第六版（2018）

数值计算与试验 II

编著：张 强

南京大学数学系

前　言

本讲义是教科书《数值计算方法（下册）》的配套材料，用于介绍线性方程组、线性最小二乘问题、矩阵特征值问题以及非线性方程（组）的数值解法。主要参考书目如下:

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

附录部分还包含常微分方程数值解和函数逼近。具体内容正在逐步扩充和完善中。

目 录

第 1 章
线性方程组的直接解法

直接解法是一种准确算法。换言之，在没有舍入误差的情况下，经过有限次的四则运算（可能包括少量的开方运算），即可给出线性方程组的准确解。本章重点介绍线性方程组的高斯消元方法及其各种变形。不同的实现方法在理论上是等价的，但是在计算机上的具体数值表现却并不完全相同。在学习过程中，请重点关注各种数值方法的具体操作流程，例如 (a) 数据操作和编程技巧; (b) 数据存储量和计算复杂度；以及 (c) 舍入误差的有效控制。

1.1 高斯消元方法

对于中小规模（未知量总数约 $1000 \sim 10000$）的线性代数方程组

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}, \tag{1.1.1}$$

高斯消元方法是最为常用的数值方法。特别地，当系数矩阵 $\mathbb{A}$ 随机稠密（几乎处处非零）的时候，该方法在某种意义下堪称是最佳的选择。

消元思想最早出现在中国秦汉时代的《九章算术》; 直至 19 世纪初，西方国家才基于这种思想，提出了高斯消元方法[i]。在 20 世纪中叶以后，科技飞速发展，使得线性方程组的求解规模（或矩阵阶数）越来越大。若以数字计算机作为高效的计算工具，相应的高斯消去方法也将遇到一些不容忽视的问题，需要细致深入的理论研究。

[i] 1809 年，发表于 Theoria Motus

1.1.1 基本过程

通常，《线性代数》课程就会讲到高斯消元方法。其基本思想是：通过消元过程不断地缩减待解未知量的个数，将复杂的线性方程组 (1.1.1) 转化为同解的简单（三角形）线性方程组。若用矩阵语言来描述上述过程，它等价于增广矩阵 $[\mathbb{A}|\boldsymbol{b}]$ 的矩阵变换，即利用一系列的初等行变换（或者初等变换矩阵的左乘）过程，将增广矩阵转化为一个上梯形矩阵。

在 Matlab 中，高斯消元方法的对应命令是 A\b，其中 A $=\mathbb{A}$ 是 (1.1.1) 的系数矩阵，而 b $=\boldsymbol{b}$ 是右端向量。

论题 1.1. *顺序高斯消元算法是最简单的处理过程。*

对于任何一个数值方法，我们都要关注它在计算机上的自动实现过程，特别是数据的存储和操作问题。基于实用的角度，数据存储量要尽可能的精简，计算结果要尽可能的可靠。

```
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
```

在左侧的图文框中，我们给出了顺序高斯消元方法的伪代码片断，其中 a_{ij} 是系数矩阵（对应二维数组）元素，n 是矩阵阶数或数组维数。请注意，这里的赋值符号“:=”蕴含着简单而重要的数据覆盖技术。换言之，在存储系数矩阵 $\mathbb{A}$ 的二维数组中，位于对角线下方位置的数据将依次被相应的消元乘子所覆盖，而位于对角线及其上方位置的数据将被最终给出的上三角矩阵元素所覆盖。**数据存储单元的元素变化是数值方法实现的一个重点。**

在实际的数值计算中，右端项 $\boldsymbol{b}$ 常常被放在矩阵 $\mathbb{A}$ 的右侧，形成相应的增广矩阵。它依旧可以用一个二维数组来存储。此时，我们只需

修改图文框中的第 5 步，将关于 j 的取值范围扩充到 $n+1$ 即可。

一个算法的计算复杂度，通常可以利用它所包含的乘除法运算次数来衡量。在系数矩阵的高斯消元过程中，乘除法运算的总次数为

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

相应右端项的消元操作，仅仅需要 $\mathcal{O}(n^2)$ 次的乘除法运算。在消元之后，我们只需求解上梯形部分对应的三角形方程组，就能得到最终的答案。相应的求解过程称为回代过程，它只需要共 $\mathcal{O}(n^2)$ 次的乘除法运算。

顺序高斯消元方法含有除法运算。在消元过程中，如果后续的对角元素为零，则顺序高斯消元方法将会意外地终止。因此，我们需要考虑该算法是否可以顺利地执行到底?

暂时假设四则运算都是准确的，有如下的理论结果。

定理 1.1. *若顺序高斯消元过程中的对角线元素*

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

则消元过程可执行到底。(1.1.2) *成立的充分必要条件是矩阵* $\mathbb{A}$ *的前* $n-1$ *阶顺序主子阵都是非奇异的。若所有对角线元素均不为零，则回代过程也可执行。*

在顺序高斯消元过程中，我们需要不断检验 (1.1.2) 是否成立。事实上，在除法运算前，我们都需要事先判断是否“除零”。但是，若系数矩阵具有某些特殊的结构，我们可以事先得到明确结论，不必担心此事发生而节省相关的判断时间。例如

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

证明：在矩阵 Cholesky 分解部分，我们会给出相应的答案。 □

代码编写是数值方法的重要研究内容之一，因为代码好坏对于数值算法的实现效率具有显著的影响。计算机代码是否高效，强烈地依赖于它所用的计算机语言以及计算机硬件结构。譬如，一个关键的影响因素是数据的读写效率。

✯ **说明 1.1.** 在图文框中的伪代码中，三重循环次序是 k–j–i。这种方法特别适用于 *Fortran* 语言，因为此时的二维数组是按列连续存放的，数据寻址所付出的代价较低。

然而，*C++* 语言中的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，数据的读取是跳跃的，使得数据指针的移动过于频繁，从而消耗过多的计算时间，影响算法的性能。此时，我们有必要交换代码最内两层的循环次序。

✯ **说明 1.2.** 代码编写和执行效率还同计算机的硬件结构有关，特别是数据的读写加速设备（例如二级缓存等）的使用。

在图文框中的伪代码中，所有的数据操作都是数和数之间的单一运算。每个运算都要进行一次数据的读写，而数据的读写要比四则运算消耗更多的 *CPU* 时间。这样的代码仅仅处于 *BLAS-1* 的代码级别[ii]，处理大规模数据的能力和效率都不高。

实际上，数值计算还可以采用更高的 *BLAS* 代码级别来实现。在 *BLAS-2* 代码级别中，数据操作基于向量与向量乘积（含矩阵与向量乘积）的块数据操作。借用 *Matlab* 中的向量语言，顺序高斯消元方法的 *BLAS-2* 版本为：

[ii] BLAS=Basic Linear Algebraic Subroutine.

1. *For* $k = 1, 2, \ldots, n$, *Do*
2. $\quad \mathbb{A}(k+1:n, k) := \mathbb{A}(k+1:n, k)/\mathbb{A}(k, k)$;
3. $\quad \mathbb{A}(k+1:n, k+1:n) :=$
$$\mathbb{A}(k+1:n, k+1:n) - \mathbb{A}(k+1:n, k) \star \mathbb{A}(k, k+1:n);$$
4. *Enddo*

这种代码可以充分减少数据的读写时间。最高的代码级别是 *BLAS-3*, 它主要基于矩阵与矩阵乘积的块数据操作。这需要涉及很多关于并行计算的实现过程。因其超出课程设置，详略。

事实上，即使顺序高斯消元过程可以在计算机上顺利地执行，最终得到的数值结果也并没有像理论所说的那样完美。在上面的所有讨论中，我们均假设计算是精确的，完全忽略了近似计算可能造成的影响。我们要特别指出：在大量的四则运算过程中，近似计算的舍入误差是不可轻视的。有时候，数值结果会因此而产生严重偏离。

构造在计算机上可行的数值算法，建立相应的数值稳定性分析，划定算法的适用范围，指出算法的缺陷和不足，完善算法的实现效果等等问题，均是计算数学的重要研究内容。在学习、使用和构造各种计算方法的时候，我们要随时注意上述问题。

1.1.2 主元技巧

由于无法回避的舍入误差问题，顺序高斯消元过程给出的计算结果可能是不准确的，甚至是毫无价值的。

首先要指出，计算结果的偏差程度同计算机的位长（或者机器精度）有关。为说明这个现象，让我们在仅仅具有三位有效数字的十进制虚拟计算机上，执行顺序高斯消元方法。针对待解的线性方程组，数据流的

具体变化是

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

进而，相应的回代过程给出数值解 $x_{\text{num}} = (0.00, 1.00)^\top$，它与精确解

$$x_\star = (1.002\cdots, 0.998\cdots)^\top$$

相去甚远。若增加计算机的有效位长，顺序高斯消元方法给出的数值解将更为准确。

因此说，如果舍入误差的影响非常严重，即使理论上准确的算法也不能在计算机上直接应用，给出相对理想的数值结果。如何在现有计算环境下构造出可行的算法，精细地控制舍入误差的产生和积累，是数值方法研究的重中之重。

为控制高斯消元方法的舍入误差，最简单易行的方法是引进所谓的高斯主元策略，提出高斯主元消元方法。高斯主元是指位于当前对角线位置右下方，按绝对值最大的那个矩阵元素。常用的主元策略有列主元/全主元（Wilkinson, 1961）策略和車型主元（Neal 和 Poole, 1992）策略。列主元策略在搜索时间上占有优势，故而常常作为首选策略。

论题 1.2. 列主元高斯消元方法的代码实现是非常简单的。对于顺序高斯消元方法的任意两个版本，我们都只需在第 *2* 行代码前，填入如下的一小段补丁代码

> - 选择 l 使得 $|a_{lk}| = \max_{k\le i\le n} |a_{ik}|$，交换第 l 行和第 k 行数据；

这样的处理，可以使所有的消元乘子按绝对值均不超过 *1*，消元过程中的乘法运算所带来的舍入误差被有效地控制。

✯ **说明 1.3.** 事实上，行交换过程要花费很多毫无价值的数据读写时间。为提高代码的执行效率，我们可以引进一个 n 维指标向量 $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$，并重写代码，避免数据的移动。

实际上，指标向量 $\boldsymbol{p}$ 记录整个消元过程中的行交换信息。它的初始状态是自然序列，即 $p_i = i$。若第 k 步消元需要进行第 k 行和第 r 行的交换，我们只需轮换指标向量 $\boldsymbol{p}$ 中的第 k 分量和第 r 个分量，并调整对应循环中的行指标。

❀ **思考 1.1.** 重写代码，实现上述目标。

计算经验表明：同全主元策略相比，列主元策略在数值稳定性方面不相上下，但主元搜索时间却得到明显的减少。因此，在中小型稠密线性方程组的数值计算方法中，列主元高斯消元法已成为最受欢迎的算法之一。

✯ **说明 1.4.** 教科书还给出了另外一种列主元选取策略，即按比例选取列主元。在每次寻求高斯消元主元之前，我们首先将右下角矩阵的所有行向量按最大模分量进行单位化。这样的处理工作等价于一个所谓的预处理过程。

✯ **说明 1.5.** 行列式的计算是简单的，我们可以利用顺序高斯消元过程得到的对角线元素乘积得到。若采用列主元策略，请注意行交换的次数。详略。

1.1.3 高斯消元变换阵

利用矩阵（或行变换）语言，高斯消元过程的本质可以描述为系数矩阵的三角化过程，或增广矩阵的上梯形化过程。相应的核心问题是：

已知 m 维非零向量 $\boldsymbol{a}=(a_1,a_2,\ldots,a_m)$，其中 $a_1\neq 0$。我们能否构造一个简单矩阵 $\mathbb{H}$，左乘向量 $\boldsymbol{a}$ 后，可将 $\mathbb{H}\boldsymbol{a}$ 转化为仅首个位置非零的向量?

事实上，这个问题也是数值代数中的一个基本问题。

这个问题可以利用矩阵左乘来实现。常见的矩阵有高斯消元阵、Householder 镜像变换矩阵、和 Givens 平面旋转矩阵。它们都是所谓的简单矩阵，即它们具有某些特殊的结构，在计算复杂度方面都具有一定的优势。本章仅仅考虑高斯消元阵，在第三章再介绍其它两个。

定义 1.1. *记 $\hat{\boldsymbol{e}}_1=(1,0,0,\ldots,0)^\top$，是首个位置为 1 的 m 维单位向量。对于向量 $\boldsymbol{a}$，相应的 m 阶 Gauss 消元阵为单位矩阵的秩一修正，即*

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

其中 $\hat{\boldsymbol{p}}=(0,a_2/a_1,a_3/a_1,\ldots,a_m/a_1)^\top$ 是由所有高斯消元乘子构成的 m 维列向量。

通常，上述数值目标可以扩展到 n 维向量

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

其中 $a_{kk}\neq 0$。此时，我们可以利用非零的对角分量 a_{kk}，将其下方的所有元素清零。它可理解为高斯消元过程的第 k 步。

定义 1.2. *令 $\ell_{ik}=a_{ik}/a_{kk}$ 为相应的消元乘子，定义*

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

对应的高斯消元矩阵为

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

其中 $\boldsymbol{e}_k$ 是第 k 个分量为 1 的 n 维单位向量。事实上，这个矩阵可以理解为 $n-k+1$ 阶的高斯消元阵的单位矩阵扩张[iii]，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 1.3.** 注意到 $\boldsymbol{\ell}_k$ 的构成方式，顺序高斯消元方法的第 k 步可描述为高斯消元阵 (1.1.6) 的左乘。因此，整个顺序高斯消元方法可以描述为

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

其中 $\mathbb{A}^{(1)} = \mathbb{A}$ 和 $\boldsymbol{b}^{(1)} = \boldsymbol{b}$ 是相应的代数方程组信息。

可证高斯消元矩阵具有如下的基本性质：

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

由高斯顺序消元过程，我们可以衍生出一个重要的矩阵分解方式，即三角分解

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

其中 $\mathbb{U} = \mathbb{A}^{(n)}$ 是上三角矩阵，而

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

[iii] 这种默认的扩张方式将在本课程中多次使用，以后不再赘述。

是单位下三角矩阵，其中

$$\ell_{ij} = a_{ij}^{(j)}/a_{jj}^{(j)}$$

是高斯消元乘子。三角分解在矩阵理论中占有非常重要的地位，类似的工作及其应用将在下节做深入的介绍。

✎ **论题 1.4.** 列主元的选取可描述为初等排列阵的左乘，从而列主元高斯消元法可矩阵描述为：

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

其中 $\mathbb{U}$ 是消元后的上三角矩阵，$\mathbb{L}_k^{-1}$ 是对应第 k 步列主元交换后的高斯消元矩阵。利用数学归纳法，我们可证

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

其中 $\mathbb{P}$ 是由单位矩阵经过相应的行交换而形成的置换阵，而

$$\widetilde{\mathbb{L}}_k^{-1} = \mathbb{I}_{n-1,r_{n-1}}\cdots\mathbb{I}_{k+1,r_{k+1}}\mathbb{L}_k^{-1}\mathbb{I}_{k+1,r_{k+1}}\cdots\mathbb{I}_{n-1,r_{n-1}} \tag{1.1.7}$$

的非零元素分布和 $\mathbb{L}_k^{-1}$ 相同，仅仅是高斯消元乘子的所在位置略有不同。如前面的讨论，可知矩阵 $\widetilde{\mathbb{L}}$ 具有单位下三角结构。

1.1.4 逆矩阵的计算

利用高斯消元方法，求解同型线性代数方程组

$$\mathbb{A}\boldsymbol{x}_j = \boldsymbol{e}_j, \quad j = 1, 2, \ldots, n.$$

即可给出矩阵 $\mathbb{A}$ 的逆矩阵 $\mathbb{A}^{-1} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$。但是，相应的计算复杂度较高，整个过程需 $\mathcal{O}(4n^3/3)$ 次乘除法运算。为此，逆矩阵的计算通常采用 Gauss-Jordan（G-J）消元方法。

G-J 消元方法是一个古老的算法，它是由测量技师 Wilhelm Jordan (1842-1899) 和 B.I. Clasen (1887) 分别独立提出。

论题 1.5. *Gauss-Jordan 消元方法的基本思想：利用对角元素，直接消去同列的其他所有元素。设第 k 列元素是 (1.1.4)，其中 $a_{kk} \neq 0$。相应的 G-J 消元过程可描述为 G-J 消元矩阵*

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

的左乘，其中的 G-J 消元因子是

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

对应不同的数据存储方式，矩阵求逆的程序实现有两种方式。一种方法是存储增广矩阵 $[\,\mathbb{A}|\,\mathbb{I}\,]$，然后执行 G-J 消元过程。另一种方法是借助数据覆盖技术，仅仅存储矩阵 $\mathbb{A}$，节省一半的数据存储空间。相应的伪代码片段如下：

```
1.  For k = 1, 2, ..., n, Do
2.      交换 𝔸 的第 k 行和第 p_k 行，其中 p_k 为相应的列主元;
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,
7.            Do 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
```

显然，这个算法总共需 $\mathcal{O}(n^3)$ 次乘除法运算。在 Matlab 中，逆矩阵的相应命令是 inv()。

请注意伪代码三个关键操作的次序：首先计算同列的单位化因子和消元乘子（第 3-4 行），然后执行其余各列的 G-J 消元（第 5-9 行），最后进行相应行的单位化（第 10 行）。

算法中的第 12-14 行是恢复计算数据的真正存储位置。因为在执行第 k 步 G-J 消元的时候，实际操作对应 $\mathbb{M}_k\mathbb{I}_{k,i_k}$ 的左乘，其中 $\mathbb{M}_k$ 为相应的 G-J 消元阵。相应的 G-J 消元因子应出现在第 i_k 列，但是计算机未执行相应的列交换，将数据依旧存储在第 k 列。

☆ **说明 1.6.** *将 G-J 消元算法应用于单个方程组的求解，它的计算效率要比高斯消元方法要差，共需 $\mathcal{O}(n^3/2)$ 次乘除法运算。*

☆ **说明 1.7.** *事实上，逆矩阵还有其他的求解方法。例如，Newton 迭代方法 $\mathbb{X}_{k+1} = 2\mathbb{X}_k(\mathbb{I} - \mathbb{A}\mathbb{X}_k)$ 也可以求解矩阵 $\mathbb{A}$ 的逆矩阵。因课时*

限制，此处不再赘述。

1.2 直接三角解法

本节考虑高斯消元方法的其他实现方式。为简单起见，我们略去回代过程，而仅仅关注系数矩阵的处理过程。设计出发点是系数矩阵的三角分解方式，相应的算法称为直接三角解法。若所有计算都是准确的，直接三角解法同顺序高斯消元方法是完全等价的。但是，数据走向和运算次序是完全不同的。

1.2.1 矩阵三角分解

定义 1.3. *若存在上三角矩阵 $\mathbb{U}$ 和下三角矩阵 $\mathbb{L}$，使得*

$$\mathbb{A} = \mathbb{L}\mathbb{U}, \tag{1.2.8}$$

则称矩阵 $\mathbb{A}$ 具有 LU 三角分解。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。

请注意：按照这个定义，并不是所有的矩阵都具有三角分解，例如

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

那么，矩阵存在 LU 三角分解的条件是什么?

定理 1.3. *若 n 阶矩阵 $\mathbb{A}$ 的前 $n-1$ 个顺序主子式*

$$\mathbb{A}_k = \det \mathbb{A}(1:k, 1:k), \quad k = 1:n-1,$$

均非奇异，则顺序高斯消元法可以给出矩阵 $\mathbb{A}$ 的 Doolittle 分解。

✯ **说明 1.8.** 定理 *1.3* 中的条件仅仅是充分的。若关于顺序主子式的条件不成立，矩阵也可以有 LU 分解。譬如，

$$\begin{bmatrix} 0 & 0 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{1.2.9}$$

请问这个矩阵的 LU 三角分解是唯一的吗？

✯ **说明 1.9.** 由论题 *1.4* 可知，即使可逆矩阵 $\mathbb{A}$ 不存在三角分解，它也会有如下的三角分解

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

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。

矩阵的三角分解是不唯一的。为保证唯一性，我们需要考虑三角分解的标准形式，即两个三角形矩阵都是单位的（对角线元素恒为一）。

✇ **定义 1.4.** 称 $\mathbb{A} = \mathbb{LDR}$ 为一个 LDR 三角分解，若 $\mathbb{D}$ 为对角阵，$\mathbb{R}$ 和 $\mathbb{L}$ 分别为上下单位三角矩阵。

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

证明：数学归纳法与矩阵分块。见教科书。 □

1.2.2 矩阵分解的应用

基于矩阵的三角分解，我们可以给出相应的数值算法。设计思想比较简单，就是矩阵乘法公式。对于 LU 三角分解，有

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

其中某个上（下）三角阵的对角线要求恒为一。若 $\mathbb{L}$（或 $\mathbb{U}$）是单位三角阵，则相应的方法称为 Doolittle（或 Crout）方法。

Crout 方法

以 Crout 方法为例[iv]，它同顺序高斯消元法的主要区别是数值计算的次序不同。位于系数矩阵的操作数据呈瀑布型方式，由左上角向右下角移动，先列后行地依次更新一次。

在下面的图文框中，我们给出了 Crout 方法的伪代码片断。其中，数据覆盖技术被采用，分解出来的两个三角型矩阵数据依旧存储在原有数据的位置上。若引进列主元策略，我们只需在伪代码的第 5 行前，添加相应的补丁代码。请注意：在选取主元之前，我们要先完成相应列的计算。详略。

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

请注意：若求和符号中的上标小于下标，则对应的求和操作（对应代码的第 3 行和第 6 行）为空操作，相应的返回值为零。这个默认准则将在后续讨论中一直使用，我们将不再赘述。

直接三角解法在本质上也是高斯消元方法，它可视为不同的执行过程而已。但是，这样的算法具有某些方面的优势，例如它不必计算和存储消元的中间结果。

1. 在每次高斯消元过程中，右下角的整块矩阵都要更新。换言之，右

[iv] Doolittle 方法是类似的。事实上，Doolittle 算法就是顺序（或列主元）高斯消元方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。在 Matlab 中，相应的命令是 lu().

下角的元素将被多次地更新。计算涉及的数据范围非常大，使得数据指针的移动距离非常大，进而耗费大量的数据读写时间。

2. 在直接三角解法中，为更新某个位置的数据，我们仅仅需要读写同目标位置处于同行或同列的相关数据。换言之，直接三角解法是一种"需求驱动"的算法，在数据读写方面的代价有明显的下降。这个属性完全不同于高斯消元法。

请注意：它们的计算复杂度是一样的，效率的提升仅仅来自数据读写方面。

Cholesky 方法

若系数矩阵 $\mathbb{A}$ 是实对称正定矩阵，三角分解的计算复杂度还会进一步地降低。我们可采用著名的 Cholesky 方法[v] 或 $\mathbb{L}\mathbb{L}^\top$ 算法。在 Matlab 中，相应的命令是 chol().

定理 1.5. *设 $\mathbb{A}$ 是实对称的正定矩阵，则它有三角分解*

$$\mathbb{A} = \mathbb{L}\mathbb{L}^\top, \tag{1.2.10}$$

其中 $\mathbb{L}$ 是一个下三角阵。若要求 $\mathbb{L}$ 的对角线元素均为正数，则这种分解是唯一的。

基于三角分解 (1.2.10) 的方法称为 Cholesky 方法。它具有重要的理论价值。例如，我们可以利用它的计算过程，判断给定的对称矩阵是否正定。

✎ **论题 1.6.** *Cholesky 方法也常被称为平方根法，因为它的实现基于如下的矩阵元素计算公式*

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

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

我们需要开根号运算，才能得到对角线元素的取值。同乘除法运算相比，开根号运算将消耗大量的 *CPU* 时间。

注意到对称性，我们只需计算矩阵 $\mathbb{L}$ 的下三角部分。相应数据，可以按照逐列或者逐行的方式进行依次更新。因为逐行更新的方式难以实现并行化，故大多采用逐列更新的方式。

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

在左侧的图文框中，我们给出了平方根方法的伪代码片段，其中的 l_{ij} 覆盖存放在原有数据 a_{ij} 的位置。

逐列更新的方式也称为“向左看”算法。因为直到第 j 步外围循环时，矩阵的第 j 列数据才会被更新，故而这种算法也称为“需求驱动的算法”或者“延迟更新的算法”。

定理 1.6. *若矩阵 $\mathbb{A}$ 是对称正定的，则 $l_{ij} \leq \sqrt{a_{ii}}$，其中 $j \leq i$。*

因此说，$\mathbb{L}$ 中的元素均是可以控制的，相应的算法是数值稳定的。此时，我们无需进行主元的选取。

论题 1.7. *为避免平方根算法中的开根号运算（它比乘除法慢很多），我们可采用修正的平方根算方法。它基于所谓的标准三角分解*

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

其中 $\mathbb{L}$ 是单位下三角阵，$\mathbb{D}$ 是正数构成的对角阵，因为 $\mathbb{A}$ 是正定的。

在 Matlab 中，修正的平方根算法的相应命令是 ldl(). 在修正的平

方根算方法中，基本计算公式为

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

此时，数据的逐行处理是便捷的。在下面的两个图文框中，我们分别给出了修正平方根算法的两个伪代码片段。

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.
7.
8.
9. 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. $a_{ii} := a_{ii} - c a_{ij}$;
8. Enddo
9. Enddo

虽然左侧代码有效地避免了开根号运算，但是乘除法的总次数却比前面的平方根算法增加了一倍。为减少左侧代码（第 3 行和第 5 行）的重复运算，引进中间辅助变量

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

它对应右侧代码第 3 行中的 a_{ij} 和第 6 行的局部变量 c。右侧代码第 6 行的代码，对应 g_{ij} 到 l_{ij} 的过程，相应数据存储在系数矩阵的原有位置（见第 3 行代码中的 a_{jk}）。这个例子是有效缩减计算复杂度的典型代表。

☆ **说明 1.10.** 我们要指出，矩阵的对称正定性是非常重要的条件，它可以保证 $LDL^\top$ 算法具有良好的数值稳定性。

为说明上述断言，不妨考虑一个简单的反例，即

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

当 $|\varepsilon| \ll 1$ 很小时，矩阵 $\mathbb{A}$ 是不定的，上述 $LDL^\top$ 分解势必引起很大的舍入误差。但是，我们直接计算可知相应的逆矩阵为

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

直接按照这个公式计算，舍入误差的影响将非常小。

因此，对于不定的对称矩阵，我们常把列主元高斯消元法和二阶块高斯消去法相结合，来保证数值上的稳定性。

带状矩阵的分解

在大规模科学计算时，线性方程组的系数矩阵通常会含有大量的零元素。它们将虚耗巨额的数据存储空间和四则运算时间，降低高斯消元算法的处理能力和计算效率。为此，我们应当充分发掘矩阵的稀疏特性。例如，利用非零元素分布的结构特性，从数据存储和算法优化两个方面，对高斯消元方法进行改进。

☆ **说明 1.11.** 稀疏矩阵的数据存储是门相对繁杂的计算机技术。因篇幅有限，我们仅仅简要介绍一些基本技术，譬如变形存储，或者利用三元数结构体 (i, j, a_{ij}) 记录非零元素。为解决数据关联和便于快速搜索，我们需要引进单向链表或者双向链表等复杂数据结构。

在 *Matlab* 中，相关的基本命令有 *sparse(), speye(), spones(), spdiag(), full()* 等等。详细内容可参阅 *Matlab* 的帮助文件。

☆ **说明 1.12.** 当利用高斯消元方法法求解稀疏的线性方程组时，零元素在消元之后可能会变成非零元素。这使得稀疏矩阵变成稠密矩阵，造成数据存储的最大困难。换言之，我们需要适当控制新增的非零元素总数。这是数值处理的关键之处，最著名的数值策略有不完全三角分解。深入的讨论需要涉及图论的内容，详略。

在本讲义中，我们仅仅以简单的带状矩阵为例。所谓的带状矩阵，是指远离对角线一定距离的矩阵元素均为零。利用数学归纳法，可以证明：对于带状矩阵，相应 LU 分解中的两个三角型矩阵依旧具有相同的带状结构。事实上，在带宽内，矩阵可能依旧存在大量的零元素；相关的深入处理非常复杂，详略。

✍ **论题 1.8.** *设矩阵的半带宽为 d，我们可按斜线（或按行）存储技术存储矩阵，即仅仅用 $(2d-1)n$ 个存储单位替代普通的 n^2 个存储单位。请针对你的数据存储方式重写高斯消去法省略无用的零操作，并估算最终所需的乘除法次数是多少？*

作为稀疏矩阵的代表，三对角矩阵堪称最简单的等带宽矩阵。为表示简单，我们简记它为

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

其中 $\boldsymbol{a} = (a_i), \boldsymbol{b} = (b_i)$ 和 $\boldsymbol{c} = (c_i)$ 分别为从下至上的三条对角线向量。

✯ **说明 1.13.** *我们需强调指出：带状矩阵的逆矩阵通常不再是带状矩阵。因此，若无特别要求，我们很少主动计算逆矩阵。*

❀ **思考 1.2.** *带状矩阵的逆矩阵可能具有漂亮的结构。设 $\mathbb{H}$ 是不可约的上 Hessenberg 矩阵，即它比上三角矩阵多一条所有元素非零的副对角线。Ikebe (1979) 指出：存在两个列向量 $\boldsymbol{p} = (p_i)_{i=1}^n$ 和 $\boldsymbol{q} = (q_j)_{j=1}^n$，使得位于逆矩阵 $\mathbb{H}^{-1}$ 下三角区域的元素可表示为*

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

利用 *Ikebe* 的结果，我们可以给出一个直接算法，计算对称三对角矩阵的逆矩阵。具体内容留作练习[vi]。

追赶法

若 $\mathbb{A}$ 是三对角矩阵，相应的线性方程组求解算法通常称为追赶法，或者 Thomas 算法。它具有非常简单的结构。

✍ **论题 1.9.** 所谓的追赶法就是 *Crout* 算法及其两个三角型线性方程组的求解过程，此时的三角形矩阵仅仅有两个斜对角线元素非零。在下面的图文框中，我们给出了相应伪代码中的 *Crout* 分解片段。

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

矩阵分解所需的乘除法次数共计 $\mathcal{O}(2n)$，追和赶的过程需要 $\mathcal{O}(3n)$ 次乘除法。

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，我们无需进行主元的选取，相应的数值结果是稳定的。

[vi]通常，$q_1 = 1$ 被预先地设定。

证明：数学归纳法。见教科书。 □

对于三对角线性方程组，相应的求解方法有很多种变形，例如变参数追赶法和线性插值法。前者基于矩阵分解 $\mathbb{A} = \mathbb{D}\mathbb{L}\mathbb{R}$，其中 $\mathbb{D}$ 是对角阵，而 $\mathbb{L}$ 和 $\mathbb{R}$ 是相应的三角型矩阵。因篇幅有限，此处不再赘述。后者基于非齐次线性方程组解的线性结构；下面给予简要介绍。

先考虑前 $n-1$ 个方程形成的多解线性方程组，它的通解可表示为两个特解的线性组合

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

其中 θ 是待定系数。若所有的 c_i 均不等于零[vii]，则右上角的 $n-1$ 阶下三角矩阵块是可逆的，两个特解可以非常轻松地确定。至于待定系数 θ，我们可利用线性方程组的最后一个代数方程来确定。

❀ **思考 1.3.** *循环三对角方程组具有重要的应用价值。所谓的循环三对角阵就是，在原有三对角阵的右上角和左下角分别补充上相应的丢失元素。请利用矩阵分解技术，给出这个循环三对角方程组的追赶法实现过程。*

1.3 向量范数和矩阵范数

为探讨数值方法的性质，我们建立数值代数分析的两个基本度量工具，即所谓的向量范数和矩阵范数[viii]。范数是泛函分析的基本概念。在数值分析（或矩阵论研究）中，它的真正有效使用，是始于上世纪 40-50 年代。

[vii] 若某个 $c_i = 0$，则线性方程组可分割为两个小规模的问题。这样的问题称为可约的。

[viii] 本课程将更多地关注实数域上的向量和矩阵，虽然相应概念和结论可以非常容易地从实数域推广到复数域。

1.3.1 向量范数和矩阵范数的定义

定义 1.5. 向量范数的定义有三条规则：*(a)* 非负性；*(b)* 齐次性；*(c)* 三角不等式。具有某个范数度量的线性空间称为赋范空间。

设 $\boldsymbol{x} = (x_i)_{i=1}^n \in \mathbb{R}^n$，相应的 l_p 向量范数为

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

其中 l_2 范数也称为 Euclid 范数。

在 $\mathbb{R}^n$ 空间，我们还可以定义两个向量的内积

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

它满足著名的 Hölder 不等式：

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

定义 1.6. 矩阵范数的定义有四条规则：*(a)* 非负性；*(b)* 齐次性；*(c)* 三角不等式；*(d)* 相容性。

说明 1.14. 在上述定义中，关于相容性的规则是非常重要的。前三条可视为向量范数的自然推广。

思考 1.4. 设 $\mathbb{A} = (a_{ij})$ 是一个 n 阶矩阵，请问 $\max_{ij} |a_{ij}|$ 是否是一个矩阵范数？那么，$n \max_{ij} |a_{ij}|$ 呢？

在 Matlab 中，向量范数和矩阵范数的相应命令都是 norm().

