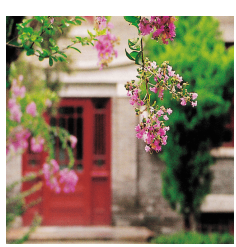
第 11 版（2023 年 8 月 21 日）

《数值代数》讲义

整理：张强
南京大学数学系

前　言

本讲义是信息与计算专业必修课程《数值代数》的配套材料[1]，介绍四类问题的常用数值方法。

1. 系数矩阵非奇异线性方程组（对应教科书第 3 章和第 6 章）：

   - 直接法没有方法误差，可以利用有限次运算求出精确解。但是，它具有舍入误差，在计算机上的运行结果却可能出现偏差。本讲义介绍 Gauss 消元法及其等价变形，重点展示舍入误差导致的数值困扰。
   - 迭代法具有方法误差，通过自动生成的向量序列逼近精确解。本讲义将介绍经典的 Jacobi 方法和 Gauss-Seidel 方法、著名的超松弛方法、半迭代方法和共轭斜量方法，展示迭代法的常见设计思路和具体分析技术。

2. 线性最小二乘问题（对应教科书的第 7 章）：

   即使系数矩阵是列满秩的，最小二乘问题的数值求解也是非常困难的，特别是舍入误差的有效控制。除正规化技术之外，本讲义重点讨论系数矩阵直交化技术及其应用。

3. 矩阵特征值问题（对应教科书的第 8 章）：

   它兼具线性问题和非线性问题的属性，相应的数值求解极具困难。本讲义介绍幂法、Jacobi 方法和 QR 方法。

[1] 2022 年以前的课程名称是《数值计算与试验 II》，教科书是文献 [6]。

4. 非线性方程求根问题（对应教科书的第 2 章和第 9 章）：

   本讲义介绍不动点迭代的基本理论，特别关注 Newton 方法及其各种改进。

本讲义的教学目标是上述算法的实现过程、设计精髓及其蕴含的数值技术。

目 录

第 1 章
线性方程组的直接法

在科学工程计算中，许多问题（例如数据拟合、偏微分方程数值解，数值优化问题）都会导致或转化为线性方程组

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{x} \in \mathbb{R}^n, \tag{1.0.1}$$

其中 $\mathbb{A}$ 是非奇异方阵。随着计算规模的增大，真解通常无法手工计算出来，需借助先进的计算工具（数字计算机）机械化地自动实现。相应的算法主要有两类，其一是本章的直接法，其二是下一章的迭代法。事实上，直接法是一种精确算法，在不计舍入误差的情况下，可以通过有限次四则运算（可能包括开方运算）精确算出问题的真解。本章重点介绍 Gauss 消元法及其（理论上）等价的其它方法，展示数据操作、程序设计、计算复杂度、舍入误差等层面的差异。

1.1 Gauss 消元法

对于中小型规模的线性方程组，特别是当系数矩阵元素几乎处处非零（称为稠密）且取值毫无规律的时候，Gauss 消元法是使用最为广泛的数值算法。相应的 Matlab 命令是 $\mathbb{A}\backslash\boldsymbol{b}$，其中 $\mathbb{A}$ 是系数矩阵，$\boldsymbol{b}$ 是右端向量。

1.1.1 编程实现

在秦汉（约公元前 150 年）时代，《九章算术》就已经出现了消元思想。在 17 世纪后叶，Leibnitz 也提出了类似的消元技术；但是直至 19

世纪初，Gauss 消元法[1]才正式出现在欧美文献中。

顺序 Gauss 消元法

《高等代数》讲过 Gauss 消元法的实现过程：逐步缩减待解未知量的个数，将原始问题转化到同解的上三角问题。用矩阵语言描述，就是：利用初等行变换（即初等行变换阵左乘），将增广矩阵 $[\mathbb{A}\,|\,\boldsymbol{b}]$ 转化为上梯形阵。

相比于理论算法，数值算法还需考虑下面三个问题：（1）节省数据存储空间，实现现有条件下的算法可行性；（2）降低数据读取时耗和四则运算总量，提高计算效率；（3）控制舍入误差的积累和放大，保证数值结果的可靠性。后续内容将给予适当的解释。

✎ **论题 1.1.** *顺序 Gauss 消元法是基于第三种初等行变换的最简单处理过程*[2]。

图文框给出顺序 Gauss 消元法的伪代码片段，其中消元结果和原始数据共享在一个二维数组中，“等号” 表示计算机语言中的 “赋值” 操作，暗含实现了**数据覆盖**技术。具体而言，首先二维数组存储系数矩阵 $\mathbb{A}$，然后对角线下方明确清零的废弃位置存储相应的消元乘子（第 3 行代码），右下方的位置逐渐更新为消元结果（第 5–9 行代码）。若二维数组存储增广矩阵 $[\mathbb{A}\,|\,\boldsymbol{b}]$，只需将第 5 行代码

```
1.  For k = 1, 2, ..., n, Do
2.    For i = k + 1, ..., n, Do
3.      a_ik = a_ik / a_kk;
4.    Enddo
5.    For j = k + 1, ..., n, Do
6.      For i = k + 1, ..., n, Do
7.        a_ij = a_ij − a_ik a_kj;
8.      Enddo
9.    Enddo
10. Enddo
```

[1] 1809 年，发表于 Theoria Motus

[2] 数学描述的符号体系：第 k 步操作的矩阵记为 $\mathbb{A}_k$ 或 $\mathbb{A}^{(k)}$，相应的矩阵元素记为 $a_{ij}^{(k)}$。通常，当 k 取值最小（有时是 0 有时是 1）时就是原始矩阵。

的循环上界由 n 改为 $n+1$。这段简单代码充分展现了编程设计的要素之一，即**数据存储空间的合理利用**。

定义 1.1. 计算复杂度通常指整体计算所消耗的 *CPU* 时间，是评价算法效率的重要指标。同乘除相比，加减的耗时可以忽略；本讲义采用乘除次数刻画计算复杂度，不统计加减次数。

将系数矩阵变换为上三角阵，相应的乘除次数是

$$\sum_{k=1}^{n-1}(n-k)(n-k+1)=\mathcal{O}(n^3/3).$$

关于右端向量的消元变换，还需增加 $\mathcal{O}(n^2/2)$ 次乘除运算。关于上三角方程组的回代求解，还需增加 $\mathcal{O}(n^2/2)$ 次乘除运算。

顺序 Gauss 消元法含有除法运算。只有当对角元满足

$$a_{kk}^{(k)}\neq 0,\quad k=1,2,\ldots,n-1, \tag{1.1.2}$$

消元过程才能执行到底。若 $a_{nn}^{(n)}\neq 0$，回代过程也可顺利执行。

定理 1.1. (1.1.2) 成立的充要条件是系数矩阵 $\mathbb{A}$ 的前 $n-1$ 阶顺序主子阵都是非奇异的。

证明：在消元操作下，各阶主子阵的行列式不变。 □

定理 1.2. 若系数矩阵 $\mathbb{A}$ 对称正定，则顺序 *Gauss* 消元法可顺利执行到底，且中间矩阵的元素绝对值不超过原矩阵元素的最大值。

说明 1.1. 当数值计算规模庞大时，数据读写速度也是影响算法效率的重要因素。通常，它们同编程语言和硬件结构有关。

1. 在前面的图文框中，伪代码采用 k–j–i 三重循环次序。它适用于 *Fortran* 语言编程，因其二维数组是按列连续存放的，数据寻址的

代价较低。然而，它不适用于 *C++* 语言，因其二维数组是按行连续存放的。若依旧使用 k–j–i 三重循环次序，则同列数据的读取大多是跳跃的，数据指针移动过于频繁，消耗过多的读取时间。为提高算法性能，我们有必要交换最内两层的循环次序。

2. 代码执行效率还同计算机硬件结构（目前还包括网络结构）有关，特别是读写加速设备（例如二级缓存等）的使用。当数据规模庞大时，数据读写（内存与寄存器的数据移动）消耗的 *CPU* 时间不容忽视。前面的伪代码仅仅处于 *BLAS-1* 代码级别[3]，浮点操作均基于数和数的四则运算，相应的数据读写效率较低。

   提高代码级别，可改善上述问题。*BLAS-2* 代码基于向量（含矩阵）与向量的操作。借用 *Matlab* 语言，*BLAS-2* 版本的伪代码是

   (a) *For* $k = 1 : n$, *Do*
   (b) $\quad \mathbb{A}(k+1:n,k) = \mathbb{A}(k+1:n,k)/\mathbb{A}(k,k)$;
   (c) $\quad \mathbb{A}(k+1:n,k+1:n) =$
   $\qquad \mathbb{A}(k+1:n,k+1:n) - \mathbb{A}(k+1:n,k) \star \mathbb{A}(k,k+1:n)$;
   (d) *Enddo*

   BLAS-3 代码是最高级别，基于矩阵与矩阵的分块操作，涉及到并行计算的相关概念。因其超出课程设置，详略。

列主元 Gauss 消元法

明确数值算法的适用范围及其优缺点，是计算数学的基本主题之一。每个算法在提出之后，都需深入考察其在计算机上的运行效果。前面的算法研究假设了计算过程精确无误，但是假设过于理想化，在位长有限

[3]BLAS=Basic Linear Algebraic Subroutine.

的数字计算机上根本无法成立，因为浮点数的数据存储和四则运算都会产生舍入误差。在多数情况下，这个现象是不可忽视的：

- 理论上执行到底的顺序 Gauss 消元法，舍入误差可能导致“除零”操作而意外停机[4]；
- 即使算法可以执行到底，舍入误差也可能影响计算结果的准确性，甚至出现严重的偏离。

事实上，计算结果的偏离程度同待解问题本身和计算机位长（或者机器精度）均有关。不妨假想在一台仅有三位有效数字的十进制虚拟机上运行顺序 Gauss 消元法，相应的增广矩阵及其消元数据变化是

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

基于上述结果进行回代求解，可得数值解 $x_{\mathrm{num}} = (0.00, 1.00)^\top$，它明显有别于问题的精确解 $x_\star = (1.002\cdots, 0.998\cdots)^\top$。随着机器精度的提升，数值准确性得到相应改善。

综上所述，精确算法在计算机上给出的数值结果也可能不可靠，甚至“错误”。换言之，控制舍入误差的积累和膨胀，保证数值结果的可靠性，是数值研究的核心问题和独特之处。

就 Gauss 消元法而言，简单且有效的解决方案是引入“消元主元”策略，如列主元/全主元（Wilkinson, 1961）和車型主元（Neal 和 Poole, 1992）策略。数值经验表明：上述三种策略的数值差异甚微。

论题 1.2. 对于中小规模的稠密方程组，列主元策略最受欢迎，因为其搜索时间最少。列主元是指位于当前对角元及其下方，具有最大绝对值的那个元素。相应的算法称为列主元 *Gauss* 消元法。其编程实现非

[4]事实上，除法运算的编程进行预判总是值得鼓励的。

常简单，只需在顺序 *Gauss* 消元法两个版本的第 *2* 行伪代码之前，添加一段补丁代码：

- 确定行号 $l = \arg\max_{k \le i \le n} |a_{ik}|$；
- 交换第 l 行和第 k 行数据；

这样的操作可以确保消元乘子的绝对值不超过一，消元过程的舍入误差得到有效的控制。

✯ **说明 1.2.** 数据移动也要消耗时间，也会影响到算法的执行效率。事实上，行交换可以不用真正移动数据，只需引进指标向量

$$\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$$

记录 *Gauss* 消元过程中的行交换信息。基本操作如下：

- 其初始值是自然序列，即 $p_i = i$；
- 若第 k 步消元进行了第 k 和第 r 行的元素交换，轮换指标向量 $\boldsymbol{p}$ 中的第 k 个和第 r 个分量，并调整对应循环中的行指标。

请重写代码，实现上述目标。

✯ **说明 1.3.** 教科书还给出了列主元选取策略，即**按比例选取列主元**，即在寻求主元之前，先将右下角矩阵的所有行向量按最大模分量进行单位化操作。此策略等价于数量矩阵左乘的预处理过程。

✯ **说明 1.4.** 顺序 *Gauss* 消元给出上三角阵，其对角元乘积就是系数矩阵 $\mathbb{A}$ 的行列式。若采用列主元策略，还要统计行交换的执行次数，乘以相应次数的 -1。

✯ **说明 1.5.** 由于舍入误差，*Gauss* 消元过程不能准确给出矩阵的秩。粗略的解释是，对角元素的非零判定无法准确实现；详细的理论解释可参见第三章。

1.1.2 Gauss 消元阵

Gauss 消元过程等同于系数（或增广）矩阵的上三角（或上梯形）化，其核心操作是**数值代数的基本问题**：

> 已知 m 维非零向量 $\boldsymbol{a} = (a_1, a_2, \ldots, a_m)^\top$。能否构造一个简单矩阵 $\mathbb{H}$，使得 $\mathbb{H}\boldsymbol{a}$ 仅首个位置非零?

除了本章给出的 Gauss 消元阵，第三章给出的 Householder 镜像变换阵和 Givens 平面旋转阵，也可以实现上述目标。

✎ **论题 1.3.** 默认 $a_1 \neq 0$；否则，进行适当的行交换即可。对应基本问题的 m 阶 *Gauss* 消元阵是单位阵 $\mathbb{I}_{m\times m}$ 的秩一修正，即

$$\mathbb{S}_{m\times m} = \mathbb{I}_{m\times m} - \boldsymbol{g}\boldsymbol{e}_1^\top, \tag{1.1.3}$$

其中 $\boldsymbol{e}_1 = (1, 0, 0, \ldots, 0)^\top$ 是首个分量为 1 的 m 维单位向量，

$$\boldsymbol{g} = (0, a_2/a_1, a_3/a_1, \ldots, a_m/a_1)^\top$$

是消元乘子 $g_j = a_j/a_1$ 构成的 m 维向量。

对应 Gauss 消元过程的第 k 步，考虑 n 维列向量

$$\boldsymbol{a} = (a_{1k}, a_{2k}, \ldots, a_{kk}, \ldots, a_{nk})^\top, \tag{1.1.4}$$

其中 $a_{kk} \neq 0$。上述概念可以拓展到 n 阶 Gauss 消元阵，实现 a_{kk} 下方的所有元素清零。

定义 1.2. 记 $\boldsymbol{e}_k$ 是仅第 k 个分量为 *1* 的 n 维单位向量。令

$$\boldsymbol{\ell}_k = (0, 0, \ldots, 0, \ell_{k+1,k}, \ell_{k+2,k}, \ldots, \ell_{n,k})^\top,$$

其中 $\ell_{ik} = a_{ik}/a_{kk}$ 是消元乘子。相应的 n 阶 *Gauss* 消元阵是

$$\mathbb{L}_k^{-1} = \mathbb{I} - \boldsymbol{\ell}_k \boldsymbol{e}_k^\top, \tag{1.1.5}$$

它也可以理解为 $n-k+1$ 阶 *Gauss* 消元阵的单位扩张[5]，即

$$\mathbb{L}_k^{-1} = \begin{bmatrix} \mathbb{I}_{(k-1)\times(k-1)} & \mathbb{O} \\ \mathbb{O} & \mathbb{S}_{(n-k+1)\times(n-k+1)} \end{bmatrix}. \tag{1.1.6}$$

论题 1.4. 注意到 *Gauss* 消元阵的定义和 $\boldsymbol{\ell}_k$ 的生成方式，顺序 *Gauss* 消元法可以矩阵描述为

$$\boxed{\mathbb{L}_{n-1}^{-1} \cdots \mathbb{L}_2^{-1} \mathbb{L}_1^{-1} \left[\mathbb{A}^{(1)} \mid \boldsymbol{b}^{(1)} \right] = \left[\mathbb{A}^{(n)} \mid \boldsymbol{b}^{(n)} \right],}$$

其中 $\mathbb{A}^{(1)} = \mathbb{A}$ 和 $\boldsymbol{b}^{(1)} = \boldsymbol{b}$，右端是消元结束时的上梯形矩阵。

简单可证，Gauss 消元阵(1.1.5)具有两个基本性质：

$$\mathbb{L}_k = \mathbb{I} + \boldsymbol{\ell}_k \boldsymbol{e}_k^\top; \qquad \mathbb{L}_i \mathbb{L}_j = \mathbb{L}_i + \mathbb{L}_j - \mathbb{I}, \quad (i < j).$$

利用论题 1.4 的结果，顺序 Gauss 消元过程衍生出矩阵的三**角分解**

$$\mathbb{A} = \mathbb{L}\mathbb{U}, \tag{1.1.7}$$

其中 $\mathbb{U} = \mathbb{A}^{(n)}$ 是上三角阵，而 $\mathbb{L}$ 是单位下三角阵。利用前面的基本性

[5]这种扩张方式将在本课程中多次使用，以后不再赘述。

质，可知

$$\mathbb{L}=\mathbb{L}_1\cdots\mathbb{L}_{n-1}=\begin{bmatrix}1&&&&\\ \ell_{21}&1&&&\\ \ell_{31}&\ell_{32}&1&&\\ \vdots&\vdots&\ddots&\ddots&\\ \ell_{n1}&\ell_{n2}&\cdots&\ell_{n,n-1}&1\end{bmatrix},$$

其中 $\ell_{ij}=a_{ij}^{(j)}/a_{jj}^{(j)}$ 是顺序 Gauss 消元法的消元乘子。换言之，$\mathbb{L}$ 不用花时间去计算，只需简单堆积消元因子即可。

✍ **论题 1.5.** 列主元选取造成的行交换可以描述为初等排列阵的左乘，从而列主元 *Gauss* 消元法可以矩阵描述为：

$$\boxed{\mathbb{U}=\mathbb{A}^{(n)}=\mathbb{L}_{n-1}^{-1}\mathbb{I}_{n-1,r_{n-1}}\cdots\mathbb{L}_2^{-1}\mathbb{I}_{2,r_2}\mathbb{L}_1^{-1}\mathbb{I}_{1,r_1}\mathbb{A}^{(1)},}$$

其中 $\mathbb{U}$ 是消元结束时的上三角阵，$\mathbb{I}_{k,r_k}$ 和 $\mathbb{L}_k^{-1}$ 分别是第 k 步列主元所导致的交换阵以及后续的 *Gauss* 消元阵。

利用数学归纳法，可以证明

$$\mathbb{U}=\underbrace{\mathbb{L}_{n-1}^{-1}\widetilde{\mathbb{L}}_{n-2}^{-1}\cdots\widetilde{\mathbb{L}}_2^{-1}\widetilde{\mathbb{L}}_1^{-1}}_{\widetilde{\mathbb{L}}^{-1}}\underbrace{\mathbb{I}_{n-1,r_{n-1}}\cdots\mathbb{I}_{2,r_2}\mathbb{I}_{1,r_1}}_{\mathbb{P}}\mathbb{A},\tag{1.1.8}$$

其中 $(k\leq n-2)$

$$\widetilde{\mathbb{L}}_k^{-1}=\mathbb{I}_{n-1,r_{n-1}}\cdots\mathbb{I}_{k+1,r_{k+1}}\mathbb{L}_k^{-1}\mathbb{I}_{k+1,r_{k+1}}\cdots\mathbb{I}_{n-1,r_{n-1}}\tag{1.1.9}$$

的非零元素分布和 $\mathbb{L}_k^{-1}$ 相同，仅仅是消元乘子的所在位置不同。由前面的讨论，可知 $\widetilde{\mathbb{L}}$ 是单位下三角阵。显然，$\mathbb{P}$ 是一个置换阵。

1.1.3 Gauss-Jordan 消元法

Gauss-Jordan(GJ)消元法由 W. Jordan (1842-1899) 和 B. I. Clasen (1887) 独立提出。基本思想同 Gauss 消元法相近，即用对角元素消去同列的其它所有元素，将系数矩阵初等行变换到单位阵。

✍ **论题 1.6.** 设消元过程中的第 k 列向量具 (1.1.4) 的形式，即

$$\boldsymbol{a}_k = (a_{1k}, a_{2k}, \ldots, a_{kk}, \ldots, a_{nk})^\top,$$

其中 $a_{kk} \neq 0$。相应的 *GJ* 消元过程可以描述为 *GJ* 消元阵

$$\mathbb{M}_k = \begin{bmatrix} 1 & & & m_{1k} & & \\ & \ddots & & \vdots & & \\ & & 1 & m_{kk} & & \\ & & & m_{k+1,k} & & \\ & & & m_{k+2,k} & 1 & \\ & & & \vdots & & \ddots \\ & & & m_{n,k} & & 1 \end{bmatrix}$$

的左乘操作，即 $\mathbb{M}_k \boldsymbol{a}_k = \boldsymbol{e}_k$，其中的 *GJ* 消元乘子定义如下

$$m_{ik} = \begin{cases} \frac{1}{a_{kk}}, & i = k; \\ -\frac{a_{ik}}{a_{kk}}, & i \neq k. \end{cases}$$

✯ **说明 1.6.** 用 *GJ* 消元法求解单个线性方程组，相应的乘除次数是 $\mathcal{O}(n^3/2)$。换言之，*GJ* 消元法的计算效率低于 *Gauss* 消元法。

GJ 消元法无需回代，特别适合矩阵求逆操作；相应的 Matlab 命令是 inv()。若在 GJ 消元法中引入主元策略，相应的逆矩阵可以表示为

$$\mathbb{A}^{-1} = \mathbb{M}_n \mathbb{M}_{n-1} \mathbb{I}_{n-1,r_{n-1}} \cdots \mathbb{M}_2 \mathbb{I}_{2,r_2} \mathbb{M}_1 \mathbb{I}_{1,r_1},$$

其中 $\mathbb{I}_{k,r_k}$ 是第 k 步列主元导致的交换阵。整个过程包含 $\mathcal{O}(n^3)$ 次乘除运算。针对不同的数据存储策略，它有两种程序实现方式。

其一，提供两倍数据空间存储增广矩阵 $[\,\mathbb{A}\,|\,\mathbb{I}\,]$，利用 GJ 消元法将其变换到 $[\,\mathbb{I}\,|\,\mathbb{A}^{-1}\,]$。它等同于求解 n 个同型方程组 $\mathbb{A}\boldsymbol{x}_j = \boldsymbol{e}_j$。

其二，仅提供数据空间存储矩阵 $\mathbb{A}$，需充分挖掘数据覆盖技术。相应的编程实现略显复杂，伪代码如下：

```
 1. For k = 1, 2, ..., n, Do
 2.     交换 A 的第 k 行和第 p_k 行，其中 p_k 为列主元；
 3.     a_kk = 1/a_kk;
 4.     For i = 1, ..., n 且 i ≠ k, Do a_ik := −a_ik a_kk; Enddo
 5.     For i = 1, ..., n 且 i ≠ k, Do
 6.         For j = 1, ..., n 且 j ≠ k, Do
 7.             a_ij := a_ij + a_ik a_kj;
 8.         Enndo
 9.     Enddo
10.     For j = 1, ..., n 且 j ≠ k, Do a_kj := a_kk a_kj; Enndo
11. Enddo
12. For k = n, n − 1, ..., 1, Do
13.     交换 A 的第 k 列和第 p_k 列；
14. Enddo
```

代码包含三步基本操作：计算同列的消元乘子（3–4 行），执行其余各列的 GJ 消元（5–9 行），进行同行的单位化操作（10 行）。

特别指出，第 12–14 行代码将数据移动到真正的存储位置。事实上，在执行第 k 步 GJ 消元时，实际操作应当是 $\mathbb{M}_k\mathbb{I}_{k,p_k}$ 的左乘，其中 $\mathbb{M}_k$ 为相应的 GJ 消元阵，p_k 是列主元行号。换言之，$\mathbb{M}_k$ 中的消元因子应出现在逆矩阵的第 p_k 列。但是，前面的 11 行代码并未执行任何列交换，相关数据仍覆盖存储在第 k 列。

☆ **说明 1.7.** 矩阵求逆也可以采用其他方法，例如 *Newton* 迭代

$$\mathbb{X}_{k+1} = 2\mathbb{X}_k(\mathbb{I} - \mathbb{A}\mathbb{X}_k),$$

其中 $\mathbb{X}_0$ 是适当选取的初始矩阵。

1.2 直接三角分解算法

本节介绍 Gauss 消元法的其它等价实现途径。直接三角分解算法基于系数矩阵的各种分解方式，将问题转化为容易求解的三角形方程组。由于计算过程（数据走向和运算次序）明显不同，它们在计算机上的舍入误差表现和最终计算结果也有差异。

1.2.1 LU 分解

定义 1.3. 称矩阵 $\mathbb{A}$ 具有三角分解（或 *LU* 分解），若有

$$\mathbb{A} = \mathbb{L}\mathbb{U}, \tag{1.2.10}$$

其中 $\mathbb{L}$ 是下三角阵，$\mathbb{U}$ 是上三角阵。特别地，若 $\mathbb{L}$ 为单位下三角阵，称其为 *Doolittle* 分解；若 $\mathbb{U}$ 为单位上三角阵，称其为 *Crout* 分解。

强调指出：不是所有矩阵都有 LU 分解，例如

$$\mathbb{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

事实上，关于顺序 Gauss 消元法的讨论已经给出了 LU 分解的一个充分条件，即：若前 $n-1$ 个顺序主子式满足

$$0 \neq \det \mathbb{A}(1:k, 1:k), \quad k = 1:n-1, \tag{1.2.11}$$

则矩阵 $\mathbb{A}$ 具有 Doolittle 分解。

☆ **说明 1.8.** 当(1.2.11) 不成立时，矩阵也可具有 LU 分解，例如

$$\begin{bmatrix} 0 & 0 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{1.2.12}$$

请问：这个矩阵的 LU 分解唯一吗？

☆ **说明 1.9.** 由论题 *1.5* 可知，可逆矩阵 $\mathbb{A}$ 一定有

$$\mathbb{PA} = \mathbb{LU}, \tag{1.2.13}$$

其中 $\mathbb{P}$ 是某个置换阵，$\mathbb{L}$ 和 $\mathbb{U}$ 分别是下三角阵和上三角阵。有时，它也称为**三角分解**。

定义 1.4. 称 $\mathbb{A}$ 具有 LDR 分解，若有 $\mathbb{A} = \mathbb{LDR}$，其中 $\mathbb{D}$ 为对角阵，$\mathbb{R}$ 为单位上三角阵，$\mathbb{L}$ 为单位下三角阵。它是矩阵三角分解的标准形式。

定理 1.3. n 阶矩阵 $\mathbb{A}$ 具有唯一的 LDR 分解，当且仅当顺序主子阵 $\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_{n-1}$ 均非奇异，即 (1.2.11) 成立。

证明：数学归纳法和矩阵分块技术的简单应用。详见教科书。 □

1.2.2 Crout 方法和 Doolittle 方法

利用三角分解 (1.2.10) 可以构造相应的数值方法。设计思想非常简单，即直接应用矩阵乘法运算的逐分量计算公式，例如

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

其中某个三角阵的对角元素锁定为 1。若 $\mathbb{U}$（或 $\mathbb{L}$）是单位三角阵，则相应算法称为 Crout（或 Doolittle）方法。

两者的设计思想和实现过程非常相似，相应的 Matlab 命令是 lu()。下面以 Crout 方法为例[6]，介绍此类算法的实现过程。

右下方图文框是相应的伪代码。它暗含了**求和符号的默认规则**：若上标比下标小，则求和运算（对应第 3 行代码和第 6 行代码）是空操作，相应的返回值为 0。伪代码还利用了数据覆盖技术，两个三角阵的关键数据（除去固定取值为 1 的对角线元素）依旧存储在二维数组的相应位置上。

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

在 Crout 方法中，数据走向呈现出瀑布型结构，由左上到右下，按照先列后行的方式进行更新。特别指出：在计算过程中，每个元素至多更新一次。

❀ **思考 1.1.** *给出 Doolittle 方法的实现过程；采用 BLAS-2 或 BLAS-3 代码级别，重写 Crout 方法和 Doolittle 方法。*

上述算法可视为 Gauss 消元法的不同实现方式。虽两者理论上等价，但存在明显的区别：

1. Gauss 消元法采用逐次消元策略，每次消元操作都要影响所在位置右下方阵的整块数据。数据指针要频繁地移动，消耗大量的数据读写时间，影响整体的计算效率。

2. 直接三角分解方法采用 “一步到位” 的消元策略，每个给定位置的

[6]Doolittle 方法是类似的。事实上，Doolittle 算法就是顺序（或列主元）Gauss 消元法的不同实现过程而已。若计算过程是精确的，相应的计算结果是完全一致的，仅仅是计算流程和数据控制略有不同。

元素至多更新一次，只需读取与其同行或同列的相关数据，在数据读写效率方面具有优势。简而言之，它是“需求驱动”的算法，更适合规模庞大的线性方程组求解。

✯ **说明 1.10.** 要在 *Crour* 方法中引入列主元策略，需在选取主元之前算出相应列的所有元素。

1.2.3 Cholesky 方法

定理 1.4. 对于实对称正定矩阵 $\mathbb{A}$，有 *Cholesky* 分解

$$\mathbb{A} = \mathbb{L}\mathbb{L}^\top, \tag{1.2.14}$$

其中 $\mathbb{L}$ 是下三角阵。若 $\mathbb{L}$ 的对角元素均为正数，则分解是唯一的。

✎ **论题 1.7.** *Cholesky* 方法[7]也称为 $LL^\top$ 方法，相应的 *Matlab* 命令是 *chol()*。基本公式是

$$\boxed{a_{ij} = \sum_{k=1}^{j} l_{ik} l_{jk}, \quad i \geq j.}$$

对角元素 l_{ii} 的计算需要开根运算（等同于多次乘除运算），故 *Cholesky* 方法也称为平方根法。

矩阵 $\mathbb{L}$ 有逐列或逐行两种计算次序，其中逐行次序不易并行，逐列次序更为普遍。下方图文框是基于逐列次序的伪代码，用 l_{ij} 覆盖了 a_{ij}。基于算法的执行特点，Cholesky 方法有两个俗称：(1)读取数据均位于待更新数据的左侧，故

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

[7] André-Louis Cholesky 是法国军官，潜心于测地学研究，勘测过希腊克里特岛和北非。

俗称“向左看”算法；(2) 第 j 列数据直到第 j 步循环才被更新，故也称“需求驱动”或者“延迟更新”算法。

定理 1.5. 设 $\mathbb{A}=(a_{ij})$ 实对称正定，则 $l_{ij}\le\sqrt{a_{ii}}$，其中 $j\le i$。

定理表明，矩阵 $\mathbb{L}$ 的元素大小可控，Cholesky 方法数值稳定[8]，不必选取所谓的主元。

论题 1.8. 为避免平方根算法中的开根运算(比乘除运算慢得多)，可采用修正平方根算法。它基于标准三角分解

$$\mathbb{A}=\mathbb{L}\mathbb{D}\mathbb{L}^{\top},$$

其中 $\mathbb{L}$ 是单位下三角阵，$\mathbb{D}$ 是一组正数构成的对角阵。

在 Matlab 中，修正平方根算法的命令是 ldl()。其计算公式是

$$a_{ij}=\sum_{k=1}^{j}l_{ik}d_kl_{jk},\quad i\ge j,$$

逐行计算 $\mathbb{L}$ 和 $\mathbb{D}$ 的编程实现更为便捷。下面两个图文框给出该算法的两个代码实现，其中左侧代码的乘除次数将是 $\mathrm{LL}^{\top}$ 算法的两倍。

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

[8]已知数据的微小扰动不会造成数值结果的无限放大。具体定义见后面的摄动理论。

右侧代码是计算复杂度缩减的典型实例。为减少左侧代码的重复运算(左侧第 3 行和第 5 行)，右侧代码引进中间变量

$$g_{ij} = l_{ij}d_j,$$

对应右侧第 3 行的 a_{ij} 和第 6 行的局部变量 c。右侧的第 6 行对应 g_{ij} 到 l_{ij} 的计算过程，相应数据存储在原有位置（对应第 3 行的 a_{jk}）。

✯ **说明 1.11.** 对于对称阵，正定性可以保证 $LDL^\top$ 算法具有数值稳定性，但是不定性可能导致很大的麻烦。下面给出一个实例。当 $|\varepsilon| \ll 1$ 很小时，非正定矩阵 $\mathbb{A}$ 依旧有 $LDL^\top$ 分解

$$\mathbb{A} = \begin{bmatrix} \varepsilon & 1 \\ 1 & \varepsilon \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \varepsilon^{-1} & 1 \end{bmatrix} \begin{bmatrix} \varepsilon & 0 \\ 0 & \varepsilon - \varepsilon^{-1} \end{bmatrix} \begin{bmatrix} 1 & \varepsilon \\ 0 & 1 \end{bmatrix},$$

但第二个对角阵元素不再恒正且取值巨大。此时，若利用上述 $LDL^\top$ 分解计算 $\mathbb{A}^{-1}$ 或求解线性方程组，浮点运算遭遇舍入误差的极大干扰。但是，直接计算可知

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

按此公式计算，舍入误差的影响将非常小。事实上，对于非正定的对称矩阵，常把列主元 *Gauss* 消元法和二阶分块矩阵技术相结合，形成所谓的块 *Gauss* 消元法，使其具有良好的数值稳定性。

1.2.4 追赶法

在实际应用中，线性方程组常常是稀疏的，即系数矩阵包含极高比例的零元素。此时，简单执行 Gauss 消元法不是最佳选择，需关注以下两点。其一是数据存储，通常二维数组存储不再经济；其二是计算复杂度的精简，尽量避免在无价值（例如乘零或加零等）的运算上浪费 CPU

时间。换言之，编程实现要发掘稀疏矩阵的结构（如非零元素分布），优化原有算法的数据存储和程序代码。

✯ **说明 1.12.** 稀疏存储涉及计算机科学中的数据结构和快速搜索等诸多技术。例如，所有的非零元素可采用结构体方式进行记录，它包括位置 (i,j)、值 a_{ij} 以及双向链表（同行相邻非零元的两个列号，同列相邻非零元的两个行号）等数据。关于稀疏矩阵的 *Matlab* 命令有 *sparse(), speye(), spones(), spdiag(), full()* 等；详细内容可参阅系统提供的帮助文件，此处不再赘述。

✯ **说明 1.13.** 利用 *Gauss* 消元法求解稀疏线性方程组，两个三角阵的非零元素分布可能与系数矩阵 $\mathbb{A}$ 不同，在原本为零的位置产生新的非零元素。若新增比例过大，数据存储将遇到危机。为此，数值算法要控制非零元素的增长数量，著名策略有不完全三角分解 (*ILU*) 技术；参见教科书，此处不再赘述。

✇ **定义 1.5.** 若远离对角线指定距离的元素 a_{ij} 均为零，相应的矩阵称为带状矩阵。若 $j>i+q$ 均有 $a_{ij}=0$，则称上带宽为 q；若 $i>j+q$ 均有 $a_{ij}=0$，则称下带宽为 q；称 $d=\max(p,q)$ 为半带宽。

✯ **说明 1.14.** 带状矩阵可采用**斜线存储**技术进行存储。通常，半带宽内会有大量的零元素，该技术并不是最好的稀疏存储技术。

定理 1.6. 若带状矩阵具有 *LU* 分解 $\mathbb{A}=\mathbb{LU}$，则两个三角形矩阵的上下带宽与 $\mathbb{A}$ 相同。

✎ **论题 1.9.** 设可逆矩阵具有半带宽 d，采用斜线存储技术。重写顺序高斯消去法，省掉无必要的零元操作，并估计相应的计算复杂度。若还要选取列主元，相应的算法怎样实现？

最简单的带状矩阵是半带宽为 1 的三对角阵

$$\mathbb{A} = \text{tridiag}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) = \begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \cdots & \cdots & \cdots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix},$$

其中三个向量 $\boldsymbol{a} = (a_i), \boldsymbol{b} = (b_i)$ 和 $\boldsymbol{c} = (c_i)$ 分别从下到上的三条对角线。所谓的斜线存储技术，就是用二维数组存储这三个向量。

✍ **论题 1.10.** 对于三对角方程组，相应的 *Gauss* 消元法称为追赶法或 *Thomas* 算法。事实上，它就是 *Crout* 算法，其伪代码片段是

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

矩阵三角分解过程需要 $\mathcal{O}(2n)$ 次乘除运算，其中每个三角阵仅仅有两条斜线非零。追和赶的过程对应两个简单三角方程组

$$\mathbb{L}\boldsymbol{y} = \boldsymbol{b}, \quad \mathbb{U}\boldsymbol{x} = \boldsymbol{y}$$

的快速求解，需要 $\mathcal{O}(3n)$ 次乘除运算。换言之，追赶法的计算复杂度同未知量成正比例。

定理 1.7. 若三对角矩阵是严格对角占优的，即

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|, \quad i = 1 : n,$$

则追赶法可以顺利进行到底。换言之，它无需进行主元选取，相应的数值结果也是稳定的。

证明：数学归纳法。见教科书。 □

事实上，三对角方程组还有其他求解方法。因篇幅限制，下面仅仅介绍两个有趣的方法。

- **变参数追赶法**：它基于矩阵分解

$$\mathbb{A} = \mathbb{D}\mathbb{L}\mathbb{R},$$

其中 $\mathbb{D} = \mathrm{diag}(d_1, d_2, \ldots, d_n)$ 是 n 阶对角阵，而

$$\mathbb{L} = \begin{bmatrix} l_1 & 1 & & & \\ & l_2 & 1 & & \\ & & \ddots & \ddots & \\ & & & l_n & 1 \end{bmatrix}, \quad \mathbb{R} = \begin{bmatrix} u_1 & & & & \\ 1 & u_2 & & & \\ & 1 & \ddots & & \\ & & \ddots & u_n & \\ & & & 1 & \end{bmatrix}$$

分别是 $n \times (n+1)$ 阶广义上三角阵和 $(n+1) \times n$ 阶广义下三角阵。其核心是两个自由参数 l_1 和 u_1 的恰当选取，要求 $l_1 u_1 + 1 \neq 0$。具体计算公式参见 [10]；因篇幅有限，不再赘述。

- **线性插值法**：考虑前 $n-1$ 个方程构成的线性方程组，其通解可表示为两个特解的线性组合，即

$$\theta(0, \xi_2, \ldots, \xi_n)^\top + (1-\theta)(1, \eta_2, \ldots, \eta_n)^\top,$$

其中 θ 是待定的组合参数。假设 c_i 均非零，则右上角的 $n-1$ 阶方阵 $\mathbb{A}(1:n-1, 2:n)$ 是一个可逆的（双斜线）下三角阵，两个特解可以轻松地得到。最后，将所得结果代入到最后一个方程，简单计算即可确定 θ。

❀ **思考 1.2.** 在三对角阵的右上角和左下角位置填补两个非零元素，所得矩阵称为循环三对角阵。利用矩阵分解技术，给出循环三对角方程组的追赶法。

✯ **说明 1.15.** 带状矩阵的逆矩阵通常不是带状的。若无特别要求，带状矩阵线性方程组不会求解系数矩阵的逆矩阵，而是采用 *Gauss* 消去法或其等价途径去求解。

设 $\mathbb{H}$ 是不可约的上 Hessenberg 阵，即非零元仅位于上三角部分和下方副对角线，且下方副对角线元均非零。Ikebe (1979) 指出：此时的逆矩阵 $\mathbb{H}^{-1}$ 具有漂亮结构，即下三角部分的元素可以表示为

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

其中 $\boldsymbol{p} = (p_i)_{i=1:n}$ 和 $\boldsymbol{q} = (q_j)_{j=1:n}$ 是某两个列向量。

❀ **思考 1.3.** 三对角阵属于上 *Hessenberg* 矩阵。请利用 *Ikebe* 的结果，给出不可约三对角对称矩阵的求逆算法。不妨预设参数 $q_1 = 1$。

1.3 向量范数和矩阵范数

向量范数和矩阵范数[9]是广泛采用的度量工具，定义方式同泛函分析的范数概念基本一致，唯一的区别是：直至 1940–1950 年，关于矩阵范数第四条规则的增补才达到共识。

1.3.1 定义和性质

✇ **定义 1.6.** 映射 $\|\cdot\|\colon \mathbb{R}^n \to \mathbb{R}$ 称为向量范数，若其满足

[9]本讲义默认实数域；概念和结论可以容易地推广到复数域。

1. 非负性：$\|\boldsymbol{x}\| \geq 0$ 且 $\|\boldsymbol{x}\| = 0$ 当且仅当 $\boldsymbol{x} = 0$；

2. 齐次性：对于任意的 $c \in \mathbb{R}$ 和 $\boldsymbol{x} \in \mathbb{R}^n$，均有 $\|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$；

3. 三角不等式：对于任意的 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$，均有 $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$。

具有范数度量的线性空间 $\mathbb{R}^n$ 称为赋范空间。

Hölder 范数是常用的向量范数。设 $\boldsymbol{x} = (x_i)_{i=1}^n$；给定 $p \geq 1$，相应的 Hölder 范数（或 l_p 范数）定义为

$$\|\boldsymbol{x}\|_p = \begin{cases} \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, & 1 \leq p < \infty; \\ \max\limits_{1 \leq i \leq n} |x_i|, & p = \infty. \end{cases}$$

当 $p = 2$ 时，它也称为 Euclid 范数；当 $p = \infty$ 时，它也称为最大模范数。在 $\mathbb{R}^n$ 空间，两个向量的（标准）内积定义为

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x}^\top \boldsymbol{y}, \quad \forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n.$$

它满足著名的 Hölder 不等式

$$|\boldsymbol{x}^\top \boldsymbol{y}| \leq \|\boldsymbol{x}\|_p \|\boldsymbol{y}\|_q, \quad 1/p + 1/q = 1.$$

定义 1.7. 矩阵范数 $\|\cdot\|$ 是 $\mathbb{R}^{n \times n} \to \mathbb{R}$ 的映射，并满足 *(a)* 非负性；*(b)* 齐次性；*(c)* 三角不等式；*(d)* 相容性，即

$$\|\mathbb{A}\mathbb{B}\| \leq \|\mathbb{A}\|\|\mathbb{B}\|, \quad \forall \mathbb{A}, \mathbb{B} \in \mathbb{R}^{n \times n}.$$

说明 1.16. 在矩阵范数的定义中，前三条规则可视为向量范数的自然推广，而第四条规则（相容性）是矩阵范数特有的规则。

❀ **思考 1.4.** 设 $\mathbb{A}=(a_{ij})$ 是 n 阶矩阵，$\max_{ij}|a_{ij}|$ 和 $n\max_{ij}|a_{ij}|$ 都是矩阵范数吗？

在 Matlab 中，向量范数和矩阵范数的命令都是 norm().

