

矩阵计算讲义*

潘建瑜

华东师范大学 数学科学学院

2026 年 8 月

学习建议

- ✿ 算法学习 (算法的基本思想、设计过程、理论分析, **HOW** and **WHY**)
- ✿ 算法实现 (编程实践、程序优化、误差分析, 真正理解与掌握算法)
- ✿ 算法创新 (算法改进、推广、融合, 新思想、新技术、新应用、跨专业、跨学科等)

纸上得来终觉浅，绝知此事要躬行。

学而不思则罔，思而不学则殆。

知识，只有当它靠积极的思考得来
而不是凭记忆得来的时候，才是真正的知识。



目 录

第零讲 引言	1
0.1 计算数学介绍	1
0.2 数值线性代数	3
第一讲 线性代数基础	5
1.1 线性空间与内积空间	5
1.1.1 线性空间	5
1.1.2 子空间与仿射子空间	7
1.1.3 内积与内积空间	8
1.1.4 正交与正交补	9
1.2 矩阵与投影	10
1.2.1 矩阵的秩	11
1.2.2 特征值与特征向量	13
1.2.3 特征值的粗略估计	15
1.2.4 不变子空间	16
1.2.5 投影变换与正交投影	17
1.3 向量范数与矩阵范数	20
1.3.1 向量范数	20
1.3.2 矩阵范数	22
1.3.3 谱半径与范数	24
1.3.4 最佳逼近与正交投影	25
1.4 矩阵标准型	25
1.4.1 Jordan 标准型	25
1.4.2 Schur 分解	27
1.5 几类特殊矩阵	28
1.5.1 对称正定矩阵	28
1.5.2 对角占优矩阵	29
1.5.3 不可约矩阵	30
1.5.4 其它常见特殊矩阵	32
1.6 Kronecker 积	34
1.7 课后习题	36

第二讲 线性方程组直接方法	41
2.1 LU 分解与 Gauss 消去法	42
2.1.1 LU 分解	42
2.1.2 LU 分解的实现	43
2.1.3 Gauss 消去法	46
2.1.4 选主元 LU 分解	48
2.1.5 矩阵求逆	49
2.1.6 分块 LU 分解	50
2.2 特殊方程组的求解	50
2.2.1 对称正定线性方程组	50
2.2.2 对称不定线性方程组	53
2.2.3 三对角线性方程组	55
2.2.4 带状线性方程组	56
2.2.5 Toeplitz 线性方程组	57
2.3 扰动分析	62
2.3.1 矩阵条件数	62
2.3.2 δx 与 $\hat{x}$ 的关系	63
2.3.3 δx 与 x_* 的关系	64
2.3.4 δx 与残量的关系	65
2.3.5 相对扰动分析	65
2.4 误差分析	67
2.4.1 LU 分解的舍入误差分析	67
2.4.2 Gauss 消去法的舍入误差分析	67
2.5 解的改进	69
2.5.1 高精度运算	69
2.5.2 矩阵元素缩放或平衡 (Scaling or equilibration)	69
2.5.3 迭代改进法	69
2.6 课后习题	70
第三讲 线性最小二乘问题	73
3.1 问题介绍	73
3.1.1 超定方程组	73
3.1.2 欠定方程组	74
3.2 几类重要的矩阵变换	74



3.2.1	初等矩阵变换	74
3.2.2	Gauss 变换	76
3.2.3	Householder 变换	76
3.2.4	Givens 变换	79
3.2.5	正交变换的舍入误差分析	81
3.3	QR 分解	82
3.3.1	QR 分解的存在性与唯一性	82
3.3.2	基于 MGS 的 QR 分解	84
3.3.3	基于 Householder 变换的 QR 分解	86
3.3.4	列主元 QR 分解	88
3.3.5	基于 Givens 变换的 QR 分解	89
3.3.6	基于 Cholesky 分解的 QR 分解	91
3.3.7	QR 分解的稳定性	91
3.4	奇异值分解	93
3.4.1	奇异值, 奇异向量和奇异值分解	94
3.4.2	奇异值基本性质	96
3.4.3	奇异值更多性质	97
3.4.4	奇异值扰动分析	99
3.5	线性最小二乘问题的求解方法	99
3.5.1	正规方程	99
3.5.2	Cholesky 分解法	100
3.5.3	QR 分解法	101
3.5.4	奇异值分解法	102
3.6	广义逆与最小二乘	103
3.6.1	广义逆	103
3.6.2	广义逆基本性质	103
3.6.3	广义逆的计算	104
3.6.4	广义逆与线性最小二乘	105
3.6.5	广义逆与条件数	105
3.6.6	左逆和右逆	106
3.7	最小二乘扰动分析	106
3.8	推广与应用	107
3.8.1	最小二乘问题的推广	107
3.8.2	最小二乘问题的应用	108
3.9	课后习题	110



第四讲 非对称特征值问题	114
4.1 幂迭代法	114
4.1.1 算法介绍	114
4.1.2 收敛性分析	115
4.1.3 位移策略	116
4.2 反迭代法	116
4.2.1 算法介绍	117
4.2.2 Rayleigh 商迭代	117
4.3 正交迭代法	118
4.4 QR 迭代法	119
4.4.1 QR 迭代与幂迭代的关系	120
4.4.2 QR 迭代与反迭代的关系	120
4.4.3 QR 迭代与正交迭代的关系	121
4.4.4 QR 迭代的收敛性	121
4.4.5 带位移的 QR 迭代法	122
4.5 带位移的隐式 QR 迭代法	122
4.5.1 上 Hessenberg 矩阵	123
4.5.2 隐式 QR 迭代	125
4.5.3 位移的选取	127
4.5.4 收缩技术	131
4.6 特征向量的计算	131
4.7 广义特征值问题	131
4.7.1 广义特征值基本理论	131
4.7.2 广义 Schur 分解	132
4.7.3 QZ 迭代法	132
4.8 应用	133
4.8.1 多项式求根	133
4.9 课后习题	134
第五讲 对称特征值问题	137
5.1 Jacobi 迭代法	137
5.2 Rayleigh 商迭代法	140
5.3 对称 QR 迭代法	141
5.4 分而治之法	142



5.5	对分法和反迭代法	148
5.6	奇异值分解	150
5.6.1	二对角化	151
5.6.2	Golub-Kahan SVD 算法	154
5.6.3	dqds 算法	154
5.6.4	Jacobi 算法	156
5.7	扰动分析	157
5.7.1	特征值与 Rayleigh 商	158
5.7.2	对称矩阵特征值的扰动分析	159
5.7.3	对称矩阵特征向量的扰动	161
5.7.4	Rayleigh 商逼近	162
5.7.5	相对扰动分析	162
5.8	应用: SVD 与图像压缩	163
5.9	课后习题	164
第六讲	线性方程组基本迭代法	167
6.1	矩阵分裂与迭代法	169
6.1.1	Jacobi 迭代法	170
6.1.2	Gauss-Seidel 迭代法	171
6.1.3	SOR 迭代法	171
6.1.4	SSOR 迭代法	172
6.1.5	AOR 迭代法	173
6.1.6	Richardson 迭代法	174
6.1.7	分块迭代法	174
6.2	收敛性分析	175
6.2.1	基本概念	175
6.2.2	不可约对角占优矩阵	177
6.2.3	对称正定矩阵	178
6.2.4	相容次序矩阵	179
6.3	应用: Poisson 方程求解	182
6.3.1	一维 Poisson 方程	182
6.3.2	二维 Poisson 方程	184
6.3.3	收敛性分析	188
6.3.4	快速求解方法	189



6.4	加速方法	192
6.4.1	外推技术	192
6.4.2	Chebyshev 多项式加速	194
6.5	交替方向法与 HSS 迭代法	198
6.5.1	多步迭代法	198
6.5.2	交替方向法	199
6.5.3	HSS 方法	199
6.6	课后习题	201
第七讲	Krylov 子空间迭代法	204
7.1	Krylov 子空间	204
7.1.1	Arnoldi 过程	205
7.1.2	Lanczos 过程	207
7.1.3	Krylov 子空间方法一般格式	208
7.1.4	最佳近似解	208
7.2	GMRES 方法	209
7.2.1	算法描述	209
7.2.2	具体实施细节	210
7.2.3	GMRES 算法的中断	214
7.2.4	带重启的 GMRES 方法	214
7.3	共轭梯度法	216
7.3.1	算法基本过程	217
7.3.2	实用迭代格式	218
7.4	收敛性分析	221
7.4.1	CG 的收敛性	221
7.4.2	CG 的超线性收敛性	223
7.4.3	GMRES 的收敛性	224
7.5	预处理方法	228
7.5.1	预处理方法介绍	228
7.5.2	预处理 CG 方法	229
7.5.3	预处理 GMRES 方法	232
7.5.4	预处理子构造	234
7.6	课后习题	237
	参考文献	238



算 法 目 录

2.1	Gauss 消去法	42
2.2	LU 分解	44
2.3	LU 分解 (用 A 存储 L 和 U)	45
2.4	LU 分解 (IKJ 型)	46
2.5	Gauss 消去法	47
2.6	回代求解 $Ly = b$ 和 $Ux = y$	47
2.7	向后回代求解 $Ux = y$ (列存储方式)	47
2.8	部分选主元 LU 分解	49
2.9	Cholesky 分解 (按列计算)	51
2.10	改进的平方根法	52
2.11	追赶法	55
2.12	带状矩阵的 LU 分解	56
2.13	求解 Yule-Walker 方程组的 Levinson-Durbin 算法	59
2.14	求解对称正定 Toeplitz 线性方程组的 Levinson-Durbin 算法	60
2.15	通过迭代改进解的精度	70
3.1	计算 Householder 向量	78
3.2	Givens 变换	80
3.3	Gram-Schmidt 正交化过程	82
3.4	基于 MGS 的 QR 分解	85
3.5	基于 Householder 变换的 QR 分解 (Householder-QR)	87
3.6	基于 Givens 变换的 QR 分解	90
3.7	基于 Cholesky 分解的 QR 分解 (Cholesky-QR)	91
4.1	幂迭代法 (Power Iteration)	114
4.2	带位移的反迭代法 (Inverse Iteration)	117
4.3	Rayleigh 商迭代法 (Rayleigh Quotient Iteration (RQI))	117
4.4	正交迭代法 (Orthogonal Iteration)	118
4.5	QR 迭代法 (QR Iteration)	119
4.6	带位移的 QR 迭代法 (QR Iteration with shift)	122
4.7	上 Hessenberg 化 (Upper Hessenberg Reduction)	124
5.1	Jacobi 迭代法	138
5.2	经典 Jacobi 迭代法	138
5.3	循环 Jacobi 迭代算法 (逐行扫描)	139
5.4	Rayleigh 商迭代算法 (RQI, Rayleigh Quotient Iterations)	140

5.5	计算对称三对角矩阵的特征值和特征向量的分而治之法 (函数形式)	144
5.6	修正的 Newton 算法	147
5.7	计算矩阵 $D + uu^T$ 的特征值和特征向量的稳定算法	148
5.8	对分法: 计算 A 在 $[a, b)$ 中的所有特征值	149
5.9	带位移的 LR 算法	154
5.10	qds 算法的单步 ($B_i \rightarrow B_{i+1}$)	155
5.11	dqds 算法的单步 ($B_i \rightarrow B_{i+1}$)	156
5.12	单边 Jacobi 旋转的单步	156
5.13	单边 Jacobi 算法: 计算 $A = U\Sigma V^T$	157
6.1	Jacobi 迭代法	170
6.2	Gauss-Seidel 迭代法	171
6.3	求解线性方程组的 SOR 迭代法	172
6.4	SSOR 迭代法	173
6.5	求解二维离散 Poisson 方程的 Jacobi 迭代法	185
6.6	求解二维离散 Poisson 方程的红黑排序 G-S 迭代法	186
6.7	求解二维离散 Poisson 方程的 SOR 迭代法	187
6.8	二维离散 Poisson 方程的快速方法	192
6.9	Chebyshev 加速算法	197
7.1	Arnoldi 过程	205
7.2	Lanczos 过程	207
7.3	Krylov 子空间迭代算法	208
7.4	GMRES 方法基本框架	210
7.5	实用 GMRES 方法	212
7.6	GMRES(k): 带重启的 GMRES 方法	214
7.7	共轭梯度法 (CG)	220
7.8	两边预处理 CG 方法	230
7.9	PCG: 预处理 CG 方法	230
7.10	基于 P -内积的 CG 方法	231
7.11	右预处理 GMRES 方法	233



第零讲 引言

本讲义主要介绍矩阵计算 (Matrix Computations, 也称数值线性代数 Numerical Linear Algebra) 中基本问题的数值求解方法, 具体内容包括:

- 线性方程组的直接解法
- 线性最小二乘问题的直接解法
- 非对称矩阵的特征值计算
- 对称矩阵的特征值计算
- 矩阵奇异值分解
- 线性方程组的基本迭代法
- 线性方程组的 Krylov 子空间迭代法

主要参考资料

- ▶ G. Golub and C. F. van Loan, *Matrix Computations*, 4th Edition, 2013. [49]
- ▶ J. W. Demmel, *Applied Numerical Linear Algebra*, 1997. [23]
- ▶ L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*, 1997. [103]
- ▶ 徐树方, *矩阵计算的理论与方法*, 1995. [122]

0.1 计算数学介绍

1947 年, Von Neumann 和 Goldstine 在 Bulletin of the AMS (美国数学会通报) 上发表了题为 “Numerical inverting of matrices of high order” (高阶矩阵的数值求逆) 的著名论文 [110], 开启了现代计算数学的研究. 半个多世纪以来, 伴随着计算机技术的不断进步, 计算数学得到了蓬勃发展, 并逐渐成为了一个独立和重要的学科.

通俗地讲, 科学计算就是通过数学建模将实际问题转化为数学问题, 然后对数学问题进行离散和数值求解, 从而得到原问题的近似解, 同时对得到的近似解进行误差估计, 以确保近似解的可靠性. 科学计算利用先进的计算能力和数学知识解决复杂的科学工程问题, 它融数学建模、算法设计、软件开发和数值模拟为一体, 是计算机实现其在高科技领域中的应用的必不可少的纽带和工具. 计算已不仅仅只是作为验证理论模型正确性的手段, 大量的事例表明, 它已成为重大科学发现的一种重要手段 [125]. 科学计算已经和理论与实验研究一起, 成为当今世界科学技术创新的主要方式 [129], 也是当前公认的从事现代科学研究的第三种方法.

高性能科学计算在国家安全和科技创新等方面的重要作用也日益受到世界各国的重视. 欧美等国家已投入巨资, 大力发展先进的计算方法, 研制大型的实用软件. 2005 年 6 月, 美国总统信息技术咨询委员会 (President’s Information Technology Advisory Committee) 在给美国总统提交的报告《计算科学: 确保美国竞争力》(Computational Science: Ensuring America’s

Competitiveness) 中明确阐述了计算科学的重要性. 报告认为, 虽然计算本身也是一门学科, 但是其有促进其他学科发展的作用, 21 世纪科学上最重要的、经济上最有前途的前沿研究都有可能通过先进的计算技术和计算科学而得到解决. 报告还认为, 在迅猛发展的高性能计算技术推动下, 计算科学将是 21 世纪确保国家核心竞争能力的战略技术之一, 而科学计算则是计算科学中最主要的内容.

科学计算的核心是计算数学 [125]. 计算数学, 也称数值分析或计算方法, 主要研究各种数学问题的有效数值计算方法及其相关的数学理论, 包括算法的设计与分析 (收敛性, 稳定性, 复杂性等). 其研究内容有数值代数 (线性和非线性), 数值逼近 (插值, 函数逼近和数据拟合), 数值积分与数值微分, 微分方程数值解 (常微分方程, 偏微分方程), 最优化理论与方法等.

 有关计算数学和数值线性代数的定义可以参考 Golub 教授撰写的 History of numerical linear algebra: A personal view [45], 或 Trefethen 教授撰写的 Numerical analysis [102].

计算数学的主要任务

- 算法设计: 构造求解各种数学问题的高效数值方法.
- 算法分析: 研究数值方法的收敛性、稳定性、复杂性、计算精度等.
- 算法实现: 编程实现、软件开发.

一个好的数值方法一般需满足以下几点:

- 有可靠的理论分析, 即收敛性、稳定性等有数学理论保证.
- 有良好的计算复杂性 (时间和空间).
- 易于在计算机上编程实现.
- 要有具体的数值试验来证明算法是行之有效的.

 关于术语“数值方法”或“数值算法”, 在本讲义中, 一般情况下我们将不加区分地使用.

 数值方法通常都是近似方法, 求出的解是有误差的, 因此误差分析也很重要.



0.2 数值线性代数

If any other mathematical topic is as fundamental to the mathematical sciences as calculus and differential equations, it is numerical linear algebra.

— Trefethen & Bau III [103], 1997.

- 数值代数, 包含数值线性代数和数值非线性代数, 是计算数学的基础 [125].

- **矩阵计算** (即**数值线性代数**) 研究的基本问题有:

- 线性方程组求解

$$Ax = b, \quad A \in \mathbb{R}^{n \times n} \text{ 非奇异.}$$

- 线性最小二乘问题

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2, \quad A \in \mathbb{R}^{m \times n}.$$

- 矩阵特征值问题

$$Ax = \lambda x, \quad A \in \mathbb{R}^{n \times n}, \lambda \in \mathbb{C}, x \in \mathbb{C}^n, x \neq 0.$$

- 矩阵奇异值问题

$$A^T Ax = \sigma^2 x, \quad A \in \mathbb{R}^{m \times n}, \sigma \geq 0, x \in \mathbb{R}^n, x \neq 0.$$

- 其它问题: 广义特征值问题, 二次特征值问题, 非线性特征值问题, 矩阵方程, 特征值反问题, 张量计算, 随机数值线性代数, 混合精度计算, 高性能计算, 等等.

二十世纪最具影响力的十大算法 (Top 10 Algorithms)

“We tried to assemble the 10 algorithms with the greatest influence on the development and practice of science and engineering in the 20th century” — Editors

1. Metropolis Algorithm for Monte Carlo / Monte Carlo method (1946)
2. Simplex Method for Linear Programming (1947)
3. Krylov Subspace Iteration Methods (1950)
4. The Decompositional Approach to Matrix Computations (1951)
5. The Fortran Optimizing Compiler (1957)
6. QR Algorithm for Computing Eigenvalues (1959-61)
7. Quicksort Algorithm for Sorting (1962)
8. Fast Fourier Transform (1965)
9. Integer Relation Detection Algorithm (1977)
10. Fast Multipole Method (1987)

(下划线: 属于数值线性代数; 波浪线: 与数值线性代数密切相关.)

- ▷ J. Dongarra and F. Sullivan, Guest Editors Introduction to the top 10 algorithms, Computing in Science & Engineering, 2 (2000), 22–23. <https://doi.org/10.1109/MCISE.2000.814652>
- ▷ B. A. Cipra, The Best of the 20th Century: Editors Name Top 10 Algorithms, SIAM News,



Volume 33, Number 4, 2000. <https://archive.siam.org/pdf/news/637.pdf>

Nicholas J. Higham 版本 (2016)

1. Newton and quasi-Newton methods
2. Matrix factorizations (LU, Cholesky, QR)
3. Singular value decomposition, QR and QZ algorithms
4. Monte-Carlo methods
5. Fast Fourier Transform
6. Krylov Subspace Iteration Methods
7. JPEG
8. PageRank
9. Simplex method
10. Kalman filter

▷ N. Higham, The Top 10 Algorithms in Applied Mathematics, 2016. <https://nhigham.com/2016/03/29/the-top-10-algorithms-in-applied-mathematics/>

• 矩阵计算常用方法 (技术, 技巧或工具)

- 矩阵分解: 比如 LU 分解, Cholesky 分解, QR 分解, Schur 分解, SVD 分解等 ¹
- 矩阵分裂: 比如基本迭代法, 交替方向法, 预处理子构造等
- 矩阵降维: 比如子空间迭代法



学习和设计算法的几点建议

- ▶ 问题的特殊结构对算法的设计具有非常重要的影响.
- ▶ 动手编写程序对理解和掌握算法非常有帮助.
- ▶ 在实际应用中, 要充分利用现有的优秀程序库.

¹The Big Six Matrix Factorizations, Nick Higham, 2022.



第一讲 线性代数基础

这里介绍本讲义所涉及的线性代数基础知识, 特别是线性空间和矩阵的相关基础知识.

本讲主要参考文献

- ▶ R. A. Horn and C. R. Johnson, *Matrix Analysis*, 1985. 2nd edition, 2013. [63]
- ▶ R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, 1991. [62]
- ▶ 戴华, 矩阵论, 2001. [124]

1.1 线性空间与内积空间

1.1.1 线性空间

线性空间是线性代数最基本的概念之一, 它是定义在某个数域上并满足一定条件的集合. 我们首先给出数域的概念.

定义 1.1 (数域) 设 $\mathbb{F}$ 是包含 0 和 1 的一个数集, 如果 $\mathbb{F}$ 中的任意两个数的和, 差, 积, 商 (除数不为 0) 仍然在 $\mathbb{F}$ 中, 则称 $\mathbb{F}$ 为一个**数域**.

例 1.1 常见的数域有: 有理数域 $\mathbb{Q}$, 实数域 $\mathbb{R}$ 和复数域 $\mathbb{C}$.

 本讲义只考虑实数域 $\mathbb{R}$ 和复数域 $\mathbb{C}$.

定义 1.2 (线性空间) 设 S 是一个非空集合, $\mathbb{F}$ 是一个数域 ($\mathbb{C}$ 或 $\mathbb{R}$). 在 S 上定义一种代数运算, 称为**加法**, 记为 “+” (即对任意 $x, y \in S$, 都存在唯一的 $z \in S$, 使得 $z = x + y$), 并定义一个从 $\mathbb{F} \times S$ 到 S 的代数运算, 称为**数乘**, 记为 “ $\cdot$ ” (即对任意 $\alpha \in \mathbb{F}$ 和任意 $x \in S$, 都存在唯一的 $y \in S$, 使得 $y = \alpha \cdot x$). 如果这两个运算满足下面的规则, 则称 $(S, +, \cdot)$ 是数域 $\mathbb{F}$ 上的一个**线性空间** (通常简称 S 是数域 $\mathbb{F}$ 上的一个线性空间):

- 加法四条规则

- (1) **交换律**: $x + y = y + x$, $\forall x, y \in S$;
- (2) **结合律**: $(x + y) + z = x + (y + z)$, $\forall x, y, z \in S$;
- (3) **零元素**: 存在一个元素 0, 使得 $x + 0 = x$, $\forall x \in S$;
- (4) **逆运算**: 对任意 $x \in S$, 都存在**负元素** $y \in S$, 使得 $x + y = 0$, 记 $y = -x$;

- 数乘四条规则

- (1) **单位元**: $1 \cdot x = x$, $1 \in \mathbb{F}$, $\forall x \in S$;

(2) **结合律**: $\alpha \cdot (\beta \cdot x) = (\alpha\beta) \cdot x, \quad \forall \alpha, \beta \in \mathbb{F}, x \in \mathbb{S};$

(3) **分配律**: $(\alpha + \beta) \cdot x = \alpha \cdot x + \beta \cdot x, \quad \forall \alpha, \beta \in \mathbb{F}, x \in \mathbb{S};$

(4) **分配律**: $\alpha \cdot (x + y) = \alpha \cdot x + \alpha \cdot y, \quad \forall \alpha \in \mathbb{F}, x, y \in \mathbb{S}.$

为了表示方便, 通常省略数乘符号, 即将 $\alpha \cdot x$ 写成 αx .

例 1.2 常见的线性空间:

- $\mathbb{R}^n \rightarrow$ 所有 n 维实向量组成的集合, 是 $\mathbb{R}$ 上的线性空间.
- $\mathbb{C}^n \rightarrow$ 所有 n 维复向量组成的集合, 是 $\mathbb{C}$ 上的线性空间.
- $\mathbb{R}^{m \times n} \rightarrow$ 所有 $m \times n$ 阶实矩阵组成的集合, 是 $\mathbb{R}$ 上的线性空间.
- $\mathbb{C}^{m \times n} \rightarrow$ 所有 $m \times n$ 阶复矩阵组成的集合, 是 $\mathbb{C}$ 上的线性空间.
- $\mathbb{P}_n \rightarrow$ 所有次数不超过 n 的多项式组成的集合.
- $C[a, b] \rightarrow$ 区间 $[a, b]$ 上所有连续函数组成的集合.
- $C^p[a, b] \rightarrow$ 区间 $[a, b]$ 上所有 p 次连续可微函数组成的集合.

 为了表述方便, 线性空间中的元素通常称为**向量**.

线性相关性和线性空间的维数

设 $\mathbb{S}$ 是数域 $\mathbb{F}$ 上的一个线性空间, $x_1, x_2, \dots, x_k$ 是 $\mathbb{S}$ 中的一组向量. 如果存在 k 个不全为零的数 $\alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{F}$, 使得

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_k x_k = 0,$$

则称 $x_1, x_2, \dots, x_k$ **线性相关**, 否则就是**线性无关**.

设 $x_1, x_2, \dots, x_k$ 是 $\mathbb{S}$ 中的一组向量, 如果 $x \in \mathbb{S}$ 可以表示为

$$x = \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_k x_k,$$

其中 $\alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{F}$, 则称 x 可以由 $x_1, x_2, \dots, x_k$ **线性表示** (或**线性表出**), 或者称 x 是 $x_1, x_2, \dots, x_k$ 的**线性组合**, $\alpha_1, \alpha_2, \dots, \alpha_k$ 称为**线性表出系数**.

设向量组 $\{x_1, x_2, \dots, x_m\}$, 如果存在其中的 r ($r \leq m$) 个线性无关向量 $x_{i_1}, x_{i_2}, \dots, x_{i_r}$, 使得所有向量都可以由它们线性表示, 则称 $x_{i_1}, x_{i_2}, \dots, x_{i_r}$ 为向量组 $\{x_1, x_2, \dots, x_m\}$ 的一个**极大线性无关组**, 并称这组向量的**秩**为 r , 记为 $\text{rank}(\{x_1, x_2, \dots, x_m\}) = r$.

设 $x_1, x_2, \dots, x_n$ 是 $\mathbb{S}$ 中的一组线性无关向量, 如果 $\mathbb{S}$ 中的任意一个向量都可以由 $x_1, x_2, \dots, x_n$ 线性表示, 则称 $x_1, x_2, \dots, x_n$ 是 $\mathbb{S}$ 的一组**基**, 并称 $\mathbb{S}$ 是 n 维的, 即 $\mathbb{S}$ 的**维数**为 n , 记为 $\dim(\mathbb{S}) = n$. 如果 $\mathbb{S}$ 中可以找到任意多个线性无关向量, 则称 $\mathbb{S}$ 是**无限维**的.



1.1.2 子空间与仿射子空间

子空间

设 S 是一个线性空间, W 是 S 的一个非空子集. 如果 W 关于 S 上的加法和数乘也构成一个线性空间, 则称 W 为 S 的一个**线性子空间**, 简称**子空间**.

例 1.3 设 S 是一个线性空间, 则由零向量组成的子集 $\{0\}$ 是 S 的一个子空间, 称为零子空间. 另外, S 本身也是 S 的子空间. 这两个特殊的子空间称为 S 的**平凡子空间**, 其他子空间都是**非平凡子空间**.

定理 1.1 (子空间的判别) 设 S 是数域 $\mathbb{F}$ 上的一个线性空间, W 是 S 的一个非空子集, 则 W 是 S 的一个子空间的充要条件是 W 关于加法和数乘封闭, 即

- (1) 对任意 $x, y \in W$, 有 $x + y \in W$;
- (2) 对任意 $\alpha \in \mathbb{F}$ 和任意 $x \in W$, 有 $\alpha x \in W$.

设 S_1, S_2 是线性空间 S 的两个子空间, 则它们的**和**定义为

$$S_1 + S_2 \triangleq \{x + y : x \in S_1, y \in S_2\}.$$

容易证明 $S_1 + S_2$ 也是 S 的子空间. 下面是关于子空间的维数的一个重要性质.

定理 1.2 (维数公式) 设 S_1, S_2 是线性空间 S 的两个有限维子空间, 则 $S_1 + S_2$ 和 $S_1 \cap S_2$ 也都是 S 的子空间, 且

$$\dim(S_1 + S_2) + \dim(S_1 \cap S_2) = \dim(S_1) + \dim(S_2).$$

直和

设 S_1, S_2 是线性空间 S 的两个子空间, 如果 $S_1 + S_2$ 中的任一元素 x 都可以唯一表示成

$$x = x_1 + x_2, \quad x_1 \in S_1, x_2 \in S_2,$$

则称 $S_1 + S_2$ 为**直和**, 记为 $S_1 \oplus S_2$.

关于子空间的直和的判定, 有下面的结论.

定理 1.3 设 S_1, S_2 是线性空间 S 的两个子空间, 则下面的论述等价:

- (1) $S_1 + S_2$ 是直和;
- (2) $S_1 + S_2$ 中的零元素表示方法唯一, 即若 $0 = x_1 + x_2, x_1 \in S_1, x_2 \in S_2$, 则 $x_1 = x_2 = 0$;
- (3) $S_1 \cap S_2 = \{0\}$;
- (4) $\dim(S_1) + \dim(S_2) = \dim(S_1 + S_2)$.



定理 1.4 设 S_1 是线性空间 S 的一个子空间, 则存在 S 的另一个子空间 S_2 , 使得

$$S = S_1 \oplus S_2.$$

我们称 S_2 是 S_1 关于 S 的**补空间**. 显然 S_1 也是 S_2 的**补空间**, 因此它们是互补的.

 **思考:** 补空间是否唯一?

仿射子空间

定义 1.3 (仿射子空间, Affine Subspace) 设 S_1 是线性空间 S 的一个子空间, $z \in S$, 则集合

$$W = z + S_1 \triangleq \{x : x = z + v, \forall v \in S_1\}$$

是 S 的**仿射子空间** (或**线性流形**, linear manifold), 其中 S_1 称为仿射子空间的**方向** (或**方向空间**, direction space), z 称为**支撑点** (support point).

 若 $z \notin S_1$, 则 $0 \notin W$, 因此仿射子空间通常不是线性子空间.

例 1.4 设 $A \in \mathbb{R}^{m \times n}$ ($m < n$) 满秩, 则

- 齐次线性方程组 $Ax = 0$ 的解集是一个线性子空间;
- 非齐次相容线性方程组 $Ax = b$ 的解集是一个仿射子空间;
- 特别地, 单个相容线性方程 $\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = \beta$ 的解空间构成 $\mathbb{R}^n$ 的一个**超平面** (hyperplane).

1.1.3 内积与内积空间

内积空间就是带有内积运算的线性空间.

定义 1.4 (内积与内积空间) 设 S 是数域 $\mathbb{F}$ ($\mathbb{C}$ 或 $\mathbb{R}$) 上的一个线性空间, 定义一个从 $S \times S$ 到 $\mathbb{F}$ 的代数运算, 记为 “ $(\cdot, \cdot)$ ”, 即对任意 $x, y \in S$, 都存在唯一的 $f \in \mathbb{F}$, 使得 $f = (x, y)$. 如果该运算满足

- (1) $(y, x) = \overline{(x, y)}, \quad \forall x, y \in S;$
- (2) $(x + y, z) = (x, z) + (y, z), \quad \forall x, y, z \in S;$
- (3) $(\alpha x, y) = \alpha(x, y), \quad \forall \alpha \in \mathbb{F}, x, y \in S;$
- (4) $(x, x) \geq 0$, 等号当且仅当 $x = 0$ 时成立;

则称 $(\cdot, \cdot)$ 为 S 上的一个**内积** (inner product), 定义了内积的线性空间称为**内积空间**.



🔴 内积有时也称为 **标量积** (scalar product).

🔴 定义在实数域 $\mathbb{R}$ 上的内积空间称为**欧氏空间** (Euclidean space), 定义在复数域 $\mathbb{C}$ 上的内积空间称为**酉空间**.

🔴 $\overline{(x, y)}$ 表示 (x, y) 的共轭. 当 $\mathbb{F} = \mathbb{R}$ 时, 条件 (1) 即为 $(y, x) = (x, y)$.

🔴 内积可以看作是从线性空间 $\mathcal{S}$ 到数域 $\mathbb{F}$ 的**二元函数**.

例 1.5 设 $(\cdot, \cdot)$ 是 $\mathcal{S}$ 上的一个内积, 则容易验证:

$$(x, \alpha y) = \bar{\alpha}(x, y), \quad \forall \alpha \in \mathbb{F}, x, y \in \mathcal{S}.$$

例 1.6 在 $\mathbb{C}^n$ 上定义内积

$$(x, y) \triangleq y^* x = \sum_{i=1}^n x_i \bar{y}_i, \quad (\text{注意是 } y^* x, \text{ 不是 } x^* y)$$

则 $\mathbb{C}^n$ 构成一个内积空间. 类似的, $\mathbb{R}^n$ 上可以定义内积

$$(x, y) \triangleq y^T x = \sum_{i=1}^n x_i y_i.$$

这种方式定义的内积称为 **欧几里得内积** (Euclidean inner product), 或 **点积** (dot product), 或 **标准内积** (standard inner product), 这也是 $\mathbb{C}^n$ 和 $\mathbb{R}^n$ 上的常用内积 [63].

🔴 $\mathbb{C}^n$ 或 $\mathbb{R}^n$ 上的内积由无穷多个. 在本讲义中, 如果没有特别指出, 涉及到 $\mathbb{C}^n$ 或 $\mathbb{R}^n$ 上的内积时, 默认就是指上面定义的内积.

例 1.7 对任意 $A, B \in \mathbb{R}^{m \times n}$, 定义

$$(A, B) \triangleq \text{tr}(B^T A),$$

其中 $\text{tr}(\cdot)$ 表示矩阵的**迹**, 即对角线元素之和, 则可以证明 (A, B) 是一个内积, 因此 $\mathbb{R}^{m \times n}$ 构成一个欧氏空间. (留作练习)

1.1.4 正交与正交补

有了内积以后, 我们就可以定义正交.

定义 1.5 (正交) 设 $\mathcal{S}$ 是内积空间, $x, y \in \mathcal{S}$, 如果 $(x, y) = 0$, 则称 x 与 y **正交**, 记为 $x \perp y$; 设 $\mathcal{S}_1$ 是 $\mathcal{S}$ 的子空间, $x \in \mathcal{S}$, 如果对任意 $y \in \mathcal{S}_1$ 都有 $(x, y) = 0$, 则称 x 与 $\mathcal{S}_1$ **正交**, 记为 $x \perp \mathcal{S}_1$; 设 $\mathcal{S}_1, \mathcal{S}_2$ 是 $\mathcal{S}$ 的两个子空间, 如果对任意 $x \in \mathcal{S}_1$, 都有 $x \perp \mathcal{S}_2$, 则称 $\mathcal{S}_1$ 与 $\mathcal{S}_2$ **正交**, 记为 $\mathcal{S}_1 \perp \mathcal{S}_2$.

定理 1.5 设 $\mathcal{S}_1, \mathcal{S}_2$ 是内积空间 $\mathcal{S}$ 的两个子空间, 如果 $\mathcal{S}_1 \perp \mathcal{S}_2$, 则 $\mathcal{S}_1 + \mathcal{S}_2$ 是直和.



(留作课外自习)

定义 1.6 (正交补, Orthogonal Complement) 设 S_1 是内积空间 S 的一个子空间, 则 S_1 的**正交补**定义为

$$S_1^\perp \triangleq \{ x \in S : x \perp S_1 \},$$

即 S 中所有与 S_1 正交的元素组成的集合.

容易验证, $S_1^\perp$ 也是 S 的一个子空间. 另外, 我们还可以得到下面的结论.

定理 1.6 设 S_1 是内积空间 S 的一个有限维子空间, 则 $S_1^\perp$ 存在唯一, 且

$$S = S_1 \oplus S_1^\perp.$$

例 1.8 设 $x_1, x_2, \dots, x_n \in \mathbb{R}^n$. 若 $x_1, x_2, \dots, x_n$ 线性无关, 则 $\{x_1, x_2, \dots, x_n\}$ 构成 $\mathbb{R}^n$ 的一组基. 进一步, 如果 $x_1, x_2, \dots, x_n$ 相互正交, 即

$$(x_i, x_j) = x_j^T x_i = 0, \quad i, j = 1, 2, \dots, n, \quad i \neq j,$$

则称它们是一组**正交基**. 如果还满足

$$(x_i, x_i) = x_i^T x_i = 1, \quad i = 1, 2, \dots, n,$$

则称它们是一组**标准正交基**或**规范正交基**. 特别地, 记 e_i 为单位矩阵的第 i 列, 则 $\{e_1, e_2, \dots, e_n\}$ 构成 $\mathbb{R}^n$ 的一组标准正交基, 这组基通常称为**自然基**.

 任何一组基都可以通过 Gram-Schmidt 正交化过程构造出一组标准正交基.

1.2 矩阵与投影

注记

为了讨论方便, 如果没有特别指出, 本节仅考虑实数情形, 对于复数情形, 我们可以得到类似的结论.

矩阵与线性变换

 设 f 是 $\mathbb{R}^n \rightarrow \mathbb{R}^m$ 的**线性变换**, 则存在唯一的矩阵 $A \in \mathbb{R}^{m \times n}$ 使得

$$f(x) = Ax, \quad \forall x \in \mathbb{R}^n,$$

即线性变换与矩阵一一对应.

 设 f 是 $\mathbb{R}^n \rightarrow \mathbb{R}^m$ 的**仿射变换** (Affine Transformation), 则 f 可表示为

$$f(x) = Ax + b, \quad \forall x \in \mathbb{R}^n,$$



其中 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

1.2.1 矩阵的秩

设 $A \in \mathbb{R}^{m \times n}$, 则称 A 的列向量组的秩为 A 的**列秩**, 称 A 的行向量组的秩为 A 的**行秩**. 可以验证, 矩阵 A 的列秩与行秩是相等的. 因此我们统一称它们为矩阵 A 的**秩**, 记为 $\text{rank}(A)$.

定理 1.7 设 $A \in \mathbb{R}^{m \times n}$, 则 $\text{rank}(A) = k$ ($0 \leq k \leq \min\{m, n\}$) 的充要条件是 A 存在非奇异的 k 阶子矩阵, 且所有 $k+1$ 阶子矩阵都奇异. (留作课外自习, 可参见 [119])

关于矩阵的秩, 我们有下面的基本性质.

定理 1.8 设 $A, B \in \mathbb{R}^{m \times n}$, 则

- $\text{rank}(A) \leq \min\{m, n\}$;
- $\text{rank}(A^T) = \text{rank}(A)$;
- $\text{rank}(A^T A) = \text{rank}(A A^T) = \text{rank}(A)$;
- $\text{rank}(A + B) \leq \text{rank}(A) + \text{rank}(B)$;
- 对任意非奇异矩阵 $P \in \mathbb{R}^{m \times m}$ 和 $Q \in \mathbb{R}^{n \times n}$, 有

$$\text{rank}(PA) = \text{rank}(AQ) = \text{rank}(PAQ) = \text{rank}(A).$$

下面是关于矩阵的秩的一些常用性质.

定理 1.9 (秩分解) 设 $\text{rank}(A) = \ell$, 则存在非奇异矩阵 $P \in \mathbb{R}^{m \times m}$ 和 $Q \in \mathbb{R}^{n \times n}$ 使得

$$A = P \begin{bmatrix} I_\ell & 0 \\ 0 & 0 \end{bmatrix} Q.$$

进一步, $\text{rank}(A) = \text{rank}(B)$ 当且仅当存在非奇异矩阵 $P \in \mathbb{R}^{m \times m}$ 和 $Q \in \mathbb{R}^{n \times n}$ 使得 $A = PBQ$.

推论 1.10 (满秩分解) 设 $\text{rank}(A) = \ell$, 则存在非奇异矩阵 $F \in \mathbb{R}^{m \times \ell}$ 和 $G \in \mathbb{R}^{\ell \times n}$ 使得

$$A = FG.$$

定理 1.11 设 $A \in \mathbb{R}^{m \times k}$, $B \in \mathbb{R}^{k \times n}$, 则

$$\text{rank}(A) + \text{rank}(B) - k \leq \text{rank}(AB) \leq \min\{\text{rank}(A), \text{rank}(B)\}.$$

推论 1.12 设 $A \in \mathbb{R}^{m \times k}$, $B \in \mathbb{R}^{k \times m}$, $k \leq m$. 若 A 和 B 都是满秩矩阵, 则

$$\text{rank}(AB) = \text{rank}(BA) = \text{rank}(A) = \text{rank}(B) = k.$$



张成的线性空间

设 $x_1, x_2, \dots, x_k \in \mathbb{R}^n$, 记

$$\text{span}\{x_1, x_2, \dots, x_k\} \triangleq \{ \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_k x_k : \alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{R} \},$$

则 $\text{span}\{x_1, x_2, \dots, x_k\}$ 构成 $\mathbb{R}^n$ 的一个线性子空间, 称为由 $x_1, x_2, \dots, x_k$ **张成的线性空间**. 特别地, 记 $\text{span}(A)$ 为由 A 的所有列向量张成的线性空间.

矩阵 A 相关的四个子空间

设 $A \in \mathbb{R}^{m \times n}$, 则 A 可以看作是从 $\mathbb{R}^n$ 到 $\mathbb{R}^m$ 的一个线性变换 (或线性映射, 线性算子)¹, 即

$$A: x \rightarrow Ax$$

我们分别称

$$\text{Ker}(A) \triangleq \{ x \in \mathbb{R}^n : Ax = 0 \} \subseteq \mathbb{R}^n$$

和

$$\text{Ran}(A) \triangleq \{ y \in \mathbb{R}^m : y = Ax, x \in \mathbb{R}^n \} \subseteq \mathbb{R}^m$$

为 A 的**零空间 (核)** 和**像空间 (列空间, 值域)**, 称

$$\text{Ker}(A^T) \triangleq \{ y \in \mathbb{R}^m : A^T y = 0 \} \subseteq \mathbb{R}^m$$

和

$$\text{Ran}(A^T) \triangleq \{ x \in \mathbb{R}^n : x = A^T y, y \in \mathbb{R}^m \} \subseteq \mathbb{R}^n$$

为 A 的**左零空间**和**行空间**. 可以证明, $\text{Ker}(A)$ 和 $\text{Ran}(A^T)$ 是 $\mathbb{R}^n$ 的线性子空间, $\text{Ran}(A)$ 和 $\text{Ker}(A^T)$ 是 $\mathbb{R}^m$ 的线性子空间, 且 $\text{Ran}(A) = \text{span}(A)$.

定理 1.13 设 $A \in \mathbb{R}^{m \times n}$, 则有

- $\dim(\text{Ran}(A)) = \dim(\text{Ran}(A^T)) = \text{rank}(A)$;
- $\dim(\text{Ker}(A)) + \dim(\text{Ran}(A^T)) = n$;
- $\text{Ran}(A^T A) = \text{Ran}(A^T)$, $\text{Ker}(A^T A) = \text{Ker}(A)$.

例 1.9 设 $A \in \mathbb{R}^{m \times n}$, 则

$$\text{Ran}(A)^\perp = \text{Ker}(A^T).$$

(板书)

 类似地, 有 $\text{Ker}(A)^\perp = \text{Ran}(A^T)$.

 结论在复数域也成立: 如果 $A \in \mathbb{C}^{m \times n}$, 则 $\text{Ran}(A)^\perp = \text{Ker}(A^*)$, $\text{Ker}(A)^\perp = \text{Ran}(A^*)$.

¹严格地讲, 是某个线性变换在某组基下的矩阵



例 1.10 设 $A \in \mathbb{R}^{m \times n}$, 则由例 1.9 可知

$$\text{Ker}(A) \oplus \text{Ran}(A^T) = \mathbb{R}^n, \quad \text{Ker}(A^T) \oplus \text{Ran}(A) = \mathbb{R}^m.$$

例 1.11 (矩阵的秩与齐次线性方程组基础解系) 设 $A \in \mathbb{R}^{m \times n}$ 的秩为 $k \leq \min\{m, n\}$, 则齐次线性方程组 $Ax = 0$ 的基础解系所含解的个数为 $n - k$, 也即 $\dim(\text{Ker}(A)) = n - k$.

1.2.2 特征值与特征向量

定义 1.7 (特征值和特征向量) 设 $A \in \mathbb{R}^{n \times n}$. 若存在 $\lambda \in \mathbb{C}$ 和非零向量 $x, y \in \mathbb{C}^n$, 满足

$$Ax = \lambda x,$$

则称 λ 为 A 的**特征值**, x 为 A 对应于 λ 的**特征向量**, 并称 (λ, x) 为 A 的一个**特征对** (eigenpair).

 **思考:** 从定义 1.7 能否判断矩阵 A 是否一定存在特征值和特征向量?

矩阵特征值也可以通过特征多项式来定义.

定义 1.8 (特征多项式和特征值) 设 $A \in \mathbb{R}^{n \times n}$, 记 $p_A(\lambda) \triangleq \det(A - \lambda I)$. 易知 $p_A(\lambda)$ 是关于 λ 的 n 次多项式, 我们称之为 A 的**特征多项式**, 其在复数域中的零点称为 A 的**特征值**.

我们知道, n 次多项式在复数域中一定存在 n 个零点 (不考虑重复零点), 因此根据定义 1.8, 一个 n 阶矩阵一定存在 n 个特征值.

下面是关于特征多项式的一个重要性质.

定理 1.14 (Cayley-Hamilton) 设 $p_A(\lambda)$ 是 $A \in \mathbb{R}^{n \times n}$ 的特征多项式, 则 $p_A(A) = 0$.

由 Cayley-Hamilton 定理 1.14 可知, 总存在多项式 $p(t)$ 使得 $p(A) = 0$. 这种特殊多项式称为零化多项式 (annihilating polynomial).

定义 1.9 (零化多项式和最小多项式) 设 $A \in \mathbb{R}^{n \times n}$, 如果多项式 $p(t)$ 满足 $p(A) = 0$, 则称其为 A 的**零化多项式**, 其中次数最低的首一 (即首项系数为 1) 多项式称为 A 的**最小多项式**.

最小多项式是矩阵的一个重要概念, 在线性代数中有着重要的应用. 容易证明, 最小多项式是存在唯一的, 而且次数不超过 n . 计算最小多项式通常是非常困难的, 一种方法是通过 Jordan 标准型来计算, 见定理 1.46.

定义 1.10 (特征子空间) 设 λ 是 $A \in \mathbb{R}^{n \times n}$ 的特征值, 令

$$\mathbb{S}_\lambda = \{x \in \mathbb{C}^n \mid Ax = \lambda x\},$$



则可以验证 S_λ 为 $\mathbb{C}^n$ 的子空间, 称为 A 对应于特征值 λ 的一个**特征子空间**.

关于特征值的几点说明

- 只有当 A 是方阵时, 特征值与特征向量才有定义.
- 实矩阵的特征值与特征向量也有可能是复数.
- 一个 n 阶矩阵总是存在 n 个特征值 (其中可能有相等的), 通常记为 $\lambda_1, \lambda_2, \dots, \lambda_n$.
- 所有特征值组成的集合称为矩阵的**谱**, 记为 $\sigma(A)$, 即

$$\sigma(A) \triangleq \{ \lambda_1, \lambda_2, \dots, \lambda_n \}.$$

- 特征值有代数重数 (所对应的特征多项式零点的重数) 和几何重数 (所对应的特征子空间的维数), 几何重数不超过代数重数.
- 相似变换不改变矩阵的特征值.
- 合同变换不改变矩阵的惯性指数 (即正特征值、负特征值和零特征值的个数).



思考: 设 $A \in \mathbb{R}^{n \times n}$, 则 A^T 与 A 的特征值和特征向量是什么关系?

设 A 非奇异, 则 A^{-1} 与 A 的特征值和特征向量是什么关系?

更一般地, 设 $p(t)$ 是一个多项式, 则 $p(A)$ 与 A 的特征值和特征向量是什么关系?

定义 1.11 (谱半径) 设 $A \in \mathbb{R}^{n \times n}$ 或 $\mathbb{C}^{n \times n}$, 其**谱半径**定义如下:

$$\rho(A) \triangleq \max_{\lambda \in \sigma(A)} |\lambda|,$$

其中 $\sigma(A)$ 表示 A 的谱.

下面给出特征值的一些常用性质.

定理 1.15 设 $A \in \mathbb{R}^{n \times n}$, 则有

$$\lambda_1 \lambda_2 \cdots \lambda_n = \det(A), \quad \lambda_1 + \lambda_2 + \cdots + \lambda_n = \operatorname{tr}(A),$$

其中 $\det(A)$ 表示 A 的行列式, $\operatorname{tr}(A)$ 表示 A 的**迹** (对角线元素之和), 即

$$\operatorname{tr}(A) \triangleq a_{11} + a_{22} + \cdots + a_{nn}.$$

(留作课外自习, 高次多项式的韦达定理)

推论 1.16 若 A 与 B 相似, 则 $\operatorname{tr}(A) = \operatorname{tr}(B)$, 即相似矩阵具有相同的迹.

定义 1.12 设 $A \in \mathbb{R}^{n \times n}$. 若存在一个非奇异矩阵 $X \in \mathbb{C}^{n \times n}$, 使得

$$X^{-1}AX = \Lambda, \tag{1.1}$$



其中 $\Lambda \in \mathbb{C}^{n \times n}$ 是对角矩阵, 则称 A 是**可对角化**的, 矩阵 Λ 的对角线元素即为 A 的特征值, 分解 (1.1) 称为矩阵 A 的**特征值分解**或**谱分解**.

 并非所有矩阵都可以对角化, 比如 $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ 是无法对角化的.

定理 1.17 设 $A \in \mathbb{R}^{n \times n}$, 则

- (1) A 可对角化当且仅当 A 有 n 个线性无关的特征向量;
- (2) A 可对角化当且仅当 A 的所有特征值的代数重数与几何重数都相等;
- (3) 若 A 有 n 个互不相等的特征值, 则 A 可对角化.

例 1.12 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 则 A 可对角化, 而且可以**正交对角化**, 即存在正交矩阵 $Q \in \mathbb{R}^{n \times n}$ 使得

$$A = Q\Lambda Q^T.$$

 酉矩阵与正交矩阵: 设 $Q \in \mathbb{C}^{n \times n}$, 若 $Q^*Q = I$, 即所有列向量都是单位向量, 且相互正交, 则称 Q 为**酉矩阵**. 如果 Q 是实矩阵, 则称为**正交矩阵**.

根据多项式零点关于多项式系数的连续性, 我们可以得到下面的结论.

定理 1.18 矩阵的特征值关于矩阵元素是连续的, 即当矩阵的元素发生变化时, 其特征值的变化是连续的.

1.2.3 特征值的粗略估计

矩阵特征值在科学与工程计算中应用非常广泛, 但直接计算特征值通常比较困难, 特别是当矩阵规模非常大时. 本小节给出两个估计特征值所在范围的方法. 我们这里假定 A 是复矩阵.

Bendixson 估计方法

设 $A \in \mathbb{C}^{n \times n}$, 我们记

$$H \triangleq \frac{1}{2}(A + A^*), \quad S \triangleq \frac{1}{2}(A - A^*).$$

易知 $A = H + S$, 且

$$H^* = H, \quad S^* = -S,$$

即 H 是 Hermite 的, S 是 Skew-Hermite (斜 Hermite) 的. 我们分别称 H 和 S 为 A 的**Hermite 部分**和**Skew-Hermite 部分**.



设 (λ, x) 是 A 的一个特征对, 且 $\|x\|_2 = 1$, 则

$$Ax = \lambda x \implies \lambda = \frac{x^* Ax}{x^* x} = x^* Hx + x^* Sx,$$

其中 $x^* Hx$ 是实数, $x^* Sx$ 是纯虚数. 因此我们有下面的结论.

定理 1.19 (Bendixson 定理) 设 $A \in \mathbb{C}^{n \times n}$, 则

$$\lambda_{\min}(H) \leq \operatorname{Re}(\lambda(A)) \leq \lambda_{\max}(H),$$

$$\lambda_{\min}(-\mathbf{i} S) \leq \operatorname{Im}(\lambda(A)) \leq \lambda_{\max}(-\mathbf{i} S),$$

其中 $\operatorname{Re}(\cdot)$ 和 $\operatorname{Im}(\cdot)$ 分别表示实部和虚部, $\mathbf{i}$ 是虚部单位.

这个定理告诉我们, 一个矩阵的特征值的实部的取值范围由其 Hermite 部分确定, 而虚部则由其 Skew-Hermite 部分确定.

Gerschgorin 圆盘估计方法

设 $A = [a_{ij}] \in \mathbb{C}^{n \times n}$, 定义集合

$$\mathcal{D}_i \triangleq \left\{ z \in \mathbb{C} : |z - a_{ii}| \leq \sum_{j=1, j \neq i}^n |a_{ij}| \right\}, \quad i = 1, 2, \dots, n. \quad (1.2)$$

我们称 $\mathcal{D}_i$ 为 A 的 **Gerschgorin 圆盘** (也称**盖尔圆盘**).

定理 1.20 (Gerschgorin 圆盘定理) 设 $A \in \mathbb{C}^{n \times n}$, 则 A 的所有特征值都包含在 A 的 Gerschgorin 圆盘的并集中, 即

$$\sigma(A) \subset \bigcup_{i=1}^n \mathcal{D}_i.$$

(板书)

将 A 的非对角线元素换成 τa_{ij} , 其中 $0 \leq \tau \leq 1$, 并利用特征值关于矩阵元素的连续性, 我们就可以得到下面的结论.

定理 1.21 设 $A \in \mathbb{C}^{n \times n}$, 如果 $\bigcup_{i=1}^n \mathcal{D}_i$ 可分解成两个互不相交的子集 S 和 T , 即

$$\bigcup_{i=1}^n \mathcal{D}_i = S \cup T \quad \text{且} \quad S \cap T = \emptyset,$$

并假定 S 由 k 个圆盘组成, 而 T 由其它 $n - k$ 个圆盘组成, 则 S 中恰好包含 A 的 k 个特征值 (重特征值按重数计算), 而 T 中则包含 A 的其它 $n - k$ 个特征值.

1.2.4 不变子空间

不变子空间在线性方程组求解 (如子空间迭代法) 和矩阵特征值计算 (如 QR 迭代法) 中有着重要的应用.



定义 1.13 设 $A \in \mathbb{R}^{n \times n}$, 子空间 $S \subseteq \mathbb{R}^n$. 若 $AS \subseteq S$, 即对任意 $x \in S$, 都有 $Ax \in S$, 则称 S 为 A 的一个**不变子空间**.

一类特殊的不变子空间就是由特征向量所张成的子空间.

定理 1.22 设 $x_1, x_2, \dots, x_m$ 是 A 的一组线性无关的特征向量, 则 $\text{span}\{x_1, x_2, \dots, x_m\}$ 是 A 的一个 m 维不变子空间.

下面的结论对矩阵特征值计算非常重要.

定理 1.23 设 $A \in \mathbb{R}^{n \times n}$, $X \in \mathbb{R}^{n \times k}$ 且 $\text{rank}(X) = k$, 则 $\text{span}(X)$ 是 A 的不变子空间的充要条件是存在一个矩阵 $B \in \mathbb{R}^{k \times k}$ 使得

$$AX = XB,$$

此时, B 的特征值都是 A 的特征值.

(板书)

推论 1.24 设 $A \in \mathbb{R}^{n \times n}$, $X \in \mathbb{R}^{n \times k}$ 且 $\text{rank}(X) = k$. 若存在一个矩阵 $B \in \mathbb{R}^{k \times k}$ 使得 $AX = XB$, 则 (λ, v) 是 B 的一个特征对当且仅当 (λ, Xv) 是 A 的一个特征对.

(留作练习)

 **思考:** 设 $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{k \times k}$, $k \leq n$. 若 B 的特征值都是 A 的特征值, 且具有相同的代数重数, 则是否存在满秩矩阵 $X \in \mathbb{R}^{n \times k}$, 使得 $AX = XB$?

1.2.5 投影变换与正交投影

投影与投影变换

设 S_1 和 S_2 是内积空间 S 的两个子空间, 且 $S = S_1 \oplus S_2$, 则 S 中的任意向量 x 都可唯一表示为

$$x = x_1 + x_2, \quad x_1 \in S_1, \quad x_2 \in S_2.$$

我们称 x_1 为 x 沿 S_2 到 S_1 上的**投影**, 记为 $x|_{S_1}$.

 需要指出的是, 由于 S_1 的补空间不唯一, 因此在讨论投影时一定要明确给定 S_2 .

例 1.13 设 $S_1 = \text{span}\{e_1\}$, $S_2 = \text{span}\{e_2\}$, $\tilde{S}_2 = \text{span}\{e\}$, 其中 $e_1 = [1, 0]^T$, $e_2 = [0, 1]^T$, $e = [1, 1]^T$. 于是有

$$\mathbb{R}^2 = S_1 \oplus S_2 = S_1 \oplus \tilde{S}_2.$$

向量 $x = [2, 3]^T$ 沿 S_2 到 S_1 上的投影是 $[2, 0]^T$, 而它沿 $\tilde{S}_2$ 到 S_1 上的投影是 $[-1, 0]^T$.



定义线性变换 $P: \mathbb{S} \rightarrow \mathbb{S}$ 如下:

$$Px = x|_{\mathbb{S}_1}, \quad \forall x \in \mathbb{S}.$$

称 P 是从 $\mathbb{S}$ 沿 $\mathbb{S}_2$ 到 $\mathbb{S}_1$ 上的**投影变换** (也称**投影算子**), 对应的矩阵称为**投影矩阵**.

几点注记

- 对于给定的子空间 $\mathbb{S}_1$ 和 $\mathbb{S}_2$ (构成直和 $\mathbb{S} = \mathbb{S}_1 \oplus \mathbb{S}_2$), 投影变换是唯一的.
- 线性变换在不同的基下对应不同的变换矩阵. 在不加特别指出时, 本讲义中如果线性空间是 $\mathbb{R}^n$ 或 $\mathbb{R}^{n \times n}$, 我们采用自然基, 即 $\{e_1, e_2, \dots, e_n\}$ 和 $\{e_{ij}\}_{i,j=1}^n$.
- 为了书写方便, 我们这里使用 P 既表示投影变换也表示其对应的投影矩阵.

设 P 是从 $\mathbb{S}$ 沿 $\mathbb{S}_2$ 到 $\mathbb{S}_1$ 上的投影变换, 则对任意 $x \in \mathbb{S}_1$ 都有 $Px = x$. 因此, $\mathbb{S}_1 \subseteq \text{Ran}(P)$. 又由定义可知 $\text{Ran}(P) \subseteq \mathbb{S}_1$, 所以

$$\mathbb{S}_1 = \text{Ran}(P).$$

类似地, 我们也可以验证

$$\mathbb{S}_2 = \text{Ker}(P).$$

于是存在直和分解

$$\mathbb{S} = \text{Ran}(P) \oplus \text{Ker}(P).$$

若 $\mathbb{S} = \mathbb{R}^n$, 则立即可以得到下面的结论.

