科学与工程计算基础

任课教师：黄忠亿

清华大学数学科学系

目录

引言

线性方程组的求解是科学与工程计算领域中的一个核心内容，许多问题都要涉及。求解线性方程组的方法基本上分成两类：

1. 直接法（或可称为消元法及其变形）：即使用有限步代数运算得到准确解（忽略舍入误差的情况下）；
2. 迭代法：即使用一种迭代算法重复运算，产生一个解序列，以期收敛到准确解。

显然，上述两种方法的选取会取决于方程组的结构。粗略地讲，直接法适用于中小规模的满阵（即阶数不大，矩阵元素基本非零）；迭代法适用于大规模稀疏阵（矩阵元素多数为零）.

几个例子

下面我们通过几个例子来看一下此类问题的来源。

例 2.1

考虑用离散方法求解以下常微方程两点边值问题

$$\begin{cases} -u''(x) = f(x), \quad x \in [0,1] \\ u(0) = u(1) = 0. \end{cases} \tag{2.1}$$

这里假设 $f : [0,1] \to \mathbb{R}$ 连续。我们期望古典解 $u : [0,1] \to \mathbb{R}$ 是二阶连续可微的。

两点边值问题的求解

此类问题出现在弦的振动、热传导等定常情形（即解不再随时间变化）。由常微分方程理论我们知道该问题(2.1)解的存在唯一性，这里我们不详细讨论.

我们下面主要看上述问题的数值解如何得到。

取 $N \in \mathbb{N}$, 将 $[0,1]$ 区间 $N+1$ 等分：

$$x_j = jh,\ j = 0, \cdots, N+1,\ h = \frac{1}{N+1}.$$

在内点 x_j $(j = 1, \cdots, N)$ 将二阶微商近似为差商：

$$u''(x_j) \approx \frac{u'(x_j + \frac{h}{2}) - u'(x_j - \frac{h}{2})}{h} \approx \frac{1}{h^2}[u(x_{j+1}) - 2u(x_j) + u(x_{j-1})]$$

两点边值问题的求解

这样得到一系列方程：

$$-\frac{1}{h^2}[u_{j-1}-2u_j+u_{j+1}]=f(x_j),\quad j=1,\cdots,N$$

这里 u_j 是 $u(x_j)$ 的近似值。

定义

$$A_{N\times N}=\frac{1}{h^2}\begin{pmatrix}2&-1&&&\\-1&2&-1&&\\&\ddots&\ddots&\ddots&\\&&\ddots&\ddots&-1\\&&&-1&2\end{pmatrix},\ U_{N\times 1}=\begin{pmatrix}u_1\\u_2\\\vdots\\u_N\end{pmatrix},\ F_{N\times 1}=\begin{pmatrix}f_1\\f_2\\\vdots\\f_N\end{pmatrix}$$

即我们要求解 $AU=F$. 这是一个三对角矩阵问题。

两点边值问题的求解

当然我们有以下两个问题需要解决：

1. 上述线性方程组是否存在唯一解？
2. 近似解误差多大？$h \to 0$ 时，近似解 u_j 是否收敛到 $u(x_j)$？

这两个问题我们稍后会详细讨论。我们需要说明的是，对于高维（例如三维问题）上述方程组的规模是很大的, 例如一个方向的离散点数 $N = 100$, 三维情形矩阵规模会是 10^6 阶. 因此我们需要用有效算法来求解.

线性方程组求解问题的例子

考虑一个数据拟合问题：

例 2.2

假设物理量 u 依赖于时间 t 及一些参变量 $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{R}^n$，满足关系 $u(t) = f(t, \mathbf{x})$. 假如我们不知道这些参数 x_i 的值，但是我们可以测得 u 在不同时刻的值 $u(t_j) = f(t_j, \mathbf{x})$, $j = 1, \cdots, m$. 我们试图通过这些数据求出 x_i 的值来。

如果 $m = n$, 求解上述非线性方程组便可以求出 $\mathbf{x}$. 但是一般测量值都有误差, 为了保证好的稳定性, 一般都会取 $m \gg n$, 然后希望通过极小化误差 $u(t_j) - f(t_j, \mathbf{x})$ 来得到 $\mathbf{x}$. 这即是常用的最小二乘方法.

最小二乘求解数据拟合问题

即定义 $$\phi(x) = \sum_{j=1}^{m} [u(t_j) - f(t_j, x)]^2,$$

然后求 $\phi(x)$ 的极小值点 x^*. 而在 x^* 取极小值的必要条件是

$$\left.\frac{\partial \phi}{\partial x_i}\right|_{x^*} = 0, \quad i = 1, \cdots, n$$

这样得到下面非线性方程组

$$\sum_{j=1}^{m} [u(t_j) - f(t_j, x^*)] \frac{\partial f}{\partial x_i}(t_j, x^*) = 0, \quad i = 1, \cdots, n.$$

我们用迭代法求解上述非线性方程组的过程即要求解一个线性方程组。这通常是一个中小规模的满矩阵。

处理不同类型的线性方程组需要不同的处理方法.

目录

高斯消去法

高斯消去法是 Gauss 在1801年发表的论文《算术研究》中首先提出来的。 考虑求解 $A\mathbf{x} = \mathbf{b}$, $A \in \mathbb{C}^{n\times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{C}^n$. 假设 $\det A \neq 0$, 那么上述方程组存在唯一解。

消元法的基本思想就是，利用第一个方程消去后面$n-1$个方程中的第一个未知数, 再利用第二个方程消去后面 $n-2$ 个方程中的第二个未知数，依此类推, 直到把系数矩阵变成上三角阵形式.

然后从最后一个方程开始，解出 x_n, 代入前一个方程解出 x_{n-1}, 再把 x_n, x_{n-1} 代入前一个方程，解出 $x_{n-2}, \cdots, x_1$.

这就是最简单的高斯顺序消元法。

顺序消元和回代

将系数矩阵 A 与右端向量 b 放在一起组成一个增广矩阵

$$[A \mid b] = \left(\begin{array}{ccc|c} a_{11} & \cdots & a_{1n} & b_1 \\ a_{21} & \cdots & a_{2n} & b_2 \\ \vdots & & \vdots & \vdots \\ a_{n1} & \cdots & a_{nn} & b_n \end{array}\right)$$

记 $\left[A^{(1)} \mid b^{(1)}\right] = [A \mid b]$, 这里 $a_{ij}^{(1)} = a_{ij}, \quad b_i^{(1)} = b_i.$

计算比值 $l_{i1} = a_{i1}^{(1)}/a_{11}^{(1)}, i = 2, \cdots, n.$

然后将增广矩阵第一行乘以 $(-l_{i1})$ 加到第 i 行，得到

顺序消元和回代

$$[A^{(2)} \mid b^{(2)}] = \left(\begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} & b_n^{(2)} \end{array}\right)$$

这里
$$a_{ij}^{(2)} = a_{ij}^{(1)} - l_{i1}a_{1j}^{(1)}, i = 2, \cdots, n, j = 2, \cdots, n.$$
$$b_i^{(2)} = b_i^{(1)} - l_{i1}b_1^{(1)}, \ i = 2, \cdots, n.$$

顺序消元和回代

重复下去, 假设已做了 $k-1$ 步, 得到了 $\left[A^{(k)} \mid b^{(k)}\right]$, 并设 $a_{kk}^{(k)} \neq 0$.

令 $l_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}, \quad i = k+1, \cdots, n.$

将增广矩阵第k行乘以 $(-l_{ik})$ 加到第 i 行，得到 $\left[A^{(k+1)} \mid b^{(k+1)}\right]$.

即首先对 $k = 1, \cdots, n-1$ 做以下消元步骤

$$
\begin{cases}
a_{ij}^{(k+1)} = \begin{cases} a_{ij}^{(k)}, & i = 1, \cdots, k; \quad j = 1, \cdots, n \\ 0, & i = k+1, \cdots, n; \ j = 1, \cdots, k, \\ a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)}, & i = k+1, \cdots, n; \ j = k+1, \cdots, n; \end{cases} \\
b_i^{(k+1)} = \begin{cases} b_i^{(k)}, & i = 1, \cdots, k; \\ b_i^{(k)} - l_{ik} b_k^{(k)}, & i = k+1, \cdots, n. \end{cases}
\end{cases} \tag{2.2}
$$

顺序消元和回代

做完第 $n-1$ 步之后，我们得到

$$\left[A^{(n)} \mid b^{(n)}\right] = \left(\begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & a_{nn}^{(n)} & b_n^{(n)} \end{array}\right)$$

得到上述上三角阵后，我们就可以从最后一个方程开始求解:

$$\begin{cases} x_n = b_n^{(n)} / a_{nn}^{(n)}, \\ x_i = \dfrac{1}{a_{ii}^{(i)}} \left(b_i^{(i)} - \displaystyle\sum_{j=i+1}^{n} a_{ij}^{(i)} x_j \right), \quad i = n-1, n-2, \cdots, 1. \end{cases} \tag{2.3}$$

(2.2)-(2.3) 便构成了顺序高斯消元法求解线性方程组的全过程。

顺序消元和回代

算法 2.1 (Gauss顺序消元算法)

1. 对 $k = 1, \cdots, n-1$ 做以下消元步骤

$$l_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}, \quad i = k+1, \cdots, n.$$

$$\begin{cases} a_{ij}^{(k+1)} = \begin{cases} a_{ij}^{(k)}, & i = 1, \cdots, k; \quad j = 1, \cdots, n \\ 0, & i = k+1, \cdots, n; \quad j = 1, \cdots, k, \\ a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)}, & i = k+1, \cdots, n; \quad j = k+1, \cdots, n; \end{cases} \\ b_i^{(k+1)} = \begin{cases} b_i^{(k)}, & i = 1, \cdots, k; \\ b_i^{(k)} - l_{ik} b_k^{(k)}, & i = k+1, \cdots, n. \end{cases} \end{cases}$$