定理 1.8. 任意的两个向量（或矩阵）范数均是彼此等价的，且它们关于其元素的变化都是（一致）连续的。

证明：见教科书。 □

1.3.2 向量范数和矩阵范数的联系

定义 1.7. 设 $\|\cdot\|_\alpha$ 为矩阵范数，而 $\|\cdot\|_\beta$ 为向量范数。若成立

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

则称矩阵范数 $\|\cdot\|_\beta$ **相容**于向量范数 $\|\cdot\|_\alpha$。

特别地，若存在非零向量，使得上述不等式变成等号成立，则称矩阵范数 $\|\cdot\|_\beta$ **从属**于向量范数 $\|\cdot\|_\alpha$.

定理 1.9. 对任意的矩阵范数 $\|\cdot\|_\beta$，均存在某个向量范数 $\|\cdot\|_\alpha$，使得两者是相容的；但是，它们不一定具有从属关系，因为

$$\|\mathbb{I}_{n\times n}\|_\beta = 1$$

是矩阵范数 $\|\cdot\|_\beta$ 从属于某个向量范数 $\|\cdot\|_\alpha$ 的必要条件。

Frobenius 范数（又称为 Suchur 范数）

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

它与 l_2 向量范数相容，但不从属于任何向量范数。

思考 1.5. 证明 $\|\mathbb{A}\mathbb{B}\|_F \leq \|\mathbb{A}\|_F \|\mathbb{B}\|_F$，进而说明 $\|\cdot\|_F$ 确实是矩阵范数。

定理 1.10. 对任意的向量范数 $\|\cdot\|_\alpha$，我们均可导出算子范数

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

它是同向量范数 $\|\cdot\|_\alpha$ 从属（显然相容）的矩阵范数。这个定义可推广到任意形状的矩阵。

论题 1.10. 分别对应三个常用的向量范数 $(1, 2, \infty$ 范数$)$，我们可得如下三个（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 $\varrho(\mathbb{A}^\top\mathbb{A})$ 是矩阵 $\mathbb{A}^\top\mathbb{A}$ 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，$\|\mathbb{A}\|_1 = \|\mathbb{A}^\top\|_\infty$ 和 $\|\mathbb{A}^\top\|_2 = \|\mathbb{A}\|_2$.

定理 1.11. 任意（相容）矩阵范数均满足 $\varrho(\mathbb{A}) \leq \|\mathbb{A}\|$.

定理 1.12. 对于任意正常数 ε，至少存在一个矩阵范数 $\|\cdot\|_\star$，使得

$$\|\mathbb{A}\|_\star \leq \varrho(\mathbb{A}) + \varepsilon. \tag{1.3.11}$$

定理 1.13. *(Banach 引理)* 设某个范数 $\|\cdot\|$，使得单位矩阵 $\mathbb{I}$ 满足 $\|\mathbb{I}\| = 1$。若 $\|\mathbb{A}\| < 1$（或者 $\varrho(\mathbb{A}) < 1$）时，则 $\mathbb{I} \pm \mathbb{A}$ 可逆，且

$$\frac{1}{1 + \|\mathbb{A}\|} \leq \|(\mathbb{I} \pm \mathbb{A})^{-1}\| \leq \frac{1}{1 - \|\mathbb{A}\|}. \tag{1.3.12}$$

定理 1.14. 矩阵的谱范数和 *Frobenius* 范数，在（左右）酉变换下均保持不变。

1.4 线性方程组的摄动理论

本节讨论线性方程组的数据敏感程度。当线性方程组的已知数据发生变化时，相应的解向量通常也会发生改变。解向量的变化是否受到限制，是一个非常重要的问题。如果变化量无法有效限制，则数值计算很难得到好结果。

1.4.1 条件数

对于线性方程组而言，解向量的变化敏感度同系数矩阵的条件数密切有关。条件数是线性方程组的固有可靠性，它描述了已知数据的改变对于解向量变化的影响程度。要奢求数值算法的计算可靠性超过原有问题的固有可靠性，通常都是无望的追求。

定义 1.8. 可逆矩阵 $\mathbb{A}$ 的条件数定义为

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

其中 $\|\cdot\|$ 是某个矩阵范数。若条件数非常大[ix]，则称矩阵 $\mathbb{A}$ 是病态的；否则，称矩阵 $\mathbb{A}$ 是良态的。

若矩阵 $\mathbb{A}$ 是实对称正定的，则谱条件数为

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

其中 $\lambda_{\max}$ 和 $\lambda_{\min}$ 为最大特征值和最小特征值。虽然这个量的数值计算是比较困难的，但是它却比较适宜于理论分析。

论题 1.11. 由定义可知，条件数具有如下的基本性质：

1. $\kappa(\mathbb{A}) \geq 1$;
2. $\kappa(c\mathbb{A}) = \kappa(\mathbb{A}), 0 \neq c = const$;
3. $\kappa(\mathbb{A}^{-1}) = \kappa(\mathbb{A})$;
4. $\kappa(\mathbb{A}\mathbb{B}) \leq \kappa(\mathbb{A})\kappa(\mathbb{B})$.
5. 任意的两个条件数都是等价的。

说明 1.15. *Hilbert* 矩阵 $\mathbb{H}_n = (h_{ij})$ 是著名的病态矩阵，其中

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

[ix]这是一个相对概念下的陈述，通常同计算环境（或机器精度）有关。

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，相应的命令是 *hilb()* 和 *invhilb()*.

其它的著名病态矩阵还有范得蒙矩阵等等。

定理 1.15. （*Kahan*, *1996*）可逆矩阵 $\mathbb{A}$ 的病态程度描述了它同奇异矩阵集合的接近程度，因为

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

换言之，条件数越大，矩阵越接近奇异。

证明：显然，利用 Banach 引理可知，结论的左端必然大于或等于右端。下面，我们只需证明等号是可以取到的。令

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

其中 $\boldsymbol{x}$ 为单位长度向量，使得 $\|\mathbb{A}^{-1}\boldsymbol{x}\|_2=\|\mathbb{A}^{-1}\|_2$。至此，定理结论是不难验证的，因为 $(\mathbb{A}+\delta\mathbb{A})\boldsymbol{y}=0$。 □

☆ **说明 1.16.** 利用行列式的大小度量矩阵是否病态，是很自然的想法。但是，两个概念几乎没有任何的关系。例如，矩阵

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

的行列式为 1，但 $\kappa(\mathbb{A}_n)=n2^{n-1}$。另一方面，一个非常良态的矩阵行列式可能非常小，例如对角线元素均为 10^{-1} 的 n 阶对角阵。

下面，考虑一个简单的 2 阶矩阵 $\mathbb{A}$ 及其逆矩阵

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

其中的 ε 是很小的正数。显然，矩阵 $\mathbb{A}$ 的谱条件数为

$$\kappa_2(\mathbb{A}) = (2+\varepsilon+\sqrt{4+\varepsilon^2})^2/(4\varepsilon) \approx 4/\varepsilon.$$

当 ε 靠近零时，矩阵 $\mathbb{A}$ 变得很病态；此时的逆矩阵 $\mathbb{A}^{-1}$ 关于 ε 的变化非常敏感。在高斯列主元消去过程中，相应的 LU 三角分解为

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

其中 $\mathbb{L}_\varepsilon$ 是良态的，而 $\mathbb{U}_\varepsilon$ 是病态的。我们不难证明（留为作业）：以 ε 为参数，$\mathbb{U}$ 的条件数同 $\mathbb{A}$ 的条件数处于同样的量级。请思考，这是一个普遍的现象吗?

1.4.2 摄动分析

下面建立本章最重要的理论结果。

考虑线性方程组 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$。设系数矩阵和右端项分别产生 $\delta\mathbb{A}$ 和 $\delta\boldsymbol{b}$ 的扰动，相应的解向量改变量记为 $\delta\boldsymbol{x}$。经过相对简单的分析，可以建立下面的扰动估计：设 $\|\mathbb{A}^{-1}\|\|\delta\mathbb{A}\| < 1$，则

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

粗略地讲，解向量的相对改变量 $\delta \boldsymbol{x}$ 都可以被有效地控制，并且同系数矩阵的条件数 $\kappa(\mathbb{A})$ 都几乎成正比例。

在一般的理论框架下，上述扰动估计是很难改进的，因为不等式中的等号是可以取到的（请给出实例）。但是，大量的数值经验却表明，这个结论是非常保守的，因为真正遇到等号成立的概率并不高。譬如，考虑一个简单的二阶线性方程组

$$\begin{bmatrix} \gamma & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \gamma \\ 1 \end{bmatrix}.$$

显见，当对角线元素发生改变时，真实扰动的放大率应该为 1。但是，问题的矩阵条件数显然同 γ 有关。若 $\gamma \gg 1$，上述扰动估计的缺陷是非常明显的，因为它给出的上界同真实误差相距甚远。

因此，摄动估计的进一步改善一直是数值工作者的努力目标。目前，对于某些具有特定性质的现象方程组，许多可用的但略带风险的上界估计被相继提出。详细的处理方式，略。

1.4.3 精度分析

在数值实践中，如何判定计算结果的可靠性是一个非常重要的实际问题。常用的方法之一是利用同型的线性方程组，进行所谓的试算，即借用已知问题的真解，通过数值解和真解的比较，测算出结算结果中的可靠位数。另一种常用的方法是基于所谓的后验误差估计

$$\frac{\|\boldsymbol{x}_\star - \boldsymbol{x}_{\text{num}}\|}{\|\boldsymbol{x}_\star\|} \leq \kappa(\mathbb{A}) \frac{\|\boldsymbol{r}\|}{\|\boldsymbol{b}\|}, \tag{1.4.13}$$

其中 $\boldsymbol{r} = \mathbb{A}\boldsymbol{x}_{\text{num}} - \boldsymbol{b}$ 是一个可信任[x]的可计算数值残量，$\boldsymbol{x}_{\text{num}}$ 是计算机给出的数值解。

[x]通常，这个残量的计算结果是较为可信的，因为它的舍入误差积累要远远小于高斯消元过程中的误差积累。

为计算方便，在后验误差估计 (1.4.13) 中，通常取无穷范数。此时的关键问题是给出条件数 $\kappa_\infty(\mathbb{A})$ 的一个合理估计。为此，需要分别估算 $\|\mathbb{A}\|_\infty$ 和 $\|\mathbb{A}^{-1}\|_\infty$. 前者的计算很简单，而后者[xi]的计算却略为困难，原因有二。首先，我们不会真正计算出逆矩阵 $\mathbb{A}^{-1}$；其次，逆矩阵 $\mathbb{A}^{-1}$ 的计算结果同样面临着可靠性的问题。在 Matlab 中，矩阵条件数可用命令 rcond() 给出。

✯ **说明 1.17.** *数值解的精度可通过迭代进行改进，其基本思想就是对残量进行同类型的线性方程组修正。略。*

1.5 列主元高斯消元法的数值稳定性分析

因为数字计算机仅能提供有限的字节位长，于是在浮点数据的存储和计算过程中，舍入误差的产生和积累是不可避免的，必然影响理论上精确的高斯消元方法。本节简略介绍列主元高斯消元法的数值稳定性，即数值解关于舍入误差的敏感程度。换言之，数值相对误差不仅依赖系数矩阵的条件数，还依赖计算机的浮点运算精度。

[xi]通常，我们会借用任意列范数 $\|\mathbb{B}\|_1$ 的一个估算方法。它是著名的最优化方法“盲人下山法”的一个应用，已被 LAPACK 中所采用。因篇幅有限，我们略过相关的理论解释，仅给出其算法描述。

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；

令 $\mathbb{B} = \mathbb{A}^{-\top}$，我们可得 $\|\mathbb{B}\|_1 = \|\mathbb{A}^{-1}\|_\infty$。在调用上述估算方法时，我们无需在第 2 步计算付出过高的代价。因为，我们可以利用 $\mathbb{A}$ 的高斯消元过程中得到的 $\mathbb{L}\mathbb{U}$，进行两个三角型线性方程组的快速求解。

1.5.1 浮点运算

首先，让我们快速回顾一下计算机浮点数表示及其运算。通常，浮点数[xii]可以表示为

$$f = \pm 0.d_1 d_2 \cdots d_t \times 2^J, \quad d_1 \neq 0,$$

其中整数 J 的取值范围决定了计算机浮点数的取值范围，t 是计算机的最大位长。最大位长决定了所谓的机器精度，即计算机能够表示的最小正数 2^{1-t}。对于双精度浮点数（$t \approx 64$）而言，最小的正浮点数位于 10^{-16} 或 10^{-17} 量级。

请注意：计算机上的全部浮点数仅仅是实数域的离散子集，不再具有四则运算（加减乘除）的封闭性。令 $\star$ 表示任意一种运算。简单的分析可知，相应的浮点数运算满足估计

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

应用这个估计，当 $n\vartheta$ 较小的时候，可知向量的内积满足

$$|fl(\boldsymbol{x}^\top \boldsymbol{y}) - \boldsymbol{x}^\top \boldsymbol{y}| \leq 1.01 n\vartheta |\boldsymbol{x}|^\top |\boldsymbol{y}|, \tag{1.5.14}$$

其中的 n 为向量的维数。类似地，矩阵的数乘、加法和乘法分别满足

$$fl(\alpha \mathbb{A}) = \alpha \mathbb{A} + \mathbb{E}, \qquad |\mathbb{E}| \leq \vartheta |\alpha \mathbb{A}|, \tag{1.5.15a}$$

$$fl(\mathbb{A} + \mathbb{B}) = \mathbb{A} + \mathbb{B} + \mathbb{E}, \quad |\mathbb{E}| \leq \vartheta |\mathbb{A} + \mathbb{B}|, \tag{1.5.15b}$$

$$fl(\mathbb{A}\mathbb{B}) = \mathbb{A}\mathbb{B} + \mathbb{E}, \qquad |\mathbb{E}| \leq 1.01 n\vartheta |\mathbb{A}||\mathbb{B}|, \tag{1.5.15c}$$

其中的 n 为矩阵的阶数，$\mathbb{E}$ 为最终的误差积累。这里的绝对值运算是指向量或矩阵的所有元素分别取绝对值，相应的不等式为对应元素的比较。

[xii] 实际上，$d_1 = 1$ 是不需要存储的。

上述工作即俗称的向前误差分析技术。相应的讨论过程较为繁琐，严重地依赖于具体计算环境。因此，数值分析更多地采用向后误差分析技术。换言之，假定所有运算均是精确的，将舍入误差的产生和积累归结为初始数据的扰动。例如，(1.5.15a) 可表示为

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

其中 $\mathbb{E}$ 为扰动矩阵。向后误差分析技术可以将计算机上的浮点运算完全转化为实数域上的精确运算，非常便于理论分析。

1.5.2 算法的舍入误差分析

设 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$，其中 $\mathbb{A} = (a_{ij})$ 是系数矩阵。利用向后误差分析技术，列主元高斯消元法的（受舍入误差影响的）数值解 $\boldsymbol{x} + \delta\boldsymbol{x}$，可以视为扰动问题

$$(\mathbb{A} + \delta\mathbb{A})(\boldsymbol{x} + \delta\boldsymbol{x}) = \boldsymbol{b} \tag{1.5.16}$$

的精确解，其中 $\delta\boldsymbol{x}$ 是因舍入误差而产生的偏差。利用线性方程组的摄动理论，要估计 $\delta\boldsymbol{x}$，只需给出摄动矩阵 $\delta\mathbb{A}$ 的合理估计。

结合列主元高斯消元算法的具体实现过程，相应的矩阵分解（或者消元）过程可以表示为

$$\mathbb{P}\mathbb{A} + \mathbb{E} = \mathbb{L}\mathbb{U},$$

其中 $\mathbb{E} = (e_{ij})$ 是扰动矩阵，$\mathbb{P}$ 是置换矩阵，$\mathbb{L} = (\ell_{ij})$ 和 $\mathbb{U} = (u_{ij})$ 是计算机上真正存储的两个三角型矩阵。后续的计算过程可描述为两个三角形方程组

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

的精确计算，其中 $\mathbb{F} = (f_{ij})$ 和 $\mathbb{G} = (g_{ij})$ 是相应的两个扰动矩阵。综上

所述，可知 (1.5.16) 的扰动矩阵为

$$\delta\mathbb{A} = \mathbb{E} + \mathbb{P}(\mathbb{F}\mathbb{U} + \mathbb{L}\mathbb{G} + \mathbb{F}\mathbb{G}). \tag{1.5.17}$$

利用数学归纳法和向前误差分析技巧，可知三个扰动矩阵元素满足

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

其中 ϑ 是机器精度，n 是矩阵的阶数，$a_{ij}^{(k)}$ 是在第 k 步高斯消元后计算机所保存的具体数据。

在列主元高斯消元过程中，$\mathbb{L}$ 的元素均以 1 为上界，相应的行范数满足 $\|\mathbb{L}\|_\infty \leq n$。为给出 $\|\mathbb{U}\|_\infty$ 的估计，定义主元增长因子

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

因此，有 $\|\mathbb{U}\|_\infty \leq n\eta(\mathbb{A})\|\mathbb{A}\|_\infty$。从而，当 $n\vartheta$ 较小的时候，由 (1.5.17) 可知[xiii]

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

其中 $C \approx 10$ 为绝对常数。利用线性方程组的摄动理论，可知：若相对扰动很小，最终的舍入误差满足

$$\frac{\|\delta\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} \leq Cn^3\vartheta\eta(\mathbb{A})\kappa_\infty(\mathbb{A}), \tag{1.5.18}$$

其中 C 是一个同 n 无关的绝对常数。

[xiii]在下面的两个估计中，关于 n 的低阶项被略去了。

(1.5.18) 表明：对于常见的线性方程组，因舍入误差而产生的相对误差可以得到较为有效的控制。换言之，列主元高斯消元方法是数值稳定的可行算法。

✯ **说明 1.18.** 结果 (1.5.18) 含有主元增长因子 $\eta(\mathbb{A})$。在每次执行列主元消元之后，对角线右上方的元素绝对值至多放大 2 倍。因此，$\eta(\mathbb{A})$ 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，$\eta(\mathbb{A})$ 通常处于 $n^{2/3}$ 或 $n^{1/2}$ 的量级，并没有理论上的那么可怕。

第 2 章
线性方程组的迭代解法

当线性方程组的规模越来越大的时候，直接解法遇到的障碍也将越来越多，譬如无法忍受的海量数据存储和（非线性多项式的）计算复杂度等等。此时，简单易行的迭代思想将脱颖而出，并显现出相应的数值优势。换言之，此时的数值目标不再是通过有限步数的四则运算给出精确解，而是构造一个可以自动生成的快速收敛向量序列。一个具有竞争力的迭代解法应具有下面的优点：其一，无需改变矩阵数据；其二，数值编程容易，可以同稀疏矩阵的存储方式相匹配；其三，迭代序列快速收敛到精确解。

2.1 基本理论

所谓迭代解法就是通过简单的计算规则，自动生成一个向量序列

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

其中前 r 个向量 $\{\boldsymbol{x}_k\}_{k=0}^{r-1}$ 是需要人工给出的启动初值。由于迭代函数 $\boldsymbol{f}_k$ 包含 r 个历史信息，故而称该方法是 r 阶的。若迭代函数同迭代步数 k 无关，则称方法是定常的；否则，称方法是非定常的。

论题 2.1. 后续讨论均默认迭代方法是完全相容的，即线性方程组的精确解

$$x_\star = \mathbb{A}^{-1}\boldsymbol{x}$$

一直（或者当步数充分大以后）满足迭代公式 (2.1.1)。

2.1.1 一阶迭代方法

为简单起见，首先讨论线性方程组 $\mathbb{A}\boldsymbol{x}=\boldsymbol{b}$ 的一阶迭代方法。通常，它具有两种表示形式：

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

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

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 $\boldsymbol{x}_k=\boldsymbol{x}_\star$ 是精确解。

✯ **说明 2.1.** 考虑一个简单的极端情况：若定义 $\mathbb{H}_{k+1}=\mathbb{A}^{-1}$，则均有 $\boldsymbol{x}_{k+1}=\boldsymbol{x}_\star$。因此说，预处理矩阵应是 $\mathbb{A}^{-1}$ 的某种近似。

✎ **论题 2.2.** 上述两个迭代公式可以互相导出。特别地，若

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

则第二种表述可以导出第一种表述。

因此，迭代方法的研究重点是如何构造迭代矩阵（或者预处理矩阵），使 $\boldsymbol{x}_k$ 能够快速地收敛到精确解。

2.1.2 收敛性分析

准备知识

关于向量序列和矩阵序列，收敛的定义是非常直接的，即所有分量都是收敛的。理论分析多用范数进行度量。

定义 2.1. $\lim_{k\to\infty}\boldsymbol{x}_k=\boldsymbol{x}\Leftrightarrow\lim_{k\to\infty}\|\boldsymbol{x}_k-\boldsymbol{x}\|=0.$

定义 2.2. $\lim_{k\to\infty}\mathbb{A}_k=\mathbb{A}\Leftrightarrow\lim_{k\to\infty}\|\mathbb{A}_k-\mathbb{A}\|=0.$

判断矩阵序列（或者矩阵级数）是否收敛，矩阵范数和谱半径是常用的分析工具。主要结论有

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

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

收敛性判定

迭代方法 (2.1.2) 的第 k 步误差记为

$$\boldsymbol{e}_k=\boldsymbol{x}_k-\boldsymbol{x}_\star.$$

既然迭代方法是完全相容的，相应的误差方程是

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

若对任意的初始向量 $\boldsymbol{e}_0$，均有 $\lim_{k\to\infty}\boldsymbol{e}_k=\boldsymbol{0}$，则称方法是收敛的。否则，称方法是发散的。

定理 2.3. 迭代方法收敛等价于迭代矩阵的乘积极限是零，即

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

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：

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

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。

收敛速度的刻画与估计

通常用误差向量 $\boldsymbol{e}_k$ 趋于零的（最坏）速度，衡量迭代算法的收敛快慢。譬如，一阶定常迭代方法 $\boldsymbol{x}_k=\mathbb{G}\boldsymbol{x}_{k-1}+\boldsymbol{g}$ 满足误差估计

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

右端系数 $\|\mathbb{G}^k\|$ 通常是无法改善的，因为等号成立的情形是真实存在的。下面以这个极端保守的数量 $\|\mathbb{G}^k\|$，作为收敛速度的讨论起点。

✎ **论题 2.3.** 为直观起见，通常用误差下降的平均效应刻画迭代误差趋零的速度：

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

这些概念产生于上世纪五六十年代，特别地，渐近收敛速度是由 *Young* 在 *1954* 年给出的。通常，我们用渐近收敛速度评判迭代算法的优劣程

度。其理论依据是：**迭代误差达到指定要求的最小迭代次数，同渐近收敛速度的倒数具有正比例关系。**

❀ **思考 2.1.** 事实上，上述两个收敛速度概念均是基于所谓的平均意义，它们同迭代误差在每一步的真实变化速度，还是具有一定的差距。为理解上述概念，观察谱范数 $\|\mathbb{A}^m\|_2$ 和 $\|\mathbb{B}^m\|_2$ 的发展过程，其中

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

显然，$\mathbb{A}$ 的谱半径略小一些。假设 α 靠近 *1*，请问：当 m 较小的初始阶段，是否发生 $\|\mathbb{A}^m\|_2>\|\mathbb{B}^m\|_2$ 的现象呢？这个实例也说明，迭代误差的初始表现同其渐近表现可以是截然不同的。

✯ **说明 2.2.** 再次强调指出，收敛快慢的评判标准是基于最差数值表现。事实上，由 (2.1.3) 可知，真实误差的下降速度还同初始向量有关，甚至产生非常明显的差异。

❀ **思考 2.2.** 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 $\varrho(\mathbb{G})$ 均难以直接计算，相应的收敛速度仅仅具有理论价值。相对而言，矩阵范数 $\|\mathbb{G}\|$ 可以简单计算出来，可以用于界定当前误差的大小。

定理 2.4. 若 $\|\mathbb{G}\|<1$，则定常迭代方法 $\boldsymbol{x}_k=\mathbb{G}\boldsymbol{x}_{k-1}+\boldsymbol{g}$ 是收敛的。相应的迭代误差满足

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 $\delta_k \equiv \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|$ 称为相邻误差，也是可以直接计算的。

✯ **说明 2.3.** 数值计算通常采用 1 范数或者无穷范数，而理论证明常常采用欧氏范数。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，相应的数值计算是比较困难的。

✯ **说明 2.4.** 在先验误差估计中，右端的上界是不可计算的；而在后验误差估计中，右端的上界是可以计算的。显然，这些估计都是相当保守的估计，实际误差可能远远小于这些上界估计。

2.1.3 停机标准

在迭代方法中，何时停机也是一个重要的问题。当然，我们希望数值误差达到用户要求[i]

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

即可停机，其中 $\mathcal{E}$ 是用户事先给出的停机指标。但是，这种停机策略只能作于理论研究（或数值实验），因为迭代误差是无法计算的量。

实际计算常常采用下面三个停机准则，特别是后两个同准则 (2.1.5) 没有明确的等价关系。它们分别是：

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

第三个标准源于后验误差估计 (I) 和关于 $\|\mathbb{G}\|$ 的估算。

❀ **思考 2.3.** 注意 $\boldsymbol{r}_k$ 同 $\boldsymbol{e}_k$ 的关系，给出相应两个停机准则的联系。

[i] 这是绝对误差。当然，我们也可使用相对误差。

☆ **说明 2.5.** 还要强调指出：舍入误差也会对迭代格式的收敛性，或者停机判断产生影响，特别是线性方程组非常病态的时候。

当应用后面两个停机准则的时候，用户还要小心出现所谓的“假停机”现象。例如，考虑如下的简单实例

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

其迭代矩阵的范数接近 1。当利用具有 *6* 位有效数字的 *10* 进制计算机求解时，取 $\mathcal{E} = 10^{-5}$。若初值为 $\boldsymbol{x}_0 = (0.1, 0.1)^\top$，则每次迭代仅让每个分量产生 10^{-6} 的增量，使得 $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty = 10^{-6} < \mathcal{E}$，从而出现所谓的“假停机”现象。

本课程重点关注迭代方法的方法误差。因篇幅有限，我们略去舍入误差的分析。

2.2 古典迭代算法

Jacobi（J）方法和 Gauss-Seidel（GS）方法属于古典迭代算法，都是不含参数的一阶定常迭代方法，具有类似的实现方式。

2.2.1 基本算法

J 方法基于同步更新策略，而 GS 方法基于异步更新策略。迭代序列按照如下方式更新每个分量：

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

两者相比，J 方法可以同时更新所有分量，无需等待其他分量的更新完成，而 GS 方法需要等待其它分量的更新，执行过程有先后次序。但是，J 方法需要两组工作单元记录解向量，而 GS 方法只需一组工作单元记录解向量。

1. J 方法由 Jacobi (1845 年) 提出。相关工作还包括 Geiringer (1945) 的同步位移法，以及 Killer (1958) 的 Richardson 方法。从物理角度来讲，J 方法也称为阻尼法。

2. GS 方法由 Gauss (1822) 最早提出，用于求解最小二乘问题的法方程组。而后，GS 方法由 Seidel (1874) 再次提出，却遭到自身的弃用。Von. Misers 和 Pollaczek-Geiringer (1949) 给出了最早的理论分析，使 GS 方法重新获得重视。

2.2.2 矩阵分裂方式

一阶定常迭代方法可以利用“矩阵分裂”进行构造。假设

$$\mathbb{A} = \mathbb{Q} - \mathbb{R},$$

其中 $\mathbb{Q}$ 是逼近 $\mathbb{A}$ 且容易求逆的矩阵，称为 $\mathbb{A}$ 的主体矩阵；有时，它也称为预处理矩阵。利用等价的不动点方程 $\boldsymbol{x} = \mathbb{Q}^{-1}(\mathbb{R}\boldsymbol{x} + \boldsymbol{b})$，定义（不动点）迭代方法

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

通常，系数矩阵 $\mathbb{A}$ 可以分割为对角线部分、严格[ii]下三角部分和严格上三角部分，即

$$\mathbb{A} = \mathbb{D} - \mathbb{DL} - \mathbb{DU}. \tag{2.2.7}$$

[ii]严格是指对角线元素也等于零。

若选取对角矩阵 $\mathbb{D}$ 或下三角矩阵 $\mathbb{D}-\mathbb{DL}$ 作为预处理矩阵，相应的矩阵分裂方式可以分别导出两个古典算法，即 J 方法和 GS 方法的迭代矩阵分别是

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

算法收敛的充要条件是相应的迭代矩阵谱半径小于一。

☆ **说明 2.6.** *迭代矩阵的谱与线性方程组的行排序有关，譬如，我们用 J 方法求解两个同解的线性方程组*

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

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

2.2.3 收敛性分析

事实上，J 方法和 GS 方法的收敛性和收敛速度没有关系。只有当矩阵具有某些特殊结构的时候，相应的收敛结论才会较为明确。

若以 J 方法为起点，有

定理 2.5. *若 $\|\mathbb{B}\|_\infty<1$，则 GS 方法也收敛，并比 J 方法更快。*

定理 2.6. *若 $\|\mathbb{B}\|_1<1$，则 GS 方法也收敛。*

若以系数矩阵为起点，有

定理 2.7. *若系数矩阵 $\mathbb{A}$ 是对角占优的，即或者是强对角占优，或者是弱对角占优且不可约，则 J 方法和 GS 方法均收敛。*

1. 强对角占优：对任意的 i 均有 $|a_{ii}| > \sum_{j\neq i} |a_{ij}|$;
2. 弱对角占优：对任意的 i 均有 $|a_{ii} \geq \sum_{j\neq i} |a_{ij}|$，且至少有一个不等式是严格成立的;
3. 不可约：大规模问题不能分解出独立求解的小规模问题。

定理 2.8. 设系数矩阵 $\mathbb{A}$ 是对称的，则

1. 若 $\mathbb{A}$ 正定，则 GS 方法必然收敛;
2. 若 $\mathbb{A}$ 和 $2\mathbb{D} - \mathbb{A}$ 均正定，则 J 方法收敛；反之亦然。

换言之，J 方法比 GS 方法的适用范围要窄。

☆ **说明 2.7.** 上述定理的证明过程将充分展示迭代方法收敛性分析的两个基本技巧：**范数估计**或**特征值估计**。具体内容见教科书。

☆ **说明 2.8.** 通常，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，在两个系数矩阵

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

中，前者的对角占优性更强一些，但是相应的 J 方法却具有略慢的收敛速度，因为迭代矩阵满足 $\varrho(\mathbb{B}_1) > \varrho(\mathbb{B}_2)$.