定理 1.8. 向量（或矩阵）范数一致连续，且彼此等价。

1.3.2 两类范数的关系

定义 1.8. 设 $\|\cdot\|_\alpha$ 为矩阵范数，$\|\cdot\|_\beta$ 为向量范数。若成立

$$\|\mathbb{A}\boldsymbol{x}\|_\alpha \le \|\mathbb{A}\|_\beta\|\boldsymbol{x}\|_\alpha, \quad \forall \boldsymbol{x}\in\mathbb{R}^n.$$

则称 $\|\cdot\|_\beta$ **相容**于 $\|\cdot\|_\alpha$。进一步地，若存在某个非零向量将不等式变成等号，则称 $\|\cdot\|_\beta$ **从属**于 $\|\cdot\|_\alpha$。

性质 1.1. 从属关系成立的必要条件是 $\|\mathbb{I}_{n\times n}\|_\beta = 1$。

定理 1.9. 对于任意的矩阵范数 $\|\cdot\|_\beta$，均存在某个向量范数 $\|\cdot\|_\alpha$，使得两者相容。

证明：利用零填充，将向量列扩张成矩阵。 □

定理 1.9 仅确保了相容关系，并不保证从属关系。例如，Frobenius 范数（又称为 Suchur 范数）

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

与 l_2 向量范数相容，却不从属于任何向量范数。

❀ **思考 1.5.** 证明不等式 $\|\mathbb{AB}\|_F \leq \|\mathbb{A}\|_F\|\mathbb{B}\|_F$，进而说明 $\|\cdot\|_F$ 是矩阵范数。

定理 1.10. 向量范数 $\|\cdot\|_\alpha$ 均可导出算子范数

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

它是从属（显然相容）于 $\|\cdot\|_\alpha$ 的矩阵范数。

✍ **论题 1.11.** 对应常用的 l_p 向量范数，定理 *1.10* 给出三个（相容且从属的）矩阵范数：

1. 列范数 $\|\mathbb{A}\|_1 = \max\limits_{1\leq j\leq n} \sum\limits_{i=1}^{n} |a_{ij}|$；
2. 谱范数 $\|\mathbb{A}\|_2 = \left[\varrho(\mathbb{A}^\top\mathbb{A})\right]^{1/2}$，其中 $\varrho(\mathbb{A}^\top\mathbb{A})$ 是 $\mathbb{A}^\top\mathbb{A}$ 的谱半径；
3. 行范数 $\|\mathbb{A}\|_\infty = \max\limits_{1\leq i\leq n} \sum\limits_{j=1}^{n} |a_{ij}|$.

显然，有 $\|\mathbb{A}\|_1 = \|\mathbb{A}^\top\|_\infty$ 和 $\|\mathbb{A}^\top\|_2 = \|\mathbb{A}\|_2$.

定理 1.11. 在（左右）酉变换下，谱范数和 *Frobenius* 范数均保持不变。

✯ **说明 1.17.** 定理 *1.10*、定理 *1.11* 和论题 *1.11* 的结论可以直接推广到任意形状的矩阵。

1.3.3 矩阵范数的重要结论

定理 1.12. 任意（相容）矩阵范数均满足 $\varrho(\mathbb{A}) \leq \|\mathbb{A}\|$.

定理 1.13. 对于给定的矩阵 $\mathbb{A}$ 和 $\varepsilon > 0$，均有矩阵范数 $\|\cdot\|_\star$ 使得

$$\|\mathbb{A}\|_\star \leq \varrho(\mathbb{A}) + \varepsilon. \tag{1.3.15}$$

定理 1.14. *(Banach 引理)* 设矩阵范数 $\|\cdot\|$ 满足 $\|\mathbb{I}\| = 1$，其中 $\mathbb{I}$ 是单位阵。若 $\|\mathbb{A}\| < 1$，则 $\mathbb{I} \pm \mathbb{A}$ 可逆，且

$$\frac{1}{1 + \|\mathbb{A}\|} \leq \|(\mathbb{I} \pm \mathbb{A})^{-1}\| \leq \frac{1}{1 - \|\mathbb{A}\|}. \tag{1.3.16}$$

1.3.4 向量序列和矩阵序列

称向量（或矩阵）序列是收敛的，若相同位置的元素均构成一个收敛的数列。理论分析常常采用范数进行整体描述，即

定义 1.9. $\lim_{k\to\infty} \boldsymbol{x}_k = \boldsymbol{x} \Leftrightarrow \lim_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}\| = 0.$

定义 1.10. $\lim_{k\to\infty} \mathbb{A}_k = \mathbb{A} \Leftrightarrow \lim_{k\to\infty} \|\mathbb{A}_k - \mathbb{A}\| = 0.$

既然向量（或矩阵）范数彼此等价，定义中的范数可以任意选取。

关于矩阵序列（或级数）的收敛性判定，矩阵范数和谱半径是常用的分析工具。主要结论有

定理 1.15. $\lim_{k\to\infty} \mathbb{A}^k = \mathbb{O} \Leftrightarrow \varrho(\mathbb{A}) < 1.$

证明：见教科书上册第 119 页。 □

定理 1.16. $\lim_{k\to\infty} \|\mathbb{A}^k\|^{1/k} = \varrho(\mathbb{A}).$

证明：利用定理 1.15 和极限夹挤原理即可。 □

定理 1.17. 矩阵级数 $\sum_{k=0}^{\infty} \mathbb{B}^k$ 收敛的充要条件是 $\varrho(\mathbb{B}) < 1$，且

$$\sum_{k=0}^{\infty} \mathbb{B}^k = (\mathbb{I} - \mathbb{B})^{-1}.$$

若存在某个范数使得 $\|\mathbb{B}\| < 1$，则矩阵级数 $\sum_{k=0}^{\infty} \mathbb{B}^k$ 也是收敛的。相应的余项满足

$$\left\| \sum_{k=m+1}^{\infty} \mathbb{B}^k \right\| \leq \sum_{k=m+1}^{\infty} \|\mathbb{B}\|^k \leq \frac{\|\mathbb{B}\|^{m+1}}{1 - \|\mathbb{B}\|}.$$

这个性质同绝对收敛幂级数的性质具有形式上的一致性，可将其视为有限项的三角不等式的推广。

证明：简单验证即可。 □

1.4 线性方程组的摄动理论

当定解数据变化时，线性方程组的解向量也会随之改变，其敏感度（或健壮性）同系数矩阵的条件数密切相关。

1.4.1 矩阵条件数

Matlab 命令 rcond() 可以给出矩阵条件数。

定义 1.11. 对于可逆矩阵 $\mathbb{A}$，关于范数 $\|\cdot\|$ 的条件数是

$$\kappa(\mathbb{A}) = \|\mathbb{A}\|\|\mathbb{A}^{-1}\|.$$

若 $\kappa(\mathbb{A})$ 非常大[10]，称其病态；否则，称其良态。

论题 1.12. 设 $\mathbb{A}$ 实对称正定，相应的谱条件数是

$$\kappa_2(\mathbb{A}) = \frac{\lambda_{\max}}{\lambda_{\min}}, \tag{1.4.17}$$

其中 $\lambda_{\max}$ 和 $\lambda_{\min}$ 分别是最大和最小特征值。

[10]断言是相对的，同当前计算机所能提供的计算能力有关。

矩阵条件数是问题本身的固有性质, 同数值方法无关。一般而言, 数值方法的可靠程度不会超过问题本身的健壮程度。

定理 1.18. 矩阵条件数具有如下性质:

1. $\kappa(\mathbb{A}) \geq 1$;
2. $\kappa(c\mathbb{A}) = \kappa(\mathbb{A}) = \kappa(\mathbb{A}^{-1})$, 其中 $c \neq 0$;
3. $\kappa(\mathbb{A}\mathbb{B}) \leq \kappa(\mathbb{A})\kappa(\mathbb{B})$。
4. 任意条件数都是等价的。

强调指出: 上述不等式都是可以等号成立的, 相关估计是不可改善的。

☆ **说明 1.18.** *Hilbert* 矩阵 $\mathbb{H}_n = (h_{ij})$ 是著名的病态矩阵, 其中

$$h_{ij} = \frac{1}{i+j-1}.$$

相应的逆矩阵为 $\mathbb{H}_n^{-1} = (b_{ij})$, 其中

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

在 *Matlab* 中, 获得相应矩阵的命令是 *hilb()* 和 *invhilb()*。

Vandermonde 矩阵 $\mathbb{V}(\boldsymbol{v}) = (x_i^{n-j})$ 也是著名的病态矩阵, 尤其当 $\boldsymbol{v} = \{x_i\}$ 是等距分布点列形成的向量。在 *Matlab* 中, 获得相应矩阵的命令是 *vander()*。

定理 1.19 (Kahan, 1996)**.** 矩阵条件数描述了可逆矩阵 $\mathbb{A}$ 同奇异矩阵集合的接近程度, 即

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

证明：Banach 引理表明左端不低于右端，故只需证明等号可以成立。取单位向量 $\boldsymbol{x}$，使得 $\|\mathbb{A}^{-1}\boldsymbol{x}\|_2 = \|\mathbb{A}^{-1}\|_2$。令

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

验证可知 $(\mathbb{A}+\delta\mathbb{A})\boldsymbol{y} = 0$。至此，定理得证。 □

✯ **说明 1.19.** 有广泛流传的错误说法：行列式越接近零，矩阵的病态程度就会越高。事实上，两个概念没有直接联系。下面给出相应的两个反例：

- 对角元恒为 10^{-1} 的 n 阶数量矩阵，其行列式是 10^{-n}，但条件数恒为 1；
- 三角阵（严格上三角部分的元素都是 -1）

$$\mathbb{A}_n = \begin{bmatrix} 1 & -1 & \cdots & -1 \\ 0 & 1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

  行列式为 1，但 $\kappa_\infty(\mathbb{A}_n) = n2^{n-1}$。

1.4.2 摄动分析

设系数矩阵 $\mathbb{A}$ 可逆。考虑线性方程组 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ 的扰动问题

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

其中 $\delta\mathbb{A}$ 是扰动矩阵，$\delta\boldsymbol{b}$ 是扰动向量。

定理 1.20. 若 $\|\mathbb{A}^{-1}\|\|\delta\mathbb{A}\| < 1$，则扰动问题也唯一可解。此时，解向量相对改变量“正比例”于矩阵条件数 $\kappa(\mathbb{A})$，具体结果有

1. 仅右端向量有扰动: $\dfrac{\|\delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \kappa(\mathbb{A}) \dfrac{\|\delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|}$.

2. 仅系数矩阵有扰动: $\dfrac{\|\delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \dfrac{\kappa(\mathbb{A}) \frac{\|\delta \mathbb{A}\|}{\|\mathbb{A}\|}}{1-\kappa(\mathbb{A}) \frac{\|\delta \mathbb{A}\|}{\|\mathbb{A}\|}}$.

上述估计不可改善，因为等号可以成立。但是，在实际应用时，它们显得过于保守。不妨举例说明。给定 $\gamma \gg 1$，设 $\delta \to 0$ 是扰动量，简单计算可知

$$\begin{bmatrix} \gamma+\delta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \gamma \\ 1 \end{bmatrix}, \quad \delta \to 0$$

的相对改变量位于 1 附近，同理论上界 $\kappa_2(\mathbb{A}) = 1/\gamma$ 相距甚远。针对某些具体问题，一些实用但略带风险的上界估计已经被提出；此处不做展开介绍。

1.4.3 可靠性分析

数值结果的可靠性判断，是自然产生的实际问题。较为常用的方法是随机产生多个“真解”进行测算，即：利用“真解”计算出右端向量，构造出同型的线性方程组，利用试算结果和已知真解的差距估计有效数字的可信长度。

可靠性分析更多地采用**后验误差估计**

$$\frac{\|\boldsymbol{x}_\star - \boldsymbol{x}_{\text{num}}\|}{\|\boldsymbol{x}_\star\|} \leq \kappa(\mathbb{A}) \frac{\|\boldsymbol{r}\|}{\|\boldsymbol{b}\|}, \tag{1.4.18}$$

其中 $\boldsymbol{x}_{\text{num}}$ 是数值解，$\boldsymbol{r} = \mathbb{A}\boldsymbol{x}_{\text{num}} - \boldsymbol{b}$ 是残量[11]。在 (1.4.18) 中，多采用无穷范数度量。

[11]残量的计算结果较为可信，相关的舍入误差积累远远小于 Gauss 消元过程。

要估计条件数 $\kappa_\infty(\mathbb{A})$，需分别估算 $\|\mathbb{A}\|_\infty$ 和 $\|\mathbb{A}^{-1}\|_\infty$。困难主要集中在后者，因为逆矩阵要么计算不易，要么可靠性堪忧。常用的解决方案如下：令 $\mathbb{B}=\mathbb{A}^{-\top}$，有 $\|\mathbb{A}^{-1}\|_\infty=\|\mathbb{B}\|_1$。基于优化理论中的“盲人下山法”，数值软件包 LAPACK 有 $\|\mathbb{B}\|_1$ 的估算方法：

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

利用 Gauss 消元法给出的三角阵 $\mathbb{L}$ 和 $\mathbb{U}$，图文框中的第 2 步可以快速实现，只需 $\mathcal{O}(n^2)$ 次乘除运算。

✯ **说明 1.20.** 计算精度可通过迭代过程进行改进，其基本思想就是利用同型线性方程组不断修正残量。略。

1.4.4 浮点误差分析

列主元 Gauss 消元法是精确算法，没有方法误差，只有舍入误差。数值结果的准确程度依赖两个要素：矩阵条件数和机器精度。

首先回顾浮点运算的相关概念。在计算机上，浮点数[12]通常表示为

$$f=\pm 0.d_1d_2\cdots d_t\times 2^J,\quad d_1\neq 0,$$

其中 t 是机器位长，J 的最大值给出浮点数集的取值范围。特别指出：浮点数仅仅构成离散子集，不满足四则运算（加减乘除）的封闭性。

[12]实际上，$d_1=1$ 是不需要存储的。

用 $a \star b$ 表示两个数的四则操作，用 $fl(a \star b)$ 表示相应的浮点运算。简单分析可知

$$fl(a \star b) = (a \star b)(1+\delta), \quad |\delta| \leq \vartheta, \tag{1.4.19}$$

其中 ϑ 称为机器精度，依赖寄存器的具体构造和工作机理，通常

$$\vartheta = \begin{cases} 0.5 \times 2^{1-t}, & \text{舍入法}, \\ 2^{1-t}, & \text{截断法}. \end{cases}$$

就双精度数据（$t \approx 64$）而言，机器精度位于 $10^{-16} \sim 10^{-17}$ 量级。

不断应用基本估计 (1.4.19)，可以建立一些拓展性结果。因篇幅限制，具体推导可参考相关文献。当 $n\vartheta$ 较小的时候，向量内积满足

$$|fl(\boldsymbol{x}^\top \boldsymbol{y}) - \boldsymbol{x}^\top \boldsymbol{y}| \leq 1.01 n\vartheta |\boldsymbol{x}|^\top |\boldsymbol{y}|, \tag{1.4.20}$$

其中 n 为向量维数。类似地，矩阵运算满足

$$|fl(\alpha \mathbb{A}) - \alpha \mathbb{A}| \leq \vartheta |\alpha \mathbb{A}|, \tag{1.4.21a}$$

$$|fl(\mathbb{A} + \mathbb{B}) - (\mathbb{A} + \mathbb{B})| \leq \vartheta |\mathbb{A} + \mathbb{B}|, \tag{1.4.21b}$$

$$|fl(\mathbb{A}\mathbb{B}) - \mathbb{A}\mathbb{B}| \leq 1.01 n\vartheta |\mathbb{A}||\mathbb{B}|, \tag{1.4.21c}$$

其中 n 为矩阵阶数。在上述估计中，绝对值运算和不等式关系都是针对每个元素而言。

上述内容俗称“向前误差分析”，相应的论证过程繁琐且严重依赖计算环境。为更好描述舍入误差在一个复杂算法中的积累和传播，数值分析更多采用**向后误差分析**技术，即：假定计算过程都是精确的，并将所有的舍入误差都归结为初始数据的某种扰动。基于此观点，(1.4.21a) 可重新表示为

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

其中 $\mathbb{E}$ 为扰动矩阵。其它估计可类似处理，不再赘述。

下面回到线性方程组 $\mathbb{A}\boldsymbol{x}=\boldsymbol{b}$ 的浮点误差分析，其中 $\mathbb{A}=(a_{ij})$ 是可逆方阵。基于向后误差分析技术，列主元 Gauss 消元法的运行结果可视为某个扰动问题

$$(\mathbb{A}+\delta\mathbb{A})(\boldsymbol{x}+\delta\boldsymbol{x})=\boldsymbol{b} \tag{1.4.22}$$

的精确解 $\boldsymbol{x}+\delta\boldsymbol{x}$，其中 $\delta\boldsymbol{x}$ 是数值偏差，$\delta\mathbb{A}$ 是扰动矩阵。相对误差的上界可由摄动理论给出，即

$$\frac{\|\delta\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty}\le\frac{\kappa_\infty(\mathbb{A})\frac{\|\delta\mathbb{A}\|_\infty}{\|\mathbb{A}\|_\infty}}{1-\kappa_\infty(\mathbb{A})\frac{\|\delta\mathbb{A}\|_\infty}{\|\mathbb{A}\|_\infty}}. \tag{1.4.23}$$

依据算法在计算机上的执行过程，可以给出 $\|\delta\mathbb{A}\|_\infty$ 的合理估计。

Gauss 消去法的实现过程可以分解为两个步骤。其一是矩阵分解

$$\mathbb{PA}+\mathbb{E}=\mathbb{LU},$$

其中 $\mathbb{E}=(e_{ij})$ 是扰动矩阵，$\mathbb{P}$ 是置换矩阵，$\mathbb{L}=(\ell_{ij})$ 和 $\mathbb{U}=(u_{ij})$ 是计算出来的两个三角阵。其二是回代过程，等价于精确求解三角形问题

$$(\mathbb{L}+\mathbb{F})\boldsymbol{y}=\mathbb{P}\boldsymbol{b},\quad(\mathbb{U}+\mathbb{G})(\boldsymbol{x}+\delta\boldsymbol{x})=\boldsymbol{y},$$

其中 $\mathbb{F}=(f_{ij})$ 和 $\mathbb{G}=(g_{ij})$ 是相应的扰动矩阵。综上所述，(1.4.22) 中的扰动矩阵可以写作

$$\delta\mathbb{A}=\mathbb{E}+\mathbb{P}(\mathbb{FU}+\mathbb{LG}+\mathbb{FG}). \tag{1.4.24}$$

利用数学归纳法和向前误差分析技术，可证

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

其中 ϑ 是机器精度，n 是矩阵阶数，$a_{ij}^{(k)}$ 是在第 k 步 Gauss 消元之后存储在计算机上的数据。在列主元 Gauss 消元法中，消元乘子的绝对值均不超过 1，即 $\|\mathbb{L}\|_\infty \leq n$。定义主元增长因子

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

则有 $\|\mathbb{U}\|_\infty \leq n\eta(\mathbb{A})\|\mathbb{A}\|_\infty$。当 $n\vartheta$ 较小的时候，利用上述结果可知

$$\|\delta\mathbb{A}\|_\infty \leq Cn^3\vartheta\eta(\mathbb{A})\|\mathbb{A}\|_\infty, \tag{1.4.25}$$

其中 $C \approx 10$ 为绝对常数，关于 n 的低阶项被省略。

☆ **说明 1.21.** 列主元消元执行一次，对角线右上方的元素绝对值至多放大两倍。因此，主元增长因子 $\eta(\mathbb{A})$ 永远不会超过 2^{n-1}。在某些个例中，这个上限是可以取到的。但是，大量的数值经验表明，$\eta(\mathbb{A})$ 常常处于 $n^{2/3}$ 或 $n^{1/2}$ 的量级。

由 (1.4.23) 和 (1.4.25) 可知：当 $n\vartheta$ 较小的时候，计算机给出的数值结果满足相对误差估计

$$\frac{\|\delta\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} \leq Cn^3\vartheta\eta(\mathbb{A})\kappa_\infty(\mathbb{A}), \tag{1.4.26}$$

其中 C 是与 n 无关的绝对常数。结合说明 1.21 和估计 (1.4.26)，可以断言：列主元 Gauss 消元法大体上可行，具有较为理想的数值稳定性，数值解相对误差可以得到合理的控制。

第 2 章
线性方程组的迭代法

随着线性方程组的计算规模不断增长，直接法面临数据存储和 CPU 时间的压力越来越大，甚至在当前计算环境下根本无法忍受。此时，一个自然的选择是放弃精确求解（不计舍入误差）策略，转向近似求解策略。相应技术就是迭代法，即：通过简单易行（甚至同稀疏存储相匹配）的迭代公式，自动生成一个序列，使其快速收敛到精确解。迭代法不用改变系数矩阵的元素，更加适合稀疏矩阵的数值求解。

2.1 基本理论

给定正整数 r，相应的 r 阶迭代方法具有如下形式：给出（或猜测）启动向量 $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{r-1}$；通过简单易行的迭代函数 $\boldsymbol{f}_k$，利用当前位置及其局部历史信息生成后续位置

$$\boldsymbol{x}_k = \boldsymbol{f}_k(\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}, \ldots, \boldsymbol{x}_{k-r}), \quad k \geq r. \tag{2.1.1}$$

若 $\boldsymbol{f}_k$ 同迭代步数 k 无关，则称迭代法是定常的；否则，称作非定常的。对于线性方程组，迭代法默认是**完全相容**的，即

> 当 k 充分大时，精确解 $\boldsymbol{x}_\star = \mathbb{A}^{-1}\boldsymbol{b}$ 一直满足迭代公式
>
> $$\boldsymbol{x}_\star = \boldsymbol{f}_k(\boldsymbol{x}_\star, \boldsymbol{x}_\star, \ldots, \boldsymbol{x}_\star).$$

换言之，一旦迭代到精确解，后续迭代就不再远离。

2.1.1 一阶迭代方法

一阶迭代方法的结构最为简单，通常有以下两种表示形式

$$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \mathbb{H}_k(\boldsymbol{b} - \mathbb{A}\boldsymbol{x}_{k-1}) = \boldsymbol{x}_{k-1} - \mathbb{H}_k\boldsymbol{r}_{k-1}, \tag{2.1.2a}$$

$$\boldsymbol{x}_k = \mathbb{G}_k\boldsymbol{x}_{k-1} + \boldsymbol{g}_k, \tag{2.1.2b}$$

其中 $\mathbb{H}_k$ 称为预处理矩阵，$\mathbb{G}_k$ 称为迭代矩阵，$\boldsymbol{r}_k = \mathbb{A}\boldsymbol{x}_k - \boldsymbol{b}$ 称为残量。若 $\boldsymbol{r}_k = \boldsymbol{0}$，则 $\boldsymbol{x}_k = \boldsymbol{x}_\star$，即预处理形式自动实现完全相容。

论题 2.1. 两种表示形式可以互相导出，且

$$\mathbb{G}_k = \mathbb{I} - \mathbb{H}_k\mathbb{A}, \quad \boldsymbol{g}_k = \mathbb{H}_k\boldsymbol{b}.$$

迭代法的核心是迭代矩阵或预处理矩阵。相较于前者，后者的设计目标非常明确。若 $\mathbb{H}_{k+1} = \mathbb{A}^{-1}$，则一步迭代就有 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_\star$。虽然这个极端设置不现实，但是它已经指出方向：预处理矩阵和系数矩阵应有近似的互逆关系。

2.1.2 收敛分析

记 $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}_\star$ 为第 k 步迭代误差。

定义 2.1. 若对任意的初始向量 $\boldsymbol{x}_0$，均有 $\lim_{k\to\infty}\boldsymbol{e}_k = \boldsymbol{0}$，则称迭代法收敛，否则称其发散。

理论分析大多源于误差方程

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

定理 2.1. 迭代法收敛等价于迭代矩阵的乘积趋于零矩阵，即

$$\lim_{k\to\infty}\Pi_{m=1}^{k}\mathbb{G}_m = \lim_{k\to\infty}\Pi_{m=1}^{k}(\mathbb{I} - \mathbb{H}_m\mathbb{A}) = \mathbb{O}.$$

定理 2.2. 若迭代矩阵 $\mathbb{G}_k \equiv \mathbb{G}$ 或预处理矩阵 $\mathbb{H}_k \equiv \mathbb{H}$，即算法是定常的，则上述结果可以简化为

$$\varrho(\mathbb{G}) < 1.$$

它是一阶定常迭代方法收敛的充要条件，而 $\|\mathbb{G}\| < 1$ 只是一阶定常迭代方法收敛的充分条件。

对于收敛算法，迭代误差的具体表现还同初值密切相关；因此，数值工作者常常采用**最差表现**来评判算法优劣。为讨论简便，不妨以一阶定常迭代 $\boldsymbol{x}_k = \mathbb{G}\boldsymbol{x}_{k-1} + \boldsymbol{g}$ 为例，其迭代误差满足不可改善的估计

$$\|\boldsymbol{e}_k\| \leq \|\mathbb{G}^k\|\|\boldsymbol{e}_0\|. \tag{2.1.4}$$

✍ **论题 2.2.** 基于 (2.1.4)，算法的收敛速度可以用迭代误差的平均下降速度来刻画。具体定义有

> *1.* 平均收敛速度 $\quad R_k(\mathbb{G}) = -\frac{1}{k}\ln\|\mathbb{G}^k\|$;
> *2.* 渐近收敛速度 $\quad R_\infty(\mathbb{G}) = \lim_{k\to\infty} R_k(\mathbb{G}) = -\ln\varrho(\mathbb{G})$.

上述概念均产生于上世纪五六十年代，特别是渐近收敛速度由 *Young* 在 *1954* 年给出，用于评判迭代算法的优劣程度，即：**迭代误差达到用户要求的最少迭代步数反比例于渐近收敛速度**。

❀ **思考 2.1.** 事实上，两个收敛速度概念均基于平均意义，并不是迭代误差变化的真实速度。为理解上述概念，考虑

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

选取 α 靠近 1，使 $\mathbb{A}$ 的谱半径略小于 $\mathbb{B}$。考察 $\|\mathbb{A}^m\|_2$ 和 $\|\mathbb{B}^m\|_2$ 的发展过程，特别是当 m 较小的时候会出现 $\|\mathbb{A}^m\|_2 > \|\mathbb{B}^m\|_2$ 吗？

这个实例也说明，迭代误差的初始表现和渐近表现可以不同。

❀ **思考 2.2.** 证明：对于任意范数均有 $\lim_{k\to\infty}\|\mathbb{G}^k\|^{1/k}=\varrho(\mathbb{G})$。

由于 $\|\mathbb{G}^k\|$ 和 $\varrho(\mathbb{G})$ 较难计算，$\|\mathbb{G}\|$ 更适宜迭代误差的界定。主要结论陈述为下面的定理。

定理 2.3. 若 $\|\mathbb{G}\|<1$，则迭代方法 $\boldsymbol{x}_k=\mathbb{G}\boldsymbol{x}_{k-1}+\boldsymbol{g}$ 收敛，且有

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

其中 $\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|$ 称为相邻误差。

✯ **说明 2.1.** 数值计算通常采用 l_1 范数或者 l_∞ 范数，而理论分析常常采用 l_2 范数。前者对应矩阵的行（或列）范数，比较容易计算；后者对应矩阵的谱范数，不太容易计算。

✯ **说明 2.2.** 在先验误差估计中，右端上界是无法计算的；在后验误差估计中，右端上界是可以计算的。请注意：这些都是保守估计，实际误差可能远远小于理论结果。

2.1.3 停机准则

只有提供适当的停机准则，迭代法才能真正求解实际问题。对于给定的用户指标 $\mathcal{E}>0$，最自然的想法是数值误差[1]达到

$$\|\boldsymbol{e}_k\|\le\mathcal{E}, \tag{2.1.5}$$

[1]这里是绝对误差，当然也可用相对误差。

其中 $\|\cdot\|$ 是某种范数。这个准则只能用于理论研究（或数值实验），不能实际应用。下面给出三个广泛应用的停机准则

1. 残量准则：$\|\boldsymbol{r}_k\| \leq \mathcal{E}$;
2. 相邻误差准则：$\delta_k \equiv \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\| \leq \mathcal{E}$;
3. 后验误差准则：$\delta_k^2/(\delta_{k-1} - \delta_k) \leq \mathcal{E}$.

强调指出：上述准则与 (2.1.5) 并不完全等价。特别地，第三个准则源于后验误差估计 (I) 和矩阵范数 $\|\mathbb{G}\|$ 的估算，相应的有效性还要受限于估算是否准确。

❀ **思考 2.3.** 注意 $\boldsymbol{r}_k$ 同 $\boldsymbol{e}_k$ 的关系 $\mathbb{A}\boldsymbol{e}_k = \boldsymbol{r}_k$，给出前两个停机准则的联系。

✯ **说明 2.3.** 当线性方程组高度病态，舍入误差会严重影响迭代法的计算效果。它可能破坏迭代法的收敛性，造成停机准则的误判。因篇幅有限，本讲义略去相关的理论分析，仅仅给出一个实例。

利用具有 6 位有效数字的十进制计算机，执行迭代算法

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

取 $\boldsymbol{x}_0 = (0.1, 0.1)^\top$，每步迭代仅有 10^{-6} 的分量变化，即 $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty = 10^{-6}$。若取用户指标 $\mathcal{E} = 10^{-5}$，并以相邻误差为停机标准，则导致“假停机”现象出现。

2.2 Jacobi/Gauss-Seidel 方法

作为著名的古典迭代算法，Jacobi（J）和 Gauss-Seidel（GS）方法均出现于计算机诞生之前：

1. J 方法由 Jacobi (1845) 提出。计算机时代的相关工作有 Geiringer (1945) 的同步位移法，以及 Killer (1958) 的 Richardson 方法。物理学家更喜欢称其为阻尼法。

2. GS 方法由 Gauss (1822) 提出，用于求解线性最小二乘问题导致的法方程组。Seidel (1874) 再次提出该方法，然后弃之不用。Von. Misers 和 Pollaczek-Geiringer (1949) 使其在计算机时代重获新生，并首次给出相应的理论分析。

两个算法都是无参数的一阶定常迭代，相应的理论分析也极具代表。

2.2.1 算法定义和矩阵分裂

算法实现方式相近，但数据更新策略不同：J 方法采用同步策略，而 GS 方法采用异步策略。具体的逐个分量更新方式如下：

1. J 方法：$\boldsymbol{x}_k^{(i)} = \dfrac{1}{a_{ii}}\Big[\boldsymbol{b}^{(i)} - \sum\limits_{j\neq i} a_{ij}\boldsymbol{x}_{k-1}^{(j)}\Big]$,
2. GS 方法：$\boldsymbol{x}_k^{(i)} = \dfrac{1}{a_{ii}}\Big[\boldsymbol{b}^{(i)} - \sum\limits_{j<i} a_{ij}\boldsymbol{x}_{k}^{(j)} - \sum\limits_{j>i} a_{ij}\boldsymbol{x}_{k-1}^{(j)}\Big]$.