2. 回代:

$$\begin{cases} x_n = b_n^{(n)} / a_{nn}^{(n)}, \\ x_i = \dfrac{1}{a_{ii}^{(i)}} \left(b_i^{(i)} - \displaystyle\sum_{j=i+1}^{n} a_{ij}^{(i)} x_j \right), \quad i = n-1, n-2, \cdots, 1. \end{cases}$$

顺序消元和回代

我们也可以用矩阵记号来表示上述消元过程：

记 $A^{(k)} = L_{k-1}^{-1} A^{(k-1)}$, $b^{(k)} = L_{k-1}^{-1} b^{(k-1)}$,

即 $A^{(n)} = L_{n-1}^{-1} \cdot L_{n-2}^{-1} \cdots L_1^{-1} A$, $b^{(n)} = L_{n-1}^{-1} \cdot L_{n-2}^{-1} \cdots L_1^{-1} b$,

其中 $L_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & l_{n,k} & & 1 \end{pmatrix}$, $L_k^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -l_{n,k} & & 1 \end{pmatrix}$

顺序消元和回代

下面再看看顺序高斯消元法的计算量：

消去过程中，乘除法的次数为：

$$\underbrace{\sum_{k=1}^{n-1}(n-k)(n+1-k)}_{\text{计算}l_{ik}\text{及}a_{ij}^{(k)}}+\underbrace{\sum_{k=1}^{n-1}(n-k)}_{\text{计算}b_i^{(k)}} = \frac{n^3}{3}-\frac{n}{3}+\frac{n(n-1)}{2} = \frac{n^3}{3}+\frac{n^2}{2}-\frac{5n}{6}$$

加减法次数为 $\sum_{k=1}^{n-1}(n-k)(n+1-k)=\frac{n^3}{3}-\frac{n}{3}$.

回代乘除法$\sum_{k=1}^{n-1}(n+1-k)=\frac{n(n+1)}{2}$, 加减法 $\sum_{k=1}^{n-1}(n-k)=\frac{n(n-1)}{2}$.

这样总的乘除法次数为 $\frac{n^3}{3}+n^2-\frac{n}{3}$, 加减法次数 $\frac{n^3}{3}+\frac{n^2}{2}-\frac{5n}{6}$.

用 Cramer's 法则计算量为$(n+1)!\gg\mathcal{O}(n^3)$ (高斯消元法计算量)

顺序消元和回代

顺序消元能进行至第k步的充要条件是前k个顺序主子式都不为零:

定理 2.1

设 $A \in \mathbb{C}^{n\times n}$, 高斯顺序消元法可以进行到第 k 步 $\Longleftrightarrow$ 前面 k 个顺序主子式 $\Delta_1, \cdots, \Delta_k$ 都不为零, 这里 $\Delta_k = \begin{vmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kk} \end{vmatrix}$.

$\lhd$ 可以用归纳法证明. $k = 1$ 显然, 因为 $\Delta_1 = a_{11}^{(1)}$.

设到第 $k-1$ 步时上述定理成立，因为消元过程只是做初等变换，不会改变顺序主子式的值，易见 $\Delta_k = a_{11}^{(1)} \cdot a_{22}^{(2)} \cdots a_{kk}^{(k)}$,

因此有 $a_{11}^{(1)} \neq 0, \cdots a_{kk}^{(k)} \neq 0 \Longleftrightarrow \Delta_1 \neq 0, \cdots, \Delta_k \neq 0$. $\rhd$

矩阵的三角分解

由消元过程的矩阵形式及上述定理，可得以下矩阵三角分解定理

定理 2.2 (矩阵的 LU 分解)

设 $A \in \mathbb{C}^{n\times n}$, 若 $\Delta_i \neq 0, i = 1, 2, \cdots, n-1$. 则存在唯一的单位下三角阵(对角元都为1的下三角阵) L 与上三角阵 U, *s.t.* $A = LU$.

$\lhd$ L, U的存在性已经由高斯顺序消元过程给出：

由 (上三角阵$\to$) $A^{(n)} = L_{n-1}^{-1} \cdot L_{n-2}^{-1} \cdots L_1^{-1} A$, 以及

$$L = L_1 \cdot L_2 \cdots L_{n-1} = \begin{pmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{pmatrix}$$ 为单位下三角阵.

即若令 $U = A^{(n)}$, 立即有 $A = LU$. 下面主要看唯一性.

矩阵的三角分解

(续前页) 用归纳法证明唯一性：$n = 1$ 时显然，$a_{11} = 1 \cdot a_{11}$.

设 $n-1$ 阶的矩阵的上述三角分解是唯一的。对于n 阶矩阵A:

假设其有两种分解(写成分块矩阵形式)：

$$A = \begin{pmatrix} A_{n-1} & \mu \\ \nu^T & a_{nn} \end{pmatrix} = \begin{pmatrix} L_{n-1} & 0 \\ \sigma^T & 1 \end{pmatrix} \begin{pmatrix} U_{n-1} & \tau \\ 0 & u_n \end{pmatrix}$$

$$= \begin{pmatrix} \widetilde{L}_{n-1} & 0 \\ \widetilde{\sigma}^T & 1 \end{pmatrix} \begin{pmatrix} \widetilde{U}_{n-1} & \widetilde{\tau} \\ 0 & \widetilde{u}_n \end{pmatrix}$$

立即有 $A_{n-1} = L_{n-1}U_{n-1} = \widetilde{L}_{n-1}\widetilde{U}_{n-1}$, $\mu = L_{n-1}\tau = \widetilde{L}_{n-1}\widetilde{\tau}$,

$\nu^T = \sigma^T U_{n-1} = \widetilde{\sigma}^T \widetilde{U}_{n-1}$, $a_{nn} = \sigma^T\tau + u_n = \widetilde{\sigma}^T\widetilde{\tau} + \widetilde{u}_n$.

矩阵的三角分解

(续前页) 已知$\Delta_{n-1} \neq 0 \Longleftrightarrow A_{n-1}$ 可逆。

$L_{n-1}, \widetilde{L}_{n-1}$ 为单位上三角矩阵，当然可逆,

因而由 $A_{n-1} = L_{n-1}U_{n-1} = \widetilde{L}_{n-1}\widetilde{U}_{n-1} \Longrightarrow U_{n-1}, \widetilde{U}_{n-1}$ 也可逆

$$\Longrightarrow \widetilde{L}_{n-1}^{-1} \cdot L_{n-1} = \widetilde{U}_{n-1} \cdot U_{n-1}^{-1} \longrightarrow \text{只能是单位阵}$$

(单位下三角阵) (上三角阵)

即 $\widetilde{L}_{n-1} = L_{n-1}, \quad \widetilde{U}_{n-1} = U_{n-1}.$

马上推出 $\tau = \widetilde{\tau} = L_{n-1}^{-1}\mu, \quad \widetilde{\sigma}^T = \sigma^T = \nu^T \cdot \widetilde{U}_{n-1}^{-1}.$

自然也有 $\widetilde{u}_n = u_n = a_{nn} - \sigma^T\tau = a_{nn} - \widetilde{\sigma}^T\widetilde{\tau}$. 归纳证毕 ▷

矩阵的三角分解

推论 2.1

若 $A \in \mathbb{C}^{n\times n}$ 的顺序主子式 $\Delta_i \neq 0, i = 1, \cdots, n-1$. 则 A 可唯一地分解为 $A = LDU$ 形式，$L(U)$ 为单位下(上)三角阵, D为对角阵.

做三角分解的好处之一在于，如需解多个右端向量的方程组，只需一次三角分解，多次回代即可，这样可节约计算量.

因为一次回代的计算量为 $\mathcal{O}(n^2)$，而分解(消元)的计算量为 $\mathcal{O}(n^3)$.

矩阵的三角分解

例 2.3

设 $A=\begin{pmatrix}1 & 1 & 1\\ 0 & 4 & -1\\ 2 & -2 & 1\end{pmatrix}$, 做三角分解 $A=LU=LD\widetilde{U}$, 有

$$L=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 2 & -1 & 1\end{pmatrix}, U=\begin{pmatrix}1 & 1 & 1\\ 0 & 4 & -1\\ 0 & 0 & -2\end{pmatrix}; D=\begin{pmatrix}1 & & \\ & 4 & \\ & & -2\end{pmatrix}, \widetilde{U}=\begin{pmatrix}1 & 1 & 1\\ 0 & 1 & -\frac{1}{4}\\ 0 & 0 & 1\end{pmatrix}$$

但是我们知道，即便 $\det A\neq 0$，我们也无法保证所有顺序主子式非零; 或者即便顺序主子式都非零，可以进行顺序消元, 但是如果出现 $|a_{kk}^{(k)}|\ll 1$ 情形，用它做分母会带来很大舍入误差.

顺序消元存在的问题

来看一个例子：

例 2.4 (假设我们仅用三位有效数字计算：)

$$\begin{cases} 1.00 \times 10^{-5} x_1 + 1.00 x_2 = 1.00 \\ 1.00 x_1 + 1.00 x_2 = 2.00 \end{cases} \text{解为} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{0.99999} \\ \frac{0.99998}{0.99999} \end{pmatrix} \approx \begin{pmatrix} 1.00 \\ 1.00 \end{pmatrix}$$

如果顺序消元，$l_{21} = \frac{a_{21}}{a_{11}} = 1.00 \times 10^5$,

$$\Longrightarrow \begin{array}{l} a_{22}^{(2)} = 1.00 - 1.00 \times 10^5 \doteq -1.00 \times 10^5, \\ b_2^{(2)} = 2.00 - 1.00 \times 10^5 \doteq -1.00 \times 10^5 \end{array} \Longrightarrow x_2 = 1.00.$$

再代入第一个式子 $\Longrightarrow x_1 = 0.00$. 显然误差太大。

事实上若用第二个式子去消元就可得到很好的近似解。

这说明选主元很有必要！

列主元高斯消去法

从上例可以看出，当我们消元进行到第 k 步时

$$[A\,|\,b] \to [A^{(k)}\,|\,b^{(k)}] = \left(\begin{array}{ccccccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} & & b_1^{(1)} \\ & a_{12}^{(2)} & \cdots & \cdots & \cdots & a_{1n}^{(2)} & & b_2^{(2)} \\ & & \ddots & & & \vdots & & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} & & b_k^{(k)} \\ & & & \vdots & & \vdots & & \vdots \\ & & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} & & b_n^{(k)} \end{array}\right)$$

一个好的方案是，选取 $|a_{i_k,k}^{(k)}| = \max_{k\le l\le n} |a_{l,k}^{(k)}|$.