引理 1.25 设 $P \in \mathbb{R}^{n \times n}$ 是一个投影矩阵, 则

$$\mathbb{R}^n = \text{Ran}(P) \oplus \text{Ker}(P). \quad (1.3)$$

 **思考:** 对于一般的矩阵 $A \in \mathbb{R}^{n \times n}$, 结论 $\mathbb{R}^n = \text{Ran}(A) \oplus \text{Ker}(A)$ 是否成立?

下面的性质表明, 投影矩阵由其像空间和零空间所唯一确定.

定理 1.26 设 $\mathbb{R}^n = \mathbb{S}_1 \oplus \mathbb{S}_2$, 则存在唯一的投影矩阵 P , 使得

$$\text{Ran}(P) = \mathbb{S}_1, \quad \text{Ker}(P) = \mathbb{S}_2,$$

即对任意向量 $x \in \mathbb{R}^n$, 有

$$Px \in \mathbb{S}_1, \quad x - Px \in \mathbb{S}_2.$$

例 1.14 若 $\mathbb{S}_1 = \mathbb{R}^n$, 则 $\mathbb{S}_2 = \{0\}$, 所对应的投影矩阵即为单位矩阵 I .

反之, 若 $\mathbb{S}_1 = \{0\}$, 则 $\mathbb{S}_2 = \mathbb{R}^n$, 此时所对应的投影矩阵即为零矩阵.



引理 1.27 设 $P \in \mathbb{R}^{n \times n}$ 是一个投影矩阵, 则

- (1) $I - P$ 也是一个投影矩阵, 且 $\text{Ker}(P) = \text{Ran}(I - P)$;
- (2) P^T 也是一个投影矩阵.

(留作练习)

下面给出投影矩阵的判别定理. 首先, 根据定义, P 是沿 $\mathbb{S}_2$ 到 $\mathbb{S}_1$ 的投影变换的充要条件是: 对任意 $x \in \mathbb{S}_1$ 有 $Px = x$, 而对任意 $x \in \mathbb{S}_2$ 有 $Px = 0$.

定理 1.28 矩阵 $P \in \mathbb{R}^{n \times n}$ 是投影矩阵的充要条件是 $P^2 = P$, 即 P 是幂等矩阵.

(板书)

 **思考:** (1) 在证明充分性时, 为什么只需证明 $\text{Ran}(P) + \text{Ker}(P) = \mathbb{R}^n$?
(2) 该结论在 $\mathbb{C}^{n \times n}$ 中是否成立?

设 $\mathbb{S}_1$ 和 $\mathbb{S}_2$ 是 $\mathbb{R}^n$ 的两个 m 维子空间且 $\mathbb{R}^n = \mathbb{S}_1 \oplus \mathbb{S}_2^\perp$, 则存在唯一的投影变换 P , 使得

$$\text{Ran}(P) = \mathbb{S}_1, \quad \text{Ker}(P) = \mathbb{S}_2^\perp.$$

此时, 我们称 P 是 $\mathbb{S}_1$ 上与 $\mathbb{S}_2$ 正交的投影变换.

令 $v_1, v_2, \dots, v_m$ 和 $w_1, w_2, \dots, w_m$ 分别是 $\mathbb{S}_1$ 和 $\mathbb{S}_2$ 的一组基, 则 P 可以由这两组基来表示.

定理 1.29 设 $P \in \mathbb{R}^{n \times n}$ 是 $\mathbb{S}_1$ 上与 $\mathbb{S}_2$ 正交的投影变换 ($\mathbb{S}_1 \oplus \mathbb{S}_2^\perp = \mathbb{R}^n$), 则

$$P = V(W^T V)^{-1} W^T, \quad (1.4)$$

其中 $V = [v_1, v_2, \dots, v_m]$, $W = [w_1, w_2, \dots, w_m]$.

(留作练习)

 **提示:** 先证明 P 是投影矩阵, 再证明 $\text{Ran}(P) = \mathbb{S}_1$, $\text{Ker}(P) = \mathbb{S}_2^\perp$.

 虽然投影矩阵 P 由 $\mathbb{S}_1$ 和 $\mathbb{S}_2$ 唯一确定, 但其矩阵表示形式 (1.4) 并不唯一 (W 和 V 不唯一).

 **思考:** 对于一般的投影变换, 即 P 是沿 $\mathbb{S}_2$ 到 $\mathbb{S}_1$ 上的投影变换, 如何给出 P 的表达式?

正交投影

设 $\mathbb{S}_1$ 是内积空间 $\mathbb{S}$ 的一个子空间, 则由定理 1.6 可知 $\mathbb{S} = \mathbb{S}_1 \oplus \mathbb{S}_1^\perp$. 因此, 任意 $x \in \mathbb{S}$ 都可唯一分解成

$$x = x_1 + x_2, \quad x_1 \in \mathbb{S}_1, \quad x_2 \in \mathbb{S}_1^\perp.$$

我们称 x_1 为 x 在 $\mathbb{S}_1$ 中的正交投影.



若 P 是从 S 沿 $S_1^\perp$ 到 S_1 上的投影变换, 则称 P 为子空间 S_1 上的**正交投影变换** (也称**正交投影算子**, orthogonal projector), 对应的矩阵称为**正交投影矩阵**, 记为 P_{S_1} . 如果 P 不是正交投影变换, 则称为**斜投影变换** (oblique projector).

由定理 1.29 可立即得到下面的结论.

推论 1.30 设 P 是子空间 S_1 上的正交投影变换, $\{v_1, v_2, \dots, v_m\}$ 是 S_1 的一组标准正交基, 则

$$P = VV^T. \quad (1.5)$$

定理 1.31 投影矩阵 $P \in \mathbb{R}^{n \times n}$ 是正交投影矩阵的充要条件 $P^T = P$.

(留作练习)

 **思考:** 正交投影矩阵 P 的特征值可能取值有哪些?

1.3 向量范数与矩阵范数

1.3.1 向量范数

定义 1.14 (向量范数) 若函数 $f: \mathbb{R}^n \rightarrow \mathbb{R}$ (或 $f: \mathbb{C}^n \rightarrow \mathbb{R}$) 满足

- (1) $f(x) \geq 0, \forall x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$), 等号当且仅当 $x = 0$ 时成立;
- (2) $f(\alpha x) = |\alpha| \cdot f(x), \forall x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$), $\alpha \in \mathbb{R}$ (或 $\mathbb{C}$);
- (3) $f(x+y) \leq f(x) + f(y), \forall x, y \in \mathbb{R}^n$ (或 $\mathbb{C}^n$);

则称 $f(x)$ 为 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的**范数** (norm), 通常记作 $\|x\|$.

 如果 f 只满足 $f(x) \geq 0$, 以及 (2) 和 (3), 则称为**半范数** (seminorm).

例 1.15 $\mathbb{R}^n$ 和 $\mathbb{C}^n$ 上常见的向量范数:

- 1-范数: $\|x\|_1 = |x_1| + |x_2| + \dots + |x_n|$;
- 2-范数: $\|x\|_2 = \sqrt{|x_1|^2 + |x_2|^2 + \dots + |x_n|^2}$;
- ∞ -范数: $\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$;
- p -范数: $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}, \quad p \geq 1$.

 容易验证, 1-范数、2-范数和 ∞ -范数就是 p -范数在 $p = 1, 2, \infty$ 的特殊情形.

 p -范数也称为 Hölder 范数 [122] 或 ℓ_p 范数.



定理 1.32 (范数的连续性) 设 $\|\cdot\|$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的一个向量范数, 则 $f(x) \triangleq \|x\|$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的连续函数. (留作课外自习, 利用范数的三角不等式)

定义 1.15 (范数的等价性) 设 $\|\cdot\|_\alpha$ 与 $\|\cdot\|_\beta$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的两个向量范数, 若存在正常数 c_1, c_2 , 使得

$$c_1 \|x\|_\alpha \leq \|x\|_\beta \leq c_2 \|x\|_\alpha$$

对任意 $x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$) 都成立, 则称 $\|\cdot\|_\alpha$ 与 $\|\cdot\|_\beta$ 是等价的.

定理 1.33 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的所有向量范数都是等价的, 特别地, 有

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{n} \|x\|_2,$$

$$\|x\|_\infty \leq \|x\|_1 \leq n \|x\|_\infty,$$

$$\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n} \|x\|_\infty.$$

(板书, 以 $\mathbb{R}^n$ 为例, 只证明等价性, 三个不等式留作练习)

更一般地, 根据 Jensen 不等式 [63]:

$$\left(\sum_{i=1}^n x_i^p \right)^{1/p} > \left(\sum_{i=1}^n x_i^q \right)^{1/q}, \quad x_i > 0, \quad 0 < p < q,$$

我们有

$$\|x\|_p \geq \|x\|_q, \quad \forall 1 \leq p \leq q.$$

事实上, 有限维赋范线性空间上的所有范数都是等价的 [121].

定理 1.34 (Cauchy-Schwarz 不等式) 设 $(\cdot, \cdot)$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的内积, 则对任意 $x, y \in \mathbb{C}^n$ (或 $\mathbb{R}^n$), 有

$$|(x, y)|^2 \leq (x, x) \cdot (y, y) \quad \text{或} \quad |(x, y)| \leq \|x\| \cdot \|y\|,$$

且等号成立的充要条件是 x 与 y 线性相关, 其中 $\|\cdot\|$ 为内积导出范数, 即 $\|x\| = \sqrt{(x, x)}$.

(板书)

注记: Cauchy-Schwarz 不等式在有的文献中也称为 **Cauchy-Bunyakovski 不等式** 或 **Cauchy-Schwarz-Bunyakovski 不等式** [89]. 中学数学中的 Cauchy 不等式

$$(a_1 b_1 + a_2 b_2 + \cdots + a_n b_n)^2 \leq (a_1^2 + a_2^2 + \cdots + a_n^2) (b_1^2 + b_2^2 + \cdots + b_n^2), \quad (a_i, b_i \in \mathbb{R})$$

是 Cauchy-Schwarz 不等式的特例.

更一般地, 我们有下面的 Holder 不等式.



定理 1.35 (Holder 不等式) 设 $(\cdot, \cdot)$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的标准内积, 则对任意 $x, y \in \mathbb{C}^n$ (或 $\mathbb{R}^n$), 有

$$|(x, y)| \leq \|x\|_p \cdot \|y\|_q,$$

其中 $p, q > 0$, 且 $\frac{1}{p} + \frac{1}{q} = 1$.

(留作课外自习)

内积导出范数

设 $\mathbb{S}$ 是内积空间, 对任意 $x \in \mathbb{S}$, 定义

$$\|x\| \triangleq (x, x)^{\frac{1}{2}}, \quad (1.6)$$

则可以验证, $\|x\|$ 是 $\mathbb{S}$ 上的范数. 这就是**由内积导出的范数**.

 任意一个内积都可以导出一个相应的范数.

例 1.16 $\mathbb{R}^n$ 上由标准内积导出的范数为

$$\|x\| = (x, x)^{\frac{1}{2}} = \left(\sum_{i=1}^n x_i^2 \right)^{\frac{1}{2}}.$$

这就是 2-范数.

例 1.17 (极化恒等式) 设 $\|\cdot\|$ 是 $\mathbb{R}^n$ 上由内积 $(\cdot, \cdot)$ 导出的范数, 则有

$$(x, y) = \frac{1}{4} (\|x + y\|^2 - \|x - y\|^2).$$

1.3.2 矩阵范数

定义 1.16 (矩阵范数) 若函数 $f: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ (或 $f: \mathbb{C}^{m \times n} \rightarrow \mathbb{R}$) 满足

- (1) $f(A) \geq 0, \forall A \in \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 且等号当且仅当 $A = 0$ 时成立;
- (2) $f(\alpha A) = |\alpha| f(A), \forall A \in \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$), $\alpha \in \mathbb{R}$ (或 $\mathbb{C}$);
- (3) $f(A + B) \leq f(A) + f(B), \forall A, B \in \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$);

则称 $f(X)$ 为 $\mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 上的矩阵范数, 通常记作 $\|X\|$.

设 f 是 $\mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$) 上的矩阵范数, 如果 f 还满足

- (4) $f(AB) \leq f(A)f(B), \forall A, B \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$),

则称 f 是**相容**的矩阵范数.

注记

在本讲义中, 如果不加特别指出, 所使用的矩阵范数都是指**相容的矩阵范数**.



设 $\|\cdot\|$ 是 $\mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 上的矩阵范数, 若对任意 $A \in \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 和任意 $x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$), 有

$$\|Ax\|_\alpha \leq \|A\| \|x\|_\beta,$$

则称矩阵范数 $\|\cdot\|$ 与向量范数 $\|\cdot\|_\alpha$ 和 $\|\cdot\|_\beta$ **相容**, 这里的 $\|\cdot\|_\alpha$ 和 $\|\cdot\|_\beta$ 分别为 $\mathbb{R}^m$ 和 $\mathbb{R}^n$ (或 $\mathbb{C}^m$ 和 $\mathbb{C}^n$) 上的向量范数.

一类常用的矩阵范数是由向量范数导出的算子范数.

引理 1.36 (算子范数) 设 $\|\cdot\|$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 上的向量范数, 则

$$\|A\| \triangleq \sup_{x \in \mathbb{R}^n, x \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|$$

是 $\mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$) 上的矩阵范数, 称为**算子范数**, 有时也称为**诱导范数**或**导出范数**. (板书)

 相应地, 可以定义 $\mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 上的算子范数, 此时涉及 $\mathbb{R}^m$ 和 $\mathbb{R}^n$ (或 $\mathbb{C}^m$ 和 $\mathbb{C}^n$) 上的向量范数.

例 1.18 设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), 常见的矩阵范数有:

- p -范数 (算子范数)

$$\|A\|_p = \sup_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p}, \quad p \geq 1.$$

- Frobenius 范数, 简称 F -范数

$$\|A\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2};$$

(留作课外自习, 验证满足矩阵范数的 4 条性质)

引理 1.37 设 $A \in \mathbb{R}^{n \times n}$, 可以证明:

(1) 矩阵 1-范数 (列范数): $\|A\|_1 = \max_{1 \leq j \leq n} \left(\sum_{i=1}^n |a_{ij}| \right);$

(2) 矩阵 ∞ -范数 (行范数): $\|A\|_\infty = \max_{1 \leq i \leq n} \left(\sum_{j=1}^n |a_{ij}| \right);$

(3) 矩阵 2-范数: $\|A\|_2 = \sqrt{\rho(A^T A)}$; (若 A 是复矩阵, 则 $\|A\|_2 = \sqrt{\rho(A^* A)}$)

(4) 矩阵 F -范数: $\|A\|_F = \sqrt{\text{tr}(A^T A)}$. (若 A 是复矩阵, 则 $\|A\|_F = \sqrt{\text{tr}(A^* A)}$)

(板书, 以 ∞ -范数和 2-范数为例, 其他留作练习)

计算 2-范数时需要谱半径, 因此通常比计算 1-范数和 ∞ -范数更困难. 但在某些情况下可以用范数等价性来估计一个矩阵的 2-范数.



定理 1.38 (矩阵范数的等价性) $\mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$) 上的所有范数都是等价的, 特别地, 有

$$\frac{1}{\sqrt{n}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1,$$

$$\frac{1}{\sqrt{n}} \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \|A\|_\infty.$$

(留作练习)

矩阵范数的更多性质

设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), $x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$).

- (1) 对任意矩阵范数 $\|\cdot\|$, 有 $\|A^k\| \leq \|A\|^k$;
- (2) 对任意算子范数 $\|\cdot\|$, 有 $\|Ax\| \leq \|A\| \cdot \|x\|$, $\|AB\| \leq \|A\| \cdot \|B\|$;
- (3) $\|Ax\|_2 \leq \|A\|_F \cdot \|x\|_2$, $\|AB\|_F \leq \|A\|_F \cdot \|B\|_F$;
- (4) F -范数不是算子范数;
- (5) $\|\cdot\|_2$ 和 $\|\cdot\|_F$ 是酉不变范数, 即对任意酉矩阵 (或正交矩阵) U, V , 有

$$\|UA\|_2 = \|AV\|_2 = \|UAV\|_2 = \|A\|_2,$$

$$\|UA\|_F = \|AV\|_F = \|UAV\|_F = \|A\|_F$$

- (6) $\|A^*\|_2 = \|A\|_2$, $\|A^*\|_1 = \|A\|_\infty$;
- (7) 若 A 是正规矩阵, 则 $\|A\|_2 = \rho(A)$.

(留作课外自习)

矩阵直和

设 $A_i \in \mathbb{R}^{n_i \times n_i}$ (或 $\mathbb{C}^{n_i \times n_i}$), $i = 1, 2, \dots, k$, 定义直和

$$\bigoplus_{i=1}^k A_i = A_1 \oplus A_2 \oplus \cdots \oplus A_k \triangleq \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_k \end{bmatrix},$$

即以 A_i 为对角块的块对角矩阵. 可以验证

$$\left\| \bigoplus_{i=1}^k A_i \right\|_p = \max_{1 \leq i \leq k} \|A_i\|_p, \quad p = 1, 2, \infty. \quad (1.7)$$

1.3.3 谱半径与范数

定理 1.39 (谱半径与范数的关系) 设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), 则

- (1) 对任意算子范数, 有 $\rho(A) \leq \|A\|$;
- (2) 反之, 对任意 $\varepsilon > 0$, 都存在一个算子范数 $\|\cdot\|_\varepsilon$, 使得 $\|A\|_\varepsilon \leq \rho(A) + \varepsilon$, 其中范数 $\|\cdot\|_\varepsilon$ 依赖于 A 和 ε . 所以, 若 $\rho(A) < 1$, 则存在算子范数 $\|\cdot\|_\varepsilon$, 使得 $\|A\|_\varepsilon < 1$. (板书)



 事实上, 定理 1.39 中的结论 (1) 对任意矩阵范数都成立, 见习题 1.10.

除此之外, 我们还有下面的性质.

推论 1.40 设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), 则 $\|A\|_2^2 \leq \|A\|_1 \|A\|_\infty$, 且

$$\max_{1 \leq i, j \leq n} \{|a_{ij}|\} \leq \|A\|_2 \leq n \max_{1 \leq i, j \leq n} \{|a_{ij}|\}.$$

(留作练习)

1.3.4 最佳逼近与正交投影

下面是关于正交投影变换的一个常用性质, 可以直接通过 2-范数的定义证明.

定理 1.41 设 $P \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$) 是正交投影矩阵, 则 $\|P\|_2 = 1$, 且对 $\forall x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$), 有

$$\|x\|_2^2 = \|Px\|_2^2 + \|(I - P)x\|_2^2.$$

正交投影可用于描述最佳逼近问题的解.

定理 1.42 设 $\mathbb{S}$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 的子空间, $z \in \mathbb{R}^n$ (或 $\mathbb{C}^n$) 是一个给定的向量, 则最佳逼近问题

$$\min_{x \in \mathbb{S}} \|x - z\|_2$$

的唯一解为

$$x_* = P_{\mathbb{S}} z.$$

即 $\mathbb{S}$ 中距离 z 最近 (在 2-范数意义下) 的向量是 z 在 $\mathbb{S}$ 中的正交投影.

(留作练习)

上述定理中的 2-范数可以推广到一般的能量范数.

推论 1.43 设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$) 对称正定 (或 Hermite 正定), $\mathbb{S}$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 的子空间, 给定 $z \in \mathbb{R}^n$ (或 $\mathbb{C}^n$), 则 x_* 是最佳逼近问题

$$\min_{x \in \mathbb{S}} \|x - z\|_A$$

的解的充要条件是

$$x_* \in \mathbb{S} \quad \text{且} \quad A(x_* - z) \perp \mathbb{S}.$$

此处**能量范数** $\|\cdot\|_A$ 的定义为: $\|x - z\|_A \triangleq \sqrt{(x - z)^* A (x - z)}$.

(留作练习)

1.4 矩阵标准型

1.4.1 Jordan 标准型

在计算矩阵的特征值时, 一个基本的思想是通过相似变换, 将其转化成一个形式尽可能简单的矩阵, 使得其特征值更易于计算. **Jordan 标准型**则是矩阵在相似变化下的最简形式.



定理 1.44 设 $A \in \mathbb{C}^{n \times n}$ (或 $\mathbb{R}^{n \times n}$) 有 p 个互不相同的特征值, 则存在非奇异矩阵 $X \in \mathbb{C}^{n \times n}$, 使得

$$X^{-1}AX = \begin{bmatrix} J_1 & & \\ & J_2 & \\ & & \ddots \\ & & & J_p \end{bmatrix} \triangleq J, \quad (1.8)$$

其中 J_i 的维数等于 λ_i 的代数重数, 且具有下面的结构

$$J_i = \begin{bmatrix} J_i^{(1)} & & \\ & J_i^{(2)} & \\ & & \ddots \\ & & & J_i^{(\nu_i)} \end{bmatrix}, \quad J_i^{(k)} = \begin{bmatrix} \lambda_i & 1 & & \\ & \ddots & \ddots & \\ & & \lambda_i & 1 \\ & & & \lambda_i \end{bmatrix}.$$

这里的 ν_i 为 λ_i 的几何重数, $J_i^{(k)}$ 称为 (对应于 λ_i 的) **Jordan 块**, J 就称为 A 的 **Jordan 标准型**.

该定理可以通过 λ -矩阵来证明 (见高等代数教材), 也可以通过后面的 Schur 分解来证明.

除了 Jordan 块的排列次序外, Jordan 标准型是唯一确定的.

可以证明, 对于每一个 Jordan 块 $J_i^{(k)}$, 都存在一个列满秩矩阵 $X_i^{(k)}$ 使得

$$AX_i^{(k)} = X_i^{(k)} J_i^{(k)}.$$

Jordan 标准型的基本性质

- Jordan 块的个数等于 A 的线性无关的特征向量的个数;
- A 可对角化的充要条件是每个 Jordan 块都是 1×1 的, 此时 X 的列向量就是 A 的特征向量.

根据 Jordan 标准型和特征值的连续性, 我们可以得到下面的结论.

推论 1.45 所有可对角化矩阵组成的集合在所有矩阵组成的集合中是稠密的, 即任何一个矩阵都可以通过可对角化矩阵来逼近.

Jordan 标准型的一个重要应用是可以用来计算矩阵的最小多项式.

定理 1.46 设 $\lambda_1, \lambda_2, \dots, \lambda_p$ 为 $A \in \mathbb{C}^{n \times n}$ 的互不相等的特征值, 则 A 的最小多项式为

$$p(\lambda) = \prod_{i=1}^q (\lambda - \lambda_i)^{r_i},$$

其中 r_i 是与 λ_i 所对应的最大 Jordan 块的维数.



1.4.2 Schur 分解

Jordan 标准型在理论研究中非常有用, 但数值计算比较困难. 下面我们介绍一个比较实用的矩阵分解, 即 **Schur 分解**.

定理 1.47 设 $A \in \mathbb{C}^{n \times n}$ (或 $\mathbb{R}^{n \times n}$), 则存在酉矩阵 $U \in \mathbb{C}^{n \times n}$ 使得

$$U^*AU = \begin{bmatrix} \lambda_1 & r_{12} & \cdots & r_{1n} \\ 0 & \lambda_2 & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \triangleq R \quad \text{或} \quad A = URU^*, \quad (1.9)$$

其中 $\lambda_1, \lambda_2, \dots, \lambda_n$ 是 A 的特征值 (可以按任意顺序排列).

(板书)

关于 Schur 分解的几点说明

- 该结论告诉我们, 任意一个矩阵都可以**酉三角化**.
- 三角矩阵可以说是一般矩阵在酉相似变化下的最简形式.
- 定理中的 U 和 R 不是唯一的.
- R 的对角线元素可以按任意顺序排列, 特别地, 可以按模从大到小排列.

推论 1.48 设 $A \in \mathbb{C}^{n \times n}$, 则

- (1) A 是正规矩阵当且仅当 R 是对角矩阵, 即 A 可酉对角化当且仅当 A 是正规矩阵;
- (2) A 是 Hermite 矩阵当且仅当 R 是实对角矩阵.

众所周知, 当 A 是实矩阵时, 其特征值和特征向量仍可能是复的. 在计算实矩阵的特征值时, 通常希望尽可能地避免复数运算. 这时, 我们就需要用到下面的**实 Schur 分解** (或**拟 Schur 分解**).

定理 1.49 设 $A \in \mathbb{R}^{n \times n}$, 则存在正交矩阵 $Q \in \mathbb{R}^{n \times n}$, 使得

$$Q^T A Q = T, \quad (1.10)$$

其中 $T \in \mathbb{R}^{n \times n}$ 是**拟上三角矩阵**, 即 T 是块上三角的, 且对角块为 1×1 或 2×2 的块矩阵. 若对角块是 1×1 的, 则其就是 A 的一个特征值, 若对角块是 2×2 的, 则其特征值是 A 的一对共轭复特征值.

(板书)

易知, 若拟上三角矩阵 T 是上三角矩阵, 则 A 的特征值都是实的. 反之, 结论也成立.

推论 1.50 设 $A \in \mathbb{R}^{n \times n}$ 的特征值都是实的, 则存在正交矩阵 $Q \in \mathbb{R}^{n \times n}$ 和上三角矩阵 $R \in \mathbb{R}^{n \times n}$ 使得

$$Q^T A Q = R,$$

其中 R 的对角线元素即为 A 的特征值.



1.5 几类特殊矩阵

1.5.1 对称正定矩阵

定义 1.17 设 $A \in \mathbb{C}^{n \times n}$.

- 若对所有向量 $x \in \mathbb{C}^n$ 有 $\operatorname{Re}(x^*Ax) \geq 0$, 则称 A 是**半正定**的;
- 若对所有非零向量 $x \in \mathbb{C}^n$ 有 $\operatorname{Re}(x^*Ax) > 0$, 则称 A 是**正定**的;
- 若 A 是 Hermite 的且半正定, 则称 A 为 **Hermite 半正定**;
- 若 A 是 Hermite 的且正定, 则称 A 为 **Hermite 正定**;
- 若 $A \in \mathbb{R}^{n \times n}$ 是对称的且半正定, 则称 A 为**对称半正定**;
- 若 $A \in \mathbb{R}^{n \times n}$ 是对称的且正定, 则称 A 为**对称正定**.

 若对所有向量 $x \in \mathbb{C}^n$ 有 $x^*Ax \in \mathbb{R}$, 则 $A^* = A$.

 本讲义中, 正定和半正定矩阵不要求是对称或 Hermite.

 若 $-A$ 是半正定的, 则称 A 是**半负定**的; 若 $-A$ 是正定的, 则称 A 是**负定**的.

定理 1.51 设 $A \in \mathbb{C}^{n \times n}$, 则 A 正定 (或半正定) 的充要条件是矩阵 $H = \frac{1}{2}(A + A^*)$ 正定 (或半正定). (留作练习)

如果 A 是实矩阵, 则只需在实数域中考虑即可.

定理 1.52 设 $A \in \mathbb{R}^{n \times n}$, 则 A 正定 (或半正定) 的充要条件是对任意非零向量 $x \in \mathbb{R}^n$ 有 $x^T Ax > 0$ (或 $x^T Ax \geq 0$). (留作练习)

对称正定矩阵的基本性质

设 $A \in \mathbb{C}^{n \times n}$.

- (1) A Hermite 正定当且仅当 A Hermite 且所有特征值都是正的;
- (2) A Hermite 正定当且仅当存在酉矩阵 U 使得 $A = U\Lambda U^*$, 其中 Λ 为对角矩阵且对角线均为正实数;
- (3) A Hermite 正定当且仅当 S^*AS Hermite 正定, 其中 $S \in \mathbb{C}^{n \times n}$ 是一个任意的非奇异矩阵;
- (4) 若 A Hermite 正定, 则 A 的任意主子矩阵都 Hermite 正定;
- (5) 若 A Hermite 正定, 则 A 的所有对角线元素都是正的, 且 $\max_{i \neq j} \{|a_{ij}|\} < \max_i \{a_{ii}\}$, 即绝对值最大的元素出现在对角线上.

 以上结论在实数域中也成立.



平方根

如果 A 是 Hermite (半) 正定矩阵, 则可以定义其平方根, 即存在唯一的 Hermite (半) 正定矩阵 B , 使得 $B^2 = A$. 事实上, 我们有下面更一般的结论 [63].

定理 1.53 设 $A \in \mathbb{C}^{n \times n}$ 是一个 Hermite 半正定矩阵, k 是一个给定的正整数, 则存在唯一的 Hermite 半正定矩阵 $B \in \mathbb{C}^{n \times n}$ 使得

$$B^k = A.$$

同时, 我们还有下面的性质:

- (1) $\text{rank}(B) = \text{rank}(A)$, 因此, 若 A 是正定的, 则 B 也正定;
- (2) 如果 A 是实矩阵的, 则 B 也是实矩阵.

特别地, 当 $k = 2$ 时, 称 B 为 A 的平方根, 记为 $A^{\frac{1}{2}}$.

(留作课外自习)

Hermite 正定矩阵与内积

Hermite 正定矩阵与内积之间有如下的一一对应关系.

定理 1.54 设 $(\cdot, \cdot)$ 是 $\mathbb{C}^n$ 上的一个内积, 则存在一个 Hermite 正定矩阵 $A \in \mathbb{C}^{n \times n}$ 使得

$$(x, y) = y^* A x.$$

反之, 若 $A \in \mathbb{C}^{n \times n}$ 是 Hermite 正定矩阵, 则

$$f(x, y) \triangleq y^* A x$$

是 $\mathbb{C}^n$ 上的一个内积.

(留作练习)

 该结论在实数域中也成立.

 有些教材用上面的定理来定义 Hermite 正定矩阵, 即: 设 A 是 Hermite 矩阵, 若

$$(x, y) = y^* A x$$

是一个内积, 则 A 是 Hermite 正定的.

1.5.2 对角占优矩阵

定义 1.18 设 $A \in \mathbb{C}^{n \times n}$, 若

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$$

对所有 $i = 1, 2, \dots, n$ 都成立, 且至少有一个不等式严格成立, 则称 A 为弱行对角占优, 简称弱对角占优或对角占优. 若所有不等式都严格成立, 则为严格行对角占优或严格对角占优.



 类似地, 可以定义**列对角占优**和**严格列对角占优**.

定理 1.55 若 $A \in \mathbb{C}^{n \times n}$ 是严格对角占优矩阵, 则 A 非奇异.

(板书)

如果 A 是严格对角占优的, 则可以给出 $\|A^{-1}\|_1$ 的一个估计.

定理 1.56 [49] 设 $A \in \mathbb{R}^{n \times n}$ 严格列对角占优, 记

$$\delta \triangleq \min_{1 \leq j \leq n} \left(|a_{jj}| - \sum_{i=1}^n |a_{ij}| \right) > 0,$$

则有

$$\|A^{-1}\|_1 \leq \delta^{-1}.$$

 定理中的 δ 是衡量对角占优程度的一个重要指标: 若 δ 接近于 0, 则表示对角占优性很弱.

1.5.3 不可约矩阵

定义 1.19 设 $A \in \mathbb{C}^{n \times n}$, 如果存在置换矩阵 P 使得 PAP^T 为块上三角矩阵, 即

$$PAP^T = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{C}^{k \times k}, \quad A_{22} \in \mathbb{C}^{(n-k) \times (n-k)} \quad (1.11)$$

其中 $1 \leq k < n$, 则称 A 是**可约**的, 否则就称 A 为**不可约**的.

 如果 A 是 1×1 矩阵, 则 A 不可约当且仅当 A 非零.

 如果 $A \in \mathbb{C}^{n \times n}$ 是可约的, 则 A 至少有 $n-1$ 个零.

不可约的意义

若 A 可约, 即存在置换矩阵 P , 使得

$$PAP^T = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix},$$

则线性方程组 $Ax = b$ 等价于 $PAP^T Px = Pb$. 记 $y \triangleq Px$, $f \triangleq Pb$, 则

$$\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}.$$

因此, 原方程组就转化为下面两个更小规模的子方程组

$$\begin{cases} A_{22}y_2 = f_2, \\ A_{11}y_1 = f_1 - A_{12}y_2. \end{cases}$$

显然, 求解这两个方程组比求解原方程组所需的运算量更少.



对于特征值问题, A 的特征值就是 A_{11} 和 A_{22} 的特征值的并, 因此也只需考虑子矩阵 A_{11} 和 A_{22} 的特征值即可.

如果 A_{22} 或 A_{11} 仍然是可约的, 则可以转化为更小规模的子问题, 直到不可约为止. 因此我们在讨论某些算法的性质时, 一般只需考虑不可约情形.

 由于 PAP^T 保持 A 的对角线元素仍然在对角线上, 因此由定理 1.59 可知, 主对角线元素是否为零并不影响矩阵的可约性.

推论 1.57 设 $A = [a_{ij}] \in \mathbb{C}^{n \times n}$ 不可约, 则 $A + D$ 也不可约, 其中 D 是任意对角矩阵. 特别地, $B \triangleq A - \text{diag}(a_{11}, a_{22}, \dots, a_{nn})$ 不可约.

如果 A 可约, 即存在置换矩阵 P 使得 PAP^T 具有 (1.11) 的形式, 则对任意正整数 k , 有

$$PA^kP^T = (PAP^T)^k = \begin{bmatrix} A_{11}^k & \tilde{A}_{12}^{(k)} \\ 0 & A_{22}^k \end{bmatrix}. \quad (1.12)$$

于是我们有下面的结论.

定理 1.58 设 $A \in \mathbb{C}^{n \times n}$. 若 A 可约, 则 A^k 也可约. 反之, 若存在一个正整数 k , 使得 A^k 是不可约的, 则 A 也不可约.

 需要指出的是, 不可约矩阵的幂不一定是不可约的. 如 $A = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}$ 不可约, 但 $A^2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ 可约.

下面的定理给出了一个矩阵可约的充要条件.

定理 1.59 设 $A = [a_{ij}] \in \mathbb{C}^{n \times n}$, 指标集 $\mathbb{Z}_n = \{1, 2, \dots, n\}$, 则 A 可约的充要条件是存在非空指标集 $S, T \subset \mathbb{Z}_n$ 满足 $S \oplus T = \mathbb{Z}_n$, 使得

$$a_{ij} = 0, \quad i \in S, \quad j \in T.$$

(板书)

下面是判断矩阵是否可约的另一个充要条件.

定理 1.60 设 $A = [a_{ij}] \in \mathbb{C}^{n \times n}$, 指标集 $\mathbb{Z}_n = \{1, 2, \dots, n\}$, 则 A 可约的充要条件是存在两个

相异的正整数 $k, l \in \mathbb{Z}_n$, 使得对任意指标序列 $\{i_1, i_2, \dots, i_r\} \subseteq \mathbb{Z}_n$, 都有

$$a_{ki_1} a_{i_1 i_2} \cdots a_{i_r l} = 0.$$

这里 $r > 0$ 是任意正整数.

(留作课外自习)

定理 1.60 的等价描述

矩阵 $A = [a_{ij}] \in \mathbb{C}^{n \times n}$ 不可约的充要条件是: 对任意两个相异正整数 $k, l \in \mathbb{Z}_n$, 都存在一个指标序列 $\{i_1, i_2, \dots, i_m\} \subseteq \mathbb{Z}_n$ 使得

$$a_{ki_1} a_{i_1 i_2} \cdots a_{i_m l} \neq 0.$$

如果有向图表示矩阵, 则不可约当且仅当有向图是强连通的, 即任意两个节点之间都存在一条路径.

我们知道, 严格对角占优矩阵是非奇异的. 如果 A 是不可约对角占优矩阵, 则可以同样证明 A 是非奇异的.

定理 1.61 设 $A \in \mathbb{C}^{n \times n}$ 是不可约的弱对角占优矩阵, 则 A 非奇异.

(留作练习)

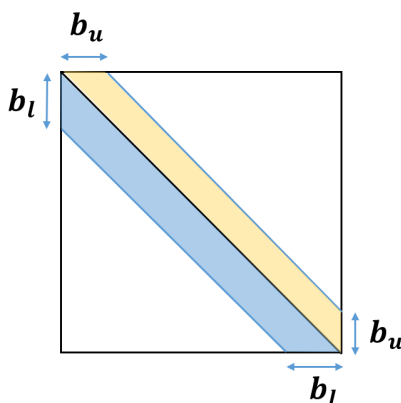
关于严格对角占优矩阵和不可约弱对角占优矩阵的非奇异性, 也可以使用 Gerschgorin 圆盘定理证明, 参见 [126].

一个有意思的现象

通常, 如果某个元素全不为零的矩阵具有某种性质, 则这个性质往往能推广到不可约矩阵.

1.5.4 其它常见特殊矩阵

- 带状矩阵 (banded matrix): $a_{ij} \neq 0$ 当且仅当 $-b_u \leq i - j \leq b_l$, 其中 b_u 和 b_l 为非负整数, 分别称为上带宽和下带宽, $b_u + b_l + 1$ 称为 A 的带宽 (bandwidth);



- 上 Hessenberg 矩阵 (upper Hessenberg matrix): $a_{ij} = 0$ for $i - j > 1$;

$$\begin{bmatrix} * & * & * & \cdots & * \\ * & * & * & \cdots & * \\ & * & * & \cdots & * \\ & & \ddots & \ddots & \vdots \\ & & & * & * \end{bmatrix}$$

- 下 Hessenberg 矩阵 (lower Hessenberg matrix): $a_{ij} = 0$ for $i - j < 1$;

$$\begin{bmatrix} * & * & & & \\ * & * & * & & \\ \vdots & \ddots & \ddots & \ddots & \\ * & \cdots & * & * & * \\ * & \cdots & * & * & * \end{bmatrix}$$

- Toeplitz 矩阵:

$$T = \begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-n+1} \\ t_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & \cdots & t_1 & t_0 \end{bmatrix}$$

- 循环矩阵 (circulant matrix):

$$C = \begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ c_2 & c_1 & c_0 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & \cdots & c_0 \end{bmatrix}$$

- Hankel 矩阵:

$$H = \begin{bmatrix} h_0 & h_1 & \cdots & h_{n-2} & h_{n-1} \\ h_1 & \ddots & \ddots & \ddots & h_n \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ h_{n-2} & \ddots & \ddots & \ddots & h_{2n-3} \\ h_{n-1} & h_n & \cdots & h_{2n-3} & h_{2n-2} \end{bmatrix}$$

- 几个能够直接写出逆的矩阵:

$$\begin{bmatrix} a_1 b_1 & & & & \\ a_2 b_1 & a_2 b_2 & & & \\ a_3 b_1 & a_3 b_2 & a_3 b_3 & & \\ \vdots & \vdots & \vdots & \ddots & \\ a_n b_1 & a_n b_2 & a_n b_3 & \cdots & a_n b_n \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{b_1 a_1} & & & & \\ -\frac{1}{b_2 a_1} & \frac{1}{b_2 a_2} & & & \\ & -\frac{1}{b_3 a_2} & \frac{1}{b_3 a_3} & & \\ & & \ddots & \ddots & \\ & & & -\frac{1}{b_n a_{n-1}} & \frac{1}{b_n a_n} \end{bmatrix}$$



特例:

$$\begin{aligned} \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ \vdots & \vdots & \vdots & \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} &= \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \\ \begin{bmatrix} b_1 & b_1 & b_1 & \cdots & b_1 \\ b_1 & b_1+b_2 & b_1+b_2 & \cdots & b_1+b_2 \\ b_1 & b_1+b_2 & b_1+b_2+b_3 & \cdots & b_1+b_2+b_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_1 & b_1+b_2 & b_1+b_2+b_3 & \cdots & b_1+b_2+\cdots+b_n \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \frac{1}{b_1} + \frac{1}{b_2} & -\frac{1}{b_2} & & & \\ -\frac{1}{b_2} & \frac{1}{b_2} + \frac{1}{b_3} & -\frac{1}{b_3} & & \\ & \ddots & \ddots & \ddots & \\ & & -\frac{1}{b_{n-1}} & \frac{1}{b_{n-1}} + \frac{1}{b_n} & -\frac{1}{b_n} \\ & & & -\frac{1}{b_n} & \frac{1}{b_n} \end{bmatrix} \end{aligned}$$

特例:

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 2 & \cdots & 2 \\ 1 & 2 & 3 & \cdots & 3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 3 & \cdots & n \end{bmatrix}^{-1} = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}$$

1.6 Kronecker 积

定义 1.20 设 $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{p \times q}$, 则 A 与 B 的 **Kronecker 积** 定义为

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{mp \times nq}.$$

 Kronecker 积也称为**直积** (direct product), 或**张量积** (tensor product).

任意两个矩阵都存在 Kronecker 积, 且 $A \otimes B$ 和 $B \otimes A$ 是同阶矩阵. 通常 $A \otimes B \neq B \otimes A$, 但它们之间存在下面的关系式.



定理 1.62 设 $A \in \mathbb{C}^{m \times m}$, $B \in \mathbb{C}^{n \times n}$, 则存在 mn 阶置换矩阵 P 使得

$$P^T(A \otimes B)P = B \otimes A.$$

定理 1.63 矩阵的 Kronecker 积有以下性质:

- (1) $(\alpha A) \otimes B = A \otimes (\alpha B) = \alpha(A \otimes B), \quad \forall \alpha \in \mathbb{R};$
- (2) $(A \otimes B)^T = A^T \otimes B^T, \quad (A \otimes B)^* = A^* \otimes B^*;$
- (3) $(A \otimes B) \otimes C = A \otimes (B \otimes C);$
- (4) $(A + B) \otimes C = A \otimes C + B \otimes C;$
- (5) $A \otimes (B + C) = A \otimes B + A \otimes C;$
- (6) 混合积: $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$
- (7) $(A_1 \otimes A_2 \otimes \cdots \otimes A_k)(B_1 \otimes B_2 \otimes \cdots \otimes B_k)$
 $= (A_1 B_1) \otimes (A_2 B_2) \otimes \cdots \otimes (A_k B_k);$
- (8) $(A_1 \otimes B_1)(A_2 \otimes B_2) \cdots (A_k \otimes B_k) = (A_1 A_2 \cdots A_k) \otimes (B_1 B_2 \cdots B_k);$
- (9) $\text{rank}(A \otimes B) = \text{rank}(A) \text{rank}(B);$

推论 1.64 设 $A = Q_1 \Lambda_1 Q_1^{-1} \in \mathbb{C}^{m \times m}$, $B = Q_2 \Lambda_2 Q_2^{-1} \in \mathbb{C}^{n \times n}$, 则

$$A \otimes B = (Q_1 \otimes Q_2)(\Lambda_1 \otimes \Lambda_2)(Q_1 \otimes Q_2)^{-1}.$$

Kronecker 积的特征值

定理 1.65 设 $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{n \times n}$, 并设 (λ, x) 和 (μ, y) 分别是 A 和 B 的一个特征对, 则 $(\lambda\mu, x \otimes y)$ 是 $A \otimes B$ 的一个特征对. 由此可知, $B \otimes A$ 与 $A \otimes B$ 具有相同的特征值.

推论 1.66 设 $A \in \mathbb{C}^{m \times m}$, $B \in \mathbb{C}^{n \times n}$, 则 $A \otimes I_n + I_m \otimes B$ 的特征值为 $\lambda_i + \mu_j$, 其中 λ_i 和 μ_j 分别为 A 和 B 的特征值.

定理 1.67 设 $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{n \times n}$, 则

- (1) $\text{tr}(A \otimes B) = \text{tr}(A)\text{tr}(B);$
- (2) $\det(A \otimes B) = \det(A)^n \det(B)^m;$
- (3) 若 A 和 B 都非奇异, 则 $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}.$

Kronecker 积与向量的乘积

定理 1.68 设矩阵 $X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{m \times n}$, 记 $\text{vec}(X)$ 为 X 按列拉成的 mn 维列向量,



即

$$\text{vec}(X) = [x_1^\top, x_2^\top, \dots, x_n^\top]^\top,$$

则有

$$\text{vec}(AX) = (I \otimes A)\text{vec}(X), \quad \text{vec}(XB) = (B^\top \otimes I)\text{vec}(X),$$

以及

$$(A \otimes B)\text{vec}(X) = \text{vec}(BXA^\top).$$

 我们称 $\text{vec}(X)$ 为**向量化算子**或**拉直算子**. 该结论对节省运算量和存储量都有好处.

Kronecker 积与矩阵方程

Kronecker 积一个重要应用是可以将某些矩阵方程转化成一般的代数方程.

定理 1.69 矩阵方程

$$AX + XB = D$$

等价于代数方程

$$(I \otimes A + B^\top \otimes I)\text{vec}(X) = \text{vec}(D).$$

1.7 课后习题

练习 1.1 证明: $(A, B) \triangleq \text{tr}(B^\top A)$ 是 $\mathbb{R}^{n \times n}$ 上的内积.

练习 1.2 证明: $\|A\|_1 = \max_{1 \leq j \leq n} \left(\sum_{i=1}^n |a_{ij}| \right)$.

练习 1.3 (定理 1.33) 证明不等式:

- (1) $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n} \|x\|_2$,
- (2) $\|x\|_\infty \leq \|x\|_1 \leq n \|x\|_\infty$,
- (3) $\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n} \|x\|_\infty$.

练习 1.4 证明: (1) $\frac{1}{\sqrt{n}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1$; (2) $\frac{1}{\sqrt{n}} \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \|A\|_\infty$;
 (3) $\frac{1}{\sqrt{n}} \|A\|_F \leq \|A\|_2 \leq \|A\|_F$; (4) $\frac{1}{\sqrt{n}} \|A\|_F \leq \|A\|_1 \leq \sqrt{n} \|A\|_F$.

练习 1.5 (推论 1.40) 设 $A \in \mathbb{R}^{n \times n}$. 证明: $\|A\|_2^2 \leq \|A\|_1 \|A\|_\infty$, 且

$$\max_{1 \leq i, j \leq n} \{|a_{ij}|\} \leq \|A\|_2 \leq n \max_{1 \leq i, j \leq n} \{|a_{ij}|\}.$$

练习 1.6 设 $\|\cdot\|$ 是 $\mathbb{R}^m$ 空间上的一个向量范数, $A \in \mathbb{R}^{m \times n}$, 且 $\text{rank}(A) = n$. 证明: $\|x\|_A \triangleq \|Ax\|$ 是一个向量范数. 特别地, 如果 $A \in \mathbb{R}^m$ 非奇异, 则 $\|x\|_A \triangleq \|Ax\|$ 是一个向量范数.

练习 1.7 设 $\|\cdot\|$ 是 $\mathbb{R}^n$ 空间上的一个向量范数. 证明: $\|A^{-1}\|^{-1} = \min_{\|x\|=1} \|Ax\|$.



练习 1.8* 设 $A \in \mathbb{R}^{n \times n}$, 证明 $\|A\|_F^2 \geq \sum_{i=1}^n |\lambda_i(A)|^2$.

练习 1.9 设 $\|\cdot\|_\alpha$ 是定义在 $\mathbb{C}^{n \times n}$ 上的一个矩阵范数. 证明: 存在 $\mathbb{C}^n$ 上的向量范数 $\|\cdot\|_\beta$, 该范数与 $\|\cdot\|_\alpha$ 相容, 即

$$\|Ax\|_\beta \leq \|A\|_\alpha \|x\|_\beta, \quad \forall A \in \mathbb{C}^{n \times n}, x \in \mathbb{C}^n.$$

练习 1.10 设 $A \in \mathbb{C}^{n \times n}$. 试证明对任意矩阵范数都有 $\rho(A) \leq \|A\|$.

练习 1.11 设 $(\cdot, \cdot)$ 是 $\mathbb{C}^n$ 上的内积. 证明: $\|x\| \triangleq \sqrt{(x, x)}$ 是 $\mathbb{C}^n$ 上的一个向量范数.

练习 1.12 设 $J = \begin{bmatrix} \lambda & \varepsilon & & \\ & \ddots & \ddots & \\ & & \ddots & \varepsilon \\ & & & \lambda \end{bmatrix}$, 其中 $\lambda \geq 0, \varepsilon \geq 0$. 证明 $\|J\|_2 \leq \lambda + \varepsilon$.

练习 1.13 设 $A \in \mathbb{R}^{n \times n}$, $x \in \mathbb{R}^n$ 且 $x \neq 0$. 证明

$$\left\| A \left(I - \frac{xx^T}{x^T x} \right) \right\|_F^2 = \|A\|_F^2 - \frac{\|Ax\|_2^2}{x^T x}.$$

练习 1.14 (定理 1.29) 设 $P \in \mathbb{R}^{n \times n}$ 是 $\mathbb{S}_1$ 上与 $\mathbb{S}_2$ 正交的投影矩阵. 证明:

$$P = V(W^T V)^{-1} W^T,$$

其中 $V = [v_1, v_2, \dots, v_m]$ 和 $W = [w_1, w_2, \dots, w_m]$ 的列向量组分别构成 $\mathbb{S}_1$ 和 $\mathbb{S}_2$ 的一组基.

练习 1.15 (定理 1.31) 证明: 投影矩阵 $P \in \mathbb{R}^{n \times n}$ 是正交投影矩阵的充要条件 $P^T = P$

练习 1.16 (定理 1.42) 设 $\mathbb{S}$ 是 $\mathbb{R}^n$ 的一个子空间, $z \in \mathbb{R}^n$ 是一个给定的向量, 则最佳逼近问题

$$\min_{x \in \mathbb{S}} \|x - z\|_2$$

的唯一解为

$$x_* = P_{\mathbb{S}} z.$$

即 $\mathbb{S}$ 中距离 z 最近 (在 2-范数意义下) 的向量是 z 在 $\mathbb{S}$ 中的正交投影.

练习 1.17 (推论 1.43) 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $\mathbb{S}$ 是 $\mathbb{R}^n$ 的一个子空间, 给定 $z \in \mathbb{R}^n$, 则 x_* 是最佳逼近问题

$$\min_{x \in \mathbb{S}} \|x - z\|_A$$

的解的充要条件是

$$x_* \in \mathbb{S} \quad \text{且} \quad A(x_* - z) \perp \mathbb{S}.$$

此处 $\|\cdot\|_A$ 的定义为: $\|x - z\|_A \triangleq \sqrt{(x - z)^T A (x - z)}$.

练习 1.18 (定理 1.51) 证明: $A \in \mathbb{C}^{n \times n}$ 正定 (半正定) 的充要条件是矩阵 $H = \frac{1}{2}(A + A^*)$ 正定 (半正定).

练习 1.19 (定理 1.52) 证明: $A \in \mathbb{R}^{n \times n}$ 正定 (半正定) 的充要条件是对任意非零向量 $x \in \mathbb{R}^n$ 有



$$x^T A x > 0 \ (x^T A x \geq 0).$$

练习 1.20 设 $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times m}$, 证明: $\text{tr}(AB) = \text{tr}(BA)$.

练习 1.21 (定理 1.61) 设 $A \in \mathbb{C}^{n \times n}$ 是不可约对角占优矩阵, 证明: A 非奇异.

练习 1.22 (Sherman-Morrison 公式) 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, $x, y \in \mathbb{R}^n$.

证明: 若 $y^T A^{-1} x \neq 1$, 则 $A - xy^T$ 可逆, 且

$$(A - xy^T)^{-1} = A^{-1} - \frac{A^{-1}xy^T A^{-1}}{y^T A^{-1}x - 1}.$$

练习 1.23 证明下面的结论:

- (1) 设 $A, B \in \mathbb{R}^{n \times n}$ 可交换, 即 $AB = BA$. 若 A 是对角矩阵且对角线元素互不相等, 则 B 也是对角矩阵.
- (2) 设 $A, B \in \mathbb{R}^{n \times n}$ 可交换, 即 $AB = BA$. 若 A 是块对角矩阵且对角块为标量矩阵, 即 $A = \lambda_1 I_{n_1} \oplus \lambda_2 I_{n_2} \oplus \cdots \oplus \lambda_p I_{n_p}$, 其中 $n_1 + n_2 + \cdots + n_p = n$, 且 λ_i 互不相等, 则 B 是具有相应分块结构的块对角矩阵, 即 $B = B_1 \oplus B_2 \oplus \cdots \oplus B_p$, 其中 $B_i \in \mathbb{R}^{n_i \times n_i}$, $i = 1, 2, \dots, p$.
- (3) 设 $A, B \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 则 $AB = BA$ 的充要条件是存在正交矩阵 Q 使得

$$Q A Q^T = \Lambda_A, \quad Q B Q^T = \Lambda_B,$$

其中 Λ_A 和 Λ_B 分别表示由 A 和 B 的特征值构成的对角矩阵.

练习 1.24 判断下列矩阵是否可约:

$$(1) \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \quad (2) \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad (3) \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 2 \\ -3 & 0 & 0 \end{bmatrix}.$$

练习 1.25* (定理 1.54) 设 $(\cdot, \cdot)$ 是 $\mathbb{C}^n$ 上的一个内积. 证明: 存在一个 Hermite 正定矩阵 $A \in \mathbb{C}^{n \times n}$ 使得 $(x, y) = y^* A x$ 对任意 $x, y \in \mathbb{C}^n$ 都成立. 反之, 若 $A \in \mathbb{C}^{n \times n}$ 是 Hermite 正定矩阵, 则 $f(x, y) \triangleq y^* A x$ 是 $\mathbb{C}^n$ 上的一个内积.

练习 1.26* 设 $B \in \mathbb{R}^{m \times n}$ ($m \leq n$) 是满秩矩阵, $C \in \mathbb{R}^{m \times m}$ 是对称半正定矩阵. 证明:

$$B^T (B B^T)^{-1} B - B^T (C + B B^T)^{-1} B$$

是对称半正定的.

练习 1.27* 设 P 是置换矩阵, 则 P 可以表示为一系列初等置换矩阵 (即交换单位矩阵的两行所得到的矩阵) 的乘积. 试问在什么条件下, 有 $P^T = P$.

..... **以下为可选题**

练习 1.28 设 $A \in \mathbb{R}^{m \times n}$ ($m < n$) 是满秩矩阵, $Z \in \mathbb{R}^{n \times (n-m)}$ 是由 $\ker(A)$ 的一组基构成的矩阵. 证明:

$$\text{Ran}(A^T) = \text{Ker}(Z^T).$$

练习 1.29 设 $A, B \in \mathbb{R}^{n \times n}$, 证明:



- (1) 若 $AB = 0$, 则 $\text{rank}(A) + \text{rank}(B) \leq n$.
 (2) 若 $A^2 = A$, 则 $\text{rank}(A) + \text{rank}(I - A) = n$.
 (3) 若 $A^2 = I$, 则 $\text{rank}(I + A) + \text{rank}(I - A) = n$.

练习 1.30 设 $A \in \mathbb{R}^{n \times n}$ 和 k ($k \geq 1$) 次首 1 多项式 $p(t)$. 证明:

- (1) 若 (λ, x) 是 A 的一个特征对, 则 $(p(\lambda), x)$ 是 $p(A)$ 的一个特征对.
 (2) 反之, 若 μ 是 $p(A)$ 的一个特征值, 则存在 A 的一个特征值 λ , 使得 $\mu = p(\lambda)$.

练习 1.31 设 $A, B \in \mathbb{R}^{n \times n}$ 都是正交矩阵, 且 $\det(A) = -\det(B)$. 证明: $A + B$ 奇异.

练习 1.32 证明: 矩阵 $\frac{xx^T}{\|x\|_2^2}$ 是正交投影矩阵, 其中 $x \in \mathbb{R}^n$ 是任意非零向量.

练习 1.33 证明: 定义在 $\mathbb{R}^{n \times n}$ 上的算子范数和 F -范数都是相容范数.

练习 1.34 证明:

- (1) 对任意的算子范数 $\|\cdot\|$, 有 $\|I\| = 1$;
 (2) 对任意的相容范数 $\|\cdot\|$, 有 $\|I\| \geq 1$.

练习 1.35 证明: $\|A\| \triangleq \max_{1 \leq i, j \leq n} |a_{ij}|$ 是矩阵范数, 但不是相容范数.

练习 1.36 将指标集 $\mathcal{Z}_n = \{1, 2, \dots, n\}$ 拆分成 p 个互不相交的非空子集 $\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_p$ 之和, 即

$$\mathcal{Z}_n = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \dots \cup \mathcal{T}_p.$$

定义

$$\|A\|_{\mathcal{T}} \triangleq \max_{1 \leq k \leq p} \|A_{\mathcal{T}_k}\|_2,$$

其中 $A_{\mathcal{T}_k}$ 表示由 $\mathcal{T}_k$ 指定的行组成的子矩阵. 证明: $\|A\|_{\mathcal{T}}$ 是矩阵范数. (不要求满足相容性)

练习 1.37 设 $A \in \mathbb{R}^{n \times n}$ 是正交矩阵. 证明: $\det(A) = \pm 1$.

练习 1.38 设 $A \in \mathbb{R}^{n \times n}$. 证明: $\text{rank}(A) = 1$ 的充要条件是存在非零向量 $a, b \in \mathbb{R}^n$ 使得 $A = ab^T$.

练习 1.39 设 A_k 是 $A \in \mathbb{R}^{n \times n}$ 的一个 k 阶子矩阵. 证明: $\|A_k\|_p \leq \|A\|_p$.

练习 1.40 设 $A \in \mathbb{R}^{n \times n}$, $U, V \in \mathbb{R}^{n \times n}$ 是正交矩阵. 证明:

$$\|UAV\|_2 = \|A\|_2, \quad \|UAV\|_F = \|A\|_F.$$

练习 1.41 设 $A, B \in \mathbb{R}^{n \times n}$, 矩阵

$$C = \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}, \quad D = \begin{bmatrix} 0 & A \\ B & 0 \end{bmatrix}.$$

证明: $\|C\|_2 = \max\{\|A\|_2, \|B\|_2\}$, $\|D\|_2 = \max\{\|A\|_2, \|B\|_2\}$.

练习 1.42 设 $\alpha \in \mathbb{R}$, $x \in \mathbb{R}^n$, 矩阵 $A = \begin{bmatrix} \alpha & x^T \\ x & 0 \end{bmatrix}$. 证明: $\|A\|_2 \leq |\alpha| + \|x\|_2$.

练习 1.43 设 $P \in \mathbb{R}^{n \times n}$ 是一个投影矩阵, 且秩为 r , 计算 P 的所有特征值.

练习 1.44 (引理 1.27) 设 $P \in \mathbb{R}^{n \times n}$ 是一个投影矩阵. 证明: $\text{Ker}(P) = \text{Ran}(I - P)$.



练习 1.45 设 P 是从 $\mathbb{S}$ 沿 $\mathbb{S}_1$ 到 $\mathbb{S}_2$ 上的投影变换, $V = [v_1, v_2, \dots, v_m]$ 构成 $\mathbb{S}_1$ 的一组基, $W = [w_1, w_2, \dots, w_{n-m}]$ 构成 $\mathbb{S}_2$ 的一组基. 证明:

$$P = \begin{bmatrix} W & 0 \end{bmatrix} \begin{bmatrix} W & V \end{bmatrix}^{-1}.$$

练习 1.46 设 $A \in \mathbb{R}^{m \times n}$ ($m < n$) 是满秩矩阵, $Z \in \mathbb{R}^{n \times (n-m)}$ 是由 $\text{Ker}(A)$ 的一组基构成的矩阵. 证明:

- (1) $P_Z \triangleq Z(Z^T Z)^{-1} Z^T$ 是 $\text{Ker}(A)$ 上的正交投影算子;
- (2) $P_A \triangleq A^T (A A^T)^{-1} A$ 是 $\text{Ran}(A^T)$ 上的正交投影算子;
- (3) $P_Z = I - P_A$.