换言之，在 J 方法中，分量更新没有次序，可以同时进行；但是，在 GS 方法中，分量更新具有次序，不同的次序会导致不同的结果。若无特殊申明，默认采用自然顺序。

关于迭代向量的数据存储，J 方法需要两组工作单元保存新旧位置，而 GS 方法利用数据覆盖技术，只需要一组工作单元。

迭代法大多基于**矩阵分裂**技术构造。设 $\mathbb{A} = \mathbb{Q} - \mathbb{R}$，其中 $\mathbb{Q}$ 逼近 $\mathbb{A}$ 且容易求逆，称为主体部分。基于同解的不动点方程 $\boldsymbol{x} = \mathbb{Q}^{-1}(\mathbb{R}\boldsymbol{x} + \boldsymbol{b})$，

定义相应的（不动点）迭代

$$\boldsymbol{x}_k = \mathbb{Q}^{-1}(\mathbb{R}\boldsymbol{x}_{k-1} + \boldsymbol{b}). \tag{2.2.6}$$

简单整理，可知 $\mathbb{Q}^{-1} = \mathbb{H}$，故 $\mathbb{Q}$ 也称为预处理矩阵。

✎ **论题 2.3.** 系数矩阵 $\mathbb{A}$ 可以表示为三个部分的和，即

$$\mathbb{A} = \mathbb{D} - \mathbb{D}\mathbb{L} - \mathbb{D}\mathbb{U}, \tag{2.2.7}$$

其中 $\mathbb{D}$ 是对角线部分，$-\mathbb{D}\mathbb{L}$ 是严格[2]上三角部分，$-\mathbb{D}\mathbb{U}$ 是严格下三角部分。本讲义将默认上述符号，不再重复说明。

以对角阵 $\mathbb{D}$ 为主体部分，矩阵分裂将导出 J 方法；以下三角阵 $\mathbb{D}-\mathbb{D}\mathbb{L}$ 为主体部分，矩阵分裂将导出 GS 方法。它们的迭代矩阵分别是

$$\mathbb{B} = \mathbb{I} - \mathbb{D}^{-1}\mathbb{A}, \quad \mathbb{T}_1 = (\mathbb{I} - \mathbb{L})^{-1}\mathbb{U}, \tag{2.2.8}$$

相应谱半径小于 1 是算法收敛的充要条件。

✯ **说明 2.4.** 迭代矩阵的特征值同代数方程的排序有关。考虑同解的两个线性方程组

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

计算 J 迭代矩阵的谱半径，并指出它们的 J 迭代是否收敛？这个问题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。

2.2.2 收敛分析与收敛速度

一般而言，J 方法和 GS 方法的收敛性没有明确的联系。具体实例参见教科书，此处不再赘述。就某些特殊情况，相关结论还是明确的，主要结果陈述如下。

[2]严格是指对角线元素也等于零。

以 J 方法的迭代矩阵为起点

定理 2.4. 若 $\|\mathbb{B}\|_\infty < 1$，则 *GS* 方法也收敛，且比 *J* 方法更快。

定理 2.5. 若 $\|\mathbb{B}\|_1 < 1$，则 *GS* 方法也收敛。

以系数矩阵为起点

定义 2.2. 称 $\mathbb{A} = (a_{ij})$ 弱对角占优，若

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|, \quad i = 1 : n,$$

且至少一个不等式是严格的。称 $\mathbb{A}$ 强对角占优，若所有不等式都是严格的。

定义 2.3. 称 $\mathbb{A} = (a_{ij})$ 可约，若指标集可以分割为两个非空子集 S_1 和 S_2，即 $S_1 \cup S_2 = \{1 : n\}$ 且 $S_1 \cap S_2 = \emptyset$，使得

$$a_{ij} = 0, \quad i \in S_1, j \in S_2.$$

若 S_1 和 S_2 一直找不到，则称 $\mathbb{A} = (a_{ij})$ 不可约。

无论矩阵 $\mathbb{A}$ 是强对角占优，或是弱对角占优且不可约，本讲义均统称其为对角占优。

定理 2.6. 若 $\mathbb{A}$ 对角占优，则 *J* 方法和 *GS* 方法均收敛。

定理 2.7. 当系数矩阵对称时，*GS* 方法比 *J* 方法的适用范围更广。相关结果是

1. 若 $\mathbb{A}$ 正定，则 *GS* 方法必定收敛；
2. 若 $2\mathbb{D} - \mathbb{A}$ 也正定，则 *J* 方法收敛；反之亦然。

✯ **说明 2.5.** 上述四个定理的证明过程充分地展示了迭代方法收敛性分析的两个基本技巧：其一是误差方程和**范数估计**，其二是迭代矩阵的**特征值估计**。具体内容见教科书。

✯ **说明 2.6.** 系数矩阵 $\mathbb{A}$ 的对角占优强度可用

$$\min_i \left[|a_{ii}| - \sum_{j \neq i} |a_{ij}| \right]$$

来衡量，但是它同收敛速度并无实质联系。例如，考虑

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

前者的对角占优性更强一些，但是 $\varrho(\mathbb{B}_1) > \varrho(\mathbb{B}_2)$，相应的 J 方法具有略慢的收敛速度。

2.3 逐次超松弛方法

在迭代方法的发展过程中，逐次超松弛（Successive Over-Relax = SOR）方法具有非常重要的历史地位。它成功地引进了松弛因子和加权平均的思想，相关讨论推进了迭代法的设计思路和理论研究，促进了迭代法在上世纪 60-80 年代的迅猛发展。

2.3.1 算法定义和收敛分析

SOR 方法以 GS 方法为基础算法，在逐个分量更新的过程中，对新旧两个信息进行加权平均：

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

其中 ω 称为松弛因子。相应的 SOR 迭代矩阵是

$$\mathbb{T}_\omega = (\mathbb{I} - \omega\mathbb{L})^{-1}[(1-\omega)\mathbb{I} + \omega\mathbb{U}]. \tag{2.3.9}$$

显然，当 $\omega = 1$，SOR 方法就是 GS 方法。

❀ **思考 2.4.** 指出 *SOR* 方法的矩阵分裂方式。

定理 2.8. *SOR* 方法收敛的必要条件是 $0 < \omega < 2$.

证明：利用行列式与特征值的关系即可。 □

定理 2.9. 若系数矩阵 $\mathbb{A}$ 对称正定，则 $0 < \omega < 2$ 是 *SOR* 方法收敛的充分必要条件。

证明：特征值估计的典型实例。见教科书。 □

2.3.2 最佳松弛因子

数值表明：对于某些类型的系数矩阵，SOR 方法有最佳松弛因子，使收敛速度获得显著提升。它唤起了数值工作者的研究热情。

✎ **论题 2.4.** 相关讨论同系数矩阵的非零元素分布相关。本讲义重点关注“相容次序”和“性质 $\boldsymbol{A}$”，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 $\boldsymbol{A}$。
3. 具有性质 $\boldsymbol{A}$ 的矩阵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。

详细内容见教科书。

对于二维椭圆型偏微分方程，五点差分格式离散而成的差分方程可以按照不同方式进行堆积，形成不同样式的线性方程组。无论怎样操作，线性方程组 $\mathbb{A}\boldsymbol{x}=\boldsymbol{b}$ 都满足以下现象：未知变量归属于两个互不相交的集合，属于相同集合的变量没有关联[3]。此时，称系数矩阵 $\mathbb{A}$ 具有**性质 A**。适当（同时）改变变量编号和方程次序，使得

$$\mathbb{P}\mathbb{A}\mathbb{P}^{\top}=\begin{bmatrix}\mathbb{D}_1 & \mathbb{H}\\ \mathbb{K} & \mathbb{D}_2\end{bmatrix}, \tag{2.3.10}$$

其中 $\mathbb{D}_1$ 和 $\mathbb{D}_2$ 均是对角阵，$\mathbb{P}$ 是置换阵。事实上，若未知变量按照红黑（或棋盘）原则编号，则系数矩阵直接呈现出 (2.3.10) 的右侧结构。

分别用 λ 和 μ 表示 SOR 迭代矩阵 $\mathbb{T}_\omega$ 和 J 迭代矩阵 $\mathbb{B}$ 的特征值。当性质 A（甚至相容次序）成立时，λ 和 μ 具有紧密联系。

定理 2.10. *J 方法和 SOR 方法的特征值具有对应关系：*

$$(\lambda+\omega-1)^2=\lambda\omega^2\mu^2. \tag{2.3.11}$$

特别地，非零的 μ 和 $-\mu$ 是成对出现的。

证明：理论证明主要源于基本事实：**若矩阵具有相容次序，当严格上三角和严格下三角的元素分别乘以互为倒数的两个数值时，行列式保持不变**。为简单起见，不妨直接考虑 (2.3.10) 右侧的矩阵结构，利用分块矩阵技术直接验证：

$$\begin{bmatrix}\mathbb{I} & \\ & \alpha^{-1}\mathbb{I}\end{bmatrix}\begin{bmatrix}\mathbb{D}_1 & \alpha^{-1}\mathbb{H}\\ \alpha\mathbb{K} & \mathbb{D}_2\end{bmatrix}\begin{bmatrix}\mathbb{I} & \\ & \alpha\mathbb{I}\end{bmatrix}=\begin{bmatrix}\mathbb{D}_1 & \mathbb{H}\\ \mathbb{K} & \mathbb{D}_2\end{bmatrix}. \tag{2.3.12}$$

基于相容次序的证明，可参阅教科书。最后计算两个迭代矩阵的特征值，即可证明此定理。详见教科书。 □

[3]两个未知量同时出现在一个差分（或线性代数）方程中，则称它们是关联的。

定理 2.11. 设 $\mu_j = \alpha_j + \sqrt{-1}\beta_j$，其中 α_j 和 β_j 均为实数。若存在正数 D，使得（换言之，特征值落在一个椭圆内）

$$\boxed{\alpha_j^2 + \beta_j^2/D < 1, \quad j = 1:n,}$$

则当 $0 < \omega < 2/(1+D)$ 时，*SOR* 迭代方法是收敛的。

特别地，若 μ_j 均为实数（对应 $D = 0$），则 *SOR* 迭代方法收敛的充分必要条件是

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

证明：本讲义仅关注实特征值的情形。该定理的证明要用到 (2.3.11) 和基本结论：**对于实系数二次方程 $z^2 + bz + c = 0$，两根按模均小于一的充要条件是 $|b| < 1 + c < 2$。** □

✍ **论题 2.5.** 设 $\mathbb{B}$ 的特征值均为实数且 $\mu \in (-1, 1)$，给出最佳松弛因子的计算公式。

固定特征值 $\mu > 0$。在实平面 (λ, y) 上，考虑直线同抛物线的交点：

$$y = \frac{\lambda + \omega - 1}{\omega}, \quad y^2 = \lambda |\mu|^2,$$

其中 ω 为参数。假设交点存在[4]，横坐标 $\lambda_1(\omega)$ 和 $\lambda_2(\omega)$ 均为实数，满足二次方程 (2.3.11)，是 $\mathbb{T}_\omega$ 的特征值。要使 $\max(|\lambda_1(\omega)|, |\lambda_2(\omega)|)$ 达到最小，直线同抛物线必须相切，即二次方程 (2.3.11) 的判别式为零。简单计算可知，对应给定 μ 的最佳参数设置是

$$\omega = \frac{2}{1 + \sqrt{1 - \mu^2}} > 1, \tag{2.3.13}$$

[4]若直线和抛物线不相交，由定理 2.10 可知 (2.3.11) 存在共轭复根，满足 $|\lambda_i(\omega)| \equiv |\omega - 1|$，后面的讨论不受影响。

相应的切点横坐标是 $\max(|\lambda_1(\omega)|, |\lambda_2(\omega)|) = |\omega - 1|$。

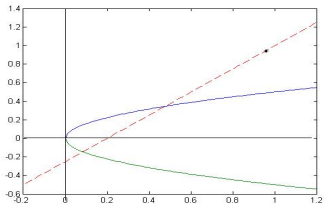


图 2.3.1: 直线与抛物线的交点

由 (2.3.13) 可知，当 $|\mu|$ 变大时，相应的 ω 随之增加。此时，抛物线开口变宽，切点位置右移，且切线以 $(1,1)$ 为中心逆时针旋转，远离窄口抛物线。因此，当 $|\mu|$ 最终增加到 $\rho(\mathbb{B})$ 时，相应的切点位置给出最佳松弛因子

$$\boxed{\omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - \varrho^2(\mathbb{B})}},}$$

相应的迭代矩阵 $\mathbb{T}_{\omega_{\text{opt}}}$ 具有最小的谱半径

$$|\omega_{\text{opt}} - 1| = \frac{1 - \sqrt{1 - \varrho(\mathbb{B})^2}}{1 + \sqrt{1 - \varrho(\mathbb{B})^2}}. \tag{2.3.14}$$

✯ **说明 2.7.** 事实上，ω_{opt} 是谱半径函数 $f(\omega) \equiv \varrho(\mathbb{T}_\omega)$ 的不可微点；参见插图 *2.3.2*。具体表达式可参见教科书。谱半径函数在不可微点的左导数为无穷大，故而在实际计算时通常偏大选取松弛因子。

✯ **说明 2.8.** 最佳松弛因子强烈依赖谱半径 $\rho(\mathbb{B})$，在实际应用时很难给出。相对折衷的策略是不断修正：先取偏大的 $\omega \in (0, 2)$，然后

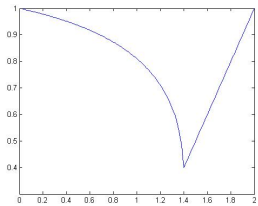


图 2.3.2: SOR 方法的谱半径函数 $\varrho(\mathbb{T}_\omega)$

- 执行 *SOR* 迭代，利用幂法（若成功的话）直至稳定取值

$$\rho \equiv \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_\infty}{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty},$$

  给出谱半径 $\rho(\mathbb{T}_w)$ 的合理近似；

- 利用公式 (2.3.11) 估计谱半径 $\rho(\mathbb{B})$，即

$$\rho(\mathbb{B}) \approx \mu \equiv \frac{\rho + \omega - 1}{\omega\sqrt{\rho}}.$$

  然后，将其代入最佳松弛因子的计算公式，给出新的 ω。

重复执行上述过程，ω 可能逐渐靠近 ω_{opt}。

✍ **论题 2.6.** 带有最佳松弛因子的 *SOR* 方法可以获得显著提升的收敛速度，达到用户要求的最少迭代步数达到开根号级别的改善。

教科书给出了一般性结果。为简单起见，不妨以线性方程组 (6.0.3) 为例进行直观展示。此时，系数矩阵可以用 Kronecker 积[5]表示为

$$\mathbb{A}_n = \mathbb{T}_n \otimes \mathbb{I}_n + \mathbb{I}_n \otimes \mathbb{T}_n,$$

[5]设 $\mathbb{A} = (a_{ij})$ 是 m 阶方阵，$\mathbb{B}$ 是 n 阶方阵，则 $\mathbb{A} \otimes \mathbb{B} = (a_{ij}\mathbb{B})$ 是 mn 阶方阵。

其中 $\mathbb{T}_n$ 是三对角阵，$\mathbb{I}_n$ 是单位阵。$\mathbb{T}_n$ 的特征信息可以精确给出，即

$$\lambda_\kappa = 2\left(1-\cos\frac{\kappa\pi}{n+1}\right),$$
$$\mathbf{v}_\kappa = \sqrt{\frac{2}{n+1}}\left(\sin\frac{\kappa\pi}{n+1}, \sin\frac{2\kappa\pi}{n+1}, \cdots, \sin\frac{n\kappa\pi}{n+1}\right)^\top,$$

其中 $\kappa = 1:n$。由 Kronecker 积的运算性质，可知 $\mathbb{A}_n$ 具有特征值

$$\lambda_{pq} = \lambda_p + \lambda_q, \quad p,q = 1:n.$$

注意到 $\mathbb{A}$ 的对角元素恒为 4，可知 $\mathbb{B}$ 的特征值是

$$\mu_{pq} = \frac{1}{4}(\lambda_p + \lambda_q - 4), \quad p,q = 1:n.$$

简单计算可知，三种迭代方法分别具有渐近收敛速度

$$R_\infty(\mathbb{B}) = -\ln\varrho(\mathbb{B}) = -\ln\cos h\pi \sim \frac{1}{2}\pi^2h^2, \tag{2.3.15a}$$
$$R_\infty(\mathbb{T}_1) = -2\ln\varrho(\mathbb{B}) \sim \pi^2h^2, \tag{2.3.15b}$$
$$R_\infty(\mathbb{T}_{\text{opt}}) = -\ln\frac{1-\sin(h\pi)}{1+\sin(h\pi)} \sim 2h\pi, \tag{2.3.15c}$$

其中 $h = 1/(n+1)$。该结果由 Franke 最早给出，由 Young 推广到一般情形。

2.4 迭代加速方法

关于 SOR 方法的研究表明：对基础算法

$$\boldsymbol{x}_k = \mathbb{G}\boldsymbol{x}_{k-1} + \boldsymbol{g} \tag{2.4.16}$$

的计算结果进行适当的平均化处理，可以改善收敛速度。最简单的处理是以给定的权重 γ 平均相邻的两个迭代位置，可得**外推方法**

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

❀ **思考 2.5.** 假设 $\mathbb{G}$ 是实对称阵。确定最优权重参数 γ，使外推方法的迭代矩阵 $\gamma\mathbb{G} + (1-\gamma)\mathbb{I}$ 谱半径最小。

半迭代方法将外推思想极致化：加权平均基础算法 (2.4.16) 的全部结果 $\{\boldsymbol{x}_k\}_{k=0}^m$，定义修正序列

$$\boldsymbol{y}_m = \sum_{k=0}^{m} \alpha_{m,k}\boldsymbol{x}_k, \tag{2.4.17}$$

其中 $\alpha_{m,k}$ 是待定参数，满足相容性条件

$$\sum_{k=0}^{m} \alpha_{m,k} = 1. \tag{2.4.18}$$

记 $\boldsymbol{\eta}_m = \boldsymbol{y}_m - \boldsymbol{x}_\star$ 为修正序列的误差，它满足误差方程

$$\boldsymbol{\eta}_m = \sum_{k=0}^{m} \alpha_{m,k}\mathbb{G}^k\boldsymbol{e}_0 = P_m(\mathbb{G})\boldsymbol{e}_0, \tag{2.4.19}$$

其中 $\boldsymbol{e}_0 = \boldsymbol{\eta}_0 = \boldsymbol{x}_0 - \boldsymbol{x}_\star$ 为初始误差，

$$P_m(\lambda) = \sum_{k=0}^{m} \alpha_{m,k}\lambda^k$$

是满足 $P_m(1) = 1$ (系数和为一) 的一个 m 次多项式。误差方程 (2.4.19) 蕴含了迭代方法的构造思想，半迭代方法将格式设计从“单项式算法”框架拓展到“多项式算法”框架。

半迭代方法成功与否的关键是能否实现目标：让修正序列 $\boldsymbol{y}_m$ 更快地收敛到精确解。为此，需确定最佳参数组 $\{\alpha_{m,k}\}_{k=0:m}$，使修正迭代矩阵 $P_m(\mathbb{G})$ 的谱半径最小。

2.4.1 变系数 Richardson 方法

在非定常算法的最佳实现中，半迭代的加速思想已经隐含实现了。典型实例有非定常 Richardson（1910）方法

$$\boldsymbol{y}_k = \boldsymbol{y}_{k-1} + \tau_k(\boldsymbol{b} - \mathbb{A}\boldsymbol{y}_{k-1}), \tag{2.4.20}$$

其中 τ_k 是适当选取的参数。若 $\tau_k \equiv \tau$ 保持不变，则 (2.4.20) 称为定常 Richardson 方法，以示区别，记其为 $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \tau(\boldsymbol{b} - \mathbb{A}\boldsymbol{x}_{k-1})$。

✍ **论题 2.7.** 非定常 R 方法可视为定常 R 方法的半迭代加速。

证明：建立迭代矩阵的关系，找到相应的半迭代多项式。 □

下面讨论加速效果。为简单起见，假设系数矩阵 $\mathbb{A}$ 实对称正定，其最大和最小两个特征值分别是 $\lambda_{\max}$ 和 $\lambda_{\min}$。

❀ **思考 2.6.** 对于定常 R 方法，最佳参数是 $\tau = 2/(\lambda_{\max} + \lambda_{\min})$，相应的收敛速度是

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

达到用户要求的最少迭代步数同 $\kappa(\mathbb{A})$ 成正比。

✍ **论题 2.8.** 给定总体迭代步数 m。找到最佳参数组 $\{\tau_k^\star\}_{k=1}^m$，使非定常 R 方法的误差下降（也称为收敛速度）最快？

由于对称矩阵的谱范数就是特征值的最大模，因此有

$$\frac{\|\boldsymbol{e}_m\|_2}{\|\boldsymbol{e}_0\|_2} \leq \max_{\lambda_i \in \lambda(\mathbb{A})} \left| \prod_{k=1}^m (1 - \tau_k \lambda_i) \right|.$$

要使右端的上界尽量小，可以考虑（略有放大的）Cheybeshev 极大极小问题

$$\{\tau_k^\star\}_{k=1}^m = \arg \min_{\{\tau_k\}_{k=1}^m} \max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} \left| \prod_{k=1}^m (1 - \tau_k \lambda) \right|.$$

由最佳一致逼近理论可知，对应最佳参数的多项式应当是

$$\prod_{k=1}^{m}(1-\tau_k^\star\lambda)=P_m^\star(\lambda)=\frac{T_m(\ell(\lambda))}{T_m(\ell(0))}, \tag{2.4.21}$$

其中 $T_m(z)$ 是 m 次标准 Cheybeshev 多项式，

$$\ell(\lambda)=\frac{2\lambda-\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}} \tag{2.4.22}$$

是 $[\lambda_{\min},\lambda_{\max}]\to[-1,1]$ 的仿射变换。注意到 $\{(\tau_k^\star)^{-1}\}_{k=1:m}$ 是 $P_m^\star(\lambda)$ 的零点，以及

$$\begin{aligned}T_m(z)=&\ \cosh(m\cosh^{-1}z)\\ =&\begin{cases}\dfrac{1}{2}\Big[(z+\sqrt{z^2-1})^m+(z-\sqrt{z^2-1})^m\Big], & |z|\geq 1;\\ \cos(m\arccos z), & |z|\leq 1.\end{cases}\end{aligned}$$

最佳参数设置是

$$\boxed{\tau_k^\star=\left[\frac{\lambda_{\max}-\lambda_{\min}}{2}\cos\left(\tfrac{2k-1}{2m}\pi\right)+\frac{\lambda_{\max}+\lambda_{\min}}{2}\right]^{-1}.}$$

利用 Cheybeshev 多项式的性质

$$T_m\left(\frac{1+r^2}{1-r^2}\right)=\frac{1}{2}\left[\left(\frac{1+r}{1-r}\right)^m+\left(\frac{1-r}{1+r}\right)^m\right],\quad r\in(-1,1),$$

联立 (2.4.21) 可知变系数 R 方法具有估计

$$\frac{\|\boldsymbol{e}_m\|_2}{\|\boldsymbol{e}_0\|_2}\leq 2\left(\frac{\sqrt{\kappa(\mathbb{A})}-1}{\sqrt{\kappa(\mathbb{A})}+1}\right)^m, \tag{2.4.23}$$

其中 $\kappa(\mathbb{A})=\lambda_{\max}/\lambda_{\min}$ 为对称正定矩阵的谱条件数。换言之，达到用户要求的最少迭代步数仅仅同 $\sqrt{\kappa(\mathbb{A})}$ 成正比，有着本质性的改善。

✯ **说明 2.9.** 着重指出，最优参数 $\{\tau_k^\star\}_{k=1:m}$ 的设置同 m 相关。在执行非定常 R 方法之前，首先用 (2.4.23) 估算出 m，然后才能设置最优参数。对于高度病态（即 $\lambda_{\min} \ll 1$ 时）的问题，当 m 很大时，最佳参数的数值计算常常会因为舍入误差的影响而产生严重偏差，使得实际的收敛速度大打折扣，甚至迭代误差出现反弹。常用的解决方法是采用小 m 和循环（或重启）策略。

2.4.2 Cheybeshev 半迭代加速

✎ **论题 2.9.** 设基础方法 (2.4.16) 的迭代矩阵 $\mathbb{G}$ 是实对称的，最大和最小特征值分别是 $\lambda_{\max}$ 和 $\lambda_{\min}$。此时，半迭代方法 (2.4.17) 怎样设置最佳参数，相应的收敛表现是什么？

设 $\mathbb{G}$ 的特征值是 $\{\lambda_i\}_{i=1}^n$，相应的单位特征向量 $\{\boldsymbol{\xi}_i\}_{k=1}^n$ 彼此正交。初始误差可以线性表示为

$$\boldsymbol{\eta}_0 = \boldsymbol{e}_0 = \sum_{1\le i\le n} \beta_i \boldsymbol{\xi}_i,$$

由 (2.4.19) 可知，半迭代方法 (2.4.17) 的误差满足

$$\boldsymbol{\eta}_k = \sum_{i=1}^n \Big[\sum_{\ell=0}^k \alpha_{k,\ell}\lambda_i^\ell\Big]\beta_i\boldsymbol{\xi}_i = \sum_{i=1}^n Q_k(\lambda_i)\beta_i\boldsymbol{\xi}_i.$$

要使 $\|\boldsymbol{\eta}_k\|_2/\|\boldsymbol{\eta}_0\|_2$ 尽可能小，可如前考虑 Chebyshev 极大极小问题

$$Q_k^\star(\lambda) = \arg\min_{Q_k\in\mathbb{P}_k^\sharp} \max_{\lambda\in[\lambda_{\min},\lambda_{\max}]} |Q_k(\lambda)|,$$

其中 $\mathbb{P}_k^\sharp$ 表示系数和为一的 k 次多项式全体。答案是归一化[6]的 Chebyshev 多项式，即

[6]此处，假设了 $\mathbb{G}$ 的特征值均小于 1；其它情形也可处理，但过程较繁，略。

$$\boxed{Q_k^\star(\lambda) = \frac{T_k\left(\ell(\lambda)\right)}{T_k\left(\ell(1)\right)},}$$

其中 $T_k(z)$ 是 k 次 Chebyshev 多项式，$\ell(\lambda)\colon [\lambda_{\min}, \lambda_{\max}] \to [-1, 1]$ 是仿射变换，其定义方式与 (2.4.22) 相同，即

$$\ell(\lambda) = \frac{2\lambda - \lambda_{\max} - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}.$$

此时，最佳参数 $\{\alpha_{k,\ell}\}_{\ell=0}^{k}$ 就是 $Q_k^\star(\lambda)$ 的系数，相应的收敛速度获得显著提升。

☆ **说明 2.10.** 上述讨论可以推广到复特征值的情形，最佳参数的设定与复特征值所属的椭圆区域有关。具体内容超出课程范围，略。

❀ **思考 2.7.** 度量 *Cheybeshev* 半迭代算法的收敛速度，并考察最佳参数给予的收敛速度效果。

✎ **论题 2.10.** 半迭代方法 (2.4.17) 存在历史数据的存储困境以及最佳参数的计算和存储困境，不能直接应用于实际计算。

Chebyshev 多项式是正交系，具有三项递推关系式

$$T_{n+1}(z) = 2zT_n(z) - T_{n-1}(z), \tag{2.4.24}$$

其中 $T_0(z) = 1$ 和 $T_1(z) = z$。利用误差方程 (2.4.19) 进行反向推导，半迭代方法 (2.4.17) 可以等价变形为非定常二阶迭代方法：

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

固定参数是

$$\nu = \frac{2}{2-\lambda_{\max}-\lambda_{\min}}, \quad \xi = \ell(1) = \frac{2-\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}, \tag{2.4.25}$$

变化参数是

$$\rho_{k+1} = \frac{2\xi T_k(\xi)}{T_{k+1}(\xi)}. \tag{2.4.26}$$

利用 (2.4.24)，可以建立 ρ_{k+1} 的递归计算公式

$$\boxed{\rho_{k+1} = \Big[1 - \frac{1}{4\xi^2}\rho_k\Big]^{-1}, \text{ 其中 } \rho_1 = 2.}$$

方法的启动方式是：任取 $\boldsymbol{y}_0$，利用基础迭代给出 $\boldsymbol{y}_1 = \mathbb{G}\boldsymbol{y}_0 + \boldsymbol{g}$。

❀ **思考 2.8.** *设系数矩阵 $\mathbb{A}$ 具有性质 $\boldsymbol{A}$，或者简单理解为 (2.3.11) 右端定义的特殊矩阵。对 J 方法进行半迭代加速，考察 ρ_k 的收敛性，它同 SOR 方法的最佳松弛因子有何关联？*

具有最佳参数的半迭代方法的收敛速度可以获得显著提升，但是最佳参数和最佳收敛速度均强烈依赖原始迭代矩阵（或系数矩阵）的特征值，至少是特征值的准确分布范围。由于特征值比线性方程组更难数值计算，上述原因极大限制了半迭代方法的应用范围和实际效果。我们期待一种快速收敛且不显式依赖特征值信息的数值方法。

2.5 共轭斜量法

共轭斜量（Conjurate Gradient = CG）法是求解**实对称正定方程组**的首选数值方法，由 Hestenes 和 Stiefel（1950）提出。核心思想包含两点，其一是将线性方程组转化为等价的二次函数极值问题，其二是采用

合适的优化算法快速解出二次函数的极值点。CG 法兼具直接法和迭代法（Reid,1971）的特性，可以利用有限次运算求得精确解，也可利用递归生成的序列逼近精确解。对于规模庞大的线性方程组，CG 法常常被视为一种无参数迭代算法，相应的执行过程无需知晓系数矩阵的任何特征值信息。

2.5.1 函数极值问题

✎ **论题 2.11.** 在共轭斜量法的诸多引进方式之中，较为直观的方式是将对称正定线性方程组 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ 转化为等价的二次函数（椭圆抛物面）优化问题：

$$\boxed{\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}),}$$

其中 $f(\boldsymbol{x})$ 是整个离散系统的总能量，即

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top\mathbb{A}\boldsymbol{x} - \boldsymbol{b}^\top\boldsymbol{x}. \tag{2.5.27}$$

定理 2.12. 优化问题的解就是线性方程组的解。

很多优化算法都可以求出目标函数 $f(\boldsymbol{x})$ 的极值点，实现策略大多都基于简单的一维搜索：从当前位置 $\boldsymbol{x}_k$ 出发，沿搜索方向 $\boldsymbol{p}_k$ 确定新的最优位置 $\boldsymbol{x}_{k+1}$，使得

$$\boldsymbol{x}_{k+1} = \arg\min_{\alpha\in\mathbb{R}} f(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k).$$

简单计算，可知

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k \quad \alpha_k = -\frac{\boldsymbol{r}_k^\top\boldsymbol{p}_k}{\boldsymbol{p}_k^\top\mathbb{A}\boldsymbol{p}_k}, \tag{2.5.28}$$

其中 $\boldsymbol{r}_k = \mathbb{A}\boldsymbol{x}_k - \boldsymbol{b}$ 是当前残量。不断执行一维搜索，即可给出所谓的**一维搜索算法**。

论题 2.12. 一维搜索算法的迭代序列满足

$$\boldsymbol{x}_k \in \pi_k \equiv \boldsymbol{x}_0 + \mathcal{L}_k, \quad k = 1, 2, \dots \tag{2.5.29}$$

其中 $\boldsymbol{x}_0$ 是初始位置，$\mathcal{L}_k$ 是历史搜索方向张成的**搜索空间**，即

$$\mathcal{L}_k = span\{\boldsymbol{p}_0, \boldsymbol{p}_1, \dots, \boldsymbol{p}_{k-1}\}. \tag{2.5.30}$$

若无特别申明，均假设搜索空间非退化，即 $\dim \mathcal{L}_k = k$。

引理 2.1. 称当前位置 $\boldsymbol{x}_k$ 关于搜索空间 $\mathcal{L}_k$ 最优，若

$$\boldsymbol{x}_k = \arg\min_{\boldsymbol{x}\in\pi_k} f(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}\in\mathcal{L}_k} f(\boldsymbol{x}_0 + \boldsymbol{z}). \tag{2.5.31}$$

相应的充要条件是当前残量与搜索空间正交，即 $\boldsymbol{r}_k \perp \mathcal{L}_k$ 或等价的

$$\boxed{\boldsymbol{r}_k^\top \boldsymbol{p}_\ell = 0, \quad \ell = 0 : k-1.}$$

设想一个超级完美的情况：在 k 逐渐增大的过程中，当前位置 $\boldsymbol{x}_k$ 关于搜索空间 $\mathcal{L}_k$ 的最优性质一**直保持**。由引理 2.1 可知，当搜索空间最终扩张到全空间 $\mathbb{R}^n$ 时，残量必定为零，即一维搜索算法可以在有限步内给出精确解。

2.5.2 共轭斜量方法的框架

一维搜索算法的核心是搜索方向的设置方式。在**最速下降法**中，搜索方向定义为当前位置的最速下降方向

$$\boldsymbol{p}_k = -\mathrm{grad} f(\boldsymbol{x}_k) = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}_k = -\boldsymbol{r}_k. \tag{2.5.32}$$

可以轻松证明：对于二次目标函数 (2.5.27)，最速下降法一定收敛。但是，当系数矩阵 $\mathbb{A}$ 高度病态（椭圆截面明显各向异性）时，其收敛表现可能极其糟糕，呈现出缓慢的“盘旋收敛”现象。

❀ **思考 2.9.** 证明最速下降法（按欧氏范数）收敛，并估计相应的收敛速度。

✎ **论题 2.13.** “盘旋收敛”可以归结为迭代序列关于搜索空间的最优性质没有保持住，或者引理 *2.1* 的充要条件没有实现到位。在最速下降法连续执行两步之后，有

$$\boldsymbol{x}_{k+2} \in \boldsymbol{x}_k + span(\boldsymbol{r}_k, \boldsymbol{r}_{k+1}).$$

它显然关于搜索方向 $\boldsymbol{p}_{k+1} = -\boldsymbol{r}_{k+1}$ 最优，但是

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

即 $\boldsymbol{x}_{k+2}$ 关于搜索方向 $\boldsymbol{p}_k = -\boldsymbol{r}_k$ 不是最优。

因此说，迭代序列关于搜索空间保持最优是必要的。不妨从必要条件入手。设 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k\boldsymbol{q}$ 是一维搜索位置，关于搜索方向 $\boldsymbol{q}$ 是最优的。若还要关于其它搜索方向 $\boldsymbol{p}$ 也是最优的，应有

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

换言之，$\boldsymbol{p}$ 和 $\boldsymbol{q}$ 是 $\mathbb{A}$-共轭的。上述讨论催生出下面的概念。

✇ **定义 2.4.** 称 $\{\boldsymbol{p}_\kappa\}_{\kappa=0}^m$ 是共轭向量系，若

$$\boldsymbol{p}_i^\top \mathbb{A}\boldsymbol{p}_j = 0, \quad i \neq j. \tag{2.5.33}$$

若搜索方向均源于共轭向量系，则一维搜索算法称为共轭斜量法。

定理 2.13. 在共轭斜量法中，当前位置 $\boldsymbol{x}_k$ 关于搜索空间 $\mathcal{L}_k$ 一直保持最优，其中 $k \leq m$。

共轭向量系必然线性无关，其张成的搜索空间是 $m+1$ 维；稍后说明：$m+1$ 可以足够大，直至全空间 $\mathbb{R}^n$ 的维数。由定理 2.13 可知：若所有计算过程都精确无误，则共轭斜量法至多 n 步即可达到精确解。换言之，共轭斜量法是直接法。

2.5.3 共轭斜量系的构造过程

共轭斜量法的设计关键是共轭向量系的构造。常见的构造方法是下面论题给出的局部递推方式。

论题 2.14. 局部递推方式是：在 $\boldsymbol{r}_{k+1}$ 和 $\boldsymbol{p}_k$ 张成的二维平面上确定搜索方向 $\boldsymbol{p}_{k+1}$，使其 $\mathbb{A}$-共轭于 $\boldsymbol{p}_k$。

相应的算法定义如下：任取 $\boldsymbol{x}_0$，令 $\boldsymbol{p}_0 = -\boldsymbol{r}_0 = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}_0$；对 $k \geq 0$，依次执行循环

$$
\begin{aligned}
&\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k, \quad \alpha_k = -\frac{\boldsymbol{r}_k^{\top}\boldsymbol{p}_k}{\boldsymbol{p}_k^{\top}\mathbb{A}\boldsymbol{p}_k};\\
&\boldsymbol{r}_{k+1} = \boldsymbol{r}_k + \alpha_k \mathbb{A}\boldsymbol{p}_k;\\
&\boldsymbol{p}_{k+1} = -\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k, \quad \beta_k = \frac{\boldsymbol{r}_{k+1}^{\top}\mathbb{A}\boldsymbol{p}_k}{\boldsymbol{p}_k^{\top}\mathbb{A}\boldsymbol{p}_k}.
\end{aligned}
$$

整个计算过程包含大量的内积运算，具有 *BLAS-2* 机制和内在并行特征，特别适合向量机上的数值计算。

要说明上述算法是共轭斜量（CG）法，还需给出深入的讨论，特别是共轭向量系是否达到预期目标。

论题 2.15. 利用数学归纳法，证明：在到达精确解（即残量为零）之前，上述算法给出了彼此等价的三个空间

$$\begin{aligned} span\{\boldsymbol{r}_0, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_k\} &= span\{\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k\} \\ &= span\{\boldsymbol{r}_0, \mathbb{A}\boldsymbol{r}_0, \ldots, \mathbb{A}^k\boldsymbol{r}_0\}, \end{aligned}$$