2.3 逐次超松弛方法

逐次超松弛[iii]方法具有极其重要的历史地位，它极大地拓展了迭代方法的设计思路：对旧的迭代序列进行加权平均，可以期待新的迭代序列具有更快的收敛速度。

✍ **论题 2.4.** *SOR 方法是以 GS 方法为蓝本，逐一加权平均两个新旧向量的对应分量。具体更新方式是*

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

其中 ω 称为松弛因子。相应的迭代矩阵是

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

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。*

2.3.1 收敛性

为保证 SOR 方法收敛，松弛因子 ω 要满足适当条件。

定理 2.9. *SOR 方法收敛的必要条件是 $0 < \omega < 2$.*

定理 2.10. *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：这是特征值估计的典型实例。见教科书。 □

[iii] Successive over-relax = SOR.

2.3.2 最佳松弛因子

数值实验表明：松弛因子 ω 影响 SOR 方法的收敛速度。特别地，在某些情况下，SOR 方法存在最佳松弛因子，使得收敛速度获得显著提升。关于最佳松弛因子的研究，极大地促进了迭代方法的发展，特别是在上世纪 80 年代之前。

✍ **论题 2.5.** 最佳松弛因子的具体设定，同线性方程组系数矩阵非零元素的分布结构相关。下面主要关注“相容次序”和“性质 $\boldsymbol{A}$”等结构，详细的内容请见教科书。常用的结论有

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

在椭圆型方程的数值求解中，离散而成的线性方程组常常满足以下情形：所有未知量归属于两个互不相交的集合，并且同属一个集合的未知量之间没有任何关联[iv]。这样的矩阵结构称为性质 A。换言之，存在某个置换阵 $\mathbb{P}$，使得系数矩阵满足

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

其中 $\mathbb{D}_1$ 和 $\mathbb{D}_2$ 是对角阵。若系数矩阵直接具有 (2.3.10) 右侧的结构，相应的未知量编号方式称为红黑（或棋盘）编号原则。事实上，置换阵的作用是未知量和方程的编号发生同样的改变。

[iv] 所谓两个变量具有关联，是指它们同时出现在一个线性方程中。

若系数矩阵 $\mathbb{A}$ 具有性质 A（甚至相容次序），J 方法同 SOR 方法的迭代矩阵在特征值方面具有某种联系。为简化理论分析的复杂程度[v]，不妨直接假设系数矩阵 $\mathbb{A}$ 恰好具有 (2.3.10) 右侧的矩阵结构。换言之，J 方法的迭代矩阵[vi]是

$$\mathbb{B} = \mathbb{L} + \mathbb{U} = \begin{bmatrix} \mathbb{O} & \mathbb{H} \\ \mathbb{K} & \mathbb{O} \end{bmatrix}.$$

定理 2.11. *设 λ 是 SOR 迭代矩阵 $\mathbb{T}_\omega$ 的特征值，而 μ 是 J 迭代矩阵 $\mathbb{B}$ 的特征值。它们具有如下的对应关系：*

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

请注意：此时的 μ 和 $-\mu$ 是成对出现的。

证明：利用相似变换的分块矩阵表述

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

对于 J 方法和 SOR 方法，分别计算相应的迭代矩阵特征值，即可得到定理结论。参见教科书。 □

定理 2.12. *设 n 阶矩阵 $\mathbb{B}$ 的所有特征值均可表达为*

$$\mu_j = \alpha_j + \sqrt{-1}\beta_j,$$

其中 α_j 和 β_j 均为实数。若存在正数 D，使得

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

[v] 繁琐的证明可参阅教课书

[vi] 为简单起见，非对角块部分仍采用了原有符号。

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

因此，若矩阵 $\mathbb{B}$ 的特征值 μ_j 均为实数，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：注意到 (2.3.11)，再利用实系数二次方程的根按模小于一同方程系数的关系。 □

下面考虑一个简单的情形：假设 J 方法迭代矩阵 $\mathbb{B}$ 的特征值均为实数，且按模小于 1。此时，SOR 方法的最佳松弛因子问题可以相对容易地得到解决。

✎ **论题 2.6.** 首先，固定 $\mathbb{B}$ 的某个特征值 μ。考虑直线

$$y = \frac{\lambda + \omega - 1}{\omega},$$

同抛物线

$$y^2 = \lambda |\mu|^2,$$

在 (λ, y) 实平面上的交点，其中 ω 为（松弛因子）参数。

不妨设交点存在[vii]，相应的横坐标 $\lambda_1(\omega)$ 和 $\lambda_2(\omega)$ 均为实数，对应二次方程 (2.3.11) 的根，是 *SOR* 迭代矩阵 $\mathbb{T}_\omega$ 的特征值。当 ω 变化时，要使 $\max_{i=1,2} |\lambda_i(\omega)|$ 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 μ 的最佳参数和切点位置是

$$\omega = \frac{2}{1 + \sqrt{1 - \mu^2}}, \quad \max_{i=1,2} |\lambda_i(\omega)| = |\omega - 1|. \tag{2.3.13}$$

[vii]若交点不存在，由定理 2.11 可知，(2.3.11) 存在共轭复根，满足 $|\lambda_i(\omega)| \equiv |\omega - 1|$，不影响后面的讨论。

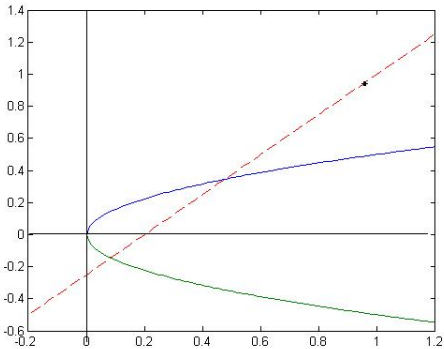

图 2.3.1: 直线与抛物线的交点

显然，当 $|\mu|$ 从小往大变化时，相应的 ω 随之增加。此时，抛物线的开口变宽，相应的切点位置会随之右移，对应的切线以 $(1,1)$ 为中心逆时针旋转。在这个过程中，切线同原有的窄口抛物线是不相交的。因此，*SOR* 方法迭代矩阵的谱半径对应这个过程的最后切点位置。

换言之，最佳松弛因子对应 $\mu = \rho(\mathbb{B})$ 的切线参数，即

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

相应的迭代矩阵谱半径为

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

事实上，ω_{opt} 对应谱半径函数 $f(\omega) \equiv \varrho(\mathbb{T}_\omega)$ 的不可微点；参见下面左侧的插图。关于谱半径函数的具体表达式，请参见教科书。

☆ **说明 2.9.** 注意到谱半径函数在不可微点的左导数为无穷大，实际计算时通常会偏大选取松弛因子。

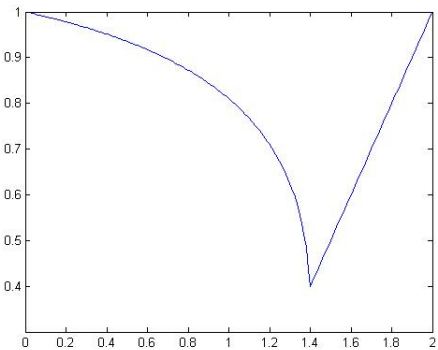

图 2.3.2: SOR 方法的谱半径函数 $\varrho(\mathbb{T}_\omega)$

✍ **论题 2.7.** 最佳松弛因子的选取方法: 先取一个偏大的 $\omega \in (0,2)$，然后执行几步 *SOR* 迭代过程。由幂法可知，在适当的条件下，迭代矩阵 $\rho(\mathbb{T}_w)$ 的绝对值最大特征值可近似表示为

$$\rho(\mathbb{T}_w) \approx \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_\infty}{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty}.$$

然后，我们利用 (2.3.11) 确定 *Jacobi* 迭代矩阵 $\mathbb{B}$ 的谱半径，即

$$\rho(\mathbb{B}) = \frac{\rho(\mathbb{T}_w) + \omega - 1}{\omega[\rho(\mathbb{T}_w)]^{1/2}}.$$

由这个较为精确的估计 $\rho(\mathbb{B})$，我们可以给出最佳松弛因子的近似。上述过程重复多次，我们可以逐步地改进 ω_{opt} 的精确度。

✍ **论题 2.8.** 对应最佳松弛因子 ω_{opt}，*SOR* 方法的收敛速度可获得本质性的提升。

2.3.3 收敛速度的比较

教科书给出了一个抽象的普适结果，下面用具体实例进行直接展现。对于线性方程组 (6.0.3)，其系数矩阵是块三对角矩阵，利用 Kronecker 积[viii]可以表示为

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

其中 $\mathbb{T}_n$ 是三对角矩阵，其特征值和特征向量可以精确地计算出来，即

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

其中整数 κ 遍历 1 到 n。利用 Kronecker 积的运算规则，可知 $\mathbb{A}_n$ 的特征值为

$$\lambda_{pq} = \lambda_p + \lambda_q,$$

其中整数 p 和 q 均遍历 1 到 n。注意到矩阵结构，由前面的讨论可知

$$\mu_{pq} = \frac{1}{4}(\lambda_p + \lambda_q - 4).$$

因此，三种迭代方法的渐近收敛速度分别为

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

其中 $h = 1/(n+1)$。这个结果由 Franke 最早给出，相应的推广结果由 Young 给出。

[viii]设 $\mathbb{A} = (a_{ij})$ 是 m 阶方阵，$\mathbb{B} = (b_{ij})$ 是 n 阶方阵，则 $\mathbb{A} \otimes \mathbb{B} = (a_{ij}\mathbb{B})$ 是一个 mn 阶方阵。

✯ **说明 2.10.** 类似于逐次超松弛方法，我们还可以构造出块 *SOR* 方法和对称 *SOR* 方法。详略。

2.4 迭代加速方法

SOR 方法的研究结果表明：充分利用已有的计算信息，可以有效提高收敛速度。为此，本节介绍基础迭代方法

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

的两种常用迭代加速技术，特别是著名的半迭代方法。

2.4.1 外推方法

外推方法是 SOR 方法的直接推广。设 γ 是给定的权重，加权平均相邻的两个数值解，定义

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

❀ **思考 2.5.** 设迭代矩阵 $\mathbb{G}$ 是实对称的，请找到最优参数 γ，使得迭代矩阵 $\gamma\mathbb{G} + (1-\gamma)\mathbb{I}$ 具有尽可能小的谱半径，使外推方法的收敛速度达到最快。

2.4.2 半迭代方法

半迭代方法可以看作外推思想的极致推广。换言之，我们想充分利用已知的所有计算结果，通过适当的加权平均处理，期待

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

比 $\boldsymbol{x}_k$ 更加接近精确解，其中 $\{\boldsymbol{x}_k\}_{k=0}^{\infty}$ 是由基础迭代算法 (2.4.16) 给出的迭代序列。

若 $\boldsymbol{x}_k \equiv \boldsymbol{x}_\star$ 是精确解，则 $\boldsymbol{y}_m \equiv \boldsymbol{x}_\star$ 应当也是精确解。因此，参数组 $\{\alpha_{m,k}\}_{k=0}^{m}$ 应满足相容性条件

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

记 $\boldsymbol{\eta}_m = \boldsymbol{y}_m - \boldsymbol{x}_\star$ 是半迭代方法 (2.4.17) 的第 m 步误差。简单计算可知，其误差方程为

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

其中 $\boldsymbol{e}_0 = \boldsymbol{\eta}_0 = \boldsymbol{x}_0 - \boldsymbol{x}_\star$ 为初始误差。这里的 $\mathbb{P}_m(\lambda)$ 是 m 次多项式，相应的系数由参数组 $\{\alpha_{m,k}\}_{k=0}^{m}$ 给出。至此，一种新的构造思想诞生了，迭代方法的研究思路也从“单项式算法”拓展到“多项式算法”。

基于误差方程 (2.4.19)，自然希望 $\mathbb{P}_m(\mathbb{G})$ 的谱半径远远小于 $\mathbb{G}$ 的谱半径。为此，研究中心是寻找 $\mathbb{P}_m(\mathbb{G})$ 的谱半径最小值，给出线性组合的最佳权重。

变系数 Richardson 方法

事实上，某些传统算法已经隐含地实现了半迭代方法的基本思想。一个典型实例是 Richardson（1910）迭代方法

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

其中 τ_k 是可以变化的迭代参数。显而易见，R 方法可以理解为残量松弛方法，比 J 方法更加简单。

✍ **论题 2.9.** *若 $\tau_k \equiv \tau$ 保持不变，则称 (2.4.20) 为定常的。变系数 R 方法可视为定常 R 方法的半迭代加速。*

为理解变参数带来的加速效果，不妨假设 $\mathbb{A}$ 是实对称正定矩阵。注意到对称矩阵的谱范数就是其谱半径，有

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

给定迭代次数 m，能否找到某个参数组 $\{\tau_k^\star\}_{k=1}^m$，使得变系数 R 方法的收敛速度最快?

为确定右端上界的最小值，不妨考虑 Cheybeshev 极大极小问题

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

其中 $\lambda_{\max} > 0$ 和 $\lambda_{\min} > 0$ 分别是 $\mathbb{A}$ 的最大特征值和最小特征值。由 Cheybeshev 理论可知，最佳多项式是

$$P_m^\star(\lambda) = \frac{T_m\left(\dfrac{\lambda_{\max} + \lambda_{\min} - 2\lambda}{\lambda_{\max} - \lambda_{\min}}\right)}{T_m\left(\dfrac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}\right)} \tag{2.4.21}$$

其中 $T_m(z)$ 是标准的 Cheybeshev 多项式[ix]。比较零点的位置，可知最佳参数组 $\{\tau_k^\star\}_{k=1}^m$ 就是 $P_m^\star(\lambda)$ 的零点，即

[ix] Cheybeshev 多项式

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

具有非常完美的性质，例如恒等式

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

证明留作练习。

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

利用 Cheybeshev 多项式的性质可知，变系数 R 方法满足

$$\frac{\|\boldsymbol{e}_m\|_2}{\|\boldsymbol{e}_0\|_2} \le 2\left(\frac{\sqrt{\kappa(\mathbb{A})}-1}{\sqrt{\kappa(\mathbb{A})}+1}\right)^m, \tag{2.4.22}$$

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

✍ **论题 2.10.** *在定常 R 方法中，最佳迭代参数 τ 是什么？相应的收敛速度满足怎样的估计？*

上述结果表明：要达到用户的精度要求，定常 R 方法所需的最小迭代步数同 $\kappa(\mathbb{A})$ 成正比；而变系数 R 方法同 $\sqrt{\kappa(\mathbb{A})}$ 成正比。换言之，变系数策略显著提升了 R 方法的收敛速度。

☆ **说明 2.11.** *请注意：最优参数组的计算同迭代次数 m 的事先估算相关。在执行最佳的变系数 R 方法之前，我们需要事先确定 m 的值，然后才能确定最优参数组。但是，若 m 很大，最佳参数组的数值计算将因舍入误差而产生严重偏差，进而导致实际的迭代收敛速度受到严重的破坏。为克服这个困难，常用的解决方法是采用较小的 m，并引入循环算法（或重新启动）策略。*

Cheybeshev 加速方法

再次考虑半迭代方法 (2.4.17)，讨论最佳参数组 $\{\alpha_{k,\ell}\}_{\ell=0}^{k}$ 的设置及其效果；同时，半迭代方法还需给出相应的实现途径，解决数据存储的困境。此时，正交的 Cheybeshev 多项式扮演着重要的角色。

✍ **论题 2.11.** 设基础迭代方法 (2.4.16) 的迭代矩阵 $\mathbb{G}$ 是实对称的。理论上最佳的参数组是什么？

此时，$\mathbb{G}$ 具有完备的特征向量系 $\{\boldsymbol{\xi}_i\}_{k=1}^n$，相应的特征值 $\{\lambda_i\}_{i=1}^n$ 均为实数。初始误差可以表达为

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

其中 β_i 是已知常数。由误差方程 (2.4.19) 可知，半迭代方法 (2.4.17) 的迭代误差满足

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

能否找到一组最佳参数 $\{\alpha_{k,\ell}\}_{\ell=0}^k$，使得迭代误差（在谱范数度量下）趋于零的速度达到最快?

类似于前面的讨论，上述目标转化为一个 Chebyshev 极大极小问题，即

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

其中 $\lambda_{\max}$ 和 $\lambda_{\min}$ 是 $\mathbb{G}$ 的最大特征值和最小特征值[x]，$\mathbb{P}_k^\sharp$ 是系数和为一的 k 次多项式全体。相应答案就是归一化[xi]的 Chebyshev 多项式

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

其中 $T_k(z)$ 为标准的 Chebyshev 多项式，

$$\ell(\lambda) = \frac{2\lambda - \lambda_{\max} - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}. \tag{2.4.23}$$

[x]请注意：不是系数矩阵 $\mathbb{A}$ 的最大特征值和最小特征值。

[xi]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。

半迭代方法 (2.4.17) 的最佳参数组 $\{\alpha_{k,\ell}\}_{\ell=0}^{k}$，就是最佳多项式 $Q_k^\star(\lambda)$ 的相应系数。

✯ **说明 2.12.** 上述讨论也可以推广到基础迭代矩阵 $\mathbb{G}$ 具有复特征值的情形，其参数组的设定与复特征值所属的椭圆区域有关。具体内容超出本课程的要求，略。

❀ **思考 2.6.** 请尝试度量一下这个算法的平均收敛速度，并回答算法收敛速度是否具有本质上的提升?

✍ **论题 2.12.** 半迭代方法 (2.4.17) 不是一个可以真正应用的数值方法，原因有二。其一是数据存储的困境，所有的历史数据和历史参数不可能完全保留；其二是舍入误差的影响，参数组信息的计算不可能完全的准确。为此，我们必须给出一个等价的且实用的计算公式。

回顾 Chebyshev 多项式的正交性，特别是三项递推关系式

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

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，可知半迭代方法 (2.4.17) 满足变系数二步迭代公式

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

其中固定参数是

$$\nu = \frac{2}{2-\lambda_{\max}-\lambda_{\min}}, \tag{2.4.25}$$

变化参数是

$$\rho_{k+1} = \frac{2\xi T_k(\xi)}{T_{k+1}(\xi)}, \quad \xi = \ell(1) = \frac{2-\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}-\lambda_{\min}}. \tag{2.4.26}$$

换言之，半迭代方法可以不必存储大量的历史数据。

算法启动如下：任取初值 $\boldsymbol{x}_0$，由基础迭代方法给出 $\boldsymbol{x}_1 = \mathbb{G}\boldsymbol{x}_0 + \boldsymbol{g}$。利用三项递推关系式 (2.4.24)，可得变化参数 ρ_k 的计算公式

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

换言之，我们也不用存储 Chebyshev 多项式信息。

❀ **思考 2.7.** *假设线性方程组的系数矩阵 $\mathbb{A}$ 具有性质 $\boldsymbol{A}$，或者直接假设系数矩阵就是由 (2.3.11) 右端所定义的特殊矩阵。请考虑 J 方法的半迭代加速，并考察 ρ_k 的极限到底是什么？这个极限同 SOR 方法的最佳松弛因子有何关联？*

在半迭代方法中，最佳参数组同基础迭代矩阵（或者间接地同系数矩阵）的谱信息有关，特别是最大特征值和最小特征值。但是，特征值问题比线性方程组更难求解，通常难以准确（或相对准确）给出相应的最大特征值和最小特征值。在某种程度上，这是一个致命的缺陷，严重限制半迭代方法的应用范围。因此，我们期待数值方法能够摆脱这个束缚。

2.5 共轭斜量法

共轭斜量（CG = Conjurate Gradient）法是对称正定线性方程组的首选数值方法，由 Hestenes 和 Stiefel 在 19050 年前后提出。其核心思想是利用合适的优化方法，快速求解等价的目标函数极值点。它无需事先估计系数矩阵的特征值，具有无参数和快速收敛等优势。尽管是一种直接算法，它常常被归属到迭代算法。

2.5.1 等价的极值问题

论题 2.13. 设 $\mathbb{A}$ 是实对称正定矩阵。在共轭斜量法的多种引进方式之中，较为直观的方式是将线性方程组 $\mathbb{A}\boldsymbol{x}=\boldsymbol{b}$ 转化为等价的二次函数（椭圆抛物面）最优化问题：

$$\boxed{\boldsymbol{x}_\star = \arg\min_{\forall \boldsymbol{x}} f(\boldsymbol{x}),}$$

其中 $f(\boldsymbol{x})=\frac{1}{2}\boldsymbol{x}^\top\mathbb{A}\boldsymbol{x}-\boldsymbol{b}^\top\boldsymbol{x}$ 称为整个离散系统的总能量。

定理 2.13. 两种表述是等价的，即 $\boldsymbol{x}_\star$ 就是线性方程组的精确解。

目标函数 $f(\boldsymbol{x})$ 是二次函数，相应的优化问题具有快速算法。通常，其核心技术基于一维搜索策略，即从当前位置 $\boldsymbol{x}_k$ 出发，沿着某个搜索方向 $\boldsymbol{p}_k$，找到后续的最优位置 $\boldsymbol{x}_{k+1}$，使得

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

简单计算可知，其答案是

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\alpha_k\boldsymbol{p}_k \qquad \alpha_k=-\frac{\boldsymbol{r}_k^\top\boldsymbol{p}_k}{\boldsymbol{p}_k^\top\mathbb{A}\boldsymbol{p}_k}, \tag{2.5.27}$$

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

不妨假设所有的搜索方向是已知的，且前 k 个搜索方向可以依次张成一个 k 维子空间

$$\mathcal{L}_k=\mathrm{span}\{\boldsymbol{p}_0,\boldsymbol{p}_1,\ldots,\boldsymbol{p}_{k-1}\}. \tag{2.5.28}$$

相应的一维搜索算法定义如下：从初始搜索位置 $\boldsymbol{x}_0$ 出发，沿着相应的搜索方向，按照公式 (2.5.27) 依次执行一维搜索策略。换言之，上述算法给出的搜索位置满足

$$\boldsymbol{x}_k\in\pi_k\equiv\boldsymbol{x}_0+\mathcal{L}_k. \tag{2.5.29}$$

定义 2.3. 称当前位置 $\boldsymbol{x}_k$ 关于搜索空间 $\mathcal{L}_k$ 是最优的，若

$$\boldsymbol{x}_k = \arg\min_{x\in\pi_k} f(\boldsymbol{x}). \tag{2.5.30}$$

引理 2.1. (2.5.30) 成立的充要条件是当前残量满足 $\boldsymbol{r}_k \perp \mathcal{L}_k$，即

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

在原有假设的基础上，继续考虑如下的理想状态：**在算法执行过程中，搜索位置关于搜索空间的最优性质一直保持，且搜索空间 $\mathcal{L}_k$ 的维数不断增加**。由引理 2.1 可知，若搜索空间逐步扩张到整个空间，则相应的残量变为零。换言之，搜索位置可以在有限步数内到达 $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ 的精确解。

2.5.2 共轭斜量方法的框架

上述假设给出的理想状态能够实现吗？考虑搜索空间 $\mathcal{L}_k$ 的扩张过程，观察引理 2.1 的充要条件是否成立。基于一维搜索策略，每个后续搜索位置关于当前搜索方向都是最优的。但是，它关于前面的搜索方向也是最优吗？

从如下的简单算法出发：在当前位置 $\boldsymbol{x}_k$ 处，非常自然的搜索方向是（局部）最速下降方向，即

$$\boldsymbol{p}_k = -\mathrm{grad} f(\boldsymbol{x}_k) = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}_k = -\boldsymbol{r}_k. \tag{2.5.31}$$

相应的一维搜索过程称为最速下降法。

最速下降法是收敛的，但是其数值表现常常极其糟糕。换言之，收敛速度变得越来越慢，收缩位置出现极差的“盘旋收敛”现象。出现上述现象的一个主要原因是，最速下降法没有一直保持搜索位置关于搜索空间的最优性质。

✍ **论题 2.14.** *在最速下降法连续执行两步之后，有*

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

显然，它关于当前搜索方向 $\boldsymbol{p}_{k+1} = -\boldsymbol{r}_{k+1}$ 是最优的。但是，简单计算可知

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

换言之，$\boldsymbol{x}_{k+2}$ 关于前一个搜索方向 $\boldsymbol{p}_k = -\boldsymbol{r}_k$ 不是最优的. 因此，最速下降法无法实现引理 *2.1* 的充要条件。

❀ **思考 2.8.** 最速下降法是收敛的。请读者给出相应的（欧氏范数）收敛估计。

怎样才能一直保持搜索位置关于搜索空间的最优性质呢? 。这个问题不易回答，不妨退而求其次，考虑一个相对简单的问题:

> 如何保证当前位置关于最近的两个搜索方向都是最优的? 换言之，当前位置关于局部的二维搜索空间是最优的。

设 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{q}$ 是从 $\boldsymbol{x}_k$ 出发，沿搜索方向 $\boldsymbol{q}$ 到达的最优位置。要它关于前一个搜索方向 $\boldsymbol{p}$ 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，相邻的两个搜索方向 $\boldsymbol{p}$ 和 $\boldsymbol{q}$ 是 $\mathbb{A}$-共轭的。这个属性诱导出如下的重要概念。

定义 2.4. *若对任意两个不同的指标 i 和 j，均有 $\boldsymbol{p}_i^{\top}\mathbb{A}\boldsymbol{p}_j=0$，则称 $\{\boldsymbol{p}_\kappa\}_{\kappa=0}^{m}$ 是共轭向量系。*

若搜索方向构成共轭向量系，则相应的一维搜索算法称为共轭斜量法。利用数学归纳法，可证

定理 2.14. *在共轭斜量法中，搜索位置 $\boldsymbol{x}_{k+1}$ 关于搜索空间 $\mathcal{L}_k$ 都是最优的。*

在共轭向量系中，搜索方向是线性无关的。因此，由定理 2.14 可知：若四则运算都是无误差的，则共轭斜量法至多 n 步即可给出相应的精确解。因此说，共轭斜量法是直接算法。

2.5.3 共轭斜量系的构造过程

论题 2.15. *共轭斜量法的应用前提是共轭斜量系的构造。事实上，共轭斜量方向可以局部递推构造。*

彼此垂直的 $\boldsymbol{r}_{k+1}$ 和 $\boldsymbol{p}_k$ 局部张成一个二维平面，它包含一个 $\mathbb{A}$-共轭于 $\boldsymbol{p}_k$ 的搜索方向 $\boldsymbol{p}_{k+1}$。基于这个思路，定义算法如下：

令 $\boldsymbol{p}_0=-\boldsymbol{r}_0=\boldsymbol{b}-\mathbb{A}\boldsymbol{x}_0$，依次执行

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

$$\boldsymbol{r}_{k+1}=\boldsymbol{r}_k+\alpha_k\mathbb{A}\boldsymbol{p}_k,$$

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

上述过程包含大量内积运算，具有内在的 BLAS-2 机制和并行特征。

论题 2.16. 在上述算法中，搜索方向是局部 $\mathbb{A}$-共轭的。下面证明：在搜索位置到达精确解之前，搜索方向构成共轭斜量系，而且算法执行过程中存在三套扮演不同角色的向量组，即

> 只要 $\boldsymbol{r}_k \neq 0$，三套向量组是彼此等价的：
>
> $$\begin{aligned} span\{\boldsymbol{r}_0, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_k\} &= span\{\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k\} \\ &= span\{\boldsymbol{r}_0, \mathbb{A}\boldsymbol{r}_0, \ldots, \mathbb{A}^k\boldsymbol{r}_0\}. \end{aligned}$$
>
> 它们分别称为残量空间、搜索空间和 *Krylov* 子空间。

因此说，上述算法是共轭斜量法。

论题 2.17. 在共轭斜量法中，所有的残量方向满足关系

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

换言之，残量仅仅是椭圆抛物面的法方向，它没有像搜索方向一样，更快速地指向椭圆抛物面的顶点。

论题 2.18. 在共轭斜量法中，搜索方向与残量方向满足关系

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

利用这个性质，参数计算可以简化为

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

相邻两步的内积运算出现重复，算法的计算复杂度有所改善。

❀ **思考 2.9.** 若忽视 α_k 和 β_k 的计算过程，将它们看作事先设定的迭代参数，则共轭斜量法可以视为二阶非定常迭代方法。请写出相应的计算公式。

2.5.4 收敛性分析

虽然是直接算法，共轭斜量法具有明显的迭代计算流程。下面讨论共轭斜量法的收敛性质。

定理 2.15. 迭代误差 l_2 模是单调下降的。

证明：以有限步到达精确解为起点，直接计算即可证明。 □

简单计算，可知

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

称其为第 k 步的误差能量。

✎ **论题 2.19.** 由定理 *2.14* 可知，第 $k+1$ 步的误差能量对应某个泛函极小，即

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

其中 $\mathbb{P}_k$ 是 k 次多项式全体。

它蕴含两个重要的推论。

定理 2.16. 若 $\mathbb{A}$ 仅仅具有 m 个相异的特征值，则 CG 方法至多 m 步即可得到线性方程组的精确解。

定理 2.17. *CG 方法的收敛速度同谱条件数 $\kappa(\mathbb{A})$ 相关，即*

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

✯ **说明 2.13.** *定理 2.17 表明：CG 方法和（最优参数的）半迭代方法具有相同的收敛速度。但是，前者无需知道 $\mathbb{A}$ 的特征值信息，实际应用更加便捷。*

✯ **说明 2.14.** *当系数矩阵 $\mathbb{A}$ 严重病态时，CG 方法的收敛速度也会变慢，舍入误差的影响也会更加强烈。即便如此，同其他方法相比，CG 方法给出的数值结果更加可信。*