练习 1.47 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, $X, Y \in \mathbb{R}^{n \times m}$ ($n \geq m$).

证明: 若 $Y^T A^{-1} X - I$ 非奇异, 则 $A - X Y^T$ 可逆, 且

$$(A - X Y^T)^{-1} = A^{-1} - A^{-1} X (Y^T A^{-1} X - I)^{-1} Y^T A^{-1}.$$

练习 1.48* 设 $A = \begin{bmatrix} B & C \\ 0 & D \end{bmatrix}$, 其中 $B, D \in \mathbb{C}^{n \times n}$ 均为上三角矩阵, 且 B 和 D 的对角线元素互不相等. 证明: 存在矩阵 S , 使得 $S^{-1} A S = \begin{bmatrix} B & 0 \\ 0 & D \end{bmatrix}$.

(提示: 可设 $S = \begin{bmatrix} I & \tilde{S} \\ 0 & I \end{bmatrix}$)

练习 1.49 设 $x, y, z \in \mathbb{R}^n$, 试证明:

- (1) $[x \otimes z, y \otimes z] = [x, y] \otimes z$;
- (2) $[x \otimes y, x \otimes z] = x \otimes [y, z]$.

练习 1.50 设 $a \in \mathbb{R}$, 证明:

$$\begin{bmatrix} 1 & -a & & & \\ & 1 & -a & & \\ & & \ddots & \ddots & \\ & & & \ddots & -a \\ & & & & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & a & a^2 & \cdots & a^{n-1} \\ & 1 & a & \ddots & \vdots \\ & & \ddots & \ddots & a^2 \\ & & & \ddots & a \\ & & & & 1 \end{bmatrix}.$$



第二讲 线性方程组直接方法

Linear algebra — in particular, *the solution of linear systems of equations* — lies at the heart of most calculations in scientific computing.

— Dongarra & Eijkhout [26], 2000.

考虑线性方程组

$$Ax = b, \quad A \in \mathbb{R}^{n \times n}, \quad b \in \mathbb{R}^n. \quad (2.1)$$

线性方程组的求解有着非常广泛的应用背景, 科学计算中的很多问题最后都可能归结为求解一个或多个线性方程组. 从纯数学角度来看, 这个问题已经得到了完美的解决, 因为它的解可以通过行列式直接表示出来, 即 Cramer 法则. 但在实际计算中, 由于运算量增长速度太快, 当 n 较大时, 用 Cramer 法则解线性方程组是不可行的. 另外, 由于实际计算中的舍入误差, 可能会导致一系列非常严重的问题.

🔴 一个从纯数学角度看似非常简单的问题, 实际计算时可能会非常困难, 有时甚至可能是一个无法解决的问题.

一般来说, 求解线性方程组的数值方法可以分为两类: **直接法**与**迭代法**. 本讲介绍直接法, 也就是 Gauss 消去法及其各种改进形式. 直接法具有良好的稳定性和健壮性, 而且在有限步内终止, 因此在工程界很受欢迎. 但由于运算量是 $\mathcal{O}(n^3)$, 对于大规模问题, 所需时间会很长 (这里 n 表示未知量的个数). 目前, Gauss 消去法是求解中小规模线性方程组或某些具有特殊结构的大规模稀疏线性方程组的首选方法.

🔴 Gauss 消去法的思想可以追溯到公元一世纪左右的《九章算术》, Newton, Gauss, Lagrange 等数学家都对该方法做出了贡献, 相关历史可以参见 [50].

关于线性方程组直接法的相关参考文献

- ▶ G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th, 2013. [49]
- ▶ J.W. Demmel, *Applied Numerical Linear Algebra*, 1997. [23]
- ▶ L. N. Trefethen and D. Bau, III, *Numerical Linear Algebra*, 1997. [103]
- ▶ I. S. Duff, A. M. Erisman and J. K. Reid, *Direct Methods for Sparse Matrices*, 2nd, 2017. [30]
- ▶ T. A. Davis, *Direct Methods for Sparse Linear Systems*, SIAM, 2006. [22]

后面两个文献主要是介绍大规模稀疏线性方程组的直接解法.

在本讲中, 我们总是假定系数矩阵 A 是非奇异的, 即线性方程组 (2.1) 的解存在且唯一. 另外, 为了讨论方便, 我们只考虑实数情形, 对于复系数线性方程组, 其求解方法是类似的.

2.1 LU 分解与 Gauss 消去法

2.1.1 LU 分解

Gauss 消去法本质上就是对系数矩阵 A 进行 **LU 分解**, 即将 A 分解成两个矩阵的乘积

$$A = LU, \quad (2.2)$$

其中 L 是单位下三角矩阵, U 为非奇异上三角矩阵, 然后再对两个三角方程组进行求解: 假定矩阵 A 存在 LU 分解 (2.2), 则方程组 (2.1) 就转化为下面两个三角方程组

$$\begin{cases} Ly = b, \\ Ux = y. \end{cases}$$

显然, 上式中的两个三角方程组都非常容易求解.

将一个复杂问题分解成若干相对简单的问题, 是矩阵分解的一个主要目的.

基于 LU 分解的 Gauss 消去法描述如下:

算法 2.1 Gauss 消去法

- 1: 对 A 进行 LU 分解: $A = LU$, 其中 L 为单位下三角矩阵, U 为非奇异上三角矩阵;
 - 2: 利用向前回代, 求解 $Ly = b$, 即得 $y = L^{-1}b$;
 - 3: 利用向后回代, 求解 $Ux = y$, 即得 $x = U^{-1}y = (LU)^{-1}b = A^{-1}b$.
-

我们知道, 当系数矩阵 A 非奇异时, 方程组 (2.1) 总是存在唯一解. 但是, 并不是每个非奇异矩阵都存在 LU 分解.

定理 2.1 (LU 分解的存在性和唯一性) 矩阵 $A \in \mathbb{R}^{n \times n}$ 存在 LU 分解 (即存在单位下三角矩阵 L 和非奇异上三角矩阵 U 使得 $A = LU$) 的充要条件是 A 的所有顺序主子矩阵都非奇异. 进一步, 若 A 存在 LU 分解, 则分解是唯一的. (板书)

记 D 为 U 的对角线部分, 则 $A = LD\tilde{U}$, 其中 $\tilde{U} = D^{-1}U$ 是单位上三角矩阵. 因此我们就有下面的 **LDU 分解**.

推论 2.2 (LDU 分解) 设 $A \in \mathbb{R}^{n \times n}$ 的所有顺序主子矩阵都非奇异, 则 A 存在 LDU 分解, 即存在单位下三角矩阵 L 和单位上三角矩阵 U , 以及非奇异对角矩阵 D , 使得 $A = LDU$, 其中 L, U, D 都是唯一的. 反之, 若 A 存在 LDU 分解, 则 A 的所有顺序主子矩阵都非奇异.

对角占优情形

一般的非奇异矩阵不一定存在 LU 分解, 但如果 A 是对角占优的, 则存在 LU 分解. 我们可以证明, 严格对角占优矩阵在 LU 分解中保持严格对角占优性 (见课后练习), 因此通过数学归纳



法可以证明严格对角占优矩阵一定存在 LU 分解. 事实上, 只要矩阵列对角占优且非奇异, 则一定存在 LU 分解 [49].

定理 2.3 [49] 设 $A \in \mathbb{R}^{n \times n}$ 非奇异且列对角占优, 则 A 存在 LU 分解且 L 中的元素的绝对值都不超过 1. (留作练习, 数学归纳法)

 需要注意的是, 在实际计算中, 由于舍入误差的影响, 即使是严格对角占优矩阵, 如果对角占优性不是很强的话, 很有可能会失去对角占优性, 从而导致 LU 分解失败.

2.1.2 LU 分解的实现

矩阵的 LU 分解可以通过初等行变换来实现. 给定矩阵

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

- 第一步: 假定 $a_{11} \neq 0$, 构造矩阵

$$L_1 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \\ l_{n1} & 0 & 0 & \cdots & 1 \end{bmatrix}, \quad \text{其中 } l_{i1} = \frac{a_{i1}}{a_{11}}, i = 2, 3, \dots, n.$$

易知 L_1 的逆为

$$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -l_{21} & 1 & 0 & \cdots & 0 \\ -l_{31} & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \\ -l_{n1} & 0 & 0 & \cdots & 1 \end{bmatrix}.$$

用 L_1^{-1} 左乘 A , 并将所得到的矩阵记为 $A^{(1)}$, 则

$$A^{(1)} = L_1^{-1}A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{bmatrix}.$$

即左乘 L_1^{-1} 后, A 的第一列中除第一个元素外其它都变为 0.

- 第二步: 类似地, 我们可以将上面的操作作用在 $A^{(1)}$ 的子矩阵 $A^{(1)}(2:n, 2:n)$ 上, 将其第



一列除第一个元素外都变为 0. 也就是说, 假定 $a_{22}^{(1)} \neq 0$, 构造矩阵

$$L_2 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ 0 & l_{n2} & 0 & \cdots & 1 \end{bmatrix}, \quad \text{其中} \quad l_{i2} = \frac{a_{i2}^{(1)}}{a_{22}^{(1)}}, i = 3, 4, \dots, n.$$

用 L_2^{-1} 左乘 $A^{(1)}$, 并将所得到的矩阵记为 $A^{(2)}$, 则

$$A^{(2)} = L_2^{-1} A^{(1)} = L_2^{-1} L_1^{-1} A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & a_{n3}^{(2)} & \cdots & a_{nn}^{(2)} \end{bmatrix}.$$

- 依此类推, 假定 $a_{kk}^{(k-1)} \neq 0$ ($k = 3, 4, \dots, n-1$), 则我们可以构造一系列的矩阵 $L_3, L_4, \dots, L_{n-1}$, 使得

$$L_{n-1}^{-1} \cdots L_2^{-1} L_1^{-1} A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & \cdots & a_{nn}^{(n-1)} \end{bmatrix}$$

为一个上三角矩阵. 我们将这个上三角矩阵记为 U , 并记

$$L \triangleq L_1 L_2 \cdots L_{n-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{bmatrix}, \quad (2.3)$$

则可得

$$A = LU,$$

这就是 A 的 LU 分解.

将上述过程写成算法, 描述如下:

算法 2.2 LU 分解

- 1: Set $L = I, U = 0$ % 将 L 设为单位矩阵, U 设为零矩阵
- 2: **for** $k = 1$ to $n - 1$ **do**
- 3: **for** $j = k$ to n **do**
- 4: $u_{kj} = a_{kj}$ % 更新 U 的第 k 行
- 5: **end for**
- 6: **for** $i = k + 1$ to n **do**



```

7:       $l_{ik} = a_{ik}/a_{kk}$     % 计算  $l_{ik}$ 
8:      for  $j = k + 1$  to  $n$  do
9:           $a_{ij} = a_{ij} - l_{ik}u_{kj}$     % 更新  $A(i, k + 1 : n)$ 
10:      end for
11:  end for
12: end for
13:  $u_{nn} = a_{nn}$     % 更新  $U$  的最后一行

```

算法评价的重要指标: 时间复杂度

评价算法的一个主要指标是执行时间, 但这依赖于计算机硬件和编程技巧等, 因此直接给出算法执行时间是不太现实的. 所以我们通常是统计算法中算术运算 (加减乘除) 的次数. 在矩阵计算中, 大多仅仅涉及加减乘除和开方运算. 一般情况下, 加减运算次数与乘法运算次数具有相同的量级, 而除法运算和开方运算次数具有更低的量级.

计算优化

为了尽可能地减少运算量, 在实际计算中, 数, 向量和矩阵做乘法运算时的先后执行次序为: 先计算数与向量的乘法, 然后计算矩阵与向量的乘法, 最后才计算矩阵与矩阵的乘法. 比如计算 αABx , 其中 α 是数, A, B 是矩阵, x 是向量, 如果按照从左往右计算的话, 则运算量为 $\mathcal{O}(n^3)$, 但是如果先计算 αx , 然后计算 $B(\alpha x)$, 最后再计算 $A(B(\alpha x))$ 的话, 运算量则为 $\mathcal{O}(n^2)$, 相差一个量级.

矩阵 L 和 U 的存储

当 A 的第 i 列 (严格下三角部分) 被用于计算 L 的第 i 列后, 在后面的计算中不再被使用. 而 A 的第 i 行 (上三角部分) 更新后就是 U 的第 i 行. 因此, 为了节省存储空间, 我们可以在计算过程中将 L 的第 i 列存放在 A 的第 i 列 (严格下三角部分, L 的对角线全部为 1, 不需要存储), 将 U 的第 i 行存放在 A 的第 i 行 (上三角部分), 这样就不需要另外分配空间存储 L 和 U . 计算结束后, A 的上三角部分为 U , 其严格下三角部分为 L 的绝对下三角部分. 此时算法可以描述为:

算法 2.3 LU 分解 (用 A 存储 L 和 U)

```

1: for  $k = 1$  to  $n - 1$  do
2:     for  $i = k + 1$  to  $n$  do
3:          $a_{ik} = a_{ik}/a_{kk}$ 
4:     for  $j = k + 1$  to  $n$  do
5:          $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
6:     end for
7: end for
8: end for

```



LU 分解的运算量

由算法 2.2 可知, LU 分解的运算量为

- 乘法次数: $T_p = \sum_{k=1}^{n-1} \sum_{i=k+1}^n \sum_{j=k+1}^n 1 = \sum_{k=1}^{n-1} (n-k)^2 = \frac{1}{6}(2n^3 - 3n^2 + n) = \frac{1}{3}n^3 + \mathcal{O}(n^2)$.
- 加法次数: $T_a = \sum_{k=1}^{n-1} \sum_{i=k+1}^n \sum_{j=k+1}^n 1 = \frac{1}{3}n^3 - \frac{1}{2}n^2 + \frac{1}{6}n = \frac{1}{3}n^3 + \mathcal{O}(n^2)$.
- 除法次数: $T_d = \sum_{k=1}^{n-1} \sum_{i=k+1}^n 1 = \frac{1}{2}n(n-1) = \frac{1}{2}n^2 - \frac{1}{2}n = \frac{1}{2}n^2 + \mathcal{O}(n)$.

因此总运算量 (含乘法、除法与加法) 为 $\frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n = \frac{2}{3}n^3 + \mathcal{O}(n^2)$.

数据更新顺序与计算效率

根据指标的循环次序, 算法 2.3 也称为 KIJ 型 LU 分解. 在实际计算中, 我们一般不建议使用这个算法. 因为对于指标 k 的每次循环, 都需要更新 A 的第 $k+1$ 至第 n 行. 这种反复读写数据的做法会使得计算效率大大降低.

如果数据是按行存储的, 特别是按压缩稀疏行 (CSR) 格式存储的稀疏矩阵, 为了提高计算效率, 我们可以采用下面的 IKJ 型 LU 分解算法. 相较于 KIJ 型每轮循环 (最外层循环) 需要修改 A 的 $n-k$ 行, IKJ 型每轮循环只需修改一行.

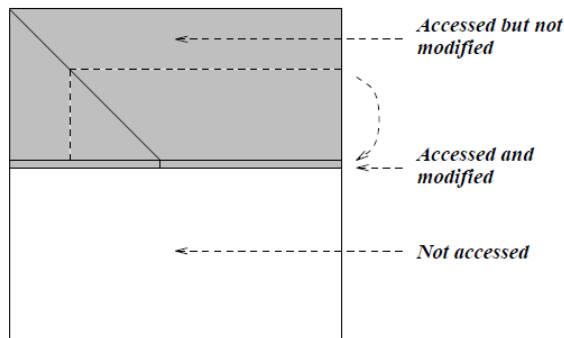
算法 2.4 LU 分解 (IKJ 型)

```

1: for i = 2 to n do
2:   for k = 1 to i - 1 do
3:      $a_{ik} = a_{ik}/a_{kk}$ 
4:     for j = k + 1 to n do
5:        $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
6:     end for
7:   end for
8: end for

```

IKJ 型 LU 分解算法图示 ([95])



思考: 在 C/C++ 语言中, 稠密矩阵是按行存储的, 但 Fortran 和 MATLAB 则是按列存储的, 此时怎样设计算法比较好?

2.1.3 Gauss 消去法

假定矩阵 A 存在 LU 分解 $A = LU$, 则方程组 $Ax = b$ 就转化为求解下面两个三角方程组

$$Ly = b, \quad Ux = y.$$

这两个方程组都非常容易求解, 于是 Gauss 消去法描述如下:



算法 2.5 Gauss 消去法

- 1: 将 A 进行 LU 分解: $A = LU$, 其中 L 为单位下三角矩阵, U 为非奇异上三角矩阵;
- 2: 利用向前回代, 求解 $Ly = b$, 即得 $y = L^{-1}b$;
- 3: 利用向后回代, 求解 $Ux = y$, 即得 $x = U^{-1}y = (LU)^{-1}b = A^{-1}b$.

得到 A 的 LU 分解后, 我们最后需要用回代法求解两个三角方程组, 计算过程描述如下.

算法 2.6 回代求解 $Ly = b$ 和 $Ux = y$

```

1:  $y_1 = b_1/l_{11}$     % 向前回代求解  $Ly = b$ 
2: for  $i = 2 : n$  do
3:   for  $j = 1 : i - 1$  do
4:      $b_i = b_i - l_{ij}y_j$ 
5:   end for
6:    $y_i = b_i/l_{ii}$ 
7: end for
8:  $x_n = y_n/u_{nn}$     % 向后回代求解  $Ux = y$ 
9: for  $i = n - 1 : -1 : 1$  do
10:  for  $j = n : -1 : i + 1$  do
11:     $y_i = y_i - u_{ij}x_j$ 
12:  end for
13:   $x_i = y_i/u_{ii}$ 
14: end for

```

如果数据是按列存储的, 则采用列存储方式效率会高一些. 下面是以 $Ux = y$ 为例, 描述按列存储时的回代求解过程.

算法 2.7 向后回代求解 $Ux = y$ (列存储方式)

```

1: for  $k = n : -1 : 1$  do
2:    $x_k = y_k/u_{kk}$ 
3:   for  $i = k - 1 : -1 : 1$  do
4:      $y_i = y_i - x_k u_{ik}$ 
5:   end for
6: end for

```

计算复杂度

这两个算法的运算量: 乘法和加法均 $n(n-1)$ 次, 除法 $2n$ 次. 加上 LU 分解的运算量, Gauss 消去法的总运算量为 $\frac{2}{3}n^3 + \mathcal{O}(n^2)$.



 可以证明, 以上两个算法都是向后稳定的 (componentwise backward stable) [61].

2.1.4 选主元 LU 分解

在 LU 分解过程中, 我们称 $a_{kk}^{(k-1)}$ 为主元. 如果 $a_{kk}^{(k-1)} = 0$, 则算法就无法进行下去. 即使 $a_{kk}^{(k-1)}$ 不为零, 但如果 $|a_{kk}^{(k-1)}|$ 的值很小, 由于舍入误差的原因, 也可能会给计算结果带来很大的误差. 此时我们就需要通过选主元来解决这个问题.

例 2.1 用 LU 分解求解线性方程组 $Ax = b$, 其中 $A = \begin{bmatrix} 0.02 & 61.3 \\ 3.43 & -8.5 \end{bmatrix}$, $b = \begin{bmatrix} 61.5 \\ 25.8 \end{bmatrix}$, 要求在运算过程中保留 3 位有效数字. (板书)

选主元是通过利用置换矩阵 (也称排列矩阵) 对行进行交换来实现的. 首先介绍置换矩阵的一些基本性质.

引理 2.4 设 $P \in \mathbb{R}^{n \times n}$ 为置换矩阵, $A \in \mathbb{R}^{n \times n}$ 为任意矩阵, 则

- (1) PA 相当于将 A 的行进行置换; AP 相当于将 A 的列进行置换;
- (2) $P^{-1} = P^T$, 即 P 是正交矩阵;
- (3) $\det(P) = \pm 1$;
- (4) 置换矩阵的乘积仍然是置换矩阵.

定理 2.5 (选主元 LU 分解的存在性) 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, 则存在置换矩阵 P_L, P_R , 以及单位下三角矩阵 L 和非奇异上三角矩阵 U , 使得

$$P_L A P_R = LU.$$

(板书)

 定理 2.5 也可以利用定理 2.1 的结论来证明, 见习题 2.3.

 事实上, 定理 2.5 中的 P_L 和 P_R 只有一个必需的.

置换矩阵的选取

问题: 第 k 步时, 如何选取置换矩阵 $P_L^{(k)}$ 和 $P_R^{(k)}$?

选法一. 选取 $P_L^{(k)}$ 和 $P_R^{(k)}$ 使得主元为剩下的矩阵中绝对值最大, 这种选取方法称为“全主元 Gauss 消去法”, 简称 GECP (Gaussian elimination with complete pivoting);

选法二. 选取 $P_L^{(k)}$ 和 $P_R^{(k)}$ 使得主元为第 k 列中第 k 到第 n 个元素中, 绝对值最大, 这种选取方法称为“部分选主元 Gauss 消去法”, 简称 GEPP (Gaussian elimination with partial pivoting), 此时 $P_R^{(k)} = I$, 因此也称为列主元 Gauss 消去法.



- 🔪 (1) GECP 比 GEPP 更稳定, 但工作量太大, 在实际应用中通常使用 GEPP 算法.
 (2) GEPP 算法能保证 L 所有的元素的绝对值都不超过 1.

算法 2.8 部分选主元 LU 分解

```

1:  $p = 1 : n$     % 用于记录置换矩阵
2: for  $k = 1$  to  $n - 1$  do
3:    $[a_{\max}, l] = \max_{k \leq i \leq n} |a_{ik}|$     % 选列主元, 其中  $l$  表示主元所在的行
4:   if  $l \neq k$  then
5:     for  $j = 1$  to  $n$  do
6:        $a_{tmp} = a_{kj}, a_{kj} = a_{lj}, a_{lj} = a_{tmp}$     % 交换  $A$  的第  $k$  行与第  $l$  行
7:     end for
8:      $p_{tmp} = p(k), p(k) = p(l), p(l) = p_{tmp}$     % 更新置换矩阵
9:   end if
10:  for  $i = k + 1$  to  $n$  do
11:     $a_{ik} = a_{ik} / a_{kk}$ 
12:    for  $j = k + 1$  to  $n$  do
13:       $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
14:    end for
15:  end for
16: end for
  
```

例 2.2 用部分选主元 LU 分解求解线性方程组 $Ax = b$, 其中 $A = \begin{bmatrix} 0.02 & 61.3 \\ 3.43 & -8.5 \end{bmatrix}$, $b = \begin{bmatrix} 61.5 \\ 25.8 \end{bmatrix}$, 要求在运算过程中保留 3 位有效数字. (板书)

2.1.5 矩阵求逆

我们可以通过部分选主元 LU 分解来计算矩阵的逆. 设 $PA = LU$, 则

$$A^{-1} = U^{-1}L^{-1}P,$$

等价于求解下面 $2n$ 个三角线性方程组

$$Ly_i = Pe_i, \quad Ux_i = y_i, \quad i = 1, 2, \dots, n.$$

🔪 **思考:** 也可以分别计算 L^{-1} 和 U^{-1} , 然后相乘. 哪种方法划算?



2.1.6 分块 LU 分解

为了提高计算效率, 实际计算中通常采用 **分块 LU 分解**, 即

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1p} \\ A_{21} & A_{22} & \cdots & A_{2p} \\ \vdots & & \ddots & \vdots \\ A_{p1} & A_{p2} & \cdots & A_{pp} \end{bmatrix} = \begin{bmatrix} I_1 & & & \\ L_{21} & I_2 & & \\ \vdots & \ddots & \ddots & \\ L_{p1} & \cdots & L_{p,p-1} & I_p \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} & \cdots & U_{1p} \\ & U_{22} & \cdots & U_{2p} \\ & & \ddots & \vdots \\ & & & U_{pp} \end{bmatrix} = LU,$$

其中 A_{ii} 是非奇异的方阵.

与定理 2.1 相类似, 对于分块 LU 分解, 我们有下面的结论.

定理 2.6 (分块 LU 分解的存在性和唯一性) 矩阵 A 存在唯一分块 LU 分解的充要条件是 A 的所有顺序分块主子矩阵都非奇异. (留作练习)

几点笔记

- ▶ 分块 LU 分解不是 LU 分解, 若 A 存在 LU 分解, 则一定存在分块 LU 分解, 反之不成立.
- ▶ 分块 LU 分解与普通 LU 分解的总运算量是一样的, 但分块 LU 分解可以借助 3 级 BLAS 运算, 因此效率更高.
- ▶ 在计算分块 LU 分解过程中需要求解一系列小规模线性方程组 (以 A_{ii} 为系数矩阵), 可以采用普通 (选主元) LU 分解方法求解.

2.2 特殊方程组的求解

如果系数矩阵具有一定的特殊结构或性质, 则可以充分利用这些特殊结构或性质来设计高效的数值算法. 本节考虑以下特殊方程组的求解:

- 对称正定矩阵
- 对称不定矩阵
- 三对角矩阵
- 带状矩阵
- Toeplitz 矩阵

2.2.1 对称正定线性方程组

如果 A 是对称正定矩阵, 则我们有下面的 Cholesky 分解.

定理 2.7 (Cholesky 分解) 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, 则存在唯一的对角线元素为正的下三角矩阵 L , 使得

$$A = LL^T.$$



该分解称为 **Cholesky 分解**.

(板书)

该定理也可以通过 LU 分解的存在唯一性来证明.

Cholesky 分解有时也可以写为

$$A = R^T R,$$

其中 R 是对角线元素为正的上三角矩阵.

Cholesky 分解的实现

我们利用待定系数法来计算对称正定矩阵的 Cholesky 分解. 设 $A = LL^T$, 即

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ & l_{22} & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & l_{nn} \end{bmatrix}.$$

直接比较等式两边的元素可得

$$a_{ij} = \sum_{k=1}^n l_{ik}l_{jk} = l_{jj}l_{ij} + \sum_{k=1}^{j-1} l_{ik}l_{jk}, \quad i, j = 1, 2, \dots, n.$$

先计算 $l_{11} = \sqrt{a_{11}}$, 然后计算 L 第 1 列的其他元素:

$$l_{i1} = \frac{a_{i1}}{l_{11}}, \quad i = 2, 3, \dots, n.$$

依此类推, 可逐次计算出 L 的第 $2, 3, \dots, n$ 列. 具体算法描述如下.

算法 2.9 Cholesky 分解 (按列计算)

```

1: for  $j = 1$  to  $n$  do
2:    $l_{jj} = \left( a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 \right)^{1/2}$  % 先计算对角线元素
3:   for  $i = j + 1$  to  $n$  do
4:      $l_{ij} = \frac{1}{l_{jj}} \left( a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk} \right)$  % 计算  $L$  的第  $j$  列
5:   end for
6: end for

```

关于 Cholesky 算法的几点说明

- 与 LU 分解一样, 可以利用 A 的下三角部分来存储 L ;
- Cholesky 分解算法的运算量: 乘法次数 $\frac{1}{6}n^3 + \frac{1}{6}n = \frac{1}{3}n^3 + \mathcal{O}(n^2)$, 加法次数 $\frac{1}{6}n^3 + \frac{1}{6}n = \frac{1}{3}n^3 + \mathcal{O}(n^2)$, 除法次数 $\frac{1}{2}n^2 - \frac{1}{2}n$, 开方次数 n .
- Cholesky 分解算法是稳定的 (稳定性与全主元 Gauss 消去法相当), 故不需要选主元.
- 利用 Cholesky 分解求解对称正定线性方程组的算法也称为**平方根法**.



例 2.3 已知矩阵 $A = \begin{bmatrix} 4 & 2 & 8 & 0 \\ 2 & 10 & 10 & 9 \\ 8 & 10 & 21 & 6 \\ 0 & 9 & 6 & 34 \end{bmatrix}$, 计算 A 的 Cholesky 分解. (板书)

LDL^T 分解

为了避免开方运算, 我们可以将 A 分解为: $A = LDL^T$, 即

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & & \ddots & \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{bmatrix} \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} 1 & l_{21} & \cdots & l_{n1} \\ & 1 & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}.$$

通过待定系数法可得

$$a_{ij} = \sum_{k=1}^n l_{ik} d_k l_{jk} = d_j l_{ij} + \sum_{k=1}^{j-1} l_{ik} d_k l_{jk}, \quad j = 1, 2, \dots, n, \quad i = j+1, j+2, \dots, n.$$

因此我们也可以依次计算出 D 和 L 的各个元素, 这就是 **LDL^T 分解**.

基于 LDL^T 分解求解对称正定线性方程组的算法称为**改进的平方根法**, 其优点是不需要计算平方根.

算法 2.10 改进的平方根法

```

1: % 先计算 LDLT 分解
2: for j = 1 to n do
3:    $d_j = a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 d_k$ 
4:   for i = j + 1 to n do
5:      $l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} d_k l_{jk}) / d_j$ 
6:   end for
7: end for
8: % 解方程组:  $Ly = b$  和  $DL^T x = y$ 
9:  $y_1 = b_1$ 
10: for i = 2 to n do
11:    $y_i = b_i - \sum_{k=1}^{i-1} l_{ik} y_k$ 
12: end for
13:  $x_n = y_n / d_n$ 
14: for i = n - 1 to 1 do
15:    $x_i = y_i / d_i - \sum_{k=i+1}^n l_{ki} x_k$ 

```



16: end for

 对于稀疏对称正定线性方程组, [?] 中详细讨论了各种具体的实施细节.

2.2.2 对称不定线性方程组

设 $A \in \mathbb{R}^{n \times n}$ 是非奇异的对称不定矩阵. 若 A 存在 LU 分解, 即 $A = LU$, 则可写成

$$A = LDL^T,$$

其中 D 是由 U 的对角线元素构成的对角矩阵. 然而, 当 A 不定时, 其 LU 分解不一定存在. 若采用选主元 LU 分解, 则其对称性可能会被破坏. 为了保持对称性, 在选主元时必须对行和列进行同样的置换, 即选取置换矩阵 P , 使得

$$PAP^T = LDL^T. \quad (2.4)$$

通常称 (2.4) 为对称矩阵的 LDL^T 分解. 但遗憾的是, 这样的置换矩阵可能不一定存在, 即分解 (2.4) 不一定存在.

例 2.4 设对称矩阵

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

由于 A 的对角线元素都是 0, 对任意置换矩阵 P , 矩阵 PAP^T 的对角线元素仍然都是 0. 因此, 矩阵 A 不存在 LDL^T 分解 (2.4).

块 LDL^T 分解

由于对称不定矩阵不一定存在 LDL^T 分解, 于是人们提出了块 LDL^T 分解, 不再要求 D 是对角矩阵, 它可以是**拟对角矩阵** (即块对角矩阵, 且对角块至多是 2 阶的).

可以证明, 对于一个非奇异对称矩阵 $A \in \mathbb{R}^{n \times n}$, 总存在置换矩阵 P_1 使得 (见习题 2.9)

$$P_1AP_1^T = \begin{bmatrix} B & E^T \\ E & C \end{bmatrix},$$

其中 $B \in \mathbb{R}$ 或 $B \in \mathbb{R}^{2 \times 2}$, 且非奇异. 因此可以对 $P_1AP_1^T$ 进行块对角化, 即

$$P_1AP_1^T = \begin{bmatrix} I & 0 \\ EB^{-1} & I \end{bmatrix} \begin{bmatrix} B & 0 \\ 0 & C - EB^{-1}E^T \end{bmatrix} \begin{bmatrix} I & B^{-1}E^T \\ 0 & I \end{bmatrix},$$

其中 $C - EB^{-1}E^T$ 是 Schur 补.

由于 $C - EB^{-1}E^T$ 仍然是非奇异对称矩阵, 不断重复以上过程, 就可以得到 A 的**块 LDL^T 分解**:

$$PAP^T = L\tilde{D}L^T,$$



其中 $\tilde{D}$ 是拟对角矩阵.

与选主元 LU 分解类似, 我们需要考虑块 LDL^T 分解的选主元策略, 即如何选取置换矩阵 P . 一方面使得分解可以进行下去, 另一方面提高算法的稳定性. Kahan 于 1965 年首先考虑了选主元块 LDL^T 分解. 目前常用的策略有

- 全主元策略 (Bunch-Parlett 策略): Bunch 和 Parlett [17] 于 1971 年提出了全主元策略来选取置换矩阵, 并证明了其稳定性, 指出其向后误差上界与全主元 Gauss 消去法几乎一样 [16]. 但需要进行 $n^3/6$ 次比较运算, 代价比较昂贵.
- 部分选主元策略 (Bunch-Kaufman 策略): 为了减少比较运算, Bunch 和 Kaufman [18] 于 1977 年提出了部分选主元策略. 这种策略在每次选主元时, 只需搜索两列, 因此将比较运算复杂度降低到了 $\mathcal{O}(n^2)$ 量级. 而且这种选主元策略具有较满意的向后稳定性 [60, 61], 运算量为 $\frac{1}{3}n^3 + \mathcal{O}(n^2)$, 因此被广泛使用. 著名的程序库 LAPACK 一开始就采用了这种策略. 具体实施也可以参见 [61], 或 [25] (分块版本).
- Rook 策略: 采用 Bunch-Kaufman 策略得到的矩阵 L 的元素不能得到很好的控制. 为了克服由此带来的不稳定性, Ashcraft, Grimes 和 Lewis [6] 于 1998 年提出采用对称的 Rook 选主元策略, 以潜在的高比较运算量, 可以将矩阵 L 的元素控制在 1 左右. 这种对称的 Rook 选主元策略最先用在 Gauss 消去法中 [81]. Rook 策略整体上与 Bunch-Kaufman 策略类似, 但在选主元时加了一层迭代, 从而能够提供更高的精度. Cheng 在其博士论文 [20] 中对这两种策略进行了数值测试, 给出了两种例子, 分别说明 Rook 策略和 Bunch-Kaufman 策略各有优势. LAPACK 从 3.5.0 版本开始增加了 Rook 策略.

 以上算法是针对稠密的对称不定线性方程组. 对于稀疏的对称不定矩阵, 可以采用 Duff 等 [31, 32] 提出的多波前法 (Multifrontal), 或者 Liu [78] 提出的稀疏阈值算法.

Aasen 算法

Aasen[1] 于 1971 年提出了下面的分解

$$PAP^T = LTL^T, \quad (2.5)$$

其中 P 为置换矩阵, L 为单位下三角矩阵, T 为对称三对角矩阵.

2011 年, Rozložník, Shklarski 和 Toledo [91] 给出了基于现代化计算机结构和 BLAS-3 的分块三对角化算法. 该算法本质上是 Aasen 算法的分块形式.

 Aasen 算法不如块 LDL^T 分解使用广泛. 2016 年 12 月, LAPACK 发行了更新版本 3.7.0, 由 Yamazaki 加入了 Aasen 算法 (<http://www.netlib.org/lapack/lapack-3.7.0.html>).

 目前大部分软件都采用部分选主元块 LDL^T 分解来求解对称线性方程组. 关于 Aasen 算法和块 LDL^T 分解的比较也可参见 [6, 11] 等.



2.2.3 三对角线性方程组

考虑三对角线性方程组 $Ax = f$, 其中 A 是三对角矩阵:

$$A = \begin{bmatrix} b_1 & c_1 & & & \\ a_1 & \ddots & \ddots & & \\ & \ddots & \ddots & c_{n-1} & \\ & & a_{n-1} & b_n & \end{bmatrix}.$$

我们假定

$$|b_1| > |c_1| > 0, \quad |b_n| > |a_{n-1}| > 0, \quad |b_i| \geq |a_{i-1}| + |c_i|, \quad i = 2, \dots, n-1. \quad (2.6)$$

且

$$a_i c_i \neq 0, \quad i = 1, \dots, n-1. \quad (2.7)$$

即 A 是不可约对角占优的, 因此是非奇异的. 此时, 我们可以得到下面的三角分解

$$A = \begin{bmatrix} b_1 & c_1 & & & \\ a_1 & \ddots & \ddots & & \\ & \ddots & \ddots & c_{n-1} & \\ & & a_{n-1} & b_n & \end{bmatrix} = \begin{bmatrix} \alpha_1 & & & & \\ a_1 & \alpha_2 & & & \\ & \ddots & \ddots & & \\ & & a_{n-1} & \alpha_n & \end{bmatrix} \begin{bmatrix} 1 & \beta_1 & & & \\ & 1 & \ddots & & \\ & & \ddots & \beta_{n-1} & \\ & & & 1 & \end{bmatrix} \triangleq LU. \quad (2.8)$$

采用待定系数法, 我们就可以得到以下的递推计算公式:

$$\begin{aligned} \alpha_1 &= b_1, \quad \beta_1 = c_1/\alpha_1 = c_1/b_1, \\ \begin{cases} \alpha_i &= b_i - a_{i-1}\beta_{i-1}, \\ \beta_i &= c_i/\alpha_i = c_i/(b_i - a_{i-1}\beta_{i-1}), \end{cases} \quad i = 2, 3, \dots, n-1 \\ \alpha_n &= b_n - a_{n-1}\beta_{n-1}. \end{aligned}$$

为了使得算法能够顺利进行下去, 我们需要证明 $\alpha_i \neq 0$.

定理 2.8 设三对角矩阵 A 满足条件 (2.6) 和 (2.7). 则 A 非奇异, 且

- (1) $|\alpha_1| = |b_1| > 0$;
- (2) $0 < |\beta_i| < 1, i = 1, 2, \dots, n-1$;
- (3) $0 < |c_i| \leq |b_i| - |a_{i-1}| < |\alpha_i| < |b_i| + |a_{i-1}|, \quad i = 2, 3, \dots, n.$ (板书)

由定理 2.8 可知, 分解 (2.8) 是存在的. 因此, 原方程就转化为求解 $Ly = f$ 和 $Ux = y$. 由此便可得求解三对角线性方程组的**追赶法**也称为 **Thomas 算法** (1949), 其运算量大约为 $8n - 6$.

算法 2.11 追赶法

- 1: $\alpha_1 = b_1$
- 2: $\beta_1 = c_1/b_1$
- 3: $y_1 = f_1/b_1$
- 4: **for** $i = 2$ **to** $n - 1$ **do**



```

5:    $\alpha_i = b_i - a_{i-1}\beta_{i-1}$ 
6:    $\beta_i = c_i/\alpha_i$ 
7:    $y_i = (f_i - a_{i-1}y_{i-1})/\alpha_i$ 
8: end for
9:  $\alpha_n = b_n - a_{n-1}\beta_{n-1}$ 
10:  $y_n = (f_n - a_{n-1}y_{n-1})/\alpha_n$ 
11:  $x_n = y_n$ 
12: for  $i = n - 1$  to 1 do
13:    $x_i = y_i - \beta_i x_{i+1}$ 
14: end for

```

具体计算时, 由于求解 $Ly = f$ 与矩阵 LU 分解是同时进行的, 因此, α_i 可以不用存储.

由于 $|\beta_i| < 1$, 因此在回代求解 x_i 时, 误差可以得到有效控制.

需要指出的是, 我们也可以考虑下面的分解

$$A = \begin{bmatrix} b_1 & c_1 & & & \\ a_1 & \ddots & \ddots & & \\ & \ddots & \ddots & c_{n-1} & \\ & & a_{n-1} & b_n & \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ \gamma_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \gamma_{n-1} & 1 & \end{bmatrix} \begin{bmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & \ddots & & \\ & & \ddots & c_{n-1} & \\ & & & \alpha_n & \end{bmatrix}. \quad (2.9)$$

但此时 $|\gamma_i|$ 可能大于 1. 比如若 $|b_1| < |a_1|$, 则 $|\gamma_1| = |a_1/b_1| > 1$. 此时回代求解时, 误差可能得不到有效控制. 另外一方面, 计算 γ_i 时也可能会产生较大的舍入误差 (大数除以小数). 但如果 A 是列对角占优, 则可以保证 $|\gamma_i| < 1$.

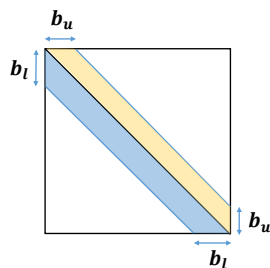
如果 A 是 (行) 对角占优, 则采用分解 (2.8); 如果 A 是列对角占优, 则可采用分解 (2.9).

2.2.4 带状线性方程组

设系数矩阵 $A \in \mathbb{R}^{n \times n}$ 是带状矩阵, 下带宽为 b_l , 上带宽为 b_u , 如右图所示, 即

当 $i > j + b_l$ 或 $i < j - b_u$ 时, 有 $a_{ij} = 0$.

对于带状矩阵, 不带选主元的 LU 分解算法如下.



算法 2.12 带状矩阵的 LU 分解

```

1: for  $k = 1$  to  $n - 1$  do
2:   for  $i = k + 1$  to  $\min(b_l, n)$  do
3:      $a_{ik} = a_{ik}/a_{kk}$ 
4:   for  $j = k + 1$  to  $\min(k + b_u, n)$  do

```



```

5:          $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
6:     end for
7: end for
8: end for

```

如果 A 存在 LU 分解, 则可以验证, L 和 U 也是带状矩阵.

定理 2.9 设 $A \in \mathbb{R}^{n \times n}$ 是带状矩阵, 其下带宽为 b_l , 上带宽为 b_u . 若 A 存在不带选主元的 LU 分解 $A = LU$, 则 L 为下带宽 b_l 的带状矩阵, U 为上带宽 b_u 的带状矩阵. (留作课外自习)

若采用部分选主元的 LU 分解, 则 L 和 U 会有所变化, 但仍具有一定的特殊结构.

定理 2.10 设 $A \in \mathbb{R}^{n \times n}$ 是带状矩阵, 其下带宽为 b_l , 上带宽为 b_u . 设 $PA = LU$ 是 A 的部分选主元 LU 分解, 则 U 为上带宽不超过 $b_l + b_u$ 的带状矩阵, L 为下带宽为 b_l 的“基本带状矩阵”, 即 L 每列的非零元素不超过 $b_l + 1$ 个. (留作课外自习)

 **思考:** 分别统计带状矩阵 LU 分解和部分选主元 LU 分解的运算量.

2.2.5 Toeplitz 线性方程组

设 $T_n \in \mathbb{R}^{n \times n}$ 是 Toeplitz 矩阵, 即

$$T_n = \begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-n+1} \\ t_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & \cdots & t_1 & t_0 \end{bmatrix}.$$

可知 Toeplitz 矩阵是反向对称 (persymmetric) 矩阵, 即关于东北-西南对角线对称. 记 J_n 为 n 阶反向单位矩阵, 即

$$J_n = \begin{bmatrix} & & & 1 \\ & & & \\ & & 1 & \\ & \ddots & & \\ 1 & & & \end{bmatrix}.$$

易知 $J_n^T = J_n^{-1} = J_n$, $J_n^2 = I_n$.

引理 2.11 矩阵 $A \in \mathbb{R}^{n \times n}$ 是反向对称矩阵当且仅当

$$A = J_n A^T J_n \quad \text{或} \quad J_n A = A^T J_n.$$

若 A 可逆, 则可得

$$A^{-1} = J_n^{-1} (A^T)^{-1} J_n^{-1} = J_n (A^{-1})^T J_n,$$

即反向对称矩阵的逆也是反向对称矩阵.



 Toeplitz 矩阵的逆是反向对称矩阵, 但不一定是 Toeplitz 矩阵.

一般情况下, Toeplitz 矩阵的乘积不再是一个 Toeplitz 矩阵, 如果是两个上三角或下三角 Toeplitz 矩阵相乘, 则乘积仍然是 Toeplitz 矩阵.

定理 2.12 设 $T, S \in \mathbb{R}^{n \times n}$ 是上三角 Toeplitz 矩阵, 则 TS 也是上三角 Toeplitz 矩阵. 进一步, 若 T 非奇异, 则 T^{-1} 也是上三角 Toeplitz 矩阵.

Yule-Walker 方程组

我们先考虑一类特殊右端项的线性方程组. 设 T_n 对称正定, 考虑线性方程组

$$T_n y = -t^{(n)}, \quad (2.10)$$

其中 $t^{(n)} = [t_1, t_2, \dots, t_{n-1}, t_n]^T$, 即右端项的前 $n-1$ 个分量为 T_n 第一列的后 $n-1$ 个元素, t_n 任意. 这类线性方程组称为 **Yule-Walker 方程组**.

由于 T_n 对称正定, 所以 $t_0 > 0$. 因此我们可以对 T_n 的对角线元素进行单位化处理. 不失一般性, 我们假定 T_n 的对角线元素为 1, 即

$$T_n = \begin{bmatrix} 1 & t_1 & \cdots & t_{n-1} \\ t_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_1 \\ t_{n-1} & \cdots & t_1 & 1 \end{bmatrix}.$$

由于方程组右端项的特殊性, 我们可以不进行矩阵分解, 而是通过递推方法来求解.

记 T_k 为 T_n 的 k 阶顺序主子矩阵, $t^{(k)}$ 为 $t^{(n)}$ 的前 k 个分量组成的向量. 设 $y^{(k)}$ 是 $T_k y = -t^{(k)}$ 的解, 下面通过递推方法来计算 $T_{k+1} y = -t^{(k+1)}$ 的解 $y^{(k+1)}$. 记

$$y^{(k+1)} = \begin{bmatrix} u^{(k)} \\ \alpha_k \end{bmatrix},$$

则 $T_{k+1} y^{(k+1)} = -t^{(k+1)}$ 可写为

$$\begin{bmatrix} T_k & J_k t^{(k)} \\ (t^{(k)})^T J_k & 1 \end{bmatrix} \begin{bmatrix} u^{(k)} \\ \alpha_k \end{bmatrix} = - \begin{bmatrix} t^{(k)} \\ t_{k+1} \end{bmatrix},$$

即

$$\begin{cases} T_k u^{(k)} = -t^{(k)} - \alpha_k J_k t^{(k)}, \\ (t^{(k)})^T J_k u^{(k)} + \alpha_k = -t_{k+1}. \end{cases} \quad (2.11)$$

$$(2.12)$$

由于 T_k 既是反向对称矩阵, 也是对称矩阵, 因此有 $T_k^{-1} J_k = J_k T_k^{-1}$. 所以根据 (2.11) 可得

$$u^{(k)} = T_k^{-1}(-t^{(k)} - \alpha_k J_k t^{(k)}) = y^{(k)} - \alpha_k T_k^{-1} J_k t^{(k)} = y^{(k)} - \alpha_k J_k T_k^{-1} t^{(k)} = y^{(k)} + \alpha_k J_k y^{(k)}.$$

代入 (2.12) 可得

$$\left(1 + (t^{(k)})^T y^{(k)}\right) \alpha_k = -t_{k+1} - (t^{(k)})^T J_k y^{(k)}.$$



又

$$\begin{bmatrix} I & J_k y^{(k)} \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} T_k & J_k t^{(k)} \\ (t^{(k)})^T J_k & 1 \end{bmatrix} \begin{bmatrix} I & J_k y^{(k)} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} T_k & 0 \\ 0 & 1 + (t^{(k)})^T y^{(k)} \end{bmatrix},$$

由 T_{k+1} 的对称正定性可知 $1 + (t^{(k)})^T y^{(k)} > 0$. 于是可得 $y^{(k+1)}$ 的计算公式

$$\alpha_k = \frac{-t_{k+1} - (t^{(k)})^T J_k y^{(k)}}{1 + (t^{(k)})^T y^{(k)}}, \quad u^{(k)} = y^{(k)} + \alpha_k J_k y^{(k)}, \quad k = 1, 2, \dots, \quad (2.13)$$

运算量为 $\mathcal{O}(k)$. 因此, 我们就可以从一阶 Yule-Walker 方程出发, 利用递推公式 (2.13) 逐步计算出 $T_n y = -r_n$ 的解. 总的运算量大约为 $3n^2$.

为了进一步减少运算量, 我们引入一个辅助变量. 记 $\tau_k \triangleq 1 + (t^{(k)})^T y^{(k)}$, 则

$$\begin{aligned} \tau_{k+1} &= 1 + (t^{(k+1)})^T y^{(k+1)} \\ &= 1 + \begin{bmatrix} (t^{(k)})^T & t_{k+1} \end{bmatrix} \begin{bmatrix} y^{(k)} + \alpha_k J_k y^{(k)} \\ \alpha_k \end{bmatrix} \\ &= 1 + (t^{(k)})^T y^{(k)} + \alpha_k \left(t_{k+1} + (t^{(k)})^T J_k y^{(k)} \right) \\ &= (1 - \alpha_k^2) \tau_k. \end{aligned}$$

于是可得求解 Yule-Walker 方程组的 **Levinson-Durbin 算法** [33, 74] (由 Levinson 和 Durbin 分别于 1946 年和 1960 年独立提出 [70]), 总运算量大约为 $2n^2$.

算法 2.13 求解 Yule-Walker 方程组的 Levinson-Durbin 算法

```

1: 输入数据:  $t = [t_1, t_2, \dots, t_n]$    % 注: 这里假定  $t_0 = 1$ 
2:  $y(1) = -t_1, \tau = 1, \alpha = -t_1$ 
3: for  $k = 1$  to  $n - 1$  do
4:    $\tau = (1 - \alpha^2)\tau$ 
5:    $\alpha = -\frac{1}{\tau} \left( t_{k+1} + \sum_{i=1}^k t_i y(k+1-i) \right)$ 
6:    $y(1:k) = y(1:k) + \alpha y(k:-1:1)$ 
7:    $y(k+1) = \alpha$ 
8: end for
```

 由 (2.13) 可知, 只要 T_{k+1} 非奇异, 就能确保 $1 + (t^{(k)})^T y^{(k)} \neq 0$, 算法就能顺利进行下去. 所以 Levinson-Durbin 算法有意义的充要条件是 T_n 的所有顺序主子式非零, 此时我们称 T_n 是**强正则** (strongly regular) 的.

一般右端项的对称正定 Toeplitz 线性方程组

考虑一般右端项的方程组

$$T_n x = b,$$



其中 T_n 是对称正定 Toeplitz 矩阵, $b = [b_1, b_2, \dots, b_n]^T$. 与求解 Yule-Walker 方程组类似, 我们利用递推方法来求解.

假定 $x^{(k)}$ 和 $y^{(k)}$ 分别是方程组

$$T_k x = [b_1, b_2, \dots, b_k]^T \triangleq b^{(k)} \quad \text{和} \quad T_k y = -[t_1, t_2, \dots, t_k]^T \triangleq -t^{(k)}$$

的解, 即 $x^{(k)} = T_k^{-1} b^{(k)}$, $y^{(k)} = -T_k^{-1} t^{(k)}$. 设 $x^{(k+1)} = \begin{bmatrix} v^{(k)} \\ \beta_k \end{bmatrix}$ 是 $T_{k+1} x = b^{(k+1)}$ 的解, 即

$$\begin{bmatrix} T_k & J_k t^{(k)} \\ (t^{(k)})^T J_k & 1 \end{bmatrix} \begin{bmatrix} v^{(k)} \\ \beta_k \end{bmatrix} = \begin{bmatrix} b^{(k)} \\ b_{k+1} \end{bmatrix}.$$

通过计算可得

$$\begin{cases} v^{(k)} = T_k^{-1} b^{(k)} - \beta_k T_k^{-1} J_k t^{(k)} = x^{(k)} - \beta_k J_k T_k^{-1} t^{(k)} = x^{(k)} + \beta_k J_k y^{(k)}, \\ \beta_k = \frac{b_{k+1} - (t^{(k)})^T J_k x^{(k)}}{1 + (t^{(k)})^T y^{(k)}}. \end{cases}$$

所以, 我们可以先计算 $T_k x = b^{(k)}$ 和 $T_k x = -t^{(k)}$ 的解, 然后利用上述公式得到 $T_{k+1} x = b^{(k+1)}$ 的解, 这就是 Levinson-Durbin 算法, 该算法的总运算量大约为 $4n^2$.

算法 2.14 求解对称正定 Toeplitz 线性方程组的 Levinson-Durbin 算法

```

1: 输入数据:  $t = [t_1, t_2, \dots, t_{n-1}]$  和  $b = [b_1, b_2, \dots, b_n]$    % 这里假定  $t_0 = 1$ 
2:  $y(1) = -t_1$ ,  $x(1) = b_1$ ,  $\tau = 1$ ,  $\alpha = -t_1$ 
3: for  $k = 1$  to  $n - 1$  do
4:    $\tau = (1 - \alpha^2)\tau$ 
5:    $\beta = \frac{1}{\tau} \left( b_{k+1} - \sum_{i=1}^k t_i x(k+1-i) \right)$ 
6:    $x(1:k) = x(1:k) + \beta y(k:-1:1)$ 
7:    $x(k+1) = \beta$ 
8:   if  $k < n - 1$  then
9:      $\alpha = -\frac{1}{\tau} \left( t_{k+1} + \sum_{i=1}^k t_i y(k+1-i) \right)$ 
10:     $y(1:k) = y(1:k) + \alpha y(k:-1:1)$ 
11:     $y(k+1) = \alpha$ 
12:   end if
13: end for
```

 通过 Levinson-Durbin 算法也可以得到 T_n 的 LDL^T 分解 [70], 因此对称正定 Toeplitz 矩阵的 LDL^T 分解运算量为 $\mathcal{O}(n^2)$.



注记

在数学与工程的许多应用中都会出现 Toeplitz 线性方程组, 如样条插值, 时间序列分析, Markov 链, 排队论, 信号与图像处理等. Levinson-Durbin 算法最早用于求解线性预测问题中的 Toeplitz 线性方程组 (即 Yule-Walker 方程组) [74]

$$T_{n+1} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} 0_n \\ E_n \end{bmatrix},$$

其中 $x \in \mathbb{R}^n$ 和 $E_n \in \mathbb{R}$ 是未知量, 0_n 表示长度为 n 的零向量. 显然该线性方程组等价于求解方程组 (2.10). 在线性预测模型中, T_{n+1} 表示相关矩阵 (correlation matrix), E_n 表示 (均方根) 预测误差 (root mean square prediction error).

非对称 Toeplitz 线性方程组

考虑非对称 Toeplitz 线性方程组 $T_n x = b$, 其中

$$T_n = \begin{bmatrix} 1 & t_1 & \cdots & t_{n-1} \\ s_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_1 \\ s_{n-1} & \cdots & s_1 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

假定 T_n 的所有顺序主子矩阵都非奇异, 与 Levinson-Durbin 算法类似, 我们可以通过递归方式来计算 $T_n x = b$ 的解, 但此时需要用到两个辅助线性方程组.

假定 $x^{(k)}$, $y^{(k)}$ 和 $z^{(k)}$ 分别是方程组

$$T_k x = b^{(k)}, \quad T_k^T y = -t^{(k)} \quad \text{和} \quad T_k z = -s^{(k)} \triangleq -[s_1, s_2, \dots, s_k]^T$$

的解. 设 $x^{(k+1)} = \begin{bmatrix} v^{(k)} \\ \beta_k \end{bmatrix}$ 是 $T_{k+1} x = b^{(k+1)}$ 的解, 则可得

$$\begin{bmatrix} T_k & J_k t^{(k)} \\ (t^{(k)})^T J_k & 1 \end{bmatrix} \begin{bmatrix} v^{(k)} \\ \beta_k \end{bmatrix} = \begin{bmatrix} b^{(k)} \\ b_{k+1} \end{bmatrix}.$$

通过计算可得

$$\begin{cases} v^{(k)} = T_k^{-1} b^{(k)} - \beta_k T_k^{-1} J_k t^{(k)} = x^{(k)} - \beta_k J_k T_k^{-1} t^{(k)} = x^{(k)} + \beta_k J_k y^{(k)}, \\ \beta_k = \frac{b_{k+1} - (s^{(k)})^T J_k x^{(k)}}{1 + (s^{(k)})^T y^{(k)}}. \end{cases} \quad (2.14)$$

下面考虑 $y^{(k+1)}$ 的计算. 类似地, 设 $y^{(k+1)} = \begin{bmatrix} u^{(k)} \\ \alpha_k \end{bmatrix}$, 代入 $T_{k+1}^T y^{(k+1)} = -t^{(k+1)}$, 整理后可得

$$\begin{cases} u^{(k)} = y^{(k)} - \alpha_k J_k T_k^{-1} s^{(k)} = y^{(k)} + \alpha_k J_k z^{(k)}, \\ \alpha_k = \frac{-t_{k+1} - (t^{(k)})^T J_k y^{(k)}}{1 + (t^{(k)})^T z^{(k)}}. \end{cases} \quad (2.15)$$



同样, 设 $z^{(k+1)} = \begin{bmatrix} w^{(k)} \\ \gamma_k \end{bmatrix}$, 代入 $T_{k+1}^T z^{(k+1)} = -s^{(k+1)}$, 整理后可得

$$\begin{cases} w^{(k)} = z^{(k)} - \gamma_k J_k \left(T_k^T \right)^{-1} t^{(k)} = z^{(k)} + \gamma_k J_k y^{(k)}, \\ \gamma_k = \frac{-s_{k+1} - (s^{(k)})^T J_k y^{(k)}}{1 + (s^{(k)})^T y^{(k)}}. \end{cases} \quad (2.16)$$

由此, 就可以根据更新公式 (2.14), (2.15), (2.16) 得到 $x^{(k+1)}$, $y^{(k+1)}$ 和 $z^{(k+1)}$. 这样, 我们就可以设计出求解非对称 Toeplitz 线性方程组的递推算法.

2.3 扰动分析

在实际应用中, 所给的数据 (系数矩阵 A 和右端项 b) 往往是通过实验或观测等方式获得的, 因此通常是带有误差的, 这也导致最后求得的数值解也是有误差的. 本节就讨论原始数据误差对最后数值解的影响.

2.3.1 矩阵条件数

定义 2.1 考虑线性方程组 $Ax = b$, 如果 A 或 b 的微小变化会导致解的巨大变化, 则称此线性方程组是**病态**的, 反之则是**良态**的.

例 2.5 考虑线性方程组 $Ax = b$, 其中 $A = \begin{bmatrix} 1 & 1 \\ 1 & 1.0001 \end{bmatrix}$, $b = [2, 2]^T$, 可求得解为 $x = [2, 0]^T$. 如果 b 的第二个元素出现细微的偏差, 变为 $b = [2, 2.0001]^T$, 则解就变为 $x = [1, 1]^T$. 由此可见, 当右端项出现细微变化时, 解会出现很大的变化, 因此该线性方程组是病态的.

线性方程组是否病态主要取决于其系数矩阵. 怎样来判断一个矩阵是否病态? 目前比较常用的一个指标就是**矩阵条件数** (condition number).

定义 2.2 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, $\|\cdot\|$ 是任一算子范数, 则称

$$\kappa(A) \triangleq \|A^{-1}\| \|A\| \quad (2.17)$$

为 A 的**条件数**.

 如果 A 是奇异的, 或者不是方阵, 其条件数可以通过广义逆来定义.