然后交换 $A^{(k)}$ 与 $b^{(k)}$ 的第 i_k 行和第 k 行，然后再消元.

这样可以有效避免用绝对值很小的数做分母带来的舍入误差.

列主元高斯消去法

这样相当于在增广矩阵前面乘上一个交换阵

$$I_{k,i_k} = \begin{pmatrix} 1 & & & & & & & & & & \\ & \ddots & & & & & & & & & \\ & & 1 & & & & & & & & \\ & & & 0 & 0 & \cdots & 0 & 1 & & & \to & k \\ & & & 0 & 1 & & & 0 & & & & \\ & & & \vdots & & \ddots & & \vdots & & & & \\ & & & 0 & & & 1 & 0 & & & & \\ & & & 1 & 0 & \cdots & 0 & 0 & & & \to & i_k \\ & & & & & & & & 1 & & & \\ & & & & & & & & & \ddots & & \\ & & & & & & & & & & 1 & \\ & & & \downarrow & & & & \downarrow & & & & \\ & & & k & & & & i_k & & & & \end{pmatrix}$$

即消元过程为 $\left[A^{(k+1)} \,\middle|\, b^{(k+1)}\right] = L_k^{-1} \cdot I_{k,i_k} \left[A^{(k)} \,\middle|\, b^{(k)}\right]$.

列主元高斯消去法

算法 2.2 (总结一下，列主元高斯消去法的算法可写为)

1. 消元：对 $k = 1, \cdots, n-1$ 做以下步骤

   先找列主元：$|a_{i_k,k}| = \max\limits_{k \le l \le n} |a_{l,k}|$.

   然后交换 $a_{i_k,j} \leftrightarrow a_{k,j}$ 与 $b_{i_k} \leftrightarrow b_k$，$\quad j = k, k+1, \cdots, n$

   再令 $l_{ik} = a_{ik}/a_{kk}$，$\quad i = k+1, \cdots, n$.

$$\begin{cases} a_{ij} = a_{ij} - l_{ik} \cdot a_{kj}, & i = k+1, \cdots, n;\ j = k+1, \cdots, n; \\ b_i = b_i - l_{ik} \cdot b_k, & i = k+1, \cdots, n. \end{cases}$$

2. 回代：
$$\begin{cases} x_n = b_n/a_{nn}, \\ x_i = (b_i - \sum\limits_{j=i+1}^{n} a_{ij} \cdot x_j)/a_{ii}, \quad i = n-1, n-2, \cdots, 1. \end{cases}$$

列主元高斯消去法

用矩阵语言来描述上述算法，可写成以下定理

定理 2.3 (列主元高斯消去法)

设 $A \in \mathbb{C}^{n\times n}$, 若 $\det A \neq 0$, 则存在一个排列阵 P、一个单位下三角阵 L 与一个上三角阵 U, *s.t.* $PA = LU$.

◁ 根据消元过程，若令 $\widetilde{P} = L_{n-1}^{-1} \cdot I_{n-1,i_{n-1}} \cdots L_1^{-1} \cdot I_{1,i_1}$,

那么有 $\widetilde{P}A = U$ 为上三角阵,

也就是说 $A = \widetilde{P}^{-1} \cdot U = I_{1,i_1} \cdot L_1 \cdots I_{n-1,i_{n-1}} \cdot L_{n-1} \cdot U$

如果令 $P = I_{n-1,i_{n-1}} \cdot I_{n-2,i_{n-2}} \cdots I_{1,i_1}$, 自然 P 为排列阵 $P^TP = I$

下面仅需验证 PA 可以写成 LU 的形式。 注意到

$$PA = (I_{n-1,i_{n-1}} \cdots I_{2,i_2} \cdot I_{1,i_1}) \cdot (I_{1,i_1} \cdot L_1 \cdots I_{n-1,i_{n-1}} \cdot L_{n-1}) \cdot U$$

列主元高斯消去法

(续前页) 利用交换阵的性质 $I_{ij}I_{ij} = I$ 立即有

$$\begin{aligned} PA &= (I_{n-1,i_{n-1}} \cdots I_{2,i_2}) \cdot L_1 \cdot (I_{2,i_2} \cdot L_2 \cdots I_{n-1,i_{n-1}} \cdot L_{n-1}) \cdot U \\ &= (I_{n-1,i_{n-1}} \cdots I_{2,i_2}) \cdot L_1 \cdot (I_{2,i_2} I_{3,i_3} \cdots I_{n-1,i_{n-1}}) \\ &\quad \cdot (I_{n-1,i_{n-1}} \cdots I_{3,i_3}) \cdot L_2 \cdot I_{3,i_3} \cdot L_3 \cdots I_{n-1,i_{n-1}} \cdot L_{n-1}) \cdot U \end{aligned}$$

易见 $\widetilde{L}_1 = (I_{n-1,i_{n-1}} \cdots I_{2,i_2}) \cdot L_1 \cdot (I_{2,i_2} \cdots I_{n-1,i_{n-1}})$ 仍是下三角阵

重复做下去就有 $PA = \widetilde{L}_1 \cdot \widetilde{L}_2 \cdots \widetilde{L}_{n-2} \cdot L_{n-1} \cdot U.$

其中 $\widetilde{L}_k = (I_{n-1,i_{n-1}} \cdots I_{k+1,i_{k+1}}) \cdot L_k \cdot (I_{k+1,i_{k+1}} \cdots I_{n-1,i_{n-1}})$

列主元高斯消去法

(续前页) 事实上注意到所有的 $i_j \geq j$,

$$L_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & l_{n,k} & & 1 \end{pmatrix} \Rightarrow \widetilde{L}_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \widetilde{l}_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & \widetilde{l}_{n,k} & & 1 \end{pmatrix}$$

其中 $(\widetilde{l}_{k+1,k}, \cdots, \widetilde{l}_{n,k})$ 只是 $(\widetilde{l}_{k+1,k}, \cdots, \widetilde{l}_{n,k})$ 换了一个次序而已。

这样令 $L = \widetilde{L}_1 \cdot \widetilde{L}_2 \cdots \widetilde{L}_{n-2} \cdot L_{n-1}$ 即证明了 $PA = LU$. $\triangleright$

当然我们也可以对每次消元过程中的右下方的小方阵选主元, 而不是仅仅选列主元, 自然可以获得更好的数值稳定性, 但是这样一来计算量大增。实际计算表明绝大多数情形列主元就够了。

Gauss-Jordan消去法

若消元时同时消去对角线上下的元素，那么就可以把 A 化成对角阵. 这样不用回代就可以得到最终解了. 但是这样一来计算量会增大到 $\frac{n^3}{2} + \mathcal{O}(n^2)$. 所以求解方程组时一般并不用此方法.

但上述方法(Gauss-Jordan消去法)可以用于计算 A^{-1}.

即求解 $AX = I$ 得到 $X = A^{-1}$.

也就是说，如果记增广矩阵为 $\left[A\middle|I\right]$, 用Gauss-Jordan消去法将之化为 $\left[I\middle|X\right]$, 便得到 $X = A^{-1}$.

我们用下面一个例子来说明具体做法：

Gauss-Jordan消去法

例 2.5

例如 $A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 6 \end{pmatrix}$, 欲求 $A^{-1} = \begin{pmatrix} -2 & 0 & 1 \\ 0 & 3 & -2 \\ 1 & -2 & 1 \end{pmatrix}$.

$\triangleleft$ 增广矩阵 $\left(\begin{array}{ccc|ccc} 1 & 2 & 3 & 1 & 0 & 0 \\ 2 & 3 & 4 & 0 & 1 & 0 \\ 3 & 4 & 6 & 0 & 0 & 1 \end{array}\right)$ 换行 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 3 & 4 & 6 & 0 & 0 & 1 \\ 2 & 3 & 4 & 0 & 1 & 0 \\ 1 & 2 & 3 & 1 & 0 & 0 \end{array}\right)$

归一化 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 1 & \frac{4}{3} & 2 & 0 & 0 & \frac{1}{3} \\ 2 & 3 & 4 & 0 & 1 & 0 \\ 1 & 2 & 3 & 1 & 0 & 0 \end{array}\right)$ 消元 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 1 & \frac{4}{3} & 2 & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & 0 & 1 & -\frac{2}{3} \\ 0 & \frac{2}{3} & 1 & 1 & 0 & -\frac{1}{3} \end{array}\right)$

Gauss-Jordan消去法

归一化 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 1 & \frac{4}{3} & 2 & 0 & 0 & \frac{1}{3} \\ 0 & 1 & 0 & 0 & 3 & -2 \\ 0 & \frac{2}{3} & 1 & 1 & 0 & -\frac{1}{3} \end{array}\right)$ 消元 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 1 & 0 & 2 & 0 & -4 & 3 \\ 0 & 1 & 0 & 0 & 3 & -2 \\ 0 & 0 & 1 & 1 & -2 & 1 \end{array}\right)$

消元 $\longrightarrow$ $\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -2 & 0 & 1 \\ 0 & 1 & 0 & 0 & 3 & -2 \\ 0 & 0 & 1 & 1 & -2 & 1 \end{array}\right)$ 即得到了 A^{-1}. $\triangleright$

由此可以看出, 利用Gauss消元法(或者其变形)还可以求 $\det A$.