综合上述推导过程，不难发现如下结论：CG 方法的收敛速度不仅同矩阵条件数相关，还同特征值的聚集状态相关。通常，CG 方法的迭代次数远远少于矩阵阶数，甚至呈现出“超线性收敛”现象。

❀ **思考 2.10.** *假设系数矩阵的特征值聚集在两个区间，不妨记为 $[a_1, b_1] \cup [a_2, b_2]$。请给出相应的收敛速度估计。*

✯ **说明 2.15.** *到目前为止，以三种不同的向量组为主体，CG 方法已经由对称正定问题推广到任意的非对称或非正定问题，形成更一般的 Galerkin 方法或者 Krylov 子空间投影方法，例如 GMRES 方法，双稳定化的 CG 方法，或者平方 CG 方法等等。这些方法都已经被收录在 Matlab 中。*

2.5.5 预处理共轭斜量方法

预处理技术是数值代数的基本技术。它用于改善线性方程组的条件数（或者特征值的分布属性），从而提高算法的计算效率。

论题 2.20. 下面以共轭斜量方法为例，阐述预处理技术的基本思想和实现过程。设预处理矩阵是 $\mathbb{Q}=\mathbb{C}\mathbb{C}^{\top}$, 考虑同解方程组

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

的 *CG* 算法。若同解方程组的条件数得到下降，则 *CG* 算法的收敛速度将会得到改善。因篇幅有限，理论细节不再赘述。

上述算法称为预处理共轭斜量（*PCG*）方法。在 *Matlab* 中，对应命令是 *pcg()*。相应的计算流程如下：

取 $\boldsymbol{r}_0=\mathbb{A}\boldsymbol{x}_0-\boldsymbol{b}$，令 $\boldsymbol{z}_0=\mathbb{Q}^{-1}\boldsymbol{r}_0, \boldsymbol{p}_0=-\boldsymbol{z}_0$，依次执行

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

换言之，算法的基本流程没有改变，只需每步额外求解一个预处理方程 $\mathbb{Q}\boldsymbol{z}=\boldsymbol{g}$。出于计算效率的考量，相应的求解应当简单易行。

矩阵分裂的主体部分可以作为预处理矩阵，例如对称超松弛方法可以给出 PCG 方法的预处理矩阵

$$\mathbb{Q}=(\mathbb{D}+\omega\mathbb{L})\mathbb{D}^{-1}(\mathbb{D}+\omega\mathbb{L})^{\top}. \tag{2.5.32}$$

不完全 LU 分解技术、或者逆矩阵的多项式近似方法等，也可以给出相应的预处理矩阵。

第 3 章
线性最小二乘问题

在数据分析和线性回归等问题中，未知数和约束条件的个数常常是不匹配的，相应的线性方程组

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

不再具有传统意义的解向量[i]。此时，线性方程组和解向量的含义需要适当拓展。

定义 3.1. 线性方程组 (3.0.1) 称为线性最小二乘问题，即其实质是一个极小值问题[ii]

$$\tilde{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\mathbb{A}\boldsymbol{x} - \boldsymbol{b}\|_2.$$

此时，$\tilde{\boldsymbol{x}}$ 称为问题的最小二乘解。

3.1 基本理论

线性最小二乘问题具有特殊性。无论是数学概念还是数值计算，它都具有足够的挑战性。

3.1.1 最小二乘解和极小最小二乘解

定理 3.1. 最小二乘解是必然存在的。

[i] 若无解，称其是矛盾的；若无穷多解，则称其是不定的。

[ii] 其他度量方式也是可以的，例如最大模、1-模以及加权范数等等。

证明：证明的方法有很多，比如用代数的方法证明。

若 $\boldsymbol{b} \in \mathrm{R}(\mathbb{A})$，由线性方程组的基本理论可知，存在解使得所有等式成立。显然，它就是最小二乘解。若 $\boldsymbol{b} \notin \mathrm{R}(\mathbb{A})$，考虑 $\boldsymbol{b}$ 在 $\mathrm{R}(\mathbb{A})$ 及其正交补空间 $[\mathrm{R}(\mathbb{A})]^{\perp}$ 的直和分解

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

此时，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.** 直接利用数学分析的方法，也可以证明最小二乘解必然存在。

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，即对应残量 $\boldsymbol{r} = \mathbb{A}\boldsymbol{x} - \boldsymbol{b}$ 同 $\mathbb{A}$ 的每个列向量都是正交的。

证明：利用极值点处导数为零的性质，简单计算即可证明。 □

定理 3.3. 若矩阵 $\mathbb{A}_{m\times n}$ 是列满秩的，即 $rank(\mathbb{A}) = n$，则最小二乘问题 (3.0.1) 具有唯一的最小二乘解

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

否则，(3.0.1) 的最小二乘解有无穷多个。

最小二乘解的公式表达具有多种形式。下面利用系数矩阵的满秩分解，给出一个重要的公式表示。

定理 3.4. 若 $r = rank(\mathbb{A}_{m\times n}) > 0$，则有满秩分解

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

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

定理 3.5. 已知满秩分解 $\mathbb{A}=\mathbb{G}\mathbb{F}$，则线性方程组 (3.0.1) 的最小二乘解可以表示为

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

事实上，这个表示具有重要的意义，称为**极小最小二乘解**，是最小二乘解集中向量长度最小的唯一元素。

3.1.2 广义逆矩阵

若线性方程组 (3.0.1) 的系数矩阵 $\mathbb{A}$ 是非奇异的方阵，则极小最小二乘解（残量为零）可以表示为

$$\boldsymbol{x}_{\mathrm{LS}}=\mathbb{A}^{-1}\boldsymbol{b},$$

其中 $\mathbb{A}^{-1}$ 是 $\mathbb{A}$ 的逆矩阵。上述表示方法能否推广到任意问题呢?

早在 1920 年，E. H. Moore 就提出了广义逆矩阵概念。但是，由于用途不明，这个概念很少被人问津。直到 1955 年，R. Penrose 明确给出广义逆矩阵的等价定义，相应理论才真正发展起来，并进入崭新的发展阶段。

定义 3.2. 给定矩阵 $\mathbb{A}\in\mathbb{R}^{m\times n}$。若 $\mathbb{X}\in\mathbb{R}^{n\times m}$ 满足[iii]

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

则称 $\mathbb{X}$ 是 $\mathbb{A}$ 的（*Moore-Penrose*）广义逆，记为 $\mathbb{X}=\mathbb{A}^{\dagger}$.

定理 3.6. 广义逆矩阵存在且唯一。

[iii]概念可以简单推广到复矩阵，即转置替换为共轭转置即可。

证明：存在性可由满秩分解 $\mathbb{A}=\mathbb{F}\mathbb{G}$ 给出。简单验证，

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

满足广义逆矩阵的四条性质。唯一性可由四条性质直接保证。 □

定理 3.7. *极小最小二乘解可表示为* $\boldsymbol{x}_{LS}=\mathbb{A}^{\dagger}\boldsymbol{b}$.

证明：只需验证其具有最短长度。 □

通常，广义逆矩阵的计算较为繁琐。对于某些特殊结构，广义逆矩阵的计算可以简化。例如，常用的结论有

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

其中 $\mathbb{X}$ 是可逆方阵。

✍ **论题 3.1.** *若矩阵是非奇异的方阵，则广义逆矩阵就是古典逆矩阵。尽管前者是后者的推广，两个概念的性质具有明显的区别。*

下面以奇异方阵为例，说明上述论点。此时，适用于逆矩阵的运算规则和理论性质有可能不再成立，例如：

1. $(\mathbb{A}\mathbb{B})^{\dagger}\neq\mathbb{B}^{\dagger}\mathbb{A}^{\dagger}$, $\mathbb{A}\mathbb{A}^{\dagger}\neq\mathbb{A}^{\dagger}\mathbb{A}$, $(\mathbb{A}^{k})^{\dagger}\neq(\mathbb{A}^{\dagger})^{k}$;
2. $\mathbb{A}$ 与 $\mathbb{A}^{\dagger}$ 的非零特征值不是互为倒数。
3. 广义逆矩阵可能不再连续依赖于矩阵元素。

特别地，第三条性质暗含指出了最小二乘问题的数值计算困难。

❀ **思考 3.2.** *请给出上述三条性质的相应实例。*

在广义逆矩阵的扰动理论中，矩阵秩保持恒定是极其重要的。若矩阵秩发生变化，则微小变化可能引起广义逆矩阵的明显变化。但是，若

矩阵秩保持不变，则广义逆矩阵也是连续变化的，其关于扰动的敏感程度同矩阵条件数相关。

❀ **思考 3.3.** 为直观理解上述结论，不妨观察两个矩阵

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

的广义逆矩阵当 $\varepsilon \to 0$ 时的具体表现。

3.1.3 数值算法综述

通常，最小二乘问题有直接法和迭代法两种计算策略[iv]。直接法包含正规化方法和直交化方法。

若未做声明，本章均假设系数矩阵 $\mathbb{A}$ 是列满秩的，即

$$r = \operatorname{rank}(\mathbb{A}) = n.$$

对于列亏秩的情形，我们仅仅给出简要说明。

☆ **说明 3.1.** 如果最小二乘问题的系数矩阵是（列）亏秩的，则最小二乘解的数值计算将变得更加困难。由于列向量是否线性相关很难给予准确的数值判定[v]，某些算法将因此而无法顺利进行到底。

即便算法能够顺利执行到底，最终的计算结果也有可能具有不同的含义，例如它仅仅是最小二乘解而已，不是极小最小二乘解。

同时，由于舍入误差的数值影响，不同算法的计算结果也可能具有较大的差异。

[iv] 其它方法包括最优化方法等等，略。

[v] 矩阵秩关于矩阵变化是不连续的。因此，舍入误差将会严重影响矩阵秩的数值确定，以及最大线性无关组的数值判定。

此时，最小二乘解就是唯一的极小最小二乘解。它满足法方程组 $\mathbb{A}^\top\mathbb{A}\boldsymbol{x} = \mathbb{A}^\top\boldsymbol{b}$，或者相应的扩展方程组

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

其中 $\boldsymbol{r}$ 是残量，$\mathbb{A}_1$ 是系数矩阵 $\mathbb{A}$ 的前 n 行向量形成的可逆方阵。相应的数据源于下面的矩阵分块

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

由于系数矩阵都是非奇异的，数值求解可以借用前面两章介绍的各种直接法或迭代法。

☆ **说明 3.2.** 事实上，扩展方程组和法方程组具有相同的数值困难。两者相比，扩展方程组回避了 $\mathbb{A}^\top\mathbb{A}$ 的直接计算，并同时计算出相应的残量，减少了舍入误差的影响。

正规化方法堪称是最容易的计算方法。对于良态问题，它常常是首选的。但是，对于病态问题，正规化方法遇到巨大的数值困难。主要原因有二个。

1. 法方程组的条件数 $\kappa_2(\mathbb{A}^\top\mathbb{A})$ 是原问题条件数

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

的平方[vi]，病态程度加剧，舍入误差的影响更加严重。

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

2. 浮点运算可能出现上溢出（或下溢出），法方程组的系数矩阵因此而出现亏秩。例如，当 $\varepsilon < \sqrt{\vartheta}$ 时，有

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

其中 ϑ 是机器精度。

✯ **说明 3.3.** *共轭斜量方法也适用于最小二乘问题的数值计算。即使系数矩阵是列亏秩的，它也可以给出某个最小二乘解。但是，答案不一定是极小最小二乘解。*

为克服舍入误差带来的严重困难，最小二乘问题的数值求解通常采用更健壮的直交化方法。它等同于系数矩阵（或增广矩阵）的直交分解过程，具体实现过程将在后续两节内容中给出。

3.2 直交分解技术

本节集中介绍一些常见的直角分解技术。它主要包含两类方法。第一类是 Gram-Schmidt 正交化过程，将系数矩阵的线性无关列向量组转化为列直交向量组；第二类是正交矩阵变换方法，利用 Householder 镜像矩阵，或者 Givens 平面旋转矩阵，将系数矩阵逐步转化为上梯形矩阵。用矩阵语言来描述，第一类方法是一系列三角形（或梯形）矩阵的右乘过程，而第二类方法是一系列直交阵的左乘过程。

3.2.1 Gram-Schmidt 直交化方法

Gram-Schmidt（GS）直交化方法是基本的代数工具，可在高等代数的教材中找到。此处不再赘述。

✎ **论题 3.2. 上梯形化的直交分解**。利用 *GS* 直交化方法，可以建立矩阵 $\mathbb{A}$ 的直交分解：

> *(A):* $\mathbb{A}_{m\times n}\mathbb{P}_{n\times n} = \mathbb{Q}_{m\times r}\mathbb{U}_{r\times n}$, 其中 $r = rank(\mathbb{A})$，$\mathbb{P}$ 为 n 阶置换, $\mathbb{Q}$ 是列直交阵，$\mathbb{U}$ 是对角线元素为正的上梯形矩阵。

有时，这种表述也称为矩阵的 QR 分解。

☆ **说明 3.4.** 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。此时，需额外开辟数据空间，存储上梯形矩阵 $\mathbb{U}_{r\times n}$ 的信息。

事实上，GS 直交化方法具有理论上完全等价的两种执行次序：

1. 传统（CGS）方法是利用当前列向量与历史列向量组的正交性，逐列计算矩阵 $\mathbb{U}$ 的信息；

2. 而修正（MGS）方法是利用当前列向量与未来列向量组的正交性，逐行计算矩阵 $\mathbb{U}$ 的信息。我们不断剔除待处理列向量在历史列向量组空间的投影，可视为计算的问题规模越来越小，舍入误差的积累影响较弱。

但是，它们的舍入误差表现具有明显的不同。数学上等价的两个数值方法可能具有截然不同的稳定性表现，这是计算数学所特有的现象。在数值方法研究中，我们应当引起足够的重视。

☆ **说明 3.5.** 同 *CGS* 方法相比，*MGS* 方法具有更好的数值健壮性，可适用于病态的最小二乘问题求解。设 $\mathbb{A}_{m\times n}$ 是列满秩矩阵，数值稳定性表现如下：

1. 基于向后误差分析理论，*MGS* 方法可等价描述为一个扰动矩阵的精确 *QR* 分解过程，即

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

其中 $\delta\mathbb{A}_{MGS}$ 是扰动矩阵。舍入误差分析理论表明：

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

其中 ϑ 是机器精度，$c_{m,n}$ 是绝对常数。

2. 类似地，*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。关于扰动矩阵的舍入误差分析结果依旧成立，即

$$\|\delta\mathbb{A}_{CGS}\|_2\le c_{m,n}\vartheta\|\mathbb{A}\|_2$$

但是，直交阵 $\mathbb{Q}_{CGS}$ 在直交方面的表现不再满足 *MGS* 方法的第二条性质。

为说明上述结论，我们给出一个数值算例。考虑 25×15 阶范德蒙矩阵

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

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

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

换言之，它们给出的 QR 乘积均很好地近似原始矩阵 $\mathbb{A}$；因此，它们是向后稳定的算法。换言之，即使数值得到的 $\mathbb{Q}_{\text{num}}$ 和 $\mathbb{R}_{\text{num}}$ 同真实结果的误差可能很大，但它们的乘积 $\mathbb{Q}_{\text{num}}\mathbb{R}_{\text{num}}$ 却神奇地非常准确地接近 $\mathbb{A}$。这是直交分解的一个优势之一。但是，计算结果表明

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

换言之，两种方法在列向量的直交性方面有明显的差异。

☆ **说明 3.6.** 对于极其病态（条件数非常大）的矩阵，*CGS* 方法和 *MGS* 方法给出的对角线元素也存在明显的数值差异。设目标矩阵 $\mathbb{C}$ 是由一个元素快速变化的对角阵 $diag\{2^{-k}\}_{k=1}^{80}$ 相继左乘和右乘两个任意的直交阵而得到的。我们将两种方法给出的对角线元素绘制在图 *3.2.1* 中，其中方框表示 *CGS* 方法，圆圈表示 *MGS* 方法，星号表示真实的对角元素。因此，在实际的数值计算中，所谓的 *GS* 直交化过程均使用修正的 *MGS* 方法。这个默认规则将不再赘述。

为避免 GS 直交化过程中出现除零而停机，系数矩阵 $\mathbb{A}$ 的前 r 列向量必须是线性无关的。简而言之，GS 直交化方法主要适用于列满秩矩阵。

☆ **说明 3.7.** 对于列亏秩矩阵，*GS* 直交化方法的数值效果通常较差。主要原因如下：若矩阵的前 r 列向量线性相关，*GS* 直交化过程需要

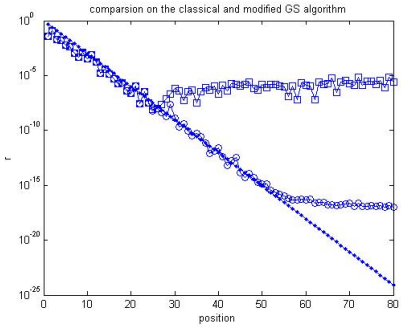

图 3.2.1: 矩阵 $\mathbb{U}\mathrm{diag}\{2^{-k}\}_{k=1}^{80}\mathbb{V}^\top$ 的对角线元素比较。

执行列交换，找到最大线性无关列向量组。对于列亏秩矩阵，虽然理论上存在相应的置换阵，但是数值实现却是非常困难的，因为舍入误差的积累会造成“数值秩”的跳跃[vii]。相关的数值处理方法超出课程范围，详略。

✍ **论题 3.3. 上三角形化的矩阵直交分解**。连续利用 *GS* 直交化过程，可以建立矩阵的**不完全直交分解**:

> *(B):* $\mathbb{A}_{m\times n} = \mathbb{Q}_{m\times r}\mathbb{R}_{r\times r}\mathbb{V}_{n\times r}^\top$, 其中 $\mathbb{R}$ 为 r 阶上三角阵, $\mathbb{Q}_{m\times r}$ 和 $\mathbb{V}_{n\times r}$ 均是列直交阵。

经过简单的列向量正交扩充，还可给出矩阵的**完全直交分解**:

> *(C):* $\mathbb{A}_{m\times n} = \mathbb{H}_{m\times m}\tilde{\mathbb{R}}_{m\times n}\mathbb{K}_{n\times n}^\top$, 其中 $\tilde{\mathbb{R}}$ 是 r 阶上三角阵 $\mathbb{R}$ 的零扩充阵, $\mathbb{H}_{m\times m}$ 和 $\mathbb{K}_{n\times n}$ 均是直交方阵。

[vii] 数值秩的概念，将容后给出。

请注意：矩阵的完全直交分解是一个非常有用的分析工具。

在列向量的正交性表现方面，MGS 方法的数值结果依旧不佳。相对而言，正交矩阵技术（Householder 镜像变换阵或 Givens 平面旋转阵）更具优势[viii]。

3.2.2 Householder 镜像变换

Householder 镜像变换最早出现在 Turnbull 和 Aitken (1932) 的书中，用于证明矩阵 Schur 分解的存在性。Householder (1958) 将其用于矩阵特征值的计算，使其名扬天下。

定义 3.3. 设 $\boldsymbol{u}_n$ 是非零的 n 维实向量，记 $b = \frac{1}{2}\|\boldsymbol{u}_n\|_2^2$；称

$$\mathbb{H}_{n\times n} = \mathbb{I}_{n\times n} - b^{-1}\boldsymbol{u}_n\boldsymbol{u}_n^{\top} \tag{3.2.2}$$

为 *Householder* 镜像变换阵。它是单位矩阵的秩一修正。

思考 3.4. 高维向量 $\boldsymbol{u}$ 的长度计算要小心“上溢”与“下溢”现象。为增强算法的健壮性，相应的代码是

```
1. m =max(abs(u));
2. y = u/m;
3. return  m * norm(y);
```

论题 3.4. 在乘法运算的计算复杂度方面，秩一修正矩阵具有显著优势。通常，n 阶矩阵同向量相乘共需 n^2 次乘除法运算，但是秩一修正矩阵只需只需 $2n+1$ 次乘除法运算。具体操作如下：

[viii] 也就仅仅在这个指标上，MGS 方法的表现弱于 Householder 方法。

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

Householder 镜像变换阵是对称正交阵，具有重要的“镜像”效应：

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

这个性质使 Householder 镜像变换阵成为重要的数值工具，其价值主要体现在如下的基本代数问题中。

论题 3.5. *已知非零的 n 维实向量 $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)^\top$，利用 Householder 镜像变换阵实现“向量的首个分量非零”，即*

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

相应的算法记为 $[\alpha, b] =$*householder*$(\boldsymbol{a})$。

图文框给出伪代码的关键片段。利用数据覆盖技术，将镜面法向量 $\boldsymbol{u}$ 保存在 $\boldsymbol{a}$ 处，故而仅需改变 $\boldsymbol{a}$ 的首个分量。输出列表包含两个浮点数：其一是 α，对应带符号的向量长度；其二是 b，对应变换参数。事实上，b 可以不用保留；这里的处理是基于后续计算的需要，采用了“空间换速度”的策略。最后强调指出：

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

1. α 的选取策略可以保证变换参数 b 具有更大的绝对值，更好地控制浮点运算的舍入误差。

2. Householder 镜像变换阵 $\mathbb{H}$ 无需保存，因为它可由保存的 b 和 $\boldsymbol{u}$ 快速给出。事实上，它也根本不需要真正计算出来，因为它和向量的乘法可以利用单位矩阵的秩一修正特点来实现。

☆ **说明 3.8.** *Wilkinson* 指出：上述算法具有极好的数值稳定性

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

其中 C 是绝对常数，ϑ 是机器精度。

❀ **思考 3.5.** 能否将论题 *3.5* 的数值目标推广到复数域？相应的障碍应当怎样解决？

✎ **论题 3.6.** 不断地左乘 *Householder* 镜像变换阵[ix]，可以将矩阵 $\mathbb{A}_{m\times n}$ 逐步变换到上三角（梯形）矩阵 $\mathbb{R}_{m\times n}$，即

$$\mathbb{H}_s \cdots \mathbb{H}_1 \mathbb{A} = \mathbb{R},$$

其中 $s = \min(m,n) - 1$。相应的数值实现方法如下：

> *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：$\mathbb{H}_k\mathbb{A}(k:m, k+1:n)$;
> *4. Enddo*

上述代码使用了数据覆盖技术。换言之，镜面法向量 $\boldsymbol{u}$ 被保存在相应的原有位置。镜像变换后的对角线元素记录在 α 中，需要开辟一个数组来保存。若要记录变换参数 b，还需再开辟一个数组来保存。至于第 *3* 步的矩阵乘积运算，它可以转化为诸多的矩阵和向量相乘，利用论题 *3.4* 的公式进行操作。

❀ **思考 3.6.** 利用保留的信息 $[\boldsymbol{u}, b]$，给出直交阵 $\mathbb{H}_s \cdots \mathbb{H}_1$ 的计算流程。

[ix]为叙述方便，不妨也将单位阵称为 Householder 变换阵。

☆ **说明 3.9.** 当基于 *Householder* 镜像变换阵进行矩阵直交化时，舍入误差的表现是完美和稳定的。换言之，上述算法也是向后稳定的。对于列满秩矩阵而言，*Householder* 镜像变换方法要好于 *MGS* 方法，因为前者给出的列直角阵具有更好的正交性。

✎ **论题 3.7.** 为更好地控制舍入误差，通常在利用 *Householder* 镜像变换进行直交化的过程中，采用相应的"主列向量"策略。换言之，在当前位置的右下角矩阵 $\mathbb{A}(k:m,k:n)$ 中，选取最大长度的列向量作为主列，并将其置换到第 k 列。

借助这种策略，上述算法也可以用于列亏秩矩阵。

3.2.3 Givens 平面旋转变换

对于稀疏矩阵，Givens 平面旋转变换可以更加快捷地实现矩阵正交化。同 Householder 镜像变换相比，它可以降低整体的计算复杂度。

✇ **定义 3.4.** 记 $c=\cos\theta$ 和 $s=\sin\theta$，称正交矩阵

$$\mathbb{G}(i,j;\theta)=\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & \cdots & s & & \\ & & \vdots & \ddots & \vdots & & \\ & & -s & \cdots & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \tag{3.2.3}$$

为 (i,j) 平面上的 *Givens* 平面旋转阵，其中 θ 为旋转角度。

除非 $s=0$，Givens 平面旋转阵是非对称的，是单位矩阵的秩二修正。这个性质异于 Householder 镜像变换阵。

继续考虑如下的基本代数问题：

论题 3.8. 已知 $\boldsymbol{a}=(\cdots,x_i,\cdots,x_j,\cdots)^\top$，其中

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

利用 *Givens* 平面旋转阵 $\mathbb{G}\equiv\mathbb{G}(i,j;\theta)$，实现“二维子向量的首个分量非零”，即

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

数值实现是简单的，相应的算法记为 $[c,s]=\mathsf{givens}(i,j,\boldsymbol{a})$。下面的图文框给出了伪代码的关键片段。

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

后面两步操作确保 $|t|\leq 1$，为了控制浮点运算的舍入误差。Wilkinson 指出，上述算法具有理想的数值稳定性，即

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

其中 C 是绝对常数，ϑ 是机器精度。

说明 3.10. 要记录平面旋转阵的具体信息，除了需保存整型的位置信息 i 和 j，还需保存浮点的角度信息 c 和 s。它们无法同时覆盖存储在 x_j 处。为此，*Stewart (1976)* 提出了一种有趣的方法，将两个角度信息 (c,s) 转化为一个浮点数 ρ。相应的伪代码片段是

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

事实上，第 *2* 步的 $|\rho|\leq 1/2$，第 *3* 步的 $|\rho|\geq 2$。要由 ρ 恢复 c 和 s 的值，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 $c=\rho/2,s=\sqrt{1-c^2}$.

因此说，*Stewart* 技术也是一种"时间换空间"的策略。

✎ **论题 3.9.** 利用 *Givens* 平面旋转变换，实现论题 *3.5* 中的数值目标？请给出相应的实现策略。

❀ **思考 3.7.** 至此，将任意向量 $\boldsymbol{a}$ 变换到首个分量非零，有两种途径：其一是多个 *Givens* 平面旋转阵 $\mathbb{G}(i_1,j_1),\ldots,\mathbb{G}(i_r,j_r)$ 的乘积，其二是单个 *Householder* 镜像变换阵。它们之间是否存在某种关系？

首先，*Householder* 镜像变换阵不可能是 *Givens* 平面旋转阵的乘积，因为 $\det\mathbb{G}=1$ 而 $\det\mathbb{H}=-1$。但是，*Givens* 平面旋转阵可表示为两个 *Householder* 镜像变换阵的乘积。请读者自己证明之。

3.2.4 三种方法的比较

对于稠密矩阵而言，Hoseholder 镜像变换方法的乘除法运算次数是 Givens 平面旋转方法的一半。事实上，前者就是为了降低后者的计算复杂度而提出的。

假设矩阵是列满秩（$m \geq n = r$）的，则 Hoseholder 镜像变换方法和 MGS 方法的乘除法运算次数分别为

$$N_{opt}^{\text{House}} \approx \sum_{k=1}^{n} 2(n+1-k)(m+1-k) \approx mn^2 - \frac{1}{3}n^3, \tag{3.2.4a}$$

$$N_{opt}^{\text{GS}} \approx \sum_{k=1}^{n} 2(k-1)m \approx mn^2. \tag{3.2.4b}$$

因此，Hoseholder 镜像变换方法具有较低的计算复杂度。

在多数情况下，Householder 镜像变换方法都占据优势。只有当矩阵还有大量零元素时，Givens 平面旋转方法才会显示出优势。

✯ **说明 3.11.** 锁定上三角阵 $\mathbb{U}$ 的对角线元素符号，则列满秩矩阵的 QR 分解是唯一的。这是一个重要的结论，详见教科书习题 *7.11.* 因此，三种直交化方法给出的列直交向量要么相同，要么反向。

✯ **说明 3.12.** *Householder* 镜像变换方法也可用于（非奇异）线性方程组的数值求解。但是，它很少被使用。原因有二。其一，*Householder* 镜像变换方法的乘除法运算次数是高斯消元方法的两倍；其二，基于列主元策略的高斯消元方法，通常已经给出可以接受的数值结果。

3.3 最小二乘解的各种表示

直交化技术可以建立最小二乘解的各种显式表达。在计算复杂度和舍入误差两个方面，基于不同表达建立的数值方法各具特色。

3.3.1 最小二乘解的基本结构

基本求解思路如下：利用正交（或者酉）变换保持向量长度不变的基本性质，将最小二乘问题等价转化为相对简单的最小二乘问题。

论题 3.10. 完全直交分解 *(C)* 可以建立最小二乘解的一般结构

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

其中 $\boldsymbol{y}$ 是任意的 $n-r$ 维向量，$r = rank(\mathbb{A}_{m\times n})$。此时，最小二乘解的残量大小为 $\|\boldsymbol{h}\|_2$。

当 $\boldsymbol{y} = 0$ 时，$\boldsymbol{x}_{LS}$ 就是极小最小二乘解。

这个结果具有重要的理论意义。但是，在实际的数值计算中，上述表达存在诸多不便，因为完全直交分解 (C) 包含列向量的直交扩张过程，相应的数值实现极其困难。事实上，对于最小二乘解的具体计算，直交扩张过程也没有提供任何有益的贡献。

3.3.2 Gram-Schmidt 直交化方法

论题 3.11. 不完全直交分解 *(B)* 可以给出极小最小二乘解

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

它需要两次 *GS* 正交化过程。

直接利用 *GS* 直交化过程 *(A)*，可以给出极小最小二乘解

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

若矩阵 $\mathbb{A}$ 是列满秩的，有更简洁的答案

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

事实上，上述三个公式中的矩阵都是 $\mathbb{A}^{\dagger}$。

✯ **说明 3.13.** 上述三个公式都需要计算 $\mathbb{H}^{\top}\boldsymbol{b}$ 或者 $\mathbb{Q}^{\top}\boldsymbol{b}$。常用的处理方法是添加 $\boldsymbol{b}$ 形成增广矩阵 $[\mathbb{A}|\boldsymbol{b}]$，并执行相应的直交化操作。在增广矩阵的最后一列，我们可以提取出相关的信息。

利用数值计算出来的直交阵 $\mathbb{H}$ 或者 $\mathbb{Q}$，进行矩阵和向量相乘，也可以得到 $\mathbb{H}^{\top}\boldsymbol{b}$ 或者 $\mathbb{Q}^{\top}\boldsymbol{b}$。但是，相应的舍入误差表现要差。这个现象也许可以解释为计算次序造成的舍入误差影响。

✯ **说明 3.14.** 在上述三个公式中，我们没有强调 $\mathbb{A}$ 是否列满秩。事实上，对于列亏秩矩阵，它们也是理论上可行的。但是，请牢记相应的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 直交矩阵变换

假设 $\mathbb{A}_{m\times n}$ 的前 r 列线性无关，其中 $r=\mathrm{rank}(\mathbb{A}_{m\times n})>0$。由于 Givens 平面旋转和 Householder 镜像变换的实现过程是类似的，不妨以后者为代表进行介绍。