常用的矩阵条件数

常用的矩阵条件数有

$$\kappa_2(A) \triangleq \|A^{-1}\|_2 \|A\|_2, \quad \kappa_1(A) \triangleq \|A^{-1}\|_1 \|A\|_1, \quad \kappa_\infty(A) \triangleq \|A^{-1}\|_\infty \|A\|_\infty.$$



谱条件数

$\kappa_2(A)$ 也称为**谱条件数**, 当 A 对称时, 有

$$\kappa_2(A) = \frac{\max_{1 \leq i \leq n} |\lambda_i|}{\min_{1 \leq i \leq n} |\lambda_i|}.$$

一般情况下, 如果没有特别指出, 则 $\kappa(A)$ 指的是矩阵 A 的谱条件数.

例 2.6 设 $A = \begin{bmatrix} 1 & 1 \\ 1 & 1.0001 \end{bmatrix}$, 则 $A^{-1} = \begin{bmatrix} 10001 & -10000 \\ -10000 & 10000 \end{bmatrix}$, 因此

$$\kappa_2(A) \approx 4.0002 \times 10^4, \quad \kappa_1(A) = \kappa_\infty \approx 4.0004 \times 10^4.$$

由此可见, A 是非常病态的.

条件数与病态

条件数是衡量一个矩阵是否病态的主要指标. 当矩阵条件数比较大时, 我们就称这个矩阵是**病态** (或者**坏条件**) 的. 由 (2.18) 可知, 如果矩阵是病态的, 则近似解的误差受数据扰动的影响就可能会很大.

 如果一个矩阵的某列的 2-范数明显远远小于其他列, 则该矩阵通常可能是病态的.

例 2.7 Hilbert 矩阵是一个典型的病态矩阵, 其定义如下:

$$H_n = [h_{ij}]_{n \times n}, \quad \text{其中 } h_{ij} = \frac{1}{i+j-1}.$$

可以验证 H_n 是对称正定的, 但随着 n 的增长, 其条件数会快速增长, 见下表:

表 2.1. Hilbert 矩阵的条件数

n	2	3	4	5	6	7	8	9	10
$\kappa_2(H_n)$	1.9e+01	5.2e+02	1.6e+04	4.8e+05	1.5e+07	4.8e+08	1.5e+10	4.9e+11	1.6e+13

2.3.2 δx 与 $\hat{x}$ 的关系

设 x_* 是精确解, $\hat{x}$ 是通过数值计算得到的近似解. 假定 $\hat{x}$ 满足线性方程组

$$(A + \delta A)\hat{x} = b + \delta b.$$

下面讨论 $\delta x \triangleq \hat{x} - x_*$ 的大小, 即**向后误差分析**.

定理 2.13 设 $\|\cdot\|$ 是任一向量范数 (当该范数作用在矩阵上时就是相应的导出范数), 则 δx



与 $\hat{x}$ 满足下面的关系式

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

当 $\delta b = 0$ 时, 有

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq \kappa(A) \frac{\|\delta A\|}{\|A\|}, \quad (2.18)$$

其中 $\kappa(A) \triangleq \|A^{-1}\| \|A\|$.

(板书)

 由 (2.18) 可知, 如果矩阵是病态的, 则近似解的误差受数据扰动的影响就可能会很大.

2.3.3 δx 与 x_* 的关系

引理 2.14 设 $\|\cdot\|$ 是任一算子范数, $B \in \mathbb{R}^{n \times n}$. 若 $\|B\| < 1$, 则 $I - B$ 可逆, 且有

$$(I - B)^{-1} = \sum_{k=0}^{\infty} B^k \quad \text{和} \quad \|(I - B)^{-1}\| \leq \frac{1}{1 - \|B\|}.$$

(板书)

由 $(A + \delta A)\hat{x} = b + \delta b$ 可得

$$\begin{aligned} \delta x &= (A + \delta A)^{-1}(b + \delta b - Ax_* - \delta Ax_*) \\ &= (I + A^{-1}\delta A)^{-1}A^{-1}(-\delta Ax_* + \delta b). \end{aligned}$$

假定 $\|\delta A\|$ 很小, 满足 $\|A^{-1}\delta A\| \leq \|A^{-1}\| \|\delta A\| < 1$, 则由引理 2.14 可得

$$\begin{aligned} \frac{\|\delta x\|}{\|x_*\|} &\leq \|(I + A^{-1}\delta A)^{-1}\| \|A^{-1}\| \left(\|\delta A\| + \frac{\|\delta b\|}{\|x_*\|} \right) \\ &\leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\| \|\delta A\|} \left(\|\delta A\| + \frac{\|\delta b\|}{\|x_*\|} \right) \\ &= \frac{\|A^{-1}\| \|A\|}{1 - \|A^{-1}\| \|A\| \frac{\|\delta A\|}{\|A\|}} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|A\| \|x_*\|} \right) \\ &\leq \frac{\kappa(A)}{1 - \kappa(A) \frac{\|\delta A\|}{\|A\|}} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right) \end{aligned}$$

当 $\|\delta A\| \rightarrow 0$ 时, 我们可得

$$\frac{\|\delta x\|}{\|x_*\|} \leq \frac{\kappa(A)}{1 - \kappa(A) \frac{\|\delta A\|}{\|A\|}} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right) \rightarrow \kappa(A) \frac{\|\delta b\|}{\|b\|}.$$

定理 2.15 设 $A \in \mathbb{R}^{n \times n}$ 非奇异且 $\|A^{-1}\| \|\delta A\| < 1$, 则

$$\frac{\|\delta x\|}{\|x_*\|} \leq \frac{\kappa(A)}{1 - \kappa(A) \frac{\|\delta A\|}{\|A\|}} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right). \quad (2.19)$$



如果 $\|\delta A\| = 0$, 则

$$\frac{1}{\kappa(A)} \frac{\|\delta b\|}{\|b\|} \leq \frac{\|\delta x\|}{\|x_*\|} \leq \kappa(A) \frac{\|\delta b\|}{\|b\|}. \quad (2.20)$$

(板书)

定理 2.16 (Eckart-Young [34]) 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, 则有

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

(板书)

 定理 2.16 的结论对所有 p -范数都成立, 参见 [43, 68].

 度量

$$\text{dist}_p(A) \triangleq \min \left\{ \frac{\|\delta A\|_p}{\|A\|_p} : A + \delta A \text{ 奇异} \right\} = \frac{1}{\kappa_p(A)},$$

表示 A 距离奇异矩阵集合的相对距离.

2.3.4 δx 与残量的关系

这是研究线性方程组的扰动理论的一个较实用的方法.

记残量 (残差) 为 $r = b - A\hat{x}$, 则有

$$\delta x = \hat{x} - x_* = \hat{x} - A^{-1}b = A^{-1}(A\hat{x} - b) = -A^{-1}r,$$

所以可得

$$\|\delta x\| \leq \|A^{-1}\| \|r\|.$$

这个估计式的优点是不用去估计 δA 和 δb 的大小. 由于在实际计算中, r 通常是可以计算的, 因此该估计式比较实用.

定理 2.17 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, $\|\cdot\|$ 为任一算子范数. 记 $r = b - A\hat{x}$, 则

- (1) 若存在 $\hat{A}$ 满足 $\hat{A}\hat{x} = b$, 则 $\|\hat{A} - A\| \geq \frac{\|r\|}{\|\hat{x}\|}$;
- (2) 存在 δA 满足 $\|\delta A\| = \frac{\|r\|}{\|\hat{x}\|}$, 使得 $(A + \delta A)\hat{x} = b$.

(板书)

2.3.5 相对扰动分析

前面给出了解的误差 δx 的界是与条件数 $\kappa(A)$, δA 和 δb 成比例的. 在许多情况下, 这个界是令人满意的. 但有时会相差很大, 这个界就不能很好的反映实际计算中解的误差.



例 2.8 设 $A = \begin{bmatrix} \gamma & 0 \\ 0 & 1 \end{bmatrix}$, $b = \begin{bmatrix} \gamma \\ 1 \end{bmatrix}$, 其中 $\gamma > 1$. 则 $Ax = b$ 的精确解为 $x_* = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, 任何合理的直接法求得的解的误差都很小. 但系数矩阵的谱条件数为 $\kappa_2(A) = \gamma$, 当 γ 很大时, $\kappa_2(A)$ 也很大, 因此误差界 (2.18) 和 (2.19) 可以是很大.

针对这个问题, 我们按分量进行分析. 记

$$\delta A = \begin{bmatrix} \delta a_{11} & \\ & \delta a_{22} \end{bmatrix}, \quad \delta b = \begin{bmatrix} \delta b_1 \\ \delta b_2 \end{bmatrix},$$

并设 $|\delta a_{ij}| \leq \varepsilon |a_{ij}|$, $|\delta b_i| \leq \varepsilon |b_i|$. 则

$$\delta x = \begin{bmatrix} \hat{x}_1 - x_1 \\ \hat{x}_2 - x_2 \end{bmatrix} = \begin{bmatrix} \frac{\delta b_1 + b_1}{\delta a_{11} + a_{11}} - 1 \\ \frac{\delta b_2 + b_2}{\delta a_{22} + a_{22}} - 1 \end{bmatrix} = \begin{bmatrix} \frac{\delta b_1 + \gamma}{\delta a_{11} + \gamma} - 1 \\ \frac{\delta b_2 + 1}{\delta a_{22} + 1} - 1 \end{bmatrix} = \begin{bmatrix} \frac{\delta b_1 - \delta a_{11}}{\delta a_{11} + \gamma} \\ \frac{\delta b_2 - \delta a_{22}}{\delta a_{22} + 1} \end{bmatrix}.$$

故

$$\|\delta x\|_\infty \leq \frac{2\varepsilon}{1-\varepsilon}.$$

如果 $\delta b = 0$, 则

$$\|\delta x\|_\infty \leq \frac{\varepsilon}{1-\varepsilon}.$$

这个界与 (2.18) 或 (2.19) 相差约 γ 倍.

相对条件数

为了得到更好误差界, 我们引入**相对条件数** $\kappa_{cr}(A)$, 即

$$\kappa_{cr}(A) \triangleq \| |A^{-1}| |A| \|,$$

有时也称为 Bauer 条件数或 Skeel 条件数.

假定 δA 和 δb 满足 $|\delta A| \leq \varepsilon |A|$ 和 $|\delta b| \leq \varepsilon |b|$. 则由 $(A + \delta A)\hat{x} = b + \delta b$ 可得

$$\begin{aligned} |\delta x| &= |A^{-1}(-\delta A\hat{x} + \delta b)| \\ &\leq |A^{-1}| (|\delta A| |\hat{x}| + |\delta b|) \\ &\leq |A^{-1}| (\varepsilon |A| |\hat{x}| + \varepsilon |b|) \\ &= \varepsilon |A^{-1}| (|A| |\hat{x}| + |b|). \end{aligned} \tag{2.21}$$

若 $\delta b = 0$, 则有

$$\|\delta x\| = \|\delta x\| \leq \varepsilon \| |A^{-1}| |A| |\hat{x}| \| \leq \varepsilon \| |A^{-1}| |A| \| \|\hat{x}\|,$$

即

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq \| |A^{-1}| |A| \| \varepsilon = \kappa_{cr}(A) \varepsilon. \tag{2.22}$$

相对条件数有下面的性质



引理 2.18 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, $D \in \mathbb{R}^{n \times n}$ 为非奇异对角矩阵, 则

$$\kappa_{cr}(DA) = \kappa_{cr}(A).$$

定理 2.19 设 $A \in \mathbb{R}^{n \times n}$ 非奇异. 使得 $|\delta A| \leq \varepsilon|A|$, $|\delta b| \leq \varepsilon|b|$ 成立, 且满足

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

的最小的 $\varepsilon > 0$ 称为**按分量的相对向后误差**, 其表达式为

$$\varepsilon = \max_{1 \leq i \leq n} \frac{|r_i|}{(|A||\hat{x}| + |b|)_i},$$

其中 $r = b - A\hat{x}$.

更多关于数值计算的稳定性和矩阵扰动分析方面的知识, 可以参考 [61, 101, 120].

2.4 误差分析

2.4.1 LU 分解的舍入误差分析

关于 LU 分解的舍入误差分析, 我们有下面的结果 [61].

定理 2.20 假定 $A \in \mathbb{R}^{n \times n}$ 的所有顺序主子式都不为 0, 则带舍入误差的 LU 分解可表示为

$$A = LU + E,$$

其中误差 E 满足

$$|E| \leq \gamma_n |L| \cdot |U|.$$

这里 $\gamma_n = \frac{n\varepsilon_u}{1 - n\varepsilon_u}$, ε_u 表示机器精度 (对于双精度浮点数, $\varepsilon_u = 2^{-53} \approx 1.1 \times 10^{-16}$; 对于单精度浮点数, $\varepsilon_u = 2^{-24} \approx 6.0 \times 10^{-8}$).

由于 $|L|$ 和 $|U|$ 可以是任意大, 比如

$$A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix} \begin{bmatrix} 10^{-20} & 1 \\ 0 & 1 - 10^{20} \end{bmatrix},$$

因此 LU 分解是不稳定的.

如果是部分选主元 LU 分解, 则有 $|l_{ij}| \leq 1$, 因此 $|E|$ 的上界主要取决于 $|U|$ 的大小.

2.4.2 Gauss 消去法的舍入误差分析

引理 2.21 [61] 设 $\hat{y}$ 和 $\hat{x}$ 是由回代算法 2.6 计算得到的数值解, 则

$$(L + \delta L)\hat{y} = b, \quad |\delta L| \leq \gamma_n |L|$$

$$(U + \delta U)\hat{x} = \hat{y}, \quad |\delta U| \leq \gamma_n |U|.$$



该引理表明, $\hat{y}$ 和 $\hat{x}$ 只有很小的误差, 因此向前回代算法和向后回代算法都是稳定的. 于是

$$\begin{aligned} b &= (L + \delta L)\hat{y} \\ &= (L + \delta L)(U + \delta U)\hat{x} \\ &= (LU + L \cdot \delta U + \delta L \cdot U + \delta L \cdot \delta U)\hat{x} \\ &= (A - E + L \cdot \delta U + \delta L \cdot U + \delta L \cdot \delta U)\hat{x}. \end{aligned}$$

记 $\delta A = -E + L \cdot \delta U + \delta L \cdot U + \delta L \cdot \delta U$, 则 $\hat{x}$ 是扰动方程 $(A + \delta A)x = b$ 精确解, 且

$$\begin{aligned} |\delta A| &= |-E + L \cdot \delta U + \delta L \cdot U + \delta L \cdot \delta U| \\ &\leq |E| + |L| \cdot |\delta U| + |\delta L| \cdot |U| + |\delta L| \cdot |\delta U| \\ &\leq (3\gamma_n + \gamma_n^2)|L| \cdot |U| \leq \gamma_{3n}|L| \cdot |U|, \end{aligned}$$

其中 $\gamma_{3n} = \frac{3n\varepsilon_u}{1 - 3n\varepsilon_u}$. 两边取范数后可得

$$\|\delta A\| \leq 3n\varepsilon_u \|L\| \cdot \|U\|$$

对 1-范数, ∞ -范数和 F -范数成立 (2-范数不成立).

根据算法向后稳定性的定义, 要说明带选主元 Gauss 消去法是向后稳定的, 必须要求 $\|\delta A\|$ 是“小”的, 即

$$\|\delta A\| = \mathcal{O}(\varepsilon_u)\|A\|.$$

数值试验表明, 部分选主元 Gauss 消去法几乎总是保持

$$\|L\| \cdot \|U\| \approx \|A\|.$$

记

$$\rho_n \triangleq \frac{\max_{i,j,k} |a_{ij}^{(k)}|}{\max_{i,j} |a_{ij}|}$$

为部分选主元 Gauss 消去法的**增长因子**, 其中 $a_{ij}^{(k)}$ 是部分选主元 Gauss 消去法过程第 k 步时 a_{ij} 的值. 由于 $\|L\|_\infty \leq n$, $\|U\|_\infty \leq n\rho_n\|A\|_\infty$, 因此 [61]

定理 2.22 设 $\hat{x}$ 是由部分选主元 Gauss 法得到的数值解, 则 $\hat{x}$ 满足

$$(A + \delta A)\hat{x} = b, \quad \|\delta A\|_\infty \leq n^2\gamma_{3n}\rho_n\|A\|_\infty. \quad (2.23)$$

所以若 ρ_n 比较小或随着 n 变大时增长比较缓慢, 则当 n 不是很大时, 部分选主元 Gauss 消去法是向后稳定的. 遗憾的是, 理论上无法保证 ρ_n 比较小 [61, page 166].

定理 2.23 部分选主元 Gauss 消去法能保证 $\rho_n \leq 2^{n-1}$, 且这个界是可以达到的.

事实上, (2.23) 中的界几乎总是远远大于真正的 $\|\delta A\|$.

在绝大多数情况下, 部分选主元 Gauss 消去法是向后稳定的, 但理论上也存在失败的例子.

全主元 Gauss 消去法是数值稳定的. 在大部分实际应用中, 部分选主元 Gauss 消去法与全



主元 Gauss 消去法具有同样的数值稳定性.

2.5 解的改进

当矩阵 A 是病态时, 即使残量 $r = b - A\hat{x}$ 很小, 所求得的数值解 $\hat{x}$ 仍可能带有较大的误差. 此时需要通过一些方法来提高解的精度.

2.5.1 高精度运算

在计算中, 尽可能采用高精度的运算. 比如, 原始数据是单精度的, 但在计算时都采用双精度运算, 或者更高精度的运算. 但更高精度的运算会带来更大的开销.

2.5.2 矩阵元素缩放或平衡 (Scaling or equilibration)

在求解线性方程组时, 先对系数矩阵元素进行适当的缩放是一种很常用的技术手段.

一方面, 如果 A 的元素在数量级上相差很大, 则在计算过程中很可能会出现大数与小数的加减运算, 这样就可能会引入更多的舍入误差. 为了避免由于这种情况而导致的舍入误差, 我们可以在求解之前先对矩阵元素进行缩放 (Scaling), 即在矩阵两边同时乘以两个适当的对角矩阵. 在对矩阵进行缩放时, 需要 $\mathcal{O}(n^2)$ 运算量, 通常不会产生较大的舍入误差.

另一方面, 由前面的扰动分析可知, 矩阵条件数对数值计算的误差有着很大的影响, 是衡量问题是否病态的一个重要指标. 在实际计算中, 如果遇到病态问题, 我们希望能够通过一些简单的方法降低其条件数. 其中一类简单有效的方法就是在矩阵两边同时乘以一个对角矩阵, 即寻找对角矩阵 D_r 和 D_c , 使得 $D_r^{-1}AD_c^{-1}$ 的条件数达到最小 (或者尽可能小). 显然, 寻找这样的对角矩阵是非常困难的, 目前仍然是一个开放问题 [98].

一种比较可行的实用方案是使得缩放后的矩阵的每行或每列具有相同的 p -范数. 比如取 $D_r = \text{diag}(d_1, d_2, \dots, d_n)$, 其中 $d_i = \sum_{j=1}^n |a_{ij}|$, 这样 $D_r^{-1}A$ 的每行的 1-范数都一样, 但没法使得所有列也具有相同的范数.

如果矩阵

$$D_r^{-1}AD_c^{-1}$$

的所有行和所有列都具有相同 (或近似相同) 的范数, 则称为 A 的**均衡化** (equilibration) [98]. 有学者提出了一种迭代方法对 A 进行均衡化 [71]. 更多信息可参见相关资料, 如 [30].

2.5.3 迭代改进法

设近似解 $\hat{x}$, 残量 $r = b - A\hat{x}$. 当 $\hat{x}$ 没达到精度要求时, 可以考虑方程组 $Az = r$. 设 z 是该方程组的精确解, 则

$$A(\hat{x} + z) = A\hat{x} + Az = (b - r) + r = b,$$

因此 $\hat{x} + z$ 就是原方程组的精确解.



在实际计算中, 我们可能得到的是近似解 $\hat{z}$, 但通常 $\|r - A\hat{z}\|$ 应该比较小, 特别地, 比 $\|r\|$ 更小. 因此 $\hat{x} + \hat{z}$ 应该比 $\hat{x}$ 更接近精确解.

如果新的近似解 $\hat{x} + \hat{z}$ 仍不满足精度要求, 则可重复以上过程.

这就是通过迭代来提高解的精度.

算法 2.15 通过迭代改进解的精度

- 1: 设 $PA = LU$, $\hat{x}$ 是 $Ax = b$ 的近似解
 - 2: **while** 近似解 $\hat{x}$ 不满足精度要求, **do**
 - 3: 计算 $r = b - A\hat{x}$
 - 4: 求解 $Ly = Pr$, 即 $y = L^{-1}Pr$
 - 5: 求解 $Uz = y$, 即 $z = U^{-1}y$
 - 6: 令 $\hat{x} = \hat{x} + z$
 - 7: **end while**
-

由于每次迭代只需计算一次残量和求解两个三角线性方程组, 因此运算量为 $O(n^2)$. 所以相对来讲还是比较经济的.

-  为了提高计算精度, 在计算残量 r 时最好使用原始数据 A , 而不是 $P^T LU$, 因此对 A 做 PLU 分解时需要保留矩阵 A , 不能被 L 和 U 覆盖.
-  由于在计算残量时, 可能会存在大量的相近的数相减运算, 因此可以采用更高精度的运算.
-  实际计算经验表明, 当 A 病态不是很严重时, 即 $\varepsilon_u \kappa_\infty(A) < 1$, 迭代法可以有效改进解的精度, 最后达到机器精度. 但 $\varepsilon_u \kappa_\infty(A) \geq 1$ 时, 一般没什么效果. 这里 ε_u 表示机器精度.

2.6 课后习题

练习 2.1 设 $A = [a_{ij}] \in \mathbb{R}^{n \times n}$, 且 $a_{11} \neq 0$, 经过第一步 LU 分解后得到 $A^{(2)} = \begin{bmatrix} a_{11} & * \\ 0 & A_{22} \end{bmatrix}$.

- 证明: (1) 若 A 对称, 则 A_{22} 也对称;
 (2) 若 A 对称正定, 则 A_{22} 也对称正定;
 (3) 若 A 严格行对角占优, 则 A_{22} 也是严格行对角占优.

练习 2.2 (定理 2.3) 设 $A \in \mathbb{R}^{n \times n}$ 非奇异且列对角占优. 证明: A 存在 LU 分解且 L 中的元素的绝对值都不超过 1.

练习 2.3 设 $A \in \mathbb{R}^{n \times n}$ 非奇异. 证明: 存在置换矩阵 P , 使得 PA 的所有顺序主子矩阵都非奇异.

练习 2.4 设矩阵 $A = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 12 \end{bmatrix}$, 计算 A 的 LU 分解和 PLU 分解.

练习 2.5 计算 $A = \begin{bmatrix} 4 & 2 & 4 \\ 2 & 37 & 8 \\ 4 & 8 & 14 \end{bmatrix}$ 的 Cholesky 分解, 并求解 $Ax = b$, 其中 $b = \begin{bmatrix} 6 \\ -9 \\ 7 \end{bmatrix}$.



练习 2.6 设矩阵 $A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & a \\ 0 & a & 2 \end{bmatrix}$, 问: 当 a 取何值时, A 存在 Cholesky 分解?

练习 2.7 证明 Cholesky 分解的唯一性.

练习 2.8 设 $A \in \mathbb{R}^{n \times n}$ 对称非奇异, 且存在分解 $A = LDM^T$, 其中 $L, M \in \mathbb{R}^{n \times n}$ 是单位下三角矩阵, $D \in \mathbb{R}^{n \times n}$ 是对角矩阵. 证明: $L = M$.

练习 2.9 设 A 对称且非零, 试证明: 存在置换矩阵 P 使得

$$PAP^T = \begin{bmatrix} B & E^T \\ E & C \end{bmatrix},$$

其中 $B \in \mathbb{R}$ 或 $B \in \mathbb{R}^{2 \times 2}$, 且非奇异.

练习 2.10 设 $\lambda \neq 0$, 矩阵 $A = \begin{bmatrix} \lambda & 2\lambda \\ 1 & 1 \end{bmatrix}$, 当 λ 取何值时, $\kappa_\infty(A)$ 达到最小.

练习 2.11 设 $A \in \mathbb{R}^{m \times n}$, 其中 $m \geq n$, 证明: $\|A^T A\|_2 = \|A\|_2^2$.

当 $m = n$ 时, 证明: $\kappa_2(A^T A) = (\kappa_2(A))^2$.

..... 以下为可选题

练习 2.12 设 $A = [a_{ij}] \in \mathbb{R}^{n \times n}$ 对称正定, 证明: $a_{ij}^2 < a_{ii}a_{jj}$.

练习 2.13* (定理 1.56) 设 $A \in \mathbb{R}^{n \times n}$ 严格列对角占优, 记

$$\delta \triangleq \min_{1 \leq j \leq n} \left(|a_{jj}| - \sum_{i=1, i \neq j}^n |a_{ij}| \right).$$

证明: $\|A^{-1}\|_1 \leq \delta^{-1}$.

练习 2.14 (验证等式 (2.3)) 证明:

$$L = L_1 L_2 \cdots L_{n-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{bmatrix}.$$

练习 2.15 将 $A \in \mathbb{R}^{n \times n}$ 写成分块形式

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

其中 $A_{11} \in \mathbb{R}^{k \times k}$ ($1 \leq k \leq n$) 非奇异. 我们称矩阵 $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ 为 A 中 A_{11} 的 Schur 补 (通常简称 **Schur 补**).

(1) 假设 A 存在 LU 分解, 证明: 对于不选主元的 Gauss 消去法, 第 k 步后, A_{22} 已被 S 覆盖.

(2) 假设 $A_{21} = A_{12}^T$, 且 A_{11} 和 $-A_{22}$ 都正定, 证明 A 非奇异.



练习 2.16* 假定已知 A 的 LU 分解: $A = LU$, 试设计算法计算 A^{-1} 的第 (i, j) 个元素.

练习 2.17* 设 $A, B \in \mathbb{R}^{n \times n}$ 是两个上三角矩阵, $\alpha \in \mathbb{R}$ 是给定常数, 且 $AB - \alpha I$ 非奇异. 试设计求解 $(AB - \alpha I)x = f$ 的算法, 使得运算量为 $\mathcal{O}(n^2)$.

..... 以下为实践题

练习 2.18 设 $L \in \mathbb{R}^{n \times n}$ 是非奇异下三角矩阵, $B \in \mathbb{R}^{m \times n}$, 统计以下运算的计算量 (加减乘除, 只需给出最高次项):

(1) 计算 L^{-1} ; (2) 计算 L^2 ; (3) 计算 LL^T ; (4) 计算 BL .

练习 2.19 写出按列存储方式下三角方程组 $Ly = b$ 的求解算法, 并编写相应的 MATLAB 程序.

练习 2.20 利用列主元 LU 分解, 给出计算矩阵的逆的实用算法, 并编写相应的 MATLAB 程序.

练习 2.21 根据算法 2.11, 编写求解对角占优三对角线性方程组的追赶法 MATLAB 程序.

练习 2.22* 考虑线性方程组 $Ax = f$, 其中:

$$A = \begin{bmatrix} b_1 & c_1 & & \alpha \\ a_1 & \ddots & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ \beta & & a_{n-1} & b_n \end{bmatrix}, \quad f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix},$$

即 A 是三对角矩阵另加两个非零元素 (分别位于东北角和西南角). 设计求解该线性方程组的直接算法, 并编写 MATLAB 程序.

练习 2.23 设 $A \in \mathbb{R}^{n \times n}$ 是非奇异三对角矩阵, 给出求解 $Ax = b$ 的列主元 LU 分解算法, 并编写相应的 MATLAB 程序. 要求: $PA = LU$ 中置换矩阵 P 用向量表示, L 和 U 只存储非零元素. (用两个向量 $L_a \in \mathbb{R}^n$ 和 $L_I \in \mathbb{Z}^n$ 表示 L , 其中 L_a 存储非零元素的值, L_I 存储这些非零元素的行标, 即 $L_a(j)$ 表示 $l_{L_I(j), j}$; 用 $U_a \in \mathbb{R}^{n \times 3}$ 表示 U , 其中第一列表示 U 的主对角线元素, 第二列表示 U 的次对角线元素, 第三列表示 U 的第二条次对角线元素, 即 $U_a(i, j)$ 表示 $u_{i, i+j-1}$.)

练习 2.24 带状矩阵的 LU 分解. 设 A 是 n 阶带状矩阵, 上带宽为 $L < n$, 下带宽为 $M < n$, 编写一个函数, 计算 A 的 LU 分解 (不带选主元), 并统计运算量.

练习 2.25 设计按行计算 L 的 Cholesky 分解算法, 并编写相应的 MATLAB 程序.

练习 2.26 设对称正定矩阵 $A = \begin{bmatrix} A_{11} & a \\ a^T & \beta \end{bmatrix} \in \mathbb{R}^{(n+1) \times (n+1)}$, 其中 $A_{11} \in \mathbb{R}^{n \times n}$, $a \in \mathbb{R}^n$, $\beta \in \mathbb{R}$.

已知 A_{11} 的 Cholesky 分解为 $A_{11} = L_1^T L_1$, 试给出 A 的 Cholesky 分解, 并由此设计计算 Cholesky 分解的递归算法.



第三讲 线性最小二乘问题

最小二乘问题 包括线性最小二乘问题, 总体最小二乘问题, 约束最小二乘问题, 非线性最小二乘问题等, 在人工智能、数据科学、统计学、材料与结构力学、信号与图像处理等方面都有着广泛的应用, 是计算数学和最优化的一个重要研究分支, 也是一个活跃的研究领域.

本讲主要介绍求解**线性最小二乘问题** (Linear Least Squares) 的三种常用方法: 正规方程法 (也称法方程法), QR 分解法和 SVD 分解法. 一般来说, 正规方程法是最快的, 特别是当 A 的条件数较小时, 正规方程法几乎与其他方法一样精确. 而 SVD 分解法是最慢的, 但结果最可靠.

关于最小二乘问题的相关参考资料

- ▶ Åke Björck, *Numerical Methods for Least Squares Problems*, 1996 [3] (2nd Edition, 2024)
- ▶ 魏木生, 李莹, 赵建立, 广义最小二乘问题的理论与计算 (第二版), 2020 [130]

3.1 问题介绍

考虑 **线性最小二乘问题**

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2, \quad (3.1)$$

其中 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. 问题 (3.1) 的解称为 **最小二乘解**.

- 当 $m = n$ 且 A 非奇异时, 这就是一个线性方程组, 解为 $x_* = A^{-1}b$;
- 当 $m > n$ 时, 约束个数大于未知量个数, 此时我们称问题 (3.1) 为**超定**的;
- 当 $m < n$ 时, 未知量个数大于约束个数, 此时我们称问题 (3.1) 为**欠定** (或亚定) 的.

🔴 为了方便起见, 我们有时将线性最小二乘问题 (3.1) 简称为**最小二乘问题**.

🔴 为了讨论方便, 本讲假定 A 是满秩的.

3.1.1 超定方程组

当 $m > n$ 时, 线性方程组 $Ax = b$ 的解可能不存在. 此时一般考虑求解最小二乘问题 (3.1). 记

$$J(x) \triangleq \frac{1}{2} \|Ax - b\|_2^2.$$

易知 $J(x)$ 是关于 x 的二次函数, 而且是凸函数 (当 A 满秩时, $J(x)$ 的 Hessian 阵是正定的). 因此, 由凸函数的性质可知, x_* 是问题 (3.1) 的解当且仅当 x_* 是 $J(x)$ 的稳定点. 对 $J(x)$ 求一阶导数, 令其为零可得

$$J'(x) = A^T Ax - A^T b = 0.$$

于是将最小二乘问题转化为一个线性方程组, 这就是正规方程.

 如果 A 不是满秩, 则 $A^T A$ 半正定, 此时解不唯一.

3.1.2 欠定方程组

若 $m < n$, 则线性方程组 $Ax = b$ 存在无穷多个解 (假定 A 满秩). 这时我们通常寻求最小范数解, 即所有解中范数最小的解. 于是原问题就转化为下面的**约束优化问题**

$$\min_{Ax=b} \frac{1}{2} \|x\|_2^2 \quad (3.2)$$

对应的 **Lagrange 函数**为

$$\mathcal{L}(x, \lambda) = \frac{1}{2} \|x\|_2^2 + \lambda^T (Ax - b),$$

其中 $\lambda = [\lambda_1, \lambda_2, \dots, \lambda_m]^T$ 是 **Lagrange 乘子**. 此时优化问题 (3.2) 的解就是 $\mathcal{L}(x, \lambda)$ 的**鞍点**, 即下面方程组的解:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial x} = x + A^T \lambda = 0, \\ \frac{\partial \mathcal{L}}{\partial \lambda} = Ax - b = 0. \end{cases}$$

写成矩阵形式为

$$\begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

如果 A 满秩, 即 $\text{rank}(A) = m$, 则系数矩阵非奇异 (见练习 3.1), 上述方程组存在唯一解.

本讲主要讨论超定线性最小二乘问题的求解.

3.2 几类重要的矩阵变换

矩阵计算的一个基本思想就是把复杂的问题转化为等价的且易于求解的问题. 完成这个转化的基本工具就是**矩阵变换**, 除了高等代数 (或线性代数) 中提到的三类初等变换以外, 在矩阵计算中常用的矩阵变换还有: Gauss 变换, Householder 变换和 Givens 变换, 其中 Gauss 变换主要用于矩阵的 LU 分解, Householder 变换和 Givens 变换是正交变换, 可用于计算线性最小二乘问题、矩阵特征值和奇异值问题等.

3.2.1 初等矩阵变换

本科高等代数教材中介绍了三类初等矩阵变换, 这里将其推广到更一般的情形. 我们称

$$E(u, v, \tau) = I - \tau uv^* \quad (3.3)$$

为**初等矩阵变换**或**初等矩阵**, 其中 $u, v \in \mathbb{C}^n$ 是非零向量, τ 是一个非零复数. 因此, 初等矩阵是单位矩阵的一个秩 1 修正.



 高等代数中介绍的三类初等变换 (elementary transformation) 分别是 (以行变换为例): (1) 交换两行; (2) 某行乘以一个非零常数; (3) 某行乘以一个常数后加到另外一行. 这三类变换所对应的矩阵为 (简单示例)

$$E_1 = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 0 & & 1 \\ & & & \ddots & \\ & 1 & & & 0 \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}, E_2 = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & c & \\ & & & & 1 \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}, E_3 = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & c & \\ & & & & 1 \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}.$$

可以验证, 它们都可以表示为 (3.3) 形式, 留作练习.

下面的定理给出了初等矩阵的基本性质.

定理 3.1 设 $E(u, v, \tau)$ 是一个初等矩阵, 我们有

(1) $\det(E(u, v, \tau)) = 1 - \tau v^* u$;

(2) 若 $1 - \tau v^* u \neq 0$, 则 $E(u, v, \tau)$ 非奇异, 且

$$(E(u, v, \tau))^{-1} = E(u, v, \gamma), \quad \text{其中 } \gamma = \frac{\tau}{\tau v^* u - 1}.$$

(3) 对任意非零向量 $x, y \in \mathbb{C}^n$, 存在 $u, v \in \mathbb{C}^n$ 和 $\tau \in \mathbb{C}$, 使得

$$E(u, v, \tau)x = y.$$

(板书)

更一般地, 我们有下面的结论.

Sylvester 降幂公式

设 $A \in \mathbb{C}^{m \times n}$, $B \in \mathbb{C}^{n \times m}$, 其中 $m \geq n$, 则有

$$\det(\lambda I - AB) = \lambda^{m-n} \det(\lambda I - BA). \quad (3.4)$$

因此 AB 和 BA 具有相同的非零特征值.

(留作练习)

定理 3.2 设 $A \in \mathbb{C}^{n \times n}$, 则 A 非奇异当且仅当 A 可以分解成若干个初等矩阵的乘积.

3.2.2 Gauss 变换

设 $l_j = [0, \dots, 0, l_{j+1,j}, \dots, l_{n,j}]^T, j = 1, 2, \dots, n$, 则 **Gauss 变换** 定义为

$$L(l_j) \triangleq E(l_j, e_j, -1) = I + l_j e_j^T = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{j+1,j} & 1 & \\ & & \vdots & & \ddots \\ & & l_{n,j} & & & 1 \end{bmatrix}.$$



显然, Gauss 变换也属于初等矩阵变换. 向量 l_j 称为 **Gauss 向量** [49]. 由定理 3.1 可知

$$\det(L(l_j)) = 1, \quad (L(l_j))^{-1} = E(l_j, e_j, 1) = E(-l_j, e_j, -1) = L(-l_j).$$

🔪 Gauss 变换是对单位矩阵的秩 1 修正, 主要功能是清零, 即将指定的某些元素变为零.

🔪 Gauss 变换有时也称为**初等下三角矩阵**.

3.2.3 Householder 变换

定义 3.1 我们称矩阵

$$H = I - \beta vv^*, \quad 0 \neq v \in \mathbb{C}^n, \quad \beta = \frac{2}{v^*v} \quad (3.5)$$

为 **Householder 矩阵** 或 **Householder 变换**, 向量 v 称为 **Householder 向量**. 我们通常将矩阵 (3.5) 记为 $H(v)$.

🔪 Householder 矩阵也是初等矩阵.

🔪 有时也将 Householder 变换定义为

$$H = I - 2vv^*, \quad v \in \mathbb{C}^n \text{ 且 } \|v\|_2 = 1.$$

🔪 Householder 矩阵有时也称为**初等 Hermite 矩阵** [123].

从几何上看, 一个 Householder 变换是一个关于超平面 $\text{span}\{v\}^\perp$ 的反射. 由于 $\mathbb{C}^n = \text{span}\{v\} \oplus \text{span}\{v\}^\perp$, 因此对任意向量 $x \in \mathbb{C}^n$, 都可写为

$$x = x_v + x_{v^\perp}, \quad \text{其中 } x_v \in \text{span}\{v\}, \quad x_{v^\perp} \in \text{span}\{v\}^\perp.$$

易知 $x_v = \frac{v^*x}{v^*v}v$. 于是

$$Hx = x - \beta vv^*x = x - \frac{2v^*x}{v^*v}v = -\frac{v^*x}{v^*v}v + x_{v^\perp} = -x_v + x_{v^\perp},$$

即 Hx 与 x 在 $\text{span}\{v\}^\perp$ 方向有着相同的分量, 而在 v 方向的分量正好相差一个符号. 也就是说, Hx 是 x 关于超平面 $\text{span}\{v\}^\perp$ 的镜面反射, 见图 3.1. 因此, Householder 变换也称为**Householder 反射**或**反射变换**.

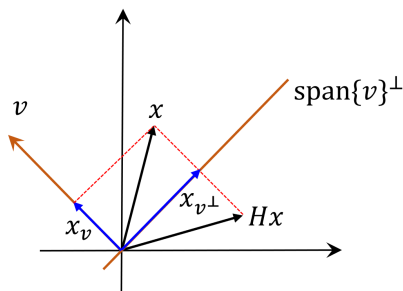


图 3.1. Householder 变换的几何意义



下面是关于 Householder 矩阵的几个基本性质.

定理 3.3 设 $H \in \mathbb{C}^{n \times n}$ 是一个 Householder 矩阵, 则

- (1) $H^* = H$, 即 H 是 Hermite 的;
- (2) $H^*H = I$, 即 H 是酉矩阵;
- (3) $H^2 = I$, 所以 $H^{-1} = H$;
- (4) $\det(H) = -1$;
- (5) H 有两个互异的特征值: $\lambda = 1$ 和 $\lambda = -1$, 其中 $\lambda = 1$ 的代数重数为 $n - 1$.

Householder 矩阵的一个非常重要的应用就是可以将一个向量除第一个元素以外的所有元素都化为零. 我们首先给出一个引理.

引理 3.4 设 $x, y \in \mathbb{C}^n$ 为任意两个互异的向量, 则存在一个 Householder 矩阵 $H(v)$ 使得 $y = H(v)x$ 的充要条件是 $\|x\|_2 = \|y\|_2$ 且 $x^*y \in \mathbb{R}$. (板书)

 Householder 向量也可以取 $v = e^{i\theta}(x - y)$, $0 \leq \theta < 2\pi$.

 如果 x, y 都是实向量, 则条件 $x^*y \in \mathbb{R}$ 自然成立, 此时充要条件就是 $\|x\|_2 = \|y\|_2$.

由引理 3.4, 我们可以立即得到下面的结论.

定理 3.5 设 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$ 是一个非零向量, 则存在 Householder 矩阵 $H(v)$ 使得 $H(v)x = \alpha e_1$, 其中 $\alpha = \|x\|_2$ (或 $\alpha = -\|x\|_2$), $e_1 = [1, 0, \dots, 0]^T \in \mathbb{R}^n$.

 在后面的讨论中, 我们将定理中的向量 v 称为 x 对应的 Householder 向量.

设 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$ 是一个实的非零向量, 下面讨论如何计算定理 3.5 中 Householder 矩阵 $H(v)$ 所对应的 Householder 向量 v . 由引理 3.4 的证明过程可知

$$v = x - \alpha e_1 = [x_1 - \alpha, x_2, \dots, x_n]^T.$$

在实际计算中, 为了尽可能地减少舍入误差, 我们通常避免两个相近的数做减运算, 否则就会损失有效数字. 因此, 我通常取

$$\alpha = -\operatorname{sign}(x_1) \cdot \|x\|_2. \quad (3.6)$$

事实上, 我们也可以取 $\alpha = \operatorname{sign}(x_1)\|x\|_2$, 但此时为了减少舍入误差, 我们需要通过下面的公式来计算 v 的第一个分量 v_1

$$\alpha = \operatorname{sign}(x_1)\|x\|_2, \quad v_1 = x_1 - \alpha = \frac{x_1^2 - \|x\|_2^2}{x_1 + \alpha} = \frac{-(x_2^2 + x_3^2 + \dots + x_n^2)}{x_1 + \alpha}. \quad (3.7)$$



在 v_1 的两种计算方法 (3.6) 和 (3.7) 中, α 的取值都与 x_1 的符号有关. 但在某些应用中, 我们需要确保 α 非负, 此时我们可以将这两种方法结合起来使用, 即:

$$v_1 = \begin{cases} x_1 - \alpha, & \text{if } \text{sign}(x_1) < 0 \\ \frac{-(x_2^2 + x_3^2 + \cdots + x_n^2)}{x_1 + \alpha}, & \text{otherwise} \end{cases}$$

无论怎样选取 α , 我们都有 $H = I - \beta vv^*$, 其中

$$\beta = \frac{2}{v^*v} = \frac{2}{(x_1 - \alpha)^2 + x_2^2 + \cdots + x_n^2} = \frac{2}{2\alpha^2 - 2\alpha x_1} = -\frac{1}{\alpha v_1}.$$

 **思考:** 如果 x 是复向量, 有没有相应的结论?

在实数域中计算 Householder 向量 v 的算法如下, 总运算量大约为 $2n$, 如果加上赋值运算的话则为 $3n$.

算法 3.1 计算 Householder 向量

```
% Given  $x \in \mathbb{R}^n$ , compute  $v \in \mathbb{R}^n$  such that  $Hx = \|x\|_2 e_1$ , where  $H = I - \beta vv^*$ 
1: function  $[\beta, v] = \text{House}(x)$ 
2:  $n = \text{length}(x)$  (here  $\text{length}(x)$  denotes the dimension of  $x$ )
3: compute  $\sigma = x_2^2 + x_3^2 + \cdots + x_n^2$  and let  $v = x$ 
4: if  $\sigma = 0$  and  $x_1 \geq 0$  then
5:    $\beta = 0$    %  $v_1 = 0$ 
6: else if  $\sigma = 0$  and  $x_1 < 0$  then
7:    $v_1 = 2x_1, \beta = 2/v_1^2$ 
8: else
9:    $\alpha = \sqrt{x_1^2 + \sigma}$    %  $\alpha = \|x\|_2$ 
10:  if  $x_1 \leq 0$  then
11:     $v_1 = x_1 - \alpha$ 
12:  else
13:     $v_1 = -\sigma/(x_1 + \alpha)$ 
14:  end if
15:   $\beta = 2/(v_1^2 + \sigma)$ 
16: end if
```

可以证明, 上述算法具有很好的数值稳定性 [115], 即

$$\|\tilde{H} - H\|_2 = \mathcal{O}(\varepsilon_u),$$

其中 $\tilde{H}$ 是由上述算法计算得到的近似矩阵, ε_u 是机器精度.



在实际计算时, 我们可以将向量 v 单位化, 使得 $v_1 = 1$. 这样, 我们就无需为 v 另外分配空间, 而是将 $v(2:n)$ 存放在 $x(2:n)$ 中, 因为经过 Householder 变换后, 向量 x 除第一个分量外, 其它都为零.

思考: 这里要求 $v_1 \neq 0$, 那么什么情况下 $v_1 = 0$?

为了避免可能产生的溢出, 我们也可以事先将 x 单位化, 即令 $x = x/\|x\|_2$.

Householder 变换与矩阵的乘积

设 $A \in \mathbb{R}^{m \times n}$, $H = I - \beta vv^* \in \mathbb{R}^{m \times m}$, 则

$$HA = (I - \beta vv^*)A = A - \beta v(v^*A).$$

因此, 在做 Householder 变换时, 并不需要生成 Householder 矩阵, 只需要 Householder 向量即可. 上面矩阵相乘的总运算量大约为 $4mn$.

注记

- Householder 矩阵早在 1932 年就出现了 [105], 但直到 1958 年才由 Householder 用于矩阵的三角化 [64].
- 关于其他类型的 Householder 变换 (包括复数情形), 可以参见 [29].

3.2.4 Givens 变换

为简单起见, 我们这里讨论实数域中的 Givens 变换. 设 $\theta \in [0, 2\pi]$, 我们称矩阵

$$G(i, j, \theta) = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & c & \cdots & s \\ & & \vdots & \ddots & \vdots \\ & & -s & \cdots & c \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad (i \leq j)$$

为 **Givens 变换** (或 **Givens 旋转**, 或 **Givens 矩阵**), 其中 $c = \cos(\theta)$, $s = \sin(\theta)$. 也就是说, 将单位矩阵的 (i, i) 和 (j, j) 位置上的元素用 c 代替, 而 (i, j) 和 (j, i) 位置上的元素分别用 s 和 $-s$ 代替, 所得到的矩阵就是 $G(i, j, \theta)$.

定理 3.6 $G(i, j, \theta)$ 是正交矩阵, 且 $\det(G(i, j, \theta)) = 1$.

Givens 变换不是初等矩阵变换, 事实上, 它是单位矩阵的一个秩 2 修正, 即

$$G(i, j, \theta) = I + [e_i, e_j] \begin{bmatrix} c-1 & s \\ -s & c-1 \end{bmatrix} [e_i, e_j]^T.$$

如果是定义在复数域上的 Givens 变换, 则 $c = e^{i\alpha} \cos \theta$, $s = e^{i\beta} \sin \theta$, $0 \leq \alpha, \beta, \theta < 2\pi$.



例 3.1 设 $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$, 则存在一个 Givens 变换 $G = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \in \mathbb{R}^{2 \times 2}$ 使得 $Gx = \begin{bmatrix} r \\ 0 \end{bmatrix}$, 其中 c, s 和 r 的值如下:

$$c = \frac{x_1}{r}, \quad s = \frac{x_2}{r}, \quad r = \sqrt{x_1^2 + x_2^2}, \quad \text{即} \quad G = \frac{1}{r} \begin{bmatrix} x_1 & x_2 \\ -x_2 & x_1 \end{bmatrix}.$$

也就是说, 通过 Givens 变换, 我们可以将向量 $x \in \mathbb{R}^2$ 的第二个分量化为 0.

算法 3.2 Givens 变换

% Given $x = [x_1, x_2]^T \in \mathbb{R}^2$, compute c and s such that $Gx = [r, 0]^T$ where $r = \|x\|_2$

1: **function** $[c, s] = \text{Givens}(x_1, x_2)$

2: **if** $x_2 = 0$ **then**

3: $c = \text{sign}(x_1), \quad s = 0$

4: **else**

5: **if** $|x_2| > |x_1|$ **then** % 为了算法稳定性, 应尽量避免大数除以小数

6: $\tau = \frac{x_1}{x_2}, \quad s = \frac{\text{sign}(x_2)}{\sqrt{1 + \tau^2}}, \quad c = s\tau$

7: **else**

8: $\tau = \frac{x_2}{x_1}, \quad c = \frac{\text{sign}(x_1)}{\sqrt{1 + \tau^2}}, \quad s = c\tau$

9: **end if**

10: **end if**

 计算 s 和 c 时, 也可以用 $s = \frac{1}{\sqrt{1 + \tau^2}}$ 和 $c = \frac{\tau}{\sqrt{1 + \tau^2}}$, 此时 r 可能等于 $-\|x\|_2$.

事实上, 对任意一个向量 $x \in \mathbb{R}^n$, 都可以通过 Givens 变换将其任意一个位置上的分量化为 0. 更进一步, 我们也可以通过若干个 Givens 变换, 将 x 中除第一个分量外的所有元素都化为 0.

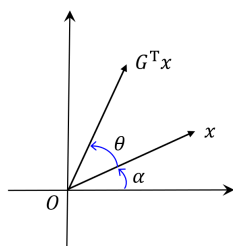
例 3.2 (Givens 变换的几何意义)

设 $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} r \cos \alpha \\ r \sin \alpha \end{bmatrix} \in \mathbb{R}^2$, 其中 $r = \sqrt{x_1^2 + x_2^2}$, $\alpha = \arctan \frac{x_2}{x_1}$, 则

$$G^T x = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix}^T \begin{bmatrix} r \cos \alpha \\ r \sin \alpha \end{bmatrix} = \begin{bmatrix} r(\cos \theta \cos \alpha - \sin \theta \sin \alpha) \\ r(\sin \theta \cos \alpha + \cos \theta \sin \alpha) \end{bmatrix} = \begin{bmatrix} r \cos(\alpha + \theta) \\ r \sin(\alpha + \theta) \end{bmatrix},$$

也就是说, $G^T x$ 相当于将向量 x 按逆时针旋转 θ 角度. 因此当 $\theta = -\alpha$ 时, $Gx = \begin{bmatrix} r \\ 0 \end{bmatrix}$.





Givens 变换与矩阵的乘积

设 $A \in \mathbb{R}^{m \times n}$, $G = G(i, j, \theta) \in \mathbb{R}^m$, 则 $G^T A$ 只会影响其第 i 行和第 j 行的元素, 也就是说, 只需对 A 的第 i 行和第 j 行做 Givens 变换即可, 运算量大约为 $6n$.

同样地, 如果是右乘 Givens 变换, 则只会影响其第 i 列和第 j 列的元素, 运算量约为 $6m$.

 任何一个正交矩阵都可以写成若干个 Householder 矩阵或 Givens 矩阵的乘积 (见习题 3.5 和 3.22), 所以正交矩阵所对应的线性变换可以看作是反射变换和旋转变换的推广, 因此它不会改变向量的长度与 (不同向量之间的) 角度.

3.2.5 正交变换的舍入误差分析

引理 3.7 设 $P \in \mathbb{R}^{n \times n}$ 是一个精确的 Householder 或 Givens 变换, $\tilde{P}$ 是其浮点运算近似, 则

$$\text{fl}(\tilde{P}A) = P(A + E), \quad \text{fl}(A\tilde{P}) = (A + F)P,$$

其中 $\|E\|_2 = \mathcal{O}(\varepsilon_u) \cdot \|A\|_2$, $\|F\|_2 = \mathcal{O}(\varepsilon_u) \cdot \|A\|_2$.

这说明对一个矩阵做 Householder 变换或 Givens 变换是向后稳定的.

定理 3.8 考虑对矩阵 A 做一系列的正交变换, 则有

$$\text{fl}(\tilde{P}_k \cdots \tilde{P}_1 A \tilde{Q}_1 \cdots \tilde{Q}_k) = P_k \cdots P_1 (A + E) Q_1 \cdots Q_k,$$

其中 $\|E\|_2 = \mathcal{O}(\varepsilon_u) \cdot (k\|A\|_2)$. 这说明整个计算过程是向后稳定的.

一般地, 假设 X 是一个非奇异的线性变换, $\tilde{X}$ 是其浮点运算近似. 当 X 作用到 A 上时, 有

$$\text{fl}(\tilde{X}A) = XA + E = X(A + X^{-1}E) \triangleq X(A + F),$$

其中 $\|E\|_2 = \mathcal{O}(\varepsilon_u)\|XA\|_2 \leq \mathcal{O}(\varepsilon_u)\|X\|_2\|A\|_2$, 故

$$\|F\|_2 = \|X^{-1}E\|_2 \leq \mathcal{O}(\varepsilon_u)\|X^{-1}\|_2\|X\|_2\|A\|_2 = \mathcal{O}(\varepsilon_u)\kappa_2(X)\|A\|_2.$$

因此, 舍入误差可能会放大 $\kappa_2(X)$ 倍. 当 X 是正交变换时, $\kappa_2(X)$ 达到最小值 1, 这就是为什么在浮点运算中尽量使用正交变换的原因.



3.3 QR 分解

QR 分解是将一个矩阵分解一个正交矩阵 (酉矩阵) 和一个三角矩阵的乘积. QR 分解被广泛应用于线性最小二乘问题的求解和矩阵特征值的计算, 以及低秩逼近、主成分分析、模型降阶和电子结构等问题的计算与求解.

3.3.1 QR 分解的存在性与唯一性

定理 3.9 (QR 分解) [63] 设 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$), 则存在一个单位列正交矩阵 $Q \in \mathbb{C}^{m \times n}$ (即 $Q^*Q = I_{n \times n}$) 和一个上三角矩阵 $R \in \mathbb{C}^{n \times n}$, 使得

$$A = QR. \quad (3.8)$$

若 A 列满秩, 则存在一个具有正对角线元素的上三角矩阵 R 使得 (3.8) 成立, 且此时 QR 分解唯一, 即 Q 和 R 都唯一.

证明. 设 $A = [a_1, a_2, \dots, a_n] \in \mathbb{C}^{m \times n}$. 若 A 列满秩, 即 $\text{rank}(A) = n$. 则 QR 分解 (3.8) 就是对 A 的列向量组进行 Gram-Schmidt 正交化过程的矩阵描述 (见算法 3.3).

算法 3.3 Gram-Schmidt 正交化过程

```

1:  $r_{11} = \|a_1\|_2$ 
2:  $q_1 = a_1/r_{11}$ 
3: for  $j = 2$  to  $n$  do
4:    $q_j = a_j$ 
5:   for  $i = 1$  to  $j - 1$  do
6:      $r_{ij} = (a_j, q_i)$   % 计算内积, 用于正交化
7:      $q_j = q_j - r_{ij}q_i$ 
8:   end for
9:    $r_{jj} = \|q_j\|_2$ 
10:   $q_j = q_j/r_{jj}$ 
11: end for

```

由算法 3.3 可知

$$a_1 = r_{11}q_1, \quad a_j = r_{1j}q_1 + r_{2j}q_2 + \cdots + r_{jj}q_j = [q_1, q_2, \dots, q_j] \begin{bmatrix} r_{1j} \\ r_{2j} \\ \vdots \\ r_{jj} \end{bmatrix}, \quad j = 2, 3, \dots, n.$$

记 $Q = [q_1, q_2, \dots, q_n]$, $R = [r_{ij}]_{n \times n}$, 其中

$$r_{ij} = \begin{cases} q_i^* a_j, & \text{for } i \leq j \\ 0, & \text{for } i > j \end{cases} \quad (3.9)$$



于是 Gram-Schmidt 正交化过程可表示为

$$[a_1, a_2, \dots, a_n] = [q_1, q_2, \dots, q_n] \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & r_{n-1,n} \\ & & & r_{nn} \end{bmatrix}, \quad \text{即} \quad A = QR.$$

如果 A 不是列满秩, 我们可以通过下面的方式做类似的正交化过程:

- 如果 $a_1 = 0$, 则令 $q_1 = 0$; 否则令 $q_1 = a_1 / \|a_1\|_2$;
- 对于 $j = 2, 3, \dots, n$, 计算 $\tilde{q}_j = a_j - \sum_{i=1}^{j-1} (q_i^* a_j) q_i$. 如果 $\tilde{q}_j = 0$, 则表明 a_j 可以由 $a_1, a_2, \dots, a_{j-1}$ 线性表出, 令 $q_j = 0$. 否则令 $q_j = \tilde{q}_j / \|\tilde{q}_j\|_2$.

于是我们有

$$A = QR,$$

其中 $Q = [q_1, q_2, \dots, q_n]$ 列正交 (但不是单位列正交), 其列向量要么是单位向量, 要么就是零向量. $R = [r_{ij}]_{n \times n}$ 的定义同 (3.9). 需要注意的是, 如果 Q 的某一行 $q_k = 0$, 那么 R 中对应的第 k 行就全部为 0.

设 $\text{rank}(A) = l < n$, 则 Q 有 l 个非零列, 设为 $q_{i_1}, q_{i_2}, \dots, q_{i_l}$. 它们形成 $\mathbb{C}^m$ 中的一个单位正交向量组, 所以我们可以将其扩展成 $\mathbb{C}^m$ 中的一组标准正交基, 即

$$q_{i_1}, q_{i_2}, \dots, q_{i_l}, \tilde{q}_1, \dots, \tilde{q}_{m-l}.$$

然后用 $\tilde{q}_1$ 替换 Q 中的第一个零列, 用 $\tilde{q}_2$ 替换 Q 中的第二个零列, 依此类推, 将 Q 中的所有零列都替换掉. 将最后得到的矩阵记为 $\tilde{Q}$, 则 $\tilde{Q} \in \mathbb{C}^{m \times n}$ 单位列正交, 且

$$\tilde{Q}R = QR.$$

这是由于 $\tilde{Q}$ 中的新添加的列向量正好与 R 中的零行相对应. 所以我们有 QR 分解

$$A = \tilde{Q}R.$$

下面证明 **满秩矩阵 QR 分解的存在唯一性**.

存在性: 由于 A 列满秩, 由 Gram-Schmidt 正交化过程 (算法 3.3) 可知, 存在上三角矩阵 $R = [r_{ij}]_{n \times n}$ 满足 $r_{jj} > 0$, 使得 $A = QR$, 其中 Q 单位列正交.