且所有搜索方向构成了一个共轭向量系。因此说，论题 *2.14* 给出的算法确实是共轭斜量法。

图文框内的三个空间分别称为残量空间、搜索空间和 Krylov 子空间。特别地，Krylov 子空间广泛应用于数值代数不同领域。

论题 2.16. 在共轭斜量法中，残量方向满足正交关系

$$\boldsymbol{r}_i^\top \boldsymbol{r}_j = 0, \quad \forall\, i \neq j.$$

再次指出：残量仅仅是椭圆抛物面 $z = f(\boldsymbol{x})$ 的法方向，没有快速地指向（各向异性）椭圆抛物面的顶点。

论题 2.17. 在共轭斜量法中，搜索方向和残量方向满足关系

$$\boldsymbol{r}_i^\top \boldsymbol{p}_j = \begin{cases} -\boldsymbol{r}_j^\top \boldsymbol{r}_j, & i \leq j; \\ 0, & i \geq j+1. \end{cases}$$

利用这个性质，共轭斜量法的参数计算可以简化为

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

由于内积运算在相邻两步出现重复，每步迭代可以节省 *1* 次内积运算，整体的计算复杂度有所改善。

❀ **思考 2.10.** 若忽视 α_k 和 β_k 的计算，将它们看作事先设定的参数，则 *CG* 算法可视为一个二阶非定常迭代。请写出相应的计算公式。

2.5.4 收敛性分析

共轭斜量法也可以看成迭代算法，并就迭代误差 $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}_\star$ 开展相应的收敛性分析。

定理 2.14. 迭代误差的 l_2 范数单调下降。

证明：既然精确解可以有限步到达，迭代误差可用搜索方向线性表示。计算迭代误差的 l_2 范数，利用搜索方向和残量方向的关系即可证明此定理。 □

✎ **论题 2.18.** 简单计算可知，第 k 步**能量误差**是

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}_\star) = \frac{1}{2}\boldsymbol{e}_k^\top \mathbb{A}\boldsymbol{e}_k = \frac{1}{2}\|\boldsymbol{e}_k\|_{\mathbb{A}}^2.$$

注意到 *CG* 算法是一种多项式迭代算法，利用定理 *2.13* 可知，能量误差满足极小问题[7]

$$\boxed{\|\boldsymbol{e}_{k+1}\|_{\mathbb{A}}^2 = \min_{Q_k \in \mathbb{P}_k} \boldsymbol{e}_0^\top \left\{ \mathbb{A}\Big[\mathbb{I} + \mathbb{A}Q_k(\mathbb{A})\Big]^2 \right\} \boldsymbol{e}_0.}$$

利用 $\mathbb{A}$ 的特征值信息，它蕴含两个非常重要的两个结论。

定理 2.15. 若系数矩阵 $\mathbb{A}$ 只有 m 个互异特征值，则 *CG* 算法至多 m 步到达精确解。

[7] 如前，$\mathbb{P}_k$ 依旧指 k 次多项式全体。

定理 2.16. *CG* 算法满足误差估计

$$\boxed{\frac{\|\boldsymbol{e}_k\|_{\mathbb{A}}}{\|\boldsymbol{e}_0\|_{\mathbb{A}}} \leq 2\left(\frac{\sqrt{\kappa_2(\mathbb{A})}-1}{\sqrt{\kappa_2(\mathbb{A})}+1}\right)^k,}$$

其达到用户要求的最少迭代步数正比例于 $\sqrt{\kappa_2(\mathbb{A})}$，类似于带最优参数的半迭代方法。

上述两个定理暗示：CG 算法的收敛速度与 $\mathbb{A}$ 的特征值聚集情况有关。当特征值扎堆出现时，收敛速度极高，甚至呈现 “超线性收敛”。

❀ **思考 2.11.** 设 $\mathbb{A}$ 的特征值聚集在两个区间上，即 s 个特征值落在 $[a_1, b_1]$，其它 $n-s$ 个特征值落在 $[a_2, b_2]$。请给出相应的收敛速度估计，充分论证上面的观点。

当问题高度病态时，CG 算法也会因舍入误差而变差。此时，建议不要将其看成直接法，利用迭代过程改善准确程度。同其他方法相比，CG 算法给出的计算结果较为可信。

✯ **说明 2.11.** 目前，*CG* 算法的思想已经推广到非对称问题或非正定问题，分别以论题 *2.15* 中的三个向量组为主体，形成 *Galerkin* 方法或者 *Krylov* 子空间投影方法。代表性工作有 *GMRES* 方法、双稳定化的 *CG* 方法、或者平方 *CG* 方法等等。均收录在 *Matlab* 中；具体内容请参阅相关文献。

2.5.5 预处理共轭斜量方法

预处理技术应用广泛，通过改善问题的性态，提高数值计算的效率或者准度。事实上，按比例选主元的 Gauss 消元法就暗含了此技术。本节以 CG 算法为例，介绍其实现过程。

论题 2.19. 引进预处理矩阵 $\mathbb{Q}=\mathbb{C}\mathbb{C}^{\top}$, 同解线性方程组

$$\mathbb{C}^{-1}\mathbb{A}\mathbb{C}^{-\top}\mathbb{C}^{\top}\boldsymbol{x}=\mathbb{C}^{-1}\boldsymbol{b} \tag{2.5.34}$$

的 *CG* 算法称为预处理共轭斜量（*PCG*）法，在 *Matlab* 中的对应命令是 *pcg()*。其基本目标是改善系数矩阵 $\mathbb{C}^{-1}\mathbb{A}\mathbb{C}^{-\top}$ 的条件数或特征值聚集状态，进而提高原始算法的收敛速度。

利用原始问题的信息直接表述, PCG 法的计算流程定义如下: 任取初始向量 $\boldsymbol{x}_0$，通常设置为零；计算 $\boldsymbol{r}_0=\mathbb{A}\boldsymbol{x}_0-\boldsymbol{b}$，令 $\boldsymbol{z}_0=\mathbb{Q}^{-1}\boldsymbol{r}_0$ 和 $\boldsymbol{p}_0=-\boldsymbol{z}_0$；对 $k\geq 0$，执行循环

$$\begin{aligned}
&\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\alpha_k\boldsymbol{p}_k,\quad \alpha_k=-\frac{\boldsymbol{r}_k^{\top}\boldsymbol{z}_k}{\boldsymbol{p}_k^{\top}\mathbb{A}\boldsymbol{p}_k},\\
&\boldsymbol{r}_{k+1}=\boldsymbol{r}_k+\alpha_k\mathbb{A}\boldsymbol{p}_k,\\
&\boldsymbol{z}_{k+1}=\mathbb{Q}^{-1}\boldsymbol{r}_{k+1},\\
&\boldsymbol{p}_{k+1}=-\boldsymbol{z}_{k+1}+\beta_k\boldsymbol{p}_k,\quad \beta_k=\frac{\boldsymbol{r}_{k+1}^{\top}\boldsymbol{z}_{k+1}}{\boldsymbol{r}_k^{\top}\boldsymbol{z}_k}.
\end{aligned}$$

换言之，CG 算法的计算流程保持不变，只需每步迭代再求解一个预处理方程 $\mathbb{Q}\boldsymbol{z}=\boldsymbol{g}$ 即可。

出于计算效率的考量，预处理方程应当容易求解。通常，矩阵分裂技术的主体部分都可以作为预处理矩阵，例如对称超松弛（SSOR）方法给出的主体部分

$$\mathbb{Q}=(\mathbb{D}-\omega\mathbb{D}\mathbb{L})\mathbb{D}^{-1}(\mathbb{D}-\omega\mathbb{D}\mathbb{L})^{\top}. \tag{2.5.35}$$

预处理矩阵还有其它构造方式，例如采用不完全 LU 分解技术、以及逆矩阵的多项式近似等。

第 3 章
线性最小二乘问题的数值方法

在线性回归和数据拟合等课题中，大量的线性方程组

$$\mathbb{A}_{m\times n}\boldsymbol{x}_n = \boldsymbol{b}_m \tag{3.0.1}$$

包含非方形或不可逆的系数矩阵。当 $m \gg n$，它常常是矛盾方程组[1]，不再具有传统意义的解向量，使所有方程均成立。称 $\boldsymbol{x}_{\mathrm{LS}}$ 是 (3.0.1) 的最小二乘（Least Square = LS）解，若[2]

$$\boldsymbol{x}_{\mathrm{LS}} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\mathbb{A}\boldsymbol{x} - \boldsymbol{b}\|_2. \tag{3.0.2}$$

此时，(3.0.1) 称为线性最小二乘问题。本章将给出常用的计算方法。

3.1 线性最小二乘问题

不同于前面两章的线性方程组，此类问题的理论分析和数值方法都极具特色。本节集中介绍一些重要的理论概念和结果。

3.1.1 最小二乘解

定理 3.1. *最小二乘解必然存在。*

证明：若 $\boldsymbol{b} \in \mathrm{R}(\mathbb{A}) = \{\mathbb{A}\boldsymbol{x} \colon \boldsymbol{x} \in \mathbb{R}^n\}$，由线性方程组基本理论，可知有解向量使所有方程成立。这个解向量自然是最小二乘解，因为其残量为零。

[1] 若 $m < n$，有无穷多解，称其为不定方程组。

[2] 相比于其它 p 范数，基于欧式范数进行度量，相应问题更易分析和求解。

若 $\boldsymbol{b} \notin \mathrm{R}(\mathbb{A})$，考虑 $\boldsymbol{b}$ 在 $\mathrm{R}(\mathbb{A})$ 及其正交补空间的直和分解

$$\boldsymbol{b} = \boldsymbol{b}_1 + \boldsymbol{b}_2, \quad \boldsymbol{b}_1 \in \mathrm{R}(\mathbb{A}), \quad \boldsymbol{b}_2 \in [\mathrm{R}(\mathbb{A})]^{\perp}.$$

最小二乘解可由 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}_1$ 决定，其存在性是显然的。 □

❀ **思考 3.1.** 此定理具有多种证明方法，例如用数学分析的技术论证二次函数的极值点。

定理 3.2. 最小二乘问题 (3.0.1) 与法方程组

$$\mathbb{A}^{\top}\mathbb{A}\boldsymbol{x} = \mathbb{A}^{\top}\boldsymbol{b}$$

同解，即残量 $\boldsymbol{r} = \mathbb{A}\boldsymbol{x} - \boldsymbol{b}$ 同 $\mathbb{A}$ 的每个列向量都正交。

证明：极值点也是驻点（所有偏导均为零），简单计算即可。 □

定理 3.3. 若 $\mathbb{A}_{m\times n}$ 列满秩，即 $rank(\mathbb{A}) = n$，则 (3.0.1) 有唯一的最小二乘解

$$\boldsymbol{x} = (\mathbb{A}^{\top}\mathbb{A})^{-1}\mathbb{A}^{\top}\boldsymbol{b}.$$

若 $\mathbb{A}_{m\times n}$ 列亏秩，则 (3.0.1) 有无穷多个最小二乘解。

✍ **论题 3.1.** 设 $r = rank(\mathbb{A}) > 0$，有满秩分解

$$\mathbb{A}_{m\times n} = \mathbb{F}_{m\times r}\mathbb{G}_{r\times n}$$

其中 $\mathbb{F}$ 是列满秩的，$\mathbb{G}$ 是行满秩的。

基于满秩分解，可以导出 (3.0.1) 的一个最小二乘解

$$\boxed{\boldsymbol{x}_{\mathrm{LS}} = \mathbb{G}^{\top}(\mathbb{G}\mathbb{G}^{\top})^{-1}(\mathbb{F}^{\top}\mathbb{F})^{-1}\mathbb{F}^{\top}\boldsymbol{b}.}$$

它是在最小二乘解集合中**长度最小**的，称为**极小最小二乘解**。简单验证可证：**极小最小二乘解唯一**。

3.1.2 广义逆矩阵

若系数矩阵可逆，则 (3.0.1) 具有唯一解 $\boldsymbol{x}_{\mathrm{LS}} = \mathbb{A}^{-1}\boldsymbol{b}$。注意到前面给出的极小最小二乘解，传统的 “逆矩阵” 概念能否推广到最小二乘问题，使 “问题的解” 具有相同形式的表达?

早在 1920 年，E. H. Moore 利用子空间正交投影，提出了 “广义逆矩阵” 概念。因用途不明，很少被问津。直到 1955 年，R. Penrose 给出此概念的四条等价表述，相关研究才被关注和应用起来。

定义 3.1. *已知 $\mathbb{A} \in \mathbb{R}^{m\times n}$。若 $\mathbb{X} \in \mathbb{R}^{n\times m}$ 满足*[3]*:*

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

则称 $\mathbb{X}$ 是 $\mathbb{A}$ 的 Moore-Penrose 广义逆或伪逆 (pseudo inverse matrix)，记为 $\mathbb{X} = \mathbb{A}^{\dagger}$。

对于非奇异方阵，广义逆矩阵就是古典逆矩阵。

定理 3.4. *广义逆矩阵存在唯一。*

证明：对于零矩阵，其广义逆矩阵也是零矩阵；利用非零矩阵的满秩分解 $\mathbb{A} = \mathbb{F}\mathbb{G}$，可以验证

$$\mathbb{A}^{\dagger} = \mathbb{G}^{\top}(\mathbb{G}\mathbb{G}^{\top})^{-1}(\mathbb{F}^{\top}\mathbb{F})^{-1}\mathbb{F}^{\top} \tag{3.1.3}$$

就是广义逆矩阵。唯一性可由四条规则直接导出，见教科书。 □

性质 3.1. *(3.0.1) 的极小最小二乘解可表示为 $\boldsymbol{x}_{LS} = \mathbb{A}^{\dagger}\boldsymbol{b}$.*

通常，广义逆的计算过程非常繁琐。(3.1.3) 是一种常用的理论算法；

[3]概念可以简单推广到复矩阵，即转置替换为共轭转置即可。

利用矩阵的特殊结构，计算过程可以简化，例如

$$\begin{bmatrix} \mathbb{X}_{r,r} & \mathbb{O}_{r,n-r} \\ \mathbb{O}_{m-r,r} & \mathbb{O}_{m-r,n-r} \end{bmatrix}^{\dagger} = \begin{bmatrix} \mathbb{X}_{r,r}^{-1} & \mathbb{O}_{r,m-r} \\ \mathbb{O}_{n-r,r} & \mathbb{O}_{n-r,m-r} \end{bmatrix}.$$

✎ **论题 3.2.** 无论是运算规则还是理论性质，广义逆矩阵都与古典逆矩阵有着明显的区别：

> *1.* $(\mathbb{AB})^{\dagger} \neq \mathbb{B}^{\dagger}\mathbb{A}^{\dagger}$，　$\mathbb{AA}^{\dagger} \neq \mathbb{A}^{\dagger}\mathbb{A}$，　$(\mathbb{A}^{k})^{\dagger} \neq (\mathbb{A}^{\dagger})^{k}$；
>
> *2.* 若 $\mathbb{A}$ 是方阵，它与 $\mathbb{A}^{\dagger}$ 的非零特征值不互为倒数；
>
> *3.* 广义逆矩阵可能不再连续依赖于原有矩阵。

特别地，第三条性质已经隐隐指出，最小二乘问题的数值计算更易受到舍入误差的干扰。

❀ **思考 3.2.** 以二阶奇异方阵为例，验证前两条性质。

在第三条性质中，“矩阵秩在扰动过程中保持不变”扮演着重要的作用。Stewart（1969）证明了：若矩阵秩保持不变，则广义逆矩阵连续依赖矩阵元素，其敏感程度（放大率）正相关于谱条件数

$$\kappa_2(\mathbb{A}) = \|\mathbb{A}\|_2 \|\mathbb{A}^{\dagger}\|_2.$$

若矩阵秩发生改变，则矩阵元素的微小变化也有可能引起广义逆矩阵元素的剧烈变化。

❀ **思考 3.3.** 为直观理解上述论述，不妨观察

$$\mathbb{A}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \varepsilon & 0 \end{bmatrix}, \quad \mathbb{A}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \varepsilon & 1 \end{bmatrix}$$

的广义逆矩阵当 $\varepsilon \to 0$ 时的具体表现。

3.1.3 正规化求解方法

线性最小二乘问题 (3.0.1) 可采用直接解法[4]，主要有正规化和直交化两类。本小节关注正规化求解方法，它特别适用于 $m \gg n$，在节省计算量和数据存储空间方面具有优势。

当 (3.0.1) 是列满秩问题时，唯一的最小二乘解满足法方程组

$$\mathbb{A}^\top \mathbb{A}\boldsymbol{x} = \mathbb{A}^\top \boldsymbol{b},$$

或等价的扩展法方程组（也称为 Karush-Kuhm-Tucker 方程）

$$\begin{bmatrix} \mathbb{I}_n & 0 & \mathbb{A}_1 \\ 0 & \mathbb{I}_{m-n} & \mathbb{A}_2 \\ \mathbb{A}_1^\top & \mathbb{A}_2^\top & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{x} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{0} \end{bmatrix}. \tag{3.1.4}$$

在 (3.1.4) 中，数据源于矩阵分块

$$\begin{bmatrix} \mathbb{A} & \boldsymbol{r} & \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \mathbb{A}_1 & \boldsymbol{r}_1 & \boldsymbol{b}_1 \\ \mathbb{A}_2 & \boldsymbol{r}_2 & \boldsymbol{b}_2 \end{bmatrix},$$

其中 $\boldsymbol{r}$ 是残量，$\mathbb{A}_1$ 是系数矩阵前 n 行构成的可逆方阵。两个方程组都可以用前两章给出的数值方法求解。

✯ **说明 3.1.** 扩展法方程组和法方程组具有相同的舍入误差困难。两者相比，扩展法方程组回避了 $\mathbb{A}^\top\mathbb{A}$ 的直接计算，并同时计算出相应残量，在某种程度上减少了舍入误差的负面影响。

✯ **说明 3.2.** 若系数矩阵是列亏秩的，则最小二乘解不再唯一，数值计算将变得更加困难。主要理由有：

- 由于舍入误差的影响，最大线性无关组（或者列向量的线性无关性）的数值判定难以做到足够准确。

[4]迭代法和优化方法也是常用的，详略。

- 当列向量线性相关时，有些算法可能无法顺利执行到底。
- 即便算法能够顺利执行到底，其计算结果也只是最小二乘解而已，不一定是极小最小二乘解。由于舍入误差的影响，不同算法给出的结果可能差异较大。

若无特别申明，本讲义重点讨论列满秩的线性最小二乘问题。

对于列满秩问题，可以理论证明 [11]：残量的灵敏度大致与 $\kappa_2(\mathbb{A})$ 成正比，而 LS 解的灵敏度大致与 $\kappa_2(\mathbb{A}) + \|\boldsymbol{r}\|_2\kappa_2^2(\mathbb{A})$ 成正比，其中[5]

$$\kappa_2(\mathbb{A}) = \|\mathbb{A}\|_2\|\mathbb{A}^\dagger\|_2$$

是最小二乘问题的条件数。对于良态问题（条件数 $\kappa_2(\mathbb{A})$ 较小），正规化方法完全可行；对于病态问题（即使残量很小但条件数 $\kappa_2(\mathbb{A})$ 很大），正规化方法将陷入严重的舍入误差困扰。主要理由有：

1. 法方程组的条件数是 $\kappa_2(\mathbb{A}^\top\mathbb{A}) = [\kappa_2(\mathbb{A})]^2$，病态程度激增，舍入误差的干扰将更加严重。
2. 当浮点运算出现上下溢出时，法方程组的系数矩阵也可能出现退化。例如，设 ϑ 是机器精度，当 $\varepsilon < \sqrt{\vartheta}$ 时，有

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

综上所述，病态问题需要更加有效的舍入误差控制。相比于正则化方法，直交化方法在此层面上占据优势。

[5]此处 $\|\mathbb{A}\|_2 = [\varrho(\mathbb{A}^\top\mathbb{A})]^{1/2}$，即矩阵 $\mathbb{A}$ 的最大奇异值。

3.2 矩阵直交分解

矩阵直交分解是直交化方法的核心操作，不仅具有重要的理论价值，也广泛应用于数值代数的各个领域。它主要有 Gram-Schmidt（GS）直交化和直交矩阵变换两种实现途径。

3.2.1 Gram-Schmidt 直交化

《高等代数》介绍过 GS 直交化的实现过程，其核心操作是将线性无关向量组转化为等价的标准直交向量组。本讲义重点关注其在数值层面的表现和应用。

✍ **论题 3.3.** 设 $r = rank(\mathbb{A}) > 0$。*GS* 直交化有两种执行次序，都可以实现矩阵的直交分解：

> 存在置换阵 $\mathbb{P}$、列直交阵 $\mathbb{Q}$ 和上梯形矩阵 $\mathbb{U}$，使得
>
> $$\mathbb{A}_{m\times n}\mathbb{P}_{n\times n} = \mathbb{Q}_{m\times r}\mathbb{U}_{r\times n},$$
>
> 其中 $\mathbb{U}$ 的对角元为正。此公式也称为（约化）*QR/QU* 分解。

利用数据覆盖技术，$\mathbb{Q}_{m\times r}$ 可以存储在 $\mathbb{A}$ 的原有位置（特别是当 $r = n$ 时）。若要记录上梯形矩阵 $\mathbb{U}_{r\times n}$ 的信息，需额外开辟相应的存储空间。

☆ **说明 3.3.** 为避免 *GS* 直交化因除零而中断，$\mathbb{A}$ 的前 r 个列向量必须线性无关。简而言之，**该技术主要适用于列满秩矩阵**。

对于列亏秩矩阵，我们必须先找到最大线性无关组，并将其交换到前 r 列，即置换阵 $\mathbb{P}_{n\times n}$ 右乘的作用。此目标理论可行，但数值实现困难，因为舍入误差会导致“数值秩”跳跃。

GS 直交化有两种执行次序，它们在理论上完全等价，但舍入误差表现不同。这个事实也展现了计算数学的一个特性：**理论等价的数值方法可能会有不同的数值表现**。具体陈述如下：

1. 传统（CGS）方法**逐列**计算 $\mathbb{U}$ 的信息，主要利用当前向量与**历史**向量的正交性；

2. 修正（MGS）方法**逐行**计算 $\mathbb{U}$ 的信息。它充分利用了当前向量与**未来**向量的正交性，不断剔除等待处理向量在历史向量张成子空间的投影。在某种程度上，上述操作可视为直交化的计算规模不断变小，令舍入误差的积累程度得到减缓。

若无特殊申明，GS 直交化操作均默认是 MGS 方法。

✯ **说明 3.4.** 同 *CGS* 方法相比，*MGS* 方法的数值健状（或稳定）性更好。设 $\mathbb{A}_{m\times n}$ 是列满秩的，有如下结论：

1. 基于向后误差分析理论，计算机上的 *MGS* 方法可等价描述为扰动矩阵的 *QR* 分解，即

$$\mathbb{A}+\delta\mathbb{A}_{MGS}=\mathbb{Q}_{MGS}\mathbb{R}_{MGS},$$

其中 $\delta\mathbb{A}_{MGS}$ 是扰动矩阵，$\mathbb{Q}_{MGS}$ 和 $\mathbb{R}_{MGS}$ 是相应的数值结果。理论分析（*Björck,1967*）表明：

$$\|\delta\mathbb{A}_{MGS}\|_2\leq c_{m,n}\vartheta\|\mathbb{A}\|_2, \tag{3.2.5a}$$

$$\|\mathbb{Q}_{MGS}^{\top}\mathbb{Q}_{MGS}-\mathbb{I}\|_2\leq c_{m,n}\vartheta\kappa_2(\mathbb{A})+O((\vartheta\kappa_2(\mathbb{A}))^2), \tag{3.2.5b}$$

其中 ϑ 是机器精度，$c_{m,n}$ 是绝对常数。

2. 类似地，计算机上的 *CGS* 方法也可等价地描述为

$$\mathbb{A}+\delta\mathbb{A}_{CGS}=\mathbb{Q}_{CGS}\mathbb{R}_{CGS},$$

其中 $\delta\mathbb{A}_{CGS}$ 是扰动矩阵，$\mathbb{Q}_{CGS}$ 和 $\mathbb{R}_{CGS}$ 是相应的数值结果。理论分析表明：$\delta\mathbb{A}_{CGS}$ 满足类似 (3.2.5a) 的估计，但 $\mathbb{Q}_{CGS}$ 的列直交性表现变差，没有类似 (3.2.5b) 的估计。

举例说明上述结论，不妨考虑 25×15 阶的范德蒙矩阵

$$\mathbb{A}=\{p_i^{j-1}\},\quad p_i=i/25.$$

此实验摘录于 *N.J.Higham* 的*"Accuracy and Stability of Numerical Algorithms"* 第二版的第 *373* 页。由数值结果

$$\|\delta\mathbb{A}_{CGS}\|_2=\|\mathbb{A}-\mathbb{Q}_{CGS}\mathbb{R}_{CGS}\|_2=5.0\times 10^{-16},$$
$$\|\delta\mathbb{A}_{MGS}\|_2=\|\mathbb{A}-\mathbb{Q}_{MGS}\mathbb{R}_{MGS}\|_2=1.0\times 10^{-15},$$

可知 *MGS* 和 *CGS* 方法都是向后稳定的，扰动信息均达到机器精度附近。换言之，即使 $\mathbb{Q}_{\rm num}$ 和 $\mathbb{R}_{\rm num}$ 同真实结果偏差很大，它们的乘积 $\mathbb{Q}_{\rm num}\mathbb{R}_{\rm num}$ 也神奇地接近 $\mathbb{A}$。事实上，这是所有直交化方法都具有的数值优势之一。计算结果还表明

$$\|\mathbb{Q}_{CGS}^{\top}\mathbb{Q}_{CGS}-\mathbb{I}\|_2=5.2,$$
$$\|\mathbb{Q}_{MGS}^{\top}\mathbb{Q}_{MGS}-\mathbb{I}\|_2=9.5\times 10^{-9}.$$

换言之，两种方法在列向量的直交性表现具有明显的差异。

☆ **说明 3.5.** 考虑对角阵 $diag\{2^{-k}\}_{k=1}^{80}$，随机左乘和右乘直交阵，最终形成一个高度病态的稠密矩阵。图 *3.2.1* 绘制了三角阵 $\mathbb{U}$ 的 *80* 个对角元素，其中方框表示 *CGS* 方法，圆圈表示 *MGS* 方法。方法差异清晰可见：*CGS* 方法仅仅达到机器精度开根号量级，而 *MGS* 方法可以达到机器精度量级。

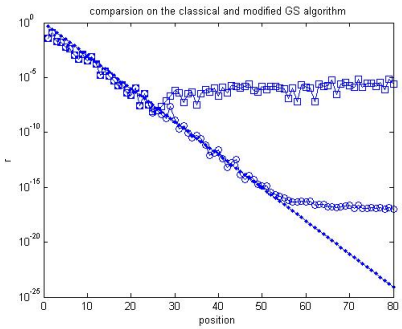


图 3.2.1: 两种 GS 方法给出的三角阵对角元。

✍ **论题 3.4.** 对于 QR 分解给出的上梯形阵 $\mathbb{U}$，继续执行其转置矩阵的 GS 直交化，可以建立 **不完全直交分解**：

> 存在上三角阵 $\mathbb{R}$，列直交阵 $\mathbb{Q}$ 和 $\mathbb{V}$，实现上三角化，即
>
> $$\mathbb{A}_{m\times n} = \mathbb{Q}_{m\times r}\mathbb{R}_{r\times r}\mathbb{V}^{\top}_{n\times r}.$$

通过列直交阵的正交扩充，可得 **完全直交分解**：

> 存在上三角阵 $\tilde{\mathbb{R}}$，直交方阵 $\mathbb{H}$ 和 $\mathbb{K}$，使得
>
> $$\mathbb{A}_{m\times n} = \mathbb{H}_{m\times m}\tilde{\mathbb{R}}_{m\times n}\mathbb{K}^{\top}_{n\times n},$$
>
> 其中 $\tilde{\mathbb{R}}$ 是上三角阵 $\mathbb{R}_{r\times r}$ 的零扩充。

完全直交分解是重要的分析工具，它可以给出最小二乘解的基本结构；具体内容见 §3.3。

✯ **说明 3.6.** 对于上梯形矩阵的转置，*GS* 直交化过程可采用从右到左的执行过程，降低四则运算的次数。

3.2.2 Householder 方法和 Givens 方法

它们都属于直交矩阵变换技术，分别利用 Householder 镜像变换阵或 Givens 平面旋转阵的不断左乘，实现矩阵的上梯形化。

Householder 镜像变换阵

它最早由 Turnbull 和 Aitken (1932) 提出，用于证明 Schur 分解的存在性。Householder (1958) 将其名扬天下，用于矩阵特征值问题的计算。

✇ **定义 3.2.** 设 $\boldsymbol{u}_n$ 是非零向量，记 $b = \frac{1}{2}\|\boldsymbol{u}_n\|_2^2$。称

$$\mathbb{H}_{n\times n} = \mathbb{I}_{n\times n} - b^{-1}\boldsymbol{u}_n\boldsymbol{u}_n^{\top} \tag{3.2.6}$$

为 *Householder* 镜像（或反射）变换阵，它是单位矩阵的秩一修正。

❀ **思考 3.4.** 高维向量的长度计算可能产生上溢和下溢。为增强算法的健壮性，建议采用如下的代码

$$m = max(abs(\boldsymbol{u}));\ \boldsymbol{y} = \boldsymbol{u}/m;\ return\ \ m{*}norm(\boldsymbol{y});$$

✎ **论题 3.5.** *Householder* 镜像变换阵继承了低秩修正技术关于计算复杂度的优势。通常，矩阵同向量相乘要 n^2 次乘除运算，而

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

只需 $2n+1$ 次乘除运算。

Householder 镜像变换阵是直交 (orthogonal) 阵, 对称 (symmetric) 阵和对合 (involutory) 阵。特别地, 它还具有“镜像”效应

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

✍ **论题 3.6.** 已知非零实向量 $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)^\top$。基于镜像效应, 可以实现“仅首个分量非零”目标, 即: 存在 *Householder* 镜像变换阵[6], 使得

$$\mathbb{H}_{n\times n}\boldsymbol{a} = (\alpha, 0, 0, \ldots, 0)^\top.$$

相应的数值实现简单, 其算法 $[\alpha, b] = \mathit{householder}(\boldsymbol{a})$ 的伪代码如下:

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

镜面法向 $\boldsymbol{u} = \boldsymbol{a} - \alpha\boldsymbol{e}_1$ 覆盖存储在 $\boldsymbol{a}$ 处, 只需修改首个分量即可。

1. 为了避免后续的重复计算, 输出列表保留了 b, 即采用了“空间换速度”的策略;
2. 出于数值稳定性, 选取 α 的符号, 使 b 具有更大绝对值。
3. 镜像阵 $\mathbb{H}$ 无需保存, 可由 b 和 $\boldsymbol{u}$ 快速重构。参照论题 *3.5*。

☆ **说明 3.7.** *Wilkinson* (*1965*) 指出: 上述算法是数值稳定的, 即

$$\|\mathbb{H}_{num} - \mathbb{H}\|_2 \leq C\vartheta,$$

[6]若 $a_1 \neq 0$ 且余下分量全为零, 则 $\mathbb{H}_{n\times n}$ 可直接定义为单位阵。为叙述方便, 有时也将单位阵归入 Householder 变换阵。

其中 C 是绝对常数，ϑ 是机器精度。

❀ **思考 3.5.** 能否将论题 *3.6* 的技术推广到复数域？

✎ **论题 3.7.** 设 $r = rank(\mathbb{A}) > 0$，矩阵 $\mathbb{A}_{m\times n}$ 的前 r 列线性无关。依次左乘 *Householder* 镜像变换阵（包括相应的单位扩张），将对角线下方的元素清零，最终可得上梯形矩阵，即

$$\mathbb{H}_s \cdots \mathbb{H}_1 \mathbb{A} = \mathbb{R}_{m\times n} = \begin{bmatrix} \mathbb{R}_{r\times r} & \mathbb{R}_{r\times(n-r)} \\ \mathbb{O} & \mathbb{O} \end{bmatrix},$$

其中 $s \le r$ 是镜像变换的次数。相应的数值实现过程是：

1. *For* $k = 1, 2, \ldots, s$, *Do*
2. 计算 $m-k+1$ 阶矩阵 $\mathbb{H}_k$ 的主要信息，即 $[\alpha, b]$ =*householder*($\mathbb{A}(k:m, k)$);
3. 计算矩阵乘积，即 $\mathbb{H}_k\mathbb{A}(k:m, k+1:n)$;
4. *Enddo*

第 *2* 行代码隐含数据覆盖技术：镜面法向 $\boldsymbol{u}$ 保存在相应位置；α 是上梯形矩阵 $\mathbb{R}_{m\times n}$ 的相应对角元，需申请一个数组；若要记录所有的 b，还需申请一个数组。第 *3* 行代码的矩阵乘积运算应转化为一个镜像变换阵同多个列向量的相乘过程，相应操作参照论题 *3.5*。

❀ **思考 3.6.** 利用上述代码记录的信息 $[\boldsymbol{u}, b]$，给出直交阵

$$\mathbb{Q}_{m\times m} = \mathbb{H}_1\mathbb{H}_2 \cdots \mathbb{H}_s$$

的快速计算流程。结合论题 *3.7*，上述操作给出了直接分解

$$\mathbb{A}_{m\times n} = \mathbb{Q}_{m\times m}\mathbb{R}_{m\times n}.$$

✯ **说明 3.8.** 论题 *3.7* 的算法也是向后稳定的，关于舍入误差的表现也是完美的。对于列满秩矩阵而言，*Householder* 镜像变换方法给出的直交阵 $\mathbb{H}_{\text{num}} = \mathbb{H}_1\mathbb{H}_2\cdots\mathbb{H}_s$ 满足

$$\|\mathbb{H}_{\text{num}}^{\top}\mathbb{H}_{\text{num}} - \mathbb{I}\| \leq C\vartheta,$$

相应的列直交性表现要强于 *MGS* 方法。

事实上，*MGS* 方法也仅仅在这个指标上弱于正交矩阵技术。对于列满秩矩阵，*MGS* 方法等同于 *Gauss* 消去阵的不断右乘（或初等列变换）将其转化为列直交阵。同正交阵相比，*Gauss* 消去阵带来更加严重的舍入误差。

✯ **说明 3.9.** 设 $\mathbb{A}_{m\times n}$ 是列满秩的，即 $m \geq n = r$，*Hoseholder* 方法和 *MGS* 方法的乘除次数[7]分别为

$$N_{opt}^{\text{House}} \approx \sum_{k=1}^{n} 2(n+1-k)(m+1-k) \approx mn^2 - \frac{1}{3}n^3, \tag{3.2.7a}$$

$$N_{opt}^{\text{GS}} \approx \sum_{k=1}^{n} 2(n-k)m \approx mn^2. \tag{3.2.7b}$$

两者相比，*Hoseholder* 方法的计算复杂度较低。

✎ **论题 3.8.** 在进行 *Householder* 镜像变换的过程中，可以引入“主列向量”策略，更好地控制舍入误差。换言之，在执行第 k 次 *Householder* 镜像变换之前，在右下块矩阵 $\mathbb{A}(k:m, k:n)$ 中选取长度最大的列向量作为主列，并将其列置换到第 k 列。

借助“主列向量”策略，*Householder* 镜像变换也可用于列亏秩矩阵，并获得较为理想的计算结果。

[7]没有统计开根次数。

Givens 平面旋转阵

Givens 平面旋转也可实现正交变换目标。相对而言，Householder 镜像变换可以同时实现多个元素的清零，处理稠密矩阵时具有更高的计算效率；Givens 平面旋转具有定位清零的特点，对于稀疏矩阵（或者非零元素分布具有结构时）更为有效。

定义 3.3. 设 θ 是在 (i,j) 平面上的（顺时针）旋转角度。简记

$$c = \cos\theta, \quad s = \sin\theta,$$

相应的 *Givens* 平面旋转阵是直交阵

$$\mathbb{G}(i,j;\theta) = \begin{bmatrix} \mathbb{I}_1 & & & & \\ & c & & s & \\ & & \mathbb{I}_2 & & \\ & -s & & c & \\ & & & & \mathbb{I}_3 \end{bmatrix}, \tag{3.2.8}$$

其中 $\mathbb{I}_1, \mathbb{I}_2$ 和 $\mathbb{I}_2$ 是不同阶数的单位矩阵 (0 阶为空)。换言之，角度信息仅仅出现在 (i,j) 井字线交叉点上。

说明 3.10. 除非 $s = 0$，*Givens* 平面旋转阵是单位阵的秩二修正，是非对称的。

论题 3.9. 已知 $\boldsymbol{a} = (\cdots, x_i, \cdots, x_j, \cdots)^\top$，其中

$$r = \sqrt{x_i^2 + x_j^2} \neq 0.$$

确定一个 *Givens* 平面旋转阵 $\mathbb{G} \equiv \mathbb{G}(i,j;\theta)$，将第 j 个分量清零，使得

$$\mathbb{G}\boldsymbol{a} = (0, \ldots, \pm r, \ldots, 0, \ldots, 0)^\top.$$

数值实现很简单，相应算法记为 $[c,s]=\mathsf{givens}(i,j,\boldsymbol{a})$。包含舍入误差有效处理的伪代码如下：

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
$$t=\frac{x_j}{x_i},c=\frac{1}{\sqrt{1+t^2}},s=ct;$$

在上述操作中，t 可能是 $\cot\theta$ 也可能是 $\tan\theta$，保证 $|t|\leq 1$ 永远成立。Wilkinson（1965）指出：上述操作具有理想的数值稳定性，即

$$|c_{\text{num}}-c|+|s_{\text{num}}-s|\leq C\vartheta,$$

其中 C 是给定的绝对常数。

✯ **说明 3.11.** *要记录 Givens 平面旋转阵，需存储位置信息 i 和 j 以及角度信息 c 和 s。*

Stewart (1976) 提出了一种压缩存储方法，将两个浮点数 (c,s) 转化为一个浮点数 ρ，直接覆盖存储在清零的 x_j 处。相应的伪代码是

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

第 2 步给出 $|\rho|\leq 1/2$，第 3 步给出 $|\rho|\geq 2$，可以清晰区别开来。技术实质是存储正弦或余弦中（绝对值）较小的那个。若要由存储的 ρ 恢复出 c 和 s，只需执行如下代码：

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 $c=2/\rho,s=\sqrt{1-c^2}$.

