1}]}_{\mathbf{H}_k} = [\mathbf{f}_{k+1}, 0, 0, \dots, 0]. \quad (5.3.4)$$

若矩阵 \mathbf{H}_k 可逆，记其逆矩阵的第一行向量为 \mathbf{r}_k^\top ，从而有 $\mathbf{A}_{k+1} - \mathbf{A}_k = \mathbf{f}_{k+1} \mathbf{r}_k^\top$. 利用 Sherman-Morrison 公式， \mathbf{A}_{k+1}^{-1} 可以由 \mathbf{A}_k^{-1} 的秩一修正得到。这种动态联系（或逐步修正）很好地解决了线性方程组的求解效率问题。

拟 Newton 法的基本结构为：任取初始向量 \mathbf{x}_0 和初始迭代矩阵 \mathbb{B}_0 ，然后计算新的位置

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbb{B}_k^{-1} \mathbf{f}(\mathbf{x}_k),$$

并利用当前信息构造下一步的迭代矩阵 \mathbb{B}_{k+1} 。我们希望其继承了 $(n+1)$ 点序列割线法中迭代矩阵的特性之一，即所谓的拟 Newton 方程

$$\mathbf{y}_k = \mathbb{B}_{k+1} \Delta \mathbf{x}_k, \quad \text{其中 } \mathbf{y}_k = \mathbf{f}_{k+1} - \mathbf{f}_k, \Delta \mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k.$$


故而，拟 Newton 方法可视为割线法的推广。

拟 Newton 算法的核心是如何建立迭代矩阵 \mathbb{B}_k 或者其逆矩阵 $\mathbb{H}_k = \mathbb{B}_k^{-1}$ 的发展策略，并保证算法具有所谓的超线性收敛。在拟 Newton 法中，一个重要的思想是逐次修正迭代矩阵。观察 $(n+1)$ 点序列割线法，不难发现其迭代矩阵满足： $\mathbb{A}_{k+1} - \mathbb{A}_k$ 是秩一的。本节，我们重点讨论基于秩一修正的 Broyden(1965) 方法。

定理 5.7. 给定方向 \mathbf{v}_k ，考虑秩一修正的 $\mathbb{B}_{k+1} = \mathbb{B}_k + \mathbf{u}_k \mathbf{v}_k^\top$ ，其中 \mathbf{u}_k 是任意的方向。显然 \mathbb{B}_{k+1} 与 \mathbb{B}_k 作用在 $\text{span}^\perp(\mathbf{v}_k)$ 的像是一样的。若满足拟 Newton 方程，则应取

$$\mathbf{u}_k = (\mathbf{y}_k - \mathbb{B}_k \Delta \mathbf{x}_k) / \mathbf{v}_k^\top \Delta \mathbf{x}_k.$$

此时， \mathbb{B}_{k+1} 是极小问题 $\min\{\|\mathbb{S} - \mathbb{B}_k\|_F : \mathbb{S} \Delta \mathbf{x}_k = \mathbf{y}_k\}$ 的解。

 **论题 5.13.** 在 Broyden 方法中，迭代矩阵的逐步修正策略是

$$\mathbb{H}_{k+1} = \mathbb{H}_k + \frac{(\Delta \mathbf{x}_k - \mathbb{H}_k \mathbf{y}_k) \mathbf{d}_k^\top}{\mathbf{d}_k^\top \mathbf{y}_k} = \mathbb{H}_k - \frac{\mathbb{H}_k \mathbf{f}_{k+1} \mathbf{d}_k^\top}{\mathbf{d}_k^\top \mathbf{y}_k},$$

其中 $\mathbf{d}_k = \mathbb{H}_k^\top \mathbf{v}_k$ 。它是 Sherman-Morrison 公式的应用，所需的计算量为 $O(n^2)$ 。这里的 \mathbf{v}_k 的选取主要有两种方式，即 $\mathbf{v}_k = \Delta \mathbf{x}_k$ 或 $\mathbf{v}_k = \mathbf{f}(\mathbf{x}_{k+1})$ 。后者更适宜所谓的对称问题求解，可一直保持 \mathbb{H}_k 的对称性。