唯一性: 假设 A 存在 QR 分解

$$A = Q_1 R_1 = Q_2 R_2,$$

其中 $Q_1, Q_2 \in \mathbb{C}^{m \times n}$ 单位列正交, $R_1, R_2 \in \mathbb{C}^{n \times n}$ 为具有正对角元素的上三角矩阵. 则有

$$Q_1 = Q_2 R_2 R_1^{-1}. \quad (3.10)$$

于是

$$1 = \|Q_1\|_2 = \|Q_2 R_2 R_1^{-1}\|_2 = \|R_2 R_1^{-1}\|_2.$$

又 R_1, R_2 均为上三角矩阵, 所以 $R_2 R_1^{-1}$ 也是上三角矩阵, 且其对角线元素为 $R_2(i, i) / R_1(i, i)$,



$i = 1, 2, \dots, n$. 因此

$$\frac{R_2(i, i)}{R_1(i, i)} \leq \rho(R_2 R_1^{-1}) \leq \|R_2 R_1^{-1}\|_2 \leq 1, \quad i = 1, 2, \dots, n.$$

同理可证 $R_1(i, i)/R_2(i, i) \leq 1$. 所以

$$R_1(i, i) = R_2(i, i), \quad i = 1, 2, \dots, n.$$

又 $\|Q_1\|_F^2 = \text{tr}(Q_1^* Q_1) = n$, 所以由 (3.10) 可知

$$\|R_2 R_1^{-1}\|_F^2 = \|Q_2 R_2 R_1^{-1}\|_F^2 = \|Q_1\|_F^2 = n.$$

由于 $R_2 R_1^{-1}$ 的对角线元素都是 1, 所以 $R_2 R_1^{-1}$ 只能是单位矩阵, 即 $R_2 = R_1$. 因此 $Q_2 = A R_2^{-1} = A R_1^{-1} = Q_1$, 即 A 的 QR 分解是唯一的. $\square$

 有时也将 QR 分解定义为: 存在酉矩阵 $Q \in \mathbb{C}^{m \times m}$ 使得

$$A = QR,$$

其中 $R = \begin{bmatrix} R_{11} \\ 0 \end{bmatrix} \in \mathbb{C}^{m \times n}$ 是上三角矩阵.

 如果 A 是实矩阵, 则上面证明中的运算都可以在实数下进行, 因此 Q 和 R 都可以是实矩阵.

 如果 A 是非奇异的方阵, 则 QR 分解也可以用来求解线性方程组 $Ax = b$.

Gram-Schmidt 正交化与正交投影

Gram-Schmidt 正交化过程的第 j 步:

$$\tilde{q}_j = a_j - r_{1j}q_1 - r_{2j}q_2 - \cdots - r_{j-1,j}q_{j-1}$$

可以看作是 a_j 减去其在 $\text{span}\{q_1, q_2, \dots, q_{j-1}\}$ 内的正交投影. 事实上, $r_{ij}q_i$ 就是 a_j 在 q_i 方向的正交投影, 即

$$P_{\text{span}\{q_i\}} a_j = (q_i q_i^\top) a_j = (q_i^\top a_j) q_i = r_{ij} q_i.$$

下面给出 QR 分解的具体实现方法, 分别基于 MGS 正交化过程, Householder 变换和 Givens 变换.

3.3.2 基于 MGS 的 QR 分解

在证明 QR 分解的存在性时, 我们利用了 Gram-Schmidt 正交化过程. 但由于数值稳定性方面的原因, 在实际计算中, 我们一般不采用 Gram-Schmidt 正交化过程, 取而代之的是 **修正的 Gram-Schmidt 正交化过程** (modified Gram-Schmidt process, **MGS**), 即对正交化过程做如下修改:

- Gram-Schmidt 正交化过程的第 j 步:

(1) 计算 $r_{ij} = (a_j, q_i)$, $i = 1, 2, \dots, j-1$;



- (2) 计算 $\tilde{q}_j = a_j - r_{1j}q_1 - r_{2j}q_2 - \cdots - r_{j-1,j}q_{j-1}$;
- (3) 计算 $r_{jj} = \|\tilde{q}_j\|$, $q_j = \tilde{q}_j/r_{jj}$;
- MGS 正交化过程的第 j 步:

(1) 令 $\tilde{q}_j = a_j$;

(2) 计算 $r_{ij} = (\tilde{q}_j, q_i)$, $\tilde{q}_j = \tilde{q}_j - r_{ij}q_i$, $i = 1, 2, \dots, j-1$;

(3) 计算 $r_{jj} = \|\tilde{q}_j\|$, $q_j = \tilde{q}_j/r_{jj}$;

可以证明, 数学上这两个算法完全等价, 即 r_{ij} 和 q_j 都一样. 但在数值上, Gram-Schmidt 正交化过程是不稳定的, 而 MGS 是向后稳定的 [84].

算法 3.4 基于 MGS 的 QR 分解

```
% Given  $A \in \mathbb{C}^{m \times n}$ , compute  $Q = [q_1, \dots, q_n] \in \mathbb{R}^{m \times n}$  and  $R \in \mathbb{R}^{n \times n}$  such that  $A = QR$ 
1: Set  $R = [r_{ij}] = 0_{n \times n}$  (the  $n \times n$  zero matrix)
2: if  $a_1 = 0$  then
3:    $q_1 = 0$ 
4: else
5:    $r_{11} = \|a_1\|_2$ ,  $q_1 = a_1/\|a_1\|_2$ 
6: end if
7: for  $j = 2$  to  $n$  do
8:    $q_j = a_j$ 
9:   for  $i = 1$  to  $j-1$  do    % MGS, 注意与 GS 的区别
10:     $r_{ij} = (q_j, q_i)$ ,  $q_j = q_j - r_{ij}q_i$ 
11:  end for
12:  if  $q_j \neq 0$  then
13:     $r_{jj} = \|q_j\|_2$ ,  $q_j = q_j/r_{jj}$ 
14:  end if
15: end for
```

 本算法的运算量大约为 $2mn^2$.

 算法中 R 是按列计算的, 因此上述算法也称为 **按列 MGS** (column-oriented MGS). 按列 MGS 不适合列主元, 因此, 如果需要计算列主元 QR 时需采用 **按行 MGS** (row-oriented MGS), 具体实现方法留作练习.

 由于按行 MGS 可以将内积转换为矩阵-向量乘积 (即同时计算 R 的一整行), 因此可以借助 2 级 BLAS 运算提高计算效率.

 由 MGS 得到的 QR 分解中, $Q \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{n \times n}$.



3.3.3 基于 Householder 变换的 QR 分解

由定理 3.5 可知, 通过 Householder 变换, 我们可以将任何一个非零变量 $x \in \mathbb{R}^n$ 转化成 $\|x\|_2 e_1$, 即除第一个元素外, 其它都为零. 下面我们就考虑通过 Householder 变换来实现矩阵的 QR 分解.

设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), 构造 Householder 变换 $H_1 \in \mathbb{R}^{m \times m}$, 使得

$$H_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} = \begin{bmatrix} r_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

于是

$$H_1 A = \left[\begin{array}{c|ccc} r_1 & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} \\ \hline 0 & & & \\ \vdots & & \tilde{A}_2 & \\ 0 & & & \end{array} \right],$$

其中 $\tilde{A}_2 \in \mathbb{R}^{(m-1) \times (n-1)}$. 同样地, 我们可以构造一个 Householder 变换 $\tilde{H}_2 \in \mathbb{R}^{(m-1) \times (m-1)}$, 将 $\tilde{A}_2$ 的第一列中除第一个元素外的所有元素都化为 0, 即

$$\tilde{H}_2 \tilde{A}_2 = \left[\begin{array}{c|ccc} r_2 & \tilde{a}_{23} & \cdots & \tilde{a}_{2n} \\ \hline 0 & & & \\ \vdots & & \tilde{A}_3 & \\ 0 & & & \end{array} \right].$$

令 $H_2 = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{H}_2 \end{bmatrix}$, 则 $H_2 \in \mathbb{R}^{m \times m}$ 也是 Householder 变换 (见习题 3.8), 且

$$H_2 H_1 A = \left[\begin{array}{cc|ccc} r_1 & \tilde{a}_{12} & \tilde{a}_{13} & \cdots & \tilde{a}_{1n} \\ 0 & r_2 & \tilde{a}_{23} & \cdots & \tilde{a}_{2n} \\ \hline 0 & 0 & & & \\ \vdots & \vdots & & \tilde{A}_3 & \\ 0 & 0 & & & \end{array} \right].$$

不断重复上述过程, 我们就可以得到一系列的 Householder 变换

$$H_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & \tilde{H}_k \end{bmatrix}, \quad \tilde{H}_k \in \mathbb{R}^{(n+1-k) \times (n+1-k)}, \quad k = 1, 2, \dots, n,$$

使得



$$H_n \cdots H_2 H_1 A = \begin{bmatrix} r_1 & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} \\ 0 & r_2 & \cdots & \tilde{a}_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & r_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \triangleq R.$$

由于 Householder 变换都是正交矩阵, 令

$$Q \triangleq (H_n \cdots H_2 H_1)^{-1} = H_1^{-1} H_2^{-1} \cdots H_n^{-1} = H_1 H_2 \cdots H_n,$$

则 Q 也是正交矩阵, 且

$$A = (H_{n-1} \cdots H_2 H_1)^{-1} R = QR.$$

以上就是基于 Householder 变换的 QR 分解的具体实现过程. 最后所得到的上三角矩阵 R 就存放在 A 的上三角部分.

 如果不需要生成 Q , 则基于 Householder 变换的 QR 分解的运算量大约为 $2mn^2 - 2n^3/3$.

矩阵 Q 可通过下面的累积方法来计算:

$$\begin{cases} Q = I_m, \\ Q = QH_k, \quad k = 1, 2, \dots, n. \end{cases}$$

如果保留了每一步的 Householder 向量, 则 Q 也可以通过下面的**向后累积方法**来计算:

$$\begin{aligned} Q &= I_m, \\ Q &= H_k Q, \quad k = n, n-1, \dots, 1. \end{aligned}$$

这样做的好处是一开始 Q 会非常稀疏 (左上角为单位矩阵), 随着迭代的进行, Q 才会慢慢变满. 而前面的计算方法, 第一步就将 Q 变成了一个满矩阵.

采用向后累积方法计算 Q 的运算量大约为 $4m^2n - 4mn^2 + 4n^3/3$. 如果只需要计算 Q 的前 n 列, 则运算量大约为 $2mn^2 - 2n^3/3$, 此时 QR 分解的总运算量为 $4mn^2 - 4n^3/3$. 若 $m = n$, 则为 $8n^3/3$.

算法 3.5 基于 Householder 变换的 QR 分解 (Householder-QR)

```
% Given  $A \in \mathbb{R}^{m \times n}$ , compute  $Q$  and  $R$  such that  $A = QR$  where  $Q \in \mathbb{R}^{m \times m}$  and  $R \in \mathbb{R}^{m \times n}$ 
1: for  $k = 1$  to  $n$  do
2:    $a_k = A(k : m, k)$ 
3:    $[\beta_k, v_k] = \text{House}(a_k)$     % 计算 Householder 向量
4:    $A(k : m, k : n) = (I_{m-k+1} - \beta_k v_k v_k^T) A(k : m, k : n)$ 
       $= A(k : m, k : n) - \beta_k v_k (v_k^T A(k : m, k : n))$ 
5: end for
```



```

6: Set  $Q = \begin{bmatrix} I_{n-1} & \\ & \tilde{H}_n \end{bmatrix}$  % 向后累积法计算  $Q$ 
7: for  $k = n - 1$  to  $1$  do
8:    $Q(k:m, k:m) = (I_{m-k+1} - \beta_k v_k v_k^T) Q(k:m, k:m)$ 
       $= Q(k:m, k:m) - (\beta_k v_k)(v_k^T Q(k:m, k:m))$ 
9: end for

```

上面的算法只是关于 Householder QR 分解的一个简单描述, 并没有考虑运算量问题. 在实际计算时, 我们通常会保留所有的 Householder 向量 (用于向后累积法计算 Q). 由于第 k 步中 $\tilde{H}_k$ 所对应的 Householder 向量 v_k 的长度为 $m - k + 1$, 因此我们先把 v_k 单位化, 使得 v_k 的第一元素为 1, 这样就只要存储 $v_k(2:end)$, 共 $m - k$ 个元素. 于是我们就可以把所有的 Householder 向量存放在 A 的严格下三角部分. 而 A 的上三角部分仍然存放 R .

事实上, 如果没有明确要求生成 Q , 通常只需存储 v_k 和 β_k , 即以因式分解 (Factored-Form) 形式存储 Q [49].

算法 3.5 中的 Q 是方阵, 如果要求 $Q \in \mathbb{R}^{m \times n}$, 则只需计算 Q 的前 n 列.

我们也可以考虑块 Householder QR 分解, 以便充分利用 3 级 BLAS 运算, 提高计算效率.

算法 3.5 针对的是实矩阵, 如果是复矩阵则要适当做些修改.

3.3.4 列主元 QR 分解

如果 A 不是满秩, 记 $l \triangleq \text{rank}(A) < n$, 则存在一个置换矩阵 P , 使得 AP 的前 l 列是线性无关的. 因此我们可以对 AP 进行 QR 分解, 于是我们可以得到下面的结论.

定理 3.10 (列主元 QR 分解) 设 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$), 且 $\text{rank}(A) = l < n$. 则存在置换矩阵 $P \in \mathbb{R}^{n \times n}$ 和正交矩阵 $Q \in \mathbb{C}^{m \times m}$, 使得

$$AP = Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}_{m \times n},$$

其中 $R_{11} \in \mathbb{C}^{l \times l}$ 是非奇异上三角矩阵, 且对角线元素满足 $r_{11} \geq r_{22} \geq \cdots \geq r_{ll} > 0$.

上述结论也可简化为

$$AP = Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix},$$

其中 $Q_1 \in \mathbb{C}^{m \times l}$ 单位列正交 (上述结论中 Q 的前 l 列).

列主元 QR 分解的实现过程与 QR 分解基本类似, 只是在第 k 步时, 需要选列主元, 同时可能需要做一个列交换.



假设经过 $k-1$ 步后, 我们得到下面的分解

$$AP^{(k-1)} = Q^{(k-1)} \begin{bmatrix} R_{11}^{(k-1)} & R_{12}^{(k-1)} \\ 0 & R_{22}^{(k-1)} \end{bmatrix} \triangleq Q^{(k-1)} R^{(k-1)}, \quad \text{即} \quad (Q^{(k-1)})^T AP^{(k-1)} = R^{(k-1)},$$

其中 $P^{(k-1)}$ 是置换矩阵, $Q^{(k-1)}$ 是正交矩阵, $R_{11}^{(k-1)} \in \mathbb{R}^{(k-1) \times (k-1)}$ 是非奇异上三角矩阵.

下面考虑第 k 步:

- (1) 首先计算 $R_{22}^{(k-1)}$ 的所有列的范数, 如果范数都为 0, 则 $R_{22}^{(k-1)} = 0$, 此时必有 $k-1 = l$, 算法结束.
- (2) 当 $k \leq l$ 时, $R_{22}^{(k-1)} \neq 0$, 记其范数最大的列为第 i_k 列 (如果有相等的, 取其中一个即可). 若 $i_k \neq 1$, 则交换 $R^{(k-1)}$ 的第 k 列与第 $i_k + k - 1$ 列, 并记相应的置换矩阵为 P_k .
- (3) 由于列交换不会影响到 $R^{(k-1)}$ 的前 $k-1$ 列, 因此列交换后的矩阵可记为

$$R^{(k-1)} P_k \triangleq \begin{bmatrix} R_{11}^{(k-1)} & \tilde{R}_{12}^{(k-1)} \\ 0 & \tilde{R}_{22}^{(k-1)} \end{bmatrix}.$$

构造 $\tilde{R}_{22}^{(k-1)}$ 的第 1 列所对应的 Householder 变换 $\tilde{H}_k$, 并令 $H_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & \tilde{H}_k \end{bmatrix}$, $P^{(k)} = P^{(k-1)} P_k$, 于是

$$H_k (Q^{(k-1)})^T AP^{(k)} = H_k R^{(k-1)} P_k = \begin{bmatrix} R_{11}^{(k-1)} & \tilde{R}_{12}^{(k-1)} \\ 0 & \tilde{H}_k \tilde{R}_{22}^{(k-1)} \end{bmatrix} \triangleq R^{(k)},$$

其中 $\tilde{H}_k \tilde{R}_{22}^{(k-1)}$ 的第一列除第一个元素外, 其余都是零, 且该元素的值等于 $\tilde{R}_{22}^{(k-1)}$ 的第 1 列的范数. 记 $Q^{(k)} \triangleq Q^{(k-1)} H_k^T$, 则

$$AP^{(k)} = Q^{(k)} R^{(k)} = \begin{bmatrix} R_{11}^{(k)} & R_{12}^{(k)} \\ 0 & R_{22}^{(k)} \end{bmatrix},$$

其中 $R_{11}^{(k)} \in \mathbb{R}^{k \times k}$ 为非奇异上三角矩阵.

依此类推, 直到第 l 步, 我们就可以得到 A 的列主元 QR 分解, 其中 R_{11} 的对角线元素非负且按降序排列是由列主元的选取方法和 Householder 变换的性质得到的.

下面是列主元 QR 分解的一个应用.

推论 3.11 (满秩分解) 设 $A \in \mathbb{C}^{m \times n}$ 且 $\text{rank}(A) = r \leq \min\{m, n\}$, 则存在满秩矩阵 $F \in \mathbb{C}^{m \times r}$ 和 $G \in \mathbb{C}^{r \times n}$, 使得

$$A = FG.$$

3.3.5 基于 Givens 变换的 QR 分解

我们同样可以利用 Givens 变换来做 QR 分解.



设 $A \in \mathbb{R}^{n \times n}$, 首先构造一个 Givens 变换 G_{21} , 作用在 A 的最前面的两行上, 使得

$$G_{21} \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix} = \begin{bmatrix} \tilde{a}_{11} \\ 0 \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix}.$$

由于 G_{21} 只改变矩阵的第 1 行和第 2 行的值, 所以其它行保存不变. 然后再构造一个 Givens 变换 G_{31} , 作用在 $G_{21}A$ 的第 1 行和第 3 行, 将其第一列的第三个元素化为零. 由于 G_{31} 只改变矩阵的第 1 行和第 3 行的值, 所以第二行的零元素维持不变. 以此类推, 我们可以构造一系列的 Givens 变换 $G_{41}, G_{51}, \dots, G_{n1}$, 使得 $G_{n1} \cdots G_{21}A$ 的第一列中除第一个元素外, 其它元素都化为零, 即

$$G_{n1} \cdots G_{21}A = \begin{bmatrix} * & * & \cdots & * \\ 0 & * & \cdots & * \\ \vdots & \vdots & & \vdots \\ 0 & * & \cdots & * \end{bmatrix}.$$

下面我们可以对第二列进行类似的处理. 构造 Givens 变换 $G_{32}, G_{42}, \dots, G_{n2}$, 将第二列的第 3 至第 n 个元素全化为零, 同时保持第一列不变.

以此类推, 我们对其他列也做类似的处理. 最后, 通过构造 $\frac{1}{2}n(n-1)$ 个 Givens 变换, 将 A 转化成一个上三角矩阵 R , 即

$$R = G_{n,n-1} \cdots G_{21}A.$$

令 $Q = (G_{n,n-1} \cdots G_{21})^T$. 由于 Givens 变换是正交矩阵, 所以 Q 也是正交矩阵. 于是, 我们就得到矩阵 A 的 QR 分解

$$A = QR.$$

与 Householder 变换一样, 在进行 Givens 变换时, 我们不需要显式地写出 Givens 矩阵.

-  对于稠密矩阵而言, 基于 Givens 变换的 QR 分解的运算量比 Householder 变换要多很多.
-  当需要连续应用一系列 Givens 变换时, 我们可以使用快速 Givens 变换 (见 [3]). 但即便如此, 其速度仍然要慢于 Householder 变换. 因此基于 Givens 变换的 QR 分解主要用于当矩阵的非零下三角元素相对较少时的情形 (比如对上 Hessenberg 矩阵进行 QR 分解), 或者稀疏矩阵 (可以尽可能地保留矩阵的稀疏性).
-  另外, 由于每次 Givens 变换只影响矩阵的两行或者两列, 因此非常适合并行计算.
-  如果 $A \in \mathbb{R}^{m \times n}$, 其中 $m > n$, 我们仍然可以通过 Givens 变换进行 QR 分解.

下面是基于 Givens 变换的 QR 分解的算法描述.

算法 3.6 基于 Givens 变换的 QR 分解

- 1: Set $Q = I_{m \times m}$
- 2: **for** $k = 1$ to n **do**
- 3: **for** $i = k + 1$ to m **do**



```

4:      [c, s]=givens( $a_{kk}$ ,  $a_{ik}$ )
5:       $\begin{bmatrix} A(k, k:n) \\ A(i, k:n) \end{bmatrix} = G \begin{bmatrix} A(k, k:n) \\ A(i, k:n) \end{bmatrix}$  where  $G = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$ 
6:       $[Q(1:m, k), Q(1:m, i)] = [Q(1:m, k), Q(1:m, i)]G^T$ 
7:  end for
8: end for

```

3.3.6 基于 Cholesky 分解的 QR 分解

设 $A \in \mathbb{R}^{m \times n}$ 列满秩, 则 $A^T A$ 对称正定, 因此存在 Cholesky 分解, 设为

$$A^T A = R^T R,$$

其中 $R \in \mathbb{R}^{n \times n}$ 是对角线都为正数的上三角矩阵. 令

$$Q \triangleq AR^{-1},$$

则有 $A = QR$, 且

$$Q^T Q = (R^{-1})^T A^T A R^{-1} = (R^{-1})^T (R^T R) R^{-1} = I.$$

由此我们就可以得到 A 的 QR 分解.

 Faddeev 等 [38] 于 1968 年就给出了该结论的证明.

算法 3.7 基于 Cholesky 分解的 QR 分解 (Cholesky-QR)

- 1: Compute $B = A^T A$
 - 2: Compute the Cholesky decomposition of B : $B = R^T R$
 - 3: Compute $Q = AR^{-1}$.
-

 易知 Cholesky-QR (算法 3.7) 的运算量大约为 $2mn^2 + \frac{2}{3}n^3 + 2mn$. 当 $m \gg n$ 时, 运算量与 MGS 基本相当. 但其中的主要运算是矩阵-矩阵乘积, 因此可以借助 3 级 BLAS 运算提升计算效率.

 由于 Cholesky-QR 中涉及计算 $A^T A$, 且 $A^T A$ 的条件数是 A 的条件数的平方, 因此当 A 的条件数较大时, 该算法不建议直接使用. 可以使用改进后的 Cholesky-QR2 [116] 或 shifted Cholesky-QR3 [42] 算法.

3.3.7 QR 分解的稳定性

基于 Householder 变换和 Givens 变换的 QR 分解都具有很好的数值稳定性. 详细分析可以参考 [115] 和 [61]. 基于 MGS 的 QR 分解也是向后稳定的 [84]. 由于 $A^T A$ 的条件数是 A 的条件数的平方, 因此基于 Cholesky-QR 是不稳定的.



当需要计算矩阵 Q 时, 基于 MGS 的 QR 分解的运算量相对较少. 因此当 A 的列向量具有很好的线性无关性时, 我们可以使用 MGS 来计算 QR 分解. 但是, 由于舍入误差的原因, 最后得到的矩阵 Q 会带有一定的误差, 可能会导致 Q 失去正交性. Björck 等 [2, 93] 证明了, 通过 MGS 计算的矩阵 Q 满足

$$Q^T Q = I + E_{MGS}, \quad \text{其中} \quad \|E_{MGS}\|_2 = \frac{c_1(m, n)\varepsilon_u \kappa_2(A)}{1 - c_1(m, n)\varepsilon_u \kappa_2(A)},$$

其中 $c_1(m, n)$ 是与 m, n 相关的常数.

而通过 Householder 变换计算的矩阵 Q 满足

$$Q^T Q = I + E_H, \quad \text{其中} \quad \|E_H\|_2 \approx \varepsilon_u.$$

因此, 如果正交性至关重要, 则当 A 的列向量接近线性相关时, 建议使用 Householder 变换.

 实际使用时也可以通过重正交来提升 MGS 的稳定性, 即在第 j 步, 将 a_j 与 $q_1, \dots, q_{j-1}$ 正交化后, 再进行一次正交化.

另外, 基于 MGS 的 QR 分解所计算得到的 Q (由于舍入误差的原因, Q 不一定是列正交的) 和 R 满足 [2, 93]

$$\|A - QR\| \approx \|A\|_2 \varepsilon_u, \quad (\text{或 } \|A - QR\| \leq c_2(m, n)\|A\|_2 \varepsilon_u)$$

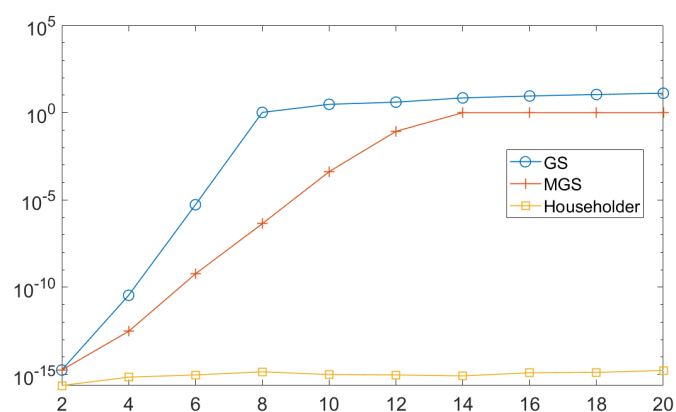
而且存在单位列正交矩阵 $\tilde{Q}$, 使得 [49]

$$\|A - \tilde{Q}R\| \approx \|A\|_2 \varepsilon_u.$$

例 3.3 编写程序, 分别用 GS, MGS 和 Householder 变换计算 n 阶 Hilbert 矩阵 H 的 QR 分解, 并比较三种算法的稳定性, 即观察 $\|\tilde{Q}\tilde{R} - H\|_2$ 和 $\|\tilde{Q}^T \tilde{Q} - I\|_2$ 的值, 其中 $\tilde{Q}$ 和 $\tilde{R}$ 是计算出来的 QR 分解矩阵因子. 试验结果如下. (QR_3methods.m)

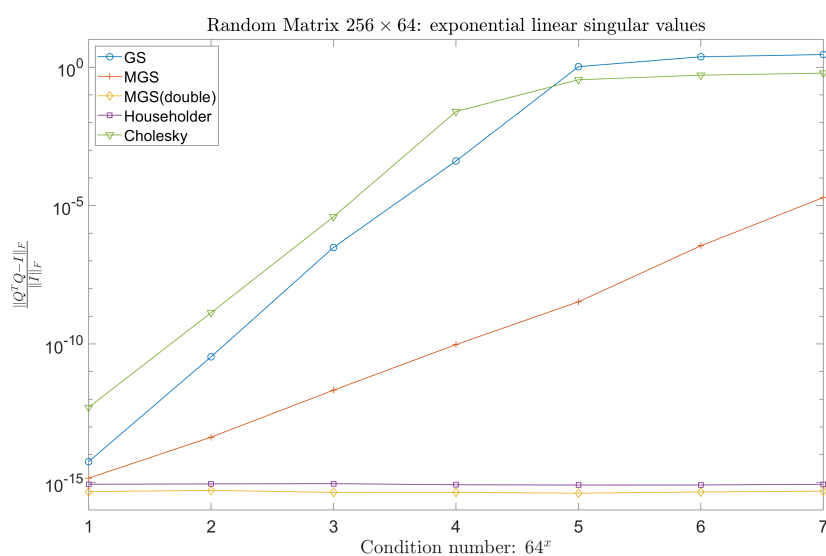
n	GS		MGS		Householder	
	$\ \tilde{Q}\tilde{R} - H\ _2$	$\ \tilde{Q}^T \tilde{Q} - I\ _2$	$\ \tilde{Q}\tilde{R} - H\ _2$	$\ \tilde{Q}^T \tilde{Q} - I\ _2$	$\ \tilde{Q}\tilde{R} - H\ _2$	$\ \tilde{Q}^T \tilde{Q} - I\ _2$
2	0.00e+00	1.81e-15	0.00e+00	1.81e-15	1.24e-16	2.36e-16
4	3.93e-17	3.45e-11	5.55e-17	2.98e-13	2.46e-16	7.08e-16
8	7.48e-17	1.03e+00	6.59e-17	4.38e-07	2.57e-16	1.44e-15
16	7.82e-17	9.00e+00	6.70e-17	9.98e-01	2.79e-16	1.27e-15





例 3.4 编写程序, 分别用 Cholesky 分解, GS, MGS, 重正交 MGS 和 Householder 变换计算 QR 分解. 下面是针对 256 的随机矩阵, 在不同条件数下各种方法的稳定性, 试验结果如下, 其中 MGS(double) 是指重正交 MGS. (由于舍入误差的原因, 当条件数较大时, $A^T A$ 可能会奇异, 因此在计算 $A^T A$ 的 Cholesky 分解时加了个正的数量阵)

(QR_3methods_MGS2.m, QR_3methods_256_64.m)



3.4 奇异值分解

奇异值分解 (Singular Value Decomposition, SVD) 分解是矩阵计算中非常有用的工具之一, 在图像处理、机器学习、数据科学等领域有着非常重要的应用.



3.4.1 奇异值, 奇异向量和奇异值分解

设 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$), 则 $A^*A \in \mathbb{C}^{n \times n}$ 和 $AA^* \in \mathbb{C}^{m \times m}$ 都是 Hermite 半正定矩阵, 且它们具有相同的非零特征值 (都是正实数).

定理 3.12 (SVD) [49] 设 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$), 则存在酉矩阵 $U \in \mathbb{C}^{m \times m}$ 和 $V \in \mathbb{C}^{n \times n}$ 使得

$$U^*AV = \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} \quad \text{或} \quad A = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^*, \quad (3.11)$$

其中 $\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_n) \in \mathbb{R}^{n \times n}$, 且 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$. 分解 (3.11) 称为 A 的**奇异值分解 (SVD)**, 而 $\sigma_1, \sigma_2, \dots, \sigma_n$ 则称为 A 的**奇异值**. (板书)

该定理也可以通过 Hermite 半正定矩阵的特征值分解来证明.

如果 $A \in \mathbb{R}^{m \times n}$ 是实矩阵, 则 U, V 也都可以是实矩阵 [63].

由 (3.11) 可知,

$$A^*A = V\Sigma^*\Sigma V^*, \quad AA^* = U \begin{bmatrix} \Sigma\Sigma^* & 0 \\ 0 & 0 \end{bmatrix} U^*.$$

所以 $\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2$ 是 A^*A 和 AA^* 的特征值. 因此, A 的奇异值就是 A^*A 的特征值的平方根.

奇异向量

由 SVD (3.11) 可知

$$Av_i = \sigma_i u_i, \quad i = 1, 2, \dots, n,$$

$$A^*u_i = \sigma_i v_i, \quad i = 1, 2, \dots, n,$$

$$A^*u_i = 0, \quad i = n+1, n+2, \dots, m.$$

我们称 $U = [u_1, u_2, \dots, u_m]$ 和 $V = [v_1, v_2, \dots, v_n]$ 的列向量分别称为 A 的**左奇异向量**和**右奇异向量**.

细 SVD (降阶 SVD)

由定理 3.12 可知

$$A = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^* = \sigma_1 u_1 v_1^* + \sigma_2 u_2 v_2^* + \dots + \sigma_n u_n v_n^* = \sum_{i=1}^n \sigma_i u_i v_i^*.$$

记 $U_n = [u_1, u_2, \dots, u_n] \in \mathbb{C}^{m \times n}$, 则 U_n 是单位列正交矩阵 (即 $U_n^* U_n = I_{n \times n}$), 且

$$A = U_n \Sigma V^*. \quad (3.12)$$

这就是所谓的**细 SVD** (或瘦 SVD, **thin SVD**, skinny SVD) [49] 或 **降阶 SVD** (**reduced SVD**) [103], 有的文献将 (3.12) 称为奇异值分解.



压缩 SVD

设 $r \triangleq \text{rank}(A) \leq n$, 则有

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0, \quad \sigma_{r+1} = \cdots = \sigma_n = 0.$$

我们称

$$A_r = \sum_{i=1}^r \sigma_i u_i v_i^*$$

为 A 的**压缩 SVD** (condensed SVD).

截断 SVD

设 $k < n$, 我们称

$$A_k = \sigma_1 u_1 v_1^* + \sigma_2 u_2 v_2^* + \cdots + \sigma_k u_k v_k^* = \sum_{i=1}^k \sigma_i u_i v_i^*$$

为 A 的**截断 SVD** (truncated SVD).

 A_k 也可以表示为 $A_k = P_{\text{Ran}(U_k)} A$, 其中 $P_{\text{Ran}(U_k)}$ 是 $\text{Ran}(U_k)$ 上的正交投影变换 ($U_k = [u_1, u_2, \dots, u_k]$).

 由于压缩 SVD 和截断 SVD 在计算和存储方面存在优势, 因此实际应用中经常使用的是压缩 SVD 或截断 SVD.

奇异值的应用

SVD 可用来计算矩阵的低秩逼近, 矩阵行空间和列空间的一组正交基, 以及矩阵的秩.

- 矩阵计算: 矩阵范数, 矩阵条件数, 最小二乘问题, 广义逆, 矩阵和张量的低秩分解, 等等.
- 工程应用: 信号处理, 图像压缩, 机器学习, 主成分分析, 数据降维, 等等.

Six Great Theorems of Linear Algebra

- **Dimension Theorem:** All bases for a vector space have the same number of vectors.
- **Counting Theorem:** Dimension of column space + dimension of nullspace = number of columns.
- **Rank Theorem:** Dimension of column space = dimension of rowspace. This is the rank.
- **Fundamental Theorem:** The row space and nullspace of A are orthogonal complements in $\mathbb{R}^n$.
- **SVD:** There are orthonormal bases (v 's and u 's for the row and column spaces) so that $Av_i = \sigma_i u_i$.
- **Spectral Theorem:** If $A^T = A$ there are orthonormal q 's so that $Aq_i = \lambda_i q_i$ and



$$A = Q\Lambda Q^T.$$

— *Introduction to Linear Algebra*, 5th, G. Strang, 2016.

3.4.2 奇异值基本性质

下面是关于奇异值的一些基本性质:

定理 3.13 设 $A = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^*$ 是 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$) 的奇异值分解, 则下面结论成立:

- (1) A^*A 的特征值是 σ_i^2 , 对应的特征向量是 $v_i, i = 1, 2, \dots, n$;
- (2) AA^* 的特征值是 σ_i^2 和 $m - n$ 个零, 对应的特征向量是 $u_i, i = 1, 2, \dots, m$;
- (3) $\|A\|_2 = \sigma_1, \|A\|_F = \sqrt{\sigma_1^2 + \sigma_2^2 + \dots + \sigma_n^2}$;
- (4) 若 $\text{rank}(A) = r \leq n$, 则

$$\text{Ran}(A) = \text{span}\{u_1, u_2, \dots, u_r\}, \quad \text{Ker}(A) = \text{span}\{v_{r+1}, v_{r+2}, \dots, v_n\};$$

- (5) 设 $x \in \mathbb{C}^n$ 且 $\|x\|_2 = 1$, 则 $\sigma_n \leq \|Ax\|_2 \leq \sigma_1$;
(事实上 $\sigma_1 = \max_{\|x\|_2=1} \|Ax\|_2, \sigma_n = \min_{\|x\|_2=1} \|Ax\|_2$)
- (6) (酉不变性) 设 $X \in \mathbb{C}^{m \times m}$ 和 $Y \in \mathbb{C}^{n \times n}$ 是酉矩阵, 则 $\sigma_i(X^*AY) = \sigma_i(A)$.

(留作练习)

 性质 (4) 可以用来计算矩阵的像空间和零空间.

定理 3.14 设 $A = U\Sigma V^*$ 是 $A \in \mathbb{C}^{n \times n}$ 的奇异值分解, 则下面结论成立:

- (1) $|\det(A)| = \sigma_1 \sigma_2 \cdots \sigma_n$;
- (2) 若 A 非奇异, 则 $\|A^{-1}\|_2 = \sigma_n^{-1}, \kappa_2(A) = \sigma_1/\sigma_n$;
- (3) 若 A 是 Hermite 的, 且 $A = U\Lambda U^*$ 是 A 的酉特征值分解, 即 $U^*U = I, \Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$. 设 $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n|$, 则 $A = U\Sigma V$ 是 A 的奇异值分解, 其中 $\sigma_i = |\lambda_i|, v_i = \text{sign}(\lambda_i)u_i$, 若 $\lambda_i = 0$, 则取 $v_i = u_i$;
- (4) 矩阵 $H = \begin{bmatrix} 0 & A^* \\ A & 0 \end{bmatrix}$ 的特征值是 $\pm\sigma_i$, 对应的单位特征向量为 $\frac{1}{\sqrt{2}} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix}$. ($A \in \mathbb{C}^{m \times n}$ 时也有类似结论)

(留作练习)

例 3.5 由奇异值的性质可知, 矩阵的谱条件数取决于最大奇异值和最小奇异值的比值, 如果最大奇异值与最小奇异值比较接近, 则谱条件数就比较小. 那么, 谱条件数与特征值有没有直接关系? 我们知道, 如果矩阵 A 对称正定, 则其谱条件数就是最大特征值与最小特征值的比值. 但



是, 对于一般的矩阵, 则没有这个性质. 如下面的矩阵:

$$A = \begin{bmatrix} 1 & -\frac{1}{2} & \cdots & -\frac{1}{2} \\ & 1 & \ddots & \vdots \\ & & \ddots & -\frac{1}{2} \\ & & & 1 \end{bmatrix} \in \mathbb{R}^{n \times n},$$

即对角线全是 1, 严格上三角部分全是 $-\frac{1}{2}$, 其它为 0. 易知 A 的所有特征值都是 1, 但其谱条件数却随着矩阵规模的增大而快速变大, 见下表:

n	10	20	30	40	50	60	70	80
$\kappa_2(A)$	6.3e+01	7.6e+03	6.8e+05	5.3e+07	3.9e+09	2.7e+11	1.8e+13	1.2e+15

另外, 通过直接计算可知

$$A^{-1} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1.5}{2} & \frac{1.5^2}{2} & \cdots & \frac{1.5^{n-2}}{2} \\ & 1 & \frac{1}{2} & \frac{1.5}{2} & \ddots & \vdots \\ & & 1 & \ddots & \ddots & \frac{1.5^2}{2} \\ & & & \ddots & \frac{1}{2} & \frac{1.5}{2} \\ & & & & 1 & \frac{1}{2} \\ & & & & & 1 \end{bmatrix}.$$

因此, $\kappa_1(A)$ 和 $\kappa_\infty(A)$ 都是矩阵维数 n 的幂函数.

3.4.3 奇异值更多性质

下面是关于矩阵 A 的一个低秩逼近.

定理 3.15 设 $A = U_n \Sigma V^*$ 是 $A \in \mathbb{C}^{m \times n}$ 的细奇异值分解, 令 $A_k = \sum_{i=1}^k \sigma_i u_i v_i^*$, 其中 $1 \leq k \leq \text{rank}(A)$, 则 A_k 是

$$\min_{B \in \mathbb{C}^{m \times n}, \text{rank}(B)=k} \|A - B\|_2 \quad (3.13)$$

的一个解, 且

$$\|A - A_k\|_2 = \sigma_{k+1}.$$

此时, 我们称 A_k 是 A 的一个秩 k 逼近.

(板书)

 定理中的 (3.13) 式也可以改写为

$$\min_{B \in \mathbb{C}^{m \times n}, \text{rank}(B) \leq k} \|A - B\|_2 \quad (3.14)$$

 对于 Frobenius 范数, 我们有类似的结论, 见习题 3.13.



定理 3.16 (Weyl) [99] 设 $A, B \in \mathbb{C}^{m \times n}$ ($m \geq n$), 且 $\text{rank}(B) = k$, 则有

$$\max_{x \in \text{Ker}(B), \|x\|_2=1} \|Ax\|_2 \geq \sigma_{k+1}(A), \quad (3.15)$$

和

$$\min_{x \in \text{Ker}(B), \|x\|_2=1} \|Ax\|_2 \leq \sigma_{n-k}(A). \quad (3.16)$$

因此,

$$\sigma_1(A - B) \geq \sigma_{k+1}(A), \quad \sigma_n(A - B) \leq \sigma_{n-k}(A). \quad (3.17)$$

(板书)

 不等式 (3.15) 和 (3.16) 也可以写为: 设 $\mathcal{L}$ 是 $\mathbb{C}^n$ 中的任意一个 $n - k$ 维子空间, 则有

$$\max_{x \in \mathcal{L}, \|x\|_2=1} \|Ax\|_2 \geq \sigma_{k+1}(A), \quad \min_{x \in \mathcal{L}, \|x\|_2=1} \|Ax\|_2 \leq \sigma_{n-k}(A).$$

特别地, 当 $k = 0$ 时 $\mathcal{L} = \mathbb{R}^n$, 因此

$$\max_{\|x\|_2=1} \|Ax\|_2 \geq \sigma_1(A), \quad \min_{\|x\|_2=1} \|Ax\|_2 \leq \sigma_n(A).$$

事实上, 我们有 (第一个等式就是 2-范数的定义, 第二个等式留作自行练习)

$$\max_{\|x\|_2=1} \|Ax\|_2 = \sigma_1(A), \quad \min_{\|x\|_2=1} \|Ax\|_2 = \sigma_n(A).$$

 从该结论可知, 如果通过对 A 进行秩 k ($1 \leq k \leq \frac{n}{2}$) 修正来改善其谱条件数, 则最佳情况是降低到 $\frac{\sigma_{n-k}(A)}{\sigma_{k+1}}$.

定理 3.17 [99] 设 $A, B \in \mathbb{C}^{m \times n}$ ($m \geq n$), 则有

$$\sigma_{i+j-1}(A + B) \leq \sigma_i(A) + \sigma_j(B), \quad i = 1, 2, \dots, n, \quad j = 1, 2, \dots, n - i + 1. \quad (3.18)$$

(板书)

 令 $A = X - Y$, $B = Y$, 则该结论也可以描述为

$$\sigma_i(X - Y) \geq \sigma_{i+j-1}(X) - \sigma_j(Y), \quad i = 1, 2, \dots, n, \quad j = 1, 2, \dots, n - i + 1.$$

根据定理 3.16, 我们可以得到矩阵奇异值的最小最大定理.

定理 3.18 设 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$ 是矩阵 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$) 的奇异值, 则

$$\sigma_k(A) = \min_{\dim(\mathcal{L})=n-k+1} \max_{x \in \mathcal{L}, \|x\|_2=1} \|Ax\|_2, \quad k = 1, 2, \dots, n$$

和

$$\sigma_k(A) = \max_{\dim(\mathcal{L})=k} \min_{x \in \mathcal{L}, \|x\|_2=1} \|Ax\|_2, \quad k = 1, 2, \dots, n,$$

其中 $\mathcal{L}$ 表示 $\mathbb{C}^n$ 的一个子空间.

(留作练习)



引理 3.19 (交错不等式) [62] 设 $A \in \mathbb{C}^{m \times n}$, A_r 是由 A 去除 r 列 (或 r 行) 后得到的子矩阵, 则

$$\sigma_i(A) \geq \sigma_i(A_r) \geq \sigma_{i+r}(A), \quad i = 1, 2, \dots, \min\{m, n\}.$$

这里, 当下标 k 大于矩阵的维数时, 我们令 $\sigma_k = 0$. 进一步, 如果 $B \in \mathbb{C}^{(m-r) \times (n-s)}$ 是 A 的子矩阵, 则

$$\sigma_i(A) \geq \sigma_i(B) \geq \sigma_{i+r+s}(A).$$

(留作课外自习)

引理 3.20 [62] 设 $A \in \mathbb{C}^{n \times n}$, $1 \leq k \leq n$, 则对任意的单位列正交矩阵 $U_k \in \mathbb{C}^{n \times k}$ 和 $V_k \in \mathbb{C}^{n \times k}$, 有

$$\sigma_i(U_k^* A V_k) \leq \sigma_i(A), \quad i = 1, 2, \dots, k. \quad (3.19)$$

因此,

$$|\det(U_k^* A V_k)| \leq \sigma_1(A) \sigma_2(A) \cdots \sigma_k(A). \quad (3.20)$$

(板书)

定理 3.21 (Weyl 不等式, 1949) [62] 设 $A \in \mathbb{C}^{n \times n}$, 其奇异值和特征值分别为 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ 和 $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$, 则

$$|\lambda_1 \lambda_2 \cdots \lambda_k| \leq \sigma_1 \sigma_2 \cdots \sigma_k, \quad k = 1, 2, \dots, n$$

且当 $k = n$ 时, 等号成立.

(板书)

3.4.4 奇异值扰动分析

下面的定理是关于奇异值的一个扰动性质.

定理 3.22 [99] 设 $A, E \in \mathbb{C}^{m \times n}$ ($m \geq n$), 则

$$|\sigma_i(A + E) - \sigma_i(A)| \leq \|E\|_2, \quad i = 1, 2, \dots, n.$$

(留作练习, 可直接利用不等式 (3.18))

3.5 线性最小二乘问题的求解方法

3.5.1 正规方程

这里我们考虑超定线性最小二乘问题的求解.

定理 3.23 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), 则 $x_* \in \mathbb{R}^n$ 是线性最小二乘问题 (3.1) 的解当且仅当残量



$r = b - Ax_*$ 与 $\text{Ran}(A)$ 正交, 即 x_* 是下面的正规方程的解

$$A^T(b - Ax) = 0 \quad \text{或} \quad A^T Ax = A^T b. \quad (3.21)$$

(板书)

由定理 3.23 可知, 求线性最小二乘问题 (3.1) 的解等价于求正规方程 (3.21) 的解. 由于

$$A^T b \in \text{Ran}(A^T) = \text{Ran}(A^T A),$$

因此正规方程 $A^T Ax = A^T b$ 是相容 (consistent) 的, 即最小二乘解总是存在的. 当 A 非奇异时, 这个解也是唯一的.

定理 3.24 设 $A \in \mathbb{R}^{m \times n}$ ($m > n$). 则 $A^T A$ 对称正定当且仅当 A 是列满秩的, 即 $\text{rank}(A) = n$. 此时, 线性最小二乘问题 (3.1) 的解是唯一的, 其表达式为

$$x = (A^T A)^{-1} A^T b.$$

最小二乘解的几何含义

根据定理 3.23, 我们可以把 b 写成

$$b = Ax_* + r, \quad \text{其中} \quad r \perp \text{Ran}(A). \quad (3.22)$$

所以 Ax_* 就是 b 在 $\text{Ran}(A)$ 上的正交投影, 见图 3.2.

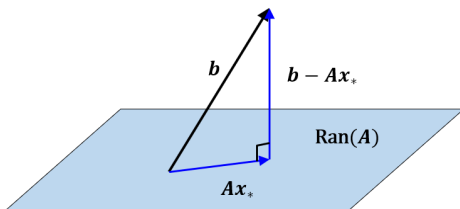


图 3.2. 最小二乘解的几何描述

🔴 最小二乘解可能并不唯一, 但分解 (3.22) 总是唯一的.

3.5.2 Cholesky 分解法

当 A 列满秩时, 我们就可以使用 Cholesky 分解来求解正规方程, 总的运算量大约为 $mn^2 + \frac{1}{3}n^3 + mn + \mathcal{O}(n^2)$, 其中大部分的运算量 (mn^2) 是用来计算 $A^T A$ (由于 $A^T A$ 对称, 因此只需计算其下三角部分).

🔴 通过直接求解正规方程来求解最小二乘问题, 运算量小, 而且简单直观.



但由于 $A^T A$ 的条件数是 A 的条件数的平方, 因此对于病态情形 (即 A 的条件数比较大), 不建议使用该方法.

例 3.6 下面的例子说明, 计算 $A^T A$ 可能会损失计算精度: 设

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

则

$$A^T A = \begin{bmatrix} 1 + \varepsilon^2 & 1 & 1 \\ 1 & 1 + \varepsilon^2 & 1 \\ 1 & 1 & 1 + \varepsilon^2 \end{bmatrix}.$$

记 ε_u 为机器精度, 则当 $\varepsilon_u < \varepsilon < \sqrt{\varepsilon_u}$ 时有 $\varepsilon^2 < \varepsilon_u$, 由于舍入误差的原因, 通过浮点运算计算得到的 $A^T A$ 是奇异的. 但我们注意到 A 是满秩的.

3.5.3 QR 分解法

这里假定 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 是满秩的. 设 A 的 QR 分解为 $A = QR$, 我们用三种不同的方法来推导线性最小二乘问题的解.

(1) 将 Q 的扩充成一个正交矩阵, 记为 $[Q, \hat{Q}] \in \mathbb{R}^{m \times m}$. 于是有

$$\begin{aligned} \|Ax - b\|_2^2 &= \|[Q, \hat{Q}]^T(Ax - b)\|_2^2 = \|[Q, \hat{Q}]^T(QRx - b)\|_2^2 \\ &= \left\| \begin{bmatrix} Rx - Q^T b \\ -\hat{Q}^T b \end{bmatrix} \right\|_2^2 = \|Rx - Q^T b\|_2^2 + \|\hat{Q}^T b\|_2^2 \geq \|\hat{Q}^T b\|_2^2, \end{aligned}$$

等号成立当且仅当 $Rx = Q^T b$. 所以最小二乘解为

$$x_* = R^{-1}Q^T b.$$

(2) 由于 Q 的列向量构成 $\text{Ran}(A)$ 的一组单位正交基, 所以 QQ^T 和 $I - QQ^T$ 分别是 $\text{Ran}(A)$ 和 $\text{Ran}(A)^\perp$ 上的正交投影矩阵. 又 $Ax \in \text{Ran}(A)$, 所以

$$\begin{aligned} \|Ax - b\|_2^2 &= \|QQ^T(Ax - b)\|_2^2 + \|(I - QQ^T)(Ax - b)\|_2^2 \\ &= \|QRx - QQ^T b\|_2^2 + \|(I - QQ^T)b\|_2^2 \\ &= \|Rx - Q^T b\|_2^2 + \|(I - QQ^T)b\|_2^2 \\ &\geq \|(I - QQ^T)b\|_2^2, \end{aligned}$$

等号成立当且仅当 $Rx = Q^T b$. 所以最小二乘解为

$$x_* = R^{-1}Q^T b.$$



 事实上, 由图 3.2 可知, x_* 满足

$$Ax_* = QQ^T b,$$

即 $QRx_* = QQ^T b$, 由此可得 $x_* = R^{-1}Q^T b$.

(3) 解正规方程. 由定理 3.23 可知, 最小二乘解为

$$x_* = (A^T A)^{-1} A^T b = (R^T Q^T Q R)^{-1} R^T Q^T b = (R^T R)^{-1} R^T Q^T b = R^{-1} Q^T b.$$

 用基于 MGS 的 QR 分解求解最小二乘问题时的运算量大约为 $2mn^2$, 而基于 Householder 变换的 QR 分解则只需 $2mn^2 - 2n^3/3$ (不需要显式计算 Q , 只需将 Householder 变换直接作用到 b 上即可计算出 $Q^T b$). 因此当 $m \gg n$ 时, QR 分解法的计算量大约是 Cholesky 分解的两倍.

 因此当 $m \gg n$ 时, QR 分解法的计算量大约是 Cholesky 分解法的两倍. 当 $m = n$ 时, 几乎相同.

 通常 QR 算法比较稳定, 是求解最小二乘问题的首选方法, 特别是当 A 条件数较大 (病态) 时.

3.5.4 奇异值分解法

设 $A \in \mathbb{R}^{m \times n}$ 列满秩, $A = U_n \Sigma V^T$ 是 A 的降阶奇异值分解, 则

$$x_* = (A^T A)^{-1} A^T b = (V \Sigma U_n^T U_n \Sigma V^T)^{-1} V \Sigma U_n^T b = (V \Sigma^{-2} V^T) V \Sigma U_n^T b = V \Sigma^{-1} U_n^T b.$$

 相比于 Cholesky 分解法和 QR 分解, 用 SVD 求解最小二乘问题具有更高的健壮性, 但由于需要计算系数矩阵的 SVD, 运算量远超 Cholesky 分解法和 QR 分解. 所以只有当系数矩阵秩亏或者接近秩亏时才使用 (此时 QR 分解法可能会失效).

例 3.7 分别用三种方法求解最小二乘问题, 比较运算时间.

(LS_3methods.m)

小结: 求解线性最小二乘问题

- Cholesky 分解法最快 (不仅运算量少, 而且计算效率高), 但精度最差, 适合条件数比较小的情形.
- QR 分解法的运算量大约是 Cholesky 分解法的 2 倍, 但算法稳定.
- SVD 法精度最高, 但运算量是前面两个方法的好几倍, 一般用于特别病态 (接近秩亏) 情形, 或者秩亏情形.



3.6 广义逆与最小二乘

满秩的正方矩阵存在逆, 一个很自然的问题就是, 对于不满秩的矩阵或者非正方矩阵, 是不是可以定义类似的矩阵逆?

3.6.1 广义逆

广义逆的概念最早由 Moore [80] 于 1920 年提出, 他给出的定义如下: 设 $A \in \mathbb{C}^{m \times n}$, 若 $X \in \mathbb{C}^{n \times m}$ 满足

$$AX = P_{\text{Ran}(A)}, \quad XA = P_{\text{Ran}(X)}, \quad (3.23)$$

即 AX 和 XA 分别为 $\text{Ran}(A)$ 和 $\text{Ran}(X)$ 上的正交投影算子, 则称 X 是 A 的广义逆.

1955 年, Penrose [88] 利用下面四个矩阵方程给出了广义逆的另一种定义, 这也是当前常用的广义逆定义方式.

定义 3.2 (广义逆) 设 $A \in \mathbb{C}^{m \times n}$, 若 $X \in \mathbb{C}^{n \times m}$ 满足

$$AXA = A \quad (3.24)$$

$$XAX = X \quad (3.25)$$

$$(AX)^* = AX \quad (3.26)$$

$$(XA)^* = XA. \quad (3.27)$$

则称 X 为 A 的**广义逆**, 记为 $A^\dagger$.

方程组 (3.23) 和 (3.24)-(3.27) 分别称为 Moore 方程组和 Penrose 方程组. 可以证明, 以上两种定义是等价的, 因此广义逆也称为 **Moore-Penrose 逆**, 简称 **MP 逆**.

-  (1) 需要指出的是, 广义逆对所有矩阵都有定义, 并不要求是正方矩阵, 也不要求满秩.
- (2) 若 $A \in \mathbb{C}^{n \times n}$ 非奇异, 则 $A^\dagger = A^{-1}$.
- (3) 在有些文献中, 广义逆也称为**伪逆**.

定理 3.25 [130] 设 $A \in \mathbb{C}^{m \times n}$, 则满足矩阵方程组 (3.24)-(3.27) 的矩阵 $X \in \mathbb{C}^{n \times m}$ 存在且唯一, 即矩阵广义逆存在唯一. (板书)

3.6.2 广义逆基本性质

定理 3.26 设 $A \in \mathbb{C}^{m \times n}$, 则

- (1) $(A^\dagger)^\dagger = A$;
- (2) $(A^T)^\dagger = (A^\dagger)^T$, $(A^*)^\dagger = (A^\dagger)^*$;
- (3) $\text{rank}(A) = \text{rank}(A^\dagger) = \text{rank}(A^\dagger A)$;
- (4) $(AA^*)^\dagger = (A^*)^\dagger A^\dagger$, $(A^*A)^\dagger = A^\dagger (A^*)^\dagger$;



- (5) $(AA^*)^\dagger AA^* = AA^\dagger$, $(A^*A)^\dagger A^*A = A^\dagger A$;
 (6) $A^\dagger = (A^*A)^\dagger A^* = A^*(AA^*)^\dagger$,
 特别地, 若 A 列满秩, 则 $A^\dagger = (A^*A)^{-1}A^*$, 若 A 行满秩, 则 $A^\dagger = A^*(AA^*)^{-1}$;
 (7) 若 U, V 是酉矩阵, 则 $(UAV)^\dagger = V^*A^\dagger U^*$.

 一般来说, 当 A, B 是正方形时,

- $(AB)^\dagger \neq B^\dagger A^\dagger$;
- $AA^\dagger \neq A^\dagger A$;
- $(A^k)^\dagger \neq (A^\dagger)^k$;
- A 和 $A^\dagger$ 的非零特征值并不是互为倒数.

定理 3.27 设 $A \in \mathbb{C}^{m \times n}$, 则

$$\begin{aligned} \text{Ran}(AA^\dagger) &= \text{Ran}(AA^*) = \text{Ran}(A), \\ \text{Ran}(A^\dagger A) &= \text{Ran}(A^*A) = \text{Ran}(A^*) = \text{Ran}(A^\dagger), \\ \text{Ker}(AA^\dagger) &= \text{Ker}(AA^*) = \text{Ker}(A^*) = \text{Ker}(A^\dagger), \\ \text{Ker}(A^\dagger A) &= \text{Ker}(A^*A) = \text{Ker}(A). \end{aligned}$$

推论 3.28 (广义逆与正交投影变换) 设 $A \in \mathbb{C}^{m \times n}$, 则

$$P_A = AA^\dagger, \quad P_{A^*} = A^\dagger A,$$

其中 P_A 和 P_{A^*} 分别表示 $\text{Ran}(A)$ 和 $\text{Ran}(A^*)$ 上的正交投影变换.

特别地, 如果 $A \in \mathbb{C}^{n \times n}$ 是正交投影矩阵, 则

$$A^\dagger = A.$$

3.6.3 广义逆的计算

利用 SVD

我们可以利用 A 的奇异值分解来计算 $A^\dagger$, 但运算量较大. 因为奇异值分解通常与 AA^* 或 A^*A 的特征值分解有关.

利用满秩分解

定理 3.29 设 $A \in \mathbb{C}^{m \times n}$.

- (1) 若 A 是列满秩矩阵, 则 $A^\dagger = (A^*A)^{-1}A^*$; 特别地, 若 Q 单位列正交, 则 $Q^\dagger = Q^*$;
- (2) 若 A 是行满秩矩阵, 则 $A^\dagger = A^*(AA^*)^{-1}$; 特别地, 若 Q 单位行正交, 则 $Q^\dagger = Q^*$;



(3) 若 A 的秩是 $r \leq \min\{m, n\}$, 且其满秩分解为 $A = FG$, 其中 $F \in \mathbb{C}^{m \times r}$, $G \in \mathbb{C}^{r \times n}$, 则

$$A^\dagger = G^\dagger F^\dagger = G^*(GG^*)^{-1}(F^*F)^{-1}F^*.$$

利用 QR 分解

定理 3.30 设 $A \in \mathbb{C}^{m \times n}$ ($m \geq n$) 是列满秩矩阵, 其 QR 分解为 $A = QR$, 其中 $Q \in \mathbb{C}^{m \times n}$, $R \in \mathbb{C}^{n \times n}$, 则

$$A^\dagger = R^{-1}Q^*.$$

 若 A 不是满秩的, 则如何利用列主元 QR 分解给出 $A^\dagger$ 的表达式?

其他算法

其他比较重要的算法有: **Greville 递推算法**, **Cline 算法**等.

3.6.4 广义逆与线性最小二乘

定理 3.31 [130] 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), 则线性最小二乘问题 (3.1) 的解为

$$x = A^\dagger b + (I - P_{\text{Ran}(A^\top)})z, \quad \forall z \in \mathbb{R}^n. \quad (3.28)$$

 由上述定理结论可知, 线性最小二乘问题的解 (3.28) 不唯一. 但当 A 列满秩时, $P_{\text{Ran}(A^\top)} = I$, 此时解唯一.

当线性最小二乘问题存在多个解时, 我们通常计算满足一定条件的解, 比较常见的是最小范数解.