因此说，*Stewart* 技术采用了“时间换空间”的策略。

✎ **论题 3.10.** 通过两两组合，*Givens* 平面旋转阵即可实现论题 *3.6* 的数值目标；在此基础上，也可类似实现论题 *3.7* 的数值目标。

答：Givens 平面旋转的乘除次数是 Householder 镜像变换的两倍，开根号次数也严重增加。事实上，后者的提出就是为了降低前者的计算复杂度。 □

❀ **思考 3.7.** 至此，有两种操作途径，实现非零向量 $\boldsymbol{a}$ 到仅首个分量非零。其一是多个 *Givens* 平面旋转阵的乘积，其二是单个 *Householder* 镜像变换阵。它们有何关系吗？

1. 因为 $\det\mathbb{G}=1$ 而 $\det\mathbb{H}=-1$，*Householder* 镜像变换阵不可能是 *Givens* 平面旋转阵的乘积。

2. 事实上，*Givens* 平面旋转阵可以表示为两个 *Householder* 镜像变换阵的乘积。请读者证明之。

✯ **说明 3.12.** 若锁定上三角阵的对角元符号为正，则列满秩矩阵的 *QR* 分解是唯一的。该结论源于教科书的习题 *7.11*：设列满秩矩阵 $\mathbb{A}$ 具有两个 *QR* 分解，即

$$\mathbb{Q}_1\mathbb{R}_1=\mathbb{A}=\mathbb{Q}_2\mathbb{R}_2,$$

则存在对角阵 $\mathbb{D}=\{\pm1\}$，使得

$$\mathbb{Q}_2\mathbb{D}=\mathbb{Q}_1,\quad \mathbb{D}\mathbb{R}_1=\mathbb{R}_2.$$

✯ **说明 3.13.** *Householder* 镜像变换也可用于线性方程组的数值求解，但是很少被使用，其理由如下：

1. *Householder* 镜像变换的乘除次数是 *Gauss* 消元法的两倍；

2. 在多数情况下，列主元 *Gauss* 消元法给出了较好的数值结果。

3.3 直交化求解方法

本节基于矩阵的直交分解，给出 (3.0.1) 最小二乘解的各种计算公式。在计算复杂度和舍入误差控制方面，它们各具特色。

3.3.1 基于完全直交分解

定理 3.5. 完全直交分解给出最小二乘解的基本结构

$$\boldsymbol{x}_{LS} = \mathbb{K} \begin{bmatrix} \mathbb{R}^{-1}\boldsymbol{g}_r \\ \boldsymbol{y}_{n-r} \end{bmatrix}, \qquad \mathbb{H}^{\top}\boldsymbol{b} = \begin{bmatrix} \boldsymbol{g}_r \\ \boldsymbol{h}_{m-r} \end{bmatrix},$$

其中 $\boldsymbol{y}_{n-r}$ 是任意的，下标表示向量维数。相应的结论，有

- 最小二乘解的残量长度是 $\|\boldsymbol{h}_{m-r}\|_2$；
- 当 $\boldsymbol{y}_{n-r} = 0$ 时，$\boldsymbol{x}_{LS}$ 就是极小最小二乘解。

证明：利用正交变换保持向量长度不变，进行简单转化即可。 □

定理结论极具理论价值，但并不适合实际应用。完全直交分解包含列向量的直交扩张过程，不仅数值实现极其困难，而且也无助于 LS 解的计算。下面给出一些更为有用的 LS 解表达公式。

3.3.2 基于 Gram-Schmidt 直交化

✎ **论题 3.11.** 不完全直交分解可以给出极小 *LS* 解

$$\boxed{\boldsymbol{x}_{LS} = \mathbb{V}_{r\times n}\mathbb{R}^{-1}_{r\times r}\mathbb{Q}^{\top}_{m\times r}\boldsymbol{b}.}$$

它需要执行两次 *GS* 正交化过程和求解一个三角形线性方程组。

✎ **论题 3.12.** *GS* 直交化过程也可以给出极小 *LS* 解

$$\boxed{\boldsymbol{x}_{LS} = \mathbb{U}^{\top}_{r\times n}(\mathbb{U}_{r\times n}\mathbb{U}^{\top}_{r\times n})^{-1}\mathbb{Q}^{\top}_{m\times r}\boldsymbol{b}.}$$

若矩阵 $\mathbb{A}$ 是列满秩的，极小 *LS* 解有更简洁的答案

$$\boxed{\boldsymbol{x}_{LS} = \mathbb{U}^{-1}_{n\times n}\mathbb{Q}^{\top}_{m\times n}\boldsymbol{b}.}$$

前者需要求解一个对称正定线性方程组，而后者只需要求解一个三角形线性方程组。

✯ **说明 3.14.** 在论题 *3.11* 和 *3.12* 的三个公式中，等号右端的矩阵都是广义逆矩阵 $\mathbb{A}^{\dagger}$。

✯ **说明 3.15.** 本节所有公式都有 $\mathbb{H}^{\top}\boldsymbol{b}$ 或 $\mathbb{Q}^{\top}\boldsymbol{b}$。基本处理是：直接对增广矩阵 $[\mathbb{A}|\boldsymbol{b}]$ 执行直交化操作，并在最后一列提取相关信息。数值经验表明：若由 $\mathbb{A}$ 计算出 $\mathbb{H}$ 或 $\mathbb{Q}$，再相乘得到 $\mathbb{H}^{\top}\boldsymbol{b}$ 或 $\mathbb{Q}^{\top}\boldsymbol{b}$，则数值操作遭遇更多的舍入误差影响，使得数值精度下降。在某种程度上，两种处理的数值差异可以归结为计算次序改变或者寄存器读取造成的。

✯ **说明 3.16.** 若 $\mathbb{A}$ 是列亏秩矩阵，上述公式在理论上依旧可行，但要牢记直交化过程存在数值不稳定性，数值结果可能产生巨大偏差，甚至计算过程出现意外停机。

3.3.3 基于正交矩阵变换技术

无论是采用 Givens 平面旋转和 Householder 镜像变换，最小二乘问题的求解过程是类似的。

论题 3.13. 设 $r = rank(\mathbb{A}) > 0$，矩阵 $\mathbb{A}_{m\times n}$ 的前 r 列线性无关。不断采用 *Householder* 镜像变换（或 *Givens* 平面旋转），对增广矩阵进行从左到右、从上到下的处理，最终得到

$$\underbrace{\mathbb{H}_{r-1}\cdots\mathbb{H}_2\mathbb{H}_1}_{\mathbb{Q}^\top}[\,\mathbb{A}\mid\boldsymbol{b}\,]=\begin{bmatrix}\mathbb{R}_{r\times r} & \mathbb{R}_{r\times(n-r)} & \mathbb{Q}_1^\top\boldsymbol{b}\\ \mathbb{O} & \mathbb{O} & \mathbb{Q}_2^\top\boldsymbol{b}\end{bmatrix},$$

其中 $\mathbb{Q}=[\mathbb{Q}_1,\mathbb{Q}_2]$ 是正交阵，$\mathbb{R}_{r\times r}$ 是可逆的上三角阵，$[\mathbb{R}_{r\times r},\mathbb{R}_{r\times(n-r)}]$ 是上梯形阵。上述信息给出 *LS* 解

$$\boldsymbol{x}_{LS}=\mathbb{R}_{r\times r}^{-1}\mathbb{Q}_1^\top\boldsymbol{b}, \tag{3.3.9}$$

对应残量的大小是 $\|\mathbb{Q}_2^\top\boldsymbol{b}\|_2$。简要说明如下：

1. 若 $\mathbb{A}$ 列满秩，则 $\mathbb{R}_{r\times(n-r)}$ 不存在，(3.3.9) 是极小最小二乘解。
2. 若 $\mathbb{A}$ 列亏秩，则 $\mathbb{R}_{r\times(n-r)}$ 非空，(3.3.9) 只是最小二乘解。要得到极小最小二乘解，还需要在增广矩阵的右侧施行 *Householder* 镜像变换，将 $\mathbb{R}_{r\times(n-r)}$ 消除。详略。

3.4 奇异值分解

奇异值分解是 Schur 分解的推广，在矩阵分析、信息处理、图像压缩、统计分析、机器学习等领域中，都发挥着非常重要的作用。相应的 Matlab 命令是 svd()。

定理 3.6. 对于任意实矩阵 $\mathbb{A}$，均有奇异值分解

$$\mathbb{A}_{m\times n} = \mathbb{U}_{m\times m}\mathbb{D}_{m\times n}\mathbb{V}^{\top}_{n\times n},$$

其中 $\mathbb{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m]$ 和 $\mathbb{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n]$ 是直交阵，$\mathbb{D}$ 是由非负数 $\sigma_1, \sigma_2, \ldots, \sigma_p$ 生成的广义对角阵，其中 $p = \min(m, n)$。

在奇异值分解中，σ_i 称为奇异值，$\boldsymbol{u}_i$ 称为左奇异向量，$\boldsymbol{v}_i$ 称为右奇异向量。换言之，对 $1 \le i \le p$ 有

$$\boldsymbol{u}_i^{\top}\mathbb{A} = \sigma_i \boldsymbol{v}_i^{\top}, \quad \mathbb{A}\boldsymbol{v}_i = \sigma_i \boldsymbol{u}_i.$$

这些概念同 $\mathbb{A}^{\top}\mathbb{A}$ 或 $\mathbb{A}\mathbb{A}^{\top}$ 的特征值问题密切有关。通常，奇异值是降序排列的，即

$$\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_r > 0, \quad \sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_p = 0,$$

其中 $r = \mathrm{rank}(\mathbb{A})$ 是矩阵的秩。

☆ **说明 3.17.** 有限维空间的线性变换可利用矩阵进行描述，其具体形式强烈依赖于坐标系设置。奇异值分解理论的几何含义是：在两个空间选取合适的正交坐标系，线性变换可以直接用坐标轴到坐标轴的伸缩变换来描述。相应的物理解释是：刚体弹性变形的本质就是旋转和拉伸的叠加。

✎ **论题 3.14.** 利用奇异值分解理论，简单验证可知

1. 值域 $R(\mathbb{A}) = span\{\boldsymbol{u}_i\}_{i=1}^{r}$；
2. 核空间 $ker(\mathbb{A}) = span\{\boldsymbol{v}_i\}_{i=r+1}^{n}$；
3. 秩一展开 $\mathbb{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\top}$.

✍ **论题 3.15.** 奇异值刻画了给定矩阵 $\mathbb{A}$ 到某个低秩矩阵集合的距离，即：若 $k \le r = \mathrm{rank}(\mathbb{A})$，则有

$$\min_{\mathrm{rank}(\mathbb{B})=k} \|\mathbb{A}-\mathbb{B}\|_2 = \|\mathbb{A}-\sum_{i=1}^{k}\sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T\|_2 = \sigma_{k+1}.$$

此性质导出两个概念，其一是矩阵 $\mathbb{A}$ 的 δ 秩，即

$$k = \min\{\mathrm{rank}(\mathbb{B})\colon \|\mathbb{A}-\mathbb{B}\|_2 \le \delta, \forall \mathbb{B}\}.$$

其二是矩阵 $\mathbb{A}$ 的**数值秩**：若奇异值 σ_k 和 σ_{k+1} 位于机器精度两侧，则称 k 是数值秩。

✍ **论题 3.16.** 若有奇异值分解 $\mathbb{A} = \mathbb{U}\mathbb{D}\mathbb{V}^\top$，则 (3.0.1) 的极小 *LS* 解是 $\boldsymbol{x}_{LS} = \mathbb{A}^\dagger \boldsymbol{b}$，其中

$$\boxed{\mathbb{A}^\dagger = \mathbb{V}\mathbb{D}^\dagger\mathbb{U}^\top.}$$

该表达式的数值稳定性很强，可适用于更加病态（含列亏秩）的最小二乘问题。

✯ **说明 3.18.** 奇异值分解的计算并不容易，通常不能在有限步内完成。常用的数值方法是：首先采用 *Householder* 镜像变换，将矩阵变换为双对角线的上三角阵，然后通过迭代求解过程（类似于特征值问题的 *QR* 方法），将其相似变换为近似对角阵。具体内容可参阅 *Golub* 和 *Kahan* 在 *20* 世纪 *60* 年代的工作；详略。

✯ **说明 3.19.** 奇异值分解理论也是重要的分析工具，例如它可以轻松证明

$$\mathbb{A}^\dagger = \lim_{a\to 0}\left[(\mathbb{A}^\top\mathbb{A}+a^2\mathbb{I})^{-1}\mathbb{A}^\top\right] = \lim_{a\to 0}\left[\mathbb{A}^\top(\mathbb{A}\mathbb{A}^\top+a^2\mathbb{I})^{-1}\right].$$

同时，它也有助于理解 *Tikhonov* 正则化方法

$$\|\mathbb{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\tau^2\|\mathbb{D}\boldsymbol{x}\|_2^2=\min,$$

其中 $\tau>0$ 且 $\mathbb{D}$ 是正定的对角阵。请读者给出相应的过程。

✯ **说明 3.20.** 奇异值分解可用于图像（或矩阵 $\mathbb{A}_{m\times n}$）的压缩存储。对于有意义的图像，其主要结构（位居前面的主奇异值）的个数远远小于 $\min(m,n)$。若截取秩一展开的前 k 项，则数据存储量从 mn 下降到 $(m+n+1)k$。为展示其效果，考虑 *Matlab* 系统自带小丑图的压缩和恢复。相应的代码是：

```
1. load clown.mat;
2. [U,S,V]=svd(X);
3. colormap('gray');
4. image(U(:,1:k)*S(1:k,1:k)*V(:,1:k)');
```

参见图 *3.4.2*，当 k 足够大时，恢复图像已经同原始图像没有明显差别。

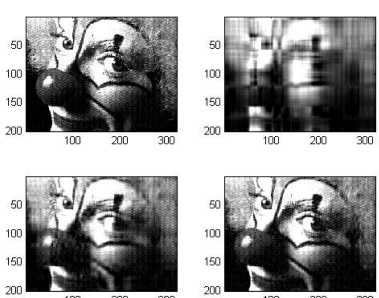


图 3.4.2: 左上角为原图，其它三个分别对应 $k=5,10,15$。

3.5 离散数据拟合

离散数据拟合就是在大量的离散数据 $\{(x_i, y_i)\}_{i=1:m}$ 中挖掘真实规律，确定近似（或经验）公式

$$y(x) = \sum_{j=0}^{n} \alpha_j \phi_j(x), \quad x \in \mathcal{U}, \tag{3.5.10}$$

其中 $\{\phi_j(x)\}_{j=0}^n$ 是事先选定的线性无关函数组，$\{\alpha_j\}_{j=0}^n$ 是待定参数。在工程材料学中，上述问题称为 “参数识别”。若要模型误差的欧式范数最小，我们可导出线性最小二乘问题

$$y_i = \sum_{j=0}^{n} \alpha_j \phi_j(x_i), \quad i = 1:m. \tag{3.5.11}$$

若系数矩阵 $\{\phi_j(x_i)\}_{j=0:n}^{i=1:m}$ 是列满秩的，则前面介绍的各种数值方法都是可行的。特别地，当待定参数个数不多时，法方程组是最简单最常用的求解方法。

论题 3.17. 以线性回归问题为例，陈述具体的计算流程。

论题 3.18. 在函数逼近论中，$\phi(x)$ 的最佳平方逼近问题

$$\int_{x\in\mathcal{U}} \Big[\phi(x) - \sum_{j=0}^{n} \alpha_j \phi_j(x)\Big]^2 \mathrm{d}x = \min$$

也会导致一个法方程组。由于 $\{\phi_j(x)\}_{j=0}^n$ 线性无关，法方程组的系数矩阵对称正定，故具有唯一解。

类似地，离散数据拟合的最小二乘问题 (3.5.11) 也会导致一个法方程组。要其也具有唯一解，需 $\{\phi_j(x_i)\}_{j=0:n}^{i=1:m}$ 列满秩。此目标不永远成立，一个著名的充分条件是 ***Haar* 条件**：

对于不全为零的 $\{\beta_j\}_{j=0}^n$，方程

$$g(x) = \sum_{j=0}^{n} \beta_j \phi_j(x)$$

在观测点集 $\{x_i\}_{i=1:m}$ 上的根不超过 n 个。

Haar 条件表明：利用多项式进行数据拟合，答案总是唯一的。

第 4 章
矩阵特征值的数值解法

特征值问题广泛应用于结构力学、电力网络、量子化学和理论物理等诸多领域。本章关注（离散型）矩阵特征值问题

$$\mathbb{A}\boldsymbol{x} = \lambda\boldsymbol{x}, \quad \boldsymbol{x} \neq 0, \tag{4.0.1}$$

其中 $\mathbb{A}$ 是给定的 n 阶（实）方阵，$(\lambda, \boldsymbol{x})$ 是由特征值和特征向量组成的特征信息。特征值问题同时包含线性和非线性两种结构，相应的数值求解极具挑战性。

4.1 预备知识

本节回顾特征值问题的基本概念，给出同数值计算相关的几个重要结果：特征信息的误差度量、特征值的定位以及敏感程度。

4.1.1 基本结论

特征多项式是首项系数为一的 n 次多项式

$$f(\lambda) \equiv \det(\lambda\mathbb{I} - \mathbb{A}) = \prod_{s=1}^{m}(\lambda - \lambda_s)^{n_s}. \tag{4.1.2}$$

其中 λ_s 是互异特征值[1]，n_s 是相应的**代数重数**。

- 全部特征值构成的集合记为 $\lambda(\mathbb{A})$。

[1]实矩阵的特征值也可能是复的；有些讨论要扩充到复数域。

- 线性方程组 $(\lambda_s\mathbb{I}-\mathbb{A})\boldsymbol{x}=\mathbf{0}$ 的解就是特征向量，其基础解系构成特征（不变）子空间，相应的维数 $\gamma_s=n-\operatorname{rank}(\lambda_s\mathbb{I}-\mathbb{A})$ 称为**几何重数**。

矩阵 $\mathbb{A}$ 同 $\mathbb{A}^\top$ 的特征值相同。矩阵 $\mathbb{AB}$ 和 $\mathbb{BA}$ 的非零特征值相同。最小多项式是零化 $\mathbb{A}$ 的最低次多项式（要求其首项系数为一）

$$p(\lambda)=\prod_{s=1}^{r}(\lambda-\lambda_s)^{\ell_s}, \tag{4.1.3}$$

其中 ℓ_s 是特征值 λ_s 对应的 Jordan 块最大阶数。由 Hamiltion-Cayley 定理可知，特征多项式满足 $f(\mathbb{A})=0$，但它不一定是最小多项式。

论题 4.1. 若 $\mathbb{B}=\mathbb{X}^{-1}\mathbb{AX}$，则称 $\mathbb{A}$ 和 $\mathbb{B}$ 相似。相似矩阵具有相同的特征多项式和特征值。有三个常见的相似操作：

1. *Jordan* 分解定理：任意矩阵都可相似变换到 *Jordan* 标准形。它可以清楚给出特征信息，判定出特征向量的亏损情况。当特征向量没有亏损时，相应的矩阵称为非亏损的，可以实现相似对角化。此时，代数重数与几何重数相等。

2. 复域的 *Schur* 分解定理：任意矩阵都可酉相似变换到上三角阵。

3. 实域的 *Schur* 分解定理：任意矩阵都可正交相似变换到块上三角矩阵，位于对角线的块矩阵至多 *2* 阶。

基于 *Jordan* 分解的数值方法都是不稳定的，而基于 *Schur* 分解的数值方法是较为稳定的，相应的实现过程也更加容易。

4.1.2 特征向量的误差度量

不同于特征值，特征向量的误差度量方式需要明确。由于特征向量的核心是其所指方向，因此特征向量的误差应理解为两个向量张成的子

空间距离（或夹角）。

定义 4.1. 对于维数相同的两个子空间 $\mathcal{P}$ 和 $\mathcal{Q}$，它们的距离是

$$\operatorname{dist}(\mathcal{P},\mathcal{Q}) = \|\mathbb{P}-\mathbb{Q}\|_2, \tag{4.1.4}$$

其中 $\mathbb{P}$ 和 $\mathbb{Q}$ 是相应的两个正交投影阵[2]。

定理 4.1. 设 $\mathbb{P}_1$ 和 $\mathbb{Q}_1$ 是 $n\times k$ 阶列直交阵，相应列向量张成两个 k 维子空间

$$\mathcal{P} = span\{\mathbb{P}_1\},\quad \mathcal{Q} = span\{\mathbb{Q}_1\},$$

相应的正交投影矩阵是 $\mathbb{P}=\mathbb{P}_1\mathbb{P}_1^{\top}$ 和 $\mathbb{Q}=\mathbb{Q}_1\mathbb{Q}_1^{\top}$，则有

$$\|\mathbb{P}-\mathbb{Q}\|_2 = \sqrt{1-\sigma_{\min}^2}, \tag{4.1.5}$$

其中 $\sigma_{\min}$ 是 $\mathbb{P}_1^{\top}\mathbb{Q}_1$ 的最小奇异值。

证明：利用正交扩充和直交阵的分块运算。 □

设 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是两个单位向量，相应的子空间和正交投影阵分别是

$$\mathcal{P} = \operatorname{span}(\boldsymbol{x}),\ \mathbb{P}_1 = \boldsymbol{x},\ \mathbb{P} = \boldsymbol{x}\boldsymbol{x}^{\mathrm{H}};$$
$$\mathcal{Q} = \operatorname{span}(\boldsymbol{y}),\ \mathbb{Q}_1 = \boldsymbol{y},\ \mathbb{Q} = \boldsymbol{y}\boldsymbol{y}^{\mathrm{H}}.$$

由定义 4.1 和定理 4.1 可知，两个一维子空间（或两个向量）的距离是

$$\operatorname{dist}(\boldsymbol{x},\boldsymbol{y}) = \operatorname{dist}(\mathcal{P},\mathcal{Q}) = \sqrt{1-(\boldsymbol{x}^{\top}\boldsymbol{y})^2} = |\sin\theta|, \tag{4.1.6}$$

其中 θ 是 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 的夹角。它表明：向量的（锐）夹角越小，它们的距离越小。

说明 4.1. 当 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 长度相同且夹角为锐时，可用 $\|\boldsymbol{x}-\boldsymbol{y}\|_2$ 刻画其距离。

[2]正交投影阵是幂等矩阵。

4.1.3 特征值的定位

矩阵元素可以直接确定特征值的范围。最简单结论是

$$|\lambda| \leq \varrho(\mathbb{A}) \leq \|\mathbb{A}\|,$$

其中 $\varrho(\mathbb{A})$ 是谱半径，$\|\mathbb{A}\|$ 是矩阵范数（通常是行范数和列范数）。换言之，特征值都落在以原点为圆心以 $\|\mathbb{A}\|$ 为半径的复圆盘上。

定理 4.2 (Gerschgorin 第一圆盘)**.** $\mathbb{A} = (a_{ij})_{n\times n}$ 的特征值至少落在复平面的某个圆盘 S_i 上，其中

$$S_i = \left\{ z \colon |z - a_{ii}| \leq \sum_{j\neq i} |a_{ij}| \right\}.$$

证明：对角占优矩阵可逆。 □

定理 4.3 (Gerschgorin 第二圆盘)**.** 若 m 个圆盘构成单联通集且与其它圆盘完全分开，则单联通集上恰好有 m 个特征值。

定理 4.4 (Gerschgorin 第三圆盘)**.** 设 $\mathbb{A}$ 不可约，λ 落在某个圆盘边界上。只有当每个圆盘边界都通过 λ 时，它才会成为特征值。

4.1.4 特征值的敏感度

定理 4.5. 矩阵特征值连续依赖于矩阵元素。

证明：显而易见，特征多项式 $f(z)$ 的系数是矩阵元素的连续（多项式）函数。利用复变函数的幅角原理，可知：位于简单闭曲线 γ 内部，多项式 $f(z)$ 的零点个数是

$$\frac{1}{2\pi\sqrt{-1}} \oint_\gamma \frac{f'(z)}{f(z)} \mathrm{d}z.$$

由于零点个数是离散的，所以它在足够小的闭曲线 γ 内部只能是常数。这意味着：只要系数扰动足够小，零点就不可能经过 γ 到达外部。换言之，零点连续依赖于系数，定理得证。 □

尽管如此，特征值的摄动表现非常复杂，存在明显的差异。举例说明，考虑简单的 n 阶 Jordan 矩阵

$$\begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & \cdots & \cdots & \cdots & \\ & & & 0 & 1 \\ 0 & & & & 0 \end{bmatrix}. \tag{4.1.7}$$

假设仅有一个位置产生 $\varepsilon > 0$ 的扰动，有如下的观察：当扰动出现在左下角时，某个特征值由零变成 $\sqrt[n]{\varepsilon}$，变化明显；当其逐渐往右上方向漂移时，特征值的变化越来越小；当其出现在对角线上方时，特征值不再发生变化。

上述现象可以利用特征值的摄动理论进行描述。因篇幅限制，本讲义仅就某些简单情形给出两种常用的刻画方式。

定理 4.6 (Bauer-Fike)**.** 已知两个矩阵 $\mathbb{A}$ 和 $\mathbb{B}$，其中 $\mathbb{A}$ 可通过 $\mathbb{Q}$ 相似变换为对角阵。任取 $\mathbb{B}$ 的某个特征值 $\mu \in \lambda(\mathbb{B})$，必存在 $\mathbb{A}$ 的某个特征值 $\lambda \in \lambda(\mathbb{A})$，使得

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

其中 $\|\cdot\|$ 是行范数（可推广到从属矩阵范数）。

证明：简单的特征信息描述，略。 □

定义 4.2. 利用 *Bauer-Fike* 定理，$\mathbb{A}$ 的特征值整体条件数是

$$\nu(\mathbb{A}) = \inf_{\mathbb{Q} \in \mathcal{D}_{\mathbb{A}}} \|\mathbb{Q}\|\|\mathbb{Q}^{-1}\|, \tag{4.1.8}$$

其中集合 $\mathcal{D}_{\mathbb{A}}$ 包含所有使 $\mathbb{A}$ 对角化的相似变换矩阵。

✯ **说明 4.2.** *Bauer-Fike* 定理可以推广到亏损矩阵，具体结果依旧同特征值整体条件数 (4.1.8) 正相关。它还与 *Jordan* 块最大阶数 p 相关，即 $\mathcal{O}(\varepsilon)$ 的矩阵扰动会带来 $\mathcal{O}(\varepsilon^{1/p})$ 的特征值扰动。

定理 4.7. 矩阵 $\mathbb{A}$ 非亏损 (或可相似对角化)。设 $(\lambda, \boldsymbol{x})$ 是近似特征信息，相应残量记为 $\boldsymbol{r} = \mathbb{A}\boldsymbol{x} - \lambda\boldsymbol{x}$，则存在某个特征值 $\lambda_i \in \lambda(\mathbb{A})$，使得

$$|\lambda - \lambda_i| \leq \nu(\mathbb{A}) \frac{\|\boldsymbol{r}\|_2}{\|\boldsymbol{x}\|_2}. \tag{4.1.9}$$

证明：注意到 $(\lambda, \boldsymbol{x})$ 是扰动矩阵的特征信息对，即

$$\left[\mathbb{A} - \frac{\boldsymbol{r}\boldsymbol{x}^H}{\|\boldsymbol{x}\|_2^2} - \lambda\mathbb{I}\right]\boldsymbol{x} = 0.$$

由 Bauer-Fike 定理，即证本结论。 □

✯ **说明 4.3.** 上述结论表明：对称矩阵的特征值条件数是最小的。若残量很小，则其特征值误差也很小。但是，相应的特征向量距离可能很大。具体实例是对称矩阵

$$\mathbb{A} = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & 1 \end{bmatrix}, \quad \varepsilon \neq 0,$$

具有特征值 $1 \pm \varepsilon$ 和特征向量 $(1, \pm 1)^\top$。取近似特征值 $\lambda = 1$ 和近似特征向量 $\boldsymbol{x} = (1, 0)^\top$，其残量是 $\boldsymbol{r} = (0, \varepsilon)^\top$。无论 ε 取值多么小，$\boldsymbol{x}$ 都不会近视平行于某个特征向量。

通常，不同的特征值具有不同的敏感程度。设 λ 是 $\mathbb{A}$ 的某个单特征值，相应的特征子空间是 $\mathrm{span}(\boldsymbol{x})$。任给扰动矩阵 $\mathbb{E}$，设 ε 是扰动参数，考虑特征值问题

$$(\mathbb{A} + \varepsilon\mathbb{E})\boldsymbol{x}(\varepsilon) = \lambda(\varepsilon)\boldsymbol{x}(\varepsilon).$$

显然，$\lambda(0)=\lambda$ 和 $\boldsymbol{x}(0)=\boldsymbol{x}$ 是 $\mathbb{A}$ 的特征信息。当 ε 充分小时，特征信息连续可导，简单计算可得

$$\lambda'(0)=\frac{\boldsymbol{y}^{\mathrm{H}}\mathbb{E}\boldsymbol{x}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}},$$

其中 $\boldsymbol{x}$ 是单位右特征向量，$\boldsymbol{y}$ 是单位左特征向量。

定义 4.3 (1972 年)**.** 设 λ 是 $\mathbb{A}$ 的某个单特征值，其特征值局部 (*Wilkinson*) 条件数是

$$W(\lambda;\mathbb{A})=\frac{1}{|\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}|},\tag{4.1.10}$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 分别是单位右特征向量和单位左特征向量。

说明 4.4. 局部条件数的定义没有要求 $\mathbb{A}$ 可对角化，仅仅要求 λ 是单根。对单特征值而言，必有 $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}\neq 0$。注意到反例

$$\mathbb{A}=\begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}$$

有 $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}=0$，不妨认为重特征值的局部条件数是无穷大。

关于两种特征值条件数，简要说明如下。

- 两种特征值条件数均为酉相似变换下的不变量。在计算矩阵特征值的时候，酉相似变换可以非常放心地进行。
- 对称矩阵的特征值问题永远都是良态，因为特征值条件数恒为 1；然而，相应的线性方程组可能是高度病态的。
- 亏损矩阵的特征值问题通常都是病态的；参见 (4.1.7)。

在后续的算法介绍时，我们将以实对称矩阵为主要计算目标。

4.1.5 特征向量的敏感度

同特征值相比，特征向量的扰动表现更加复杂。因课时限制，仅仅举例说明；详细内容可参阅 Wilkinson 的专著 [9]。

设 λ 是单特征值，可证：在适当条件下，特征向量的扰动距离反比于特征值的分离情况（对应量是 $\min_{\mu\neq\lambda}|\mu-\lambda|$）。换言之，当特征值非常靠近时，特征向量的计算效果将明显不如特征值。例如，二阶矩阵

$$\begin{bmatrix} 1.01 & 0.01 \\ 0.00 & 0.99 \end{bmatrix}$$

关于单特征值 $\lambda=0.99$ 的 Wilkinson 条件数约是 1.118，相应的特征向量约是 $(0.4472,-0.8944)^\top$。当右下角元素增加 0.01 时，相应的特征向量约是 $(0.7071,-0.7071)^\top$，变化极大。

✯ **说明 4.5.** 可以预见：对于重特征值，特征向量无论是否有亏损，都有可能因扰动而产生大变化。假设原始矩阵和扰动矩阵分别是

$$\mathbb{A}=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\quad \mathbb{E}=\begin{bmatrix} \varepsilon & \delta \\ 0 & 0 \end{bmatrix},$$

其中 ε 和 δ 不同时为零。简单计算，有如下结论：

1. 当 ε 和 δ 均不为零时，$\mathbb{A}+\mathbb{E}$ 的特征值是 1 和 $1+\varepsilon$，相应的特征向量是 $(\delta,-\varepsilon)^\top$ 和 $(1,0)^\top$。选取 ε 和 δ 的比值，可以使第一个特征向量指向任意方向。

2. 当 $\varepsilon=0$ 且 $\delta\neq 0$ 时，$\mathbb{A}+\mathbb{E}$ 的特征值保持不变，只有一个线性无关的特征向量。注意：$\mathbb{A}$ 具有两个线性无关的特征向量。

即使扰动矩阵是对称的，结论也是类似的；参见说明 *4.3*。

4.2 幂法

按模最大的特征值称为主特征值，相应的特征向量称为主特征向量；统称为主特征信息。在适当的条件下，幂法（Power method）可以简单快速地求出主特征信息。其实现方法和理论分析，在特征值问题的数值研究中均有着深远的影响。

4.2.1 正幂法

算法思想非常直观：不断地矩阵左乘，生成初始向量的 Krylov 序列。由于不同特征成分有不同的增长速度，主特征信息被凸显出来。

从最简单的情形入手。设矩阵 $\mathbb{A}$ 可以相似对角化，具有完备特征向量系 $\{\boldsymbol{x}_j\}_{j=1:n}$，主特征值 λ_1 完全分离，即（不计重数）

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|. \tag{4.2.11}$$

任取初始向量 $\boldsymbol{v}_0 \neq 0$，将其按特征向量系展开，简单计算可得

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

显然，$\lim_{k\to\infty} \mathbb{A}^k \boldsymbol{v}_0/\lambda_1^k = \alpha_1 \boldsymbol{x}_1$ 给出了主特征向量；但是，它无法直接应用，主要原因有两个。其一，主特征值是未知的，极限号里的表达式是不清楚的；其二，矩阵幂次运算会破坏矩阵的稀疏性，导致计算工作量和舍入误差无法承受。

✎ **论题 4.2.** 主要解决思路是采用递推技术，并随时进行适当的向量单位化，以避免数值计算“上下溢出”。幂法的基本结构是：对 $k \geq 1$，执行循环

$$\boxed{\boldsymbol{u}_k = \mathbb{A}\boldsymbol{v}_{k-1}, \quad m_k = \overline{\max}(\boldsymbol{u}_k), \quad \boldsymbol{v}_k = \boldsymbol{u}_k/m_k,}$$

其中 $\overline{\max}(\boldsymbol{a})$ 表示在向量 $\boldsymbol{a}$ 中按模最大的**首个分量**，例如

$$\boldsymbol{a} = (1, -3, 2, 3)^\top \quad \Rightarrow \quad \overline{\max}(\boldsymbol{a}) = -3.$$

定理 4.8. 在前面的假设条件下，若 $\alpha_1 \neq 0$，则幂法给出的 $\boldsymbol{v}_k$ 收敛[3]到主特征向量 $\boldsymbol{x}_1$，单位化指标 m_k 线性收敛到主特征值 λ_1，即

$$m_k = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right).$$

换言之，主次特征值的比率决定了幂法的收敛速度。

证明：利用格式和 $\max(\cdot)$ 函数的齐次性，可以建立

$$\boldsymbol{v}_k = \frac{\mathbb{A}^k\boldsymbol{v}_0}{m_k m_{k-1}\cdots m_1} = \frac{\mathbb{A}^k\boldsymbol{v}_0}{\overline{\max}(\mathbb{A}^k\boldsymbol{v}_0)},$$

即可证明上述结论。 □

☆ **说明 4.6.** 要幂法收集到主特征信息，初始向量在 $span(\boldsymbol{x}_1)$ 的投影必须非零，即 $\alpha_1 \neq 0$。

- 当其接近零时，计算机给出的 m_k 将缓慢收敛，甚至出现假收敛，即数值收敛到其它特征值。

- 事实上，即使 $\alpha_1 = 0$，将迭代不断地执行下去，舍入误差（经过很长时间！）的积累会慢慢产生积极作用：逐渐加快收敛速度，或调整到正确的收敛目标。

[3]其真正含义是 $\text{span}(\boldsymbol{v}_k) \to \text{span}(\boldsymbol{x}_1)$，或者等价于向量夹角趋于零。

通常，实际计算会选取至少三个线性无关的初始向量。若在指定步数内没有达到用户要求，则可认为不收敛。若两个或以上初始向量给出的数值结果相同，则可视其为主特征信息的正确近似。

幂法的收敛表现强烈依赖主特征信息的具体状态。当存在多个“等模”的主特征值时，即使特征向量系完备（矩阵非亏损），幂法或多或少都会遇到一些问题。

1. 主特征值是实重根，即 $\lambda_1 = \cdots = \lambda_r$。算法给出的 m_k 依旧收敛到主特征值，即

$$m_k = \lambda_1 + O\left(\left|\frac{\lambda_{r+1}}{\lambda_1}\right|^k\right).$$