★ **说明 5.12.** 可以证明：在适当的条件下，Broyden 方法具有局部的超线性收敛。


★ **说明 5.13.** 若算法是收敛的，其具有超线性收敛只需一个较弱的充要条件

$$\lim_{k \rightarrow \infty} \frac{\|[\mathbb{B}_k - \mathbf{f}'(\mathbf{x}_*)] \Delta \mathbf{x}_k\|}{\|\Delta \mathbf{x}_k\|} = 0.$$

在适当的条件下，上述条件还保障

$$\mathbf{s}_k^{\text{Qn}} - \mathbf{s}_k^{\text{Nt}} = \Delta \mathbf{x}_k + [\mathbf{f}'(\mathbf{x}_k)]^{-1} \mathbf{f}'(\mathbf{x}_k) = [\mathbf{f}'(\mathbf{x}_k)]^{-1} \{\mathbf{f}'(\mathbf{x}_k) - \mathbb{B}_k\} \Delta \mathbf{x}_k \rightarrow 0.$$

换言之，要得到所谓的超线性收敛算法，只要每步的迭代修正方向 \mathbf{s}_k^{Qn} 在大小和方向上都渐进地逼近 Newton 修正方向 \mathbf{s}_k^{Nt} ，而不必要求迭代矩阵 \mathbb{B}_k 趋向于真解 \mathbf{x}_* 处的 Jacobi 矩阵 $\mathbf{f}'(\mathbf{x}_*)$ 。

 **思考 5.2.** 考虑 $\mathbf{f}(\mathbf{x}) = (x_1, x_2^2 + x_2)^\top$ 的求根问题，其真解为 $\mathbf{x}_* = (0, 0)^\top$ 。若初始猜测值和初始迭代矩阵分别取为

$$\mathbf{x}_0 = (0, \varepsilon)^\top, \quad \mathbb{B}_0 = \begin{bmatrix} 1 + \delta & 0 \\ 0 & 1 \end{bmatrix}.$$

观测 \mathbb{B} 的 $(1, 1)$ 位置处的元素不收敛到 $\mathbf{f}'(\mathbf{x}_*)$ 的对应元素。

5.3.4 极值算法

求解非线性方程组 $\mathbf{f}(\mathbf{x}) = 0$ 等价于求解最小优化问题 $\min_{\mathbf{x}} \|\mathbf{f}(\mathbf{x})\|_2^2$ 。各种优化方法的具体应用略。

5.4 数值实验

❖ 练习 5.1. 用 Newton 方法求解方程 $x^3 - x^2 - 8x + 12 = 0$ 的根, 并绘制误差下降曲线。然后, 试用割线法重复上述工作。

❖ 练习 5.2. 编制 Newton 方法和 Broyden 方法求解非线性方程组

$$\begin{cases} (x+3)(y^2-7)+18=0, \\ \sin(ye^x-1)=0, \end{cases} \quad (5.4.5)$$

取相同的初值 $(-0.15, 1.4)$, 比较两者的迭代次数; 若初值为 $(0, 1)$ 呢? 观察 Broyden 中的 \mathbb{B}_k 是否收敛到 Jacobi 矩阵呢? 若非线性方程组为

$$\begin{cases} x+y-3=0, \\ x^2+y^2-9=0, \end{cases} \quad (5.4.6)$$

呢? 初值取为 $(2, 4)$, 观察 Broyden 中的 \mathbb{B}_k 是否收敛到 Jacobi 矩阵呢?

❖ 练习 5.3. 编制修正 Newton 法 (与 m 的关系)、离散 Newton 法和两点序列割线法求解(5.4.6)。

❖ 练习 5.4. 非线性方程组的应用: Newton 法可应用于特征值问题的求解, 方式如下

$$\begin{cases} \mathbb{T}\mathbf{x} - \lambda\mathbf{x} = 0, \\ \mathbf{x}^\top \mathbf{x} = 1. \end{cases} \quad (5.4.7)$$

取 \mathbb{T} 为以前的三对角矩阵, 阶数为 $n = 5$; 取任意的初值 \mathbf{x}_0 和 $\lambda_0 = \mathbf{x}_0^\top \mathbb{T} \mathbf{x}_0$, 做 Newton 迭代。将得到的结果与幂法做比较。

qzh@aliyun.com