定理 3.32 [130] 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$). 记线性最小二乘问题 (3.1) 的解集为 S , 则

$$\min_{x \in S} \|x\|_2 \quad (3.29)$$

存在唯一解, 即线性最小二乘问题 (3.1) 存在唯一的最小范数解.

3.6.5 广义逆与条件数

我们在定义矩阵的条件数 (2.17) 时, 要求矩阵是方阵. 但在最小二乘问题中, 矩阵通常不是方阵, 此时我们可以通过广义逆来定义其条件数.

设 $A \in \mathbb{R}^{m \times n}$, 其条件数可定义为

$$\kappa(A) = \|A\| \|A^\dagger\|.$$

如果 $m = n$ 且 A 非奇异, 则以上定义就是通常的矩阵条件数. 如果 A 列满秩, 则 $A^\dagger = (A^\top A)^{-1}A^\top$, 如果 A 行满秩, 则 $A^\dagger = A^\top(AA^\top)^{-1}$.



3.6.6 左逆和右逆

在工程计算中, 有时也会用到矩阵的左逆和右逆.

定义 3.3 设 $A \in \mathbb{C}^{m \times n}$, 如果存在矩阵 $A_{\text{left}}^{-1} \in \mathbb{C}^{n \times m}$ 使得 $A_{\text{left}}^{-1}A = I_n$, 则称 A_{left}^{-1} 是 A 的**左逆**. 类似地, 如果存在矩阵 $A_{\text{right}}^{-1} \in \mathbb{C}^{n \times m}$ 使得 $AA_{\text{right}}^{-1} = I_m$, 则称 A_{right}^{-1} 是 A 的**右逆**.

定理 3.33 设 $A \in \mathbb{C}^{m \times n}$, 则 A 存在左逆的充要条件是 A 列满秩; A 存在右逆的充要条件是 A 行满秩.

例 3.8 左逆和右逆一般是不唯一的. 比如

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \text{则} \quad A_{\text{left}}^{-1} = \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \end{bmatrix}, \quad \text{其中 } x, y \text{ 可以取任意值.}$$

3.7 最小二乘扰动分析

定理 3.34 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 且 $\text{rank}(A) = n$. 设 x 是线性最小二乘问题 (3.1) 的解, $\tilde{x}$ 极小化 $\|(A + \delta A)\tilde{x} - (b + \delta b)\|_2$, 则

$$\frac{\|\tilde{x} - x\|_2}{\|x\|_2} \leq \varepsilon \cdot \left\{ \frac{2\kappa_2(A)}{\cos \theta} + \kappa_2^2(A) \tan \theta \right\} + \mathcal{O}(\varepsilon^2),$$

其中 $\kappa_2(A) = \sigma_1(A)/\sigma_n(A)$, θ 为 b 与 $\text{Ran}(A)$ 的夹角,

$$\varepsilon \triangleq \max \left\{ \frac{\|\delta A\|_2}{\|A\|_2}, \frac{\|\delta b\|_2}{\|b\|_2} \right\},$$

并假定 $\varepsilon \cdot \kappa_2(A) < 1$ (确保 $A + \delta A$ 满秩, 从而 $\tilde{x}$ 唯一确定).

我们记

$$\kappa_{LS} \triangleq \frac{2\kappa_2(A)}{\cos \theta} + \kappa_2^2(A) \tan \theta,$$

这就是最小二乘问题的条件数. 当 $\theta = 0$ 时, $b \in \text{Ran}(A)$, 此时 $\kappa_{LS} = 2\kappa_2(A)$; 当 $\theta = \pi/2$ 时, $b \perp \text{Ran}(A)$, 此时最小二乘解为 $x = 0$, 而 $\kappa_{LS} = \infty$; 当 $0 < \theta < \pi/2$ 时, $\kappa_{LS} = \mathcal{O}(\kappa_2^2(A))$.

定义残量 $r = b - Ax$, $\tilde{r} = (b + \delta b) - (A + \delta A)\tilde{x}$, 我们有下面的性质 [61]

$$\frac{\|\tilde{r} - r\|_2}{\|r\|_2} \leq \varepsilon \cdot (1 + 2\kappa_2(A)).$$

当我们使用 QR 分解或 SVD 分解求解最小二乘问题时, 由于采用的是正交变换, 它们都是数值稳定的. 而正规方程涉及求解方程组 $A^T Ax = A^T b$, 其精度依赖于条件数 $\kappa_2(A^T A) = \kappa_2^2(A)$, 因为其误差是以 $\kappa_2^2(A)$ 倍数增长. 因此当 A 的条件数较大时, 正规方程法的精度会大大降低.



3.8 推广与应用

3.8.1 最小二乘问题的推广

正则化

在求解超定线性方程组时, 我们极小化 $\|Ax - b\|_2^2$. 而对于欠定线性方程组, 由于解不唯一, 我们往往还需要极小化 $\|x\|_2^2$. 两者合起来就是

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|_2^2 + \frac{\alpha}{2} \|x\|_2^2, \quad (3.30)$$

其中 $\alpha > 0$ 是正则化参数. 对应的目标函数记为

$$J(x) = \|Ax - b\|_2^2 + \alpha \|x\|_2^2.$$

当 $\alpha > 0$ 时, $J(x)$ 是一个严格凸的二次函数, 因此存在唯一的最小值点. 令 $J(x)$ 关于 x 的一阶导数为零, 则可得

$$(A^T A + \alpha I)x = A^T b.$$

由于 $A^T A + \alpha I$ 对称正定, 故非奇异. 所以问题 (3.30) 的唯一解为

$$x = (A^T A + \alpha I)^{-1} A^T b.$$

加权正则化

一类应用更广泛的问题是下面的加权正则化问题

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|_2^2 + \frac{\alpha}{2} \|Wx\|_2^2, \quad (3.31)$$

其中 $\alpha > 0$ 是正则化参数, $W \in \mathbb{R}^{m \times n}$ 是广义加权矩阵, 可以是非负对角矩阵, 对称正定矩阵, 也可以是一般矩阵 (如一阶差分算子, 二阶差分算子等). 需要指出的是, W 不一定要是方阵.

经过类似的推导, 可知问题 (3.31) 的解满足

$$(A^T A + \alpha W^T W)x = A^T b.$$

如果 $A^T A + \alpha W^T W$ 非奇异, 则存在唯一解

$$x = (A^T A + \alpha W^T W)^{-1} A^T b.$$

约束最小二乘问题

考虑带有约束的最小二乘问题

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|_2^2 \quad (3.32)$$

$$\text{s.t. } Bx = f$$

其中 $Bx = f$ 是约束条件, $B \in \mathbb{R}^{p \times n}$ 和 $f \in \mathbb{R}^p$ 都是给定的. 对应的 Lagrange 函数为

$$J(x) = \frac{1}{2} \|Ax - b\|_2^2 + \lambda^T (Bx - f).$$



分别对 x 和 λ 求一阶导数, 并令其等于零, 可得

$$\begin{bmatrix} A^T A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} A^T b \\ f \end{bmatrix}.$$

如果 $A^T A$ 非奇异, 且 B 行满秩, 则存在唯一解

$$\begin{aligned} \lambda &= [B(A^T A)^{-1} B^T]^{-1} [B(A^T A)^{-1} A^T b - f] \\ x &= (A^T A)^{-1} (A^T b - B^T \lambda). \end{aligned}$$

3.8.2 最小二乘问题的应用

多项式数据拟合

最小二乘的一个重要应用就是低次多项式数据拟合: 已知平面上 n 个点 $\{(t_i, f_i)\}_{i=1}^n$, 寻找一个低次多项式来拟合这些数据.

设拟合多项式为

$$p(x) = a_0 + a_1 t + a_2 t^2 + \cdots + a_m t^m.$$

通常 $m \ll n$. 将上述 n 个点 $\{(t_i, f_i)\}_{i=1}^n$ 代入可得

$$\begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^m \\ 1 & t_2 & t_2^2 & \cdots & t_2^m \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & t_n & t_n^2 & \cdots & t_n^m \end{bmatrix}_{n \times (m+1)} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \quad \text{或} \quad Ax = f,$$

其中 $A \in \mathbb{R}^{n \times (m+1)}$, $x = [a_0, a_1, \dots, a_m]^T$. 由于 $m \ll n$, 该方程组是超定的, 解通常是不存在的. 因此, 我们寻找一个近似解, 使得残量 $\|f - Ax\|_2$ 最小, 即求解最小二乘问题

$$\min_{x \in \mathbb{R}^{m+1}} \|f - Ax\|_2^2$$

线性预测

预测一个时间序列的未来走向, 其中一个常用方法就是线性预测 (linear prediction). 假定一个时间序列在 t_k 时刻的值 f_k 线性依赖于其前 m 个时刻的值 $f_{k-1}, f_{k-2}, \dots, f_{k-m}$, 即

$$f_k = a_1 f_{k-1} + a_2 f_{k-2} + \cdots + a_m f_{k-m}. \quad (3.33)$$

现在已经测得该时间序列的前 n 个值 $f_i, i = 0, 1, 2, \dots, n-1$, 如何预测其未来的取值. 这里 $n \gg m$. 将现有的数据代入关系式 (3.33) 可得

$$\begin{bmatrix} f_{m-1} & f_{m-2} & f_{m-3} & \cdots & f_0 \\ f_m & f_{m-1} & f_{m-2} & \cdots & f_1 \\ \vdots & \vdots & \vdots & & \vdots \\ f_{n-2} & f_{n-3} & f_{n-4} & \cdots & f_{n-m-1} \end{bmatrix}_{(n-m) \times m} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} f_m \\ f_{m+1} \\ \vdots \\ f_{n-1} \end{bmatrix} \quad \text{或} \quad Ax = f,$$



其中 $A \in \mathbb{R}^{(n-m) \times m}$, $x = [a_1, a_2, \dots, a_m]^\top$. 这也是一个超定问题, 其解也是通过求解下面的最小二乘问题来获取

$$\min_{x \in \mathbb{R}^m} \|f - Ax\|_2^2$$

信号恢复

在获取数字信号时, 由于各种各样的原因, 最后得到的信号总会带有一定的噪声, 即

$$b = x + \eta,$$

其中 x 是真实的信号, b 是观察到的信号, η 是噪声.

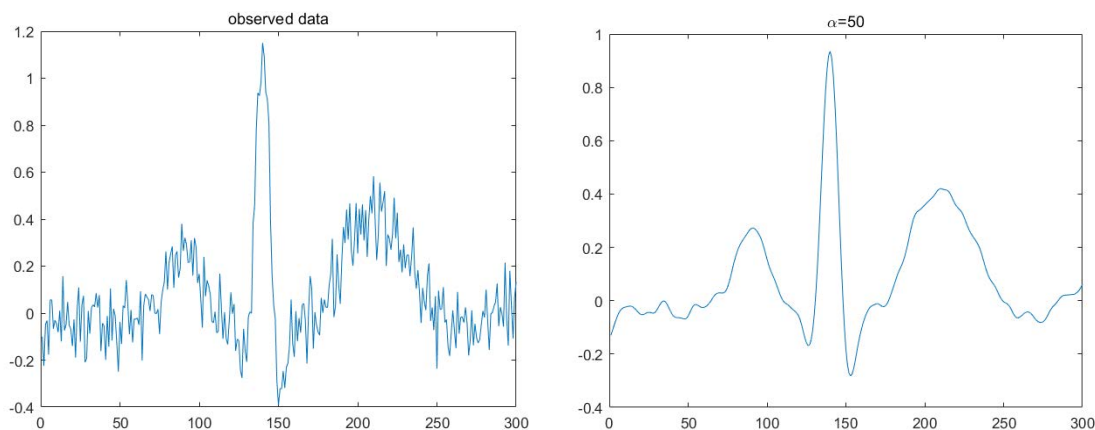
去噪是数字信号和图像处理中的一个基本问题, 其中一个有效方法就是加权最小二乘法, 即

$$\min_x \frac{1}{2} \|x - b\|_2^2 + \frac{1}{2} \alpha \|Dx\|_2^2,$$

其中 D 是离散的二阶导算子或 TV 算子.

例 3.9 数字信号去噪.

(LS_denoise.m)



图像恢复

除了带噪声以外, 在获取数字图像时也经常会由于各种原因 (设备仪器, 拍摄环境等) 导致图像模糊. 通常数字图像的获取模型为

$$f = \mathcal{B}(x) + \eta,$$

其中 x 是真实图像, f 是观察到的图像, $\mathcal{B}(\cdot)$ 是卷积算子 (代表模糊机制), η 是噪声.

由于问题本身是不适定的, 因此求解时需要进行正则化, 常用的模型有 **Tikhonov 正则化**模型:

$$\min \| \mathcal{B}(x) - f \|_2^2 + \mu^2 \|x\|_2^2$$



或加权正则化模型:

$$\min \|B(x) - f\|_2^2 + \mu^2 \|Dx\|_2^2,$$

其中 D 是广义加权矩阵, 可以是非负对角矩阵或 TV 算子等.

3.9 课后习题

练习 3.1 设 $A \in \mathbb{R}^{m \times n}$, 其中 $m \leq n$. 试证明: 矩阵 $\begin{bmatrix} I & A^T \\ A & 0 \end{bmatrix}$ 非奇异的充要条件是 $\text{rank}(A) = m$.

练习 3.2 设 $B \in \mathbb{R}^{m \times m}$ 对称半正定, $A \in \mathbb{R}^{m \times n}$, 其中 $m \geq n$. 试证明: 矩阵 $\begin{bmatrix} B & A \\ A^T & 0 \end{bmatrix}$ 非奇异的充要条件是 A 列满秩且矩阵 $[B, A]$ 行满秩 (即 A 列满秩且 $\text{Ker}(B) \cap \text{Ker}(A^T) = \{0\}$).

练习 3.3 设 $\tau \neq 0$, 向量 $u, v \in \mathbb{R}^n$ 均不为零, 求矩阵 $E(u, v, \tau) = I - \tau uv^T$ 的特征值.

练习 3.4 (Sylvester 降幂公式 (3.4)) 设 $A \in \mathbb{C}^{m \times n}$, $B \in \mathbb{C}^{n \times m}$, 其中 $m \geq n$, 则有

$$\det(\lambda I - AB) = \lambda^{m-n} \det(\lambda I - BA).$$

(因此 AB 和 BA 具有相同的非零特征值)

练习 3.5 设 $A \in \mathbb{R}^{n \times n}$ 是正交矩阵, 则 A 可表示成不超过 n 个 Householder 变换的乘积.

练习 3.6 设 $G = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \in \mathbb{R}^{2 \times 2}$ 是 Givens 变换, 由练习 3.5 可知, G 可以表示为 2 个 Householder 变换的乘积, 试给出这两个 Householder 变换所对应的 Householder 向量.

练习 3.7 设 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$ 是一个非零向量, H 是 Householder 矩阵, 满足 $Hx = \alpha e_1$.

证明: H 的第一列与 x 平行.

(注: 该结论提供了一种构造指定第一列的正交矩阵或酉矩阵的实用方法)

练习 3.8 设 $H_k = I - \beta_k v_k v_k^T \in \mathbb{R}^{k \times k}$ 是 Householder 变换, 其中 $k < n$.

证明: $H_n = \begin{bmatrix} I_{n-k} & 0 \\ 0 & H_k \end{bmatrix}$ 是 n 阶 Householder 变换, 并写出对应的 Householder 向量.

练习 3.9 设 $R = \begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{bmatrix} \in \mathbb{R}^{2 \times 2}$, 求一个 Givens 变换 G , 使得 $G^T R G = \begin{bmatrix} r_{22} & r_{12} \\ 0 & r_{11} \end{bmatrix}$.

练习 3.10 设 $R \in \mathbb{R}^{n \times n}$ 是一个上三角矩阵, 且对角线元素互不相同. 证明: 存在正交矩阵 Q , 使得 $Q^T R Q$ 为上三角矩阵, 且对角线元素为 R 的对角线元素的降序排列.

(注: 该结论可以看作是练习 3.9 的推广)

练习 3.11 利用数学归纳法证明 QR 分解的存在性.

练习 3.12 设 $A \in \mathbb{R}^{n \times n}$, 证明: $\|A\|_2 = \max_{x \neq 0, y \neq 0} \frac{y^T A x}{\|x\|_2 \|y\|_2}$.



练习 3.13* 设 A 的奇异值分解为 $A = U_n \Sigma V^* = \sum_{i=1}^n \sigma_i u_i v_i^* \in \mathbb{C}^{m \times n}$, 证明

$$\min_{B \in \mathbb{C}^{m \times n}, \text{rank}(B)=k} \|A - B\|_F = \|A - A_k\|_F = \sqrt{\sigma_{k+1}^2 + \sigma_{k+2}^2 + \cdots + \sigma_n^2},$$

其中 $A_k = \sum_{i=1}^k \sigma_i u_i v_i^*$, $1 \leq k \leq \text{rank}(A)$. (提示: 利用定理 3.17)

练习 3.14 设 $A \in \mathbb{C}^{n \times n}$ 有 n 个互不相同的非零特征值, 其 Schur 分解为 $A = URU^*$, 其中 U 为酉矩阵, R 为上三角矩阵. 设矩阵 $B \in \mathbb{C}^{n \times n}$ 满足 $AB = BA$. 证明: U^*BU 是上三角矩阵.

练习 3.15 用 Householder 变换计算矩阵 $A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & -1 & -1 \\ 2 & -4 & 10 \end{bmatrix}$ 的 QR 分解.

练习 3.16 求解最小二乘问题: $\min \|Ax - b\|_2$, 其中 $A = \begin{bmatrix} 2 & 0 \\ -2 & 3 \\ 0 & -1 \\ 1 & -2 \end{bmatrix}$, $b = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 2 \end{bmatrix}$.

练习 3.17* 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 且 $\text{rank}(A) = n$. 计算矩阵 $\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix}$ 的谱条件数. (用 A 的奇异值表示)

..... 以下为可选题

练习 3.18 设 $A = [a_1, a_2, \dots, a_n] \in \mathbb{R}^{m \times n}$, 下面是另外一种修正 Gram-Schmidt 正交化过程 [49], 试指出与 Gram-Schmidt 正交化过程的区别.

```

1: for  $i = 1$  to  $n$  do
2:    $r_{ii} = \|a_i\|_2$ 
3:    $q_i = a_i / r_{ii}$ 
4:   for  $j = i + 1$  to  $n$  do
5:      $r_{ij} = q_i^T a_j$ ,  $a_j = a_j - r_{ij} q_i$ 
6:   end for
7: end for

```

练习 3.19 设 $A \in \mathbb{R}^{m \times n}$, 证明: $\text{Ran}(A^T) = \text{Ran}(A^T A)$.

练习 3.20 试证明三类初等矩阵都可以写成 (3.3) 中的形式. (以第 74 页上的 E_1, E_2, E_3 为例)

练习 3.21 证明定理 3.1 中的结论 (3), 即:

对任意非零向量 $x, y \in \mathbb{C}^n$, 存在 $u, v \in \mathbb{C}^n$ 和 $\tau \in \mathbb{C}$, 使得

$$E(u, v, \tau)x = y.$$

练习 3.22 设 $A \in \mathbb{R}^{n \times n}$ 是正交矩阵, 则 A 可表示成至多 $\frac{1}{2}n(n-1)$ 个 Givens 变换和 1 个对角线为 ± 1 的对角矩阵的乘积. (更进一步, 该对角矩阵除了最后一个对角线元素是 ± 1 外, 其他对角线元素都是 1)



练习 3.23* 试给出定理 3.5 在复数空间上的相对应的结论, 并证明.

练习 3.24 证明定理 3.13 和定理 3.14. (奇异值的相关性质)

练习 3.25* 证明奇异值的最小最大定理, 即定理 3.18.

练习 3.26 证明奇异值的扰动性质, 即定理 3.22.

练习 3.27 试给出加权正则化问题 (3.31) 存在唯一解的充要条件.

练习 3.28* 设 $A \in \mathbb{R}^{n \times n}$ 是一个对角加边矩阵

$$A = \begin{bmatrix} a_1 & b_2 & b_3 & \cdots & b_n \\ c_2 & a_2 & & & \\ c_3 & & a_3 & & \\ \vdots & & & \ddots & \\ c_n & & & & a_n \end{bmatrix}.$$

试给出用 Givens 变换计算 A 的 QR 分解的详细算法, 使得运算量为 $\mathcal{O}(n^2)$.

练习 3.29* 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), 如何求解下面的数据拟合问题

$$\min_{x \in \mathbb{R}^n} \|b - Ax\|_1.$$

练习 3.30* 设 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{C}^n$ 是一个非零复向量, 问: 是否存在 Householder 矩阵 $H(v)$ 使得 $H(v)x = \alpha e_1$? 若存在, 如何计算 v ? (提示: 这里 α 可以是复数)

练习 3.31* 如果矩阵在内存中是按行排列的, 怎么实现 QR 分解比较高效?

练习 3.32 证明: 分别由 Moore 方程组 (3.23) 和 Penrose 方程组 (3.24)-(3.27) 定义的广义逆是等价的.

练习 3.33 设 $A = FG$ 是 A 的满秩分解, 其中 $A \in \mathbb{R}^{m \times n}$, $F \in \mathbb{R}^{m \times r}$, $G \in \mathbb{R}^{r \times n}$, $\text{rank}(A) = r$, $0 \leq r \leq \min\{m, n\}$. 证明: $A^\dagger = G^\dagger F^\dagger$.

练习 3.34* 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), 如果 A 不是满秩的, 如何求解相应的最小二乘问题?

练习 3.35 设 A, B 是正方形, 举例说明:

- $(AB)^\dagger \neq B^\dagger A^\dagger$;
- $AA^\dagger \neq A^\dagger A$;
- $(A^k)^\dagger \neq (A^\dagger)^k$;
- A 和 $A^\dagger$ 的非零特征值并不是互为倒数.

..... 以下为实践题

练习 3.36 写出按行 MGS 算法过程, 并编程实现.

练习 3.37 编写 Householder 变换函数: 对任意给定的变量 $x \in \mathbb{R}^n$, 输出 v 使得 $H(v)x = \|x\|_2 e_1$, 其中 $H(v) = I - 2vv^T$, 函数原型: **v=House2(x)**

练习 3.38 编写 Householder 变换函数: 对任意给定的变量 $x \in \mathbb{R}^n$, 输出 β 和 v 使得 $H(v)x =$



$\|x\|_2 e_1$, 其中 $H(v) = I - \beta vv^T$ 且 v 的第一个分量为 1. 函数原型: `[beta,v]=House3(x)`

练习 3.39 编写基于 Householder 变换的 QR 分解, 函数原型: `[Q,R]=QR_Householder(A)`

练习 3.40 设 $A = \begin{bmatrix} R \\ S \end{bmatrix}$, 其中 $R \in \mathbb{R}^{n \times n}$ 是上三角矩阵, $S \in \mathbb{R}^{m \times n}$ 是稠密矩阵, 试描述 A 的基于 Householder 变换的 QR 算法. 要求算法过程中始终保持 R 中的零元.

练习 3.41 设 $A = R + uv^T$, 其中 $R \in \mathbb{R}^{n \times n}$ 是上三角矩阵, $u, v \in \mathbb{R}^n$ 为非零列向量, 试给出计算 A 的 QR 分解的实用算法, 要求总运算量为 $\mathcal{O}(n^2)$. (提示: 使用 Givens 变换)

练习 3.42 设 $B = A + uv^T$, 其中 $A \in \mathbb{R}^{n \times n}$, $u, v \in \mathbb{R}^n$. 假定 A 的 QR 分解已知, 试给出计算 A 的 QR 分解算法, 要求总运算量大约为 $\mathcal{O}(n^2)$.

练习 3.43 构造并实现列主元 QR 分解的算法. (选列主元过程中, 在计算 $R_{22}^{(k-1)}$ 各列的范数时, 可利用前一步选主元时所计算的范数, 这样能减少计算量, 可参见 [49])

练习 3.44 构造并实现矩阵的 LQ 分解算法, 即 $A = LQ$, 其中 L 为下三角矩阵, Q 为正交矩阵.



第四讲 非对称特征值问题

设 A 是一个非对称的稠密矩阵, 本讲主要讨论如何计算 A 的全部特征值和特征向量. 为了讨论方便, 本讲同样只考虑实矩阵情形.

记 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$. 本讲中我们总是假定

$$|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n| \geq 0,$$

即 A 的特征值按模降序排列.

关于稠密矩阵特征值计算的相关参考资料

- ▶ J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, 1965. [115] (有中文翻译)
- ▶ B. N. Parlett, *The Symmetric Eigenvalue Problem*, 2nd Eds., 1998. [85]
- ▶ G. W. Stewart, *Matrix Algorithms, Vol II: Eigensystems*, 2001. [100]
- ▶ G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2013. [49]
- ▶ Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, 2000. [9]
- ▷ G. H. Golub and H. A. van der Vorst, [Eigenvalue computation in the 20th century](#), 2000. [48]

4.1 幂迭代法

幂迭代法是计算特征值和特征向量的一种简单易用的算法. 幂迭代法虽然简单, 但它却建立了计算特征值和特征向量的算法的一个基本框架.

4.1.1 算法介绍

任意给定一个非零向量 $x^{(0)}$, 不断用 A 左乘该向量, 得到向量序列:

$$x^{(0)}, Ax^{(0)}, A^2x^{(0)}, A^3x^{(0)}, \dots$$

在一定条件下, 可以证明该向量序列收敛到模最大特征值 λ_1 所对应的特征向量, 然后通过计算 Rayleigh 商 (见定义 (5.10)) 就可以得到 λ_1 . 这就是幂迭代法, 算法描述如下 (为了防止溢出, 算法中对向量进行了单位化处理).

算法 4.1 幂迭代法 (Power Iteration)

- 1: Choose an initial guess $x^{(0)}$ with $\|x^{(0)}\|_2 = 1$
- 2: set $k = 0$

```

3: while not convergence do
4:    $y^{(k+1)} = Ax^{(k)}$ 
5:    $x^{(k+1)} = y^{(k+1)} / \|y^{(k+1)}\|_2$    % 单位化, 防止溢出
6:    $\mu_{k+1} = (Ax^{(k+1)}, x^{(k+1)})$    % 内积, 计算 Rayleigh 商
7:    $k = k + 1$ 
8: end while

```

4.1.2 收敛性分析

下面讨论幂迭代法的收敛性. 我们假定

- (1) $A \in \mathbb{R}^{n \times n}$ 是可对角化的, 即 $A = V\Lambda V^{-1}$, 其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) \in \mathbb{C}^{n \times n}$, $V = [v_1, v_2, \dots, v_n] \in \mathbb{C}^{n \times n}$, 且 $\|v_i\|_2 = 1$, $i = 1, 2, \dots, n$.
- (2) $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$.

由于 $V \in \mathbb{C}^{n \times n}$ 非奇异, 所以它的列向量组构成 $\mathbb{C}^n$ 的一组基. 因此迭代初始向量 $x^{(0)}$ 可表示为

$$x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n = V[\alpha_1, \alpha_2, \dots, \alpha_n]^T.$$

我们假定 $\alpha_1 \neq 0$, 即 $x^{(0)}$ 不属于 $\text{span}\{v_2, v_3, \dots, v_n\}$ (由于 $x^{(0)}$ 是随机选取的, 从概率意义上讲, 这个假设通常是成立的). 于是我们可得

$$A^k x^{(0)} = (V\Lambda V^{-1})^k V \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = V \Lambda^k \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = V \begin{bmatrix} \alpha_1 \lambda_1^k \\ \alpha_2 \lambda_2^k \\ \vdots \\ \alpha_n \lambda_n^k \end{bmatrix} = \alpha_1 \lambda_1^k V \begin{bmatrix} 1 \\ \frac{\alpha_2}{\alpha_1} \left(\frac{\lambda_2}{\lambda_1}\right)^k \\ \vdots \\ \frac{\alpha_n}{\alpha_1} \left(\frac{\lambda_n}{\lambda_1}\right)^k \end{bmatrix}.$$

又 $|\lambda_i/\lambda_1| < 1$, $i = 2, 3, \dots, n$, 所以

$$\lim_{k \rightarrow \infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k = 0, \quad i = 2, 3, \dots, n.$$

故当 k 趋向于无穷大时, 向量

$$\left[1, \frac{\alpha_2}{\alpha_1} \left(\frac{\lambda_2}{\lambda_1}\right)^k, \dots, \frac{\alpha_n}{\alpha_1} \left(\frac{\lambda_n}{\lambda_1}\right)^k \right]^T, \quad k = 0, 1, 2, \dots$$

收敛到 $e_1 = [1, 0, \dots, 0]^T$. 所以向量 $x^{(k)} = A^k x^{(0)} / \|A^k x^{(0)}\|_2$ 收敛到 $\pm v_1$, 即 A 的对应于 (模) 最大的特征值 λ_1 的特征向量. 而 $\mu_k = (Ax^{(k)}, x^{(k)})$ 则收敛到 $v_1^* A v_1 = \lambda_1$.

显然, 幂迭代的收敛快慢取决于 $|\lambda_2/\lambda_1|$ 的大小, $|\lambda_2/\lambda_1|$ 越小, 收敛越快.

结论

- (1) $x^{(k)}$ 收敛到 $\pm v_1$; (2) μ_k 收敛到 λ_1 ; (3) $|\lambda_2/\lambda_1|$ 越小, 收敛越快.

通过上面的分析可知, 幂迭代只能用于计算矩阵的模最大的特征值和其相应的特征向量. 如果 A 的模最大的特征值是唯一的, 则称其为**主特征值**. 当 $|\lambda_2/\lambda_1|$ 接近于 1 时, 收敛速度会非常慢. 同时, 如果 A 的模最大特征值不唯一, 比如一对共轭复特征值, 则幂迭代就可能就会失效.



 **思考：**如果 A 不可以对角化，则能否得到类似的结论？

 如果需要计算其他特征值，比如模第二大特征值 λ_2 ，则可以在模最大特征值 λ_1 计算出来后，采用 **收缩 (Deflation)** 技术：构造酉矩阵 U ，使得

$$U^*AU = \begin{bmatrix} \lambda_1 & A_{12} \\ 0 & A_{22} \end{bmatrix}.$$

然后将幂迭代作用到 A_{22} 上，就可以求出 λ_2 。以此类推，可以依次求出所有特征值（这里假定特征值互不相同）。

 **思考：**上面的收缩技术中的 U 怎么选取？

4.1.3 位移策略

前面已经提到，幂迭代法的收敛速度取决于 $|\lambda_2/\lambda_1|$ 的大小。当它的值接近于 1 时，收敛速度会非常缓慢。因此，为了加快幂迭代法的收敛速度，我们希望 $|\lambda_2/\lambda_1|$ 的值越小越好。

一个简单易用的方法就是使用**位移策略**，即将计算 A 的特征值转化为计算 $A - \sigma I$ 的特征值，即对 A 做一个移位。这里 σ 是一个给定的数，称 σ 为**位移** (shift)。为了使得幂迭代作用到 $A - \sigma I$ 时具有更快的收敛速度，我们要求 σ 满足下面的两个条件：

(1) $\lambda_1 - \sigma$ 是 $A - \sigma I$ 的模最大的特征值；

(2) $\max_{2 \leq i \leq n} \left| \frac{\lambda_i - \sigma}{\lambda_1 - \sigma} \right|$ 尽可能地小。

其中第一个条件保证最后所求得特征值是我们所要的，第二个条件用于加快幂迭代的收敛速度。

显然，在实际应用中， σ 的取值并不是一件容易的事。

例 4.1 设 $A = X\Lambda X^{-1}$ ，其中 Λ 为对角矩阵，分别用幂迭代法和带位移的幂迭代法计算 A 的主特征值。
(见 `Eig_Power_shift.m`)

 位移策略在特征值计算中非常重要，特别是在反迭代法和 QR 迭代法中。

4.2 反迭代法

如果我们将幂迭代法作用在 A^{-1} 上，则可求出 A 的模最小的特征值，此时的幂迭代法称为**反迭代法**。



4.2.1 算法介绍

事实上, 结合位移策略, 我们就可以计算矩阵的任意一个特征值. 下面给出带位移反迭代法的算法描述.

算法 4.2 带位移的反迭代法 (Inverse Iteration)

```

1: Choose a scalar  $\sigma$  and an initial vector  $x^{(0)}$  with  $\|x^{(0)}\|_2 = 1$ 
2: set  $k = 0$ 
3: while not convergence do
4:    $y^{(k+1)} = (A - \sigma I)^{-1}x^{(k)}$ 
5:    $x^{(k+1)} = y^{(k+1)} / \|y^{(k+1)}\|_2$ 
6:    $\mu_{k+1} = (Ax^{(k+1)}, x^{(k+1)})$ 
7:    $k = k + 1$ 
8: end while
  
```

显然, 在反迭代法中, μ_k 收敛到距离 σ 最近的特征值, 而 $x^{(k)}$ 则收敛到其对应的特征向量. 设距离 σ 最近的特征值为 λ_k , 则算法的收敛速度取决于

$$\max_{1 \leq i \leq n, i \neq k} \left| \frac{\lambda_k - \sigma}{\lambda_i - \sigma} \right|$$

的大小. 显然, σ 越接近于 λ_k , 其值越小, 即算法收敛越快. 若 $\sigma \approx \lambda_k$, 则迭代几步就可以了.

反迭代法的另一个优点是, 只要选取合适的位移 σ , 就可以计算 A 的任意一个特征值.

但反迭代法的缺点也很明显: 每步迭代需要解一个线性方程组 $(A - \sigma I)y^{(k+1)} = x^{(k)}$, 这就需要对 $A - \sigma I$ 做一次 LU 分解. 另外, 与幂迭代一样, 反迭代法一次只能求一个特征值.

4.2.2 Rayleigh 商迭代

在反迭代法中, 有一个很重要的问题, 就是位移 σ 的选取.

显然, 选取 σ 的基本原则是使得其与所求的特征值越靠近越好. 这里需要指出的是, 在每步迭代中, 位移 σ 可以不一样, 即在迭代过程中可以不断更新位移 σ .

由于在反迭代法 4.2 中, μ_k 是收敛到所求的特征值的, 所以我们可以选取 μ_k 作为第 k 步的位移. 这时所得到的反迭代法就称为 **Rayleigh 商迭代** (Rayleigh Quotient Iteration), 简记为 RQI.

算法 4.3 Rayleigh 商迭代法 (Rayleigh Quotient Iteration (RQI))

```

1: Choose an initial vector  $x^{(0)}$  with  $\|x^{(0)}\|_2 = 1$ 
2: set  $k = 0$ 
3: compute  $\sigma = (x^{(0)}, Ax^{(0)})$ 
4: while not converge do
5:    $y^{(k+1)} = (A - \sigma I)^{-1}x^{(k)}$ 
6:    $x^{(k+1)} = y^{(k+1)} / \|y^{(k+1)}\|_2$ 
  
```



7: $\mu_{k+1} = (Ax^{(k+1)}, x^{(k+1)})$

8: $\sigma = \mu_{k+1}$

9: $k = k + 1$

10: **end while**

一般来说, 如果 Rayleigh 商迭代收敛到 A 的一个单特征值, 则至少是二次收敛的, 即具有局部二次收敛性. 如果 A 是对称的, 则能达到局部三次收敛.

在 Rayleigh 商迭代中, 由于每次迭代的位移是不同的, 因此每次迭代需要求解一个不同的线性方程组, 这使得运算量大大增加.

例 4.2 设 $A = X\Lambda X^{-1}$, 其中 Λ 为对角矩阵, 用 Rayleigh 商迭代计算 A 的特征值.

(见 [Fig_Rayleigh.m](#))

4.3 正交迭代法

幂迭代和反迭代都只能同时计算一个特征对. 如果想同时计算多个特征对, 我们可以采用多个初始向量进行迭代. 而**正交迭代**算法就是基于这种思想, 它能够计算 A 的一个不变子空间, 从而可以同时计算出多个特征值.

算法 4.4 正交迭代法 (Orthogonal Iteration)

1: Choose an $n \times p$ column orthogonal matrix Z_0

2: set $k = 0$

3: **while** not convergence **do**

4: compute $Y_{k+1} = AZ_k$

5: $Y_{k+1} = Z_{k+1}\hat{R}_{k+1}$ % QR 分解

6: $k = k + 1$

7: **end while**

 正交迭代方法有时也称为**子空间迭代方法** (subspace iteration method) 和**同步迭代方法** (simultaneous iteration method).

在算法中使用 QR 分解是为了保持 Z_k 的列正交性, 使得其列向量组构成子空间 $\text{span}\{A^k Z_0\}$ 的一组正交基. 一方面提高算法的数值稳定性, 另一方面避免所有列都收敛到最大特征值所对应的特征向量.

下面我们分析该算法的收敛性质. 假设 A 是可对角化的, 即 $A = V\Lambda V^{-1}$, 其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, 且 $|\lambda_1| \geq \dots \geq |\lambda_p| > |\lambda_{p+1}| \geq \dots \geq |\lambda_n|$. 则可得

$$\text{span}\{Z_k\} = \text{span}\{Y_k\} = \text{span}\{AZ_{k-1}\}, \quad k = 1, 2, \dots,$$

由此可知

$$\text{span}\{Z_k\} = \text{span}\{A^k Z_0\} = \text{span}\{V\Lambda^k V^{-1} Z_0\}.$$



我们注意到

$$\Lambda^k V^{-1} Z_0 = \lambda_p^k \begin{bmatrix} (\lambda_1/\lambda_p)^k & & & \\ & \ddots & & \\ & & 1 & \\ & & & \ddots \\ & & & & (\lambda_n/\lambda_p)^k \end{bmatrix} V^{-1} Z_0 \triangleq \lambda_p^k \begin{bmatrix} W_p^{(k)} \\ W_{n-p}^{(k)} \end{bmatrix}.$$

由于当 $i > p$ 时有 $|\lambda_i/\lambda_p| < 1$, 所以当 k 趋于无穷大时, $W_{n-p}^{(k)}$ 趋向于 0. 令 $V = [V_p, V_{n-p}]$, 则

$$V \Lambda^k V^{-1} Z_0 = \lambda_p^k [V_p, V_{n-p}] \begin{bmatrix} W_p^{(k)} \\ W_{n-p}^{(k)} \end{bmatrix} = \lambda_p^k (V_p W_p^{(k)} + V_{n-p} W_{n-p}^{(k)}).$$

所以当 $k \rightarrow \infty$ 时, 有

$$\begin{aligned} \text{span}\{Z_k\} &= \text{span}\{V \Lambda^k V^{-1} Z_0\} = \text{span}\{V_p W_p^{(k)} + V_{n-p} W_{n-p}^{(k)}\} \\ &\rightarrow \text{span}\{V_p W_p^{(k)}\} = \text{span}\{V_p\}, \end{aligned}$$

即 $\text{span}\{Z_k\}$ 趋向于 A 的一个 p 维不变子空间 $\text{span}\{V_p\}$.

定理 4.1 给定正整数 p ($1 \leq p \leq n$), 考虑算法 4.4. 假设 A 是可对角化的, 且 $|\lambda_1| \geq \cdots \geq |\lambda_p| > |\lambda_{p+1}| \geq \cdots \geq |\lambda_n|$. 则 $\text{span}\{Z_k\}$ 收敛到 A 的一个 p 维不变子空间.

当 A 不可对角化时, 利用 Jordan 标准型, 我们可以得到同样的结论, 见 [112, 113].

在正交迭代中, 如果我们取 $Z_0 = I$, 则可得到一类特殊的正交迭代法. 此时, 在一定条件下, 正交迭代会收敛到 A 的 Schur 分解.

4.4 QR 迭代法

QR 迭代法的基本思想是通过不断的正交相似变换, 使得 A 趋向于一个上三角形式 (或拟上三角形式). 算法形式非常简洁, 描述如下:

算法 4.5 QR 迭代法 (QR Iteration)

- 1: Set $A_1 = A$ and $k = 1$
 - 2: **while** not convergence **do**
 - 3: $[Q_k, R_k] = \text{QR}(A_k)$ % 计算 A_k 的 QR 分解
 - 4: compute $A_{k+1} = R_k Q_k$
 - 5: $k = k + 1$
 - 6: **end while**
-

在该算法中, 我们有

$$A_{k+1} = R_k Q_k = (Q_k^T Q_k) R_k Q_k = Q_k^T (Q_k R_k) Q_k = Q_k^T A_k Q_k.$$

由这个递推关系可得

$$A_{k+1} = Q_k^T A_k Q_k = Q_k^T Q_{k-1}^T A_{k-1} Q_{k-1} Q_k = \cdots = Q_k^T Q_{k-1}^T \cdots Q_1^T A Q_1 \cdots Q_{k-1} Q_k.$$



记 $\tilde{Q}_k = Q_1 \cdots Q_{k-1} Q_k = [\tilde{q}_1^{(k)}, \tilde{q}_2^{(k)}, \dots, \tilde{q}_n^{(k)}]$, 则

$$A_{k+1} = \tilde{Q}_k^T A \tilde{Q}_k, \quad (4.1)$$

即 A_{k+1} 与 A 正交相似.

4.4.1 QR 迭代与幂迭代的关系

记 $\tilde{R}_k = R_k R_{k-1} \cdots R_1$, 则有

$$\tilde{Q}_k \tilde{R}_k = \tilde{Q}_{k-1} (Q_k R_k) \tilde{R}_{k-1} = \tilde{Q}_{k-1} (A_k) \tilde{R}_{k-1} = \tilde{Q}_{k-1} (\tilde{Q}_{k-1}^T A \tilde{Q}_{k-1}) \tilde{R}_{k-1} = A \tilde{Q}_{k-1} \tilde{R}_{k-1},$$

由此递推下去, 即可得

$$\tilde{Q}_k \tilde{R}_k = A^{k-1} \tilde{Q}_1 \tilde{R}_1 = A^{k-1} Q_1 R_1 = A^k. \quad (4.2)$$

故

$$\tilde{Q}_k \tilde{R}_k e_1 = A^k e_1,$$

这说明 QR 迭代与幂迭代有关.

假设 $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$, 则当 k 充分大时, $A^k e_1$ 收敛到 A 的模最大特征值 λ_1 所对应的特征向量, 故 $\tilde{Q}_k$ 的第一列 $\tilde{q}_1^{(k)}$ 也收敛到 λ_1 所对应的特征向量. 因此, 当 k 充分大时, $A \tilde{q}_1^{(k)} \rightarrow \lambda_1 \tilde{q}_1^{(k)}$, 此时由 (4.1) 可知, A_{k+1} 的第一列

$$A_{k+1}(:, 1) = \tilde{Q}_k^T A \tilde{q}_1^{(k)} \rightarrow \lambda_1 \tilde{Q}_k^T \tilde{q}_1^{(k)} = \lambda_1 e_1,$$

即 A_{k+1} 的第一列的第一个元素收敛到 λ_1 , 而其它元素都趋向于 0, 收敛速度取决于 $|\lambda_2/\lambda_1|$ 的大小.

4.4.2 QR 迭代与反迭代的关系

下面观察 $\tilde{Q}_k$ 的最后一列. 由 (4.1) 可知

$$A \tilde{Q}_k = \tilde{Q}_k A_{k+1} = \tilde{Q}_k Q_{k+1} R_{k+1} = \tilde{Q}_{k+1} R_{k+1},$$

所以有

$$\tilde{Q}_{k+1} = A \tilde{Q}_k R_{k+1}^{-1}.$$

由于 $\tilde{Q}_{k+1}$ 和 $\tilde{Q}_k$ 都是正交矩阵, 上式两边转置后求逆, 可得

$$\tilde{Q}_{k+1} = \left(\tilde{Q}_k^T \right)^{-1} = \left((R_{k+1}^{-1})^T \tilde{Q}_k^T A^T \right)^{-1} = \left(A^T \right)^{-1} \tilde{Q}_k R_{k+1}^T.$$

观察等式两边矩阵的最后一列, 可得

$$\tilde{q}_n^{(k+1)} = c_1 \left(A^T \right)^{-1} \tilde{q}_n^{(k)},$$

其中 c_1 为某个常数. 依此类推, 可知

$$\tilde{q}_n^{(k+1)} = c \left(A^T \right)^{-k} \tilde{q}_n^{(1)},$$

其中 c 为某个常数. 假设 A 的特征值满足 $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_{n-1}| > |\lambda_n| > 0$, 则 λ_n^{-1} 是 $(A^T)^{-1}$ 的模最大的特征值. 由幂迭代的收敛性可知, $\tilde{q}_n^{(k+1)}$ 收敛到 $(A^T)^{-1}$ 的模最大特征值 λ_n^{-1}



所对应的特征向量, 即当 k 充分大时, 有

$$(A^T)^{-1} \tilde{q}_n^{(k+1)} \rightarrow \lambda_n^{-1} \tilde{q}_n^{(k+1)}.$$

所以

$$A^T \tilde{q}_n^{(k+1)} \rightarrow \lambda_n \tilde{q}_n^{(k+1)}.$$

由 (4.1) 可知, A_{k+1}^T 的最后一列为

$$A_{k+1}^T(:, n) = \tilde{Q}_k^T A^T \tilde{q}_n^{(k)} \rightarrow \lambda_n \tilde{Q}_k^T \tilde{q}_n^{(k)} = \lambda_n e_n,$$

即 A_{k+1} 的最后一行的最后一个元素收敛到 λ_n , 而其它元素都趋向于 0, 收敛速度取决于 $|\lambda_n/\lambda_{n-1}|$ 的大小.

4.4.3 QR 迭代与正交迭代的关系

下面的定理给出了 QR 迭代法与正交迭代法 (取 $Z_0 = I$) 之间的关系.

定理 4.2 设正交迭代法 4.4 和 QR 算法 4.5 中所涉及的 QR 分解都是唯一的. 设 A_k 是由 QR 迭代法 4.5 生成的矩阵, Z_k 是由正交迭代法 4.4 (取 $Z_0 = I$) 生成的矩阵, 则有

$$A_{k+1} = Z_k^T A Z_k.$$

(板书)

4.4.4 QR 迭代的收敛性

定理 4.3 设 $A = V \Lambda V^{-1} \in \mathbb{R}^{n \times n}$, 其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, 且 $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n| > 0$. 若 V^{-1} 的所有顺序主子矩阵都非奇异 (即 V^{-1} 存在 LU 分解), 则 A_k 的对角线以下的元素均收敛到 0.

(板书)

需要指出的是, 由于 D_k 的元素不一定收敛, 故 A_{k+1} 对角线以上 (不含对角线) 的元素不一定收敛, 但这不妨碍 A_{k+1} 的对角线元素收敛到 A 的特征值 (即 A_{k+1} 的对角线元素是收敛的).

例 4.3 QR 迭代法演示. 设

$$A = X \begin{bmatrix} 9 & & \\ & 5 & \\ & & 3 \\ & & & 1 \end{bmatrix} X^{-1},$$

其中 X 是由 MATLAB 随机生成的非奇异矩阵.

在迭代过程中, 对于 A_k 的下三角部分中元素, 如果其绝对值小于某个阈值 tol , 则直接将其设为 0, 即

$$a_{ij}^{(k)} = 0 \quad \text{if } i > j \text{ and } |a_{ij}^{(k)}| < tol.$$

这里我们取 $tol = 10^{-6} \max_{1 \leq i, j \leq n} \{|a_{ij}^{(k)}|\}$.

(Eig_QR.m)



4.4.5 带位移的 QR 迭代法

为了加快 QR 迭代的收敛速度, 我们可以采用位移策略和反迭代思想.

算法 4.6 带位移的 QR 迭代法 (QR Iteration with shift)

```

1: Set  $A_1 = A$  and  $k = 1$ 
2: while not convergence do
3:   Choose a shift  $\sigma_k$ 
4:    $[Q_k, R_k] = \text{QR}(A_k - \sigma_k I)$     % 计算  $A_k - \sigma_k I$  的 QR 分解
5:   Compute  $A_{k+1} = R_k Q_k + \sigma_k I$ 
6:    $k = k + 1$ 
7: end while
  
```

与不带位移的 QR 迭代一样, 我们有

$$A_{k+1} = R_k Q_k + \sigma_k I = (Q_k^T Q_k) R_k Q_k + \sigma_k I = Q_k^T (A_k - \sigma_k I) Q_k + \sigma_k I = Q_k^T A_k Q_k$$

所以, 带位移的 QR 算法中所得到的矩阵 A_k 仍然与 $A_1 = A$ 正交相似.

在带位移的 QR 迭代法中, 一个很重要的问题就是位移 σ_k 的选取. 在前面的分析中我们已经知道, $A_{k+1}(n, n)$ 将收敛到 A 的模最小的特征值, 且收敛速度取决于模最小特征值与模第二小特征值之间的比值. 显然, 若 σ_k 就是 A 的一个特征值, 则 $A_k - \sigma_k I$ 的模最小特征值为 0, 故 QR 算法迭代一步就收敛. 此时

$$A_{k+1} = R_k Q_k + \sigma_k I = \begin{bmatrix} A_{k+1}^{(n-1) \times (n-1)} & * \\ 0 & \sigma_k \end{bmatrix}.$$

如果需要计算 A 的其它特征值, 则可对子矩阵 $A_{k+1}^{(n-1) \times (n-1)}$ 使用带位移的 QR 迭代法.

通常, 如果 σ_k 与 A 的某个特征值非常接近, 则收敛速度通常会很快. 由于 $A_k(n, n)$ 收敛到 A 的一个特征值, 所以在实际使用中, 一个比较直观的位移选择策略是 $\sigma_k = A_k(n, n)$. 事实上, 这样的位移选取方法通常会使得 QR 迭代有二次收敛速度.

例 4.4 带位移的 QR 迭代法演示. 所有数据和设置与例 4.3 相同, 在迭代过程中, 取 $\sigma_k = A_k(n, n)$. 如果 $A_k(n, n)$ 已经收敛, 则取 $\sigma_k = A_k(n-1, n-1)$. (Eig_QR_shift.m)

4.5 带位移的隐式 QR 迭代法

QR 迭代法中需要考虑的另一个重要问题就是运算量: 每一步迭代都需要做一次 QR 分解和矩阵乘积, 运算量为 $\mathcal{O}(n^3)$. 即使每计算一个特征值只需迭代一步, 则计算所有特征值也需要 $\mathcal{O}(n^4)$ 的运算量. 因此 QR 迭代算法 4.5 虽然简单易用, 但离实际应用还很远. 下面我们就想办法将总运算量从 $\mathcal{O}(n^4)$ 减小到 $\mathcal{O}(n^3)$.

为了实现这个目标, 我们需要利用 Hessenberg 矩阵. 具体步骤如下: 首先通过相似变化将 A 转化成一个上 Hessenberg 矩阵, 然后再对这个 Hessenberg 矩阵实施**隐式 QR 迭代**. 所谓隐式



QR 迭代, 就是在 QR 迭代中, 我们不需要进行显式的 QR 分解, 以及 R 和 Q 的乘积. 这样就可以将 QR 迭代的每一步运算量从 $\mathcal{O}(n^3)$ 降低到 $\mathcal{O}(n^2)$. 从而将总的运算量降低到 $\mathcal{O}(n^3)$.

带位移的隐式 QR 迭代法是当前计算中小规模矩阵的所有特征值的标准方法.

4.5.1 上 Hessenberg 矩阵

设 $H = [h_{ij}] \in \mathbb{R}^{n \times n}$, 若当 $i > j + 1$ 时, 有 $h_{ij} = 0$, 则称 H 为 **上 Hessenberg 矩阵**.

定理 4.4 设 $A \in \mathbb{R}^{n \times n}$, 则存在正交矩阵 $Q \in \mathbb{R}^{n \times n}$, 使得 QAQ^T 是上 Hessenberg 矩阵.

下面我们以一个 5×5 的矩阵 A 为例, 给出具体的**上 Hessenberg 化**过程, 所采用的工具仍然是 Householder 变换.

第一步: 令 $Q_1 = \text{diag}(I_{1 \times 1}, H_1)$, 其中 H_1 是对应于向量 $A(2:5, 1)$ 的 Householder 矩阵. 于是可得

$$Q_1 A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}.$$

由于用 Q_1^T 右乘 $Q_1 A$ 时, 不会改变 $Q_1 A$ 第一列元素的值, 故

$$A_1 \triangleq Q_1 A Q_1^T = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}.$$

第二步: 令 $Q_2 = \text{diag}(I_{2 \times 2}, H_2)$, 其中 H_2 是对应于向量 $A_1(3:5, 2)$ 的 Householder 矩阵, 则用 Q_2 左乘 A_1 时, 不会改变 A_1 的第一列元素的值. 用 Q_2^T 右乘 $Q_2 A_1$ 时, 不会改变 $Q_2 A_1$ 前两列元素的值. 因此,

$$Q_2 A_1 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix} \quad \text{和} \quad A_2 \triangleq Q_2 A_1 Q_2^T = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix}.$$

第三步: 令 $Q_3 = \text{diag}(I_{3 \times 3}, H_3)$, 其中 H_3 是对应于向量 $A_2(4:5, 3)$ 的 Householder 矩阵, 则有

$$Q_3 A_2 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A_3 \triangleq Q_3 A_2 Q_3^T = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

这时, 我们就将 A 转化成一个上 Hessenberg 矩阵, 即 $QAQ^T = A_3$ 其中 $Q = Q_3 Q_2 Q_1$ 是正交矩阵, A_3 是上 Hessenberg 矩阵.



下面是将任意一个矩阵转化成上 Hessenberg 矩阵的算法.

算法 4.7 上 Hessenberg 化 (Upper Hessenberg Reduction)

- 1: **for** $k = 1$ to $n - 2$ **do**
 - 2: compute Householder matrix $H_k = I - \beta_k v_k v_k^\top$ with respect to $A(k+1:n, k)$
 - 3: $w_k^\top = v_k^\top A(k+1:n, k:n)$
 - 4: $A(k+1:n, k:n) = H_k A(k+1:n, k:n) = A(k+1:n, k:n) - \beta_k v_k w_k^\top$
 - 5: $w_k = A(1:n, k+1:n) v_k$
 - 6: $A(k:n, k+1:n) = A(k:n, k+1:n) H_k^\top = A(k:n, k+1:n) - \beta_k w_k v_k^\top$
 - 7: **end for**
 - 8: 计算 Q
-

 在实际计算时, 我们不需要显式地形成 Householder 矩阵 H_k .

 可以存储所有的 Householder 向量, 最后用向前累积法计算 Q .

上述算法的运算量大约为 $\frac{14}{3}n^3 + \mathcal{O}(n^2)$. 如果不需要计算特征向量, 则正交矩阵 Q 也不用计算, 此时运算量大约为 $\frac{10}{3}n^3 + \mathcal{O}(n^2)$.

上 Hessenberg 矩阵的一个很重要的性质就是在 QR 迭代中能保持形状不变.

定理 4.5 设 $A \in \mathbb{R}^{n \times n}$ 是非奇异上 Hessenberg 矩阵, 其 QR 分解为 $A = QR$, 则 $\tilde{A} \triangleq RQ$ 也是上 Hessenberg 矩阵. (板书)

 若 A 是奇异的, 也可以通过选取适当的 Q , 使得定理 4.5 中的结论成立. 事实上, 由基于 Gram-Schmidt 过程的 QR 分解可知, 如果 A 是奇异的上 Hessenberg 矩阵, 则 Q 仍可取为上 Hessenberg 矩阵.

由这个性质可知, 如果 $A \in \mathbb{R}^{n \times n}$ 是上 Hessenberg 矩阵, 则 QR 迭代中的每一个 A_k 都是上 Hessenberg 矩阵. 这样, 在进行 QR 分解时, 运算量可大大降低.

Hessenberg 矩阵还有一个重要性质, 就是在 QR 迭代过程中能保持下次对角线元素非零.

定理 4.6 设 $A \in \mathbb{R}^{n \times n}$ 是非奇异的上 Hessenberg 矩阵, 且下次对角线元素均非零, 即 $a_{i+1,i} \neq 0, i = 1, 2, \dots, n-1$. 设其 QR 分解为 $A = QR$, 则 $\tilde{A} \triangleq RQ$ 的下次对角线元素也都非零.

(留作练习)

 易知, 如果上 Hessenberg 矩阵 A 存在某个下次对角线元素为零, 则 A 一定可约, 此时计算 A 的特征值就转化为计算两个更小规模的矩阵的特征值. 因此, 我们只需考虑下次对



线均非零的情形.

 需要指出的是, 如果上 Hessenberg 矩阵 A 存在某个下次对角线元素为零, 则 A 一定可约, 但反之却不成立, 即如果 A 是可约的上 Hessenberg 矩阵, A 的下次对角线元素仍可能均非零.

推论 4.7 设 $A \in \mathbb{R}^{n \times n}$ 是非奇异的上 Hessenberg 矩阵, 且下次对角线元素均非零, 则在带位移的 QR 迭代中, 所有的 A_k 的下次对角线元素均非零.

4.5.2 隐式 QR 迭代

在 QR 迭代中, 我们需要先做 QR 分解 $A_k = Q_k R_k$, 然后再计算 $A_{k+1} = R_k Q_k$. 但事实上, 我们可以将这个过程进行简化, 即在不对 A_k 进行 QR 分解的前提下, 直接计算出 A_{k+1} . 这就是**隐式 QR 迭代**.

我们这里考虑不可约的上 Hessenberg 矩阵, 即 A 的下次对角线元素都不为 0. 事实上, 若 A 是可约的, 则 A 就是一个块上三角矩阵, 这时 A 的特征值计算问题就转化成计算两个对角块的特征值问题. 隐式 QR 迭代的理论基础就是下面的**隐式 Q 定理**.

定理 4.8 (Implicit Q Theorem) 设 $H = Q^T A Q \in \mathbb{R}^{n \times n}$ 是一个不可约上 Hessenberg 矩阵, 其中 $Q \in \mathbb{R}^{n \times n}$ 是正交矩阵, 则 Q 的第 2 至第 n 列均由 Q 的第一列所唯一确定 (可相差一个符号). (板书)

由于 Q_k 的其它列都是由 Q_k 的第一列唯一确定 (至多相差一个符号), 所以我们只要找到一个正交矩阵 $\tilde{Q}_k$ 使得其第一列与 Q_k 的第一列相等, 且 $\tilde{Q}_k^T A_k \tilde{Q}_k$ 为上 Hessenberg 矩阵, 则由隐式 Q 定理可知 $\tilde{Q}_k = W Q_k$, 其中 $W = \text{diag}(1, \pm 1, \dots, \pm 1)$. 于是

$$\tilde{Q}_k^T A_k \tilde{Q}_k = W^T Q_k^T A_k Q_k W = W^T A_{k+1} W.$$

又 $W^T A_{k+1} W$ 与 A_{k+1} 相似, 且对角线元素相等, 而其它元素也至多相差一个符号, 所以不会影响 A_{k+1} 的收敛性, 即下三角元素收敛到 0, 对角线元素收敛到 A 的特征值. 换句话说, 在 QR 迭代法中, 如果我们用 $\tilde{Q}_k^T A_k \tilde{Q}_k$ 代替 $Q_k A_k Q_k^T$, 即直接令 $A_{k+1} = \tilde{Q}_k^T A_k \tilde{Q}_k$, 则其收敛性与原 QR 迭代法没有任何区别! 这就是隐式 QR 迭代的基本思想.