   此时，$\boldsymbol{v}_k$ 仍然收敛到某个特征向量，但具体结果依赖于初始向量。

2. 主特征值是实的相反数，即 $\lambda_1 = -\lambda_2$。此时 m_k 不再收敛，算法需适当修正：连续执行两步幂法，即

$$\boldsymbol{u}_{2k+1} = \mathbb{A}\boldsymbol{v}_{2k}, \quad \boldsymbol{u}_{2k+2} = \mathbb{A}\boldsymbol{u}_{2k+1},$$
$$m_{2k+2} = \overline{\max}(\boldsymbol{u}_{2k+2}), \quad \boldsymbol{v}_{2k+2} = \boldsymbol{u}_{2k+2}/m_{2k+2}.$$

   修正算法给出的 m_{2k+2} 收敛到 λ_1^2。在确定近似特征值后，用 $\boldsymbol{u}_{2k+1}$ 和 $\boldsymbol{u}_{2k+2}$ 给出特征向量 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 的近似。

3. 主特征值是一对共轭复数，即 $\lambda_1^H = \lambda_2$。若初始向量和四则运算均限于实数域，幂法不可能收敛到复的特征信息，需要引入额外的技术手段。

   - 共轭复根满足二次实多项式 $\lambda^2 + p\lambda + q = 0$，相应的待定系数 p 和 q 可以按如下方式生成：

– 利用幂法提供的迭代序列，生成最小二乘问题（它有 2 个未知量和 n 个方程）

$$m_{k+2}m_{k+1}\boldsymbol{v}_{k+2} + p_{k+2}m_{k+1}\boldsymbol{v}_{k+1} + q_{k+2}\boldsymbol{v}_k = \mathbf{0},$$

求出相应的最小二乘解 p_{k+2} 和 q_{k+2}；

– 直至 p 和 q 趋于不变，给出相应的近似值。

- 利用共轭复根的实部和虚部，结合幂法的迭代向量，可以算出主特征信息的实部和虚部。计算公式见教科书。

由于涉及到最小二乘问题，幂法的计算效果和计算效率不够理想。特别地，当主特征值靠近实轴（虚部很小）时，数值精度很差。

综上所述，主特征信息状态的事前分析是决定幂法成效的关键步骤。

✯ **说明 4.7.** 当特征向量系不完备（矩阵亏损）时，幂法的收敛速度非常慢。举例说明，不妨考虑仅有特征值 $\lambda = \lambda_1$ 的 *Jordan* 矩阵 $\mathbb{A}$。任给非零向量 $\boldsymbol{v}_0$，计算 $\mathbb{A}^k\boldsymbol{v}_0$ 可知：幂法中的 m_k 和 $\boldsymbol{v}_k$ 以调和方式收敛到主特征信息 $(\lambda, \boldsymbol{e}_1)$，即

$$|m_k - \lambda_1| = \mathcal{O}(1/k), \quad dist(\mathbb{A}^k\boldsymbol{v}_0, \boldsymbol{e}_1) = \mathcal{O}(1/k).$$

❀ **思考 4.1.** 验证前面的例子及其结论；若有两个 *Jordan* 块，是否有类似的结论？

一般而言，当矩阵非亏损且主特征值是（按模分离的）单根时，幂法都是非常有效的首选方法。此外，对于稀疏矩阵，幂法的左乘运算具有极好的计算复杂度。事实上，幂法已经成功应用于互联网信息检索技术；相关细节可查阅资料，此处不再赘述。

4.2.2 加速技术

以下的讨论均默认幂法具有很好的收敛性。

✍ **论题 4.3.** 定理 *4.8* 表明，单位化指标 m_k 几何收敛到主特征值。相应的收敛速度称为线性（或一阶）的。对于线性收敛算法，*Aitken*（或称为 Δ^2）方法

$$\boxed{\tilde{m}_k = m_k - \frac{(\Delta m_k)^2}{\Delta^2 m_k}}$$

是行之有效的加速技术，其中

$$\Delta m_k = m_{k+1} - m_k, \quad \Delta^2 m_k = m_{k+2} - 2m_{k+1} + m_k$$

分别为一阶和二阶向前差分。参阅非线性方程求根部分，简单可证：

$$\lim_{k\to\infty} \frac{\tilde{m}_k - \lambda_1}{m_k - \lambda_1} = 0,$$

即 $\tilde{m}_k$ 比 m_k 更快地趋近主特征值。

✍ **论题 4.4.** 原点平移方法是简单易行和应用广泛的加速技术：引入平移量 μ，考虑平移矩阵 $\mathbb{A} - \mu\mathbb{I}$ 的特征值问题。通过 μ 的适当选取，期待更快的收敛速度。对幂法而言，平移量的设置要实现两个目标：

- $\lambda_1 - \mu$ 是 $\mathbb{A} - \mu\mathbb{I}$ 的主特征值；
- $\mathbb{A} - \mu\mathbb{I}$ 的前两个主特征值按模分离程度增大，其小于 1 的比值绝对值要尽可能小。

虽然原点平移策略在通常情况下很难实现，但是它可以成功应用于后面介绍的反幂法和 QR 方法等其它算法。

✍ **论题 4.5.** 对于实对称矩阵 $\mathbb{A}$，英国人 *L. Rayleigh*（*1870* 年）提出了 *Rayleigh*（或 *Rayleigh-Ritz*）商

$$R(\boldsymbol{x}) = \boldsymbol{x}^{\top}\mathbb{A}\boldsymbol{x}/\boldsymbol{x}^{\top}\boldsymbol{x}. \tag{4.2.12}$$

应用此技术，论题 *4.2* 的幂法可以任意替换单位化指标，通常修改为

$$\boldsymbol{u}_k = \mathbb{A}\boldsymbol{v}_{k-1}, \quad \boldsymbol{v}_k = \boldsymbol{u}_k/\|\boldsymbol{u}_k\|_2.$$

简单分析，有

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

同定理 *4.8* 相比，*Rayleigh* 商加速技术可以将主特征值的线性收敛速度提高至平方倍。

✯ **说明 4.8.** *Rayleigh* 商 $R(\boldsymbol{x})$ 是矛盾方程组 $\mu\boldsymbol{x} = \mathbb{A}\boldsymbol{x}$ 的极小最小二乘解，是基于 $span\{\boldsymbol{x}\}$ 空间所能给出的最佳特征值近似。此时，有 *Krylov-Bogoljubov-Weinstein* 不等式

$$|\lambda - R(\boldsymbol{x})| \leq \|\boldsymbol{r}\|_2,$$

其中 $\boldsymbol{r} = \mathbb{A}\boldsymbol{x} - R(\boldsymbol{x})\boldsymbol{x}$ 为残量。

Rayleigh 商也是重要的矩阵分析工具。下面给出相关的四个著名结论 [9]，其中特征值按照大小关系排序。

定理 4.9 (Courant-Fischer 极大极小)**.** 将实对称阵 $\mathbb{A}$ 的特征值按大小排序为 $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$，有

$$\mu_i = \min_{\dim \mathbb{V}=n+1-i} \max_{\boldsymbol{x}\in\mathbb{V}} R(\boldsymbol{x}) = \max_{\dim \mathbb{V}=i} \min_{\boldsymbol{x}\in\mathbb{V}} R(\boldsymbol{x}).$$

定理 4.10 (扰动估计)**.** 设 $\mathbb{A}$ 和 $\mathbb{B}$ 两个实对称阵的特征值分别是

$$\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n, \quad \nu_1 \geq \nu_2 \geq \cdots \geq \nu_n.$$

令 $\mathbb{E} = \mathbb{B} - \mathbb{A}$ 的最大特征值和最小特征值分别是 ε_1 和 ε_n，则有

$$\mu_i + \varepsilon_n \leq \nu_i \leq \mu_i + \varepsilon_1.$$

定理 4.11 (Weyl)**.** 作为扰动定理的简单推论，有 $|\mu_i - \nu_i| \leq \|\mathbb{E}\|_2$。

定理 4.12 (交错分布)**.** 设 $\mathbb{A}$ 是两个实对称矩阵，$\mathbb{U}$ 是 $n \times (n-1)$ 阶单位列直交阵，则 $\mathbb{A}$ 和 $\mathbb{B} = \mathbb{U}^\top \mathbb{A} \mathbb{U}$ 的的特征值满足交错分布，即

$$\mu_1 \geq \nu_1 \geq \mu_2 \geq \nu_2 \geq \cdots \geq \mu_{n-1} \geq \nu_{n-1} \geq \mu_n.$$

✯ **说明 4.9.** *Rayleigh* 商概念可以推广到复数阵 $\mathbb{A}$，它的值域

$$V(\mathbb{A}) = \left\{ R(\boldsymbol{x}) = \frac{\boldsymbol{x}^{\mathrm{H}} \mathbb{A} \boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}} \boldsymbol{x}} : \forall \boldsymbol{x} \right\}$$

构成复平面上的一个点集，具有如下性质：

1. $V(\mathbb{A})$ 是酉不变的，包含 $\mathbb{A}$ 的全部特征值；
2. $V(\mathbb{A})$ 是有界闭凸集。若 $\mathbb{A}$ 是规范矩阵，则 $V(\mathbb{A})$ 是以特征值为顶点的复平面单纯形。特别地，若 $\mathbb{A}$ 是 *Hermite* 实矩阵，$V(\mathbb{A})$ 是以最大特征值和最小特征值为端点的闭区间。

其他讨论略，可参阅相关文献。

4.2.3 反幂法

对于非奇异矩阵 $\mathbb{A}$，反幂法可以求解按模最小的特征值 λ_n 及其特征向量 $\boldsymbol{x}_n$。事实上，它就是逆矩阵 $\mathbb{A}^{-1}$ 的正幂法，即

$$\mathbb{A}\boldsymbol{u}_k = \boldsymbol{v}_{k-1}, \quad m_k = \overline{\max}(\boldsymbol{u}_k), \quad \boldsymbol{v}_k = \boldsymbol{u}_k/m_k.$$

类似前面的讨论可知：在适当条件下，$\boldsymbol{v}_k$ 收敛到特征向量 $\boldsymbol{x}_n$，且

$$m_k = \frac{1}{\lambda_n} + O\left(\left|\frac{\lambda_n}{\lambda_{n-1}}\right|^k\right).$$

✯ **说明 4.10.** 反幂法需要求解大量的同型线性方程组，单步迭代的计算复杂度必须降低。事先给出 LU 分解 $\mathbb{A} = \mathbb{L}\mathbb{U}$，每步迭代只用求解两个三角形方程组；既然初始向量可以任取，第一步迭代只需直接求解一个三角形方程组 $\mathbb{U}\boldsymbol{u}_1 = \boldsymbol{v}_0$。

反幂法的应用范围更广，相应的变形和改进也较多。例如，给定一个数 q，以其为固定平移量，可构造算法

$$(\mathbb{A} - q\mathbb{I})\boldsymbol{u}_k = \boldsymbol{v}_{k-1}, \quad m_k = \overline{\max}(\boldsymbol{u}_k), \quad \boldsymbol{v}_k = \boldsymbol{u}_k/m_k.$$

在适当的条件下，$q + m_k^{-1}$ 收敛到离 q 最近的某个特征值，$\boldsymbol{v}_k$ 收敛到相应的特征向量。当 q 是某个特征值的粗糙近似时，这个算法可以改善特征信息的精度。

✯ **说明 4.11.** 当 q 非常接近于某个特征值时，反幂法中出现的线性方程组通常是高度病态的，相应的 $\boldsymbol{u}_k$ 很难精确地数值计算出来。此时，有一个非常有趣的数值现象：舍入误差似乎提供了某种帮助，让反幂法呈现出“一步收敛”特性，即：除了第一步的改善非常显著，后续迭代的改善并不显著，甚至可能误差反弹。

这个数值现象有相应的理论阐述。为简单起见，下面以可对角化矩阵的单特征值计算为例：

1. 在第一步迭代中，舍入误差具有正面作用。简单地说，舍入误差主要体现在解向量在特征空间的投影长度，而解向量与特征空间的夹角可以得到某种程度的改善。对于特征向量的计算而言，这是有利因素，因为核心计算的目标是特征方向，不是特征长度。

2. 在第二步迭代中，舍入误差开始起到负面作用，解向量与特征空间的夹角开始变大。

相关讨论 *[1]* 要用到线性方程组的摄动理论、特征值条件数以及奇异值分解理论等等；因篇幅有限，详略。

✯ **说明 4.12.** 当 q 恰好是某个特征值时，相应的 *Gauss* 消元无法顺利执行到底。此时，不妨对 q 施加一个微小扰动，并以此为固定平移量执行反幂法。

用 Rayleigh 商作为动态平移量，由反幂法可以导出著名的 Rayleigh 商算法：任取初始向量 $\boldsymbol{q}_0$，记 $\mu_0 = R(\boldsymbol{q}_0)$；对 $k \geq 1$，执行循环

$$(\mathbb{A} - \mu_{k-1}\mathbb{I})\boldsymbol{u}_k = \boldsymbol{q}_{k-1}, \quad \boldsymbol{q}_k = \boldsymbol{u}_k/\|\boldsymbol{u}_k\|_2, \quad \mu_k = R(\boldsymbol{q}_k).$$

定理 4.7 表明，$\rho_k = \|(\mathbb{A} - \mu_k\mathbb{I})\boldsymbol{q}_k\|_2$ 可以刻画 μ_k 到某个特征值的收敛程度。通过详细和繁琐的理论分析 [1, 7]，有：

1. 对于单特征值信息（$\mathbb{A}$ 可以非对称），Rayleigh 商算法至少平方收敛，即 ρ_{k+1} 受控于 ρ_k^2 的某个倍数。

2. 当 $\mathbb{A}$ 是对称矩阵时，Rayleigh 商算法三次收敛，即 ρ_{k+1} 受控于 ρ_k^3 的某个倍数。

具体内容，略。

4.2.4 其它特征值的求解

幂法也可以解出其它重要特征信息，例如前 m 个按模互异的主特征信息。下面以 $m=2$ 为例，介绍“逐次”和“同时”两种策略。

基于收缩技术的逐次求解

收缩（Deflation）技术的策略是：利用主特征信息 $(\lambda_1, \boldsymbol{x}_1)$ 构造一个矩阵，使其主特征值恰好是原有问题的次特征值 λ_2。

论题 4.6. 降维收缩 技术就是利用相似变换和分块矩阵技术，从原有矩阵中剔除主特征值信息，构造一个包含其它所有特征信息的低阶矩阵。

核心操作是找到一个可逆矩阵 $\mathbb{S}$，将主特征向量 $\boldsymbol{x}_1$（实际上是幂法或其它方法给出的近似）转换为仅首个分量非零的向量，即

$$\mathbb{S}\boldsymbol{x}_1 = t\boldsymbol{e}_1.$$

它对应数值代数的基本问题，$\mathbb{S}$ 可以取做 Gauss 消去阵、Householder 镜像变换阵和 Givens 平面旋转阵。相似变换可得

$$\mathbb{S}^{-1}\mathbb{A}\mathbb{S} = \begin{bmatrix} \lambda_1 & \omega^{\top} \\ \mathbf{0} & \mathbb{B} \end{bmatrix},$$

其中 $n-1$ 阶矩阵 $\mathbb{B}$ 的主特征值就是 $\mathbb{A}$ 的次特征信息。整个计算过程包含两个技术细节，即

- 给出 $\mathbb{S}^{-1}\mathbb{A}\mathbb{S}$ 的快速算法；
- 建立高阶矩阵 $\mathbb{A}$ 和低阶矩阵 $\mathbb{B}$ 的特征向量联系。

详细工作参见教科书。

论题 4.7. 降维收缩技术会破坏矩阵稀疏结构。为克服这个缺陷，***Wieldant* 收缩** 技术引进秩一修正矩阵

$$\mathbb{A}_1 = \mathbb{A} - \sigma \boldsymbol{x}_1 \boldsymbol{v}^\top, \tag{4.2.13}$$

让 $\mathbb{A}$ 的次特征信息成为 $\mathbb{A}_1$ 的主特征信息。技术关键是 $(\sigma, \boldsymbol{v})$ 的设置，通常取刚刚得到的主特征信息 $(\lambda_1, \boldsymbol{x}_1)$。

Wieldant 收缩技术特别适合对称矩阵[4]，理论上可以将 $\mathbb{A}$ 的主特征值完全转化为零。此外，它还具有优势：不必计算和存储 $\mathbb{A}_1$，只需额外存储 σ 和 $\boldsymbol{v}$，相应乘法运算按照 (4.2.13) 的右侧表达式执行。

子空间同时迭代法

采用逐次求解策略，次特征信息的计算精度势必受限于主特征信息。为避免误差的累积和传递，自然的选择是希望同时求出前两个主特征信息，即所谓的子空间同时迭代方法。

论题 4.8. 子空间同时迭代方法包含两个基本操作，即矩阵左乘和 QR 分解。基本流程是：选取列直交矩阵 $\mathbb{V}_0 \in \mathbb{R}^{n\times m}$，对 $k \geq 1$ 执行循环

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

QR 分解是算法成功的关键。单纯执行矩阵左乘操作，$\mathbb{U}_k$ 的所有列向量都将趋于同一个主特征向量，导致列向量组的线性无关性在数值层面上越来越差。不断利用 QR 分解重构正交基底，可以避免维数出现数值坍塌。

[4]此方法也可用于非对称矩阵，相应的数值效果也不错。

✯ **说明 4.13.** 初始列直交阵有两种常用的生成方法。其一，随机给出一个非零向量，构造相应的 *Householder* 阵，再随机选出某些列；其二，随机生成一组列向量，检验它们的线性无关性，并进行 *Gram-Schmidt* 直交化。数值经验表明，后者的数值表现似乎好一些。

将 Rayleigh 商技术推广到多维子空间（或线性无关的多个向量），可以导出广义 Rayleigh 商技术。它等价于高维特征值问题局限到某个低维子空间的近似和求解，故也称子空间投影算法。

✎ **论题 4.9.** 子空间投影算法的基本思想：设低维空间是由一组列直交向量 $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m$ 张成的子空间，相应的矩阵表示是

$$\mathbb{V}_{k-1} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m].$$

将待解问题的特征信息局限于这个低维空间，基于投影技术或最小二乘思想，可得一个相对容易求解的低阶矩阵特征值问题

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

相应的 $\tilde{\lambda}_{k-1}$ 称为 *Ritz* 值，可视作 $\mathbb{A}$ 的某个特征值近似；$\mathbb{V}_{k-1}\boldsymbol{y}_{k-1}$ 称为 *Ritz* 向量，是相应的特征向量近似。

对于实对称矩阵，上述两个算法可以完美结合起来，形成子空间同时迭代的加速算法。具体实现过程和证明，可参见教科书。

定理 4.13. 若实对称矩阵的特征值（按模）互异，则子空间同时迭代的加速算法关于前 m 个主特征值信息均具有良好的收敛表现。

✯ **说明 4.14.** 二阶矩阵的特征值信息可以直接公式计算。事实上，当 k 充分大时，低阶矩阵是近似的对角阵，其特征信息可以用下一节的 *Jacobi* 方法快速求解。

4.3 Jacobi 方法

实对称阵可以直交相似对角化。《高等代数》课程给出的理论算法包含三个步骤：首先求出（高次）特征多项式的根，然后给出每个奇异方程组的基础解系，最后进行相应的 Gram-Schmildt 正交化。上述流程在计算机上根本无法实施，需要转换角度实现目标：

> 能否基于某类直交阵，通过系列的相似正交变换，逐步完成矩阵的对角化?

Jacobi 方法 (1846) 基于这个策略，利用 Givens 平面旋转阵，在一定程度上成功地实现了对称矩阵的直交相似对角化。Jacobi 方法能够同时算出全部特征信息，具有编程简单和高度并行等优点。

4.3.1 基本思想和计算公式

对于二阶矩阵，Jacobi 方法可以完美实现相似对角化目标。换言之，存在一个 Givens 平面旋转阵

$$\mathbb{G}(p,q) \equiv \mathbb{G}(p,q;\theta) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix},$$

相应的正交相似变换

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

实现非对角元素清零，即 $b_{pq} = 0$。对应此目标的操作称为 Jacobi 旋转，其中 a_{pq} 是旋转元素，θ 是旋转角度。

✍ **论题 4.10.** 利用相似变换的计算公式，简单推导可知

$$\cot 2\theta = \frac{a_{pp} - a_{qq}}{2a_{pq}} \equiv \xi. \tag{4.3.14}$$

通常要求 $\theta \in [-\pi/4, \pi/4]$，确保 $t = \tan\theta$ 的绝对值不超过 1。其理由稍后给出。

利用二次方程求根公式，由 (4.3.14) 可知

$$t = sgn(\xi)\Big[|\xi| + \sqrt{1+\xi^2}\Big]^{-1}, \tag{4.3.15}$$

进而导出 $\mathbb{G}(p,q)$ 的两个关键元素

$$c = \cos\theta = \frac{1}{\sqrt{1+t^2}}, \quad s = \sin\theta = ct. \tag{4.3.16}$$

✯ **说明 4.15.** 当 ξ 的绝对值很大时，上述计算过程遇到困难。相应的问题和解决方案主要有两种：

1. 当 ξ 的平方超过计算机最大浮点数时，(4.3.15) 存在开根号的问题。为此，将其修正为 $t = 1/(2\xi)$，再按照 (4.3.16) 计算 c 和 s；

2. 即使按照前一方案修正计算，给出的旋转角度也可能非常接近于零，数值结果总是 $c \approx 1$ 和 $s \approx 0$，相应的 *Jacobi* 旋转没有效果。此时，可以定义

$$T = \tan\frac{\phi}{2} = \frac{a_{pq}}{2(a_{pp} - a_{qq})}.$$

当 $\theta \to 0$ 时，可证 $\phi - \theta \approx 5\theta^3/4$。因此，可用 ϕ 替代 θ，利用万能公式进行修正计算，即

$$c = \cos\phi = \frac{1-T^2}{1+T^2}, \quad s = \sin\phi = \frac{2T}{1+T^2}. \tag{4.3.17}$$

✍ **论题 4.11.** *Jacobi 旋转可以分解为 Givens 平面旋转阵的左乘和右乘两次操作，仅仅位于同行或同列的“井字线”元素受到影响。*

- 位于非交叉位置的每个元素，需 *2* 次乘除运算即可得到；
- 位于对角线位置的两个元素，貌似需要 *6* 次乘除运算才能得到；事实上，它可以简化为 *1* 次乘法，相应的公式是

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

注意到矩阵的对称性，*Jacobi* 旋转操作需 $\mathcal{O}(4n)$ 次乘除运算。

✯ **说明 4.16.** 设 $\mathcal{E}$ 是用户指定的小量。只要 $|a_{pq}| < \mathcal{E}\sqrt{a_{pp}a_{qq}}$，即可数值上认定 $a_{pq} = 0$。

4.3.2 古典 Jacobi 方法

对于高阶矩阵，Jacobi 旋转策略遇到麻烦，（除特殊问题）它无法在有限步内实现相似对角化目标。理由很简单，位于同列（或同行）的 Jacobi 旋转会影响已经零化的非对角元，使其重新回到非零状态。我们不得不降低目标：构造一个迭代序列，近似实现直交相似对角化。

✍ **论题 4.12.** *古典 Jacobi 方法是最简单的解决方案。记 $\mathbb{A}_1 = \mathbb{A}$ 为原始矩阵；对 $k \geq 1$，执行循环：先搜索 $\mathbb{A}_k = (a_{ij}^{(k)})$ 的旋转主元*

$$a_{pq}^{(k)} = \arg\max_{i \neq j} |a_{ij}^{(k)}|, \tag{4.3.18}$$

再执行 Jacobi 旋转

$$\mathbb{A}_{k+1} = \mathbb{G}_k \mathbb{A}_k \mathbb{G}_k^\top, \tag{4.3.19}$$

其中 $\mathbb{G}_k = \mathbb{G}_k(p, q; \theta)$ 是 Givens 平面旋转阵，相应的旋转角度 θ 可按前面的公式计算。

定理 4.14. 在古典 *Jacobi* 方法中，$\mathbb{A}_k$ 本质收敛到对角阵，即其非对角部分 $\mathbb{E}_k$ 趋于零。本质收敛是指在集合意义下的收敛。

证明：估计 $\|\mathbb{E}_k\|_{\mathrm{F}}^2$ 经 Jacobi 旋转后的衰减比率。 □

✍ **论题 4.13.** 在适当的条件下，$|t| \leq 1$ 可以确保迭代序列 $\mathbb{A}_k$ 真正收敛到某个对角阵[5]。换言之，当迭代步数足够大时，指定位置的对角元不会随意跳转，而是稳定地趋向某个特征值。

不妨考虑简单情形，假设 $\mathbb{A}$ 的特征值 λ_i 互异。换言之，当古典 Jacobi 方法在 k 充分大之后，有 $\varepsilon > 0$，使每个 $(\lambda_i - \varepsilon, \lambda_i - \varepsilon)$ 内只有一个对角元。进一步，假设当前旋转主元满足 $|a_{pq}^{(k)}| < \varepsilon$，两个特征值满足 $|\lambda_p - \lambda_q| \geq 4\varepsilon$。注意到 $|\tan 2\theta| < 1$，简单计算可知

$$|a_{qq}^{(k+1)} - \lambda_p| \geq 4\varepsilon c^2 - 2\varepsilon = 2\varepsilon \cos 2\theta > \sqrt{2}\varepsilon,$$

其中 $|t| \leq 1$ 确保了 $\cos 2\theta \geq 0$。换言之，$a_{qq}^{(k+1)}$ 不会跳转到 λ_p 的所属区间。

☆ **说明 4.17.** 事实上，*Jacobi* 方法的渐近收敛表现要好于前面的估计。*Schonhage (1964)* 和 *Van Kempen (1966)* 指出：当 k 充分大时，*Jacobi* 方法平方收敛，即存在固定常数 C，使得

$$\|\mathbb{E}_{k+N}\|_F \leq C\|\mathbb{E}_k\|_F^2,$$

其中 $N = n(n-1)/2$。习惯上称 N 次 *Jacobi* 旋转为一次扫描。下面的

[5]证明用到如下结论：设 $\{\boldsymbol{u}_k\}_{k=0}^{\infty}$ 是有限维赋范空间的有界序列。若聚点有限且

$$\lim_{k\to\infty} \|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\| = 0,$$

则 $\boldsymbol{u}_k$ 收敛到某个聚点。

例子选自 *[11]*，考虑

$$\mathbb{A}=\begin{bmatrix}1&1&1&1\\1&2&3&4\\1&3&6&10\\1&4&10&20\end{bmatrix}$$

扫描次数	$\|\mathbb{E}\|_{\mathrm{F}}$	扫描次数	$\|\mathbb{E}\|_{\mathrm{F}}$
0	*E+2*	*3*	*E-11*
1	*E+1*	*4*	*E-17*
2	*E-2*		

目前还没有严格的理论能够预测达到用户缩减要求的最少扫描步数，但是 *Brent* 和 *Luk* (*1985*) 凭经验指出：扫描次数要同 $\log n$ 成正比例，其中 n 为矩阵阶数。实践经验表明，该结论似乎是正确的。

✯ **说明 4.18.** *Jacobi* 方法是数值稳定的。*Demmel* 和 *Veselić*(*1992*) 指出：对于正定矩阵，特征值相对误差可以被

$$\left|\frac{\lambda_{\mathrm{num}}-\lambda}{\lambda}\right|\approx\vartheta\kappa_2(\mathbb{D}^{-1/2}\mathbb{A}\mathbb{D}^{-1/2}),$$

其中 $\mathbb{D}=diag(\mathbb{A})$，$\vartheta$ 是机器精度，$\kappa_2(\cdot)$ 是谱条件数。

✯ **说明 4.19.** 若特征值互异，则 *Jacobi* 方法给出的特征向量也是收敛的。若特征值有重根，特征向量的收敛性不再保证，但特征子空间的收敛性依旧成立。

✍ **论题 4.14.** 设 $\{(p_\kappa,q_\kappa;c_\kappa,s_\kappa)\}_{\kappa=1:K}$ 是 *Jacobi* 旋转操作信息，其中 K 是操作次数。对应部分指定特征值的特征向量可以按照下面对递推方式快速恢复，即

1. 利用对应的 m 个单位列向量，构成列直交阵 $\mathbb{Q}_0\in\mathbb{R}^{n\times m}$；
2. 对 $\kappa=1:K$，计算 $\mathbb{Q}_\kappa=\mathbb{G}_\kappa\mathbb{Q}_{\kappa-1}$；

若只求解某个特征值的特征向量，更多采用带有偏移量的反幂法。由于“一步收敛”性质，其计算效率是比较高的。

4.3.3 循环 Jacobi 方法

在古典 Jacobi 方法中，主元搜索需要 $\mathcal{O}(n^2)$ 次判断，旋转操作仅需 $\mathcal{O}(n)$ 次乘除，计算复杂度呈现出“主次不清”的状态。常用的解决方式是放弃旋转主元的搜索，按固定顺序执行 **Jacobi 扫描**，即：从左到右、从上到下地执行 Jacobi 旋转，依次处理（严格下三角部分的）N 个非对角元。相应的算法称为（行）循环 Jacobi 方法。

✯ **说明 4.20.** *Wilkinson (1962)* 和 *Van Kempen (1966)* 指出：循环 *Jacobi* 方法也渐近平方收敛。

✎ **论题 4.15.** 循环 *Jacobi* 方法常常引入**阈值扫描策略**：只要非对角元按模小于阈值 δ_k，就直接跳过相关的 *Jacobi* 旋转。阈值设置如下：

- 取 $\delta_1 = \|E_0\|_F/\sigma$ 为初始阈值，其中 $\sigma \geq n$ 是用户指定的常数；
- 对 $k \geq 1$，按循环 *Jacobi* 方法执行 *Jacobi* 扫描，直至所有元素均被跳过（按模小于 δ_k）时，设置下一个阈值

$$\delta_{k+1} = \delta_k/\sigma, \quad k \geq 0.$$

❀ **思考 4.2.** 条件 $\sigma \geq n$ 是有意义的。证明：当此条件成立时，带阈值的循环 *Jacobi* 方法产生收敛的矩阵序列。

✯ **说明 4.21.** 虽然计算速度不如 *QR* 方法，循环 *Jacobi* 方法具有并行计算的优势。行列指标集 $\{1:n\}$ 可以轮换分组，使得扫描过程中的 *Givens* 平面旋转阵可以在不同的 *CPU* 上同时开展矩阵的左（或右）乘运算。

4.4 Givens-Householder 方法

Givens-Householder 方法也可用于实对称阵的特征值计算，其执行过程结合了两个重要的数值策略：一个是可以有限步完成的直交相似三对角化，另一个是基于 Sturm 序列的二分求根法。

4.4.1 直交相似三对角化

由对称矩阵出发，有限次直交相似变换能够实现的最简单结构是对称三对角阵。实现过程是标准化的，可采用 Givens 平面旋转和 Householder 镜像变换。

✎ **论题 4.16.** 假设左上角的 k 阶矩阵 $\mathbb{A}_k$ 已经实现了直交相似三对角化，右下角的 $n-k$ 阶矩阵是 $\mathbb{B}_{n-k}$，位于对角元 a_{kk} 下方的 $n-k$ 维向量是

$$\boldsymbol{a}_k = (a_{k+1,k}, a_{k+2,k}, \ldots, a_{n,k})^\top.$$

假设有 $n-k$ 阶直交阵 $\mathbb{Q}_{n-k}$，将 $\boldsymbol{a}_k$ 转化为仅首个分量非零的 $\mathbb{Q}_{n-k}\boldsymbol{a}_k$，相应的直交相似矩阵

$$\left[\begin{array}{cc:c} & & 0 \\ & \mathbb{A}_k & \\ & & (\mathbb{Q}_{n-k}\boldsymbol{a}_k)^\top \\ \hdashline 0 & \mathbb{Q}_{n-k}\boldsymbol{a}_k & \mathbb{Q}_{n-k}\mathbb{B}_{n-k}\mathbb{Q}_{n-k}^\top \end{array}\right],$$

完成了左上角 $k+1$ 阶矩阵的三对角化。$\mathbb{Q}_{n-k}$ 有两种生成方式：

- 若采用 *Givens* 平面旋转阵，可将旋转信息覆盖存储在原有位置。详细处理可参见说明 *3.11*。由于数值目标同 *Jacobi* 方法截然不同，旋转角度的计算公式是完全不同的。

- 若采用 *Householder* 镜像变换阵，可将关键信息覆盖保存在原有位置，变换后的副对角元素需额外开辟空间。

两种实现过程具有不同的计算复杂度。对于稠密矩阵而言，Householder 镜像变换比 Givens 平面旋转更具优势，乘除次数由 $\mathcal{O}(4n^3/3)$ 下降到 $\mathcal{O}(2n^3/3)$，开根次数也由 $n^2/2$ 下降到 $n-2$。只有当矩阵高度稀疏且要定点清零时，Givens 平面旋转才显现出优势。

4.4.2 Sturm 序列二分求根法

正交相似变换给出实对称三对角阵

$$\mathbb{T}_n = \text{symtridiag}(\{\alpha_i\}_{i=1}^n, \{\beta_i\}_{i=2}^n), \tag{4.4.20}$$

其中 α_i 是对角元，β_i 是副对角元。以下均假设 $\mathbb{T}_n$ 不可约，即副对角元均非零[6]。

$\mathbb{T}_n - \lambda\mathbb{I}_n$ 的各阶顺序主子式构成多项式序列

$$p_i(\lambda) = \det(\mathbb{T}_n - \lambda\mathbb{I}_n)(1:i, 1:i), \quad i = 1:n. \tag{4.4.21}$$

显然，$p_n(\lambda)$ 是 $\mathbb{T}_n$ 的特征多项式，其根是特征值。利用行列式按行（或列）展开，可得三项递推关系式

$$\boxed{p_i(\lambda) = (\alpha_i - \lambda)p_{i-1}(\lambda) - \beta_i^2 p_{i-2}(\lambda), \quad i = 2:n,}$$

其中 $p_1(\lambda) = \alpha_1 - \lambda$，并补充定义 $p_0(\lambda) = 1$。可证：

1. $\operatorname{sgn} p_i(-\infty) = 1$，$\operatorname{sgn} p_i(+\infty) = (-1)^i$；
2. 相邻多项式没有公共根；

[6]若某个 β_i 为零，则特征值问题可以分割为两个低阶矩阵的特征值问题。

3. 若 $p_i(\mu)=0$，则 $p_{i-1}(\mu)p_{i+1}(\mu)<0$；

4. $p_i(\lambda)$ 只有实的单根，将 $p_{i+1}(\lambda)$ 的根严格隔开。

任意给定实数 μ；在 **Sturm 序列** $\{p_i(\mu)\}_{i=0:n}$ 中，前 $k+1$ 个数中相邻符号相同的次数 $s_k(\mu)$ 称为**符号相同数**。特别规定：当 $p_i(\mu)=0$ 时，称 $p_i(\mu)$ 和 $p_{i-1}(\mu)$ 反号，$p_{i+1}(\mu)$ 和 $p_i(\mu)$ 同号。

定理 4.15. *多项式 $p_r(\lambda)$ 在 $(\mu,+\infty)$ 内恰有 $s_r(\mu)$ 个根。*

作为简单推论，$\mathbb{T}_n$ 在 $(a,b]$ 内恰有 $s_n(a)-s_n(b)$ 个特征值。利用二分技术折半缩减区间，第 k 个（从大到小排序）特征值可以采用如下方式解出：

1. 特征值的隔离：
   (a) 界定特征值的范围 $[a,b]$；
   (b) 取中点位置 $c=(a+b)/2$，计算符号相同数 $s_n(c)$。折半缩减区间 $[a,b]$ 到 $[a,c]$ 或 $[c,b]$，使左端点的符号相同数为 k，而右端点的符号相同数为 $k+1$。
2. 特征值的确定：
   (a) 继续区间等分操作，放弃端点符号相同数一致的折半区间，直到区间长度达到用户要求为止。

若运算过程都是精确的，则上述算法无条件收敛。由于舍入误差的影响，Sturm 序列的符号相同数可能出现计算偏差，导致相应算法陷入死循环。因此，停机指标不能设置太小，计算结果相对粗糙。

☆ **说明 4.22.** *对于高阶矩阵，Sturm 序列 $\{p_i(\mu)\}_{i=0}^n$ 的数值计算还存在上（下）溢出的风险。*

符号相同数的计算过程修正如下：令 $q_1(\mu)=p_1(\mu)$，计算

$$q_i(\mu)=\alpha_i-\mu-\frac{\beta_i^2}{q_{i-1}(\mu)},\quad i=2,3,\ldots,$$

其中 $q_i(\mu)$ 是比值 $p_i(\mu)/p_{i-1}(\mu)$ 或其极限，即

1. 若 $q_{i-1}(\mu)=0$，直接定义 $q_i(\mu)=-\infty$；
2. 若 $q_{i-1}(\mu)=-\infty$，直接定义 $q_i(\mu)=\alpha_i-\mu$。

序列 $\{q_i(\mu)\}_{i=1}^k$ 所含的非负元素个数就是符号相同数 $s_k(\mu)$。

论题 4.17. 在求出对称三对角阵 $\mathbb{T}_n$ 的特征值之后，利用线性方程组

$$(\mathbb{T}_n-\lambda\mathbb{I}_n)\,\boldsymbol{x}=\boldsymbol{0}$$

的直接法（令 $x_1=1$）可以给出 $\mathbb{T}_n$ 的特征向量。此法的数值稳定性较差。一般而言，带原点平移的反幂法更加有效，可以改善特征值和特征向量的计算精度。