目录

Doolittle三角分解(LU,LR分解)

利用矩阵的三角分解来求解 $A\mathbf{x}=\mathbf{b}$, 可以在求解多个右端向量情形明显减少计算量（一次分解、多次回代）. 针对某些特殊形式的矩阵，也可以减少计算量并减少舍入误差影响。

我们先来看顺序消元情形：假设 A 的顺序主子式都不为零，利用前面的结果，我们知道可以将 A 分解为 $A=LU$, 其中 L 为单位下三角阵，U 为上三角阵:

$$L=\begin{pmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{pmatrix},\ U=\begin{pmatrix} u_{11} & \cdots & u_{1n} \\ & \ddots & \vdots \\ & & u_{nn} \end{pmatrix},\ a_{ij}=u_{ij}+\sum_{k=1}^{i-1} l_{ik}u_{kj}.$$

然后再解 $LU\mathbf{x}=\mathbf{b}$, 即分为两步：$L\mathbf{y}=\mathbf{b}, U\mathbf{x}=\mathbf{y}$.

Doolittle三角分解(LU,LR分解)

算法 2.3 (小结一下，可以写成以下算法：)

1. 对 $r = 1, \cdots, n$，计算：

$$\begin{cases} u_{ri} = a_{ri} - \sum_{k=1}^{r-1} l_{rk}u_{ki}, & i = r, r+1, \cdots, n \\ l_{ir} = (a_{ir} - \sum_{k=1}^{r-1} l_{ik}u_{kr})/u_{rr}, & i = r, r+1, \cdots, n \end{cases}$$

2. 解 $L\mathbf{y} = \mathbf{b}$:

$$\begin{cases} y_1 = b_1, \\ y_i = b_i - \sum_{k=1}^{i-1} l_{ik}y_k, & i = 2, 3, \cdots, n \end{cases}$$

3. 解 $U\mathbf{x} = \mathbf{y}$:

$$\begin{cases} x_n = y_n/u_{nn}, \\ x_i = \left(y_i - \sum_{k=i+1}^{n} u_{ik}x_k\right)/u_{ii}, & i = n-1, n-2, \cdots, 1 \end{cases}$$

选主元LU分解

实际问题中无法保证上述分解过程中 $u_{rr} \neq 0$, 或者可能有 $|u_{rr}| \ll 1$，这时我们需要选主元的三角分解:

即将 $PA = LU$, 然后解 $L\mathbf{y} = P\mathbf{b}, U\mathbf{x} = \mathbf{y}$.

这里我们只需修改上面算法中的第一步即可：

1. 对 $r = 1, \cdots, n$: 先计算 $s_i = a_{ir} - \sum_{k=1}^{r-1} l_{ik}u_{kr}, i = r, \cdots, n$

   选主元 $|s_{i_r}| = \max_{r \leq i \leq n} |s_i|$. 再交换 $a_{rj} \leftrightarrow a_{i_r j}$, $b_r \leftrightarrow b_{i_r}$

   再计算 U 的第 r 行和 L 的第 r 列: $u_{rr} = s_{i_r}$,

$$\begin{cases} l_{ir} = s_i/u_{rr}, & i = r, r+1, \cdots, n \\ u_{ri} = a_{ri} - \sum_{k=1}^{r-1} l_{rk}u_{ki}, & i = r, r+1, \cdots, n \end{cases}$$

对称阵的三角分解

定义 2.1 (Cholesky分解)

若 A 为对称阵, 且 $A = LL^T$, 其中 L 为下三角阵且对角元为正, 则称该分解为 A 的一个*Cholesky分解*.

我们先看另一种形式分解:

定理 2.4 (LDL^T形式分解)

设 A 为对称阵且顺序主子式 $\Delta_k \neq 0, k = 1, \cdots, n-1$. 则 A 可唯一地分解为 $A = LDL^T$，其中 L 为单位下三角阵，D 为对角阵.

对称阵的三角分解

◁已证过A可唯一地分解为LU:

L为单位下三角阵, U 为上三角阵.

令 $D = diag(U_{ii})$, 由已知条件

$$\Delta_k \neq 0 \Longrightarrow U_{kk} \neq 0, \quad 1 \leq k \leq n-1.$$

再令 $U = D\widetilde{U}, \widetilde{u}_{ij} = u_{ii}^{-1} u_{ij}$, 有 $\widetilde{U}$为单位下三角阵. 即 $A = LD\widetilde{U}$.

再由 $A^T = A$ 及LU分解的唯一性 $\Longrightarrow \widetilde{U} = L^T$. ▷

再来看一下什么样的对称阵有Cholesky分解.

对称阵的三角分解

定理 2.5

设 A 为对称正定阵. 则 A 可唯一地分解为 $A = LL^T$，其中 L 为对角元为正的下三角阵.

唯一的cholesky分解

$\triangleleft$ 由对称正定矩阵的性质 $\Longrightarrow \Delta_k > 0, a_{kk} > 0, k = 1, \cdots, n.$

又上面定理 2.4 $\Longrightarrow$ A 可唯一分解为 $\widetilde{L}D\widetilde{L}^T$ 形式. 因而

$$\det A_i = \det(\widetilde{L}_i)\det(D_i)\det(\widetilde{L}_i^T) > 0 \Longrightarrow \det(D_i) > 0 \Longrightarrow d_{ii} > 0.$$

取 $D^{\frac{1}{2}} = diag(\sqrt{d_{ii}})$, 即得

$$A = \widetilde{L}D^{\frac{1}{2}}D^{\frac{1}{2}}\widetilde{L}^T \equiv LL^T, \quad \text{其中 } L = \widetilde{L}D^{\frac{1}{2}}. \triangleright$$

对称阵的三角分解

这样我们可以利用Cholesky分解求解对称正定情形 $A\mathbf{x} = \mathbf{b}$:

算法 2.4 (平方根法)

1. (*Cholesky* 分解) 对 $j = 1, \cdots, n$:
$$\begin{cases} l_{jj} &= (a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2)^{1/2}, \\ l_{ij} &= (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk})/l_{jj}, \quad i = j+1, \cdots, n \end{cases}$$
2. 先后求解 $L\mathbf{y} = \mathbf{b}$, 及 $L^T\mathbf{x} = \mathbf{y}$:
$$y_i = (b_i - \sum_{k=1}^{i-1} l_{ik} y_k)/l_{ii}, \; i = 1, 2, \cdots, n,$$
$$x_i = (y_i - \sum_{k=i+1}^{n} l_{ki} x_k)/l_{ii}, \; i = n, n-1, \cdots, 1.$$

从这里可以看出，使用Cholesky分解可以节约计算量, 这个算法计算量为$\frac{n^3}{6} + \mathcal{O}(n^2)$.

对称阵的三角分解

另外，由 $A = LL^T \Longrightarrow a_{jj} = \sum_{k=1}^{j} l_{jk}^2, j = 1, \cdots, n.$

这说明 $l_{jk}^2 \leq a_{jj} \leq \max_{1\leq j\leq n}\{a_{jj}\}$ 总有上界，且 l_{jj} 总是会大于 0.

即用上述平方根法求解对称矩阵情形方程组，不用选主元也能保持数值稳定，且计算量为Gauss消元法的一半左右。

我们从上述算法中可以看到，分解时需要计算许多平方根。而我们知道计算平方根比一次乘除法的计算量大很多.

为了避免计算平方根，我们可以用 $A = LDL^T$ 分解来计算, 即

$$a_{ij} = \sum_{k=1}^{n} l_{ik}d_k l_{jk} = l_{ij}d_j l_{jj} + \sum_{k=1}^{j-1} l_{ik}d_k l_{jk} \equiv t_{ij} + \sum_{k=1}^{j-1} t_{ik}l_{jk}.$$

对称阵的三角分解

即算法可写成：(利用 $a_{ij} = t_{ij} + \sum_{k=1}^{j-1} t_{ik}l_{jk}, \quad t_{ij} = l_{ij}d_j$)

算法 2.5 (改进的平方根法)

1. $d_1 = a_{11}$.

   对 $i = 2, \cdots, n$:

$$\begin{cases} t_{ij} = a_{ij} - \sum_{k=1}^{j-1} t_{ik}l_{jk}, & j = 1, \cdots, i-1, \\ l_{ij} = t_{ij}/d_j, & j = 1, \cdots, i-1, \\ d_i = a_{ii} - \sum_{k=1}^{i-1} t_{ik}l_{ik}. \end{cases}$$

2. 先后求解 $L\mathbf{y} = \mathbf{b}$, 及 $DL^T\mathbf{x} = \mathbf{y}$:

   $y_i = b_i - \sum_{k=1}^{i-1} l_{ik}y_k$, $i = 1, 2, \cdots, n$,

   $x_i = y_i/d_i - \sum_{k=i+1}^{n} l_{ki}x_k$, $i = n, n-1, \cdots, 1$.

带状矩阵的三角分解法

许多实际问题（如样条拟合、有限元或有限差分方法求解微分方程边值问题等）常会出现带状结构矩阵的线性方程组求解问题。这里三对角情形最为常见:

$$A=\begin{pmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{pmatrix}, \text{欲求解} \quad A\mathbf{x}=F\equiv\begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}.$$

假设 A 为对角占优矩阵，满足条件

$$|b_1|>|c_1|, \quad |b_n|>|a_n|, \quad |b_i|\geq|a_i|+|c_i|, i=2,\cdots,n-1. \quad (2.4)$$

带状矩阵的三角分解法

这样对矩阵 A 做 LU 分解有

$$A=LU\equiv\begin{pmatrix}1&&&\\l_2&1&&\\&\ddots&\ddots&\\&&l_n&1\end{pmatrix}\begin{pmatrix}u_1&c_1&&\\&\ddots&\ddots&\\&&u_{n-1}&c_{n-1}\\&&&u_n\end{pmatrix}\Longrightarrow\begin{cases}a_i=l_iu_{i-1}\\b_i=l_ic_{i-1}+u_i\end{cases}$$

总结一下可以写出以下算法：$(L\mathbf{y}=F,U\mathbf{x}=\mathbf{y})$

算法 2.6 (追赶法 (Thomas 方法))

1. $u_1=b_1$, 对 $i=2,\cdots,n$ 计算 $\begin{cases}l_i=a_i/u_{i-1}\\u_i=b_i-l_ic_{i-1}\end{cases}$
2. $\begin{cases}y_1=f_1\\y_i=f_i-l_iy_{i-1},\ i=2,\cdots,n\end{cases}$ $\begin{cases}x_n=y_n/u_n\\x_i=(y_i-c_ix_{i+1})/u_i,\ i=n-1,\cdots,1\end{cases}$

带状矩阵的三角分解法

显然，上述算法可以进行下去的条件是 $u_i \neq 0$.