✎ **论题 3.12.** 不断采用 *Householder* 镜像变换方法，对增广矩阵进行从左到右、从上到下的处理，最终可得

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

其中 $\mathbb{H}_k$ 是 $m-k+1$ 阶 *Householder* 矩阵 $\widetilde{\mathbb{H}}_{m-k+1}$ 的单位阵扩充，即

$$\mathbb{H}_k = diag\{\mathbb{I}_{k-1}, \widetilde{\mathbb{H}}_{m-k+1}\}.$$

等号右端的 $\mathbb{R}_{r\times r}$ 是可逆上三角阵，$\mathbb{U}_{r\times n} = [\mathbb{R}_{r\times r}, \mathbb{R}_{r\times(n-r)}]$ 是行满秩的上梯形阵，$\mathbb{Q} = [\mathbb{Q}_1, \mathbb{Q}_2]$ 是正交阵。

利用上述结果，可以给出一个最小二乘解

$$\boldsymbol{x}_{LS} = \mathbb{R}^{-1}\mathbb{Q}_1^\top \boldsymbol{b}, \tag{3.3.5}$$

相应的残量长度是 $\|\mathbb{Q}_2^\top \boldsymbol{b}\|_2$。

1. 若 $\mathbb{A}$ 是列满秩的，则 $\mathbb{R}_{r\times(n-r)}$ 是空的[x]，(3.3.5) 是极小最小二乘解。
2. 若 $\mathbb{A}$ 是列亏秩的，则 $\mathbb{R}_{r\times(n-r)}$ 是非空的，(3.3.5) 不一定是极小最小二乘解。要得到极小最小二乘解，增广矩阵需要在右侧施行 Householder 变换。详略。

☆ **说明 3.15.** 采纳主列向量（最长的列向量）策略，*Householder* 镜像变换方法的数值稳定性可以增强，甚至关于前 r 个列向量线性无关的限制也可以取消。换言之，此时的 *Householder* 镜像变换方法也可以用于列亏秩矩阵。

3.4 奇异值分解

奇异值分解是 Schur 分解的推广，在矩阵分析和实际应用（例如信息处理、图像压缩、多元统计分析等工程技术领域）中都具有非常重要的作用。在 Matlab 中，相应命令是 svd()。

[x]即不存在。

定理 3.8. 对于任意矩阵 $\mathbb{A}$，均存在直交阵 $\mathbb{U}$ 和 $\mathbb{V}$，使得

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

其中 $\mathbb{D} = gendiag(\sigma_1, \sigma_2, \ldots, \sigma_p)$ 是广义对角阵，$p = \min(m, n)$。

在上述公式中，$\sigma_i \geq 0$ 称为奇异值，$\mathbb{U}$ 的列向量 $\boldsymbol{u}_i$ 称为左奇异向量，$\mathbb{V}$ 的列向量 $\boldsymbol{v}_i$ 称为右奇异向量。换言之，有

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

请注意：奇异值是降序排列的，即

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

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

论题 3.13. 两个空间的线性变换可以用矩阵描述，具体的矩阵元素依赖两个空间的坐标轴选取方式。奇异值分解的几何含义是：在两个空间重新选取两个正交坐标系，使得已知的线性变换可以用坐标轴的直接变换来描述，即某个坐标轴方向映射到另一个空间的某个坐标轴方向。在物理上，上述结论说明刚体的弹性变形均可描述为旋转和拉伸两个过程。

论题 3.14. 利用奇异值分解理论，可知

1. 值域空间：$R(\mathbb{A}) = span\{\boldsymbol{u}_i\}_{i=1}^{r}$,
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.** 奇异值可以刻画给定矩阵到某个低秩矩阵集合之间的距离。设矩阵 $\mathbb{A}$ 的真实秩是 $r = \mathrm{rank}(\mathbb{A})$，若给定 $k \leq r$，则有

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

这个性质导出一个重要的概念：若奇异值 σ_k 和 σ_{k+1} 位于机器精度两侧，则称 k 是矩阵 $\mathbb{A}$ 的**数值秩**。

✍ **论题 3.16.** 若已知奇异值分解 $\mathbb{A} = \mathbb{U}\mathbb{D}\mathbb{V}^\top$，则问题 (3.0.1) 的极小最小二乘解可表示为

$$\boxed{\boldsymbol{x}_{LS} = \mathbb{V}\mathbb{D}^\dagger\mathbb{U}^\top\boldsymbol{b},}$$

其中的右端矩阵就是 $\mathbb{A}^\dagger$。该方法的数值稳定性很强，适用于病态（含列亏秩）的最小二乘问题。

✯ **说明 3.16.** 但是，奇异值分解的具体计算是非常困难的，需要耗费大量的机时。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上三角阵，然后通过迭代求解过程（类似于特征值的 *QR* 方法），将其相似变换为近似对角矩阵。在有限步数内，通常无法得到精确的奇异值信息。具体内容可参阅 *Golub* 和 *Kahan* 在 *20* 世纪 *60* 年代的工作；详略。

✯ **说明 3.17.** 作为重要的分析工具，奇异值分解可以建立各种结论，例如

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

✯ **说明 3.18.** 通常，一个图像包含的主要结构是非常有限的。利用矩阵描述图像，相应的数学解读是：反映图像重要信息的主奇异值个数是有限的，而且其个数通常远远地小于矩阵阶数。

此时，矩阵的奇异值分解理论可用于图像的压缩存储。例如，截断矩阵 $\mathbb{A}_{m\times n}$ 的秩一展开，利用前 k 个主奇异信息可以给出 $\mathbb{A}$ 的合理近似。此时，只需记录前 k 个主奇异值及其相应的左右奇异向量，图像的数据存储量可以从 mn 下降到 $(m+n+1)k$。

在图 *3.4.2* 中，我们利用上述技术压缩和恢复了 *Matlab* 系统自带的小丑图。在 *Matlab* 中，相应的实现过程非常简单，即

```
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)');
```

当 k 适当大（右下角的子图）时，恢复后的小丑图像已经同原始图像没有明显的差别了。

3.5 离散数据拟合

所谓的离散数据拟合，就是在大量的离散数据 $\{(x_i,y_i)\}_{i=1:m}$ 中，挖掘出真实规律 $\phi(x)$ 的近似（或经验）公式

$$y(x)=\sum_{j=0}^{n}\alpha_j\phi_j(x),\quad x\in\mathcal{U}, \tag{3.5.6}$$

其中 $\{\phi_j(x)\}_{j=0}^n$ 是给定的线性无关函数，$\{\alpha_j\}_{j=0}^n$ 是待定的参数。在工程材料学中，上述问题也称为“参数识别”。

通常，上述问题可以转化为线性最小二乘问题

$$y_i=\sum_{j=0}^{n}\alpha_j\phi_j(x_i),\quad i=1:m. \tag{3.5.7}$$

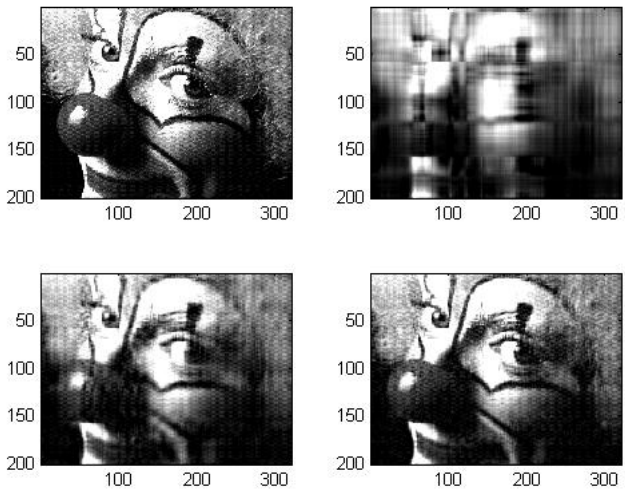

图 3.4.2: 小丑图：左上角为原图，余下三个分别对应 $k = 5, 10, 15$.

由于待定参数个数较少，我们常常采用法方程组进行求解最小二乘解。当然，前面介绍的各种数值方法均可使用。

论题 3.17. *以线性回归问题为例，陈述具有计算流程。*

为保证最小二乘解（或待定参数）的唯一性，自然希望法方程组的系数矩阵是非奇异的，或者最小二乘问题的系数矩阵是列满秩的。

论题 3.18. *在函数逼近论中，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 $\{\phi_j(x)\}_{j=0}^n$ 是线性无关的，法方程组具有唯一解，因为系数矩阵是对称正定的。

类似地，在离散数据的拟合中，最小二乘问题 (3.5.7) 也会导致一个法方程组。事实上，两个问题具有非常密切的关系。特别地，我们将会遇到一个重要问题：最小二乘问题的法方程组是否唯一可解？

要强调指出，上述目标不是永远成立的。为此，*Haar* 给出了上述目标可以成立的一个充分条件：

> ***Haar* 条件**：对于不全为零的 $\{\beta_j\}_{j=0}^n$，方程
>
> $$g(x) = \sum_{j=0}^{n} \beta_j \phi_j(x)$$
>
> 的根不超过 n 个。

因此，利用多项式进行数据拟合，答案总是存在且唯一的。

第 4 章
矩阵特征值问题的数值方法

在结构力学、电力网络、量子化学和理论物理中，存在大量的矩阵特征值问题

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

其中 $\mathbb{A}$ 是已知的 n 阶矩阵，$(\lambda, \boldsymbol{x})$ 是待解的特征值和（右）特征向量。为简单起见，本章默认 $\mathbb{A}$ 是一个实矩阵。特征值问题同时包含线性结构和非线性结构，相应的数值求解具有许多困难。

4.1 矩阵特征值问题的相关知识

本节列出矩阵特征值问题的基本概念和主要结论，特别是特征值的简单定位和敏感度分析。

4.1.1 基本概念和结论

特征多项式是首项为一的 n 次多项式

$$f(\lambda) \equiv \det(\lambda\mathbb{I} - \mathbb{A}) = \prod_{s=1}^{r}(\lambda - \lambda_s)^{n_s}, \tag{4.1.1}$$

其中 λ_s 是互异的特征值，n_s 是代数重数。换言之，特征值是特征多项式的根，其全体记为 $\lambda(\mathbb{A})$。特征向量是奇异线性方程组

$$(\lambda_s\mathbb{I} - \mathbb{A})\boldsymbol{x} = \boldsymbol{0} \tag{4.1.2}$$

的解，其全体构成特征（不变）子空间 $\ker(\lambda_s\mathbb{I}-\mathbb{A})$，相应维数

$$\gamma_s = n - \operatorname{rank}(\lambda_s\mathbb{I}-\mathbb{A})$$

称为几何重数。请注意，实矩阵的特征值和特征向量也可能是复的故而以下的讨论扩充到复数域进行。

论题 4.1. 矩阵 $\mathbb{A}$ 同 $\mathbb{A}^\top$ 的特征值是相同的。

论题 4.2. 矩阵 $\mathbb{AB}$ 和 $\mathbb{BA}$ 的非零特征值是相同的。

最小多项式是首项系数为一且使 $p(\mathbb{A})=0$ 的最低次多项式

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

可以证明，$\ker((\lambda_s\mathbb{I}-\mathbb{A})^{\ell_s})$ 构成 n_s 维的不变子空间。由凯莱定理可知，特征多项式满足 $f(\mathbb{A})=0$。但是，它不一定是最小多项式。

定义 4.1. 若存在可逆矩阵 $\mathbb{X}$，使得 $\mathbb{B}=\mathbb{X}^{-1}\mathbb{A}\mathbb{X}$，则称 $\mathbb{A}$ 和 $\mathbb{B}$ 是相似的。

论题 4.3. 相似矩阵具有相同的特征多项式和特征值。下面给出三个常见的标准形：

1. *Jordan* 分解定理：任意矩阵都可通过相似变换到 *Jordan* 标准形。作为重要的矩阵分析工具之一，*Jordan* 分解可以清楚给出所有的特征信息，并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。事实上，它就是可对角化矩阵，相应的代数重数与几何重数都是相等的。

2. 复数域的 *Schur* 分解定理：任意矩阵都可通过酉相似变换到上三角阵。

3. 实数域的 *Schur* 分解定理：任意矩阵都可通过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。

☆ **说明 4.1.** 基于 *Jordan* 分解的数值方法通常是不稳定的，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。

特征值的逼近程度可以直接用距离来刻画。由于特征向量的非零数乘依旧是特征向量，特征向量的逼近程度应当用相应子空间的距离来刻画。

✇ **定义 4.2.** 设子空间 $\mathcal{P}$ 和 $\mathcal{Q}$ 的维数相同，相应的子空间正交投影分别表示为两个幂等矩阵 $\mathbb{P}$ 和 $\mathbb{Q}$，则两个子空间的距离是

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

利用子空间的正交扩充，可以证明：

$$\|\mathbb{P}-\mathbb{Q}\|_2 = \left\{1-[\sigma_{\min}(\mathbb{P}^{\mathrm{H}}\mathbb{Q})]^2\right\}^{1/2}, \tag{4.1.5}$$

其中 $\sigma_{\min}(\mathbb{P}^{\mathrm{H}}\mathbb{Q})$ 是相应矩阵的最小奇异值。

考虑常用的简单实例。设 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是两个单位向量，相应的子空间和正交投影矩阵分别是

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

由定义和 (4.1.5) 可知，两个子空间的距离是

$$\operatorname{dist}(\mathcal{P},\mathcal{Q}) = [1-(\boldsymbol{x}^{\mathrm{H}}\boldsymbol{y})^2]^{1/2} = |\sin\theta|, \tag{4.1.6}$$

其中 θ 是向量夹角。换言之，靠近的一维子空间是指两个基底向量的夹角接近零。

❀ **思考 4.1.** 若 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 等长，$\|\boldsymbol{x}-\boldsymbol{y}\|_2$ 同夹角 θ 有何关系？

4.1.2 特征值的简单定位

直接利用矩阵元素确定特征值的分布情况，是矩阵理论的主要研究内容之一。最简单的特征值定位结论是

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

即所有的特征值都落在复平面的一个圆盘上，其中 $\varrho(\mathbb{A})$ 是谱半径，$\|\mathbb{A}\|$ 可以是任意范数。常用的矩阵范数是行范数和列范数。

定理 4.1. 【*Gerschgorin* 第一圆盘定理】矩阵 $\mathbb{A} = (a_{ij})_{n\times n}$ 的任意一个特征值至少落在复平面上 n 个圆盘

$$S_i = \left\{ \lambda \colon |\lambda - a_{ii}| \leq \sum_{j\neq i} |a_{ij}| \right\}, \quad i = 1:n$$

中的某个圆盘上。

定理 4.2. 【*Gerschgorin* 第一圆盘定理】矩阵 $\mathbb{A} = (a_{ij})_{n\times n}$ 的任意一个特征值至少落在复平面上 n 个圆盘

$$S_i = \left\{ \lambda \colon |\lambda - a_{ii}| \leq \sum_{j\neq i} |a_{ij}| \right\}, \quad i = 1:n$$

中的某个圆盘上。

定理 4.3. 【*Gerschgorin* 第三圆盘定理】设 $\mathbb{A}$ 是不可约的，λ 落在某个圆盘的边界点上。仅当每个圆盘边界都通过 λ 时，它才会成为 $\mathbb{A}$ 的特征值。

上述讨论也可用于矩阵 $\mathbb{A}^{\top}$，给出相应的结果。在实际应用时，常常引入合适的对角阵 $\mathbb{D}$，再利用圆盘定理估计 $\mathbb{D}^{-1}\mathbb{A}\mathbb{D}$ 的特征值，希望更加准确地估计 $\mathbb{A}$ 的特征值位置。

4.1.3 特征值的敏感程度

理论可证，矩阵特征值连续依赖于矩阵元素[i]。但是，在数值计算中，单有这样的结论是不够的，因为特征值连续变化的敏感程度常常差别巨大。为理解上述结论，不妨考虑特征值为零的 Jordan 矩阵

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

若最左下角的位置产生扰动 $\varepsilon > 0$，则特征值由零变成 $\sqrt[n]{\varepsilon}$，扰动产生很大影响。当扰动位置逐渐向右上方漂移时，扰动产生越来越弱的影响。特别地，当扰动位置出现在对角线上方时，特征值保持不变，扰动没有任何影响。这个实例清楚地表明，矩阵特征值问题具有非常复杂的扰动现象。

关于矩阵特征值的敏感程度，我们有整体和局部两种刻画方式。下面仅仅给出常见的定义方式，略去详细的理论证明。

整体条件数

定理 4.4 (Bauer-Fike 定理)**.** 设 $\mathbb{A}$ 和 $\mathbb{B}$ 为两个已知矩阵，其中 $\mathbb{A}$ 可通过矩阵 $\mathbb{Q}$ 相似变换为对角阵。任取 $\mathbb{B}$ 的某个特征值 $\mu \in \lambda(\mathbb{B})$，均存在 $\mathbb{A}$ 的特征值 $\lambda \in \lambda(\mathbb{A})$，使得

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

其中 $\|\cdot\|$ 是任意的从属矩阵范数。

[i]利用复变函数中的留数定理可以证明这个结论，详略。

证明：简单的特征信息描述，略。 □

定义 4.3. 矩阵 $\mathbb{A}$ 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* 定理的结果可以推广到亏损矩阵，具体结果同 *Jordan* 标准形的 *Jordan* 块最大阶数有关。尽管如此，特征值整体条件数的定义依旧有效。

定理 4.5. 设 $(\lambda, \boldsymbol{x})$ 是非亏损（或可对角化）矩阵 $\mathbb{A}$ 的近似特征信息，相应的残量记为

$$\boldsymbol{r} = \mathbb{A}\boldsymbol{x} - \lambda\boldsymbol{x},$$

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. 对于对称矩阵而言，上述结论是非常惬意的：若残量很小，则特征值的误差必然很小。但是，这个惬意的结论不适用于特征向量。例如，对称矩阵

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

的两个线性无关特征向量是 $(1,1)^\top$ 和 $(1,-1)^\top$。取近似特征信息 $\lambda=1$ 和 $\boldsymbol{x}=(1,0)^\top$，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。

Wilinson 条件数

事实上，不同特征值关于扰动的敏感程度可能存在很大差异。为说明这个现象，不妨考虑一个简单情况，即特征值是单根，其特征子空间是一维的。设 $\mathbb{E}$ 是给定的任意矩阵，考虑扰动矩阵的特征值问题

$$(\mathbb{A}+\varepsilon\mathbb{E})\boldsymbol{x}(\varepsilon)=\lambda(\varepsilon)\boldsymbol{x}(\varepsilon).$$

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，特征信息关于 ε 连续可导。计算导数值 $\lambda'(0)$，有

$$\lambda'(0)=\frac{\boldsymbol{y}^{\mathrm{H}}\mathbb{E}\boldsymbol{x}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}},$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 分别是相应的单位右特征向量和单位做特征向量。它给出描述这个单特征值关于扰动的敏感程度。

定义 4.4. 设 λ 是单特征值，相应的特征值局部条件数是

$$W(\lambda;\mathbb{A})=\frac{1}{|\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}|},\tag{4.1.10}$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 分别是相应的单位右特征向量和单位做特征向量。

说明 4.4. 上述讨论没有要求矩阵 $\mathbb{A}$ 必须可对角化，只要求 λ 是单特征根，以保证 $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}\neq 0$；请读者自行证明最后的结论。考虑反例

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

可知最后的结论不能推广到重特征值。

上述两种特征值条件数均为酉相似变换下的不变量。因此，在计算矩阵特征值的时候，我们可以放心地进行酉相似变换。

无论是整体刻画还是局部刻画，对称矩阵的特征值条件数都是最小的 1。因此说，对称矩阵的特征值问题永远都是良态的。这个结论也说明：特征值问题和线性方程组的病态概念是完全不同的。

☆ 说明 4.5. 当重特征值是亏损的时候，相应的特征值问题通常都是病态的。参见 (4.1.7) 的 *Jordan* 矩阵，可知微小的扰动可以造成特征值的巨大变化。

4.1.4 特征向量的敏感程度

特征向量关于扰动的影响比较复杂，特别是重特征值的时候。具体讨论超出课程设置，可参见 Wilkinson 的专著。下面仅仅给出简要的说明。

☆ 说明 4.6. 即使对角矩阵的特征向量，也可能因为矩阵元素的扰动而产生很大的变化。例如，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。我们有如下结论：

1. 若 ε 和 δ 均不为零，则 $\mathbb{A}+\mathbb{E}$ 的特征值是 $\lambda = 1$ 和 $\lambda = 1+\varepsilon$，而相应的特征向量是 $(\delta, -\varepsilon)^\top$ 和 $(1, 0)^\top$。适当选取 ε 和 δ 的比值，可以使第一个特征向量是任意方向。

2. 若 $\varepsilon = 0$，则特征值保持不变，但是 $\mathbb{A}+\mathbb{E}$ 只有一个线性无关的特征向量，而 $\mathbb{A}$ 具有两个线性无关的特征向量。

4.2 幂法

幂法可以快速地计算（某些）矩阵（对应最大模特征值的）的主特征信息，其设计思想和实现方法都是非常简单的。在特征值的其他计算方法研究中，幂法的讨论过程和证明技巧常常扮演着重要的角色。

4.2.1 正幂法

论题 4.4. 幂法的设计思想是：在算子（或矩阵左乘）的不断作用下，初始向量包含的不同特征成分呈现出不同的增长速度，使得主特征信息被筛选和分离出来。

假设矩阵 $\mathbb{A}$ 具有完备的特征向量系，它等价于 $\mathbb{A}$ 的初等因子都是线性的。任取一个非零向量 $\boldsymbol{v}_0$，有

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

其中 λ_1 是完全分离的主（实）特征值，即

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

这个公式表明 $\mathbb{A}^k\boldsymbol{v}_0/\lambda_1^k$ 逼近 $\mathbb{A}$ 的主特征向量。

然而，如此直接的操作在实际计算中是行不通的。主要原因如下：其一，待解的主特征信息是未知的；其二，高幂次矩阵的计算不仅破坏稀疏性，而且造成庞大的工作量和严重的舍入误差积累。出于数值操作的考量，上述公式必须进行适当的变形。

论题 4.5. 主要想法是采用递推和迭代的思想，并在执行过程中随时单位化，以避免数值计算出现上下溢出。基本结构是

$$\boldsymbol{u}_k = \mathbb{A}\boldsymbol{v}_{k-1}, \quad m_k = \max(\boldsymbol{u}_k), \quad \boldsymbol{v}_k = \boldsymbol{u}_k/m_k,$$

其中 $\max(\boldsymbol{u}_k)$ 表示在向量 $\boldsymbol{u}_k$ 中按模最大的首个分量。

定理 4.6. *在适当的条件下，上述幂法的向量 $\boldsymbol{v}_k$ 收敛*[ii]*到主特征向量 $\boldsymbol{x}_1$，而 m_k 收敛到主特征值 λ_1，且*

$$m_k = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right),$$

其中 λ_2 是按模第二大的主特征值。换言之，前两个主特征值的比值决定了幂法的收敛速度。

☆ **说明 4.7.** *要在幂法的迭代过程中有效收集到相关的主特征信息，理论上要求初始向量在主特征空间 $span(\boldsymbol{x}_1)$ 上投影分量非零。当投影分量接近零（或等于零）时，m_k 将呈现出非常缓慢的收敛过程，甚至产生所谓的假收敛过程，即它没有收敛到正确的主特征值。但是，随着数值计算的进行，舍入误差的（长时间!）积累可能起到积极的作用，或者将缓慢的收敛过程加快，或者将收敛目标纠正到正确位置。*

事实上，我们不用过分担心初始向量的选取问题。数值计算通常选取若干个初始向量。当发现收敛缓慢的时候，我们可以终止幂法，更换初始向量重新计算。

幂法的数值表现强烈地依赖于主特征信息的分布情况。对于一般矩阵而言，幂法的数值表现可能不够理想。下面假设 $\mathbb{A}$ 是一个实矩阵，且主特征值出现等模。

1. 若特征向量系没有亏损，“主特征值出现等模”只有下面三种不同状态：

[ii] 其真正含义应是收敛到特征子空间，即 $\mathrm{span}(\boldsymbol{v}_k) \to \mathrm{span}(\boldsymbol{x}_1)$，或者等价于这两个空间的距离或者两个向量的夹角趋于零。

(a) 主特征值是半单的实重根，即特征值是重根，但几何重数等于代数重数：此时，幂法依旧有效，但是特征向量将同初始向量相关。

(b) 主特征值互为相反数：此时幂法本身是不收敛的，它含有两个收敛的子列。此时，可以考虑连续执行两步幂法，并由此得到相应的主特征信息。

(c) 主特征值为等模的共轭复数：幂法虽然可用，但是相应的推广过于复杂，且计算效率不高。数值计算要用到最小二乘技术，确定对应共轭复数的实二次方程的两个系数。当特征值的虚部非常小的时候，相应的数值稳定性很差。此外，我们还要在实数域进行四则运算，实现复数域的计算效果。

2. 若主特征值是重根并造成特征向量系的亏损，幂法将不再是几何收敛，而是按照极其缓慢的调和方式收敛。

❀ **思考 4.2.** 设 $\mathbb{J}$ 是具有单一特征值 λ 的 *Jordan* 阵，$\boldsymbol{v}_0$ 是任意的非零初始向量。请观察 $\mathbb{J}^k\boldsymbol{v}_0$ 当 $k \to \infty$ 时的具体表现。

在本节结束之前，我们再次指出：幂法的计算流程和收敛情形，都强烈地依赖最大模特征值在 $r = \rho(\mathbb{A})$ 圆周上的分布情况。因此说，幂法的实际应用存在某种不便性，特别是当特征值的分布情形无法事先明确的时候。只有当矩阵阶数非常高，无法利用其它算法的时候，幂法才会成为首选方法。

4.2.2 加速技巧

✎ **论题 4.6.** 由定理 *4.6* 可知，单位化信息 m_k 以几何方式收敛到主特征值，收敛速度是线性（或者一阶）的。此时，*Aitken*（或 Δ^2）方法是行之有效的加速技术，即

$$\tilde{m}_k = m_k - \frac{(\Delta m_k)^2}{\Delta^2 m_k}$$

可以更快地趋近主特征值，其中 $\Delta m_k = m_{k+1} - m_k$ 为向前差分。

✎ **论题 4.7.** 原点平移方法也是简单易行的加速技术，即给出适当的平移量 μ，执行平移矩阵 $\mathbb{A} - \mu\mathbb{I}$ 的主特征值计算。平移量的选取，要争取实现两件事：其一，$\lambda_1 - \mu$ 依旧是平移矩阵的主特征值；其二，平移矩阵前两个主特征值的模具有尽可能小的比值。因此，最佳平移量通常是难以确定的。

原点平移方法更多地用于特征值问题的其他算法中，例如反幂法。

✎ **论题 4.8.** 若矩阵是实对称的，则 *Rayleigh* 商加速方法将被广泛采用。换言之，将幂法中的 m_k 替换为 *Rayleigh* 商，有

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

相应的向量单位化直接采用欧氏范数进行。我们不必再计算 m_k。同定理 *4.6* 相比，主特征值的线性收敛速度可以提高至平方倍。

✯ **说明 4.8.** 在矩阵理论中，*Rayleigh-Ritz* 商

$$R(\boldsymbol{x}) = \boldsymbol{x}^\top \mathbb{A}\boldsymbol{x} / \boldsymbol{x}^\top \boldsymbol{x}$$

是非常重要的概念。事实上，它是关于 μ 的矛盾方程组 $\mathbb{A}\boldsymbol{x} - \mu\boldsymbol{x} = 0$ 的最小二乘解，是 $span\{\boldsymbol{x}\}$ 空间给出的某个特征值近似。

✯ **说明 4.9.** 对于 n 阶实对称矩阵 $\mathbb{A}$，特征值可以按大小排序为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n.$$

著名的 *Rayleigh* 商定理（*Courant-Fischer* 极大极小定理）指出

$$\lambda_i = \min_{\dim \mathbb{V}=i'} \max_{\boldsymbol{x}\in\mathbb{V}} R(\boldsymbol{x}) = \max_{\dim \mathbb{V}=i} \min_{\boldsymbol{x}\in\mathbb{V}} R(\boldsymbol{x}), \tag{4.2.11}$$

其中 $i' = n+1-i$。基于这个结论，我们可以建立实对称矩阵的特征值扰动定理和交错分布定理。

☆ **说明 4.10.** *Rayleigh* 商可推广到复数域。对于任意方阵 $\mathbb{A}$，称

$$V(\mathbb{A}) = \left\{ R(\boldsymbol{x}) = \frac{\boldsymbol{x}^{\mathrm{H}}\mathbb{A}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}} : \forall \boldsymbol{x} \right\}$$

是 $\mathbb{A}$ 的值域。它构成复平面上的一个点集，具有如下性质：

1. $V(\mathbb{A})$ 包含 $\mathbb{A}$ 的全部特征值；
2. $V(\mathbb{A})$ 是酉不变的；
3. $V(\mathbb{A})$ 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，当 $\mathbb{A}$ 是 *Hermite* 实矩阵时，$V(\mathbb{A})$ 是以最大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

反幂法可以求解非奇异矩阵 $\mathbb{A}$ 按模最小的特征值信息。事实上，它就是逆矩阵 $\mathbb{A}^{-1}$ 的正幂法，即

$$\mathbb{A}\boldsymbol{v}_k = \boldsymbol{u}_{k-1}, \quad m_k = \max(\boldsymbol{v}_k), \quad \boldsymbol{u}_k = \boldsymbol{v}_k/m_k.$$

计算量主要来自同型线性方程组的数值求解。在适当的条件下，可以证明 m_k^{-1} 趋于模最小的特征值，而 $\boldsymbol{u}_k$ 趋于相应的特征向量。

✯ **说明 4.11.** 迭代过程包含大量的同型线性方程组。数值计算可以事先给出矩阵 $\mathbb{A}$ 的 LU 三角分解，只需在每次迭代中求解两个三角形线性方程组。既然初始向量是任意选取的，第一个半步的三角形线性方程组计算也是可以略去的，相应的处理称为半次迭代技术。

反幂法可用于改善特征信息的精度。设 q 是某个特征值的粗糙近似。利用原点平移方法，执行反幂法

$$(\mathbb{A}-q\mathbb{I})\boldsymbol{v}_k=\boldsymbol{u}_{k-1},\quad m_k=\max(\boldsymbol{v}_k),\quad \boldsymbol{u}_k=\boldsymbol{v}_k/m_k.$$

在适当的条件下，$q+m_k^{-1}$ 给出最接近 q 的特征值，$\boldsymbol{u}_k$ 给出相应的特征向量。

✯ **说明 4.12.** 当 q 接近 λ 时，反幂法的收敛速度极快。通常，第一步迭代即可得到理想的近似特征向量。相对而言，第二步迭代对计算效果的改善非常有限，甚至可能产生破坏作用。这个现象可以给出相应的理论保障。以可对角化的单特征值为例，其主要理由如下：

1. 当 q 接近 λ 时，矩阵 $\mathbb{A}-q\mathbb{I}$ 接近奇异，相应的线性方程组是病态的。由于舍入误差的影响，数值求解不再具有良好的准确性。