利用三对角化过程中记录下来的变换信息，由 $\mathbb{T}_n$ 的特征向量快速重构出 $\mathbb{A}$ 的特征向量。

4.5 QR 方法

作为计算机时代数值计算的重大进展之一，QR 方法可以同时且高效地计算出矩阵的全部特征信息。它由 Francis 和 Kublanovskaya 在 1960 年代初期独立提出，其前身是基于三角分解的 LR 方法 (1958)，同 Schur 分解理论密切相关。

4.5.1 基本思想

✍ **论题 4.18.** *QR* 算法的基本结构很简单：记 $\mathbb{A}_1 = \mathbb{A}$；对 $k \geq 1$，依次执行直交分解和交换相乘，即

$$\boxed{\mathbb{A}_k = \mathbb{Q}_k\mathbb{R}_k, \quad \mathbb{A}_{k+1} = \mathbb{R}_k\mathbb{Q}_k,}$$

其中 $\mathbb{Q}_k$ 是正交阵，$\mathbb{R}_k$ 是上三角阵。显然，序列 $\{\mathbb{A}_k\}_{k=1}^{\infty}$ 中的任意矩阵都是直交相似，具有相同的特征值。

定理 4.16. 设 $\mathbb{A}$ 的特征值均为实数，按模严格分离。若以左特征向量为行组成的矩阵 $\mathbb{X}$ 具有 *LU* 分解，则 *QR* 方法给出的 $\mathbb{A}_k$ 本质收敛到上三角阵。

证明：类似于子空间同时迭代法的证明 [8]，详略。 □

✯ **说明 4.23.** 考虑具有等模特征值的二阶置换矩阵

$$\mathbb{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

QR 方法的迭代序列陷入循环状态。

✍ **论题 4.19.** *QR* 方法的收敛性同幂法具有密切联系，因为

$$\boxed{\mathbb{A}^k = \underbrace{\mathbb{Q}_1\mathbb{Q}_2\cdots\mathbb{Q}_k}_{\widetilde{\mathbb{Q}}_k}\underbrace{\mathbb{R}_k\cdots\mathbb{R}_2\mathbb{R}_1}_{\widetilde{\mathbb{R}}_k}, \quad \mathbb{A}_{k+1} = \widetilde{\mathbb{Q}}_k^{\top}\mathbb{A}_1\widetilde{\mathbb{Q}}_k.}$$

注意到 $\widetilde{\mathbb{R}}_k$ 的上三角结构，利用正幂法的收敛性结果可知

$$\boldsymbol{x}_1 \leftarrow \mathbb{A}^k\boldsymbol{e}_1 = \widetilde{\mathbb{Q}}_k\widetilde{\mathbb{R}}_k\boldsymbol{e}_1 = \widetilde{r}_{11}^{(k)}\widetilde{\mathbb{Q}}_k\boldsymbol{e}_1,$$

其中 $(\lambda_1, \boldsymbol{x}_1)$ 是主特征信息。注意到

$$(\mathbb{A}_{k+1} - \lambda_1 \mathbb{I})\boldsymbol{e}_1 = \widetilde{\mathbb{Q}}_k^\top (\mathbb{A} - \lambda_1 \mathbb{I}) \widetilde{\mathbb{Q}}_k \boldsymbol{e}_1 \to \boldsymbol{0},$$

可知 $\mathbb{A}_{k+1}$ 的第一列收敛到 $\lambda_1 \boldsymbol{e}_1$。类似地，利用反幂法即可给出最后一行的收敛情况。

✯ **说明 4.24.** 通常，迭代矩阵的最右下角元素收敛最快。

✯ **说明 4.25.** 事实上，*QR* 方法同 *Rayleight* 商迭代的联系更为紧密。一般而言，它至少具有平方收敛速度；对于实对称矩阵，它甚至可以达到三次方收敛速度。

4.5.2 实现细节

对于稠密矩阵 $\mathbb{A}$，直接应用 QR 方法的计算效率很差。下面给出常用的改良和加速技术。

✎ **论题 4.20.** 要提高单步迭代的效率，可以先执行上 *Hessenberg* 化（即正交相似变换到上 *Hessenberg* 阵），再执行 *QR* 方法。

上 *Hessenberg* 化需 $\mathcal{O}(5n^3/3)$ 次乘除运算。其操作过程同对称矩阵三对角化几乎一样，区别仅仅是上三角元素要花费时间来计算。

对于上 Hessenberg 阵 $\mathbb{A}_k$，其 QR 分解可以快速实现，即

$$\mathbb{G}_{n-1} \cdots \mathbb{G}_1 \mathbb{A}_k = \mathbb{R}_k,$$

其中 $\mathbb{G}_i = \mathbb{G}_i^{(k)}$ 是 Givens 平面旋转阵，用对角元将其下方的副对角元清零。无需计算上三角阵 $\mathbb{R}_k$，相应的 QR 迭代可以直接表示为

$$\mathbb{A}_{k+1} = \mathbb{G}_{n-1} \cdots \mathbb{G}_1 \mathbb{A}_k \mathbb{G}_1^\top \cdots \mathbb{G}_{n-1}^\top. \tag{4.5.22}$$

相应计算只需 $\mathcal{O}(n^2)$ 量级的乘除运算，计算效率令人满意。

(4.5.22) 可采用递推方式实现。例如，记 $\mathbb{A}_k^{(1)} = \mathbb{A}_k$，依次计算

$$\mathbb{A}_k^{(m+1)} = \mathbb{G}_m \mathbb{A}_k^{(m)} \mathbb{G}_m^\top, \quad m = 1 : n-1,$$

则最终给出的矩阵就是 $\mathbb{A}_{k+1}$，但中间矩阵不具有上 Hessenberg 结构。

若要中间矩阵保持上 Hessenberg 结构，递推过程可采用 **错位相乘技术**进行修正：

1. 左乘 $\mathbb{G}_1$，得到上 Hessenberg 矩阵 $\mathbb{G}_1\mathbb{A}_k$，且 $(2,1)$ 位已经清零；
2. 左乘 $\mathbb{G}_2$，再右乘 $\mathbb{G}_1^\top$，得到上 Hessenberg 矩阵 $\mathbb{G}_2\mathbb{G}_1\mathbb{A}_k\mathbb{G}_1^\top$；
3. 沿用上述思路，继续执行下去，直到左乘部分结束；
4. 右乘 $\mathbb{G}_{n-1}^\top$，完成整体操作。

计算过程也清楚说明：$\mathbb{A}_{k+1}$ 也是上 Hessenberg 矩阵。

论题 4.21. *QR 方法可以采用原点平移技术提高收敛速度。最简单的操作策略是单步位移：*

$$\boxed{\mathbb{A}_k - t_k\mathbb{I} = \mathbb{Q}_k\mathbb{R}_k, \quad \mathbb{A}_{k+1} = \mathbb{R}_k\mathbb{Q}_k + t_k\mathbb{I},}$$

其中 t_k 为位移量，常见的设置有两种。

1. *平凡位移量，即定义 $t_k = a_{nn}^{(k)}$ 是右下角元素。*
2. *Wilkinson 位移量：计算右下角二阶矩阵*

$$\begin{bmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{n,n}^{(k)} \end{bmatrix} \tag{4.5.23}$$

的两个特征值，用最靠近 $a_{nn}^{(k)}$ 的那个特征值作为平移量。它特别适合于对称三对角阵，简单计算可知

$$\boxed{t_k = a_{nn}^{(k)} + \alpha - sign(\alpha)\sqrt{\alpha^2 + \beta^2},}$$

其中 $\alpha = (a_{n-1,n-1}^{(k)} - a_{n,n}^{(k)})/2$ 和 $\beta = a_{n-1,n}^{(k)} = a_{n,n-1}^{(k)}$。

✯ **说明 4.26.** 上述位移策略也不保证 QR 方法一定收敛。譬如，采用平凡位移量，QR 方法求解

$$\mathbb{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

的迭代序列陷入循环状态。

✍ **论题 4.22.** 副对角元 $a_{n,n-1}^{(k)} \approx 0$ 的常用判断准则是

$$|a_{n,n-1}^{(k)}| \leq \mathcal{E} \min(|a_{n,n}^{(k)}|, |a_{n-1,n-1}^{(k)}|),$$

其中 $\mathcal{E}$ 是预设的指标。此时，$a_{nn}^{(k)}$ 可视为某个近似特征值。

副对角元 $a_{n-1,n-2}^{(k)} \approx 0$ 可类似判定。此时，可以直接解出 (4.5.23) 的两个特征值，作为 $\mathbb{A}$ 的近似特征值。

不断剔除右下角的一阶或二阶矩阵，通过缩减矩阵阶数可以获得更快的收敛表现。

4.5.3 隐式 QR 方法

类似于直交分解，上 Hessenberg 化也有类似的唯一性结果。

定理 4.17. *考虑 $\mathbb{A}$ 的两个上 Hessenberg 化*

$$\mathbb{U}^\top \mathbb{A} \mathbb{U} = \mathbb{H}, \quad \mathbb{V}^\top \mathbb{A} \mathbb{V} = \mathbb{G},$$

其中 $\mathbb{U}$ 和 $\mathbb{V}$ 都是直交阵，$\mathbb{H}$ 和 $\mathbb{G}$ 都是不可约的上 *Hessenberg* 阵。如果 $\mathbb{U}$ 和 $\mathbb{V}$ 的第一列是相同的，则整个操作过程可视为唯一的，即

$$\mathbb{U} = \mathbb{V}\mathbb{D}, \quad \mathbb{H} = \mathbb{D}\mathbb{G}\mathbb{D}, \quad \mathbb{D} = \mathrm{diag}\{\pm 1\}.$$

证明：略。 □

(4.5.22) 可视为 $\mathbb{A}_k$ 的一种上 Hessenberg 化方式。基于定理 4.17，隐式 QR 方法可以采用不同路径实现相同目标，即

> 第一个直交阵 $\mathbb{G}_1$ 得到保留，后续操作的直交阵可以采用其它生成方法。

对于上 Hessenberg 矩阵，每次执行 Givens 相似变换后，都会有且只有一个非零元素坠落到同列副对角线的下方位置。要将其清零，只需确定一个 Givens 平面旋转阵，利用副对角元素将坠落元素旋转为零，然后执行相应的矩阵右乘，完成直交相似变换。通过这样的操作，坠落位置将会移到下一列的副对角线下方位置。类似过程执行下去，整体的视觉效果就是坠落元素不断地从当前位置“下沉”到下一列，直至被“驱逐出境”。

论题 4.23. 设 $\mathbb{T}$ 是一个 n 阶不可约的实对称三对角阵，带原点位移加速的隐式 *QR* 算法可以按照下面代码实现：

1. 计算 *Wilkinson* 位移量 μ;

2. 定义临时变量 $x = \mathbb{T}(1,1) - \mu$ 和 $y = \mathbb{T}(2,1)$;

3. For $k = 1 : n-1$, *Do*

4. 计算 $[c, s] = \text{GIVENS}(x, y)$，将 $(x, y)^\top$ 中的 y 旋转为零; 相应的旋转阵记为 $\mathbb{G}(k, k+1)$;

5. 计算 $\mathbb{G}(k, k+1)\mathbb{T}\mathbb{G}(k, k+1)^\top$，并赋值给 $\mathbb{T}$;

6. 若 $k < n-1$，令 $x = \mathbb{T}(k+1, k)$ 和 $y = \mathbb{T}(k+2, k)$;

7. Enddo

若直接存储为两个向量，上述代码需要相应的修改；略。

4.5.4 双重位移 QR 方法

类似于幂法，要计算实矩阵的共轭复特征值，QR 方法的单步平移量必须设置为复数。

论题 4.24. 注意到实矩阵的共轭特征值含于 *Schur* 阵的二阶对角块中，连续两步的位移量可取为共轭复数，使迭代操作回归到实数运算。相应的方法称为双重位移 *QR* 方法，具体操作如下：

$$\mathbb{A}_{2k} - t_{2k}\mathbb{I} = \mathbb{Q}_{2k}\mathbb{R}_{2k}, \quad \mathbb{A}_{2k+1} = \mathbb{R}_k\mathbb{Q}_{2k} + t_{2k}\mathbb{I},$$
$$\mathbb{A}_{2k+1} - t_{2k}^{\mathrm{H}}\mathbb{I} = \mathbb{Q}_{2k+1}\mathbb{R}_{2k+1}, \quad \mathbb{A}_{2k+2} = \mathbb{R}_{2k+1}\mathbb{Q}_{2k+1} + t_{2k}^{\mathrm{H}}\mathbb{I},$$

其中 $\mathbb{A}_0 = \mathbb{A}$。简单计算，可知

$$\mathbb{Q}_{2k}\mathbb{Q}_{2k+1}\mathbb{R}_{2k+1}\mathbb{R}_{2k} = (\mathbb{A}_{2k} - t_{2k}^{\mathrm{H}}\mathbb{I})(\mathbb{A}_{2k} - t_{2k}\mathbb{I}) \tag{4.5.24}$$

是一个实矩阵。

直接使用双重位移 QR 方法，要计算 (4.5.24) 中的矩阵平方，相应的计算量和舍入误差都会变得很高。因此，需采用隐式 QR 算法进行优化，具体内容超出课程范围，详略。

✯ **说明 4.27.** *Matlab* 命令 *roots()* 给出多项式

$$p(x) = a_0 + a_1 x + \cdots a_{n-1} x^{n-1} + x^n$$

的所有零点，基于 *QR* 方法求解友矩阵

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

充分利用友矩阵的特点，计算复杂度还可降到 $\mathcal{O}(n^2)$；详略。

✯ **说明 4.28.** 对于大型稀疏矩阵的特征值问题，常用方法是投影算法，例如对称问题的 *Lanczos* 方法和非对称问题的 *Arnoldi* 方法。详细内容可参阅相关文献，此处略。

第 5 章
非线性方程的数值方法

非线性方程求根是常见的数值问题。很多实际问题（例如非线性偏微分方程和非线性最小二乘问题的数值求解）都会导出非线性方程

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0},$$

其中 $\boldsymbol{f}(\boldsymbol{x})\colon \mathbb{R}^n \to \mathbb{R}^n$ 是非线性（代数）映射。无论是成熟性还是有效性，非线性方程的研究都不如线性方程。在理论层面上，根的存在性、总数、重数和分布情况，还没有给出清楚完整的解答。在数值层面上，建立快速算法并给出完美分析，依旧极具挑战性和迫切性。本章重点介绍常用的非线性方程求根方法，特别是不动点迭代技术及著名的 Newton 方法。

5.1 基本概念

类似于线性方程组，非线性方程的 r 阶迭代方法也可表示为

$$\boldsymbol{x}_k = \boldsymbol{g}(\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}, \ldots, \boldsymbol{x}_{k-r}), \tag{5.1.1}$$

其中 $\boldsymbol{g}$ 是给定的迭代函数[1]，$\{\boldsymbol{x}_k\}_{k=0}^{r-1}$ 是人工给出的启动初值。同线性情形相比，以下概念需要强调和明确。

1. **迭代序列是确定的**，即迭代公式要保持有效，确保计算过程顺利进行。最简单的要求是 $\boldsymbol{g}$ 的值域包含于它的定义域。

[1]它可以与 k 有关。

2. 对于非线性问题，迭代序列的收敛性同初值密切相关。若收敛，相应的表现有两种状态。

   (a) **全局收敛**：初值可以全局（或大范围地）选取，相应的迭代序列均收敛；

   (b) **局部收敛**：只有当初值设置在某个小区域内，相应的迭代序列才会收敛。此时，相关的理论分析有两种模式。

      i. 若假定问题和方法在**真解附近**的局部信息，则相应的收敛性分析称为**局部收敛分析**。

      ii. 若仅仅假定问题和方法在**初值附近**的局部信息，则相应的收敛性分析称为**半局部收敛分析**。

   强调指出：对于线性方程组，迭代序列只有收敛和发散两种状态。若收敛，必全局收敛。

3. 收敛速度：

   设 $\boldsymbol{x}_\star$ 是某个真解，记 $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}_\star$ 为第 k 步迭代误差。当 k 充分大时，若有

   $$\|\boldsymbol{e}_{k+1}\| \leq C\|\boldsymbol{e}_k\|^p, \tag{5.1.2}$$

   其中 $\|\cdot\|$ 是某个向量范数[2]，$p \geq 1$ 和 $C > 0$ 是两个常数，则相应的收敛速度定义如下：

   (a) 当 $p > 1$ 时，称算法至少 p 阶收敛；

   (b) 当 $p = 1$ 时，还需额外要求 $C < 1$，称算法至少线性收敛。

   若收敛速度 p 不能被改善，则称算法是 p 阶的。

[2]对于标量方程，$\|\cdot\|$ 就是绝对值。

若下面的极限存在[3]，(5.1.2) 常常被简化为极限形式

$$\lim_{k\to\infty}\frac{\|\boldsymbol{e}_{k+1}\|}{\|\boldsymbol{e}_k\|^p}=C. \tag{5.1.3}$$

此时，C 可以是零。若 $C=0$，则称**超 p 阶收敛**；否则，称 p 阶收敛。对于单个方程，范数也可以抹去，相应的 C 允许为负。

4. 算法效率：

   迭代误差达到指定要求所消耗的计算时长，称为算法效率。它是评价算法优劣的重要指标。

   假设单步迭代的计算复杂度是 W，通常用乘除次数或 CPU 时间等信息来描述。对于 p 阶算法，相应的效率指标定义为

$$\eta=\begin{cases}\frac{1}{W}\ln p, & \text{若 } p>1;\\ \frac{1}{W}\ln C, & \text{若 } p=1,\end{cases}$$

   其中 C 是线性收敛概念中的参数。换言之，要构造高效算法只有两条途径，要么提高收敛阶，要么降低单步计算复杂度。

5. 数值稳定性：

   在非线性问题的实际计算中，舍入误差的影响也是不可避免的，甚至比线性情形更为严重。对于理论上收敛的算法，只有当它还具有良好的数值稳定性时，收敛表现和计算结果才能得到保障。相关的摄动分析非常困难和繁琐，本讲义不打算展开讨论，希望读者通过数值试验来体会这个现象。

[3]Ortega 和 Rheinboldt（1970 年）引进

$$Q_p=\limsup_{k\to\infty}\frac{\|\boldsymbol{e}_{k+1}\|}{\|\boldsymbol{e}_k\|^p},$$

称其为序列的商收敛因子或 Q 因子。

✯ **说明 5.1.** 对于非线性问题，停机准则的设置方式是类似的。常用准则有残量和相邻误差，即

$$\|\boldsymbol{f}(\boldsymbol{x}_k)\| \le \mathcal{E}, \quad \text{或者} \quad \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\| \le \mathcal{E},$$

其中 $\mathcal{E}$ 是用户指标。求解非线性问题的用户指标通常要略高一些。实际计算常常同时采用上述两种准则，并警惕假收敛现象。

5.2 标量方程的数值求解

本节集中讨论 $n = 1$ 的情形，介绍标量方程 $f(x) = 0$ 的数值求根方法。Matlab 命令 fzero() 可以求出位于猜测值附近的某个根。

5.2.1 区间二分法

✎ **论题 5.1.** 设 $f(x)$ 是连续函数。区间二分法是介值定理的简单应用：在区间折半的过程中保持端点值异号，通过长度趋零的区间套序列，求出 $f(x)$ 在给定区间的唯一实根。它理论上收敛，但误差的下降速度并不高，通常是线性收敛。

区间二分法简单易行，但不能计算复根和应用于非线性方程组。

✯ **说明 5.2.** 由于舍入误差的影响，端点函数值的计算可能不够准确。特别地，当区间端点趋近零点位置时，函数值的符号可能判断失误，进而导致结果错误。因此，区间二分法的精度要求不能太高，相应的停机标准不能设置过低。

✎ **论题 5.2.** 类似的计算方法还有**试位法**，即选取的中间位置不是区间 $[a, b]$ 的中点，而是在两个端点处的线性插值函数同 x 轴的交点

$$c = b - \frac{f(b)(b-a)}{f(b) - f(a)}.$$

其余的处理是类似的。通常，试位法的数值表现略好于区间二分法。

5.2.2 不动点迭代及加速技术

数值求根更多采用不动点迭代。将 $f(x)=0$ 等价变形为不动点方程 $x=g(x)$，进而构造出相应的不动点（或 Picard）迭代公式

$$x_{k+1}=g(x_k), \tag{5.2.4}$$

其中 $g(x)$ 称为（不动点）迭代函数。

关于收敛性的两个重要定理陈述如下。

定理 5.1 (压缩映像)**.** 称 $g(x)\colon [a,b]\to[a,b]$ 是一个压缩映射，若

> 存在 *Lip* 常数 $0\le L<1$，使得
> $$|g(x)-g(y)|\le L|x-y|,\quad \forall x,y\in[a,b].$$

对于任取的初值 $x_0\in[a,b]$，不动点迭代 (5.2.4) 给出的序列均线性收敛到真解 $x_\star$，且迭代误差 $e_k=x_k-x_\star$ 满足估计

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

定理 5.2. 若 $g(x)$ 在零点 $x_\star$ 的某个邻域内 m 阶连续可微，且

> $$g^{(j)}(x_\star)=0,\quad j=1:m-1;\qquad g^{(m)}(x_\star)\neq 0,$$

则不动点迭代 (5.2.4) 具有 m 阶局部收敛性。

下面讨论加速技术。

✍ **论题 5.3.** 若迭代序列 $\{x_k\}_{k=0}^{\infty}$ 线性收敛到 $x_\star$，*Aitken* 加速技术[4]可以给出收敛更快的迭代序列

$$\tilde{x}_k = x_k - \frac{(x_{k+1}-x_k)^2}{x_{k+2}-2x_{k+1}+x_k}. \tag{5.2.5}$$

换言之，若 $|C|<1$，则有结论（要求 $x_k \neq x_\star$）

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

✯ **说明 5.3.** 论题中的条件 $|C|<1$ 是必需的。若 $C=1$，则 *Aitken* 方法可能不会给出明显的加速效果，例如序列 $x_k = 1/k$。

✍ **论题 5.4.** 局部应用 *Aitken* 加速技术，可形成 *Steffensen* 迭代法，相应的迭代函数是

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

其几何解释是，对残量函数 $g(x)-x$ 在 x_k 和 $g(x_k)$ 两个位置进行线性插值，利用直线的零点 x_{k+1} 给出残量函数的零点近似。可证：在适当的条件下，*Steffensen* 方法局部平方收敛。

5.2.3 切线法

切线法是著名的非线性方程求根方法，原始思想由 Vieta 在 1600 年左右提出；Newton 在 1664 年知晓 Vieta 的工作，并于 1669 年解出了三次多项式 $x^3-2x-5=0$ 的最大实根。基本操作就是逐位试根，依次确定 $x_\star = 2.d_1d_2d_3\cdots$ 的每位数字。在 1690 年，Raphson 将 Newton

[4]该技术曾用于幂法的加速。

的求根方法略作修改，并重新发表。事实上，Simpson（1710-1761）给出的版本更加接近现在的描述。

✎ **论题 5.5.** *切线法就是 Newton-Raphson 方法，或更多地简称为 Newton 方法。迭代公式非常简单，即*

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

其几何含义是利用当前位置的切线方程进行曲线的局部线性化，并将线性函数的根作为新的零点近似。特别指出：局部线性化思想在非线性问题中应用非常广泛。

Newton 方法的收敛速度同零点 $x_\star$ 的性质相关。假设 $f(x)$ 在 $x_\star$ 附近足够光滑[5]，相应的结论有

定理 5.3. *若 $x_\star$ 是单根，则 Newton 方法局部平方收敛。*

定理 5.4. *若 $x_\star$ 是 m 重根，则 Newton 方法局部线性收敛，迭代误差的渐近下降速度是 $1 - m^{-1}$.*

✯ **说明 5.4.** *当 x_k 趋向重根 $x_\star$ 时，位于分母位置的 $f'(\cdot)$ 趋于零，相应的舍入误差干扰变得越来越严重。*

✎ **论题 5.6.** *经适当修正，Newton 方法对于重根也可获得高阶局部收敛。具体实现方式有：*

1. *若重数 m 是已知的，则算法可以简单修正为*

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

[5]Newton 方法可推广到非光滑函数；超出课程范围，详略。

2. 当重数 m 未知时，有两种解决方法。

- 利用 $F(x) = f(x)/f'(x)$ 滤掉重根，将问题转化为 $F(x) = 0$ 的求解。相应的 *Newton* 迭代包含二阶导数的计算，即
$$x_{k+1} = x_k - \frac{f(x_k)f'(x_k)}{[f'(x_k)]^2 - f(x_k)f''(x_k)}.$$
- 若要避免出现二阶导数，可以采用 *Newton* 方法的 *Steffensen* 加速。

3. 事实上，重数 m 可以自动探测：计算标准 *Newton* 解的重根指标
$$h(x_k) = \frac{\ln|f(x_k)|}{\ln|f(x_k)| - \ln|f'(x_k)|}.$$
当其取值稳定时，它必定趋向于 m；此时，对 $h(x_k)$ 取整，并跳转到算法 *1* 即可.

在适当条件下，Newton 方法可以实现全局（或大范围）收敛。主要结论和具体应用陈述如下。

定理 5.5. 假设函数 $f(x) : [a, b] \to \mathbb{R}$ 满足：

> *1.* 单调保凸，$f(a)$ 和 $f(b)$ 异号；
> *2.* 从两个端点出发的迭代位置依旧落在 $[a, b]$ 上，

那么只要初值落在 $[a, b]$ 上，*Newton* 方法都是收敛的。

✎ **论题 5.7.** 定理 *5.5* 给出 *Newton* 迭代的两个重要应用：

> *1.* 根号 $\sqrt{a}$ 的计算：$x_{k+1} = \frac{1}{2}(x_k + \frac{a}{x_k})$；
> *2.* 用加减乘计算倒数 $1/a$：$x_{k+1} = 2x_k - ax_k^2$；

请论证它们是否全局（或大范围）收敛。

5.2.4 割线法

✎ **论题 5.8.** 利用最近的历史数据 x_{k-1} 和 x_k，对 $f(x)$ 进行局部线性插值近似，并以其零点作为新的迭代位置，可得

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

该方法称为弦截法（或割线法），可视为 *Newton* 方法中的一阶导数替换为一阶差商近似，回避了导数 $f'(\cdot)$ 的计算困难。

定理 5.6. 割线法的收敛速度稍慢于 *Newton* 法。在适当条件下，收敛阶达到黄金分割值，约 *1.618.*

证明：见教科书。 □

✯ **说明 5.5.** 即使待解零点是一个单根，割线法也可能遇到分母为零，导致算法意外停机。此时需要给出新的猜测位置，再次启动计算流程。

5.2.5 高次多项式求根

前面给出的算法均可用于多项式求根。计算过程包含多项式函数值和导数值的大量计算，相应的计算复杂度需要尽量降低。

✎ **论题 5.9.** *Horner* 算法或秦九韶算法是常用的高效算法。设

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0, \tag{5.2.6}$$

利用多项式除法

$$p(x) = (x - \mu) g(x) + b_0, \quad g(x) = \sum_{i=1}^{n} b_i x^{i-1},$$

可得函数值 $p(\mu)=b_0$，其中 $\{b_i\}_{i=0}^n$ 的计算公式是

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

由于 $p'(\mu)=g(\mu)$，导数值的计算再次转化为多项式的取值，可以仿照前面处理。详略。

抛物线法（或 Müller 方法）的设计思想类似于弦截法。

✎ **论题 5.10.** 基本操作如下：假设当前的三个历史位置

$$A(x_{k-2},f(x_{k-2})),\quad B(x_{k-1},f(x_{k-1})),\quad C(x_k,f(x_k))$$

可以确定非退化的抛物线，将最靠近 x_k 的抛物线零点作为新的迭代位置 x_{k+1}。

理论可证：在适当的条件下，*Müller* 方法是超线性局部收敛的，其收敛阶约 *1.840*，即 $\lambda^3-(\lambda^2+\lambda+1)=0$ 的唯一正实根。

✯ **说明 5.6.** 抛物线零点可以是复数，故 *Müller* 方法也可求解实系数多项式的共轭复根。

✯ **说明 5.7.** 多项式求根可以转化为矩阵特征值问题，在 *Matlab* 中的命令是 *root()*。除此之外，关于多项式的求根，还有很多专门设计的特殊算法。因篇幅限制，本课程不做展开。

✯ **说明 5.8.** 直接利用系数确定实系数多项式 $p(x)$ 的实根位置，是一个历史悠久的研究课题。相关的著名结论有

1. *Langrange* 法：假设 $a_n>0$。设 a_{n-k} 是（按降幂方式排列的）首个负系数，b 是所有负系数的最大模，则正根上限为 $1+(b/a_n)^{1/k}$.

2. *Sturm* 序列法（*1829* 年）：令 $f_0(x)=p(x)$ 和 $f_1(x)=p'(x)$，辗转相除可得 *Sturm* 序列

$$f_{k-1}(x)=f_k(x)q_k(x)-f_{k+1}(x),\quad k=1:m,$$

它们在 μ 点的相邻数符号变化次数（删除掉零值）表示严格大于 μ 的实根数目。

3. *Descartes* 符号律：将实系数多项式按降幂方式排列，则它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。

详细内容，请查阅相关文献；此处不再赘述。

5.3 向量方程的数值求解

除区间二分法，前一节的求根方法[6]基本都可以由标量方程推广到方程组（或向量方程），但数值表现还缺乏足够完善的理论保障。本节关注 Newton 方法的基本理论及其各种改良。

5.3.1 向量值函数的基本理论

作为多元函数的简单推广，向量值函数

$$\boldsymbol{f}(\boldsymbol{x})=\{f_i(x_1,x_2,\ldots,x_n)\}_{i=1}^m:\mathbb{R}^n\to\mathbb{R}^m$$

也有类似的微积分理论。重点内容简述如下，详细内容见教科书。

定义 5.1. 给定位置 $\boldsymbol{x}$ 和方向 $\boldsymbol{\eta}$，相应的 *Gateaux* 导数是

$$D\boldsymbol{f}(\boldsymbol{x})(\boldsymbol{\eta})=\lim_{t\to0}\frac{\boldsymbol{f}(\boldsymbol{x}+t\boldsymbol{\eta})-\boldsymbol{f}(\boldsymbol{x})}{t}.$$

[6]将线性方程组和非线性标量方程的迭代求解技术结合起来，也可建立非线性方程组的求根算法。因篇幅限制，本节对此不做深入讨论。

它可视为多元函数方向导数的直接推广。若沿任何方向都 *Gateaux* 可导，则称 $D\boldsymbol{f}(\boldsymbol{x})\colon \mathbb{R}^n \to \mathbb{R}^m$ 是 $\boldsymbol{f}$ 在该点的 *Gateaux* 导数。

定义 5.2. 给定位置 $\boldsymbol{x}$。若有线性映射 $\boldsymbol{f}'(\boldsymbol{x})\colon \mathbb{R}^n \to \mathbb{R}^m$，使得

$$\lim_{\|\triangle \boldsymbol{x}\|\to 0} \frac{\|\boldsymbol{f}(\boldsymbol{x}+\triangle\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}'(\boldsymbol{x})\triangle\boldsymbol{x}\|}{\|\triangle\boldsymbol{x}\|} = 0,$$

则称 $\boldsymbol{f}'(\boldsymbol{x})$ 是 $\boldsymbol{f}$ 在该点的 *Frechét* 导数（或 *Frechét* 可微）。

定理 5.7. *Frechét* 可微必 *Gateaux* 可导，且

$$\boldsymbol{f}'(\boldsymbol{x}) = D\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}(\boldsymbol{x}) \tag{5.3.7}$$

是 $m \times n$ 阶 *Jacobi* 矩阵。若上面的一阶偏导数在 $\boldsymbol{x}$ 点均连续，则 $\boldsymbol{f}$ 在该点 *Frechét* 可微。

若 $f(\boldsymbol{x})$ 是（标量）多元函数，有 $f'(\boldsymbol{x}) = [\nabla f(\boldsymbol{x})]^\top$。

定理 5.8. *Frechét* 可微必然连续。

定义 5.3. 设 $\boldsymbol{f}(s)\colon [0,1] \to \mathbb{R}^m$，其定积分是

$$\int_0^1 \boldsymbol{f}(s)\,\mathrm{d}s = \left(\int_0^1 f_i(s)\,\mathrm{d}s\right)_{i=1:m},$$

由每个分量函数的定积分构成。

常用的分析工具有**积分不等式**：对于任意的向量范数 $\|\cdot\|$，均有

$$\left\|\int_0^1 \boldsymbol{f}(s)\,\mathrm{d}s\right\| \le \int_0^1 \|\boldsymbol{f}(s)\|\,\mathrm{d}s. \tag{5.3.8}$$

不同于多元函数，向量值函数**没有**微分中值定理，

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

事实上，局部位置 $\boldsymbol{\xi}$ 不一定存在，因为每个 $f_i(\boldsymbol{x})$ 对应的微分中值位置可能不同。上述差距常常表示为一个直线积分，即

$$\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{x}) = \int_0^1 \boldsymbol{f}'(\boldsymbol{x} + s(\boldsymbol{y} - \boldsymbol{x}))\,\mathrm{d}s \cdot (\boldsymbol{y} - \boldsymbol{x}). \tag{5.3.9}$$

定理 5.9. 设 $\boldsymbol{f}$ 在凸集 Ω 上 *Frechét* 可微，且 $\boldsymbol{f}'$ 是 *Lip* 连续的，即

$$\|\boldsymbol{f}'(\boldsymbol{y}) - \boldsymbol{f}'(\boldsymbol{x})\| \leq \gamma \|\boldsymbol{y} - \boldsymbol{x}\|, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \Omega,$$

其中 γ 是固定常数，则有估计

$$\|\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}'(\boldsymbol{x})(\boldsymbol{y} - \boldsymbol{x})\| \leq \frac{\gamma}{2} \|\boldsymbol{y} - \boldsymbol{x}\|^2, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \Omega.$$

✯ **说明 5.9.** 上述定理可视为 *Taylor* 公式的推广。

5.3.2 不动点迭代和 Newton 方法

非线性方程组的主流求根方法是不动点迭代

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k),$$

相应的主要分析工具是压缩映像原理。

定理 5.10 (见教科书的定理 1)**.** 若在以不动点 $\boldsymbol{x}_\star$ 为中心的某个开球内，不动点函数 $\boldsymbol{g}(\cdot)$ 满足中心压缩条件，则相应的不动点算法至少线性收敛。

定理 5.11 (见教科书的定理 2)**.** 设不动点函数是定义在某个闭集上自身到自身的压缩映射，则初值只要落在闭集上，相应的不动点算法至少线性收敛到其唯一的不动点。

对于非线性方程组，Newton 方法形式不变，也可表示为

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

换言之，每步迭代都要构造和求解一个（系数矩阵常常是稠密的）线性方程组。它是一个不动点迭代，相应的迭代函数是

$$\boldsymbol{g}(x) = \boldsymbol{x} - [\boldsymbol{f}'(\boldsymbol{x})]^{-1}\boldsymbol{f}(\boldsymbol{x}). \tag{5.3.10}$$

收敛分析一直备受关注。Cauchy（1829）和 Runge（1899）分别给出了 $n = 1$ 和 $n \geq 2$ 的局部收敛性分析; Fine (1916) 给出了半局部收敛分析。著名工作还有 Ostrowski (1936)、Willers (1938) 和 Kantovich (1948) 的证明。

定理 5.12 (局部收敛分析)**.** 设 $\boldsymbol{f}(\boldsymbol{x}_\star) = \boldsymbol{0}$ 且 *Frechét* 导数 $\boldsymbol{f}'(\boldsymbol{x}_\star)$ 非奇异。若 $\boldsymbol{f}$ 在 $\boldsymbol{x}_\star$ 的某个开球内连续可微，则 *Newton* 方法局部超线性收敛。若 $\boldsymbol{f}'$ 还在 $\boldsymbol{x}_\star$ 附近 *Lipschitz* 连续，则 *Newton* 方法至少局部平方收敛。

☆ **说明 5.10.** 设序列 $\{\boldsymbol{x}_k\}$ 超线性收敛到 $\boldsymbol{x}_\star$，则其必满足

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|}{\|\boldsymbol{x}_k - \boldsymbol{x}_\star\|} = 1.$$

因此说，*Newton* 方法的相邻误差可以估算迭代误差。特别指出：上述结论的逆命题不成立，例如奇偶子列分别定义为

$$\boldsymbol{x}_{2k+1} = \frac{2}{(2k)!}, \quad \boldsymbol{x}_{2k} = \frac{1}{(2k)!}.$$

定理 5.13 (半局部收敛分析)**.** 设 $\boldsymbol{f}(\boldsymbol{x})$ 在某个闭凸集 Ω 上 *Frechét* 可微，存在三个参数 α, β 和 γ 使得

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

若 $h = \alpha\beta\gamma/2 < 1$，则 *Newton* 迭代序列至少平方收敛到 $S_r(\boldsymbol{x}_0) \subset \Omega$ 内的唯一解 $\boldsymbol{x}_\star$，其中 $S_r(\boldsymbol{x}_0)$ 是以 $\boldsymbol{x}_0$ 为中心和 $r = \alpha/(1-h)$ 为半径的开球。