可以证明，在 A 满足条件 (2.4) 时，就有以下定理

定理 2.6

设 A 满足条件 (2.4), 且 a_i, c_i 均不为零. 则 A 非奇异，且在上面追赶法*(算法 2.6)*中有

$$u_i \neq 0, \quad i = 1, \cdots, n; \qquad 0 < \frac{|c_i|}{|u_i|} < 1, \quad i = 1, \cdots, n-1 \tag{2.5}$$

$$|b_i| - |a_i| < |u_i| < |b_i| + |a_i|, \quad i = 2, \cdots, n \tag{2.6}$$

◁ 可以用归纳法来证明 (2.5)–(2.6) 成立。

$i = 1$ 显然, 因为 $u_1 = b_1 \Longrightarrow |u_1| = |b_1| > |c_1| > 0 \Longrightarrow 0 < \frac{|c_1|}{|u_1|} < 1.$

带状矩阵的三角分解法

(续前页) 下面设 $u_{i-1} \neq 0$, 且 $0 < \frac{|c_{i-1}|}{|u_{i-1}|} < 1$.

由追赶法中的公式立即有

$$|u_i| = |b_i - l_i c_{i-1}| = |b_i - \frac{a_i}{u_{i-1}} c_{i-1}| \geq |b_i| - |a_i| \frac{|c_{i-1}|}{|u_{i-1}|}$$

$$> |b_i| - |a_i| \geq |c_i| > 0 \quad \Longrightarrow \quad 0 < \frac{|c_i|}{|u_i|} < 1,$$

另外又有 $|u_i| \leq |b_i| + |a_i| \frac{|c_{i-1}|}{|u_{i-1}|} < |b_i| + |a_i|$.

这样就由归纳法证明了 (2.5)–(2.6) 成立.

再由 $\det A = \det L \det U = 1 \cdot \prod_{i=1}^{n} u_i \neq 0$ 立即得到 A 非奇异. ▷

带状矩阵的三角分解法

注 2.1

如果有某个 $a_{i0}=0$ (或者 $c_{i_0}=0$ 类似), 可以将 A 如下形式分块

$$A=\left(\begin{array}{cccc|cccc} b_1 & c_1 & & & & & & \\ a_2 & \ddots & \ddots & & & & & \\ & \ddots & \ddots & c_{i_0-2} & & & & \\ & & a_{i_0-1} & b_{i_0-1} & c_{i_0-1} & & & \\ \hline & & & 0 & b_{i_0} & c_{i_0} & & \\ & & & & a_{i_0+1} & \ddots & \ddots & \\ & & & & & \ddots & \ddots & c_{n-1} \\ & & & & & & a_n & b_n \end{array}\right) \equiv \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}$$

带状矩阵的三角分解法

将 $\mathbf{x}$ 与 F 做类似分块 $\mathbf{x} = \begin{pmatrix} \bar{\mathbf{x}}_1 \\ \bar{\mathbf{x}}_2 \end{pmatrix}$ $F = \begin{pmatrix} \bar{F}_1 \\ \bar{F}_2 \end{pmatrix}$,

原问题等价于求解 $\begin{cases} A_{22}\bar{\mathbf{x}}_2 = \bar{F}_2, \\ A_{11}\bar{\mathbf{x}}_1 = \bar{F}_1 - A_{12}\bar{\mathbf{x}}_2. \end{cases}$ 均可用追赶法求解.

注 2.2 (循环三对角阵的LU分解算法)

如果是周期边界条件情形常出现如下形式循环三对角阵

$$\begin{pmatrix} b_1 & c_1 & 0 & \cdots & 0 & a_1 \\ a_2 & b_2 & c_2 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & \ddots & c_{n-1} \\ c_n & 0 & \cdots & 0 & a_n & b_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}$$

带状矩阵的三角分解法

进行分块: 令 $\bar{A} = \begin{pmatrix} b_1 & c_1 & & \\ a_2 & \ddots & \ddots & \\ & \ddots & \ddots & c_{n-2} \\ & & a_{n-1} & b_{n-1} \end{pmatrix}$ 为 $(n-1)\times(n-1)$ 阶方阵

$\bar{\mathbf{x}} = \begin{pmatrix} x_1 \\ \vdots \\ x_{n-1} \end{pmatrix}, \bar{F} = \begin{pmatrix} f_1 \\ \vdots \\ f_{n-1} \end{pmatrix}, \boldsymbol{\alpha} = \begin{pmatrix} a_1 \\ 0 \\ \vdots \\ 0 \\ c_{n-1} \end{pmatrix}$ 均为 $n-1$ 维向量.

假设知道了 x_n, 我们来求解 $\bar{A}\bar{\mathbf{x}} = \bar{F} - x_n\boldsymbol{\alpha}$: 将 $\bar{A}$ 分解为 $\bar{A} = \bar{L}\bar{U}$.

求解 $\bar{L}\bar{U}\xi = \bar{F}$ 及 $\bar{L}\bar{U}\varepsilon = \boldsymbol{\alpha}$ 便可知 $\bar{\mathbf{x}} = \xi - x_n\varepsilon$ (线性叠加原理)

带状矩阵的三角分解法

也就是说 $x_i = \xi_i - x_n\varepsilon_i, i = 1, \cdots, n-1.$

特别 $x_1 = \xi_1 - x_n\varepsilon_1, x_{n-1} = \xi_{n-1} - x_n\varepsilon_{n-1}$

代入原来的最后一个方程 $c_n x_1 + a_n x_{n-1} + b_n x_n = f_n$

$$\Longrightarrow x_n = \frac{f_n - (c_n\xi_1 + a_n\xi_{n-1})}{b_n - (c_n\varepsilon_1 + a_n\varepsilon_{n-1})}$$

最后便得到其他的 $x_i = \xi_i - x_n\varepsilon_i,\ i = 1, \cdots, n-1.$

目录

病态方程组

由于舍入误差的影响，实际上我们在求解方程组 $A\mathbf{x} = \mathbf{b}$ 时，总会有误差. 即相当于我们是在求解

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

这里的 $\delta A, \delta\mathbf{b}$ 分别为求解过程中舍入误差对 A 和 $\mathbf{b}$ 带来的扰动.

我们关心的是，当 $\delta A, \delta\mathbf{b}$ 相对于 A 和 $\delta\mathbf{b}$ 来说较小时, $\delta\mathbf{x}$ 相对于 $\mathbf{x}$ 是不是也很小？ 我们从下例来得到这个问题的答案.

例 2.6 (病态方程组)

假设我们来求解

$$\begin{pmatrix} 1 & 1 \\ 1 & 1.00001 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

病态方程组

显然上述方程组的解为 $\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$.

假设我们得到 $\mathbf{x}$ 的数据时有（或者说在求解过程中相当于在 $\mathbf{b}$ 上增加了）一个小扰动, 即

$$\mathbf{b} \to \mathbf{b} + \delta\mathbf{b} = \begin{pmatrix} 2 \\ 2.00001 \end{pmatrix}, \quad \text{即相当于} \quad \delta\mathbf{b} = \begin{pmatrix} 0 \\ 0.00001 \end{pmatrix}.$$

这样解就成为 $\mathbf{x} + \delta\mathbf{x} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. 这相当于 $\delta\mathbf{x} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$.

显然误差太大了.

病态方程组

同样如果系数矩阵 A 有一个小扰动

$$A \to A + \delta A = \begin{pmatrix} 1 & 1 \\ 0.99999 & 1.00001 \end{pmatrix}, \text{ 即 } \delta A = \begin{pmatrix} 0 & 0 \\ -1 \times 10^{-5} & 0 \end{pmatrix}.$$

此时也有 $\mathbf{x} + \delta\widetilde{\mathbf{x}} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. 这相当于 $\delta\widetilde{\mathbf{x}} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$.

同样误差也太大了.

此例说明，有时方程组的解对于系数矩阵和右端向量的扰动非常敏感。此时称之为病态方程组 (ill-conditioned linear system).

为了能定量地估计 $\delta\mathbf{x}$ 相对于 $\mathbf{x}$ 的相对误差大小，我们需要用到向量、矩阵的范数概念及其性质。

算子（矩阵）范数

下面我们来看一下矩阵范数该如何引入.

定义 2.2 (线性映射)

映射 $A : X \to Y$ (X, Y 均为线性赋范空间) 称为**线性映射**, 如果

$$\forall \mathbf{x}, \mathbf{y} \in X, \forall \alpha, \beta \in \mathbb{C}(\mathbb{R}), \quad A(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha A\mathbf{x} + \beta A\mathbf{y}.$$

定义 2.3 (有界算子)

一个线性映射 $A : X \to Y$ 称为有界的， 是指存在常数 $C > 0$, *s.t.*

$\forall \mathbf{x} \in X, \quad \|A\mathbf{x}\|_Y \le C\|\mathbf{x}\|_X$. 对于有界线性算子 A，可以定义

$$\|A\| \coloneqq \sup_{\|\mathbf{x}\|_X = 1} \|A\mathbf{x}\|_Y < +\infty$$

为算子 A 的范数（是上面的最小上界 “C”）.

算子（矩阵）范数

定义 2.4 (连续算子)

算子 $A: U \subset X \to Y$ 称为在 $\mathbf{x} \in U$ 连续, 是指 $\forall \{\mathbf{x}_n\} \subset U$, 如果 $\lim\limits_{n\to\infty} \mathbf{x}_n = \mathbf{x}$, 我们都有 $\lim\limits_{n\to\infty} A\mathbf{x}_n = A\mathbf{x} \in Y$.

如果 $\forall \mathbf{x} \in U$, 算子 A 都在 $\mathbf{x}$ 连续，则称 A 为**连续算子**.