由于 A 是上 Hessenberg 矩阵, 因此在计算中可以使用 Givens 变换, 即 $\tilde{Q}_k$ 是一系列 Givens 变换的乘积. 下面我们举一个例子, 具体说明如何利用隐式 Q 定理, 由 A_1 得到 A_2 .

例 4.5 设 $A \in \mathbb{R}^{5 \times 5}$ 是一个不可约上 Hessenberg 矩阵, 即

$$A_1 = A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$



第一步: 构造一个 Givens 变换 G_1 , 其转置如下

$$G_1^T \triangleq G(1, 2, \theta_1) = \begin{bmatrix} c_1 & s_1 & & & \\ -s_1 & c_1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}.$$

这里 G_1 的第一列 $[c_1, s_1, 0, \dots, 0]^T$ 就是 $A_1 - \sigma_1 I$ 的第一列 $[a_{11} - \sigma_1, a_{21}, 0, \dots, 0]^T$ 的单位化后的列向量, 其中 σ_1 是位移. 于是有

$$G_1^T A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(1)} \triangleq G_1^T A G_1 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ + & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

与 A_1 相比较, $A^{(1)}$ 在 $(3, 1)$ 位置上多出一个非零元, 我们把它记为 “+”, 并称之为 **bulge**. 在下面的计算过程中, 我们的目标就是将其“赶”出矩阵, 从而得到一个新的上 Hessenberg 矩阵, 即 A_2 .

第二步: 为了消去这个 bulge, 我们可以构造 Givens 变换

$$G_2^T \triangleq G(2, 3, \theta_2) = \begin{bmatrix} 1 & & & & \\ & c_2 & s_2 & & \\ & -s_2 & c_2 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix} \quad \text{使得} \quad G_2^T A^{(1)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

为了保持与原矩阵的相似性, 需要再右乘 G_2 , 所以

$$A^{(2)} \triangleq G_2^T A^{(1)} G_2 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & + & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

此时, bulge 从 $(3, 1)$ 位置被“赶”到 $(4, 2)$ 位置.

第三步: 与第二步类似, 构造 Givens 变换

$$G_3^T \triangleq G(3, 4, \theta_3) = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & c_3 & s_3 & \\ & & -s_3 & c_3 & \\ & & & & 1 \end{bmatrix} \quad \text{使得} \quad G_3^T A^{(2)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

这时

$$A^{(3)} \triangleq G_3^T A^{(2)} G_3 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & + & * & * \end{bmatrix}.$$



于是, bulge 又从 (4, 2) 位置又被“赶”到 (5, 3) 位置.

第四步: 再次构造 Givens 变换

$$G_4^T \triangleq G(4, 5, \theta_4) = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & c_4 & s_4 \\ & & & -s_4 & c_4 \end{bmatrix} \quad \text{使得} \quad G_4^T A^{(3)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}$$

于是

$$A^{(4)} \triangleq G_4^T A^{(3)} G_4 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

现在, bulge 已经被“赶”出矩阵, 且

$$A^{(4)} = G_4^T G_3^T G_2^T G_1^T A_1 G_1 G_2 G_3 G_4 = \tilde{Q}_1^T A_1 \tilde{Q}_1,$$

其中 $\tilde{Q}_1 = G_1 G_2 G_3 G_4$. 通过直接计算可知, $\tilde{Q}_1$ 的第一列为 $[c_1, s_1, 0, 0, 0]^T$, 即 $A_1 - \sigma_1 I$ 的第一列的单位化. 根据隐式 Q 定理, $A_2 \triangleq A^{(4)} = \tilde{Q}_1^T A_1 \tilde{Q}_1$ 就是我们所需要的矩阵.

如果 $A \in \mathbb{R}^{n \times n}$ 是上 Hessenberg 矩阵, 则使用上面的算法, 带位移的隐式 QR 迭代中每一步的运算量为 $6n^2 + \mathcal{O}(n)$, 如果需要计算 Q 的话, 则总运算量为 $12n^2 + \mathcal{O}(n)$.

4.5.3 位移的选取

在带位移的 QR 迭代法中, 位移的选取非常重要. 通常, 位移越离某个特征值越近, 收敛速度就越快. 由习题 4.5 可知, 如果位移 σ 与某个特征值非常接近, 则 $A_k(n, n) - \sigma$ 就非常接近于 0. 这说明 $A_k(n, n)$ 通常会首先收敛到 A 的一个特征值, 所以 $\sigma = A_k(n, n)$ 是一个不错的选择. 但是, 如果这个特征值是复数, 这种位移选取方法就可能失效.

下面我们介绍一种针对共轭复特征值的位移选取方法, 即双位移策略.

设 $\sigma \in \mathbb{C}$ 是 A 的某个复特征值 λ 的一个很好的近似, 则其共轭 $\bar{\sigma}$ 也应该是 $\bar{\lambda}$ 的一个很好的近似. 因此我们可以考虑**双位移**策略, 即先以 σ 为位移迭代一次, 然后再以 $\bar{\sigma}$ 为位移迭代一次, 如此不断交替进行迭代. 这样就有

$$\begin{aligned} A_1 - \sigma I &= Q_1 R_1, \\ A_2 &= R_1 Q_1 + \sigma I, \\ A_2 - \bar{\sigma} I &= Q_2 R_2, \\ A_3 &= R_2 Q_2 + \bar{\sigma} I. \end{aligned} \tag{4.3}$$

容易验证

$$A_3 = Q_2^* A_2 Q_2 = Q_2^* Q_1^* A_1 Q_1 Q_2 = Q^* A_1 Q,$$

其中 $Q = Q_1 Q_2$. 我们注意到 σ 是复的, 所以 Q_1 和 Q_2 都可能是复矩阵. 但我们却可以选取适当的 Q_1 和 Q_2 , 使得 $Q = Q_1 Q_2$ 是实正交矩阵.



引理 4.9 在双位移 QR 迭代 (4.3) 中, 我们可以选取酉矩阵 Q_1 和 Q_2 使得 $Q = Q_1 Q_2$ 是实矩阵. (板书)

由这个引理可知, 存在 Q_1 和 Q_2 , 使得 $Q = Q_1 Q_2$ 是实正交矩阵, 从而 $A_3 = Q^T A_1 Q$ 也是实矩阵. 这时, 在迭代过程 (4.3) 中, 我们无需计算 A_2 , 可直接由 A_1 计算出 A_3 .

具体计算过程仍然是根据隐式 Q 定理: 我们只要找到一个实正交矩阵 Q , 使得其第一列与 $A_1^2 - 2\operatorname{Re}(\sigma)A_1 + |\sigma|^2 I$ 的第一列平行, 并且 $A_3 = Q^T A_1 Q$ 是上 Hessenberg 矩阵即可.

由于 $A_1^2 - 2\operatorname{Re}(\sigma)A_1 + |\sigma|^2 I$ 的第一列为

$$\begin{bmatrix} a_{11}^2 + a_{12}a_{21} - 2\operatorname{Re}(\sigma)a_{11} + |\sigma|^2 \\ a_{21}(a_{11} + a_{22} - 2\operatorname{Re}(\sigma)) \\ a_{21}a_{32} \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \quad (4.4)$$

所以 Q 的第一列是上述向量的单位化. 其它过程可以通过隐式 QR 迭代来实现. 但此时的“bulge”是一个 2×2 的小矩阵. 因此, 在双位移隐式 QR 迭代过程中, 我们需要使用 Householder 变换.

需要指出的是, 双位移 QR 迭代法中的运算都是实数运算.

下面我们举一个例子, 具体说明如何在实数运算下实现双位移隐式 QR 迭代法.

例 4.6 设 $A \in \mathbb{R}^{6 \times 6}$ 是一个不可约上 Hessenberg 矩阵, 即

$$A_1 = A = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$

第一步: 构造一个正交矩阵 $H_1 = \begin{bmatrix} \tilde{H}_1^T & 0 \\ 0 & I_{3 \times 3} \end{bmatrix}$, 其中 $\tilde{H}_1 \in \mathbb{R}^{3 \times 3}$, 使得其第一列与向量 (4.4) 平行. 于是有

$$H_1^T A = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ + & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(1)} \triangleq H_1^T A H_1 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ + & * & * & * & * & * \\ + & + & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$



与 A_1 相比较, $A^{(1)}$ 在 $(3, 1)$, $(4, 1)$ 和 $(4, 2)$ 位置上出现 **bulge**. 在下面的计算过程中, 我们的目标就是要把它们“赶”出矩阵, 从而得到一个新的上 Hessenberg 矩阵.

第二步: 令 $H_2 = \begin{bmatrix} I_{1 \times 1} & 0 & 0 \\ 0 & \tilde{H}_2^T & 0 \\ 0 & 0 & I_{2 \times 2} \end{bmatrix}$, 其中 $\tilde{H}_2 \in \mathbb{R}^{3 \times 3}$ 是对应于 $A(2:4, 1)$ 的 Householder 变换, 使得

$$H_2^T A^{(1)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & + & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(2)} \triangleq H_2^T A^{(1)} H_2 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & + & * & * & * & * \\ 0 & + & + & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$

这时, 我们将 bulge 向右下角方向“赶”了一个位置.

第三步: 与第二步类似, 令 $H_3 = \begin{bmatrix} I_{2 \times 2} & 0 & 0 \\ 0 & \tilde{H}_3^T & 0 \\ 0 & 0 & I_{1 \times 1} \end{bmatrix}$, 其中 $\tilde{H}_3 \in \mathbb{R}^{3 \times 3}$ 是对应于 $A(3:5, 2)$ 的 Householder 变换, 使得

$$H_3^T A^{(2)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & + & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(3)} \triangleq H_3^T A^{(2)} H_3 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & + & * & * & * \\ 0 & 0 & + & + & * & * \end{bmatrix}.$$

此时, bulge 又被向右下角方向“赶”了一个位置.

第四步: 令 $H_4 = \begin{bmatrix} I_{3 \times 3} & 0 \\ 0 & \tilde{H}_4^T \end{bmatrix}$, 其中 $\tilde{H}_4 \in \mathbb{R}^{3 \times 3}$ 是对应于 $A(4:6, 3)$ 的 Householder 变换, 使得

$$H_4^T A^{(3)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & + & * & * \end{bmatrix} \quad \text{和} \quad A^{(4)} \triangleq H_4^T A^{(3)} H_4 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & + & * & * \end{bmatrix}.$$



第五步: 此时, 只需构造一个 Givens 变换 $G_5 = \begin{bmatrix} I_{4 \times 4} & 0 \\ 0 & G(4, 5, \theta)^T \end{bmatrix}$, 使得

$$G_5^T A^{(4)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(5)} \triangleq G_5^T A^{(4)} G_5 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$

现在, bulge 已经被全部消除, 且

$$A^{(5)} = Q^T A Q,$$

其中 $Q = H_1 H_2 H_3 H_4 G_5$. 通过直接计算可知, Q 的第一列即为 H_1 的第一列, 也就是向量 (4.4) 的单位化. 根据隐式 Q 定理, 可以直接令 $A_3 \triangleq A^{(5)} = Q^T A Q$.

最后要考虑位移 σ 的具体选取问题. 在单位 QR 迭代法中, 如果 A 的特征值都是实数的话, 我们可以取 $\sigma = A_k(n, n)$. 推广到复共轭特征值上, 我们可以取 A_k 的右下角矩阵

$$\begin{bmatrix} A_k(n-1, n-1) & A_k(n-1, n) \\ A_k(n, n-1) & A_k(n, n) \end{bmatrix}$$

的复共轭特征值作为双位移. 这样选取的位移就是 **Francis 位移**.

 如果上述矩阵的两个特征值都是实的, 则选取其中模较小的特征值做单位 QR 迭代.

一般来说, 采用 Francis 位移的 QR 迭代法会使得迭代矩阵的右下角收敛到一个上三角矩阵 (两个实特征值) 或一个 2×2 的矩阵 (一对复共轭特征值), 即 $A_k(n, n-1)$ 或 $A_k(n-1, n-2)$ 趋向于 0. 带 Francis 位移的隐式 QR 迭代法通常具有二次渐进收敛性, 在实际计算中, 计算一个特征值大约平均迭代两步.

 作为一种实用的算法, 需要给出一种有效的收敛判别准则, 一种简单实用的准则是

$$|A_k(n, n-1)| \leq (|A_k(n-1, n-1)| + |A_k(n, n)|) \varepsilon_u,$$

其中 ε_u 是机器精度或给定的阈值.

 有时也可能是中间某个下次对角线元素 (不是最后两个) 首先收敛到 0, 此时可以转化为计算两个子矩阵 (对角块) 的特征值. 因此在迭代中需要判断所有下次对角线元素是否满足精度要求.

 如果同时有多个下次对角线元素满足精度要求, 则可转化为多个子矩阵.

但需要指出的是, QR 迭代并不是对所有矩阵都收敛. 例如:

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



对于上面的矩阵, 采用 Francis 位移的 QR 迭代法无效.

另外, 也可以考虑多重位移策略, 参见 [113].

4.5.4 收缩技术

在隐式 QR 迭代过程中, 当矩阵 A_{k+1} 的某个下次对角线元素 $a_{i+1,i}$ 很小时, 我们可以将其设为 0. 由于 A_{k+1} 是上 Hessenberg 矩阵, 这时 A_{k+1} 就可以写成分块上三角形式, 其中两个对角块都是上 Hessenberg 矩阵. 因此我们可以将隐式 QR 迭代作用在这两个规模相对较小的矩阵上, 从而可以大大节约运算量, 这就是收缩 (deflation) 技术.

4.6 特征向量的计算

设 A 的特征值都是实的, $R = Q^T A Q$ 是其 Schur 标准型. 若 $Ax = \lambda x$, 则 $Ry = \lambda y$, 其中 $y = Q^T x$ 或 $x = Qy$. 故只需计算 R 对应于 λ 的特征向量 y 即可.

因为 R 的对角线元素即为 A 的特征值, 不妨设 $\lambda = R(i, i)$. 假定 λ 是单重特征值, 则方程 $(R - \lambda I)y = 0$ 即为

$$\begin{bmatrix} R_{11} - \lambda I & R_{12} & R_{13} \\ 0 & 0 & R_{23} \\ 0 & 0 & R_{33} - \lambda I \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = 0,$$

即

$$(R_{11} - \lambda I)y_1 + R_{12}y_2 + R_{13}y_3 = 0, \quad (4.5)$$

$$R_{23}y_3 = 0, \quad (4.6)$$

$$(R_{33} - \lambda I)y_3 = 0, \quad (4.7)$$

其中 $R_{11} \in \mathbb{R}^{(i-1) \times (i-1)}$, $R_{33} \in \mathbb{R}^{(n-i) \times (n-i)}$. 由于 λ 是单重特征值, 故 $R_{33} - \lambda I$ 非奇异, 因此 $y_3 = 0$. 令 $y_2 = 1$, 则可得

$$y_1 = (R_{11} - \lambda I)^{-1} R_{12}.$$

因此计算特征向量 y 只需求解一个上三角线性方程组.

若 λ 是多重特征值, 则计算方法类似. 但如果 A 有复特征值, 则需要利用实 Schur 标准型.

实际计算中也可以

4.7 广义特征值问题

4.7.1 广义特征值基本理论

设 $A, B \in \mathbb{R}^{n \times n}$, 若存在 $\lambda \in \mathbb{C}$ 和非零向量 $x \in \mathbb{C}^n$ 使得

$$Ax = \lambda Bx, \quad (4.8)$$

则称 λ 为**矩阵对** (或**矩阵束**, matrix pair, matrix pencil) (A, B) 的特征值, x 为相应的特征向量. 计算矩阵对 (A, B) 的特征值和特征向量就是**广义特征值问题**. 当 $B = I$ 时, 广义特征值问题就



退化为标准特征值问题. 当 B 非奇异时, 广义特征值问题就等价于标准特征值问题

$$B^{-1}Ax = \lambda x \quad \text{或} \quad AB^{-1}y = \lambda y,$$

其中 $y = Bx$.

容易看出, λ 是 (A, B) 的一个特征值当且仅当

$$\det(A - \lambda B) = 0. \quad (4.9)$$

若 (4.9) 对所有 $\lambda \in \mathbb{C}$ 都成立, 则称矩阵对 (A, B) 是 **奇异矩阵对**, 否则称为 **正则矩阵对**.

当 B 非奇异时, 特征方程 (4.9) 是一个 n 次多项式, 因此恰好有 n 个特征值. 当 B 奇异时, 特征方程 (4.9) 的次数低于 n , 因此方程的解的个数小于 n . 但是, 注意到 $\lambda \neq 0$ 是 (A, B) 的特征值当且仅当 $\mu = \frac{1}{\lambda}$ 是 (B, A) 的特征值. 因此, 当 B 奇异时, $\mu = 0$ 是 (B, A) 的特征值, 于是我们自然地把 $\lambda = \frac{1}{\mu} = \infty$ 当作是 (A, B) 的特征值. 所以, 广义特征值不是分布在 $\mathbb{C}$ 上, 而是分布在 $\mathbb{C} \cup \{\infty\}$ 上.

容易验证, 若 U, V 非奇异, 则矩阵对 (U^*AV, U^*BV) 的特征值与 (A, B) 是一样的. 因此我们称这种变换为 **矩阵对的等价变换**. 如果 U, V 是酉矩阵, 则称为 **酉等价变换**.

4.7.2 广义 Schur 分解

广义 Schur 分解 是矩阵对在酉等价变化下的最简形式.

定理 4.10 (广义 Schur 分解) 设 $A, B \in \mathbb{C}^{n \times n}$, 则存在酉矩阵 $Q, Z \in \mathbb{C}^{n \times n}$, 使得

$$Q^*AZ = R_A, \quad Q^*BZ = R_B, \quad (4.10)$$

其中 $R_A, R_B \in \mathbb{C}^{n \times n}$ 都是上三角矩阵. 此时矩阵对 (A, B) 的特征值为 R_A 和 R_B 的对角线元素的比值, 即

$$\lambda_i = \frac{R_A(i, i)}{R_B(i, i)}, \quad i = 1, 2, \dots, n.$$

当 $R_B(i, i) = 0$ 时, 对应的特征值 $\lambda_i = \infty$. (留作课外自习)

与实 Schur 分解类似, 当 A, B 都是实矩阵时, 我们有相应的 **广义实 Schur 分解**, 具体证明过程可参见相关资料.

定理 4.11 (广义实 Schur 分解) 设 $A, B \in \mathbb{R}^{n \times n}$, 则存在正交矩阵 $Q, Z \in \mathbb{R}^{n \times n}$, 使得

$$Q^T AZ = T_A, \quad Q^T BZ = T_B, \quad (4.11)$$

其中 $T_A, T_B \in \mathbb{R}^{n \times n}$ 都是拟上三角矩阵. (留作课外自习)

4.7.3 QZ 迭代法

QZ 迭代法 是用于计算 (A, B) 的广义 Schur 分解的方法, 是 QR 方法的自然推广, 本质上可以看作是将 QR 方法作用到矩阵 AB^{-1} 上, 在具体实施时需要做一些优化, 以提高执行效率. QZ 方法的详细推导和实现过程可参见相关资料, 如 [72, 100, 123].



4.8 应用

4.8.1 多项式求根

考虑 n 次多项式

$$p_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0. \quad (4.12)$$

这里假定 a_i 都是实数, 且 $a_n \neq 0$. 由代数学基本定理可知, $p_n(x)$ 在复数域中有且仅有 n 的零点 (其中重根按重数计).

当 $n = 1$ 时, 可直接求解. 当 $n = 2$ 时, 可以通过求根公式求解. 当 $n = 3$ 时, 也存在相应的求根公式, 但不那么直观, 需要一定的构造技巧才能导出. 当 $n = 4$ 时, 可以从三次方程求根公式中导出相应的求根公式.

而当 $n \geq 5$ 时, Abel 证明了不存在求根公式, 这意味着只能用数值方法 (迭代法) 求解. 从理论上讲, 任何求解非线性方程的迭代法都可以用来计算多项式的零点, 如著名的 Newton 法. 计算出一个零点后, 通过收缩技术, 将原问题转化为 $n - 1$ 多项式零点问题, 然后继续用迭代法求解. 如此不断重复, 直到求出所有的零点. 但由于舍入误差等原因, 对于高次多项式, 随着计算过程的推进, 计算误差会越来越大, 最后的计算结果往往无法令人满意.

在 MATLAB 中, 命令 `roots` 可以计算出多项式的所有零点, 其使用的方法就是计算矩阵特征值的 QR 迭代法.

首先将多项式 (4.12) 转化为首项系数为 1 的多项式, 记为

$$q_n(x) = x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0.$$

多项式 $q_n(x)$ 可以看作是某个 n 阶矩阵的特征值多项式, 如:

$$A = \begin{bmatrix} 0 & & & -c_0 \\ 1 & 0 & & -c_1 \\ & \ddots & \ddots & \vdots \\ & & 1 & -c_{n-1} \end{bmatrix}. \quad (4.13)$$

我们称矩阵 A 为 $q_n(x)$ 的友矩阵. 这样, 计算多项式 $q_n(x)$ 的零点问题就转化为计算 A 的特征值问题.

由于 A 已经是上 Hessenberg 矩阵, 因此隐式 QR 迭代法的第一步 (上 Hessenberg 化) 就不用做了. 虽然 A 上三角部分大多是零, 而且分布也很有规律, 但无论是单位移 QR 迭代还是双位移 QR 迭代, 经过一步迭代后, 这些零元素都会消失, 因此总运算量仍然是 $\mathcal{O}(n^3)$.

于是, 如何利用 A 的特殊结构, 降低 QR 方法的运算量和存储量, 一直是颇受关注的问题. 最近, 陆续有学者 [7, 13, 19, 106] 提出了快速 QR 方法, 将运算量降为 $\mathcal{O}(n^2)$, 存储量也降为 $\mathcal{O}(n)$. 主要思想是将 A 写成一个酉矩阵与秩一矩阵之差:

$$A = \begin{bmatrix} 0 & & 1 \\ 1 & 0 & 0 \\ & \ddots & \ddots & \vdots \\ & & 1 & 0 \end{bmatrix} - \begin{bmatrix} c_0 + 1 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} [0, 0, \dots, 1] \triangleq U - xy^T.$$



易知, 经过一次 QR 迭代后, 仍可以写成一个酉矩阵与秩一矩阵之差. 基于这种观察, 就可以设计出快速的 QR 迭代法, 详细过程可参见 [7, 13, 19, 106] 或 [123].

4.9 课后习题

练习 4.1 设 $A \in \mathbb{C}^{n \times n}$. 证明:

- (1) 若 A 是上三角矩阵, 且 $A^*A = AA^*$, 则 A 必定是对角矩阵;
- (2) A 是正规矩阵的充要条件是 A 酉相似于一个对角矩阵;
- (3) 设 $\lambda_1, \lambda_2, \dots, \lambda_n$ 是 A 的特征值, 则 A 是正规矩阵的充要条件是 $\sum_{i=1}^n |\lambda_i|^2 = \|A\|_F^2$.

练习 4.2 设 $\lambda_1, \lambda_2 \in \mathbb{C}$ 是 $A \in \mathbb{C}^{n \times n}$ 的两个互不相同的特征值, $x \in \mathbb{C}^n$ 是 λ_1 的特征向量, $y \in \mathbb{C}^n$ 是 λ_2 的左特征向量. 证明: $y^*x = 0$.

练习 4.3 设 $A \in \mathbb{C}^{n \times n}$ 可对角化, 其特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$, 证明:

$$\sum_{i=1}^n |\lambda_i|^2 = \min_{\det(S) \neq 0} \|S^{-1}AS\|_F^2.$$

练习 4.4 设 $A = QR = UG$ 是非奇异矩阵 $A \in \mathbb{C}^{n \times n}$ 的两个 QR 分解, 其中 $Q, U \in \mathbb{C}^{n \times n}$ 是酉矩阵, $R, G \in \mathbb{C}^{n \times n}$ 是上三角矩阵. 证明: 存在对角矩阵 $W = \text{diag}(\alpha_1, \alpha_2, \dots, \alpha_n) \in \mathbb{C}^{n \times n}$ 满足 $|\alpha_i| = 1$, 使得

$$Q = UW, \quad R = W^{-1}G.$$

练习 4.5 设 $H = [h_{ij}] \in \mathbb{R}^{n \times n}$ 是上 Hessenberg 矩阵, 其 QR 分解为 $H = QR$, 其中 $R = [r_{ij}] \in \mathbb{R}^{n \times n}$ 是上三角矩阵且对角线元素均非负. 证明:

$$r_{kk} \geq |h_{k+1,k}|, \quad k = 1, 2, \dots, n-1.$$

因此, (1) 若 H 不可约, 则 $r_{kk} > 0, k = 1, 2, \dots, n-1$; (2) 若 H 不可约且奇异, 则 $r_{nn} = 0$.

(提示: 观察 H 的 QR 分解过程, 借助 Givens 变换)

练习 4.6 (定理 4.6) 设 $A \in \mathbb{R}^{n \times n}$ 是非奇异上 Hessenberg 矩阵且下次对角线元素均非零, 即 $a_{i+1,i} \neq 0, i = 1, 2, \dots, n-1$. 设其 QR 分解为 $A = QR$, 则 $\tilde{A} \triangleq RQ$ 的下次对角线元素也都非零.

练习 4.7 用 Householder 变换, 通过相似变换将矩阵 A 化为上 Hessenberg 型, 其中

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & -1 & -1 & 3 \\ 2 & -4 & 7 & 3 \\ 1 & 4 & -3 & 6 \end{bmatrix}.$$

练习 4.8 考虑基于 Householder 变换的上 Hessenberg 化算法, 统计乘法运算次数和加减运算次数.



练习 4.9* 设 $A \in \mathbb{C}^{m \times m}$, $B \in \mathbb{C}^{n \times n}$, $C \in \mathbb{C}^{m \times n}$, 考虑矩阵方程

$$AX - XB = C.$$

(1) 证明: 当 A 和 B 没有共同特征值时, 矩阵方程存在唯一解.

(提示: 利用 Kronecker 积, 上述矩阵方程等价于 $(I_n \otimes A - B^T \otimes I_m) \text{vec}(X) = \text{vec}(C)$, 其中 $\text{vec}(\cdot)$ 表示将矩阵按列排列得到的向量)

(2) 当 A 和 B 没有共同特征值时, 给出求解算法.

(提示: 利用 Schur 分解, 将 A 和 B 转化为相应的上三角矩阵)

练习 4.10 设 $R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$, $R_{11} \in \mathbb{R}^{p \times p}$, $R_{22} \in \mathbb{R}^{q \times q}$, 其中 $p, q = 1$ 或 2 , 且 R_{11} 和 R_{22} 没有

相同的特征值. 证明: 存在正交矩阵 Q , 使得 $Q^T R Q = \begin{bmatrix} \tilde{R}_{22} & \tilde{R}_{12} \\ 0 & \tilde{R}_{11} \end{bmatrix}$, 其中 $\tilde{R}_{11}$ 与 R_{11} 有相同的特征值, $\tilde{R}_{22}$ 与 R_{22} 有相同的特征值.

(注: 由该结论可知, 实 Schur 分解中 R 的对角块可以按任意顺序排列)

练习 4.11 统计双位移隐式 QR 迭代法执行一次迭代所需的乘法和加法运算次数.

..... 以下为可选题

练习 4.12 设 $J = \begin{bmatrix} \lambda & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda \end{bmatrix} \in \mathbb{C}^{m \times m}$, 计算其特征值 λ 对应的左、右特征向量.

练习 4.13 设

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1k} \\ & A_{22} & \cdots & A_{2k} \\ & & \ddots & \vdots \\ & & & A_{kk} \end{bmatrix},$$

其中 A_{ii} 都是方阵. 证明: A 的特征值即为对角块 $A_{11}, A_{22}, \dots, A_{kk}$ 的特征值的并.

练习 4.14* 设 $H \in \mathbb{R}^{n \times n}$ 是不可约上 Hessenberg 矩阵, 证明: 存在对角矩阵 D 使得 $D^{-1} H D$ 的次对角元素都为 1. 若 H 的次对角元素都小于 1 或都大于 1, 估计 D 的谱条件数 $\kappa_2(D)$.

练习 4.15 证明矩阵

$$A = \begin{bmatrix} 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & -c_{n-2} \\ 0 & 0 & \cdots & 1 & -c_{n-1} \end{bmatrix}$$

的特征多项式是

$$p(\lambda) = \lambda^n + c_{n-1}\lambda^{n-1} + \cdots + c_1\lambda + c_0.$$



并利用这个结果给出计算一个多项式所有零点的实用算法.

..... 以下为实践题

练习 4.16 编写函数, 实现矩阵的上 Hessenberg 化, 即算法 4.7.

练习 4.17 编写函数, 实现带 Francis 位移的隐式 QR 迭代算法的单个迭代步, 即从 A_k 到 A_{k+1} , 这里假定 A_k 是上 Hessenberg 矩阵, 且下次对角线上的元素都非零.

练习 4.18 编写函数, 实现计算友矩阵 (4.13) 的带 Francis 位移的快速 QR 迭代算法.



第五讲 对称特征值问题

设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵. 在计算 A 的特征值和特征值向量时, 我们可以充分利用 A 的对称结构, 一方面尽可能地减少运算量, 另一方面也能构造出更加快速高效的算法.

本讲主要介绍以下方法.

- **Jacobi 迭代法**: 较古老的方法, 收敛速度较慢, 但能达到很高的计算精度, 且非常适合并行.
- **Rayleigh 商迭代法**: 利用 Rayleigh 商作为动态位移的反迭代算法, 一般具有全局线性收敛和局部三次收敛性.
- **对称 QR 迭代法**: 针对对称矩阵的带位移隐式 QR 迭代算法. 如果只需计算一个对称三对角矩阵的所有特征值, 则该算法是目前最快的方法, 运算量为 $\mathcal{O}(n^2)$. 如果需要计算所有的特征值和特征向量, 则运算量约为 $6n^3$.
- **分而治之法**: 同时计算对称矩阵的特征值和特征向量的一种快速算法. 基本思想是将大矩阵分解成小矩阵, 然后利用递归思想求解, 是目前求解对称矩阵的所有特征值和特征向量的最快方法之一.
- **对分法**: 主要用于求解对称矩阵在某个区间中的特征值.

 除了 Jacobi 迭代法和 Rayleigh 商迭代法外, 其余方法都需要预先通过正交相似变换将矩阵转化为对称三对角矩阵. 这个过程大约需花费 $\frac{4}{3}n^3$ 的运算量, 如果需要计算特征向量的话, 则总运算量约为 $\frac{8}{3}n^3$.

5.1 Jacobi 迭代法

该方法的基本思想是通过一系列的 **Jacobi 旋转**, 将 A 正交相似于一个对角矩阵, 即

$$A^{(0)} = A, \quad A^{(k+1)} = J_k A^{(k)} J_k^T, \quad k = 0, 1, \dots,$$

且 $A^{(k)}$ 收敛到一个对角矩阵, 其中 J_k 为 **Jacobi 旋转**, 通常选取 J_k 为 Givens 变换, 即

$$J_k = G(i_k, j_k, \theta_k) = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & \cos \theta_k & \sin \theta_k & \\ & & & -\sin \theta_k & \cos \theta_k & \\ & & & & & 1 & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix}.$$

易知, 在 $A^{(k)}$ 两边分别左乘 J_k 和右乘 J_k^T 时, 只会修改 $A^{(k)}$ 的第 i_k 行和第 j_k 行, 以及第 i_k 列和第 j_k 列的值.

由于 $A^{(k)}$ 是对称矩阵, 由下面的引理可知, 通过选取适当的 θ_k , 可以将 $A^{(k)}(i_k, j_k)$ 和 $A^{(k)}(j_k, i_k)$ 同时化为 0.

引理 5.1 设 $A \in \mathbb{R}^{2 \times 2}$ 是对称矩阵, 则存在 Givens 变换 $G \in \mathbb{R}^{2 \times 2}$, 使得 GAG^T 为对角矩阵.

(板书)

为了使得 $A^{(k)}$ 收敛到一个对角矩阵, 其非对角线元素必须趋向于 0. 记 $\text{off}(A)$ 为所有非对角线元素的平方和, 即

$$\text{off}(A) = \sum_{i \neq j} a_{ij}^2 = \|A\|_F^2 - \sum_{i=1}^n a_{ii}^2,$$

我们的目标就是使得 $\text{off}(A)$ 尽快趋于 0.

引理 5.2 设 $A = [a_{ij}] \in \mathbb{R}^{n \times n}$ 是对称矩阵, $\hat{A} = [\hat{a}_{ij}] = JAJ^T$, $J = G(i, j, \theta)$, 其中 θ 的选取使得 $\hat{a}_{ij} = \hat{a}_{ji} = 0$, 则

$$\text{off}(\hat{A}) = \text{off}(A) - 2a_{ij}^2, \quad i \neq j.$$

(板书)

由此可知, $\text{off}(A^{(k)})$ 总是不断减小的. 下面给出 Jacobi 迭代法.

算法 5.1 Jacobi 迭代法

```

1: Given a symmetric matrix  $A \in \mathbb{R}^{n \times n}$ 
2: if eigenvectors are desired then
3:   set  $J = I$  and  $flag = 1$ 
4: end if
5: while not converge do
6:   choose an index pair  $(i, j)$  such that  $a_{ij} \neq 0$ 
7:    $\tau = (a_{ii} - a_{jj}) / (2a_{ij})$ 
8:    $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$  % 计算  $\tan \theta$ 
9:    $c = 1 / \sqrt{1 + t^2}$ ,  $s = ct$  % 计算  $\cos \theta$  和  $\sin \theta$ 
10:   $A = G(i, j, \theta)AG(i, j, \theta)^T$ 
11:  if  $flag = 1$  then
12:     $J = G(i, j, \theta)J$ 
13:  end if
14: end while

```

该算法涉及到 a_{ij} 的选取问题, 一种直观的选取方法就是使得 a_{ij} 为所有非对角线元素中绝对值最大的一个, 这就是**经典 Jacobi 迭代法**.

算法 5.2 经典 Jacobi 迭代法

```

1: Given a symmetric matrix  $A \in \mathbb{R}^{n \times n}$ 

```



```

2: if eigenvectors are desired then
3:   set  $J = I$  and  $flag = 1$ 
4: end if
5: while  $\text{off}(A) > tol$  do
6:   choose  $(i, j)$  such that  $|a_{ij}| = \max_{k \neq l} |a_{kl}|$   % 选取绝对值最大的元素
7:    $\tau = (a_{ii} - a_{jj}) / (2a_{ij})$ 
8:    $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$ 
9:    $c = 1 / \sqrt{1 + t^2}$ ,  $s = ct$   % 计算  $\cos \theta$  和  $\sin \theta$ 
10:   $A = G(i, j, \theta) A G(i, j, \theta)^T$ 
11:  if  $flag = 1$  then
12:     $J = G(i, j, \theta) J$ 
13:  end if
14: end while

```

可以证明, 经典 Jacobi 迭代法至少是线性收敛的.

定理 5.3 对于经典 Jacobi 迭代法 5.2, 我们有

$$\text{off}(A^{(k+1)}) \leq \left(1 - \frac{1}{N}\right) \text{off}(A^{(k)}), \quad N = \frac{n(n-1)}{2}.$$

故 k 步迭代后, 有

$$\text{off}(A^{(k)}) \leq \left(1 - \frac{1}{N}\right)^k \text{off}(A^{(0)}) = \left(1 - \frac{1}{N}\right)^k \text{off}(A).$$

(板书)

事实上, 经典 Jacobi 迭代法最终是 (渐进) 二次收敛的 [23, 85].

定理 5.4 经典 Jacobi 迭代法 5.2 是 N 步 (渐进) 二次收敛的, 即对足够大的 k , 有

$$\text{off}(A^{(k+N)}) = O\left(\text{off}^2(A^{(k)})\right).$$

由于在经典 Jacobi 迭代法中, 每一步都要寻找绝对值最大的非对角元, 比较费时, 实用性较差. 因此我们可以采用一种比较简单的策略, 即通过逐行扫描来选取 (i, j) , 这就是**循环 Jacobi 迭代算法**.

算法 5.3 循环 Jacobi 迭代算法 (逐行扫描)

```

1: Given a symmetric matrix  $A \in \mathbb{R}^{n \times n}$ 
2: if eigenvectors are desired then
3:   set  $J = I$  and  $flag = 1$ 
4: end if
5: while  $\text{off}(A) > tol$  do
6:   for  $i = 1$  to  $n - 1$  do

```



```

7:      for  $j = i + 1$  to  $n$  do
8:          if  $a_{ij} \neq 0$  then
9:               $\tau = (a_{ii} - a_{jj}) / (2a_{ij})$ 
10:              $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$ 
11:              $c = 1 / \sqrt{1 + t^2}$ 
12:              $s = c \cdot t$ 
13:              $A = G(i, j, \theta)^\top A G(i, j, \theta)$ 
14:             if  $flag = 1$  then
15:                  $J = J \cdot G(i, j, \theta)$ 
16:             end if
17:         end if
18:     end for
19: end for
20: end while

```

循环 Jacobi 也具有 (渐进) 二次收敛性 [115].

Jacobi 迭代法的优缺点

- 优点: 能够达到很高的计算精度 (特别是小特征值); 同时非常适合并行计算.
- 缺点: 计算速度较慢; 矩阵稀疏性得不到充分的利用.

5.2 Rayleigh 商迭代法

在反迭代算法中, $x^{(k)}$ 的 Rayleigh 商是特征值 λ_k 的近似, 因此我们可以把它作为位移, 于是就得到下面的 **Rayleigh 商迭代法**.

算法 5.4 Rayleigh 商迭代算法 (RQI, Rayleigh Quotient Iterations)

```

1: Given an initial guess  $x^{(0)}$  with  $\|x^{(0)}\|_2 = 1$ 
2: compute the Rayleigh quotient  $\rho_0 = (x^{(0)}, Ax^{(0)})$ 
3: set  $k = 1$ 
4: while not converge do
5:      $\sigma = \rho_{k-1}$ 
6:      $y^{(k)} = (A - \sigma I)^{-1} x^{(k-1)}$ 
7:      $x^{(k)} = y^{(k)} / \|y^{(k)}\|_2$ 
8:      $\rho_k = (x^{(k)}, Ax^{(k)})$ 
9:      $k = k + 1$ 
10: end while

```

关于 Rayleigh 商迭代的收敛性, 我们有下面的结论.



定理 5.5 如果特征值是单重的, 则当误差足够小时, Rayleigh 商迭代法中每步迭代所得的正确数字的位数增至三倍, 即 Rayleigh 商迭代是局部三次收敛的. (板书)

 RQI 算法具有局部三次收敛性, 但无法确定收敛到哪个特征向量 (特征值), 因此可以作为其他算法的加速手段, 即先使用其他算法 (比如幂迭代) 计算出所需特征值的近似值, 然后再使用 RQI 算法加速.

 实际计算时, 判断 $(\rho_k, x^{(k)})$ 是否收敛可以观察残量 $r_k = (A - \rho_k I)x^{(k)}$ 是否趋于零.

下面是关于 RQI 算法的全局收敛性, 可参见文献 [85].

定理 5.6 在 RQI 算法中, 设 $r_k = (A - \rho_k I)x^{(k)}$, 则有

$$\|r_{k+1}\| \leq \|r_k\|,$$

其中等号成立当且仅当 $\rho_{k+1} = \rho_k$ 且 $x^{(k)}$ 是 $(A - \rho_k I)^2$ 的特征向量.

5.3 对称 QR 迭代法

将带位移的隐式 QR 方法运用到对称矩阵, 就得到**对称 QR 迭代法**. 由于此时 A 是对称的, 所以上 Hessenberg 化后就转化为一个对称三对角矩阵, 相应的过程就称为**对称三对角化**.

 任何一个对称矩阵 $A \in \mathbb{R}^{n \times n}$ 都可以通过正交变换转化成一个对称三对角矩阵 T . 这个过程可以通过 Householder 变换或 Givens 变换来实现.

对称 QR 迭代算法基本步骤

1. 对称三对角化: 利用 Householder 变换, 将 A 化为对称三对角矩阵, 即寻找正交矩阵 Q 使得 $T = QAQ^T$ 为对称三对角矩阵;
2. 使用带 (单) 位移的隐式 QR 迭代算法计算 T 的特征值与特征值向量;
3. 计算 A 的特征向量.

对称 QR 迭代算法的运算量

- (1) 三对角化需要 $4n^3/3 + \mathcal{O}(n^2)$, 如果需要计算特征向量, 即形成矩阵 Q , 则需要额外 $4n^3/3 + \mathcal{O}(n^2)$, 因此总运算量为 $8n^3/3 + \mathcal{O}(n^2)$; 也可以暂时不计算 Q , 而是将所有 Householder 向量存放在 A 的严格上三角部分 (A 对称, 因此 A 只需保存下三角部分).
- (2) 对 T 做带位移的隐式 QR 迭代, 每次迭代的运算量为 $6n$;
- (3) 计算 T 的特征值时, 假定每个平均迭代 2 步, 则计算所有特征值的运算量为 $12n^2$; (每求出一个特征值后, 通过 deflation, 矩阵规模也会相应减小, 因此运算量可能会更少)



- (4) 若要计算 T 的所有特征值和特征向量, 则运算量为 $6n^3 + \mathcal{O}(n^2)$;
 (5) 若只要计算 A 的所有特征值, 运算量为 $4n^3/3 + \mathcal{O}(n^2)$;
 (6) 若需要计算 A 的所有特征值和特征向量, 则运算量为 $26n^3/3 + \mathcal{O}(n^2)$;

位移的选取 — Wilkinson 位移

位移的选取直接影响到算法的收敛速度. 我们可以通过下面的方式来选取位移. 设

$$A_k = \begin{bmatrix} a_1^{(k)} & b_1^{(k)} & & \\ b_1^{(k)} & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1}^{(k)} \\ & & b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix},$$

一种简单的位移选取策略就是令 $\sigma_k = a_n^{(k)}$. 这种位移选取方法几乎对所有的矩阵都有三次渐进收敛速度, 但也存在不收敛的例子, 故我们需要对其做一些改进.

 事实上, $a_n^{(k)}$ 就是收敛到特征向量的迭代向量的 Rayleigh 商: 如果在 QR 迭代算法中使用位移 $\sigma_k = a_n^{(k)}$, 而在 Rayleigh 商迭代算法 5.4 中取 $x_0 = e_n = [0, \dots, 0, 1]^T$, 则利用 QR 迭代与反迭代之间的关系可以证明: QR 迭代算法中的 σ_k 与 Rayleigh 商迭代算法中的 ρ_k 相等.

一种有效的位移是 **Wilkinson 位移**, 即取子矩阵 $\begin{bmatrix} a_{n-1}^{(k)} & b_{n-1}^{(k)} \\ b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix}$ 的最接近 $a_n^{(k)}$ 的特征值作为位移. 通过计算可得 Wilkinson 位移为

$$\sigma = a_n^{(k)} + \delta - \text{sign}(\delta) \sqrt{\delta^2 + (b_{n-1}^{(k)})^2}, \quad \text{其中} \quad \delta = \frac{1}{2}(a_{n-1}^{(k)} - a_n^{(k)}).$$

出于稳定性方面的考虑, 我们通常用下面的计算公式

$$\sigma = a_n^{(k)} - \frac{(b_{n-1}^{(k)})^2}{\delta + \text{sign}(\delta) \sqrt{\delta^2 + (b_{n-1}^{(k)})^2}}.$$

定理 5.7 [49, 85] 采用 Wilkinson 位移的 QR 迭代是整体收敛的, 且至少是线性收敛. 事实上, 几乎对所有的对称矩阵都是渐进三次收敛的.

例 5.1 带 Wilkinson 位移的隐式 QR 迭代算法收敛性演示.

(Eig_TriQR.m)

5.4 分而治之法

分而治之 (Divide-and-Conquer, DC) 算法是由 Cuppen [21] 于 1981 年首次提出, 但直到 1995 年才出现稳定的实现方式 [57]. 该算法是目前计算大规模矩阵的所有特征值和特征向量的



最快算法. 下面我们介绍该算法.

考虑不可约对称三对角矩阵

$$\begin{aligned}
 T &= \left[\begin{array}{ccc|ccc} a_1 & b_1 & & & & \\ b_1 & \ddots & & & & \\ & \ddots & a_{m-1} & b_{m-1} & & \\ & & b_{m-1} & a_m & b_m & \\ \hline & & & b_m & a_{m+1} & b_{m+1} \\ & & & & b_{m+1} & \ddots \\ & & & & & \ddots & b_{n-1} \\ & & & & & & b_{n-1} & a_n \end{array} \right] \\
 &= \left[\begin{array}{ccc|ccc} a_1 & b_1 & & & & \\ b_1 & \ddots & & & & \\ & \ddots & a_{m-1} & b_{m-1} & & \\ & & b_{m-1} & a_m - b_m & & \\ \hline & & & a_{m+1} - b_m & b_{m+1} & \\ & & & b_{m+1} & \ddots & \\ & & & & \ddots & b_{n-1} \\ & & & & & b_{n-1} & a_n \end{array} \right] + \left[\begin{array}{ccc|ccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & b_m & b_m & \\ \hline & & & b_m & b_m & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} \right] \\
 &= \left[\begin{array}{c|c} T_1 & 0 \\ \hline 0 & T_2 \end{array} \right] + b_m v v^T,
 \end{aligned}$$

其中 $v = [0, \dots, 0, 1, 1, 0, \dots, 0]^T$. 假定 T_1 和 T_2 的特征值分解已经计算出来了, 即 $T_1 = Q_1 \Lambda_1 Q_1^T$, $T_2 = Q_2 \Lambda_2 Q_2^T$, 下面考虑 T 的特征值分解.

引理 5.8 设 $x, y \in \mathbb{R}^n$, 则 $\det(I + xy^T) = 1 + y^T x$.

(留作练习)

我们首先考虑 T 的特征值与 T_1 和 T_2 的特征值之间的关系.

$$\begin{aligned}
 T &= \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^T = \begin{bmatrix} Q_1 \Lambda_1 Q_1^T & 0 \\ 0 & Q_2 \Lambda_2 Q_2^T \end{bmatrix} + b_m v v^T \\
 &= \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \left(\begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} + b_m u u^T \right) \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T,
 \end{aligned}$$

其中

$$u = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T v = \begin{bmatrix} Q_1^T \text{ 的最后一列} \\ Q_2^T \text{ 的第一列} \end{bmatrix}.$$

令 $\alpha = b_m$, $D = \text{diag}(\Lambda_1, \Lambda_2) = \text{diag}(d_1, d_2, \dots, d_n)$, 并假定 $d_1 \geq d_2 \geq \dots \geq d_n$. 则 T 的特征值与 $D + \alpha u u^T$ 的特征值相同.



下面计算 $D + \alpha uu^T$ 的特征值. 设 λ 是 $D + \alpha uu^T$ 的一个特征值, 若 $D - \lambda I$ 非奇异, 则

$$\det(D + \alpha uu^T - \lambda I) = \det(D - \lambda I) \cdot \det(I + \alpha(D - \lambda I)^{-1}uu^T).$$

故 $\det(I + \alpha(D - \lambda I)^{-1}uu^T) = 0$. 又由引理 5.8 可知

$$\det(I + \alpha(D - \lambda I)^{-1}uu^T) = 1 + \alpha u^T(D - \lambda I)^{-1}u = 1 + \alpha \sum_{i=1}^n \frac{u_i^2}{d_i - \lambda} \triangleq f(\lambda).$$

故求 A 的特征值等价于求**特征方程** (secular equation) $f(\lambda) = 0$ 的根. 由于

$$f'(\lambda) = \alpha \sum_{i=1}^n \frac{u_i^2}{(d_i - \lambda)^2},$$

当所有的 d_i 都互不相同, 且所有的 u_i 都不为零时, $f(\lambda)$ 在 $\lambda \neq d_i$ 处都是严格单调的 (见图 5.1).

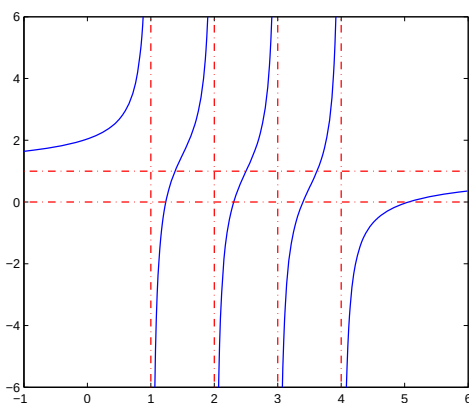


图 5.1. $f(\lambda) = 1 + 0.5 \left(\frac{1}{4-\lambda} + \frac{1}{3-\lambda} + \frac{1}{2-\lambda} + \frac{1}{1-\lambda} \right)$ 的图像

所以 $f(\lambda)$ 在每个区间 (d_{i+1}, d_i) 内都有一个根, 共 $n-1$ 个, 另一个根在 (d_1, ∞) (若 $\alpha > 0$) 或 $(-\infty, d_n)$ (若 $\alpha < 0$) 中. 由于 $f(\lambda)$ 在每个区间 (d_{i+1}, d_i) 内光滑且严格单调递增 ($\alpha > 0$) 或递减 ($\alpha < 0$), 所以在实际计算中, 可以使用对分法, 牛顿类方法, 或有理逼近等算法来求解. 通常都能很快收敛, 一般只需迭代几步即可. 因此, 计算一个特征值的运算量约为 $\mathcal{O}(n)$, 计算 $D + \alpha uu^T$ 的所有特征值的运算量约为 $\mathcal{O}(n^2)$.

当所有特征值计算出来后, 我们可以利用下面的引理来计算特征向量.

引理 5.9 设 $D \in \mathbb{R}^{n \times n}$ 为对角矩阵, $u \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$, 若 λ 是 $D + \alpha uu^T$ 的特征值, 且 $\lambda \neq d_i$, $i = 1, 2, \dots, n$, 则 $(D - \lambda I)^{-1}u$ 是其对应的特征向量. (板书)

由此, 分而治之法算法描述如下.

算法 5.5 计算对称三对角矩阵的特征值和特征向量的分而治之法 (函数形式)

```

1: function [Q, Λ] = eig_dc(T)      % T = QΛQ^T
2: if T is of 1 × 1 then
3:   Q = 1, Λ = T
4:   return

```



```

5: end if
6: form  $T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^T$ 
7:  $[Q_1, \Lambda_1] = \text{eig\_dc}(T_1)$ 
8:  $[Q_2, \Lambda_2] = \text{eig\_dc}(T_2)$ 
9: form  $D + \alpha u u^T$  from  $\Lambda_1, \Lambda_2, Q_1, Q_2$ 
10: compute the eigenvalues  $\Lambda$  and eigenvectors  $\hat{Q}$  of  $D + \alpha u u^T$ 
11: compute the eigenvectors of  $T$  with  $Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \cdot \hat{Q}$ 
12: end

```

 在分而治之法中, 特征值和特征向量是同时计算的.

实施细节

在实际使用分而治之算法时, 我们需要考虑以下几个细节问题:

- (1) 如何减小运算量;
- (2) 如何求解特征方程 $f(\lambda) = 0$;
- (3) 如何稳定地计算特征向量.

(1) 如何减小运算量 — 收缩技巧 (deflation)

分而治之算法的计算复杂性分析: 用 $t(n)$ 表示对 n 阶矩阵调用函数 `eig_dc` 的运算量, 则

$$\begin{aligned}
 t(n) &= 2 t(n/2) && \text{递归调用 DC 算法两次} \\
 &+ \mathcal{O}(n^2) && \text{计算 } D + \alpha u u^T \text{ 的特征值和特征向量} \\
 &+ c \cdot n^3 && \text{计算 } Q.
 \end{aligned}$$

 如果计算 Q 时使用的是稠密矩阵乘法, 则 $c = 2$;
若不计 $\mathcal{O}(n^2)$ 项, 则由递归公式 $t(n) = 2t(n/2) + c \cdot n^3$ 可得 $t(n) \approx c \cdot 4n^3/3$.

但事实上, 由于**收缩 (deflation)** 现象的存在, 常数 c 通常比 1 小得多.

在前面的算法描述过程中, 我们假定 d_i 互不相等且 u_i 不能为零. 事实上, 容易证明当 $d_i = d_{i+1}$ 或 $u_i = 0$ 时, d_i 即为 $D + \alpha u u^T$ 的特征值, 这种现象我们称为**收缩 (deflation)**. 在实际计算时, 当 $d_i - d_{i+1}$ 或 $|u_i|$ 小于一个给定的阈值时, 我们就近似认为 d_i 为 $D + \alpha u u^T$ 的特征值, 即出现收缩现象.

在实际计算中, 收缩现象会经常发生, 而且非常频繁, 所以我们可以而且应该利用这种优点加快分而治之算法的速度 [21, 92].

由于主要的计算量集中在计算 Q , 即算法最后一步的矩阵乘积. 如果 $u_i = 0$, 则 d_i 为特征值, 其对应的特征向量为 e_i , 即 $\hat{Q}$ 的第 i 列为 e_i , 故计算 Q 的第 i 列时不需要做任何的计算. 当 $d_i = d_{i+1}$ 时, 也存在一个类似的简化.



(2) 特征方程求解

通常我们可以使用牛顿法来计算特征方程 $f(\lambda) = 0$ 的解.

当 $d_i \neq d_{i+1}$ 且 $u_i \neq 0$ 时, 我们用牛顿法计算 $f(\lambda)$ 在 (d_{i+1}, d_i) 中的零点 λ_i . 如果 $|u_i|$ 小于给定的阈值时, 我们可直接将 d_i 作为特征值 λ_i 的一个近似. 但当 u_i 很小 (却大于给定的阈值) 时, 此时 $f(\lambda)$ 在区间 $[d_{i+1}, d_i]$ 中的大部分处的斜率几乎为 0 (见图 5.2). 这时, 如果任取 $[d_{i+1}, d_i]$ 中的一个点作为迭代初始点, 经过一次牛顿迭代后, 迭代解可能会跑到区间 $[d_{i+1}, d_i]$ 的外面, 造成不收敛.

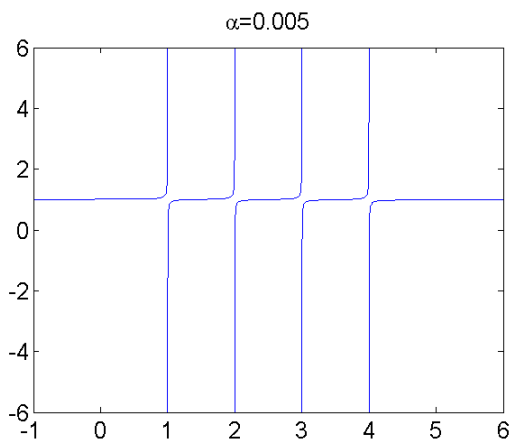


图 5.2. $f(\lambda) = 1 + 0.005 \left(\frac{1}{4-\lambda} + \frac{1}{3-\lambda} + \frac{1}{2-\lambda} + \frac{1}{1-\lambda} \right)$ 的图像

这时需要采用修正的牛顿法. 假设我们已经计算出 λ_i 的一个近似 $\tilde{\lambda}$, 下面我们需要从 $\tilde{\lambda}$ 出发, 利用牛顿迭代计算下一个近似, 直至收敛. 我们知道牛顿法的基本原理是使用 $f(\lambda)$ 在点 $\tilde{\lambda}$ 的切线来近似 $f(\lambda)$, 并将切线的零点作为下一个近似, 即用切线来近似曲线 $f(\lambda)$.

当 u_i 很小时, 这种近似方法可能会出现问题. 这时, 我们需要寻求用其它简单函数 $h(\lambda)$ (不一定是直线) 来近似 $f(\lambda)$, 然后用 $h(\lambda)$ 的零点作为 $f(\lambda)$ 零点的近似, 并不断迭代下去, 直至收敛.

关于 $h(\lambda)$ 的选取, 通常需要满足以下几个要求: (1) $h(\lambda)$ 必须容易构造; (2) $h(\lambda)$ 的零点容易计算; (3) $h(\lambda)$ 尽可能地与 $f(\lambda)$ 相近.

当然, 这样的 $h(\lambda)$ 有多种不同的构造方法. 这里介绍其中的一种方法. 假定我们需要计算的是 $f(\lambda)$ 在 (d_{i+1}, d_i) 中的零点 λ_i . 因为 d_i 和 d_{i+1} 是 $f(\lambda)$ 的奇点, 所以我们令

$$h(\lambda) = \frac{c_1}{d_i - \lambda} + \frac{c_2}{d_{i+1} - \lambda} + c_3,$$

其中 c_1, c_2, c_3 是待定的参数. 显然, $h(\lambda)$ 的零点很容易计算, 只需求解一个一元二次方程即可, 运算量与牛顿法相差无几. 选取参数 c_1, c_2, c_3 的基本原则是使得 $h(\lambda)$ 在 $\tilde{\lambda}$ 附近尽可能地接近 $f(\lambda)$. 又 $f(\lambda)$ 可写为

$$f(\lambda) = 1 + \alpha \sum_{k=1}^n \frac{u_k^2}{d_k - \lambda} = 1 + \alpha \left(\sum_{k=1}^i \frac{u_k^2}{d_k - \lambda} + \sum_{k=i+1}^n \frac{u_k^2}{d_k - \lambda} \right) \triangleq 1 + \alpha (\Psi_1(\lambda) + \Psi_2(\lambda)).$$



当 $\lambda \in (d_{i+1}, d_i)$ 时, $\Psi_1(\lambda)$ 为所有正项的和, $\Psi_2(\lambda)$ 为所有负项的和, 因此它们都可以较精确地计算. 但如果把它们加在一起时可能会引起对消, 从而可能失去相对精度. 因此我们将 $h(\lambda)$ 相应地写成

$$h(\lambda) = 1 + \alpha(h_1(\lambda) + h_2(\lambda)),$$

其中

$$h_1(\lambda) = \frac{c_1}{d_i - \lambda} + \hat{c}_1, \quad h_2(\lambda) = \frac{c_2}{d_{i+1} - \lambda} + \hat{c}_2$$

满足

$$\begin{aligned} h_1(\tilde{\lambda}) &= \Psi_1(\tilde{\lambda}), & h'_1(\tilde{\lambda}) &= \Psi'_1(\tilde{\lambda}), \\ h_2(\tilde{\lambda}) &= \Psi_2(\tilde{\lambda}), & h'_2(\tilde{\lambda}) &= \Psi'_2(\tilde{\lambda}). \end{aligned}$$

即 $h_1(\lambda)$ 和 $h_2(\lambda)$ 分别是 $\Psi_1(\lambda)$ 和 $\Psi_2(\lambda)$ 的一次 Hermite 插值函数. 通过直接计算可得

$$\begin{cases} c_1 = \Psi'_1(\tilde{\lambda})(d_i - \tilde{\lambda})^2, & \hat{c}_1 = \Psi_1(\tilde{\lambda}) - \Psi'_1(\tilde{\lambda})(d_i - \tilde{\lambda}), \\ c_2 = \Psi'_2(\tilde{\lambda})(d_{i+1} - \tilde{\lambda})^2, & \hat{c}_2 = \Psi_2(\tilde{\lambda}) - \Psi'_2(\tilde{\lambda})(d_{i+1} - \tilde{\lambda}). \end{cases} \quad (5.1)$$

所以

$$h(\lambda) = 1 + \alpha(\hat{c}_1 + \hat{c}_2) + \alpha \left(\frac{c_1}{d_i - \lambda} + \frac{c_2}{d_{i+1} - \lambda} \right). \quad (5.2)$$

这就是迭代函数.