✯ **说明 5.11.** 初始向量 $\boldsymbol{x}_0$ 的选取是 *Newton* 方法实际应用的一个难点。由半局部收敛分析过程可知，定理 *5.13* 中的收敛条件成立的一个必要条件是

$$\|\boldsymbol{x}_2 - \boldsymbol{x}_1\| < \|\boldsymbol{x}_1 - \boldsymbol{x}_0\|. \tag{5.3.11}$$

若初始两步的计算表明 (5.3.11) 不成立，则应停止计算，放弃这个没有前途的初始向量。

✯ **说明 5.12.** 由于 $\boldsymbol{x}_{k+1}$ 仅依赖当前位置 $\boldsymbol{x}_k$，*Newton* 方法是一个自校正算法，相应的计算结果不受历史记录的影响。

✯ **说明 5.13.** 引入仿射变换，考虑 $\boldsymbol{g}(\boldsymbol{x}) = \mathbb{A}\boldsymbol{f}(\boldsymbol{x})$ 的求根问题，其中 $\mathbb{A}$ 是非奇异矩阵。注意到仿射变换不会影响 *Newton* 方法的迭代序列，定理 *5.13* 的条件可以借此得到改善。

✯ **说明 5.14.** 为克服线性方程组系数矩阵接近奇异带来的麻烦，可采用基于 *Tiknonov* 正则化技术的修正算法：

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

其中 $\lambda_k > 0$ 是适当选取的阻尼因子，改善系数矩阵的属性，例如对角占优或正定等等。

✯ **说明 5.15.** 逐次缩减 *Newton* 位移，不断检测向量值函数的范数（进行所谓的线性搜索），可以构造出“盲人下山法”或 *Newton* 下降法，实现大范围收敛。基本描述如下：

1. $\boldsymbol{f}'(\boldsymbol{x}_k)\Delta\boldsymbol{x}_k = -\boldsymbol{f}(\boldsymbol{x}_k)$；令 $\ell = 0$ 和 $\lambda_\ell = 1$;
2. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda_\ell\Delta\boldsymbol{x}_k$; 若 $\|\boldsymbol{f}(\boldsymbol{x}_{k+1})\| \geq \|\boldsymbol{f}(\boldsymbol{x}_k)\|$，则
   - $\lambda_{\ell+1} = \lambda_\ell/2$ 和 $\ell := \ell + 1$;
   - 若 λ_ℓ 太小，则重启算法，再次（随机）选择迭代初值；否则，继续循环；
3. 回到第一步，开始 *Newton* 迭代的下一步；

通常，用该算法给出某个真解的大概位置，期待后续进行的 *Newton* 方法获得快速的平方收敛。

5.3.3 修正 Newton 法

沙文思基（1967）提出了修正 Newton 法，通过导数矩阵的局部锁定，节省线性方程组的构造代价和求解时间。对于给定的正整数 m，其基本结构是

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

由于线性方程组是局部同型的，若事先得到系数矩阵的 LU 分解，则 m 步后续迭代只需不断地求解三角形方程组即可。在实际应用中，$m = 2$ 较为常用。

定理 5.14. *若在定理 5.12 的条件下 Newton 迭代平方收敛，则相应的修正 Newton 法是 $m+1$ 阶收敛的。*

证明：数学归纳，证明思路是类似的。 □

☆ **说明 5.16.** *假设函数值和导数值的计算工作量是 $1/n$，矩阵求逆的计算工作量是 ν，则 Newton 方法和修正方法的算法效率分别是*

$$\eta_1 = \frac{\ln 2}{n+1+\nu}, \quad \eta_m = \frac{\ln(m+1)}{n+m+\nu}.$$

当 m 适当大时，修正 Newton 方法可获得更好的算法效率。

5.3.4 割线法

割线法与 Newton 方法的区别是切平面改为割平面，利用历史数据近似 Jacobi 矩阵，回避大量导数值的计算。依据构造思想和实现方法，割线法分为两类：

✎ **论题 5.11.** *直接用差商矩阵 $\mathbb{J}(\boldsymbol{x}_k, \mathbb{H}_k)$ 代替 Jacobi 矩阵，可得* **离散（或单点）*Newton* 法**：

$$\boxed{\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \Big[\mathbb{J}(\boldsymbol{x}_k, \boldsymbol{h}_k)\Big]^{-1} \boldsymbol{f}(\boldsymbol{x}_k),}$$

其中 $\mathbb{H}_k = \{h_{ij}^{(k)}\}$ 是趋于零的参数矩阵，差商矩阵的每个元素是

$$\mathbb{J}(\boldsymbol{x}_k, \mathbb{H}_k)_{ij} = \frac{1}{h_{ij}^{(k)}}\Big[\boldsymbol{f}_i(\boldsymbol{x}_k + h_{ij}^{(k)}\boldsymbol{e}_j) - \boldsymbol{f}_i(\boldsymbol{x}_k)\Big].$$

通常，事先给定参数 $\bar{h}_{ij}$，设置

$$h_{ij}^{(k)} = \bar{h}_{ij}\|\boldsymbol{f}(\boldsymbol{x}_k)\|.$$

为简单起见，习惯上取 $\bar{h}_{ij} = \bar{h}_i$。

定理 5.15. *在定理 5.12 的条件下，离散 Newton 法的收敛表现与 Newton 法相同。*

✍ **论题 5.12.** 利用线性插值或者平面化思想，当前位置的非线性问题可以局部线性化。采用历史数据进行操作，可得**割线法**：

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

其中 $\mathbf{H}_k$ 和 $\boldsymbol{\Gamma}_k$ 是由 $n+1$ 个辅助点 $\{\boldsymbol{x}_{k,\ell}, \boldsymbol{f}_{k,\ell}\}_{\ell=0}^{n}$ 生成的矩阵，即

$$\mathbf{H}_k = [\boldsymbol{x}_{k,1} - \boldsymbol{x}_{k,0}, \boldsymbol{x}_{k,2} - \boldsymbol{x}_{k,0}, \cdots, \boldsymbol{x}_{k,n} - \boldsymbol{x}_{k,0}], \tag{5.3.12a}$$

$$\boldsymbol{\Gamma}_k = [\boldsymbol{f}_{k,1} - \boldsymbol{f}_{k,0}, \boldsymbol{f}_{k,2} - \boldsymbol{f}_{k,0}, \cdots, \boldsymbol{f}_{k,n} - \boldsymbol{f}_{k,0}]. \tag{5.3.12b}$$

为保证算法具有可操作性，即矩阵 $\mathbf{H}_k$ 和 $\boldsymbol{\Gamma}_k$ 均可逆，$n+1$ 个辅助点要处于一般位置（即不能共处一个平面）。否则，需要调整某些历史数据，重启算法。

以下恒假设 $\boldsymbol{x}_{k,0} = \boldsymbol{x}_k$，其它辅助点可由历史信息提供。依据不同的选取策略，割线法主要有两种实现过程。

1. 两点序列割线法：辅助点借用当前位置 $\boldsymbol{x}_k$ 和历史信息 $\boldsymbol{x}_{k-1}$ 生成，常见的两种构造方式是

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_k + \left[\boldsymbol{x}_{k-1}^{(\ell)} - \boldsymbol{x}_k^{(\ell)}\right]\boldsymbol{e}_\ell, \quad \ell = 1:n, \tag{5.3.13a}$$

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_k + \sum_{j=1}^{\ell}\left[\boldsymbol{x}_{k-1}^{(j)} - \boldsymbol{x}_k^{(j)}\right]\boldsymbol{e}_j, \quad \ell = 1:n, \tag{5.3.13b}$$

其中 $\boldsymbol{x}_k^{(\ell)}$ 是 $\boldsymbol{x}_k$ 的第 ℓ 个分量，$\boldsymbol{e}_\ell$ 是仅仅第 ℓ 个分量非零的单位向量。若采用 (5.3.13a)，每步迭代需要计算 n^2+n 个函数值；若采用 (5.3.13b)，每步迭代只需计算 n^2 个函数值。

2. $(n+1)$ 点序列割线法：辅助点直接定义为 n 个历史信息，即

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_{k-\ell}, \quad \ell = 1:n. \tag{5.3.14}$$

每步迭代只需计算 $\boldsymbol{f}(\boldsymbol{x}_k)$ 的 n 个函数值，其它信息都是已知的。引进置换阵，(5.3.12) 中的 $\mathbf{H}_k$ 和 $\boldsymbol{\Gamma}_k$ 可以改写为

$$\mathbf{H}_k = [\boldsymbol{x}_k - \boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-1} - \boldsymbol{x}_{k-2}, \cdots, \boldsymbol{x}_{k-n+1} - \boldsymbol{x}_{k-n}], \tag{5.3.15a}$$

$$\boldsymbol{\Gamma}_k = [\boldsymbol{f}_k - \boldsymbol{f}_{k-1}, \boldsymbol{f}_{k-1} - \boldsymbol{f}_{k-2}, \cdots, \boldsymbol{f}_{k-n+1} - \boldsymbol{f}_{k-n}]. \tag{5.3.15b}$$

类似于标量方程，可证：在适当（参见定理 5.12 ）条件下，p 点序列割线法局部收敛，收敛阶恰好是 $\lambda^p = \lambda^{p-1}+1$ 的最大正根。随着 p 增加，收敛阶逐渐单减到 1。

✯ **说明 5.17.** 算法效率从高到低排序：$(n+1)$ 点序列割线法、两点序列割线法、离散 *Newton* 方法。

✯ **说明 5.18.** $(n+1)$ 点序列割线法容易产生数值不稳定现象，特别当迭代序列落在某个真解附近时，$\mathbf{H}_k$ 和 $\boldsymbol{\Gamma}_k$ 均可能高度病态，造成数值计算出现极大偏差。改进的 n 点割线法[7]是较为有效的解决方案，即迭代公式中的 $\mathbb{A}_k$ 按如下方式构造：

- 若列号 j 和步数 k 关于 n 同余，相应位置的列向量按照离散 *Newton* 方法的计算公式进行更新；

[7] E. Polak, *A globally converging secant method with application to boundary value prolem*, SIAM J. Numer. Anal., 11(1974), 529-537

- 其余列向量保持不变，与 $\mathbb{A}_{k-1}$ 的列向量相同。

显然，$\mathbb{A}_k$ 是 $\mathbb{A}_{k-1}$ 的秩一修正。

5.3.5 拟 Newton 法

拟 Newton 法产生于上世纪 60 年代，具有较高的单步计算效率。其主要特点是针对系数矩阵的构造和求逆过程，提出了一个可以快速执行的递推方式。作为割线法的一种推广，拟 Newton 法的基本结构是

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \mathbb{B}_k^{-1}\boldsymbol{f}_k, \tag{5.3.16}$$

其中 $\mathbb{B}_k$ 是“导数矩阵”，$\boldsymbol{f}_k = \boldsymbol{f}(\boldsymbol{x}_k)$。其核心问题是：

> 在确保算法超线性收敛的前提下，利用当前信息 $\boldsymbol{x}_k, \boldsymbol{f}_k$ 和 $\mathbb{B}_k$，以及刚得到的计算信息 $\boldsymbol{x}_{k+1}$ 和 $\boldsymbol{f}_{k+1}$，快速构造出新的导数矩阵 $\mathbb{B}_{k+1}$ 或其逆矩阵。

回忆 $(n+1)$ 点序列割线法，导数矩阵 $\mathbb{A}_k$ 处处满足差商结构，即

$$\mathbb{A}_{k+1}\Delta\boldsymbol{x}_j = \Delta\boldsymbol{f}_j, \quad j = k : k-n+1, \tag{5.3.17}$$

其中 $\Delta\boldsymbol{f}_j = \boldsymbol{f}_{j+1} - \boldsymbol{f}_j$ 和 $\Delta\boldsymbol{x}_j = \boldsymbol{x}_{j+1} - \boldsymbol{x}_j$。由 (5.3.17) 可知

$$(\mathbb{A}_{k+1} - \mathbb{A}_k)\underbrace{[\Delta\boldsymbol{x}_{k-1}, \ldots, \Delta\boldsymbol{x}_{k-n+1}]}_{\mathbf{H}_k(:,1:n-1)} = [0, 0, \ldots, 0], \tag{5.3.18}$$

其中 $\mathbf{H}_k(:, 1:n-1)$ 是列满秩矩阵。换言之，$\mathbb{A}_{k+1}$ 是 $\mathbb{A}_k$ 的秩一修正。拟 Newton 法部分保留了这两个性质：

- $\mathbb{B}_{k+1}$ 满足最近两个位置的差商结构，即拟 Newton 方程

$$\mathbb{B}_{k+1}\Delta\boldsymbol{x}_k = \Delta\boldsymbol{f}_k. \tag{5.3.19}$$

- $\mathbb{B}_{k+1}$ 是 $\mathbb{B}_k$ 的低秩修正。此时，$\mathbb{B}_{k+1}^{-1}$ 可以在 $\mathbb{B}_k^{-1}$ 的基础上快速修改而成。

拟 Newton 方程 (5.3.19) 仅提供了 n 个约束条件，但有 n^2 个未知量；当 $n>1$ 时，它有无穷多解。

✎ **论题 5.13.** 不妨要求 $\mathbb{B}_{k+1}$ 是 $\mathbb{B}_k$ 的秩一修正，即

$$\mathbb{B}_{k+1}=\mathbb{B}_k+\boldsymbol{u}_k\boldsymbol{v}_k^\top, \tag{5.3.20}$$

其中 $\boldsymbol{v}_k$ 是给定向量，$\boldsymbol{u}_k$ 由拟 *Newton* 方法确定。相应的拟 *Newton* 方法就是著名的 *Broyden (1965)* 方法。

由 (5.3.20) 可知，$\mathbb{B}_{k+1}$ 和 $\mathbb{B}_k$ 作用在 $\mathrm{span}^\perp(\boldsymbol{v}_k)$ 的像是一样的。若 $\mathbb{B}_{k+1}$ 满足拟 Newton 方程，则应取

$$\boldsymbol{u}_k=(\Delta\boldsymbol{f}_k-\mathbb{B}_k\Delta\boldsymbol{x}_k)/\boldsymbol{v}_k^\top\Delta\boldsymbol{x}_k. \tag{5.3.21}$$

此时，$\mathbb{B}_{k+1}$ 是一个极小问题的解，即

$$\mathbb{B}_{k+1}=\arg\min_{\mathbb{S}\Delta\boldsymbol{x}_k=\Delta\boldsymbol{f}_k}\|\mathbb{S}-\mathbb{B}_k\|_F.$$

✎ **论题 5.14.** 记 $\mathbb{H}_k=\mathbb{B}_k^{-1}$。利用 *Sherman-Morrison* 公式，可得逐步修正公式

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

其中 $\boldsymbol{d}_k=\mathbb{H}_k\boldsymbol{v}_k$。关于 $\boldsymbol{v}_k$ 的选取，主要有两种方式，即

$$\boldsymbol{v}_k=\Delta\boldsymbol{x}_k,\quad \text{或}\quad \boldsymbol{v}_k=\boldsymbol{f}_{k+1}.$$

后者更适宜对称问题的求解，它可以一直保持 $\mathbb{H}_k$ 的对称性。将上述公式融合到拟 *Newton* 迭代中，每步迭代只需 $\mathcal{O}(n^2)$ 次乘除法运算。

✯ **说明 5.19.** 拟 *Newton* 方法的收敛阶不如 *Newton* 方法高。在适当条件下，可证 *Broyden* 方法具有局部的超线性收敛。

✯ **说明 5.20.** 假设方法 (5.3.16) 是收敛的。为保证其超线性收敛，导数矩阵 $\mathbb{B}_k$ 不用必须趋向 $\boldsymbol{f}'(\boldsymbol{x}_\star)$，只需满足一个较弱的充要条件[8]

$$\lim_{k\to\infty} \frac{\|\big[\mathbb{B}_k - \boldsymbol{f}'(\boldsymbol{x}_\star)\big]\Delta\boldsymbol{x}_k\|}{\|\Delta\boldsymbol{x}_k\|} = 0,$$

其中 $\boldsymbol{x}_\star$ 是问题的真解。在适当的条件下，上述条件等价于

$$\begin{aligned}\boldsymbol{s}_k^{\mathrm{Qn}} - \boldsymbol{s}_k^{\mathrm{Nt}} &= \Delta\boldsymbol{x}_k + [\boldsymbol{f}'(\boldsymbol{x}_k)]^{-1}\boldsymbol{f}(\boldsymbol{x}_k)\\ &= [\boldsymbol{f}'(\boldsymbol{x}_k)]^{-1}\{\boldsymbol{f}'(\boldsymbol{x}_k) - \mathbb{B}_k\}\Delta\boldsymbol{x}_k \to 0,\end{aligned}$$

其中 $\boldsymbol{s}_k^{\mathrm{Qn}}$ 是拟 *Newton* 方法中的修正方向，$\boldsymbol{s}_k^{\mathrm{Nt}}$ 是原始 *Newton* 迭代方法的修正方向。

❀ **思考 5.1.** 利用 *Broyden* 方法求解 $\boldsymbol{f}(\boldsymbol{x}) = (x_1, x_2^2 + x_2)^\top$，其真解是 $\boldsymbol{x}_\star = (0,0)^\top$。取初始向量和初始矩阵

$$\boldsymbol{x}_0 = (0,\varepsilon)^\top, \quad \mathbb{B}_0 = \begin{bmatrix} 1+\delta & 0 \\ 0 & 1 \end{bmatrix},$$

其中 ε 和 δ 均非零。计算迭代矩阵 $\mathbb{B}_k$ 的左上角元素，请问它是否收敛到 $\boldsymbol{f}'(\boldsymbol{x}_\star)$ 的左上角元素？

✯ **说明 5.21.** 虽然 *Broyden* 方法的单步计算效率获得极大改善，但是它理论上仅仅超线性收敛，算法效率不一定强过 *Newton* 方法。

$\mathbb{B}_{k+1}$ 也可是 $\mathbb{B}_k$ 的秩二修正，相应的拟 Newton 方法有 DFP 方法和 BFS 方法；相关内容可参考教科书，此处不再赘述。

[8] J.E. Dennis, and J. J. Moré, *A characterization of superlinear convergence and its application to quasi-Newton methods*, Math. Comp., 28 (1974), 549–560

5.3.6 其它算法简介

方程组 $\boldsymbol{f}(\boldsymbol{x})=\mathbf{0}$ 等价于优化问题

$$\boldsymbol{x}_\star=\arg\min_{\forall \boldsymbol{x}}\|\boldsymbol{f}(\boldsymbol{x})\|_2^2,$$

可用利用各种优化算法（例如最速下降法）求解。此处不做展开。

延拓法也称为同伦算法，其理论依据是同伦方程

$$\mathbf{h}(\boldsymbol{x})=t\boldsymbol{f}(\boldsymbol{x})+(1-t)\boldsymbol{g}(\boldsymbol{x}),\quad t\in[0,1]$$

的根 $\boldsymbol{x}(t)$ 连续依赖参数 t。假设 $\boldsymbol{g}(\boldsymbol{x})=\mathbf{0}$ 的根 $\boldsymbol{x}(0)$ 易求。利用各种高效的数值方法，求解常微分方程组

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t)=-[\mathbb{J}(\boldsymbol{x}(t))]^{-1}\boldsymbol{f}(\boldsymbol{x}(0)),\quad t\in[0,1],$$

即可得到 $\boldsymbol{f}(\boldsymbol{x})=\mathbf{0}$ 的根 $\boldsymbol{x}(1)$，其中

$$\mathbb{J}(\boldsymbol{x}(t))=\begin{bmatrix}\frac{\partial f_1(\boldsymbol{x}(t))}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x}(t))}{\partial x_2} & \cdots & \frac{\partial f_2(\boldsymbol{x}(t))}{\partial x_n}\\ \frac{\partial f_2(\boldsymbol{x}(t))}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x}(t))}{\partial x_2} & \cdots & \frac{\partial f_n(\boldsymbol{x}(t))}{\partial x_n}\\ \vdots & \vdots & \cdots & \vdots\\ \frac{\partial f_n(\boldsymbol{x}(t))}{\partial x_1} & \frac{\partial f_n(\boldsymbol{x}(t))}{\partial x_2} & \cdots & \frac{\partial f_n(\boldsymbol{x}(t))}{\partial x_n}\end{bmatrix}.$$

第 6 章
附录：数值实验

数值实验部分的两个线性方程组（或系数矩阵）均源于 Possion 方程的有限差分格式。相应的离散过程如下：

- 先考虑两点边值问题

$$-u''(x) = f(x), \quad x \in (0,1), \quad u(0) = u(1) = 0.$$

在网格点 $\{x_i = ih\}_{i=1:n}$ 处，利用二阶中心差商代替二阶导数，可得差分方程

$$-u_{i-1} + 2u_i - u_{i+1} = h^2 f(ih), \quad i = 1:n,$$

其中 $h = 1/(n+1)$ 为网格步长，u_i 是 $u(ih)$ 的近似。注意到零边值条件，差分方程可以汇总为线性方程组

$$\mathbb{T}_n \boldsymbol{x} = \boldsymbol{b}_n, \tag{6.0.1}$$

其中 $\boldsymbol{x} = \{u_j\}_{j=1:n}$ 和 $\boldsymbol{b}_n = \{h^2 f(jh)\}_{j=1:n}$ 是向量，

$$\mathbb{T}_n = \text{tridiag}(-1, 2, -1) = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \tag{6.0.2}$$

是三**对角对称正定阵**。

- 再考虑正方形区域的 Poisson 方程

$$-u_{xx}(x,y)-u_{yy}(x,y)=f(x,y),\quad (x,y)\in(0,1)^2,$$

相应的边界条件是 $u(x,y)=0$。在内部网格点

$$(x_i,y_j)=(ih,jh),\quad i=1:n,\quad j=1:n,$$

利用二阶中心差商离散两个二阶偏导数，可得差分方程

$$4u_{ij}-u_{i-1,j}-u_{i+1,j}-u_{i,j-1}-u_{i,j+1}=h^2f(ih,jh),$$

其中 $h=1/(n+1)$ 为网格步长，u_{ij} 是 $u(ih,jh)$ 的近似。注意到边值条件，差分方程可以汇总为线性方程组

$$\mathbb{A}_{n^2}\boldsymbol{x}=\boldsymbol{c}_{n^2}, \tag{6.0.3}$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{c}_{n^2}$ 分别是 $\{u_{ij}\}_{i=1:n}^{j=1:n}$ 和 $\{h^2f(ih,jh)\}_{i=1:n}^{j=1:n}$ 逐行（从下到上从左到右）排序而成的向量，

$$\begin{aligned}\mathbb{A}_{n^2}&=\begin{bmatrix}\mathbb{T}_n+2\mathbb{I}_n & -\mathbb{I}_n & & \\ -\mathbb{I}_n & \mathbb{T}_n+2\mathbb{I}_n & & \\ & \ddots & \ddots & -\mathbb{I}_n \\ & & -\mathbb{I}_n & \mathbb{T}_n+2\mathbb{I}_n\end{bmatrix}\\ &=\mathbb{T}_n\otimes\mathbb{I}_n+\mathbb{I}_n\otimes\mathbb{T}_n\end{aligned} \tag{6.0.4}$$

是**块三对角对称正定阵**。在上述公式中，$\mathbb{I}_n$ 是 n 阶单位阵，$\mathbb{T}_n$ 是 (6.0.2) 给出的三对角阵，$\otimes$ 是矩阵 Kronecker 乘积。

✯ **说明 6.1.** 实验报告要格式规范，按论文体成文，包含题目、摘要、前言（简介实验目的和意义）、数学原理和程序设计流程（简述数值方法实现过程，编程设计思想）、实验结果和数据讨论（这是主要内容!）、以及最后小结和参考文献等基本内容。

✯ **说明 6.2.** 提交作业必须包含试验报告的 *PDF* 文档，其他文档或材料可作为附件。实验报告要发送到邮箱：*qzh_nk@aliyun.com*，邮件标题格式是：第 *XXX* 次上机作业（姓名）。

✯ **说明 6.3.** 编程语言不限；鼓励采用国内自主开发的“北太天元”数值计算通用软件，可在 *https://www.baltamatica.com* 下载。

6.1 线性方程组的直接法

❖ **练习 6.1.1.** 利用列主元 *Gauss* 消元法、$LL^\top$ 法和 $LDL^\top$ 法求解线性方程组 (6.0.3)，真解取为 $\boldsymbol{x}_\star = (1,1,1,\ldots,1)^\top$，右端向量由利用真解计算出来。

1. 绘制数值误差同矩阵阶数 n 的关系，其中数值误差采用对数坐标；
2. 绘制 *CPU* 时间同 n 的关系；
3. 绘制矩阵条件数与 n 的关系。请问：摄动理论给出的 (1.4.26) 是否完美刻画了相对误差的大小？

❖ **练习 6.1.2.** 非零元素分布可以影响数值计算的效率。考虑行列重排后相等的两个 n 阶矩阵

$$\mathbb{B}_1 = \begin{bmatrix} 1 & & & & a \\ & 1 & & & a \\ & & \ddots & & \vdots \\ & & & 1 & a \\ a & a & \cdots & a & 1 \end{bmatrix}, \quad \mathbb{B}_2 = \begin{bmatrix} 1 & a & \cdots & a & a \\ a & 1 & & & \\ \vdots & & \ddots & & \\ a & & & 1 & \\ a & & & & 1 \end{bmatrix}, \tag{6.1.5}$$

执行相应的 *Crout* 算法；利用 *Matlab* 命令 *spy()* 绘制它们在三角分解后的非零元素分布（或结构图），并比较相应的 *CPU* 时间。

利用矩阵的元素分布特点[1]，修改 *Crout* 算法，删除那些无用的运算时间。重复上述操作，观察 *CPU* 时间是否得到节省？

❖ **练习 6.1.3.** 计算三对角阵 $\mathbb{T}_n$ 或块三对角阵 $\mathbb{A}_{n^2}$ 的逆矩阵，并观测它们的运行效率（关于 n 的计算复杂度）。

❖ **练习 6.1.4.** 设 $\mathbb{D}_n = diag\{2^{-i}\}_{i=1}^n$，定义 $\widetilde{\mathbb{T}}_n = \mathbb{D}_n\mathbb{T}_n$；考虑两个同解的三对角线性方程组

$$\widetilde{\mathbb{T}}_n\boldsymbol{x} = \tilde{\boldsymbol{b}}_n, \quad \mathbb{T}_n\boldsymbol{x} = \boldsymbol{b}_n,$$

其右端项均由真解 $\boldsymbol{x}_\star = (1,1,1,\ldots,1)^\top$ 生成。用追赶法求解它们，观测数值误差同 n 的关系。

❖ **练习 6.1.5.** 取定一组关于阶数 n 的序列（建议尽可能大些），然后就每个 n 随机产生 $500 \sim 1000$ 个 n 阶可逆阵 $\mathbb{A}$，执行列主元高斯消元过程。观测并统计主元增长因子 $\eta(\mathbb{A})$ 同 n 的量级关系？

6.2 线性方程组的迭代法

考虑线性方程组 (6.0.3)，其右端向量由真解 $\boldsymbol{x}_\star = (1,1,\ldots,1)^\top$ 给出。取初始向量 $\boldsymbol{x}_0 = 0$，用户指标 $\mathcal{E} = 10^{-6}$。

❖ **练习 6.2.1.** 采用不同的停机标准和向量范数，观测和比较 J 方法和 *GS* 方法达到用户要求的迭代次数，以及停机时的真实误差。尝试给出上述信息关于矩阵阶数 n 的关系。

❖ **练习 6.2.2.** 以真实误差的欧式范数作为停机标准，运行 *SOR* 方法，观察松弛因子 ω 对于迭代次数的影响。随机取定一组关于阶数 n 的

[1]事实上，非零元素的最优分布很难找到。这是一个 NP（非多项式）问题。

序列（建议尽可能大些），数值扫描出最佳松弛因子，并论证数值结果与理论结果是否吻合？

❖ **练习 6.2.3.** 采用欧式范数和半对数坐标系。考虑 *J* 方法、*GS* 方法和（带最佳松弛因子的）*SOR* 方法，进行下面的数值观察：

1. 绘制误差曲线和残量曲线，比较三种算法的优劣；

2. 以真实误差为停机标准，绘图并指出迭代次数同矩阵阶数的关系。

❖ **练习 6.2.4.** 采用欧式范数和半对数坐标系。绘制变系数 R 方法的误差曲线以及残量曲线，观测循环指标 m 的数值影响。

❖ **练习 6.2.5.** 运行 *J* 方法的半迭代加速，相应的循环指标设置为 $m = +\infty, 100, 50, 25$。绘制相应的误差曲线和残量曲线。它与变系数 R 方法有何区别？

❖ **练习 6.2.6.** 取 $n = 50$ 和 $n = 51$，对应不同的奇偶状态。取欧式范数，执行 *CG* 方法。

1. 绘制相应的误差曲线和残量曲线；

2. 观测迭代次数同矩阵阶数的关系；它同带有最佳松弛因子的 *SOR* 方法有何差异？

❖ **练习 6.2.7.** 以 *SSOR* 方法的主体部分做为预处理阵，运行相应的预处理 *CG* 算法。随机生成不同阶数的系数矩阵，考察迭代次数同参数 ω 的关系。

6.3 线性最小二乘问题的数值方法

❖ **练习 6.3.1.** 随机构造 *100* 个 $5000 \sim 10000$ 阶的可逆方阵，分别利用 *CGS* 方法、*MGS* 方法、*Householder* 方法和 *Givens* 方法给出相

应的 QR 分解。统计和比较它们在列正交性、CPU 时间以及向后稳定性表现方面的差异。

❖ **练习 6.3.2.** 列满秩最小二乘问题 $\mathbb{A}_{n\times(n-1)}\boldsymbol{x}_{n-1}=\boldsymbol{b}_n$ 具有最小二乘解 $(1,1,1,\ldots,1,1)^\top$，其中

$$\mathbb{A}_{n\times(n-1)}=\mathbb{T}_n(1:n,1:n-1)=\begin{bmatrix}2&-1&&&\\-1&2&-1&&\\&\ddots&\ddots&\ddots&\\&&-1&2&-1\\&&&-1&2\\&&&&-1\end{bmatrix}$$

和

$$\boldsymbol{b}_n=\mathbb{U}_{n\times n}\left(1+\frac{1}{n},\frac{2}{n},\frac{3}{n},\ldots,\frac{n-2}{n},1+\frac{n-1}{n},0\right)^\top.$$

以 500 为间隔，将 n 从 500 增加到 2500；利用不同方法（法方程组、扩展法方程组、MGS、$Householder$ 和 $Givens$）求解最小二乘问题

$$\mathbb{B}_{n\times(n-1)}\boldsymbol{x}_{n-1}=\boldsymbol{c}_n,$$

并绘制计算机时和数值精度关于 n 的关系。这里，要求

$$\mathbb{B}_{n\times(n-1)}=\mathbb{U}_{n\times n}\mathbb{A}_{n\times(n-1)}$$

是一个稠密矩阵，其中 $\mathbb{U}_{n\times n}$ 是有限 $(1000\sim2000)$ 个随机生成的 $Given$ 平面旋转阵乘积。此时，$\boldsymbol{c}_n=\mathbb{U}_{n\times n}\boldsymbol{b}_n$。

❖ **练习 6.3.3.** 参考讲义中的图 *3.2.1*和图 *3.4.2*，（可选取不同的例子）进行数值计算和绘制其效果。

6.4 矩阵特征值的数值方法

默认的求解对象是对称三对角阵 $\mathbb{T}_n$，其中 $n = 100$ 或 $n = 101$。用户指标设定为 $\mathcal{E} = 10^{-8}$。

❖ **练习 6.4.1.** 取初始向量 $\boldsymbol{v}_0 = (1, 1, 1, \ldots, 1)^\top$，用幂法计算主特征值及其特征向量。

1. 绘制特征值的误差曲线，以及特征子空间的距离曲线；

2. 采用 *Atiken* 技巧和 *Rayleigh* 商技术进行加速，绘制相应的特征值误差曲线。

若改变初始向量，结果有什么区别吗？

❖ **练习 6.4.2.** 利用反幂法，分别求解离 $q = 2$ 和 $q = 3$ 最近的特征值及其特征向量。绘制相应的特征值和特征向量误差曲线。

❖ **练习 6.4.3.** 取一个非常接近某个特征值的 q，观察反幂法是否呈现出“一次迭代”特性？

❖ **练习 6.4.4.** 首先，利用幂法和降维技巧，求解前两个主特征值及其特征向量；然后，利用同时迭代方法求解前两个主特征值。比较两者的计算效果有何差异。

❖ **练习 6.4.5.** 分别用古典 *Jacobi* 方法、循环 *Jacobi* 方法和阈值 *Jacobi* 方法求解全部特征值，并绘制 $\|E_k\|_F$ 和对角元的收敛过程。

❖ **练习 6.4.6.** 首先，利用 *Strum* 序列二分法，求解位于开区间 $(1, 2)$ 内的所有特征值；绘制相应的收敛过程。然后，考虑带原点位移的反幂法，观测数值精度是否得到改善？

❖ **练习 6.4.7.** 利用隐式对称 *QR* 方法求解全部特征值。

❖ **练习 6.4.8.** 阈值 *Jacobi* 方法求解（绝对值）小特征值时具有优势。考虑对称正定矩阵

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

直接计算可知其特征值为 $10^{40}, 9.9 \times 10^{19}$ 和 9.81818×10^{-1}。用阈值 *Jacobi* 方法求解三个特征值，并同 *Matlab* 命令 *eig()* 给出的结果进行比较。

6.5 非线性方程的数值方法

用真实误差作为停机标准，用户指标设定为 $\mathcal{E} = 10^{-6}$。

❖ **练习 6.5.1.** 用 *Newton* 方法，计算多项式 $x^3 - x^2 - 8x + 12 = 0$ 的最大实根。绘制误差曲线，计算其数值收敛阶。

❖ **练习 6.5.2.** 已知 $x = 0$ 是 $\sin x = x - x^3/6$ 的重根。采用 *Newton* 方法求解，观察其误差曲线和数值收敛阶。修改算法，改善数值收敛阶，并给出相应的实验数据。

❖ **练习 6.5.3.** 用割线法，计算前两题的问题；比较其与切线法的区别（例如收敛阶和 *CPU* 时间），并给出相应的数值观察。

❖ **练习 6.5.4.** 考虑非线性方程组（不要进行任何手工化简）

$$\begin{cases} (x+3)(y^2-7) + 18 = 0, \\ \sin(ye^x - 1) = 0. \end{cases} \tag{6.5.6}$$

取初始位置 $(-0.15, 1.4)$，比较 *Newton* 方法和 *Broyden* 方法（第一步同前）的误差曲线和迭代次数；尝试其它初始位置，看看其具体情况是

什么？

❖ **练习 6.5.5.** 依旧考虑前面的非线性方程组 (6.5.6)，分别用修正 *Newton* 法（不同的 m）、离散 *Newton* 法、两点序列割线法和三点序列割线法四种方法求根，绘制并比较相应的误差曲线。

❖ **练习 6.5.6.** 考虑非线性方程组

$$\begin{cases} x+y-3=0, \\ x^2+y^2-9=0. \end{cases} \tag{6.5.7}$$

取初始位置 $(2,4)$，$\mathbb{B}_0$ 为该点的 *Jacobi* 矩阵。观察 *Broyden* 方法的迭代矩阵 $\mathbb{B}_k$ 是否收敛到相应的 *Jacobi* 矩阵？

❖ **练习 6.5.7.** 设 $\mathbb{T}_n$ 是 (6.0.2) 给出的三对角对称矩阵，阶数分别是 $n=5$ 和 $n=8$。相应的特征值问题可以陈述为非线性方程组

$$\begin{cases} \mathbb{T}_n\boldsymbol{x}-\lambda\boldsymbol{x}=0, \\ \boldsymbol{x}^\top\boldsymbol{x}=1. \end{cases} \tag{6.5.8}$$

任取一个单位长度的向量 $\boldsymbol{x}_0$，令 $\lambda_0=\boldsymbol{x}_0^\top\mathbb{T}_n\boldsymbol{x}_0$，执行 *Newton* 方法，观察其数值结果同幂法有何区别，并给出相应的解释。

参考文献

[1] 曹志浩,, 新华书店上海发行所, 1996

[2] 蔡大用, 数值代数, 清华大学出版社, 2005

[3] 李大明, 数值线性代数, 清华大学出版社, 2010

[4] 李庆扬，王能超，易大义，数值分析，清华大学出版社，2010

[5] 李庆扬，关治，白峰杉，数值计算原理，清华大学出版社，2009

[6] 林成森, 数值计算方法（第二版）, 科学出版社, 2005

[7] 徐树方, 矩阵计算的理论与方法，北京大学出版社，1995

[8] 徐树方, 高立, 张平文, 数值线性代数, 北京大学出版社, 2010

[9] 威尔金森，代数特征值问题, 石钟慈，邓建新译，科学出版社，2001

[10] 张凯院，徐仲，数值代数，（第二版）修订本，科学出版社，2011

[11] G. H. Golub, C. F. Van Loan, *Matrix Computations*