引理 2.1 (线性算子的连续性)

若线性算子在零点连续，则它为连续算子.

$\lhd$ 设线性算子 $A: U \subset X \to Y$ 在零点 $\mathbf{0} \in U$ 连续, 即 $\forall \{\mathbf{x}_n\} \subset U$, 如果 $\lim\limits_{n\to\infty} \mathbf{x}_n = \mathbf{0}$, 都有 $\lim\limits_{n\to\infty} A\mathbf{x}_n = A\mathbf{0} = \mathbf{0} \in Y$.

那么 $\forall \mathbf{y} \in U, \forall \{\mathbf{y}_n\} \subset U$, 如果 $\lim\limits_{n\to\infty} \mathbf{y}_n = \mathbf{y}$, 利用算子的线性性:

$A\mathbf{y}_n = A(\mathbf{y} + \mathbf{y}_n - \mathbf{y}) = A\mathbf{y} + A(\mathbf{y}_n - \mathbf{y})$, 而且 $\lim\limits_{n\to\infty} (\mathbf{y}_n - \mathbf{y}) = \mathbf{0}$.

因此立即有 $\lim\limits_{n\to\infty} A\mathbf{y}_n = A\mathbf{y} + A\mathbf{0} = A\mathbf{y}$. 即 A 为连续算子. $\rhd$

算子（矩阵）范数

定理 2.7 (线性算子的连续性与有界性的等价关系)

一个线性算子连续 $\Longleftrightarrow$ 该线性算子有界.

$\lhd$ 设 $A: U \subset X \to Y$ 有界，即 $\forall \mathbf{x} \in U,\ \|A\mathbf{x}\|_Y \leq C\|\mathbf{x}\|_X$.

设 $\{\mathbf{x}_n\} \to \mathbf{0}$, 那么立即有 $\|A\mathbf{x}_n\|_Y \leq C\|\mathbf{x}_n\|_X \to 0$. 即 $A\mathbf{0} = \mathbf{0}$.

算子在零点连续. 由上述引理马上有 A 为连续算子.

再看如果 A 为连续算子, 用反证法来证明其有界.

即如果不存在 $C > 0$ *s.t.* $\|A\mathbf{x}\|_Y \leq C\|\mathbf{x}\|_X$.

即 $\forall n \in \mathbb{N}, \exists \mathbf{x}_n \in U$, *s.t.* $\|\mathbf{x}_n\|_X = 1, \|A\mathbf{x}_n\|_Y \geq n$.

令 $\mathbf{y}_n = \frac{\mathbf{x}_n}{\|A\mathbf{x}_n\|_Y}$, 即有 $\mathbf{y}_n \to \mathbf{0}$. 由 A 之连续性，有 $A\mathbf{y}_n \to \mathbf{0}$.

但是我们有 $\|A\mathbf{y}_n\|_Y = 1$. 矛盾. 即 A 必然为有界算子. $\rhd$

算子（矩阵）范数

矩阵 $A \in \mathbb{R}^{n\times n}$ (或 $\mathbb{C}^{n\times n}$), 可看成 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 到自身的线性算子.

即 $\forall \mathbf{x} = (x_1, \cdots, x_n)^T \in \mathbb{R}^n(\text{或}C^n)$, $(A\mathbf{x})_j = \sum_{k=1}^{n} a_{jk}x_k, 1 \le j \le n.$

对于前面常用向量范数诱导出的算子（矩阵）范数, 有以下定理

定理 2.8 (几种常见矩阵范数的具体表达式)

显然矩阵 A 对于 $\mathbb{R}^n(\mathbb{C}^n)$ 中任一种范数都是有界算子. 特别有

$$\|A\|_1 := \max_{1\le j\le n} \sum_{i=1}^{n} |a_{ij}|,$$

$$\|A\|_\infty := \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}|,$$

$$\|A\|_2 := \sqrt{\rho(A^*A)}, \quad \text{其中 } \rho(A)\text{为}A\text{的谱半径.}$$

算子（矩阵）范数

◁ 我们先来看 $\|A\|_1$ 的表达式. 记 $C \equiv \max\limits_{1\le j\le n}\sum\limits_{i=1}^{n}|a_{ij}|$,

我们先证 $\max\limits_{\|\mathbf{x}\|_1=1}\|A\mathbf{x}\|_1 \le C$, 再证明 $\exists \mathbf{x}, \|\mathbf{x}\|_1 = 1, \textit{s.t.}\ \|A\mathbf{x}\|_1 = C$.

记 $A = (\vec{a}_1, \cdots, \vec{a}_n)$ 为列向量形式, 有 $\| \vec{a}_j \|_1 = \sum\limits_{i=1}^{n}|a_{ij}|$.

无妨设 $\| \vec{a}_k \|_1 = \max\limits_{1\le j\le n}\| \vec{a}_j \|_1 = C$. 那么 $\forall\|\mathbf{x}\|_1 = 1$, 有

$$\|A\mathbf{x}\|_1 = \left\|\sum_{j=1}^{n} x_j \vec{a}_j\right\|_1 \le \sum_{j=1}^{n}|x_j|\| \vec{a}_j \|_1 \le \| \vec{a}_k \|_1\|\mathbf{x}\|_1 = C.$$

特别取 $\mathbf{x} = \mathbf{e}_k = (0, \cdots, 0, 1, 0, \cdots, 0)^T$, 有 $\|A\mathbf{e}_k\|_1 = \| \vec{a}_k \|_1 = C$.

这样证明了 $\|A\|_1 = \max\limits_{1\le j\le n}\sum\limits_{i=1}^{n}|a_{ij}|$.

算子（矩阵）范数

类似地可以证明其他两个表达式.

我们来看 $\|A\|_\infty$ 的表达式. 记 $b_k \equiv \sum_{j=1}^{n} |a_{kj}| = \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}|$.

类似前面的证明过程, $\forall \|\mathbf{x}\|_\infty = 1$, 有 $\|A\mathbf{x}\|_\infty = \left\| \sum_{j=1}^{n} x_j \vec{a}_j \right\|_\infty$

$$= \max_{1\le i\le n} \left| \sum_{j=1}^{n} x_j a_{ij} \right| \le \max_{1\le j\le n} |x_j| \max_{1\le i\le n} \left(\sum_{j=1}^{n} |a_{ij}| \right) = b_k.$$

特别取 $\mathbf{x} = (\cdots, x_j, \cdots)^T$, 其中 $x_j = \mathrm{sgn}(a_{kj})$, 或者 $\bar{a}_{kj}/|a_{kj}|$.

有 $\|A\mathbf{x}\|_\infty = \max_{1\le i\le n} \left| \sum_{j=1}^{n} x_j a_{ij} \right| = \sum_{j=1}^{n} |a_{kj}| = b_k$. 即证明了 $\|A\|_\infty = b_k$.

算子（矩阵）范数

最后来看 $\|A\|_2$ 的表达式. 用内积形式来表达

$$0 \le \|A\mathbf{x}\|_2^2 = (A\mathbf{x}, A\mathbf{x}) = (A\mathbf{x})^* A\mathbf{x} = (A^*A\mathbf{x}, \mathbf{x}),$$

这说明 A^*A 是半正定阵, 特征值均为非负实数: $\lambda_1 \ge \cdots \ge \lambda_n \ge 0$.

相应特征向量规范基为 $\mathbf{v}_1, \cdots, \mathbf{v}_n$: 即 $(\mathbf{v}_i, \mathbf{v}_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \ne j \end{cases}$

$\forall \|\mathbf{x}\|_2 = 1$, 有 $\mathbf{x} = \sum_{i=1}^n \alpha_i \mathbf{v}_i \Longrightarrow \sum_{i=1}^n |\alpha_i|^2 = 1$. 而

$$\|A\mathbf{x}\|_2^2 = (A^*A\mathbf{x}, \mathbf{x}) = \left(\sum_{i=1}^n \lambda_i \alpha_i \mathbf{v}_i, \sum_{i=1}^n \alpha_i \mathbf{v}_i \right) = \sum_{i=1}^n \lambda_i |\alpha_i|^2 \le \lambda_1.$$

特别取 $\mathbf{x} = \mathbf{v}_1$, 有 $\|A\mathbf{v}_1\|_2^2 = \lambda_1 = \rho(A^*A)$.

这样证明了 $\|A\|_2 = \max\limits_{\|\mathbf{x}\|_2 = 1} \|A\mathbf{x}\|_2 = \sqrt{\lambda_1} = \sqrt{\rho(A^*A)}$. $\triangleright$

矩阵范数与谱半径关系

定理 2.9

对 $\mathbb{C}^n$ 上任一种范数 $\|\cdot\|$ 及任一方阵 $A \in \mathbb{C}^{n\times n}$, 有

$$\rho(A) \le \|A\|.$$

另一方面，$\forall A \in \mathbb{C}^{n\times n}$, $\forall \varepsilon > 0$, 存在范数 $\|\cdot\|_\varepsilon$ (会依赖于 A 和 ε)

s.t.

$$\|A\|_\varepsilon \le \rho(A) + \varepsilon.$$

◁ 我们只证前半部分，后半部分可以用构造法证明，这里略。

设 λ 是 A 的任一特征值: $A\mathbf{y} = \lambda\mathbf{y}$, 且设 $\|\mathbf{y}\| = 1$.

$$\|A\| = \sup_{\|\mathbf{x}\|=1} \|A\mathbf{x}\| \ge \|A\mathbf{y}\| = |\lambda| \cdot \|\mathbf{y}\| = |\lambda| \Longrightarrow \rho(A) \le \|A\| \quad \triangleright$$

矩阵条件数

我们回到求解线性方程组的舍入误差问题：

假设我们求解 $A\mathbf{x}=\mathbf{b}\to(A+\delta A)(\mathbf{x}+\delta\mathbf{x})=\mathbf{b}+\delta\mathbf{b}$. 我们先得看看扰动后方程组是否可解: $A+\delta A=A(I+A^{-1}\delta A)$

引理 2.2

若 $\|B\|<1$, 则 $I\pm B$ 可逆，且 $\|(I\pm B)^{-1}\|\le\frac{1}{1-\|B\|}$.