2. 在第一步迭代中，舍入误差起到了正面作用。简单地说，舍入误差主要影响数值解在特征空间的投影长度。舍入误差的影响越大，其在特征空间的投影长度越大。对于特征向量的计算而言，这是非常有利的因素，因为数值计算的核心目标是特征方向，而不是特征向量的长度。

3. 在第二步迭代中，舍入误差将起到了负面作用。具体讨论，略。

✯ **说明 4.13.** 反幂法的数值实现是以列主元高斯消元方法为基础。若消元方法执行失败，则可以认为 q 就是 $\mathbb{A}$ 的特征值。要求解相应的特

征向量，不妨对 q 进行一个极小的扰动，使得带有新平移量的反幂法可以顺利进行。

若以 Rayleigh 商为平移量加速反幂法，可得著名的 Rayleigh 商算法：任取单位向量 $\boldsymbol{q}_0$，计算 $\mu_0 = \boldsymbol{q}_0^{\mathrm{H}}\mathbb{A}\boldsymbol{q}_0$。对 $k \geq 1$，依次执行

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

注意到定理 4.5，残量长度 $\rho_k = \|(\mathbb{A} - \mu_k\mathbb{I})\boldsymbol{q}_k\|_2$ 可以刻画 μ_k 同某个特征值的靠近程度。可以证明：

1. 当 Rayleigh 商算法收敛于某个单特征值信息时，它至少具有平方收敛速度，即 ρ_{k+1} 受控于 ρ_k^2 的某个倍数。

2. 当 $\mathbb{A}$ 是对称矩阵的时候，Rayleigh 商算法具有三次方收敛速度。

详细的理论分析，略。

4.2.4 其他特征值的求解

除主特征信息之外，幂法也可以求解余下的主特征信息。下面重点介绍两种方法，即矩阵收缩技术和同时迭代方法。

矩阵收缩技术

下面以第二个主特征信息为目标，介绍矩阵收缩技术的两种实现方式：降维收缩技术和 Wieldant 收缩技术。

✎ **论题 4.9.** 降维收缩技术就是应用幂法求解某个剔除第一主特征值的低阶矩阵，从而计算出第二主特征信息。

其核心内容是找到某个可逆矩阵 $\mathbb{S}$，将已知的主特征向量 $\boldsymbol{x}_1$ 转换为首个位置非零，即

$$\mathbb{S}\boldsymbol{x}_1 = t\boldsymbol{e}_1.$$

Gauss 消元阵、Householder 镜像变换阵和 Givens 平面旋转阵，都是可行的代数工具。截取 $\mathbb{S}^{-1}\mathbb{A}\mathbb{S}$ 的 $n-1$ 阶右下角矩阵，其主特征信息蕴含待解的第二主特征信息。

❀ **思考 4.3.** 请思考下面两个技术细节：其一，给出 $\mathbb{S}^{-1}\mathbb{A}\mathbb{S}$ 的快速算法；其二，建立高阶矩阵和低阶矩阵的特征信息联系。

✍ **论题 4.10.** 降维收缩技术破坏矩阵的稀疏结构。为解决这个缺陷，可以考虑 *Wieldant* 收缩技术，即秩一修正矩阵

$$\mathbb{A}_1 = \mathbb{A} - \sigma\boldsymbol{x}_1\boldsymbol{v}^H$$

的主特征信息蕴含待解的第二主特征信息。这里，σ 和 $\boldsymbol{v}$ 的设置是关键，通常令 $\sigma = \lambda_1$ 和 $\boldsymbol{v} = \boldsymbol{x}_1$ 为已知的主特征信息。

Wieldant 收缩技术的优势是不必真正计算和存储 $\mathbb{A}_1$，只需存储 σ 和 $\boldsymbol{v}$ 即可。此外，Wieldant 收缩技术特别适用于对称矩阵[iii]，因为第一主特征值转化为零，对幂法的计算不造成任何影响。

子空间同时迭代法

在矩阵收缩技术中，第二主特征信息的精度都要明显受制于第一主特征信息的精度。逐次求解过程势必造成方法误差的不断积累，后续特征信息的计算精度变得越来越差。为克服这个困难，我们希望前 m 个主特征信息可以同时求出。

[iii] 当然，这种方法也可用于非对称矩阵，数值效果也不错。

✍ **论题 4.11.** 基本的子空间同时迭代方法：任取 m 个列直交向量，组成列直交矩阵 $\mathbb{V}_0$；然后，执行循环操作

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

每步迭代包含两个步骤：第一步是正幂法的左乘过程；第二步是矩阵的 QR 分解。

第二步的 QR 分解是子空间同时迭代方法的关键。它可以重构（斜）像空间的正交基底，避免特征空间的数值坍塌，减少舍入误差对线性无关性质的影响。

✯ **说明 4.14.** 初始正交列向量组选取，主要有两种方法。其一，给出一个随机向量，构造相应的 *Householder* 矩阵，再随机地选取某些列向量；其二，随机构造一组向量，进行 *Gram-Schmidt* 正交化。数值经验表明，后者的数值表现要好一些。

当 $\mathbb{A}$ 是实对称矩阵的时候，广义 Rayleigh 商技术是常用的加速技术。它是 Rayleigh 商技术的推广，也称为子空间投影算法。

✍ **论题 4.12.** 子空间投影算法是常用的数值技术，即在低维子空间求解高维问题的近似解。基本思想如下：

为简单起见，不妨假设低维空间是由 $\mathbb{V}_{k-1}$ 的 m 个列直交向量张成的子空间。将待解的特征信息局限于此，基于投影技术或最小二乘思想，可以导出一个小规模的 m 阶特征值问题

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

显而易见，这个低维问题的求解要相对容易。在某种程度上，$\tilde{\lambda}_{k-1}$ 是矩阵 $\mathbb{A}$ 的某个近似特征值，$\mathbb{V}_{k-1}\boldsymbol{y}_{k-1}$ 是某个近似特征向量。

教科书的算法就是上述算法的结合，也称为子空间同时迭代法。

定理 4.7. 设实对称矩阵 $\mathbb{A}$ 的特征值彼此互异，则子空间同时迭代法关于特征值和特征向量均具有很好的收敛性质。

证明：证明是典型的。见教科书。 □

☆ **说明 4.15.** 在子空间同时迭代法中，低阶矩阵的特征值信息可以直接计算或公式表达。当迭代步数 k 足够大时，低阶矩阵接近一个对角阵，即将介绍的 *Jacobi* 方法是非常有效和快捷的。

4.3 实对称矩阵的 Jacobi 方法

回忆熟知的代数结论：实对称矩阵可以正交相似变换到对角阵。通常，我们无法简单机械地直接给出相应的正交矩阵。那么，它能否通过一系列的正交矩阵来实现吗？注意到平面旋转阵相乘依旧是正交矩阵的性质，Jacobi (1846) 基于上述想法，提出了 Jacobi 方法。

事实上，Jacobi 方法没有实现最初的目标。在通常情况下，它无法在有限步数内完成特征值的精确计算。尽管如此，这个古老算法具有编程简单和并行率高等优点，依据具有相当广泛的数值应用。

4.3.1 基本思想

对于二阶实对称矩阵，Jacobi 方法的思想可以得到完美体现。换言之，存在一个二阶 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

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

可以实现矩阵对角化，即 $b_{pq} = b_{qp} = 0$。相应操作称为 Jacobi 旋转，非对角元素 a_{pq} 称为旋转主元，θ 称为旋转角度。这里，p 和 q 是二阶子阵的两个行号。

简单计算可知，旋转角度 θ 满足关系式

$$\cot 2\theta = \frac{a_{pp} - a_{qq}}{2a_{pq}} \equiv \xi. \tag{4.3.12}$$

通常要求 $\theta \in [-\pi/4, \pi/4]$，使得 $t = \tan\theta$ 的绝对值不超过 1，即

$$t = \tan\theta = \operatorname{sgn}(\xi)\left[|\xi| + \sqrt{1+\xi^2}\right]^{-1}. \tag{4.3.13}$$

此时，平面旋转矩阵 $\mathbb{G}(p,q;\theta)$ 中的两个元素是

$$c = \cos\theta = \frac{1}{\sqrt{1+t^2}}, \quad s = \sin\theta = ct. \tag{4.3.14}$$

论题 4.13. *若 $|\xi|$ 非常大的时候，旋转角度 θ 接近零，上述公式损失大量的有效位数，使得计算结果总是 $c \approx 1$ 和 $s \approx 0$，相应的 Jacobi 旋转没有任何实质性贡献。为克服舍入误差的影响，我们有两种方案：或者将 (4.3.13) 修正为*

$$t = 1/(2\xi),$$

或者采用新的计算公式

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

可以证明：当 $\theta \to 0$ 时，有 $\phi - \theta \approx \frac{5}{4}\theta^3$，两个三角函数值 $\cos\phi$ 和 $\sin\phi$ 分别是 c 和 s 的有效逼近。

✯ **说明 4.16.** 设 $\mathcal{E}$ 是用户的精度要求，例如 $\mathcal{E}=10^{-14}$。在实际计算中，只要探测到 $|a_{pq}|<\mathcal{E}\sqrt{a_{pp}a_{qq}}$，即可数值上认定 a_{pq} 为零。

4.3.2 古典 Jacobi 方法

当 Jacobi 思想应用到高阶矩阵时，非对角线元素的零化过程遇到障碍。一般来讲，正交相似对角化过程无法在有限步数内结束。主要原因是：旋转相似变换影响 (p,q) 井字线上的全体数据，已经零化的非对角元素可能在后续操作中重新变成非零。

✎ **论题 4.14.** *古典 Jacobi 方法*：令 $\mathbb{A}_0=\mathbb{A}$。在第 k 步迭代中，首先搜索旋转主元

$$a_{pq}^{(k)}=\arg\max_{i\neq j}|a_{ij}|,$$

然后执行相应的 *Jacobi* 旋转

$$\mathbb{A}_{k+1}=\mathbb{G}_k\mathbb{A}_k\mathbb{G}_k^{\top},$$

其中 $\mathbb{G}_k=\mathbb{G}_k(p,q;\theta)$ 是 (p,q) 平面的 *Jacobi* 旋转矩阵。

定理 4.8. *在古典 Jacobi 方法中，矩阵序列 $\mathbb{A}_k$ 本质收敛到对角矩阵。本质收敛是指在集合意义下的收敛。*

证明：观察 Jacobi 旋转变换过程前后的非对角元素平方和，可证相应的 Frobenius 范数趋于零。 □

✯ **说明 4.17.** *在 Jacobi 方法中，$|t|\leq 1$ 保证对角线元素可以目标稳定地趋向某个特征值，不会跳转到其他的特征值。换言之，矩阵序列可以真正地收敛到某个固定的对角阵*[iv]。

[iv] 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。

✯ **说明 4.18.** 若特征值互异，可以证明 *Jacobi* 方法给出的特征向量也是收敛的。若有重特征值，虽然不再保证正交矩阵的列向量关于特征向量的收敛性，但可确保多维特征子空间的收敛性。

✯ **说明 4.19.** *Demmel* 和 *Veselić* 指出：*Jacobi* 方法是数值稳定的，特征值的相对误差被 $\vartheta\kappa_2(\mathbb{D}^{-1/2}\mathbb{A}\mathbb{D}^{-1/2})$ 所控制，其中 $\mathbb{D} = diag(\mathbb{A})$ 是对角阵，ϑ 是机器精度，$\kappa_2(\cdot)$ 是谱条件数。

✎ **论题 4.15.** *Jacobi* 旋转包含 *Givens* 平面旋转阵的左乘和右乘，仅仅影响位于 (p,q) 井字线上的元素。注意到旋转角度的计算公式，对角线元素的计算公式可以简化为

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

位于井字线上非交叉点的矩阵元素，仅需两次乘除法运算即可得到。注意到矩阵的对称性，*Jacobi* 旋转所需的乘除法次数是 $\mathcal{O}(4n)$。

4.3.3 循环 Jacobi 方法

旋转主元的全局搜索需要消耗大量的 CPU 时间。因此，不妨放弃全局搜索，逐个地 Jacobi 旋转非对角线元素。相应的 $n(n-1)/2$ 次旋转构成一个 Jacobi 扫描。不断执行 Jacobi 扫描，相应的算法称为循环 Jacobi 方法。

✯ **说明 4.20.** *Schonhage (1964)* 和 *Van Kempen (1966)* 指出：若循环 *Jacobi* 方法收敛，则它具有渐近平方收敛速度。此外，*Brent* 和 *Luk (1985)* 指出：扫描总次数同 $\log n$ 成正比例，其中 n 为矩阵阶数。

✎ **论题 4.16.** 当非对角线元素 a_{pq} 接近零时，相应的 *Jacobi* 旋转不会有助于算法的收敛性表现。在实际计算中，我们通常会引进**阈值扫**

描策略：设 $\sigma \geq n$ 是固定常数，逐步设置阈值

$$\delta_k = \delta_{k-1}/\sigma, \quad k = 1, 2, \ldots.$$

直至矩阵元素达到机器精度或者满足用户要求为止，其中 δ_0 为矩阵 $\mathbb{A}$ 非对角元素平方和开根号。

循环 *Jacobi* 算法修正如下：只要非对角线元素满足 $|a_{pq}| < \delta_k$，相应的 *Jacobi* 旋转可以跳过；当所有非对角线元素均被跳过的时候，阈值需要按照上述规则重新设置。

❀ **思考 4.4.** 证明：在阈值扫描策略下，循环 *Jacobi* 方法给出的矩阵序列也是收敛的。

✯ **说明 4.21.** 通常，循环 *Jacobi* 方法的收敛速度不如 *QR* 方法，但是它具有并行优势。换言之，行列指标集 $\{(p, q)\}$ 可以适当地轮换分组，*Givens* 平面旋转矩阵的左（右）乘运算可以安排在不同的 *CPU* 上，同时进行。

4.3.4 特征向量的计算

✎ **论题 4.17.** 在 *Jacobi* 方法执行过程中保存的 $\{(p, q; c, s)\}$ 是相关的旋转信息，可以用于快速恢复某个（或全部）特征值对应的特征向量。若仅仅求解某个特征值对应的特征向量，更常用的方法是带有偏移量的反幂法。

4.4 实对称矩阵的 Givens-Householder 方法

它是两种数值策略的有机结合。其一是在有限步数内结束的三对角化过程，其二是迭代收敛的二分法求根法。

4.4.1 三对角化策略

利用有限次正交相似变换，对称矩阵能够达到的最简单结构是对称三对角矩阵。虽然实现过程不同，两种方法的基本思想是一样的。

论题 4.18. *Givens* 平面旋转阵实现对称矩阵的三对角化。主要思路是依次构造 *Givens* 平面旋转阵，利用**副对角线元素**将所有的副对角线下方元素清零。相应的旋转信息可存储在原有的位置。详细处理可参见说明 *3.10*。

由于实现目标同 *Jacobi* 方法截然不同，相应的计算公式（特别是旋转角度）也是完全不同的。

论题 4.19. *Householder* 镜像变换阵也能实现上述目标。主要思路如下：假设左上角矩阵 $\mathbb{A}_k$ 已经成功实现三对角化，在 **第 k 列下方**存在非零的 $n-k$ 维向量 $\boldsymbol{a}_k = (a_{k,k+1}, a_{k,k+2}, \dots, a_{kn})^\top$。利用 $n-k$ 阶 *Householder* 镜像变换阵 $\mathbb{H}_{n-k}$，进行矩阵左乘运算，有

$$\begin{bmatrix} \mathbb{I}_k & 0 \\ 0 & \mathbb{H}_{n-k} \end{bmatrix} \left[\begin{array}{cc:c} & & 0 \\ & \mathbb{A}_k & \\ & & \boldsymbol{a}_k^\top \\ \hdashline 0 & \boldsymbol{a}_k & \mathbb{A}_{n-k} \end{array}\right] = \left[\begin{array}{cc:c} & & 0 \\ & \mathbb{A}_k & \\ & & (\mathbb{H}_{n-k}\boldsymbol{a}_k)^\top \\ \hdashline 0 & \mathbb{H}_{n-k}\boldsymbol{a}_k & \mathbb{H}_{n-k}\mathbb{A}_{n-k} \end{array}\right],$$

为实现三对角化，向量 $\mathbb{H}_{n-k}\boldsymbol{a}_k$ 应当仅首个位置非零。数值计算可以采用数据覆盖技术，将 *Householder* 镜像变换矩阵的信息保存在相应位置。变换之后的副对角元素可以额外开辟空间存储。

上述两种方法的优劣主要体现在它们的计算复杂度。当矩阵非常稠密的时候，Hoseholder 镜像变换方法更具优势，乘除法运算次数由 $\mathcal{O}(4n^3/3)$ 下降到 $\mathcal{O}(2n^3/3)$，开根号运算次数也由 $n^2/2$ 下降到 $n-2$。

当矩阵非常稀疏的时候，Givens 平面旋转方法才会显现出优势。

❀ **思考 4.5.** 能否使用 *Gauss* 消元阵，在有限步数内完成对称矩阵的三对角化呢？请给出具体的实现过程，并评价其优缺点。

4.4.2 二分法

下面利用二分求根法，求解实对称三对角矩阵

$$\mathbb{T}_n = \mathrm{symtridiag}(\{\alpha_i\}_{i=1}^n, \{\beta_i\}_{i=2}^n)$$

的特征值，其中 α_i 是对角线元素，β_i 是副对角线元素。不妨假设 $\mathbb{T}_n$ 是不可约矩阵，即 β_i 都不是零[v]。

特征值的计算

对应各阶顺序主子式，定义一个次数递增的多项式序列

$$p_i(\lambda) = \det[(\mathbb{T}_n - \lambda\mathbb{I}_n)(1:i, 1:i)], \quad i = 1, 2, \ldots, n.$$

显然，$p_n(\lambda)$ 是 $\mathbb{T}_n$ 的特征多项式，它的根就是 $\mathbb{T}_n$ 的特征值。利用行列式的运算规则，有递推关系式

$$\boxed{p_i(\lambda) = (\alpha_i - \lambda)p_{i-1}(\lambda) - \beta_i^2 p_{i-2}(\lambda), \quad i = 2, \ldots, n,}$$

其中 $p_1(\lambda) = \alpha_1 - \lambda$，补充定义 $p_0(\lambda) = 1$。理论证明，这个多项式序列还满足如下性质：

1. $\mathrm{sgn}\, p_i(-\infty) = 1$，$\mathrm{sgn}\, p_i(+\infty) = (-1)^i$；
2. 相邻两个多项式没有公共根；

[v]若某个 β_i 为零，则高阶矩阵的特征值问题可以分割为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. $p_i(\lambda)$ 的根全是单根，并且严格地隔开 $p_{i+1}(\lambda)$ 的根。

请注意，最后一条性质是至关重要的。

定义 4.5. 对于给定的实数 μ，数列 $\{p_i(\mu)\}_{i=0}^{k}$ 构成一个 *Sturm* 序列；在这个 *Sturm* 序列中，相邻两数符号相同的总数称为符号相同数，记为 $s_k(\mu)$。

补充说明：若 $p_i(\mu)=0$，则称 $p_i(\mu)$ 和 $p_{i-1}(\mu)$ 符号相反，$p_{i+1}(\mu)$ 和 $p_i(\mu)$ 符号相同。

利用数学归纳法，我们可以证明

定理 4.9. 在 $(\mu,+\infty)$ 内，$p_r(\lambda)$ 恰好有 $s_r(\mu)$ 个根。

该定理表明，矩阵 $\mathbb{T}_n$ 在 $(a,b]$ 内有 $s_n(a)-s_n(b)$ 个特征值。结合二分法的思想，不断地缩减区间长度，可以确定第 k 个（从大到小排序）特征值。基本计算过程如下：

1. 特征值的隔离：
   (a) 简单界定特征值范围 $[a,b]$;
   (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 $[c,b]$，使左端点的符号同号数为 k，而右端点的符号同号数为 $k+1$。
2. 特征值的确定：
   继续区间的等分操作，放弃左右端点符号相同数相等的那个小区间，直到区间长度达到用户的精度要求。

显然，算法在理论上是无条件收敛的。

✯ **说明 4.22.** 舍入误差限制二分法的计算精度。尽管如此，标准的浮点运算误差分析指出，二分法是数值稳定的。

✯ **说明 4.23.** 对于高阶矩阵，*Sturm* 序列 $\{p_i(\mu)\}_{i=0}^n$ 的数值计算可能遇到严重的舍入误差问题，甚至存在上（下）溢出的风险。为此，符号相同数的统计过程可以修正如下：令 $q_1(\mu)=p_1(\mu)$，递归计算

$$q_i(\mu)=\alpha_i-\mu-\frac{\beta_i^2}{q_{i-1}(\mu)},\quad i=2,3,\ldots.$$

最后一项的比值可以理解为相应的极限：

1. 若 $q_{i-1}(\mu)=0$，则直接定义 $q_i(\mu)=-\infty$；
2. 若 $q_{i-1}(\mu)=-\infty$，则直接定义 $q_i(\mu)=\alpha_i-\mu$；

此时，符号相同数 $s_k(\mu)$ 就是序列 $\{q_i(\mu)\}_{i=1}^k$ 的非负个数。

特征向量的计算

在确定特征值之后，利用直接法（例如令 $x_1=1$）可以解出 $\mathbb{T}_n$ 的特征向量。但是，这种方法的数值稳定性较差。此时，带有原点平移的反幂法更为有效，它不仅改善特征值的近似程度，还可以给出相应的特征向量。

利用正交相似约化过程的保存信息，由 $\mathbb{T}_n$ 的特征向量可以重构矩阵 $\mathbb{A}$ 的特征向量。

4.5 QR 方法

QR 方法在 1961-1962 年提出，其前身是基于矩阵三角分解的 LR 方法 (1958)。它可以同时求解中小规模稠密矩阵全部特征信息，同矩阵

Schur 分解密切关系。

高次多项式 $p_n(x) = a_0 + a_1x + a_2x^2 + \cdots a_{n-1}x^{n_1} + x^n$ 的求根问题，通常转化为友矩阵

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

的特征值求解问题。在 Matlab 中，多项式求根命令 roots() 就是利用 QR 方法求解友矩阵，其计算复杂度约为 $\mathcal{O}(n^3)$。最近研究表明，充分利用友矩阵的特点，计算复杂度可以降到 $\mathcal{O}(n^2)$；详略。

4.5.1 基本思想

论题 4.20. *QR 算法的基本结构：记 $\mathbb{A}_0 = \mathbb{A}$。对 $k \geq 0$，依次进行矩阵的直交分解和交换相乘：*

$$\mathbb{A}_k = \mathbb{Q}_k\mathbb{R}_k, \quad \mathbb{A}_{k+1} = \mathbb{R}_k\mathbb{Q}_k,$$

其中 $\mathbb{Q}_k$ 是正交阵，$\mathbb{R}_k$ 是上三角阵。

显然，$\{\mathbb{A}_k\}_{k=0}^{\infty}$ 的矩阵都是正交相似的。QR 方法具有较为复杂的收敛性表现，明确的收敛结论有

定理 4.10. *设 $\mathbb{A}$ 的特征值均为实数，且按绝对值是严格分离的。若以 $\mathbb{A}$ 的左特征向量为行向量组成的矩阵 $\mathbb{X}$ 具有 LU 分解，则 QR 方法给出的迭代序列 $\mathbb{A}_k$ 本质收敛到上三角阵。*

证明类似于前面的子空间同时迭代法，具体过程略。下面仅仅给出一些简要的说明。

✍ **论题 4.21.** 事实上，*QR* 方法同幂法具有密切联系，因为

$$\mathbb{A}^k = \underbrace{\mathbb{Q}_1\mathbb{Q}_2\cdots\mathbb{Q}_k}_{\widetilde{\mathbb{Q}}_k}\underbrace{\mathbb{R}_k\cdots\mathbb{R}_2\mathbb{R}_1}_{\widetilde{\mathbb{R}}_k}.$$

讨论 $\mathbb{A}^k\boldsymbol{e}_1$ 同 $\mathbb{A}_k\boldsymbol{e}_1$ 的关系，利用正幂法的收敛性，即可阐述最左面一列的收敛情况。类似地，利用反幂法即可给出最下面一行的收敛情况。通常，迭代矩阵的最右下角元素具有最快的收敛速度。

✯ **说明 4.24.** 事实上，*QR* 方法同 *Rayleight* 商迭代的联系更为密切。通常，它具有平方收敛速度。对于实对称矩阵，它还可以达到三次方的收敛速度。

4.5.2 数值实现

直接对矩阵 $\mathbb{A}$ 执行 QR 方法，每步迭代均需较高的计算复杂度，影响 QR 方法的计算效率。下面介绍两个常用的改良方法，即事前处理和平移加速。

上 Hessenberg 化过程

✍ **论题 4.22.** 在迭代进行之前，将 $\mathbb{A}$ 正交相似变换到上 *Hessenberg* 矩阵，可以降低 *QR* 方法每步迭代的计算复杂度。

上 *Hessenberg* 过程同三对角化过程是非常类似的，详略。主要区别是，对于非对称矩阵，上三角部分需要花费时间来计算。通常，上 *Hessenberg* 过程需 $\mathcal{O}(5n^3/3)$ 次乘除法运算。

错位相乘方法

✎ **论题 4.23.** 对于上 *Hessenberg* 矩阵，*QR* 方法的直交分解可以利用有限个 *Givens* 平面旋转阵来实现。换言之，用对角线元素，将其下方的副对角线元素依次旋转为零，即

$$\mathbb{A}_{k+1} = \mathbb{G}_{k-1}^{\top} \cdots \mathbb{G}_1^{\top} \mathbb{A}_k \mathbb{G}_1 \cdots \mathbb{G}_{k-1}$$

因此，每次 *QR* 迭代只需 $\mathcal{O}(2n^2)$ 乘除法运算。

但是，若要计算过程中上 *Hessenberg* 矩阵结构保持不变，具体的操作过程还需要深入的研究。

1. 若从 $\mathbb{A}_k$ 出发，不断地进行 *Givens* 旋转相似变换，则相应的相似矩阵，例如 $\mathbb{G}_1^{\top}\mathbb{A}_k\mathbb{G}_1$，不再保持上 *Hessenberg* 结构。

2. 错位相乘技术可以解决上述问题。首先，左乘 $\mathbb{G}_1^{\top}$，得到上 *Hessenberg* 矩阵 $\mathbb{G}_1^{\top}\mathbb{A}_k$，而且位于 $(2,1)$ 位置的元素已经清零；然后，分别左乘和右乘不同的 *Givens* 平面旋转阵，得到上 *Hessenberg* 矩阵 $\mathbb{G}_2^{\top}\mathbb{G}_1^{\top}\mathbb{A}_k\mathbb{G}_1$。继续沿用上述思路，直到乘以所有的 *Givens* 平面旋转阵。

原点平移技术

✎ **论题 4.24.** 原点平移技术可以提高 *QR* 方法的收敛速度。最简单策略是单步位移的 *QR* 方法

$$\boxed{\mathbb{A}_k - t_k\mathbb{I} = \mathbb{Q}_k\mathbb{R}_k, \quad \mathbb{A}_{k+1} = \mathbb{R}_k\mathbb{Q}_k + t_k\mathbb{I},}$$

其中 t_k 为位移量，有如下两种选取策略：

1. 用右下角元素作为位移量，即 $t_k = a_{nn}^{(k)}$.