◁ 反证法: 设 $I\pm B$ 是奇异阵, 即 $(I\pm B)\mathbf{x}=\mathbf{0}$ 有非零解 $\mathbf{x}\neq\mathbf{0}$.

即 $\mathbf{x}=\mp B\mathbf{x}$, 这说明 B 有特征值 ∓1, 即 $\rho(B)\ge1$.

这与 $\rho(B)\le\|B\|<1$ 矛盾. 因此必有 $I\pm B$ 可逆.

矩阵条件数

下面计算其范数：

$$1 = \|I\| = \|(I \pm B)(I \pm B)^{-1}\| = \|(I \pm B)^{-1} \pm B(I \pm B)^{-1}\|$$

$$\geq \|(I \pm B)^{-1}\| - \|B(I \pm B)^{-1}\| \geq \|(I \pm B)^{-1}\| - \|B\| \cdot \|(I \pm B)^{-1}\|.$$

提取公因子即有 $\|(I \pm B)^{-1}\| \leq 1/(1 - \|B\|)$. ▷

下面来估计舍入误差带来的解的误差：

定理 2.10 (先验误差估计)

设 $\det A \neq 0$, $\mathbf{b} \neq \boldsymbol{0}$, $\|A^{-1}\|\|\delta A\| < 1$, 则有

$$\frac{\|\delta \mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\|A\|\,\|A^{-1}\|}{1 - \|A\|\,\|A^{-1}\|\,\frac{\|\delta A\|}{\|A\|}} \cdot \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta \mathbf{b}\|}{\|\mathbf{b}\|} \right).$$

矩阵条件数

$\triangleleft$ 由于 $\|A^{-1}\|\|\delta A\| < 1$, 故 $\|A^{-1}\cdot\delta A\| \le \|A^{-1}\|\|\delta A\| < 1$,

即 $I + A^{-1}\delta A$ 可逆. 亦即 $A + \delta A$ 可逆. 扰动后的方程组有唯一解:

$\mathbf{x}+\delta\mathbf{x} = (A+\delta A)^{-1}(\mathbf{b}+\delta\mathbf{b})$, 又 $\mathbf{x} = (A+\delta A)^{-1}(A+\delta A)\mathbf{x}$, 相减

$$\delta\mathbf{x} = (A+\delta A)^{-1}[\mathbf{b}+\delta\mathbf{b}-(A+\delta A)\mathbf{x}] = (I+A^{-1}\delta A)^{-1}A^{-1}[\delta\mathbf{b}-\delta A\cdot\mathbf{x}]$$

$$\Longrightarrow \|\delta\mathbf{x}\| \le \frac{\|A^{-1}\|}{1-\|A^{-1}\|\|\delta A\|}\cdot[\|\delta\mathbf{b}\| + \|\delta A\|\cdot\|\mathbf{x}\|],$$

再利用 $\|b\| = \|A\mathbf{x}\| \le \|A\|\cdot\|\mathbf{x}\|$ 即得. $\triangleright$

矩阵条件数

由上述定理可以看到, 舍入误差可能会被放大 $\|A\|\,\|A^{-1}\|$ 倍!

这个数对于线性方程组求解是个重要指标.

定义 2.5 (矩阵条件数)

设 $\det A \neq 0$, 令 $\operatorname{cond}(A) = \|A\| \cdot \|A^{-1}\|$ 称为矩阵 A 的条件数.

显然, $\operatorname{cond}(A)$ 越大, 求解 $A\mathbf{x} = \mathbf{b}$ 这个问题越病态, 舍入误差带来的影响可能会越大. 另外易见

$$\operatorname{cond}(A) = \|A\| \cdot \left\|A^{-1}\right\| \geq \left\|A \cdot A^{-1}\right\| = \|I\| = 1.$$

如果 U 为酉(正交)阵, 即 $U^*U = I$, 有2-范数意义下 $\operatorname{cond}_2(U) = 1$.

矩阵条件数

定理 2.11 (后验误差误差估计)

设 $\bar{\mathbf{x}}$ 为 $A\mathbf{x}=\mathbf{b}$ 的一个近似解. $\mathbf{r}=\mathbf{b}-A\bar{\mathbf{x}}$ 称为"残差". 有

$$\frac{1}{\mathrm{cond}(A)}\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}\leq\frac{\|\mathbf{x}-\bar{\mathbf{x}}\|}{\|\mathbf{x}\|}\leq\mathrm{cond}(A)\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}.$$

$\triangleleft$ 由 $\mathbf{x}=A^{-1}\mathbf{b},\bar{\mathbf{x}}=A^{-1}(\mathbf{b}-\mathbf{r})\Longrightarrow\mathbf{x}-\bar{\mathbf{x}}=A^{-1}\mathbf{r}.$

$\Longrightarrow\|\mathbf{x}-\bar{\mathbf{x}}\|\leq\|A^{-1}\|\cdot\|\mathbf{r}\|$, 再利用 $\|\mathbf{b}\|\leq\|A\|\cdot\|\mathbf{x}\|$, 即 $\dfrac{1}{\|\mathbf{x}\|}\leq\dfrac{\|A\|}{\|\mathbf{b}\|}$,

$\Longrightarrow\dfrac{\|\mathbf{x}-\bar{\mathbf{x}}\|}{\|\mathbf{x}\|}\leq\mathrm{cond}(A)\dfrac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$. 左半不等式可类似证明. $\triangleright$

该定理表明，即便残差很小时，如果矩阵条件数很大，得到的近似解依然可能不是一个好的近似. 如何改善近似解呢？

如何改善近似解

如果矩阵条件数不是太大，我们可以用如下迭代法来改善:

算法 2.7 (解残差方程迭代改善)

为节约计算量，先将 A 做选主元的 LU 分解, 无妨设 $A = LU$.

求解 $L\mathbf{y}_0 = \mathbf{b}, U\mathbf{x}_0 = \mathbf{y}_0$. 计算残差 $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$.

如果 $\|\mathbf{r}_0\| > \varepsilon > 0$ (ε 为预先选取的阈值), 则迭代改善:

对 $k = 1, 2, \cdots$, 做如下计算

求解 $L\mathbf{y}_k = \mathbf{r}_{k-1}, U\mathbf{e}_k = \mathbf{y}_k, \quad \mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{e}_k$

计算残差 $\mathbf{r}_k = \mathbf{r}_{k-1} - A\mathbf{e}_k$.

如果 $\|\mathbf{r}_k\| > \varepsilon$ 则继续迭代; 否则就输出近似解 $\bar{\mathbf{x}} = \mathbf{x}_k$.

如何降低矩阵条件数

通常按照上述迭代几次便可以提高几位有效数字. 但是如果矩阵条件数太大，此方法无效. 如果条件数太大, 则需对原矩阵做一些处理, 以期降低其条件数. 比如可以取适当的矩阵 $\bar{D}_1$ 与 $\bar{D}_2$ (一般取对角阵或三角阵) *s.t.*

$$\mathrm{cond}(\bar{D}_1 A \bar{D}_2) = \inf_{D_1 D_2} \mathrm{cond}(D_1 A D_2).$$

这种降低条件数的方法称之为矩阵平衡法.

如何降低矩阵条件数

看一个例子:

例 2.7

设 $A=\begin{pmatrix}10 & 10^5\\ 1 & 1\end{pmatrix}$, $\mathbf{b}=\begin{pmatrix}10^5\\ 2\end{pmatrix}$, 解为 $\mathbf{x}=\begin{pmatrix}1.00010001\cdots\\ 0.99989998\cdots\end{pmatrix}$.

$\mathrm{cond}_\infty(A)\approx 10^5$. 但若取 $\bar{D}_1=\begin{pmatrix}10^{-5} & 0\\ 0 & 1\end{pmatrix}$, 有 $\bar{D}_1A=\begin{pmatrix}10^{-4} & 1\\ 1 & 1\end{pmatrix}$.

此时$\mathrm{cond}(\bar{D}_1A)\approx 4$, 大为改善. 一般取 $\bar{D}_1=diag(s_i^{-1})$, 其中 $s_i=\max\limits_{1\le j\le n}|a_{ij}|$, 这称为行平衡. 类似地也可做列平衡。

正则化方法——改善矩阵条件数

一般而言最有效的办法是用正则化方法来改善矩阵条件数 (上述方法有时不一定有效). 俄罗斯数学家 Tikhonov (吉洪诺夫, 1906-1993) 提出了一种正则化方法，利用矩阵奇异值分解来进行.

我们先用矩阵奇异值分解给出 $A\mathbf{x} = \mathbf{b}$ 的解的表达式, 然后再给出正则化的办法.

定义 2.6

对 $A \in \mathbb{C}^{m\times n}$, 我们有 $M = \bar{A}^T A \in \mathbb{C}^{n\times n}$ 为半正定的 *Hermite* 阵. 即 M 的特征值均为非负实数, 记为 $\mu_1^2, \cdots, \mu_n^2$. 定义它们的非负平方根为 A 的奇异值, 即 $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n \geq 0$ 称为 A 的奇异值.

吉洪诺夫正则化方法

定理 2.12 (奇异值分解)

设 $A \in \mathbb{C}^{m\times n}$ 的秩为 r $\left(\leq \min(m,n)\right)$, 则存在实数

$$\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r > \mu_{r+1} = \mu_{r+2} = \cdots = \mu_n = 0,\ s.t.\ A = VDU^*$$

其中$V=(\mathbf{v}_1,\cdots,\mathbf{v}_m)\in \mathbb{C}^{m\times m}$, $U=(\mathbf{u}_1,\cdots,\mathbf{u}_n)\in \mathbb{C}^{n\times n}$ 为酉方阵,

$$D = \left(\begin{array}{ccc|c} \mu_1 & & & \\ & \ddots & & 0 \\ & & \mu_r & \\ \hline & 0 & & 0 \end{array}\right) \in \mathbb{R}^{m\times n}.$$

◁ 由矩阵 $M = \bar{A}^T A \in \mathbb{C}^{n\times n}$ 为半正定的 Hermite 阵.

吉洪诺夫正则化方法

M 的特征值均为非负实数, 无妨设为 $\mu_1^2 \geq \mu_2^2 \cdots \geq \mu_n^2 \geq 0$, 相应的特征向量设为 $\mathbf{u}_1, \cdots, \mathbf{u}_n$. 即 $M\mathbf{u}_i = \mu_i^2 \mathbf{u}_i, i = 1, \cdots, n$.

由于 M 是Hermite阵, 我们知道 $\{\mathbf{u}_i\}_{i=1}^n$ 可以构成 $\mathbb{C}^n$ 中的一组标准正交基, 即 $(\mathbf{u}_i, \mathbf{u}_j) = \delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$ 由于 A 的秩是 r, 因此 M 的秩也是 r, 也就是说有

$$\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r > \mu_{r+1} = \mu_{r+2} \cdots = \mu_n = 0.$$

对 $1 \leq j \leq r$, 令 $\mathbf{v}_j = \dfrac{A\mathbf{u}_j}{\mu_j}$. 立即有 $(\mathbf{v}_i, \mathbf{v}_j) = \dfrac{(A\mathbf{u}_i, A\mathbf{u}_j)}{\mu_i \mu_j} = \delta_{ij}$, 且

$$A^* \mathbf{v}_j = \frac{1}{\mu_j} A^* A \mathbf{u}_j = \mu_j \mathbf{u}_j, \quad j = 1, \cdots, r.$$

吉洪诺夫正则化方法

显然，若 $r < m$, 可以补充 $\mathbf{v}_r + 1, \cdots, \mathbf{v}_m$ ***s.t.*** $\{\mathbf{v}_j\}_{j=1}^m$ 是 $\mathbb{C}^m$ 的一组规范正交基. 因为 A 的秩为 r, 所以有A的零空间

$$N(A) = \{\mathbf{x} \in \mathbb{C}^n | A\mathbf{x} = 0\}$$

的维数为 $n - r$. 类似有A^*的零空间

$$N(A^*) = \{\mathbf{x} \in \mathbb{C}^m | A^*\mathbf{x} = 0\}$$

的维数为 $m - r$. 这样,

$$A\mathbf{u}_j = 0,\ j = r + 1, \cdots, n.\ A^*\mathbf{v}_k = 0, k = r + 1, \cdots, m.$$

因此有 $AU = VD$, 即 $A = VDU^*$. $\triangleright$

吉洪诺夫正则化方法

推论 2.2

$\forall \mathbf{x} \in \mathbb{C}^n$, 有分解式 $A\mathbf{x} = \sum_{j=1}^{r} \mu_j(\mathbf{x}, \mathbf{u}_j)\mathbf{v}_j$.

$\lhd$ 由 $\{\mathbf{u}_j\}_{j=1}^n$ 是规范正交基 $\Longrightarrow$ $\mathbf{x} = \sum_{j=1}^{n}(\mathbf{x}, \mathbf{u}_j)\mathbf{u}_j$

$$\Longrightarrow A\mathbf{x} = \sum_{j=1}^{n}(\mathbf{x}, \mathbf{u}_j)A\mathbf{u}_j = \sum_{j=1}^{r}\mu_j(\mathbf{x}, \mathbf{u}_j)\mathbf{v}_j,$$

这里用到 $A\mathbf{u}_j = \mu_j\mathbf{v}_j, j = 1, \cdots, r;\ A\mathbf{u}_j = 0, j = r+1, \cdots, n.$ $\rhd$

吉洪诺夫正则化方法

利用上述分解，我们立即有以下结论:

推论 2.3

$A\mathbf{x}=\mathbf{b}$ 有解 $\Longleftrightarrow (\mathbf{b},\mathbf{y})=0, \forall \mathbf{y}\in N(A^*)=\{\mathbf{y}\in\mathbb{C}^m | A^*\mathbf{y}=0\}$.

由上面推论可知此时解为 $\mathbf{x}=\sum_{j=1}^{r}\frac{1}{\mu_j}(\mathbf{b},\mathbf{v}_j)\mathbf{u}_j$.

$\lhd$ 若 $A\mathbf{x}=\mathbf{b}$ 有解, 则 $(\mathbf{b},\mathbf{y})=(A\mathbf{x},\mathbf{y})=(\mathbf{x},A^*\mathbf{y})=0,\ \forall \mathbf{y}\in N(A^*)$.

反之, 若 $(\mathbf{b},\mathbf{y})=0, \forall \mathbf{y}\in N(A^*)$, 那么由 $\mathbf{b}=\sum_{j=1}^{n}(\mathbf{b},\mathbf{v}_j)\mathbf{v}_j$

$\Longrightarrow \mathbf{b}=\sum_{j=1}^{r}(\mathbf{b},\mathbf{v}_j)\mathbf{v}_j$. (因为 $\mathbf{v}_{r+1},\cdots,\mathbf{v}_n\in N(A^*)$)

这样 $\mathbf{b}=\sum_{j=1}^{r}\frac{1}{\mu_j}(\mathbf{b},\mathbf{v}_j)A\mathbf{u}_j$ (注意 $\mathbf{v}_j=\frac{1}{\mu_j}A\mathbf{u}_j$)

此时取 $\mathbf{x}=\sum_{j=1}^{r}\frac{1}{\mu_j}(\mathbf{b},\mathbf{v}_j)\mathbf{u}_j\in\mathbb{C}^n \Longrightarrow A\mathbf{x}=\mathbf{b}$. $\rhd$

吉洪诺夫正则化方法

这样对一般的算子 $A(\in \mathbb{C}^{m\times n}) : \mathbb{C}^n \to \mathbb{C}^m$ 也可以定义其广义逆 $A^\dagger : \mathbb{C}^m \to \mathbb{C}^n$:

$$A^\dagger \mathbf{b} = \sum_{j=1}^{r} \frac{1}{\mu_j}(\mathbf{b}, \mathbf{v}_j)\mathbf{u}_j, \quad \forall \mathbf{b} \in \mathbb{C}^m.$$

从 $A\mathbf{x} = \mathbf{b}$ 的解之表达式 $\mathbf{x} = \sum_{j=1}^{r} \frac{1}{\mu_j}(\mathbf{b}, \mathbf{v}_j)\mathbf{u}_j$ 可以看出:

当 $\mu_j \ll 1$ 时,舍入误差有可能会给近似解带来大的误差.

因此我们可以考虑将 $\frac{1}{\mu_j}$ 乘以一个因子 $\frac{\mu_j^2}{\alpha + \mu_j^2}$,

这里 α (通常 $1 \gg \alpha > 0$) 称为正则化参数.

吉洪诺夫正则化方法

定理 2.13

设 $A \in \mathbb{C}^{m\times n}$ 秩为 $r, \forall \alpha > 0, \forall \mathbf{b} \in \mathbb{C}^m$, 方程组 $(\alpha I + A^*A)\mathbf{x}_\alpha = A^*\mathbf{b}$ 存在唯一解: $\mathbf{x}_\alpha = \sum_{j=1}^{r} \frac{\mu_j}{\alpha + \mu_j^2}(\mathbf{b}, \mathbf{v}_j)\mathbf{u}_j \in \mathbb{C}^n$.

◁ 因 $\alpha I + A^*A$ 为正定 Hermite 阵, 上面方程组自然存在唯一解. 且其特征值为 $\alpha + \mu_j^2 > 0$, 特征向量仍为 $\mathbf{u}_j, j = 1, \cdots, n$.

由上面推论 2.3 的证明过程及结论, 结合 $\mathbf{b} = \sum_{j=1}^{m}(\mathbf{b}, \mathbf{v}_j)\mathbf{v}_j \Longrightarrow A^*\mathbf{b} = \sum_{j=1}^{r}\mu_j(\mathbf{b}, \mathbf{v}_j)\mathbf{u}_j$ 立即有上述结论. ▷

吉洪诺夫正则化方法

这表明, 求解 $A\mathbf{x}=\mathbf{b}$ 可以用 $(\alpha I+A^*A)\mathbf{x}_\alpha=A^*\mathbf{b}$ 来近似 $(0<\alpha\ll 1)$.

可以证明, $\forall \mathbf{b}\in\mathbb{C}^m$, $\lim\limits_{\alpha\to 0}(\alpha I+A^*A)^{-1}A^*\mathbf{b}=A^\dagger\mathbf{b}$.

也就是说, 只要 α 充分小, $(\alpha I+A^*A)\mathbf{x}_\alpha=A^*\mathbf{b}$ 的解 $\mathbf{x}_\alpha$ 会是原问题 $A\mathbf{x}=\mathbf{b}$ 的解 $\mathbf{x}$ 的一个好的近似.

另一方面, 我们需注意, $\mathrm{cond}_2(\alpha I+A^*A)=\frac{\alpha+\mu_1^2}{\alpha+\mu_n^2}$.

A 为方阵时, $\mathrm{cond}_2(A)=\frac{\mu_1}{\mu_n}$.

我们自然希望 $\frac{\alpha+\mu_1^2}{\alpha+\mu_n^2}<\frac{\mu_1}{\mu_n}$, 即要 $\alpha>\mu_1\mu_n$. 这一般要求取 α 比 $\mu_1\mu_n$ 至少大一个量级.

吉洪诺夫正则化方法

又,

$$\|\mathbf{x}_\alpha - \mathbf{x}\|_2 \leq \|(\alpha I + A^*A)^{-1}A^*\|_2 \cdot \delta + \|(\alpha I + A^*A)^{-1}A^*\mathbf{b} - A^\dagger \mathbf{b}\|_2,$$

其中δ表示求解方程组过程中右端向量 $\mathbf{b}$ 的扰动带来的误差.

这样，若让上式右端两项差不多一个量级, 粗略一点可以取

$$\mu_1\mu_n < \alpha \sim \mathcal{O}(\mu_1^2\delta)$$