2. 截取右下角的二阶矩阵

$$\begin{bmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{n,n}^{(k)} \end{bmatrix}, \tag{4.5.15}$$

用最靠近 $a_{n,n}^{(k)}$ 的那个特征值作为平移量，称为 *Wilkinson* 位移量。它特别适合于对称三对角矩阵，相应的位移量是

$$\boxed{t_k = a_{n,n}^{(k)} + \delta - sgn(\delta)\sqrt{\delta^2 + \beta_{n-1}^2},}$$

其中 $\delta = (a_{n-1,n-1}^{(k)} - a_{n,n}^{(k)})/2$ 和 $\beta_{n-1} = a_{n-1,n}^{(k)} = a_{n,n-1}^{(k)}$。

☆ **说明 4.25.** 利用上述位移策略，QR 方法依旧可能是不收敛的。例如，二阶置换矩阵

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

具有等模的两个特征值。若采用第一种位移量，迭代序列陷入循环状态，相应的 QR 方法是不收敛的。

矩阵收缩

当 $a_{n,n-1}^{(k)} \approx 0$ 时，$a_{nn}^{(k)}$ 可视为某个特征值的近似。当 $a_{n-1,n-2}^{(k)} \approx 0$ 时，可将 (4.5.15) 的两个特征值作为相应的近似特征值。此时，右下角的一阶或者二阶矩阵可以剔除出去，矩阵阶数得以缩减。

✎ **论题 4.25.** 副对角线元素为零的数值判断准则是

$$|a_{n,n-1}^{(k)}| \leq \mathcal{E} \min(|a_{n,n}^{(k)}|, |a_{n-1,n-1}^{(k)}|),$$

其中 $\mathcal{E}$ 是预先给定的很小正数。

4.5.3 隐式对称 QR 方法

从上 Hessenberg 矩阵正交相似变换到另一个上 Hessenberg 矩阵的过程，是 QR 方法数值实现的核心。除了前面介绍的错位相乘技术之外，我们还可以基于“驱逐出境”策略，采用不同的 Givens 平面旋转和序列，隐含地实现上述目标。

相应的理论基础是矩阵上 Hessenberg 化的唯一性，即下面的定理：

定理 4.11. *考虑 $\mathbb{A}$ 的两个上 Hessenberg 化过程*

$$\mathbb{U}^\top \mathbb{A}\mathbb{U} = \mathbb{H}, \quad \mathbb{V}^\top \mathbb{A}\mathbb{V} = \mathbb{G},$$

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

$$\mathbb{U} = \mathbb{V}\mathbb{D}, \quad \mathbb{H} = \mathbb{D}\mathbb{G}\mathbb{D}, \quad \mathbb{D} = \mathrm{diag}\{\pm 1\}.$$

为简单起见，假设 $\mathbb{T}$ 是不可约的实对称三对角矩阵。带有单步位移的 QR 方法可以解出全部的（实）特征值。若利用“驱逐出境”策略，则可以建立相应的隐式对称 QR 算法。关键的伪代码片段如下：

1. 计算 Wilkinson 位移量 μ; 令 $x = \mathbb{T}(1,1) - \mu$ 和 $y = \mathbb{T}(2,1)$;
2. For $k = 1 : n-1$, Do
3. 　　计算 $[c, s] = \textsc{givens}(x, y)$，将 $(x, y)^\top$ 中的 y 旋转为零; 相应的旋转阵记为 $\mathbb{G}(k, k+1)$;
4. 　　计算 $\mathbb{G}(k, k+1)\mathbb{T}\mathbb{G}(k, k+1)^\top$，并赋值给 $\mathbb{T}$;
5. 　　若 $k < n-1$，令 $x = \mathbb{T}(k+1, k)$ 和 $y = \mathbb{T}(k+2, k)$;
6. Enddo

若将对称三对角矩阵存储为两个向量，上述伪代码需要进行相应的修改。

4.5.4 双重位移的 QR 方法

若平移量是实的，QR 方法不可能计算出共轭的复特征值。此时，平移量必须是复数，相应的 QR 方法也要在复数域中执行。但是，数据存储的负担将会增加，计算效率也会严重下降。

✎ **论题 4.26.** *注意到共轭特征值含于 Schur 阵的二阶对角块中，连续两步的位移量不妨取为共轭复数。此时，QR 方法的连续两步迭代操作可以回归到实数域中完成。这种方法称为双重位移的 QR 方法。*

若直接使用双重位移的 QR 方法，则需计算矩阵的平方，产生较高的计算复杂度和舍入误差影响。此时，我们可用隐式 QR 算法进行优化。详略。

4.6 奇异值分解

略。

第 5 章
非线性方程（组）的数值方法

非线性方程（组）是普遍存在的数学问题，相应问题的数值方法研究极富挑战性。基本思路是基于某个不动点方程的迭代求解过程。不同于线性方程组，非线性方程（组）的数值方法具有截然不同的数值表现和理论问题。

5.1 基本概念

要求解非线性方程（组）$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$ 的解（或）根，我们通常构造一个迭代序列 $\{\boldsymbol{x}_k\}$，它由 r 阶迭代公式

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

生成，其中 $\boldsymbol{g}$ 是给定的迭代函数，$\{\boldsymbol{x}_k\}_{k=0}^{r-1}$ 是启动初值。相比于线性方程组的迭代方法，我们还要明确以下概念：

1. 迭代序列是确定的：迭代序列一直落在迭代函数 $\boldsymbol{g}$ 的定义域，使相应的迭代公式保持有效。
2. 收敛情况的刻画：非线性问题的迭代收敛性较为复杂，包括全局（或大范围）收敛和局部收敛。局部收敛分析还分两种情况：
   (a) 若假定问题和方法在**真解附近**的局部信息，则相应的分析称为（传统的）局部收敛分析。
   (b) 若假定问题和方法在**初值附近**的局部信息，则相应的分析称为半局部收敛分析。

对于线性问题而言，迭代方法仅有收敛或发散两种状态，因为若其收敛必全局收敛。

3. 收敛速度的刻画：记第 k 步迭代误差为 $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}_\star$，其中 $\boldsymbol{x}_\star$ 是某个真解。假设存在两个正常数 p 和 C，使得[i]

$$\|\boldsymbol{e}_{k+1}\| \leq C\|\boldsymbol{e}_k\|^p, \quad \forall k \geq k_0, \tag{5.1.1}$$

其中 $\|\cdot\|$ 是某种向量范数[ii]。为了分析简便，我们常常将 (5.1.1) 简化为极限形式

$$\lim_{k\to\infty} \frac{\|\boldsymbol{e}_{k+1}\|}{\|\boldsymbol{e}_k\|^p} = C. \tag{5.1.2}$$

此时，有如下概念：

(a) 若 $C \neq 0$，称算法是 p 阶收敛。若 $C = 0$，称算法至少 p 阶收敛。

(b) 当 $p = 1$ 时，界定常数还需满足 $0 \leq C < 1$。若 $C \neq 0$，称算法是线性收敛；若 $C = 0$，称算法是超线性收敛。

4. 算法效率的刻画：迭代误差达到同样要求所花费的计算时间，决定了算法的效率。这是一个非常重要的问题。对于非线性方程组，算法的效率常常称为限制其应用的瓶颈之一。

设 W 是单步迭代的计算复杂度，例如所需乘除法总次数或者 CPU 时间等。通常，算法效率可以定义为

$$\eta = \begin{cases} \frac{1}{W}\ln p, & \text{若 } p > 1; \\ \frac{1}{W}\ln C, & \text{若 } p = 1. \end{cases}$$

[i] 从实用角度出发，通常要求 $p \geq 1$。

[ii] 对标量方程而言，$\|\cdot\|$ 就是绝对值。

换言之，为提高计算效率，要么提高收敛阶（或者收敛速度），要么降低单步迭代的计算复杂度。

5. 数值稳定性：算法必须有效控制舍入误差的影响，保证理论上的收敛效果。具体分析较为繁琐，详略。通常，数值观察舍入误差在不同算法中的演化过程，有助于数值稳定性的理解。

✯ **说明 5.1.** 停机标准的设置是类似的。最主要的度量方式是残量和相邻误差，即

$$\|\boldsymbol{f}(\boldsymbol{x}_k)\| \leq \mathcal{E}, \quad \text{或者} \quad \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\| \leq \mathcal{E},$$

其中 $\mathcal{E}$ 是用户给定的要求。实际计算常常同时采用上述两种策略。

5.2 标量方程的数值求解

本节讨论单个非线性方程的数值求解技术。在 Matlab 中，命令 fzero() 可以计算在猜测值附近的某个根。

5.2.1 二分法

✎ **论题 5.1.** 它是闭区间上连续函数介值定理的直接应用。若初始区间内仅有一个实根，我们可以不断地折半区间，得到一列长度趋于零的嵌套区间。在这个序列中，两个端点处的函数值一直具有相反的符号。

区间二分法的思想非常简单，但是它给出的数值解近似度不高。此外，它不易推广到非线性方程组。

✎ **论题 5.2.** 利用向后误差分析技术，我们可以简单探讨舍入误差对于二分算法的具体影响。我们要警惕假停机现象，以及停机标准不能过低等等。

论题 5.3. 类似的计算方法还有所谓的试位法，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。

5.2.2 不动点迭代

不动点迭代（或者 Picard 迭代）就是找到同解于 $f(x)=0$ 的某个不动点方程

$$x = g(x),$$

进而构造出相应的一阶迭代公式

$$x_{k+1} = g(x_k), \tag{5.2.3}$$

其中 $g(x)$ 称为迭代函数或者不动点函数。

迭代函数决定算法的收敛表现。常用的结论有：

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

1. $g(x) \in [a,b], \forall x \in [a,b]$;
2. 存在 Lip 常数 $0 \le L < 1$，使得
$$|g(x)-g(y)| \le L|x-y|, \forall x,y \in [a,b];$$

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

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

证明：略。 □

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

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

则不动点迭代 (5.2.3) 是局部收敛的，且具有 m 阶收敛。

5.2.3 加速迭代收敛

✍ **论题 5.4.** 若迭代序列 $\{x_k\}_{k=0}^{\infty}$ 线性收敛到某个真解 $x_\star$，我们可采用 *Aitken* 加速技术，定义新序列 $\{\tilde{x}_k\}_{k=1}^{\infty}$，其中

$$\tilde{x}_k = x_k - \frac{(x_{k+1}-x_k)^2}{x_{k+2}-2x_{k+1}+x_k}.$$

若 x_k 一直不等于真解 $x_\star$，我们有结论

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

换言之，*Aitken* 加速技术可以明显提升收敛速度。

☆ **说明 5.2.** 图文框中的条件 $C<1$ 是非常重要的。若 $C=1$ 时，*Aitken* 方法可能没有明显的加速效率。相应的反例可考虑序列

$$x_k = 1/k$$

的 *Aitken* 加速过程。尽管如此，*Aitken* 方法的近似程度通常还会得到某种程度的改善；见教科书的例子。

论题 5.5. *局部应用 Aitken 加速技术，可形成 Steffensen 迭代法，相应的迭代函数为*

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

该算法的几何解释是，对不动点迭代的残量

$$(x_k, g(x_k) - x_k)^\top$$

进行线性外推。可以证明：在适当的条件下，Steffensen 方法具有局部的平方收敛。

5.2.4 Newton 方法

在非线性方程的数值方法中，Newton-Raphson 方法是著名的高效算法之一。据考证，它由 Newton 在 1669 年给出，用于计算

$$x^3 - 2x - 5 = 0$$

的最大根。当时的基本思想是试根，即依次计算 $x = 2.d_1d_2d_3\cdots$ 的每位数字。在 1690 年，Raphson 以略有修改的方式，重新发表了这个方法。

论题 5.6. *Newton 方法又称切线法，其迭代公式是*

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

其几何含义就是用切线方程局部线性化非线性方程，利用线性问题的根作为更好的逼近。我们要指出：局部线性化是一种常用的想法，在非线性问题中广泛使用。

Newton 迭代方法的收敛速度同真解的局部信息密切相关。下面给出一些著名的结论。

为简单起见，假设 $f(x)$ 充分光滑。若 $f(x)$ 非光滑，Newton 方法的相应推广及其分析是近代数值方法的热门研究方向；因超出课程范围，详略。

定理 5.3. *若 $x_\star$ 是单根，则 Newton 迭代方法是局部二阶收敛的。*

定理 5.4. *若 $x_\star$ 是 m 重根，则 Newton 迭代方法退化为线性收敛，且误差下降的渐近速度为 $1 - m^{-1}$.*

☆ **说明 5.3.** 重根还会导致严重的舍入误差问题。

✎ **论题 5.7.** *对于重根，我们可以利用以下策略提升 Newton 迭代方法的收敛阶或收敛速度：*

1. *若重数 m 是已知的，算法可以简单修正为*

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

2. *若重数 m 是未知的，利用方程 $F(x) = f(x)/f'(x)$ 滤掉重根，建立相应的 Newton 迭代*

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

   为避免二阶导数的计算，我们还可采用 Steffensen 加速算法。

3. *重根次数 m 的自动探测：对于标准 Newton 迭代解，计算相应的重根指标*

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

   当其取值稳定时，它必然趋向于 m；此时，对 $h(x_k)$ 取整，并跳转到算法 1 即可.

在适当的条件下，Newton 迭代方法可以整体收敛。例如，

定理 5.5. *假设函数 $f(x):[a,b]\to\mathbb{R}$ 满足：*

1. *单调保凸；*
2. *$f(a)$ 和 $f(b)$ 异号；*
3. *从两个端点出发的迭代位置依旧落在 $[a,b]$ 上，*

则 Newton 方法对于区间 $[a,b]$ 上的任意初值都是收敛的。

证明是简单的数学分析问题；详见教科书。

❀ **思考 5.1.** *Newton 迭代的重要应用：*

1. *根号 $\sqrt{a}$ 的计算：$x_{k+1}=\frac{1}{2}(x_k+\frac{a}{x_k})$；*
2. *倒数 $1/a$ 的计算：$x_{k+1}=2x_k-ax_k^2$；*

5.2.5 割线法

利用历史计算数据，进行非线性函数 $f(x)$ 的局部线性插值。利用线性函数的相应零点作为更好的迭代逼近，可得算法

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

它也可解释为 Newton 迭代方法的导数近似为相应的一阶差商。

定理 5.6. *一般来说，割线法的收敛速度稍慢于 Newton 法。在适当条件下，可以证明割线法的收敛阶为 1.618.*

☆ **说明 5.4.** 割线法可能出现分母为零的情形，从而算法出现意外停机。此时需要执行迭代重启，更换旧的迭代位置。

5.2.6 实多项式的实根计算

考虑实系数多项式

$$p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0. \tag{5.2.4}$$

假设系数产生微小扰动，形成多项式 $p_\varepsilon(x) = p(x) + \varepsilon q(x)$，其中 $q(x)$ 是 n 次多项式。扰动多项式的根记为 $x_k(\varepsilon)$，简单计算可知

$$x_k'(0) = -\frac{q(x_k)}{p'(x_k)},$$

其中 $x_k = x_k(0)$ 是多项式 $p(x)$ 的某个根。换言之，比值 $1/p'(x_k)$ 刻画了多项式 $p(x) = 0$ 的根在 x_k 处关于扰动的敏感程度；若比值很大，则相应的求根问题是病态的。

通常，高次多项式的求根问题都是病态的，相应的数值求解具有较大的难度。

非线性方程的求解方法，例如 Newton 迭代方法，均可以求解高次多项式的根。此时，我们需要大量计算多项式 $p(x)$ 及其导数 $p'(x)$ 的函数值。因此，多项式的计算复杂度问题急需解决。事实上，它也同舍入误差的控制有关。

✍ **论题 5.8.** *Horner* 算法或秦九韶算法是常用的高效算法。注意到多项式除法运算满足

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

显然 $p(\mu) = b_0$，而 $\{b_i\}_{i=0}^n$ 的计算公式为

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

因为 $p'(\mu) = g(\mu)$，导数值的计算再次转化为多项式的取值。详略。

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

☆ **说明 5.5.** 对于一个实系数多项式 $f(x)$，根的定位是一个非常重要的问题。相关结论有

1. *Lagrange* 法：设 $a_n > 0$，a_{n-k} 为第一个负系数，b 是负系数中的最大绝对值，则 $f(x)$ 的正根上限为 $1 + (b/a_n)^{1/k}$.
2. *Sturm* 序列法：由 $f(x)$ 与 $f'(x)$ 辗转相除得到的序列称为 *Sturm* 序列。这个 *Sturm* 序列在 μ 点的符号变化的次数（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。

具体讨论超出本课程的要求，详略。

☆ **说明 5.6.** 多项式求根可以转化为矩阵特征值问题，在 *Matlab* 中的对应命令是 *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，本课程不做过多讨论。

5.3 方程组的数值求解

粗略地讲，前面介绍的大部分数值方法，均可以由标量方程推广到方程组 $\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}$。但是，我们将面临下面的困难：

1. 关于精确解的数学理论（存在性和唯一性）不清楚；

2. 对于某些适用于标量方程的算法，例如二分法，其基本原理或者设计思想不再成立；

3. 即使适用于标量方程的算法能够推广到方程组，相应的计算效率也将成为非常重要的研究内容。

下面将以 Newton 方法为核心，介绍非线性方程组求解的基本框架和常见方法。

5.3.1 预备知识

本节简要介绍向量值函数 $\boldsymbol{f}(\boldsymbol{x}) = \{f_i(x_1, x_2, \ldots, x_n)\}_{i=1}^m$ 的微积分理论。它可以简单地理解为多元函数到向量值函数的推广。下面仅仅列出同后续内容紧密相关的一些概念和结论。

定义 5.1. *设 $\boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^m$，相应的 Frechet 导数 $\boldsymbol{f}'(\boldsymbol{x})$ 可以精确地定义为*

$$\lim_{\|\triangle \boldsymbol{x}\| \to 0} \frac{\|\boldsymbol{f}(\boldsymbol{x} + \triangle \boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}'(\boldsymbol{x}) \triangle \boldsymbol{x}\|}{\|\triangle \boldsymbol{x}\|} = 0. \tag{5.3.5}$$

若所有偏导数都是连续的，则 Frechet 导数就是 $m \times n$ 阶 Jacobi 矩阵，即

$$\boldsymbol{f}'(\boldsymbol{x}) = D\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} (\boldsymbol{x}). \tag{5.3.6}$$

若 $f(\boldsymbol{x})$ 是标量多元函数，有 $f'(\boldsymbol{x}) = [\nabla f(\boldsymbol{x})]^\top$。

说明 5.7. *同多元函数的主要区别是，即使函数充分光滑，向量值函数也可能不具有微分中值定理。换言之，没有一个局部位置 $\boldsymbol{\xi}$，使*

得

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

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是给定的两个位置。主要原因是每个分量函数 $f_i(\cdot, \ldots, \cdot)$ 通常对应不同的中值。尽管如此，由此造成的分析困难并不可怕，因为函数值差距可以表示为一个积分

$$\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{x}) = \int_0^1 \boldsymbol{f}'(\boldsymbol{x} + s(\boldsymbol{y} - \boldsymbol{x}))\mathrm{d}s \cdot (\boldsymbol{y} - \boldsymbol{x}).$$

它称为 *Jacobi* 矩阵 $\boldsymbol{f}'$ 从 $\boldsymbol{x}$ 到 $\boldsymbol{y}$ 的 *Riemann* 积分，对应于每个分量函数的积分。

✍ **论题 5.10.** 在后续研究中，向量范数和矩阵范数是常用的度量工具。相关的常用不等式有

1. 积分不等式:

$$\left\| \int_0^1 \boldsymbol{f}(t)dt \right\| \leq \int_0^1 \|\boldsymbol{f}(t)\|dt.$$

2. 设 $\boldsymbol{f}$ 在凸集上处处 *Frechet* 可微，且导数 $\boldsymbol{f}'$ 是 *Lip* 连续的，即

$$\|\boldsymbol{f}'(\boldsymbol{y}) - \boldsymbol{f}'(\boldsymbol{x})\| \leq \gamma\|\boldsymbol{y} - \boldsymbol{x}\|, \quad \forall \boldsymbol{y}\, \forall \boldsymbol{x},$$

利用积分表达式，可以建立估计

$$\|\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}'(\boldsymbol{x})(\boldsymbol{y} - \boldsymbol{x})\| \leq \frac{\gamma}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2, \quad \forall \boldsymbol{y}\, \forall \boldsymbol{x}.$$

5.3.2 不动点迭代方法

相关概念和分析方法是类似的。特别地，压缩映像定理依旧是主要分析工具。具体内容不再赘述。

5.3.3 Newton 方法

同标量方程的 Newton 方法表述一样，相应的算法是

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

换言之，每步迭代都要求解一个线性方程组。

Newton 方法的收敛研究有着悠久的历史。在 1829 年，Cauchy 研究了 $n = 1$ 时的局部收敛性，直到 1899 年，Runge 给出了 $n \geq 2$ 时的局部收敛性。在 1916 年，Fine 给出了半局部收敛分析。著名工作还有 Ostrowski(1936),、Willers(1938) 和 Kantovich(1948) 给出的证明。

定理 5.7. 【局部收敛分析】若 $\boldsymbol{f}'(\boldsymbol{x})$ 在真解 $\boldsymbol{x}_\star$ 附近是连续的非奇异矩阵，则 *Newton* 方法是局部超线性收敛。

若进一步假设 $\boldsymbol{f}'$ 在真解 $\boldsymbol{x}_\star$ 附近是 *Lipschitz* 连续的，则 *Newton* 方法至少是局部二阶收敛的。

☆ **说明 5.8.** 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，在 *Newton* 方法中，相邻误差可以估算迭代误差。

请注意，上述逆命题不成立，如奇偶子列分别定义为

$$\boldsymbol{x}_{2k+1} = \frac{2}{(2k)!}, \quad \boldsymbol{x}_{2k} = \frac{1}{(2k)!}.$$

它们虽然满足上述极限关系，但却不是超线性收敛到零。

定理 5.8. 【半收敛分析】: 设 $\boldsymbol{f}(\boldsymbol{x})$ 在开凸集 D_0 内处处 *Frechet* 可微，且满足如下性质：

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

若 $\boldsymbol{x}_0\in D_0$ 且 $h=\alpha\beta\gamma/2<1$，则 *Newton* 迭代序列至少二阶收敛到某个真解 $\boldsymbol{x}_\star\in S_r(\boldsymbol{x}_0)$，其中 $r=\alpha/(1-h)$。

✯ **说明 5.9.** 初始向量 $\boldsymbol{x}_0$ 的选取是难点之一。由半收敛分析的证明过程可知，收敛条件成立的一个必要条件是

$$\|\boldsymbol{x}_2-\boldsymbol{x}_1\|<\|\boldsymbol{x}_1-\boldsymbol{x}_0\|. \tag{5.3.7}$$

通常，(5.3.7) 会保证 $\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_k\|$ 的下降。若在初始两步的计算中发现 (5.3.7) 不成立，则应当主动放弃这个没有前途的初值。

✯ **说明 5.10.** 由于 $\boldsymbol{x}_{k+1}$ 仅仅依赖 $\boldsymbol{x}_k$，故而 *Newton* 方法也常常被称为自校正算法，相应的计算结果不受历史记录的影响。

✯ **说明 5.11.** 当迭代矩阵接近奇异时，通常采用基于 *Tiknonov* 正则化方法的修正算法：

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

其中 $\lambda_k>0$ 也称为阻尼因子。

✯ **说明 5.12.** 利用 *Newton* 位移的逐次缩减，不断检测向量值函数的长度，可以实现所谓的“盲人下山法”。算法的基本描述如下：

1. $\boldsymbol{f}'(\boldsymbol{x}_k)\Delta\boldsymbol{x}_k = -\boldsymbol{f}(\boldsymbol{x}_k)$；令 $\ell = 0$ 和 $\lambda_\ell = 1$;
2. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda_\ell \Delta\boldsymbol{x}_k$;
3. 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，则重启这个算法，再次（随机）选择新的迭代初值；否则，回到第 *3* 步；
4. 回到第一步，开始 *Newton* 迭代的下一步；

上述操作可以更好地找到某个真解的大概位置。而后，期待 *Newton* 迭代方法可以给出快速的平方收敛。

5.3.4 Newton 方法的简单改进

Newton 方法的计算效率很差。每次迭代都要计算 n^2 个导数值和 n 个函数值，还要求解一个 n 阶线性方程组。下面介绍 Newton 方法的简单改进，特别是线性方程组的构造过程。

修正 Newton 法

它也称为沙文基思方法，主要手段是局部锁定 Jacobi 矩阵，减少这个方面的计算时间。算法的基本描述是

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

对于局部同型的线性方程组，LU 分解只需执行一次，线性方程组的构造过程和求解效率均得到明显的提高。

在适当的假设下，可证修正 Newton 法具有 $m+1$ 阶收敛速度。在实际应用中，较为常用的是 $m=2$ 的情形。

割线法

割线法利用历史数据近似导数值，改善了 Jacobi 矩阵的计算效率。依据构造思想和实现方法，割线法主要包含两大类实现方法：

论题 5.11. 离散的 *Newton* 方法：直接用差商矩阵 $\mathbb{J}(\boldsymbol{x}_k, \boldsymbol{h}_k)$ 代替迭代矩阵，可得如下算法

$$\boxed{\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \Big[\mathbb{J}(\boldsymbol{x}_k, \boldsymbol{h}_k)\Big]^{-1} \boldsymbol{f}(\boldsymbol{x}_k),}$$

其中 $\boldsymbol{h}_k$ 是趋于零的参数矩阵。通常，每个元素设置为

$$(\boldsymbol{h}_k)_{ij} = \bar{h}_{ij} \|\boldsymbol{f}(\boldsymbol{x}_k)\|,$$

其中 $\bar{h}_{ij}$ 是事先给定的参数。为简单起见，通常要求 $\bar{h}_{ij} = \bar{h}_i$。

论题 5.12. 或者利用非线性方程的局部线性插值思想，或者利用曲面的局部平面化思想，非线性问题可以局部地近似为一个线性问题。将线性方程组的解视为非线性问题真解的更好近似，可得**割线算法**：

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

其中 $\mathbf{H}_k$ 和 $\mathbf{\Gamma}_k$ 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

$$\mathbf{H}_k = [\boldsymbol{x}_{k,1} - \boldsymbol{x}_{k,0}, \cdots, \boldsymbol{x}_{k,n} - \boldsymbol{x}_{k,0}], \tag{5.3.8a}$$

$$\mathbf{\Gamma}_k = [\boldsymbol{f}_{k,1} - \boldsymbol{f}_{k,0}, \cdots, \boldsymbol{f}_{k,n} - \boldsymbol{f}_{k,0}], \tag{5.3.8b}$$

为保证算法的可操作性（或者矩阵 $\mathbf{H}_k$ 和 $\mathbf{\Gamma}_k$ 均可逆），$n+1$ 个辅助点要处于一般位置，即它们不能共处一个平面。否则，算法需要重启。

恒假定 $\boldsymbol{x}_{k,0} = \boldsymbol{x}_k$ 是当前位置，其它辅助点由已知的周边信息提供。依据辅助点的选取策略，割线方法有两种实现过程。

1. 两点序列割线法：

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_k + \{(\boldsymbol{x}_{k-1})_\ell - (\boldsymbol{x}_k)_\ell\}\boldsymbol{e}_\ell, \tag{5.3.9a}$$

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_k + \sum_{j=1}^{\ell}\{(\boldsymbol{x}_{k-1})_j - (\boldsymbol{x}_k)_j\}\boldsymbol{e}_j, \tag{5.3.9b}$$

其中 $(\boldsymbol{x}_k)_\ell$ 表示 $\boldsymbol{x}_k$ 的第 ℓ 个分量，$\ell = 1:n$。在每次迭代中，前者计算 n^2+n 个函数值，而后者仅仅计算 n^2 个函数值。

2. $(n+1)$ 点序列割线法：

$$\boldsymbol{x}_{k,\ell} = \boldsymbol{x}_{k-\ell}, \quad \ell = 1:n. \tag{5.3.10}$$

每次迭代仅需计算 n 个函数值，其它都是已知的。

在适当的条件下，可证 p 点序列割线法的收敛阶为

$$\lambda^p = \lambda^{p-1} + 1$$

的最大正根。随着 p 的增加，最大正根逐渐单增到 2。

☆ **说明 5.13.** 虽然收敛阶占优，但是 $(n+1)$ 点序列割线法容易产生数值不稳定现象。

☆ **说明 5.14.** 从高到低，算法效率的排序是：$(n+1)$ 点序列割线法、两点序列割线法、离散 *Newton* 方法。

5.3.5 拟 Newton 方法

拟 Newton 方法产生于上世纪 60 年代，有效继承了 $(n+1)$ 点序列割线法的优点。首先，它无需计算导数值，改善了线性方程组的构造效率。同时，它提高了线性方程组的求解效率。当然，同割线法相比，拟 Newton 方法的收敛阶有所降低。

基本思想

拟 Newton 方法可视为割线法的推广。其基本结构依旧是：

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \mathbb{B}_k^{-1}\boldsymbol{f}(\boldsymbol{x}_k), \tag{5.3.11}$$

若当前位置 $\boldsymbol{x}_k$ 和迭代矩阵 $\mathbb{B}_k$ 是已知的，则公式 (5.3.11) 可以给出后续位置 $\boldsymbol{x}_{k+1}$。此时，能否利用相邻两步的计算信息，简单地构造出后续的迭代矩阵 $\mathbb{B}_{k+1}$?

在割线法中，迭代矩阵具有线性结构 $\boldsymbol{\Gamma}_k = \mathbb{A}_k\mathbf{H}_k$。我们希望这个结构能够有所继承，故而不妨要求 $\mathbb{B}_{k+1}$ 满足拟 Newton 方程

$$\boxed{\Delta\boldsymbol{f}_k = \mathbb{B}_{k+1}\Delta\boldsymbol{x}_k,}$$

其中 $\Delta\boldsymbol{f}_k = \boldsymbol{f}_{k+1} - \boldsymbol{f}_k := \boldsymbol{f}(\boldsymbol{u}_{k+1}) - \boldsymbol{f}(\boldsymbol{u}_k)$ 和 $\Delta\boldsymbol{x}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k$ 是当前位置的两个改变量。

拟 Newton 方程有 n 个约束条件，却有 n^2 个未知数。因此，当 $n > 1$ 时它有无穷多解。为此，我们需要提出适当的约束条件，给出构造 $\mathbb{B}_{k+1}$ 的明确途径。

不妨从计算效率入手。若 $\mathbb{B}_{k+1}$ 同 $\mathbb{B}_k$ 具有相似性，线性方程组求解（或逆矩阵 $\mathbb{B}_{k+1}^{-1}$ 的计算）效率可以期待某种程度的提升。回忆 $(n+1)$ 点序列割线法，其迭代矩阵满足性质

$$\mathbb{A}_k \Delta \boldsymbol{x}_j = \Delta \boldsymbol{f}_j, \quad j = k-1, k-2, \dots, k-n; \tag{5.3.12a}$$

$$\mathbb{A}_{k+1} \Delta \boldsymbol{x}_j = \Delta \boldsymbol{f}_j, \quad j = k, k-1, \dots, k-n+1. \tag{5.3.12b}$$

这些条件蕴含着

$$(\mathbb{A}_{k+1} - \mathbb{A}_k) \underbrace{[\Delta \boldsymbol{x}_{k-1}, \dots, \Delta \boldsymbol{x}_{k-n+1}]}_{\mathbf{H}_k} = [0, 0, \dots, 0]. \tag{5.3.13}$$

由于矩阵 $\mathbf{H}_k$ 是列满秩的，因此 $\mathbb{A}_{k+1}$ 是 $\mathbb{A}_k$ 的秩一修正。受此启发，我们不妨假设迭代矩阵 $\mathbb{B}_{k+1}$ 是 $\mathbb{B}_k$ 的秩一修正，即

$$\mathbb{B}_{k+1} = \mathbb{B}_k + \boldsymbol{u}_k \boldsymbol{v}_k^\top, \tag{5.3.14}$$

其中 $\boldsymbol{u}_k$ 是待定的方向，$\boldsymbol{v}_k$ 是给定的方向。相应的拟 Newton 方法 (5.3.11) 就是著名的 Broyden (1965) 方法。

定理 5.9. *由 (5.3.14) 可知，$\mathbb{B}_{k+1}$ 和 $\mathbb{B}_k$ 作用在 $span^\perp(\boldsymbol{v}_k)$ 的像是一样的。若 $\mathbb{B}_{k+1}$ 还满足拟 Newton 方程，则应取*

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

此时，相应的 $\mathbb{B}_{k+1}$ 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 5.13. *记 $\mathbb{H}_k$ 是 $\mathbb{B}_k$ 的逆矩阵。利用 Sherman-Morrison 公式，迭代矩阵的逐步修正公式是*

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

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求解，一直保持 $\mathbb{H}_k$ 的对称性。

将上述公式融合到拟 *Newton* 迭代中，为获得新的迭代位置，每次迭代只需 $\mathcal{O}(n^2)$ 次乘除法运算。

✯ **说明 5.15.** 拟 *Newton* 方法的收敛阶不如 *Newton* 方法高。可以在适当的条件下，依旧可证 *Broyden* 方法具有局部的超线性收敛。

✯ **说明 5.16.** 假设方法 (5.3.11) 是收敛的。为保证其超线性收敛，迭代矩阵 $\mathbb{B}_k$ 不必趋向 *Jacobi* 矩阵 $\boldsymbol{f}'(\boldsymbol{x}_\star)$，只需满足较弱的充要条件

$$\lim_{k\to\infty} \frac{\|\big[\mathbb{B}_k - \boldsymbol{f}'(\boldsymbol{x}_\star)\big]\Delta \boldsymbol{x}_k\|}{\|\Delta \boldsymbol{x}_k\|} = 0,$$

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，而 $\boldsymbol{s}_k^{\mathrm{Nt}}$ 是原始 *Newton* 迭代方法的修正方向。

❀ **思考 5.2.** 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？

假定 $\mathbb{B}_{k+1}$ 是 $\mathbb{B}_k$ 的秩二修正矩阵，也可构造出相应的拟 Newton 方法。著名的算法有 DFP 方法和 BFS 方法，略。

5.3.6 极值算法

对于光滑函数 $\boldsymbol{f}(\boldsymbol{x})$ 而言，方程组 $\boldsymbol{f}(\boldsymbol{x}) = 0$ 和最优化问题

$$\boldsymbol{x}_\star = \arg\min_{\forall \boldsymbol{x}} \|\boldsymbol{f}(\boldsymbol{x})\|_2^2$$

是等价的。此时，各种优化算法（例如最速下降方法等）都是有效的计算方法。因篇幅有限，详略。

5.3.7 延拓方法

略。

第 6 章

数值实验

数值实验将主要用到下面两个矩阵。它们均对应椭圆方程的有限差分离散过程。

三对角矩阵：考虑两点边值问题

$$-u''(x) = g(x), \quad x \in (0,1),$$

相应的边界条件是 $u(0) = u(1) = 0$。在网格点 $\{x_i = ih\}_{i=1:n}$ 处，利用差商代替导数，可得差分方程

$$-u_{i-1} + 2u_i - u_{i+1} = h^2 g(ih), \quad i = 1:n,$$

其中 $h = 1/(n+1)$ 为网格步长，u_i 是 $u(ih)$ 的数值近似。注意到零边值条件，全体差分方程可以汇总为如下的线性方程组

$$\mathbb{T}_n \boldsymbol{x} = \boldsymbol{b}_n, \tag{6.0.1}$$

其中 $\boldsymbol{x} = (u_1, u_2, \ldots, u_n)^\top$，系数矩阵是 n 阶三对角对称阵

$$\mathbb{T}_n = \mathrm{tridiag}(-1, 2, -1) = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}. \tag{6.0.2}$$

块三对角矩阵：考虑二维 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

$$u(x,y)=0,\quad x=0,1 \text{ 或者 } y=0,1.$$

在网格点 $\{(x_i,y_j)=(ih,jh)\}_{i=1:n}^{j=1:n}$ 处，利用不同方向的差商代替相应方向的二阶偏导数，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 $u(ih,jh)$ 的数值近似。注意到零边值条件，全体差分方程可以汇总为如下的线性方程组

$$\mathbb{A}_n\boldsymbol{x}=\boldsymbol{b}_n, \tag{6.0.3}$$

其中 $\boldsymbol{x}$ 是数值解 $\{u_{ij}\}_{i=1:n}^{j=1:n}$ 从下到上、从左到右、逐行排序而成的向量，系数矩阵 $\mathbb{A}_n$ 是 n^2 阶对称正定矩阵

$$\begin{aligned}\mathbb{A}_n&=\begin{bmatrix}\mathbb{T}_n+2\mathbb{I}_n & -\mathbb{I}_n & & \\ -\mathbb{I}_n & \mathbb{T}_n+2\mathbb{I}_n & & \\ & \ddots & \ddots & -\mathbb{I}_n \\ & & -\mathbb{I}_n & \mathbb{T}_n+2\mathbb{I}_n\end{bmatrix}\\ &=\mathbb{T}_n\otimes\mathbb{I}_n+\mathbb{I}_n\otimes\mathbb{T}_n.\end{aligned} \tag{6.0.4}$$

这里的 $\mathbb{I}_n$ 是 n 阶单位矩阵，$\mathbb{T}_n$ 是由 (6.0.2) 给出的三对角矩阵，$\otimes$ 是矩阵的 Kronecker 乘积。

✯ **说明 6.1.** 请将数值实验作业电邮到 *qzh_nk@aliyun.com*，请给出报告的 *PDF* 文档。其他材料可作为附件上传。

✯ **说明 6.2.** 数值实验作业要包含以下内容：题目（摘要）、前言（目的和意义）、数学原理、程序的设计流程、实验结果结论和讨论、以及小结等。

6.1 线性方程组的直接解法

❖ **练习 6.1.1.** 利用高斯消元法和 $LDL^\top$ 算法，求解线性方程组 (6.0.3)。设真解是 $\boldsymbol{x}_\star = (1,1,1,\ldots,1)^\top$，右端向量 $\boldsymbol{b}_n$ 由真解给出。

1. 绘制数值误差同参数 n 的关系，其中数值误差采用对数坐标;

2. 绘制 *CPU* 时间同参数 n 的关系;

3. 绘制矩阵条件数与参数 n 的关系。请问：摄动理论给出的舍入误差估计 (1.5.18) 是否完美地刻画了相对误差的大小?

❖ **练习 6.1.2.** 矩阵非零元素分布可以影响数值计算的效率，最优答案的确定是个 *NP* 问题。考虑行列重排后相等的两个矩阵

$$\mathbb{B}_1 = \begin{bmatrix} 1 & & & & a \\ & 1 & & & a \\ & & \ddots & & \vdots \\ & & & 1 & a \\ a & a & \cdots & a & 1 \end{bmatrix}, \quad \mathbb{B}_2 = \begin{bmatrix} 1 & a & \cdots & a & a \\ a & 1 & & & \\ \vdots & & \ddots & & \\ a & & & 1 & \\ a & & & & 1 \end{bmatrix}, \tag{6.1.5}$$

比较三角分解后的非零元素分布。在 *Matlab* 中观测矩阵结构图的命令是 *spy()*。

利用这个矩阵的元素分布特点，修改 *Crout* 算法，使其可以省掉那些无用的零运算时间。观察 *CPU* 时间是否有所节省?

❖ **练习 6.1.3.** 计算三对角阵 $\mathbb{T}_n$ 的逆矩阵。

❖ **练习 6.1.4.** 设 $\mathbb{D}_n = diag\{2^{-i}\}_{i=1}^n$。利用追赶法，分别求解两个等价的三对角线性方程组

$$\mathbb{D}_n\mathbb{T}_n\boldsymbol{x} = \boldsymbol{b}_n, \quad \mathbb{T}_n\boldsymbol{x} = \mathbb{D}_n^{-1}\boldsymbol{b}_n,$$

其中 n 从 500 变换到 2000。随机选取真解 $x_\star$，观测数值误差同 n 的关系，以及两个计算结果有何区别?

❖ **练习 6.1.5.** 随机构造 n 阶可逆矩阵 $\mathbb{A}$，进行相应的列主元高斯消元。收集大量数据，观测主元增长因子 $\eta(\mathbb{A})$ 是否处于 $n^{2/3}$ 或 $n^{1/2}$ 的量级?

6.2 线性方程组的迭代解法

设线性方程组 (6.0.3) 的真解是 $\boldsymbol{x}_\star = (1, 1, \ldots, 1)^\top$，右端向量由真解给出。均取初始向量 $\boldsymbol{x}_0 = 0$，用户指标为 $\mathcal{E} = 10^{-6}$, 度量是欧氏范数或无穷范数。

❖ **练习 6.2.1.** 分别采用残量、相邻差量和后验误差作为停机标准，比较 J 方法和 GS 方法停机时的迭代次数和真实误差。

❖ **练习 6.2.2.** 以真实误差作为停机标准，数值观察 SOR 方法的松弛因子 ω 对于迭代次数的影响，并找到相应的最佳迭代因子。

❖ **练习 6.2.3.** 考虑 J 方法、GS 方法和(带有最佳松弛因子的) SOR 方法，进行下面的数值观察:

1. 绘制相应的误差曲线和残量曲线，均采用半对数坐标系;

2. 以真实误差为停机标准，绘图指出迭代次数同矩阵阶数的关系。

❖ **练习 6.2.4.** 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 的影响。

❖ **练习 6.2.5.** 取循环指标 $m = 5$，半迭代加速 J 方法；绘制相应的误差曲线和残量曲线。其他的 m 呢?

❖ **练习 6.2.6.** 执行 *CG* 方法，进行下面的数值观察:

1. 取 $n = 30$ 和 $n = 31$ 两种奇偶状态，绘制相应的误差曲线和残量曲线;

2. 绘图指出迭代次数同矩阵阶数的关系，比较这个关系同 *SOR* 方法的差异。

❖ **练习 6.2.7.** 采用 *SSOR* 做为预处理因子, 编制相应的预处理 *CG* 算法。随机取定矩阵阶数，考察迭代次数同参数 ω 的关系。

6.3 线性最小二乘问题

❖ **练习 6.3.1.** 随机构造 20 30 阶的可逆方阵，分别利用

(a)CGS 方法; *(b)MGS* 方法; *(c)Householder* 法; *(d)Givens* 法,

给出相应的 *QR* 分解。比较上述四种方法关于列向量的正交性、*CPU* 时间和向后稳定性表现。

❖ **练习 6.3.2.** 考虑列满秩的最小二乘问题

$$U_{n\times n}\mathbb{A}_{n\times(n-1)}\boldsymbol{x}_{n-1} = U_{n\times n}\boldsymbol{b}_n,$$

其中 $U_{n\times n}$ 是有限（10 20）个随机生成的 *Given* 阵乘积,

$$\mathbb{A} = \mathbb{T}_n(1:n, 1:n-1) = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \\ & & & & -1 \end{bmatrix},$$

右端项是

$$\boldsymbol{b}=\left(1+\frac{1}{n},\frac{2}{n},\frac{3}{n},\ldots,\frac{n-2}{n},1+\frac{n-1}{n},0\right)^{\top}.$$

它具有唯一解 $\boldsymbol{x}_{LS}=(1,1,1,\ldots,1,1)^{\top}$。

令 n 从 *100* 逐渐增加到 *500*，分别利用下面四种方法求解：

(a) 法方程组；*(b)MGS* 方法；*(c)Householder* 法；*(d)Givens* 法。

比较计算工作量（可用 *CPU* 时间表示）和数值计算精度。

❖ **练习 6.3.3.** 实现本章的两张图。

6.4 矩阵特征值问题

除了最后一题，均考虑对称三对角矩阵 $\mathbb{T}_n$ 的特征值问题，相应的阶数分别是偶数 $n=100$ 和奇数 $n=101$。要求计算结果精确到小数点后第 6 位。

❖ **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. 绘制主特征值的误差曲线，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，绘制相应的曲线。

❖ **练习 6.4.2.** 用反幂法求解离 $q=2$ 最近的特征值，并给出相应的特征向量。绘制特征值误差曲线，数值观察反幂法是否具有“一次迭代”特性？当 n 越来越大的时候呢？

❖ **练习 6.4.3.** 用幂法和降维技巧，求解第二主特征值及其特征向量；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。

❖ **练习 6.4.4.** 用古典 *Jacobi* 方法、循环 *Jacobi* 方法和阈值 *Jacobi* 方法，求解全部特征值，并绘制相应的收敛过程。

❖ **练习 6.4.5.** 用二分法求解位于开区间 $(1,2)$ 内的特征值；考虑带原点位移的反幂法，改善数值结果的精度。

❖ **练习 6.4.6.** 用对称隐式 QR 方法，求解全部特征值。

❖ **练习 6.4.7.** 阈值 *Jacobi* 方法具有求解小特征值的优势。考虑对称正定矩阵

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

直接计算可知其特征值为 $10^{40}, 9.9 \times 10^{19}$ 和 9.81818×10^{-1}。请用阈值 *Jacobi* 方法求解三个特征值。利用 *Matlab* 的 *eig()* 计算特征值，比较两种方法给出的数值结果。

6.5 非线性方程（组）的数值方法

停机标准均取为 $\mathcal{E} = 10^{-6}$。

❖ **练习 6.5.1.** 用 *Newton* 方法，求多项式 $x^3 - x^2 - 8x + 12 = 0$ 的根，绘制相应的误差曲线，给出相应的数值收敛阶。

❖ **练习 6.5.2.** 用割线法，重复前一题的工作。

❖ **练习 6.5.3.** 考虑非线性方程组

$$\begin{cases} (x+3)(y^2-7)+18=0, \\ \sin(ye^x-1)=0, \end{cases} \tag{6.5.6}$$

取初值 $(-0.15, 1.4)$，比较 *Newton* 方法和 *Broyden* 方法的误差曲线和迭代次数；若初值为 $(0, 1)$ 呢?

❖ **练习 6.5.4.** 考虑非线性方程组

$$\begin{cases} x + y - 3 = 0, \\ x^2 + y^2 - 9 = 0. \end{cases} \tag{6.5.7}$$

取初值为 $(2, 4)$，$\mathbb{B}_0$ 为该点的 *Jacobi* 矩阵。观察 *Broyden* 方法的 $\mathbb{B}_k$ 是否收敛到相应的 *Jacobi* 矩阵?

❖ **练习 6.5.5.** 仍考虑非线性方程组 (6.5.7)。分别用修正 *Newton* 法（采用不同的 m）、离散 *Newton* 法和两点序列割线法求解，绘制相应的误差曲线。

❖ **练习 6.5.6.** 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，观察数值结果同幂法的区别。

第 7 章
附录 1: 常微分方程的数值方法

常微分方程（组）具有相当广泛的应用，相应的数值解法是非常重要的研究课题。本章重点考虑一阶常微分方程初值问题

$$\begin{cases} y' = f(t,y), \quad a < t \le b, \\ y(a) = c, \end{cases} \tag{7.0.1}$$

其中 $f(t,y) : [a,b] \times \mathbb{R} \to \mathbb{R}$ 是已知的连续函数，且关于 y 具有 Lipschitz 连续性。此时，问题具有唯一的真解 $y = y(t)$。

7.1 单步法

7.1.1 基本概念

在 $[a,b]$ 上选取一组离散点

$$a = t_0 < t_1 < t_2 < \cdots < t_n < \cdots < t_{N-1} < t_N = b,$$

其中 $h_n = t_{n+1} - t_n$ 称为时间步长。在离散时刻 t_n 处，真解记为 $y(t_n)$，数值解记为 y_n。

通常，取 $h_n = h$，即离散点是等距分布的。当 h 趋于零时，离散点的个数趋于无穷。

✎ **论题 7.1.** 格式设计的关键是导数的离散处理，它主要有三种构造方法：

1. 差商代替导数；
2. Taylor 公式；
3. 数值积分。

利用上述三种构造方法，建立相应的 Euler 方法。

Euler 方法属于单步方法。通常，单步方法可以写为

$$y_{n+1} = y_n + h\Phi(t_n, y_n, y_{n+1}, h), \tag{7.1.2}$$

其中 $\Phi(\cdot)$ 称为增量函数。若增量函数同 y_{n+1} 无关，则单步方法是显式的；若增量函数同 y_{n+1} 相关，则单步方法是隐式的。

7.1.2 Euler 方法及其误差分析

定义 7.1. 单步方法 (7.1.2) 的局部离散误差是

$$R_n = y(t_{n+1}) - y(t_n) - h\Phi(t_n, y(t_n), y(t_{n+1}), h),$$

其中 $y(\cdot)$ 是常微分方程的真解。若对任意的 n 均有 $R_n = \mathcal{O}(h^{p+1})$，则称单步方法是 p 阶的。

利用 Taylor 公式，简单计算可知 Euler 方法是一阶的。利用局部离散误差和数学归纳法，可以证明

定理 7.1. *Euler 方法的整体离散误差是一阶的，即*

$$\max_{0\le n\le N} |y(t_n) - y_n| = \mathcal{O}(h).$$

说明 7.1. 请注意，此处的局部离散误差定义略有不同。它异于有限差分方法的局部离散误差定义。

7.1.3 改进的 Euler 方法

利用数值积分的梯形方法，可得隐式的计算公式

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_{n+1}, y_{n+1})]. \tag{7.1.3}$$

它可以采用简单迭代方法求解。可以证明，在适当的条件下，迭代序列是收敛的。

若仅仅执行两次迭代，可得改进的 Euler 方法

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))]. \tag{7.1.4}$$

它是一个显式方法。

7.1.4 Runge-Kutta 方法

RK 方法是由 19 世纪末德国科学家 C. Runge 和 M. W. Kutta 最早提出的。它可以避免高阶导数的计算，相应的一般结构是

$$y_{n+1} = y_n + h\sum_{j=1}^{s} c_s K_s, \tag{7.1.5}$$

其中 s 称为方法的级数，

$$K_i = f(t_n + a_i h, y_n + h\sum_{j=1}^{s} b_{ij} K_j), \quad i = 1 : s, \tag{7.1.6}$$

是需要计算的中间值。通常，RK 方法描述为如下的 Butcher 表格：

$$\begin{array}{c|cccc} a_1 & b_{11} & b_{12} & \cdots & b_{1s} \\ a_2 & b_{21} & b_{22} & \cdots & b_{2s} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ a_s & b_{s1} & b_{s2} & \cdots & b_{ss} \\ \hline & c_1 & c_2 & \cdots & c_s \end{array}$$

若当 $j \geq i$ 时恒有 $b_{ij} = 0$，则称 RK 方法是显式的；否则，称 RK 方法是隐式的。

二阶 RK 方法

令增量函数是 $\phi(t, y, h) = c_1 K_1 + c_2 K_2$，其中

$$K_1 = f(t, y), \quad K_2 = f(t + a_2 h, y + b_{21} h K_1).$$

为确定参数 c_1, c_2, a_2 和 b_{21}，我们进行 Taylor 展开，并比较两端的展开项系数，可得

$$c_1 = \frac{2a_2 - 1}{2a_2}, \quad c_2 = \frac{1}{1} 2a_2,$$

其中 a_2 为自由参数。例如，当 $a_2 = 1$ 时，有 $c_1 = c_2 = \frac{1}{2}$，相应的方法就是改进的 Euler 方法。特别地，当 $a_2 = \frac{1}{2}$ 时，有 $c_1 = 0$ 且 $c_2 = 1$，可得**二阶变形 RK 方法**

$$y_{n+1} = y_n + hK_2,$$

其中

$$K_1 = f(t_n, y_n), \quad K_2 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_1).$$

由于数据 K_2 可以覆盖掉数据 K_1，该方法具有一定的数据存储优势。

三阶 RK 方法

略。

四阶 RK 方法

它也称为经典 RK 方法，具体公式是

$$y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4),$$

其中

$$
\begin{aligned}
K_1 &= f(t_n, y_n),\\
K_2 &= f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_1),\\
K_3 &= f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_2),\\
K_4 &= f(t_n + h, y_n + hK_3).
\end{aligned}
$$

它同 Simpson 数值积分公式密切相关。

❀ **思考 7.1.** 请写出上述 *RK* 方法的 *Butcher* 表格。

7.1.5 自适应 Runge-Kutta 方法

✎ **论题 7.2.** 利用三阶 *RK* 方法估计二阶变形 *RK* 方法的数值误差，进而给出时间步长的调整方案。

7.1.6 Richardson 方法

✎ **论题 7.3.** 设时间步长是 τ，位于离散时刻 $t_j = j\tau$ 的数值解记为 y_j^τ。建立单步法的误差渐近展开公式

$$y_j^\tau - y(j\tau) = d_p\tau^p + d_{p+1}\tau^{p+1} + \cdots,$$

其中 d_p 和 d_{p+1} 是非零的固定常数。

此时，y_n^h 和 $y_{2n}^{h/2}$ 的适当组合可以给出 $p+1$ 阶的数值解，即

$$\frac{y_n^h - 2^p y_{2n}^{h/2}}{1 - 2^p} = y(nh) + \mathcal{O}(h^{p+1}).$$

这就是 Richardson 方法。

7.2 单步法的相容性、收敛性和稳定性

7.2.1 相容性

注意到 $t_{n+1} = t_n + h$，不妨将增量函数记为 $\Phi(t, y, h)$。

定义 7.2. 若 $\Phi(t, y, 0) = f(t, y)$，则称单步法是相容的。

定理 7.2. 假设 $\Phi(t, y, h)$ 关于 h 联系。若单步法是相容的，则它至少一阶。

7.2.2 收敛性

定义 7.3. 若对任意时刻 $t_\star \in [a, b]$，均有

$$\lim_{h \to 0, t_n \to t_\star} y_n = y(t_\star),$$

则称单步法是收敛的。

定理 7.3. 若 $\Phi(t, y, h)$ 关于每个变量都是 *Lipschitz* 连续，则单步方法收敛的充要条件是相容性成立。

定理 7.4. 在前一个定理的条件下，p 阶单步法的整体误差是 $\mathcal{O}(h^p)$。

7.2.3 稳定性

定义 7.4. 若后续时刻的数值解扰动均被初始时刻的扰动所整体控制，则称单步法是稳定的。有时，它也称为渐近稳定性、古典稳定性、*D*-稳定性。

定理 7.5. 若 $\Phi(t, y, h)$ 关于所有变量是连续的，且关于 y 是 *Lipschitz* 连续的，则单步方法是稳定的。

定义 7.5. 若后续时刻的数值解扰动规模均不超过初始时刻的扰动规模，则称单步法是绝对稳定的。

论题 7.4. 采用模型方程 $y' = \mu y$，计算上述算法的绝对稳定区间（或区域），其中 λ 是给定的常数。

7.3 多步法

设 k 是给定的正整数，相应的线性 k 步法可以写为

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f(t_{n+j}, y_{n+j}), \tag{7.3.7}$$

其中 α_j 和 β_j 均为常数，且 $\alpha_k \neq 0$，α_0 和 β_0 不同时为零。当 $k = 1$ 时，它退化为前面的单步法。

7.3.1 Adams 方法

Adams 方法基于恒等式

$$y(t_n + h) - y(t_n) = \int_{t^n}^{t^{n+1}} f(s, y(s))ds$$

的积分近似计算。若采用多项式外插逼近，则可得显式 Adams 方法；若采用多项式内插逼近，则可得隐式 Adams 方法。

7.3.2 预测校正方法

显式 Adams 方法和隐式 Adams 方法的综合运用。

7.3.3 Hamming 方法

设 p 是一个正整数，考虑恒等式

$$y(t_n+h)-y(t_n-ph)=\int_{t^{n-p}}^{t^{n+1}} f(s,y(s))ds$$

进行多项式外插逼近即可。

7.4 线性多步法的相容性、收敛性和稳定性

7.4.1 预备知识

✍ **论题 7.5.** 线性差分方程的基本理论

1. 齐次差分方程的通解结构；
2. 非齐次差分方程的通解可以表示为齐次差分方程的通解与非齐次差分方程的某个特解之和；
3. 利用线性叠加原理，给出非齐次差分方程的一个特解；
4. 利用特征方程，给出常系数齐次线性差分方程的通解。

7.4.2 相容性

定理 7.6. 线性 k 步法 (7.3.7) 相容的充要条件是

$$\rho(1)=0,\quad \rho'(1)=\sigma(1), \tag{7.4.8}$$

其中

$$\rho(\lambda)=a_k\lambda^k+\cdots+\alpha_1\lambda+\alpha_0,\quad \sigma(\lambda)=\beta_k\lambda^k+\cdots+\beta_1\lambda+\beta_0,$$

是对应左右两侧系数的特征多项式。

7.4.3 收敛性

定义 7.6. 类似于单步法的收敛性，不再赘述。

定理 7.7. 若多步法收敛，则其必相容。

证明：利用多步法求解某些特殊问题，通过收敛性结论直接验证 (7.4.8) 成立即可。 □

7.4.4 稳定性

定义 7.7. 类似于单步法的稳定性，不再赘述。

定理 7.8. 线性多步方法稳定的充要条件是特征根条件，即特征多项式 $\rho(\lambda)$ 的根均在单位圆中，且单位圆周上只有单根。

定理 7.9. 若线性多步方法收敛，则必稳定。

定理 7.10. 若线性多步方法相容且稳定，则必收敛。

定义 7.8. 弱根条件就是特征根条件。强根条件是指特征多项式 $\rho(\lambda)$ 的根除 $\lambda = 1$ 外均在单位圆内。

同两个根条件相对应，稳定性也分别称为弱稳定性和强稳定性。

定义 7.9. 考虑模型方程 $y' = \mu y$ 的线性多步法，它给出一个线性差分方程。若特征方程

$$\rho(\lambda) - \mu h\sigma(\lambda) = 0 \tag{7.4.9}$$

的根都落在单位圆内，则称算法关于 μh 绝对稳定。相应的取值范围称为绝对稳定区间。

定理 7.11. 设线性多步法 (7.3.7) 是 p 阶的，则特征方程 (7.4.9) 必有一个根满足 $\xi(\mu h) = e^{\mu h} + \mathcal{O}(h^{p+1})$。

☆ **说明 7.2.** 多步法存在主根和寄生根。

✍ **论题 7.6.** *Adams* 内插法的绝对稳定区间比 *Adams* 外插法的绝对稳定区间大很多。对于内插方法来说，离散误差也是较小的。

✍ **论题 7.7.** 说明 *Milne* 方法对于任意的 μh 都不是绝对稳定的。

7.5 数值实验

❖ **练习 7.5.1.** 采用 *Euler* 折线法、梯形法、*RK3*、*RK4* 以及三步三阶 *Adams* 方法，求解常微分方程的初值问题

$$u' = 1 - \frac{2tu}{1+t^2}, \quad u(0) = 0.$$

首先绘制误差曲线，然后确定每个方法的"数值阶"。

❖ **练习 7.5.2.** 考虑常微分方程的初值问题

$$u' = xu(u-2), \quad u(0) = 2,$$

其真解是 $u(x) \equiv 2$。若初值扰动为 $u(0) = 2 + \varepsilon$，请用上述五种方法求解，观察相应的数值变化。

❖ **练习 7.5.3.** 考虑常微分方程组的初值问题

$$\begin{cases} p' = \alpha p + betapq, \\ q' = \gamma q + \delta pq, \end{cases}$$

其中 $\alpha = -1, \beta = 0.01, \gamma = 0.25, \delta = -0.01$。分别取不同的初值

$$(p(0), q(0)) = (30 \pm 1, 80 \pm 1),$$

利用上述五种方法进行模拟，观察数值解相图 (p, q) 的具体形态。

第 8 章
附录 2: 函数逼近

8.1 最佳一致逼近问题

给定连续函数 $f(x):[a,b]\to\mathbb{R}$ 和正整数 n，是否存在一个多项式

$$p(x)\in H_n=\mathrm{span}\{1,x,\ldots,x^n\},$$

使得

$$\min=\|f(x)-p(x)\|_\infty\equiv\max_{x\in[a,b]}|f(x)-p(x)|$$

这个问题就是最佳一致逼近问题。

定理 8.1. 最佳一致多项式是存在的。

证明：标准的连续函数极值问题。 □

定义 8.1. 常用的术语:

1. 偏差 $\Delta(f,p)=\|f-p\|_\infty$;
2. 最小偏差 $E_n=\min_{p\in H_n}\Delta(f,p)$;
3. 交错点组 $\{x_i\}$: 对于函数 $g(\cdot)$ 而言，有

$$|g(x_i)=(-1)^i\sigma\|g\|_\infty,$$

其中 $\sigma=\pm1$ 依赖具体的起始标号设置。

论题 8.1. 若 $p(x)$ 是 $f(x)$ 的最佳一致逼近多项式, 则 $f(x)-p(x)$ 必然同时存在正偏差点和负偏差点。

定理 8.2. 【*Chebyshev* 定理】设连续函数 $f(x) \notin H_n$，则以下两个命题是等价的：

> 1. $p(x)$ 是 $f(x)$ 的最佳一致逼近多项式；
> 2. $f(x) - p(x)$ 的交错点组至少含有 $n+2$ 个点。

证明：见教科书。 □

定理 8.3. 最佳一致多项式是唯一的。

☆ **说明 8.1.** 交错点组通常是不唯一的，而且很难确定。

> 设 $f^{(n)}(\cdot)$ 在 $[a,b]$ 上存在且不变号，则交错点组必然包含两个端点。

✎ **论题 8.2.** 设 $f(x) = x^n : [-1,1] \to \mathbb{R}$，在 H_{n-1} 中存在最佳一致逼近多项式 $p_{n-1}(x)$，相应的残量

$$x^n - p_{n-1}(x) = \frac{1}{2^{n-1}} T_n(x)$$

称为零偏差最小多项式，其中 $T_n(x) = \cos(n \arccos x)$ 是标准的 *Chebyshev* 多项式。

❀ **思考 8.1.** 在任意区间 $[a,b]$ 上，零偏差最小多项式是什么？

✎ **论题 8.3.** 里米兹算法可以数值求解最佳一致多项式。计算过程较为困难，特别是交错点组的确定。因此，常用以下的替代品：

> 1. *Chebyshev* 插值多项式：
> 2. 最佳平方逼近多项式：

论题 8.4. 在相同的逼近要求下，*Chebyshev* 节约化技术可以有效降低多项式的次数。

8.2 最佳平方逼近问题

给定平方可积 $f(x):[a,b]\to\mathbb{R}$，是否存在函数

$$S_\star(x)\in \operatorname{span}\{\phi_k(x):k=1:n\},$$

使得

$$\min=\|f(x)-p(x)\|_2\equiv\int_a^b\Big[f(x)-p(x)\Big]^2\mathrm{d}x,$$

其中 $W(x)$ 是已知的权函数。这个问题就是最佳一致平方问题。

论题 8.5. 最佳一致平方问题存在唯一解。

论题 8.6. 计算：法方程组和误差正交性

$$\Big\langle f(x)-S_\star(x),\phi_k(x)\Big\rangle=0,\quad k=1:n,$$

其中 $S_\star(x)=\sum_{k=1}^n\alpha_k\phi_k(x)$ 是最佳平方逼近多项式，$\langle\cdot,\cdot\rangle$ 是相应的内积运算。

论题 8.7. 广义 *Fourier* 级数的截断就是相应的最佳平方逼近。

8.3 离散的 Fourier 变换

设 $f(x)\colon[-\pi,\pi]\to\mathbb{C}$ 是平方可积（周期）函数。基于标准的连续型 L^2 内积，它在有限维函数空间

$$\mathcal{H}=\operatorname{span}\left\{\phi_k(x)=\frac{1}{\sqrt{2\pi}}e^{\mathbf{i}kx},x\in[-\pi,\pi]:k=0,\pm1,\ldots,\pm J\right\}$$

的最佳平方逼近函数就是 Fourier 级数的截断，即

$$\mathcal{P}f(x)=\frac{1}{\sqrt{2\pi}}\sum_{k=-J}^{J}e^{\mathbf{i}kx}\hat{f}_k,\quad x\in[-\pi,\pi],\tag{8.3.1}$$

其中

$$\hat{f}_k=\frac{1}{\sqrt{2\pi}}\int_{-\pi}^{\pi}e^{-\mathbf{i}kx}f(x)\mathrm{d}x.\tag{8.3.2}$$

若基于离散型 L^2 内积，最佳平方逼近函数将略有不同。设等距的离散网格为

$$x_j=j\pi/J,\quad j=-J:J,$$

相应的离散型 L^2 内积满足

$$\left\langle\frac{1}{\sqrt{2\pi}}e^{\mathbf{i}kx},\frac{1}{\sqrt{2\pi}}e^{\mathbf{i}lx}\right\rangle_{\triangle x}\equiv\frac{1}{2\pi}{\sum_{j=-J}^{J}}'\overline{e^{\mathbf{i}kj\triangle x}}e^{\mathbf{i}lj\triangle x}\triangle x=\delta_{kl},$$

其中撇号是指和项中下标为 $\pm J$ 的两项所具有权重 $1/2$，不是默认的权重 1。此时，$f(x)$ 在 $\mathcal{H}$ 内的（离散）最佳平方逼近函数是

$$\mathcal{I}f(x)=\frac{1}{\sqrt{2\pi}}\sum_{k=-J}^{J}e^{\mathbf{i}kx}\tilde{f}_k,\quad x\in[-\pi,\pi],\tag{8.3.3}$$

其中

$$\tilde{f}_k=\frac{1}{\sqrt{2\pi}}{\sum_{j=-J}^{J}}'f_je^{-\mathbf{i}kj\triangle x}\triangle x.\tag{8.3.4}$$

定理 8.4. *最佳平方逼近函数 (8.3.3) 也是插值函数，即*

$$f(x_j)=\mathcal{I}f(x_j),\quad j=0,\pm1,\ldots,\pm J.$$

因此说，两个周期序列 $\{f(x_j)\}_{j=-J:J}$ 和 $\{\hat{f}_k\}_{k=-J:J}$ 存在一一对应关系，(8.3.4) 称为离散 Fourier 变换。

✎ **论题 8.8.** *FFT*