算法 5.6 修正的 Newton 算法

- 1: set $k = 0$
 - 2: choose an initial guess $\lambda_0 \in [d_{i+1}, d_i]$
 - 3: **while** not convergence **do**
 - 4: let $\tilde{\lambda} = \lambda_k$ and compute $c_1, c_2, \hat{c}_1, \hat{c}_2$ from (5.1)
 - 5: set $k = k + 1$
 - 6: compute the solution λ_k of $h(\lambda)$ defined by (5.2)
 - 7: **end while**
-

(3) 计算特征向量的稳定算法

设 λ_i 是 $D + \alpha uu^T$ 的特征值, 则根据引理 5.9, 可利用公式 $(D - \lambda_i I)^{-1}u$ 来计算其对应的特征向量. 但遗憾的是, 当相邻的两个特征值非常接近时, 这个公式可能不稳定. 如果 λ_i 与 λ_{i+1} 非常接近, 则它们都很接近 d_{i+1} (这里假定 $\lambda_i \in (d_{i+1}, d_i)$, $\lambda_{i+1} \in (d_{i+2}, d_{i+1})$), 此时计算 $d_{i+1} - \lambda_i$ 和 $d_{i+1} - \lambda_{i+1}$ 可能会存在对消情形, 从而有可能会损失有效数字, 产生较大的相对误差. 这样就导致 $(D - \lambda_i I)^{-1}u$ 与 $(D - \lambda_{i+1} I)^{-1}u$ 的计算精度下降, 从而使得特征向量之间的正交性也因此而失去.

下面的定理可以解决这个问题. 详情参见 [55, 127].



定理 5.10 (Löwner) 设对角阵 $D = \text{diag}(d_1, d_2, \dots, d_n)$ 满足 $d_1 > d_2 > \dots > d_n$. 若矩阵 $\hat{D} = D + \hat{u}\hat{u}^T$ 的特征值 $\lambda_1, \lambda_2, \dots, \lambda_n$ 满足交错性质

$$\lambda_1 > d_1 > \lambda_2 > d_2 > \dots > \lambda_n > d_n, \quad (5.3)$$

则向量 $\hat{u}$ 的分量满足

$$|\hat{u}_i| = \left(\frac{\prod_{k=1}^n (\lambda_k - d_i)}{\prod_{k=1, k \neq i}^n (d_k - d_i)} \right)^{1/2}. \quad (5.4)$$

(留作课外自习)

设 $\lambda_1, \lambda_2, \dots, \lambda_n$ 是 $D + \alpha uu^T$ 的特征值, 且满足交错性质 (5.3). 假定 $\alpha > 0$, 则根据定理 5.10 可知, 向量 $\sqrt{\alpha}u$ 的分量满足 (5.4).

 **思考:** 如果 $\alpha < 0$, 结论会是怎样?

因此, 我们可以采用公式 (5.4) 来计算特征向量. 这样就尽可能地避免了出现分母很小的情形.

下面是计算矩阵 $D + uu^T$ 的特征值和特征向量的稳定算法.

算法 5.7 计算矩阵 $D + uu^T$ 的特征值和特征向量的稳定算法

- 1: Compute the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ by solving $f(\lambda) = 0$
- 2: Compute $\hat{u}_i$ by Löwner Theorem so that $\lambda_1, \lambda_2, \dots, \lambda_n$ are the exact eigenvalues of $D + \hat{u}\hat{u}^T$
- 3: Compute the eigenvectors of $D + \hat{u}\hat{u}^T$ by Lemma 5.9.

通过分析可以说明上述算法计算出来的 $D + \hat{u}\hat{u}^T$ 的特征值和特征值向量是非常精确的, 这意味着该算法是数值稳定的. 同时, $D + \hat{u}\hat{u}^T$ 与原矩阵 $D + uu^T$ 具有相同的特征值和特征向量, 见习题 5.4.

箭型分而治之法

分而治之算法于 1981 年被首次提出, 但直到 1995 年才由 Gu 和 Eisenstat 给出了一种快速稳定的实现方式, 称为**箭型分而治之法** (Arrowhead Divide-and-Conquer, ADC). 他们做了大量的数值试验, 在试验中, 当矩阵规模不超过 6 时, 就采用对称 QR 迭代来计算特征值和特征向量. 在对特征方程求解时, 他们采用的是修正的有理逼近法. 数值结果表明, ADC 算法的计算精度可以与其他算法媲美, 而计算速度通常比对称 QR 迭代法和 Cuppen 的分而治之法都要快. 详细介绍见 [56, 57].

5.5 对分法和反迭代法

对分法 (Bisection) 的基本思想是利用惯性定理来计算所需的部分特征值.



定义 5.1 设 A 为对称矩阵, 则其**惯性**定义为

$$\text{Inertia}(A) = (\nu, \zeta, \pi)$$

其中 ν, ζ, π 分别表示 A 的负特征值, 零特征值和正特征值的个数.

定理 5.11 (Sylvester 惯性定理) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $X \in \mathbb{R}^{n \times n}$ 非奇异, 则 $X^T A X$ 与 A 有相同的惯性.

利用 LU 分解可得 $A - zI = LDL^T$, 其中 L 为非奇异下三角矩阵, D 为对角阵, 则

$$\text{Inertia}(A - zI) = \text{Inertia}(D).$$

由于 D 是对角矩阵, 所以 $\text{Inertia}(D)$ 很容易计算.

设 $\alpha \in \mathbb{R}$, 记 $\text{Negcount}(A, \alpha)$ 为小于 α 的 A 的特征值的个数, 即

$$\text{Negcount}(A, \alpha) = \#(\lambda(A) < \alpha).$$

设 $\alpha_1 < \alpha_2$, 则 A 在区间 $[\alpha_1, \alpha_2)$ 中的特征值个数为

$$\text{Negcount}(A, \alpha_2) - \text{Negcount}(A, \alpha_1).$$

如果 $\alpha_2 - \alpha_1 < \text{tol}$ (其中 $\text{tol} \ll 1$ 为事先给定的阈值), 且 A 在 $[\alpha_1, \alpha_2)$ 中有特征值, 则我们可将 $[\alpha_1, \alpha_2)$ 中的任意一个值作为 A 在该区间中的特征值的近似. 由此我们可以给出下面的对分法.

算法 5.8 对分法: 计算 A 在 $[a, b)$ 中的所有特征值

```

1: Let  $\text{tol}$  be a given threshold
2: compute  $n_a = \text{Negcount}(A, a)$ 
3: compute  $n_b = \text{Negcount}(A, b)$ 
4: if  $n_a = n_b$  then return    % 此时  $[a, b)$  中没有  $A$  的特征值
5: put  $(a, n_a, b, n_b)$  onto worklist
6:    % worklist 中的元素是“四元素对”, 即由四个数组成的数对
7: while worklist not empty do
8:   remove  $(low, n_{low}, up, n_{up})$  from the worklist
9:   %  $(low, n_{low}, up, n_{up})$  是 worklist 中的任意一个元素
10:  if  $(up - low) < \text{tol}$  then
11:   print “There are  $n_{up} - n_{low}$  eigenvalues in  $[low, up)$ ”
12:  else
13:   compute  $mid = (low + up)/2$ 
14:   compute  $n_{mid} = \text{Negcount}(A, mid)$ 
15:   if  $(n_{mid} > n_{low})$  then
16:    put  $(low, n_{low}, mid, n_{mid})$  onto worklist
17:   end if
18:   if  $(n_{up} > n_{mid})$  then

```



```

19:         put (mid, n_mid, up, n_up) onto worklist
20:     end if
21: end if
22: end while

```

显然, 对分法的主要运算量集中在计算 $\text{Negcount}(A, z)$. 通常是事先将 A 转化成对称三对角矩阵, 这样计算 $A - zI$ 的 LDL^T 分解就非常简单:

$$\begin{aligned}
 A - zI &= \begin{bmatrix} a_1 - z & b_1 & & \\ b_1 & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n - z \end{bmatrix} \\
 &= \begin{bmatrix} 1 & & & \\ l_1 & \ddots & & \\ & \ddots & \ddots & \\ & & l_{n-1} & 1 \end{bmatrix} \begin{bmatrix} d_1 & & & \\ & \ddots & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} 1 & l_1 & & \\ & \ddots & \ddots & \\ & & \ddots & l_{n-1} \\ & & & 1 \end{bmatrix} \triangleq \text{LDL}^T.
 \end{aligned}$$

利用待定系数法, 可以得到下面的递推公式

$$d_1 = a_1 - z, \quad d_i = (a_i - z) - \frac{b_{i-1}^2}{d_{i-1}}, \quad i = 2, 3, \dots, n. \quad (5.5)$$

用上面的公式计算 d_i 的运算量约为 $4n$.

注意这里没有选主元, 但针对对称三对角矩阵, 该算法是非常稳定的, 即使当 d_i 有可能很小时, 算法依然很稳定.

定理 5.12 [23] 利用公式 (5.5) 计算所得的 d_i 与精确计算 $\hat{A}$ 的 $\hat{d}_i$ 有相同的符号, 故有相同的惯性. 这里 $\hat{A}$ 与 A 非常接近, 即

$$\hat{A}(i, i) = a_i, \quad \hat{A}(i, i+1) = b_i(1 + \varepsilon_i),$$

其中 $|\varepsilon_i| \leq 2.5\varepsilon + \mathcal{O}(\varepsilon^2)$, 这里的 ε 为机器精度.

 由于单独调用一次 Negcount 的运算量为 $4n$, 故计算 k 个特征值的总运算量约为 $\mathcal{O}(kn)$.

当特征值计算出来后, 我们可以使用带位移的迭代来计算对应的特征向量. 通常只需迭代 1 至 2 次即可, 由于 A 是三对角矩阵, 故计算每个特征向量的运算量为 $\mathcal{O}(n)$. 整个合起来就构成 **对分法和迭代**.

当特征值紧靠在一起时, 计算出来的特征向量可能会失去正交性, 此时需要进行**再正交化**, 可通过 MGS 的 QR 分解来实现.

5.6 奇异值分解

奇异值分解 (SVD) 具有十分广泛的应用背景, 因此, 如何更好更快地计算一个给定矩阵的 SVD 是科学与工程计算领域中的一个热门研究课题, 吸引了众多专家进行这方面的研究, 也涌现



出了许多奇妙的方法. 本章主要介绍计算 SVD 的常用算法.

对任意矩阵 $A \in \mathbb{R}^{m \times n}$, 其奇异值与对称矩阵 $A^T A$, AA^T 和 $\begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$ 的特征值是密切相关的, 故理论上计算对称特征值的算法都可以用于计算奇异值. 但在实际计算中, 我们需要对算法进行优化, 使得计算过程更加高效, 计算结果更加精确.

当前的 SVD 算主要可分为两类 [24]: 一类是基于二对角化的算法, 包括 Golub-Kahan SVD 算法 (即 QR 算法)、DC 算法、dqds 算法、对分法、MRRR 算法等; 另一类是 Jacobi 算法, 包括双边 Jacobi 算法和单边 Jacobi 算法.

关于 SVD 的相关参考资料

- J. Dongarra, et al., *The Singular Value Decomposition: Anatomy of Optimizing an Algorithm for Extreme Scale*, 2018. [24]

5.6.1 二对角化

基于二对角化的 SVD 算法通常分为以下三个步骤:

1. 将 A 二对角化: $B = U_1^T A V_1$, 其中 B 为上二对角矩阵, U_1, V_1 为正交阵;
2. 计算 B 的 SVD: $B = U_2 \Sigma V_2^T$, 其中 Σ 为对角阵, U_2, V_2 为正交阵;
3. 合并得到 A 的 SVD: $A = U_1 B V_1^T = (U_1 U_2) \Sigma (V_1 V_2)^T$.

 Cleve Moler's movie about SVD <https://www.youtube.com/watch?v=R9UoFyqJca8>

我们知道, 对称矩阵可以通过一系列 Householder 变换转化为对称三对角矩阵. 对于一般矩阵 $A \in \mathbb{R}^{m \times n}$, 我们也可以通过 Householder 变换, 将其转化为二对角矩阵, 即计算正交矩阵 U_1 和 V_1 使得

$$U_1^T A V_1 = B, \quad (5.6)$$

其中 B 是一个实 (上) 二对角矩阵. 这个过程就称为**二对角化**.

 需要注意的是, 与对称矩阵的对称三对角化不同, A 与 B 是不相似的.

设 $A \in \mathbb{R}^{m \times n}$, 二对角化过程大致如下:

- (1) 首先构造一个 Householder 矩阵 $H_1 \in \mathbb{R}^{m \times m}$, 使得 $H_1 A$ 的第一列除第一个元素外, 其它分量都为零, 即

$$H_1 A = \begin{bmatrix} * & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & * & * & \cdots & * \end{bmatrix}.$$

- (2) 再构造一个 Householder 矩阵 $\tilde{H}_1 \in \mathbb{R}^{(n-1) \times (n-1)}$, 把 $H_1 A$ 的第一行的第 3 至第 n 个元素



化为零, 即

$$H_1 A \begin{bmatrix} 1 & 0 \\ 0 & \tilde{H}_1 \end{bmatrix} = \begin{bmatrix} * & * & 0 & \cdots & 0 \\ 0 & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & * & * & \cdots & * \end{bmatrix}.$$

(3) 重复上面的过程, 直到把 A 最终化为二对角矩阵.

有了分解 (5.6) 以后, 我们可得

$$A^T A = (U_1 B V_1^T)^T U_1 B V_1^T = V_1 B^T B V_1^T,$$

即 $V_1^T A^T A V_1 = B^T B$. 由于 $B^T B$ 是对称三对角的, 所以这就相当于将 $A^T A$ 三对角化.

 整个二对角化过程的运算量约为 $4mn^2 + 4m^2n - 4n^3/3$, 若不需要计算 U_1 和 V_1 , 则运算量约为 $4mn^2 - 4n^3/3$.

二对角矩阵的奇异值分解

设 $B \in \mathbb{R}^{n \times n}$ 是一个二对角矩阵

$$B = \begin{bmatrix} a_1 & b_1 & & \\ & \ddots & \ddots & \\ & & \ddots & b_{n-1} \\ & & & a_n \end{bmatrix}. \quad (5.7)$$

我们假定二对角矩阵的元素都非负, 即 $a_k, b_k \geq 0$, 否则可以通过在左右分别乘以一个正交矩阵来实现 (见习题 5.8). 下面三种方法均可将计算 B 的 SVD 转化成计算对称三对角矩阵的特征分解:

- (1) 构造**增广矩阵** $T = \begin{bmatrix} 0 & B^T \\ B & 0 \end{bmatrix}$, 取置换矩阵 $P = [e_1, e_{n+1}, e_2, e_{n+2}, \dots, e_n, e_{2n}]$, 则 $T_{ps} = P^T T P$ 是对称三对角矩阵, 且主对角线元素全为 0, 次对角线元素为 $a_1, b_1, a_2, b_2, \dots, a_{n-1}, b_{n-1}, a_n$. 设 (λ_i, x_i) 是 T_{ps} 的一个特征对, 则

$$\lambda_i = \pm \sigma_i, \quad P x_i = \frac{1}{\sqrt{2}} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix},$$

其中 σ_i 为 B 一个奇异值, u_i 和 v_i 分别为对应的左和右奇异向量.

- (2) 令 $T_{BB^T} = B B^T$, 则

$$T_{BB^T} = \begin{bmatrix} a_1^2 + b_1^2 & a_2 b_1 & & \\ a_2 b_1 & \ddots & \ddots & \\ & \ddots & a_{n-1}^2 + b_{n-1}^2 & a_n b_{n-1} \\ & & a_n b_{n-1} & a_n^2 \end{bmatrix}.$$

T_{BB^T} 的特征值为 B 的奇异值的平方, 且 T_{BB^T} 的特征向量为 B 的左奇异向量.



(3) 令 $T_{B^T B} = B^T B$, 则

$$T_{B^T B} = \begin{bmatrix} a_1^2 & a_1 b_1 & & \\ a_1 b_1 & a_2^2 + b_1^2 & \ddots & \\ & \ddots & \ddots & a_{n-1} b_{n-1} \\ & & a_{n-1} b_{n-1} & a_n^2 + b_{n-1}^2 \end{bmatrix}.$$

$T_{B^T B}$ 的特征值为 B 的奇异值的平方, 且 $T_{B^T B}$ 的特征向量为 B 的右奇异向量.

 Golub & Kahan (1965) 采用前面两种做法. Reinsch (1970) 采用第三种做法, 其巧妙之处是, 只需对 B 做 Givens 变换即可, 类似隐式 QR 迭代.

理论上, 我们可以直接使用 QR 迭代、分而治之法或带反迭代的对分法, 计算对称三对角矩阵 T_{ps} , T_{BB^T} 和 $T_{B^T B}$ 的特征值和特征向量. 但一般来说, 这种做法并不是最佳的, 原因如下:

- (1) 对 T_{ps} 做 QR 迭代并不划算, 因为 QR 迭代计算所有的特征值和特征向量, 而事实上只要计算正的特征值即可;
- (2) 直接构成 T_{BB^T} 或 $T_{B^T B}$ 是数值不稳定的. 事实上, 这样做可能会使得 B 的小奇异值的精度丢失一半.

下面是一些计算奇异值分解的比较实用的算法.

1. **Golub-Kahan SVD 算法**: 由 Golub 和 Kahan [46] 于 1965 年提出, 并在 1970 年进行了完善 [44], 是一种十分稳定且高效的计算 SVD 的算法, 也是最早的实用 SVD 算法. 主要思想是将带位移的对称 QR 迭代算法隐式地用到 $B^T B$ 上, 在该算法中, 并不需要显示地把 $B^T B$ 计算出来. 该算法也通常就称为 SVD 算法, 是一个基本且实用的算法, 目前仍然是计算小规模矩阵奇异值分解的常用算法. 关于这个算法的详细描述, 也可以参见 [49, 123].
2. **dqds 算法**: 由 Fernando 和 Parlett [39] 于 1994 年提出, 是计算二对角矩阵所有奇异值的最快算法, 而且能达到很高的相对精度, 包括奇异值很小的情形. 该算法主要基于对 $B^T B$ 的 Cholesky 迭代, 可以看作是 LR 迭代算法的改进. 由于 LR 迭代算法在一定条件下与对称 QR 算法是等价的, 因此该算法也可以看作是 QR 迭代的变形. 该算法是 LAPACK 中计算二对角矩阵奇异值 (不计算奇异向量) 的标准算法 [24].
3. **分而治之法**: 该算法是计算较大规模矩阵的所有奇异值和奇异向量的最快算法, 但不能保证小奇异值的相对精度, 即 σ_i 的相对精度为 $\mathcal{O}(\epsilon)\sigma_1$, 而不是 $\mathcal{O}(\epsilon)\sigma_i$.
4. **对分法和反迭代**: 主要用于计算某个区间内的奇异值及对应的奇异向量, 能保证较高的相对精度.
5. **Jacobi 迭代**: 可隐式地对 AA^T 或 $A^T A$ 实施对称 Jacobi 迭代, 能保证较高的相对精度. 最近, Z. Drmač 和 K. Veselić [27, 28] 改进了最初的 Jacobi 算法, 使其变成一个速度快、精度高的实用算法.

在这里, 我们介绍 Golub-Kahan SVD 算法, dqds 算法和 Jacobi 迭代.



5.6.2 Golub-Kahan SVD 算法

该算法主要思想是将带位移的对称 QR 迭代算法隐式地用到 $B^T B$ 上, 而无需将 $B^T B$ 显式地计算出来. Golub-Kahan SVD 算法有时也简称 SVD 算法, 其基本框架是:

- 将矩阵 A 二对角化, 得到上二对角矩阵 B ;
- 用隐式 QR 迭代计算 $B^T B$ 的特征值分解, 即

$$B^T B = Q \Lambda Q^T, \quad \Lambda = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2). \quad (5.8)$$

- 计算 BQ 的列主元 QR 分解, 即

$$(BQ)P = UR, \quad (5.9)$$

其中 P 是置换矩阵, U 是正交矩阵, R 是上三角矩阵.

由 (5.8) 可知

$$(BQ)^T BQ = \Lambda,$$

因此 BQ 是列正交矩阵 (但不是单位列正交). 再由 (5.9) 可知 $R = U^T (BQ)P$ 也是列正交矩阵. 又 R 是上三角矩阵, 所以 R 必定是对角矩阵. 令 $V = QP$, 则由 (5.9) 可知

$$U^T B V = R.$$

这就是二对角矩阵 B 的奇异值分解.

算法的具体实现可参见 [49, 123].

5.6.3 dqds 算法

我们首先介绍针对实对称正定矩阵的 LR 算法, 该算法是基于 Cholesky 分解.

算法 5.9 带位移的 LR 算法

- 1: Let T_0 be a given real symmetric positive definite matrix
 - 2: set $i = 0$
 - 3: **while** not converge **do**
 - 4: choose a shift τ_i^2 satisfying $\tau_i^2 < \min\{\lambda(T_i)\}$
 - 5: compute B_i such that $T_i - \tau_i^2 I = B_i^T B_i$ % Cholesky factorization
 - 6: $T_{i+1} = B_i B_i^T + \tau_i^2 I$
 - 7: $i = i + 1$
 - 8: **end while**
-

LR 迭代算法在形式上与 QR 迭代算法非常类似, 但提出时间更早. 事实上, 对于不带位移的 LR 迭代算法, 我们可以证明, 两步 LR 迭代等价于一步 QR 迭代.

引理 5.13 设 $\tilde{T}$ 是不带位移的 LR 算法迭代两步后生成的矩阵, $\hat{T}$ 是不带位移的 QR 算法迭代一步后生成的矩阵, 则 $\tilde{T} = \hat{T}$.



- 🔥 (1) LR 算法中要求 T_0 对称正定, 但并不一定是三对角矩阵;
 (2) 由该引理可知, QR 算法与 LR 算法有相同的收敛性.

下面我们介绍 dqds (differential quotient difference with shifts) 算法, 主要参考 Demmel [23].

dqds 算法是计算二对角矩阵的奇异值的主要算法之一. 在数学上, dqds 算法与带位移的 LR 算法是等价的, 但在该算法中, 我们是直接通过 B_i 来计算 B_{i+1} , 从而避免计算中间矩阵 T_{i+1} , 这样也就尽可能地避免了由于计算 $B_i B_i^T$ 而可能带来的数值不稳定性.

下面推导如何从 B_i 直接计算 B_{i+1} . 设

$$B_i = \begin{bmatrix} a_1 & b_1 & & \\ & a_2 & \ddots & \\ & & \ddots & b_{n-1} \\ & & & a_n \end{bmatrix}, \quad B_{i+1} = \begin{bmatrix} \tilde{a}_1 & \tilde{b}_1 & & \\ & \tilde{a}_2 & \ddots & \\ & & \ddots & \tilde{b}_{n-1} \\ & & & \tilde{a}_n \end{bmatrix}.$$

为了书写方便, 我们记 $b_0 = b_n = \tilde{b}_0 = \tilde{b}_n = 0$. 由带位移的 LR 算法 5.9 可知

$$B_{i+1}^T B_{i+1} + \tau_{i+1}^2 I = B_i B_i^T + \tau_i^2 I.$$

比较等式两边矩阵的对角线和上对角线元素, 可得

$$\begin{aligned} \tilde{a}_k^2 + \tilde{b}_{k-1}^2 + \tau_{i+1}^2 &= a_k^2 + b_k^2 + \tau_i^2, \quad k = 1, 2, \dots, n \\ \tilde{a}_k \tilde{b}_k &= a_{k+1} b_k \quad \text{或} \quad \tilde{a}_k^2 \tilde{b}_k^2 = a_{k+1}^2 b_k^2, \quad k = 1, 2, \dots, n-1. \end{aligned}$$

记 $\delta = \tau_{i+1}^2 - \tau_i^2$, $p_k = a_k^2$, $q_k = b_k^2$, $\tilde{p}_k = \tilde{a}_k^2$, $\tilde{q}_k = \tilde{b}_k^2$, 则可得下面的 **qds 算法**.

算法 5.10 qds 算法的单步 ($B_i \rightarrow B_{i+1}$)

- 1: $\delta = \tau_{i+1}^2 - \tau_i^2$
 - 2: **for** $k = 1$ to $n - 1$ **do**
 - 3: $\tilde{p}_k = p_k + q_k - \tilde{q}_{k-1} - \delta$
 - 4: $\tilde{q}_k = q_k \cdot (p_{k+1} / \tilde{p}_k)$
 - 5: **end for**
 - 6: $\tilde{p}_n = p_n - \tilde{q}_{n-1} - \delta$
-

qds 算法中的每个循环仅需 5 个浮点运算, 所以运算量较少.

为了提高算法的精度, 我们引入一个辅助变量 $d_k \triangleq p_k - \tilde{q}_{k-1} - \delta$, 则

$$\begin{aligned} d_k &= p_k - \tilde{q}_{k-1} - \delta \\ &= p_k - \frac{q_{k-1} p_k}{\tilde{p}_{k-1}} - \delta \\ &= p_k \cdot \frac{\tilde{p}_{k-1} - q_{k-1}}{\tilde{p}_{k-1}} - \delta \\ &= p_k \cdot \frac{p_{k-1} - \tilde{q}_{k-2} - \delta}{\tilde{p}_{k-1}} - \delta = \frac{p_k}{\tilde{p}_{k-1}} \cdot d_{k-1} - \delta. \end{aligned}$$

于是就可得到 **dqds 算法**:



算法 5.11 dqds 算法的单步 ($B_i \rightarrow B_{i+1}$)

```

1:  $\delta = \tau_{i+1}^2 - \tau_i^2$ 
2:  $d_1 = p_1 - \delta$ 
3: for  $k = 1$  to  $n - 1$  do
4:    $\tilde{p}_k = d_k + q_k$ 
5:    $t = p_{k+1} / \tilde{p}_k$ 
6:    $\tilde{q}_k = q_k \cdot t$ 
7:    $d_{k+1} = d_k \cdot t - \delta$ 
8: end for
9:  $\tilde{p}_n = d_n$ 

```

dqds 算法是数值稳定的, 运算量与 dqs 算法差不多.

考虑不带位移的情形, 即 $\tau_i = 0$, 此时 $\delta = 0$, 对应的算法为 qd 算法和 dqd 算法. 我们看到, qd 算法中有正数之间的减法运算 ($p_k, q_k, \tilde{q}_k$ 都是正数), 因此有可能出现相近的数相减, 从而损失计算精度. 而 dqd 算法避免了这种情况的出现, 因此计算更加精确.

 qd 算法和 dqd 算法由 Rutishauser (1954) 提出, 但他并没有考虑位移 [39]. Fernando 和 Parlett [39] 则给出了一种有效的位移选择策略, 从而有效提升了算法的计算效率 (如果只计算二对角矩阵的奇异值的话, dqds 算法比 QR 算法更快). 关于 dqds 算法中位移的选取, 以及如何判断收敛性 (收缩), 可以参见 [39].

 dqds 算法是求解二对角矩阵所有奇异值的主要方法之一, 2000 年被加入到 LAPACK 中 [4, 86]. 关于 dqds 的进一步改进可以参见 [75].

5.6.4 Jacobi 算法

本节讨论对矩阵 $M = A^T A$ 实施隐式的 Jacobi 算法来计算 A 的奇异值.

我们知道, Jacobi 算法的每一步就是对矩阵作 Jacobi 旋转, 即 $A^T A \rightarrow J^T A^T A J$, 其中 J 的选取将某两个非对角元化为 0. 在实际计算中, 我们只需计算 AJ , 故该算法称为**单边 Jacobi 旋转**.

算法 5.12 单边 Jacobi 旋转的单步

```

% 对  $M = A^T A$  作 Jacobi 旋转, 将  $M(i, j), M(j, i)$  化为 0
1: Compute  $m_{ii} = (A^T A)_{ii}, m_{ij} = (A^T A)_{ij}, m_{jj} = (A^T A)_{jj}$ 
2: if  $m_{ij}$  is not small enough then
3:    $\tau = (m_{ii} - m_{jj}) / (2 \cdot m_{ij})$ 
4:    $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$ 
5:    $c = 1 / \sqrt{1 + t^2}$ 
6:    $s = c \cdot t$ 

```



```

7:   A = AG(i, j,  $\theta$ )   % G(i, j,  $\theta$ ) 为 Givens 变换
8:   if eigenvectors are desired then
9:       J = J · G(i, j,  $\theta$ )
10:   end if
11: end if

```

在上面算法的基础上, 我们可以给出完整的单边 Jacobi 算法.

算法 5.13 单边 Jacobi 算法: 计算 $A = U\Sigma V^T$

```

1: while  $A^T A$  is not diagonal enough do
2:   for  $i = 1$  to  $n - 1$  do
3:     for  $j = i + 1$  to  $n$  do
4:       调用单边 Jacobi 旋转
5:     end for
6:   end for
7: end while
8: compute  $\sigma_i = \|A(:, i)\|_2, i = 1, 2, \dots, n$ 
9:  $U = [u_1, \dots, u_n]$  with  $u_i = A(:, i)/\sigma_i$ 
10:  $V = J$ 

```

 单边 Jacobi 算法也可以看作是对 A 做一系列的 Givens 变换, 使得 A 的列相互正交 (即对 A 的列进行正交化).

Jacobi 算法的特点:

- 不需要双对角化, 这样可以避免双对角化引入的误差;
- 可达到相对较高的计算精度;
- 速度较慢. (目前已有快速的改进算法, 参见 [27, 28])

定理 5.14 设 $A = DX \in \mathbb{R}^{n \times n}$, 其中 D 为非奇异对角阵, X 非奇异. 设 $\hat{A}$ 是按浮点运算单边 Jacobi 旋转 m 次后所得到的矩阵. 若 A 和 $\hat{A}$ 的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$ 和 $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \dots \geq \hat{\sigma}_n$, 则

$$\frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} \leq \mathcal{O}(m\varepsilon)\kappa(X).$$

故 X 的条件数越小, 计算矩阵 A 的奇异值时相对误差越小.

5.7 扰动分析

设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 则有下面的谱分解.



定理 5.15 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵. 则存在一个正交矩阵 Q 使得

$$A = Q\Lambda Q^T,$$

其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 是一个实对角矩阵.

这里的 λ_i 就是 A 的特征值, 我们假设 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$. 令 $Q = [q_1, q_2, \dots, q_n]$, 则 q_i 就是 λ_i 对应的单位正交特征向量.

关于对称矩阵特征值问题的扰动理论, 这里只做一些简单介绍, 若要深入了解这方面的信息, 可以参考 [62, 63, 101, 120].

5.7.1 特征值与 Rayleigh 商

定义 5.2 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 向量 $x \in \mathbb{R}^n$ 非零, 则 x 关于 A 的 **Rayleigh 商** 定义为

$$\rho(x, A) = \frac{x^T A x}{x^T x}. \quad (5.10)$$

有时简记为 $\rho(x)$.

下面是关于实对称矩阵 Rayleigh 商的一些基本性质:

- (1) $\rho(\alpha x) = \rho(x)$, $\forall \alpha \in \mathbb{R}, \alpha \neq 0$;
- (2) $\rho(q_i) = \lambda_i$, $i = 1, 2, \dots, n$;
- (3) 设 $x = \alpha_1 q_1 + \alpha_2 q_2 + \dots + \alpha_n q_n$, 则

$$\rho(x) = \frac{\alpha_1^2 \lambda_1 + \alpha_2^2 \lambda_2 + \dots + \alpha_n^2 \lambda_n}{\alpha_1^2 + \alpha_2^2 + \dots + \alpha_n^2};$$

- (4) $\lambda_n \leq \rho(x) \leq \lambda_1$, $|\rho(x)| \leq \|A\|_2$.

实对称矩阵的特征值与 Rayleigh 商之间的一个重要性质是 Courant-Fischer 极小极大定理.

定理 5.16 (Courant-Fischer) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 其特征值为 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, 则有

$$\lambda_k = \max_{\mathbb{U} \in \mathbb{S}_k^n} \min_{x \in \mathbb{U}, x \neq 0} \frac{x^T A x}{x^T x} = \min_{\mathbb{V} \in \mathbb{S}_{n-k+1}^n} \max_{x \in \mathbb{V}, x \neq 0} \frac{x^T A x}{x^T x},$$

其中 $\mathbb{S}_i^n$ 表示 $\mathbb{R}^n$ 中所有 i 维子空间构成的集合. 当

$$\mathbb{U} = \text{span}\{q_1, \dots, q_k\}, \quad \mathbb{V} = \text{span}\{q_k, \dots, q_n\}, \quad x = q_k$$

时, 上式中的等号成立.

(板书)

 该结论在复数域中也成立 [63].

当 $k = 1$ 和 $k = n$ 时, 就可以得到下面的定理.



定理 5.17 (Rayleigh-Ritz) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 其特征值为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, 则有

$$\lambda_1 = \max_{x \in \mathbb{R}^n, x \neq 0} \frac{x^T A x}{x^T x}, \quad \lambda_n = \min_{x \in \mathbb{R}^n, x \neq 0} \frac{x^T A x}{x^T x}.$$

由极小极大定理, 我们可以得到下面的特征值分隔定理.

定理 5.18 (分隔定理) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $B = Q^T A Q$, 其中 $Q \in \mathbb{R}^{n \times (n-1)}$ 满足 $Q^T Q = I_{n-1}$. 再设 A 和 B 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_{n-1},$$

则有

$$\lambda_1 \geq \tilde{\lambda}_1 \geq \lambda_2 \geq \tilde{\lambda}_2 \cdots \geq \tilde{\lambda}_{n-1} \geq \lambda_n.$$

特别地, 在定理 5.18 中, 取 $Q = [e_1, \dots, e_{i-1}, e_{i+1}, \dots, e_n]$, 则可以得到下面的结论.

推论 5.19 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $\tilde{A}$ 是 A 的一个 $n-1$ 阶主子矩阵, A 和 $\tilde{A}$ 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_{n-1},$$

则有

$$\lambda_1 \geq \tilde{\lambda}_1 \geq \lambda_2 \geq \tilde{\lambda}_2 \cdots \geq \tilde{\lambda}_{n-1} \geq \lambda_n.$$

反复应用上面的推论, 即可得到下面的结论.

推论 5.20 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $\tilde{A}$ 是 A 的一个 k 阶主子矩阵 ($1 \leq k \leq n-1$), A 和 $\tilde{A}$ 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_k,$$

则有

$$\lambda_i \geq \tilde{\lambda}_i \geq \lambda_{n-k+i}, \quad i = 1, 2, \dots, k.$$

5.7.2 对称矩阵特征值的扰动分析

设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 扰动矩阵 $E \in \mathbb{R}^{n \times n}$ 也是对称矩阵, 下面讨论 $A + E$ 的特征值与 A 的特征值之间的关系.

由极小极大定理, 我们可以证明下面的性质.



定理 5.21 设 $A \in \mathbb{R}^{n \times n}$ 和 $B = A + E \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 其特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n.$$

假定 E 的最大和最小特征值分别为 μ_1 和 μ_n , 则有

$$\lambda_i + \mu_1 \geq \tilde{\lambda}_i \geq \lambda_i + \mu_n, \quad i = 1, 2, \dots, n.$$

(板书)

根据这个定理, 我们立即可以得到下面的 Weyl 定理.

定理 5.22 (Weyl) 设 $A \in \mathbb{R}^{n \times n}$ 和 $B = A + E \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 其特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$, 则

$$|\tilde{\lambda}_j - \lambda_j| \leq \|E\|_2, \quad j = 1, 2, \dots, n.$$

该定理的结论可以推广到奇异值情形. 我们首先给出下面的引理.

引理 5.23 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 的奇异值分解为 $A = U\Sigma V$, 其中 $U = [u_1, \dots, u_n] \in \mathbb{R}^{m \times n}$ 为列正交矩阵, $V = [v_1, \dots, v_n] \in \mathbb{R}^{n \times n}$ 为正交矩阵, $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$. 将 U 扩展成 $n \times n$ 的正交矩阵 $[U, \tilde{U}] = [u_1, \dots, u_n, \tilde{u}_1, \dots, \tilde{u}_{m-n}]$, 令

$$H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix} \in \mathbb{R}^{(m+n) \times (m+n)},$$

则 H 对称, 且特征值为 $\pm\sigma_i$ 和 0 (其中 0 至少为 $m-n$ 重特征值), 对应的特征向量分别为 $\frac{\sqrt{2}}{2} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix}$, $i = 1, 2, \dots, n$, 和 $\begin{bmatrix} 0 \\ \tilde{u}_j \end{bmatrix}$, $j = 1, 2, \dots, m-n$.

(留作课外自习, 直接代入验证即可)

由上面的引理和 Weyl 定理 5.22 立即可得

定理 5.24 设 $A, B \in \mathbb{R}^{m \times n}$ ($m \geq n$), 它们的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ 和 $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n$. 则

$$|\tilde{\sigma}_j - \sigma_j| \leq \|B - A\|_2, \quad j = 1, 2, \dots, n.$$

最后给出一个基于 F-范数的扰动性质 [63].

定理 5.25 设 $A, E \in \mathbb{C}^{n \times n}$ 且 A 是 Hermite 的, $A + E$ 是正规矩阵. 并设 A 的特征值满足

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n,$$

$A + E$ 的特征值 $\tilde{\lambda}_1, \tilde{\lambda}_2, \dots, \tilde{\lambda}_n$ 满足

$$\text{Re}(\tilde{\lambda}_1) \geq \text{Re}(\tilde{\lambda}_2) \geq \cdots \geq \text{Re}(\tilde{\lambda}_n).$$



则

$$\sum_{i=1}^n |\tilde{\lambda}_i - \lambda_i|^2 \leq \|E\|_F^2.$$

5.7.3 对称矩阵特征向量的扰动

定义 5.3 设 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, 则 λ_i 与其余特征值之间的**间隙 (gap)** 定义为

$$\text{gap}(\lambda_i, A) = \min_{j \neq i} |\lambda_j - \lambda_i|.$$

有时简记为 $\text{gap}(\lambda_i)$.

 特征向量的敏感性依赖于其对应的特征值的 gap, 一般来说, gap 越小, 特征向量越敏感.

例 5.2 设

$$A = \begin{bmatrix} 1+g & \\ & 1 \end{bmatrix}, \quad E = \begin{bmatrix} 0 & \varepsilon \\ \varepsilon & 0 \end{bmatrix}, \quad (0 < \varepsilon < g)$$

则 A 的特征值为 $\lambda_1 = 1+g$, $\lambda_2 = 1$, 对应的单位特征向量为 $q_1 = e_1$, $q_2 = e_2$. 当 ε 充分小时, $A+E$ 的特征值为 $\hat{\lambda}_{1,2} = 1 + (g \pm \sqrt{g^2 + 4\varepsilon^2})/2$, 对应的单位特征向量为

$$\begin{aligned} \hat{q}_1 &= \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{\sqrt{1+4\varepsilon^2/g^2}-1}{2\varepsilon/g} \end{bmatrix} = \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{\sqrt{(1+2\varepsilon^2/g^2)^2-4(\varepsilon/g)^4}-1}{2\varepsilon/g} \end{bmatrix} \\ &\approx \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{(1+2\varepsilon^2/g^2)-1}{2\varepsilon/g} \end{bmatrix} \\ &= \frac{1}{\sqrt{1+\varepsilon^2/g^2}} \begin{bmatrix} 1 \\ \varepsilon/g \end{bmatrix}, \\ \hat{q}_2 &= \beta_2 \cdot \begin{bmatrix} 1 \\ \frac{-\sqrt{1+4\varepsilon^2/g^2}-1}{2\varepsilon/g} \end{bmatrix} \approx \frac{1}{\sqrt{1+\varepsilon^2/g^2}} \begin{bmatrix} -\varepsilon/g \\ 1 \end{bmatrix}, \end{aligned}$$

其中 β_1, β_2 为规范化因子. 故特征向量的扰动约为 ε/g , 与特征值的间隙 $\text{gap}(\lambda_i, A) = g$ 成反比.

定理 5.26 设 $A = Q\Lambda Q^T$ 和 $A+E = \tilde{Q}\tilde{\Lambda}\tilde{Q}^T$ 分别为对称矩阵 $A \in \mathbb{R}^{n \times n}$ 和 $A+E \in \mathbb{R}^{n \times n}$ 的特征值分解, 其中 $Q = [q_1, q_2, \dots, q_n]$ 和 $\tilde{Q} = [\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_n]$ 均为正交矩阵, 且 $\tilde{q}_i$ 为 q_i 对应的扰动特征向量. 用 θ_i 表示 q_i 和 $\tilde{q}_i$ 之间的锐角, 则当 $\text{gap}(\lambda_i, A) > 0$ 时

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\|E\|_2}{\text{gap}(\lambda_i, A)}.$$



类似地, 当 $\text{gap}(\tilde{\lambda}_i, A + E) > 0$ 时

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\|E\|_2}{\text{gap}(\tilde{\lambda}_i, A + E)}.$$

(留作课外自习)

- 🔴 当 $\theta_i \ll 1$ 时, $\frac{1}{2} \sin 2\theta_i \approx \theta_i \approx \sin \theta_i$;
- 🔴 若 $\|E\|_2 \geq \frac{1}{2} \text{gap}(\lambda_i, A)$, 则定理中给出的上界就失去了实际意义;
- 🔴 在该定理中, 没有对特征值进行排序;
- 🔴 在实际计算中, 我们通常所知道的是 $\text{gap}(\tilde{\lambda}_i, A + E)$.

5.7.4 Rayleigh 商逼近

定理 5.27 设对称矩阵 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$,

(1) 若 $x \in \mathbb{R}^n$ 是单位向量, $\beta \in \mathbb{R}$, 则

$$\min_{1 \leq i \leq n} |\lambda_i - \beta| \leq \|Ax - \beta x\|_2; \quad (5.11)$$

(2) 对于给定的非零向量 $x \in \mathbb{R}^n$, 当 $\beta = \rho(x)$ 时, $\|Ax - \beta x\|_2$ 达到最小, 即

$$\min_{\beta \in \mathbb{R}} \|Ax - \beta x\|_2 = \|Ax - \rho(x)x\|_2; \quad (5.12)$$

(3) 令 $r = Ax - \rho(x)x$, 其中 $x \in \mathbb{R}^n$ 为单位向量. 设 λ_i 是距离 $\rho(x)$ 最近的特征值, $\text{gap}' = \min_{j \neq i} |\lambda_j - \rho(x)|$, θ 是 x 和 q_i 之间的锐角, 其中 q_i 是 λ_i 对应的单位特征向量, 则

$$\sin \theta \leq \frac{\|r\|_2}{\text{gap}'} \quad \text{且} \quad |\lambda_i - \rho(x)| \leq \frac{\|r\|_2^2}{\text{gap}'}. \quad (5.13)$$

(留作课外自习)

🔴 由 (5.11) 可知, 在幂迭代和反迭代中可以使用残量 $\|Ax - \tilde{\lambda}x\|_2 < \text{tol}$ 作为停机准则, 这里 $\tilde{\lambda}$ 是迭代过程中计算得到的近似特征值. 等式 (5.12) 则解释了为什么用 Rayleigh 商来近似特征值.

🔴 不等式 (5.13) 表明 $|\lambda_i - \rho(x)|$ 的值与残量范数 $\|r\|_2$ 的平方成正比, 这个结论是 Rayleigh 商迭代局部三次收敛的基础.

5.7.5 相对扰动分析

这里主要讨论 A 和 $X^T A X$ 的特征值和特征向量之间的扰动关系, 其中 X 非奇异且满足 $\|X^T X - I\|_2 = \varepsilon$. 这是因为在计算特征向量时, 由于舍入误差的原因, 最后得到的正交矩阵 Q 会带有误差, 从而失去正交性.



定理 5.28 (相对 Weyl 定理) 设对称矩阵 A 和 X^TAX 的特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$, 令 $\varepsilon = \|X^T X - I\|_2$, 则

$$|\tilde{\lambda}_i - \lambda_i| \leq \varepsilon |\lambda_i| \quad \text{或} \quad \frac{|\tilde{\lambda}_i - \lambda_i|}{|\lambda_i|} \leq \varepsilon \quad (\text{if } \lambda_i \neq 0).$$

(留作课外自习)

当 X 正交时, $\varepsilon = 0$, 故 X^TAX 与 A 有相同的特征值. 当 X 几乎正交时, ε 很小, 此时 X^TAX 与 A 的特征值几乎相同.

推论 5.29 设 G 和 Y^TGY 的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ 和 $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n$, 令 $\varepsilon = \max\{\|X^T X - I\|_2, \|Y^T Y - I\|_2\}$, 则

$$|\tilde{\sigma}_i - \sigma_i| \leq \varepsilon |\sigma_i| \quad \text{或} \quad \frac{|\tilde{\sigma}_i - \sigma_i|}{|\sigma_i|} \leq \varepsilon \quad (\text{if } \sigma_i \neq 0).$$

下面给出特征向量的相对扰动性质.

定义 5.4 设 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$, 若 $\lambda_i \neq 0$, 则 λ_i 与其余特征值之间的**相对间隙 (relative gap)** 定义为

$$\text{relgap}(\lambda_i, A) = \min_{j \neq i} \frac{|\lambda_j - \lambda_i|}{|\lambda_i|}.$$

定理 5.30 设 $A \in \mathbb{R}^{n \times n}$ 和 $X^TAX \in \mathbb{R}^{n \times n}$ 的特征值分解分别为 $A = Q\Lambda Q^T$ 和 $X^TAX = \tilde{Q}\tilde{\Lambda}\tilde{Q}^T$, 其中 $Q = [q_1, q_2, \dots, q_n]$ 和 $\tilde{Q} = [\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_n]$ 均为正交矩阵, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, $\tilde{\Lambda} = \text{diag}(\tilde{\lambda}_1, \tilde{\lambda}_2, \dots, \tilde{\lambda}_n)$ 且 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$. 设 θ_i 表示 q_i 和 $\tilde{q}_i$ 之间的锐角, 令 $\varepsilon_1 = \|I - X^{-T}X^{-1}\|_2$, $\varepsilon_2 = \|X - I\|_2$, 若 $\varepsilon_1 < 1$ 且 $\text{relgap}(\tilde{\lambda}_i, X^TAX) > 0$, 则

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\varepsilon_1}{1 - \varepsilon_1} \cdot \frac{1}{\text{relgap}(\tilde{\lambda}_i, X^TAX)} + \varepsilon_2.$$

(留作课外自习)

5.8 应用: SVD 与图像压缩

关于 SVD 的几何意义, 可以参考 AMS 的 Feature Column 的一篇文献: [We Recommend a Singular Value Decomposition](#), 2009.

用矩阵 A 表示图像, 然后求它的 SVD, 最后保留前 k 个奇异值, 即用 A 的截断 SVD 来近似 A , 从而达到图像压缩的目的.

例 5.3 举例: [LS_SVD_ImageCompress_01.m](#), [LS_SVD_ImageCompress_02.m](#)



5.9 课后习题

练习 5.1 设 $A \in \mathbb{R}^{n \times n}$, 若 (λ_1, x_1) 和 (λ_2, x_2) 是 A 的特征值对, 且 $\lambda_1 \neq \lambda_2$, 证明: x_1 和 x_2 线性无关. 进一步, 若 A 对称, 则 x_1 和 x_2 正交.

练习 5.2 设 $x, y \in \mathbb{R}^n$, 试证明: $\det(I + xy^T) = 1 + y^T x$. (注: 在复数域也成立)

练习 5.3 设 $A = D + uu^T$, 其中 $D = \text{diag}(d_1, d_2, \dots, d_n)$ 满足 $d_1 \geq d_2 \geq \dots \geq d_n$, $u = [u_1, u_2, \dots, u_n]^T \in \mathbb{R}^n$.

(1) 证明: d_i 是 A 的特征值的充要条件是 $d_i = d_{i+1}$ 或 $d_i = d_{i-1}$ 或 $u_i = 0$;

(2) 若 $u_i = 0$, 则 e_i 是与 d_i 对应的特征向量;

(3) 若 $d_{i-1} > d_i = d_{i+1} > d_{i+2}$ 且 $u_i \neq 0$, 证明: 对应于 $\lambda = d_i$ 的特征向量 x 除 x_i 和 x_{i+1} 外, 其余分量全部为 0, 且 $x_i u_i + x_{i+1} u_{i+1} = 0$.

思考: 如果 $d_{i-1} > d_i = d_{i+1} = d_{i+2} > d_{i+3}$, 则结论如何?

练习 5.4* 设 $\alpha \in \mathbb{R}$, $D = \text{diag}(d_1, d_2, \dots, d_n)$. 若 $u, \hat{u} \in \mathbb{R}^n$ 非负, 矩阵 $D + \alpha uu^T$ 和 $D + \hat{u}\hat{u}^T$ 具有相同的特征值, 记为 $\lambda_1, \lambda_2, \dots, \lambda_n$, 且满足交错性质 $\lambda_1 > d_1 > \lambda_2 > d_2 > \dots > \lambda_n > d_n$, 则它们具有相同的特征向量.

思考: 如果没有交错性质, 则结论如何?

练习 5.5* 设 $q \in \mathbb{R}^n$ 满足 $\|q\|_2 = 1$. 对任意与 q 正交的向量 $d \in \mathbb{R}^n$, 试证明:

$$\|(q+d)q^T - I\|_2 = \|q+d\|_2.$$

练习 5.6 设 $S \in \mathbb{C}^{n \times n}$ 是 skew-Hermite 矩阵, 即 $S^* = -S$. 证明:

(1) S 的非零特征值是纯虚数;

(2) $I + S$ 非奇异;

(3) 矩阵 $(I + S)^{-1}(I - S)$ 是酉矩阵. (该矩阵称为 S 的 Cayley 变换)

练习 5.7 设 $B \in \mathbb{R}^{m \times n}$, $m \geq n$ 且 $\|B\|_2 < 1$. 若 $A = \begin{bmatrix} I & B \\ B^T & I \end{bmatrix}$, 证明:

$$\kappa_2(A) = \frac{1 + \|B\|_2}{1 - \|B\|_2}.$$

练习 5.8* 设 B 是二对角矩阵

$$B = \begin{bmatrix} a_1 & b_1 & & & \\ & \ddots & \ddots & & \\ & & \ddots & b_{n-1} & \\ & & & a_n & \end{bmatrix}.$$

证明: 存在正交矩阵 Q_1 和 Q_2 , 使得 $Q_1^T B Q_2$ 仍然是二对角矩阵且所有元素都非负.

练习 5.9 设 $x, y \in \mathbb{R}^n$, 若 $y^T x$ 只有零特征值, 证明: xy^T 也只有零特征值.

设 $X, Y \in \mathbb{R}^{n \times 2}$, 若 $Y^T X$ 只有零特征值, 则 XY^T 是否也只有零特征值?

练习 5.10*(极分解) 设 $A \in \mathbb{C}^{n \times n}$. 证明:



(1) 存在酉矩阵 U 和唯一的 Hermite 半正定矩阵 P , 使得 $A = PU$.

(2) 进一步, 若 A 非奇异, 则 U 也唯一.

练习 5.11* 设 $A \in \mathbb{C}^{n \times n}$. 证明: A 可对角化当且仅当存在 Hermite 正定矩阵 P 使得 $P^{-1}AP$ 是正规矩阵.

(提示: 利用极分解, 但不是对 A 进行极分解)

练习 5.12 设 $A, B \in \mathbb{C}^{n \times n}$ 非奇异且 $A^*A = B^*B$. 证明: $A^{-1}B$ 是酉矩阵.

..... 以下为可选题

练习 5.13 设 $\lambda \in \mathbb{R}$ 是对称矩阵 $A \in \mathbb{R}^{n \times n}$ 的一个特征值, 对应的特征向量为 $x \in \mathbb{R}^n$. 若 $\tilde{x} \in \mathbb{R}^n$ 是 x 的一个 $\mathcal{O}(\varepsilon)$ 近似, 即 $\tilde{x} = x + \mathcal{O}(\varepsilon)$, 证明:

$$\frac{\tilde{x}^T A \tilde{x}}{\tilde{x}^T \tilde{x}} = \lambda + \mathcal{O}(\varepsilon^2),$$

即 $\tilde{x}$ 对应的 Rayleigh 商是 λ 的 $\mathcal{O}(\varepsilon^2)$ 逼近.

练习 5.14 设 $A, E \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 它们的特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\theta_1 \geq \theta_2 \geq \cdots \geq \theta_n$. 设 $A + E$ 的特征值为 $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_n$, 试证明

$$\lambda_i + \theta_n \leq \hat{\lambda}_i \leq \lambda_i + \theta_1, \quad i = 1, 2, \dots, n.$$

并由此可知, 若 E 对称正定, 则 $\hat{\lambda}_i \geq \lambda_i$.

练习 5.15 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, A_{n-1} 是 A 的 $n-1$ 阶顺序主子矩阵, 它们的特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\theta_1 \geq \theta_2 \geq \cdots \geq \theta_{n-1}$. 试证明

$$\lambda_1 \geq \theta_1 \geq \lambda_2 \geq \theta_2 \geq \cdots \geq \lambda_{n-1} \geq \theta_{n-1} \geq \lambda_n.$$

更一般地, 记 $\lambda_i(B)$ 为对称矩阵 B 的第 i 个特征值 (按降序排列), 设 A_k 是 A 任意一个 k 阶主子矩阵 ($1 \leq k \leq n-1$), 则有

$$\lambda_i(A) \geq \lambda_i(A_k) \geq \lambda_{n-k+i}(A), \quad i = 1, 2, \dots, k.$$

练习 5.16 设 $A = D + uu^T$, 其中 $D = \text{diag}(d_1, d_2, \dots, d_n)$ 满足 $d_1 > d_2 > \cdots > d_n$. 由此可知, A 的 $n-1$ 个特征值满足 $\lambda_{i+1} \in (d_{i+1}, d_i)$, $i = 1, 2, \dots, n-1$. 试给出特征值 λ_1 的取值范围.

练习 5.17* 证明以下结论:

(1) 设 $x \in \mathbb{R}^n$ 是一个正向量, 即 $x_i > 0$. 证明: 由 x 定义的 Cauchy 矩阵 A

$$a_{ij} = \frac{1}{x_i + x_j}$$

是对称半正定的. 进一步, 若 x_i 互不相等, 则 A 对称正定. (参见 [128])

(2) 证明: Hilbert 矩阵是对称正定的.

..... 以下为实践题

练习 5.18 编写程序, 实现对称矩阵 $A \in \mathbb{R}^{n \times n}$ 的三对角化.



练习 5.19 编写程序, 实现矩阵 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 的二对角化.

练习 5.20 编写程序, 实现计算对称三对角矩阵特征值的带 Wilkinson 位移的 QR 算法.

练习 5.21 编写程序, 实现计算二对角矩阵奇异值的 dqds 算法.



第六讲 线性方程组基本迭代法

考虑线性方程组

$$Ax = b, \quad A \in \mathbb{R}^{n \times n}, \quad b \in \mathbb{R}^n.$$

目前, 求解线性方程组的方法有:

- 直接法: PLU 分解, LDL^T 分解, Cholesky 分解等.
- 迭代法:
 - ▷ 基本迭代法: Jacobi, Gauss-Seidel, SOR, SSOR, Richardson, ADI 等.
 - ▷ Krylov 子空间迭代法: CG, MINRES, GMRES, BiCGStab 等.
- 快速方法:
 - ▷ 基于各种快速变换的求解方法, 如 FFT, DCT, DST 等; 代数多重网格法 (Algebraic multigrid); 快速多极子法 (Fast multipole), 等等.

 快速方法通常只适用于某些具有特殊结构的方程组.

 实际应用中, 这些方法常常结合使用, 如混合 (hybrid) 方法, 预处理 (preconditioning) 方法等.

直接法的优点是稳定可靠, 能在有限步内得到近似解 (如果不考虑舍入误差, 则得到精确解), 而且所需存储量和运算量都是可知的. 缺点是所需运算量约为 $\mathcal{O}(n^3)$, 这对于大规模线性方程组来说是非常巨大的. 而且在实际应用中, 很多问题中需要求解的大规模线性方程组都是稀疏的 (如偏微分方程的有限差分或有限元离散), 但对于一般的稀疏线性方程组, 直接法很难充分利用问题的稀疏性, 而迭代法可以很好地利用系数矩阵的稀疏性, 大幅降低运算量.

从历史上看, 最早的迭代法可以追溯到十九世纪 Gauss (1823), Jacobi (1845), Seidel (1874) 和 Nekrasov (1884) 等的工作 [12, 65]. 但是针对迭代法的系统研究主要还是在计算机出现以后, 大约是从二十世纪五十年代开始. 在开始阶段主要研究的是基本迭代法 [8, 23, 109], 典型代表有 Jacobi, GS, SOR, SSOR, ADI, Chebyshev 多项式迭代法等. 在这期间, 有两本非常有名的经典著作, 一本是 Varga 的 “Matrix Iterative Analysis” (1962) [109], 另一本是 Yong 的 “Iterative Solution of Large Linear Systems” (1971) [118]. 基本迭代法的收敛性有着非常完美的理论分析, 但在实际使用中却存在着许多不足, 比如收敛速度较慢, 最优参数估计困难, 等等.

从二十世纪七十年代中期开始, 研究重点慢慢转向 Krylov 子空间迭代法. 事实上, 早在 1952 年, Lanczos [73] 和 Hestenes & Stiefel [59] 就同时独立地提出了求解对称正定线性方程组的共轭梯度法 (CG). 对于一个 n 阶的线性方程组, 如果不考虑舍入误差的影响, 则共轭梯度法在 n 步后就一定会得到精确解. 因此共轭梯度法一开始被认作是直接法. 但在实际使用中发现, 由于舍入误差的影响, 迭代步数可能会超过 n , 特别是对于坏条件问题. 而对于条件数较小的线性方程组, 在给定精度下, 所需迭代步数则可能远远小于 n , 这使得共轭梯度法具有一定的吸引力. 但由

于种种原因 [47], 共轭梯度法提出后并没有受到重视, 在其出现后的近二十年里, 主流方法仍然是 Gauss 消去法, SOR 迭代法和 Chebyshev 迭代法.

1971 年, Reid [90] 指出, 对于好条件的大规模稀疏线性方程组, 共轭梯度法能在很少的迭代步数内得到一个很好的近似解 (事实上, Engeli 等 [35] 在 1959 年就发现了该现象, 但并没有引起关注). 1977 年, Meijerink 和 van der Vorst [79] 提出了不完全 Cholesky 分解 (incomplete Cholesky decomposition, IC), 并将其作为 CG 算法的预处理子 (preconditioner), 大幅提升了共轭梯度法的收敛速度, 这使得预处理 CG 算法成为了求解大规模稀疏对称正定线性方程组的首选方法. 同时, 这也极大地促发了大家对共轭梯度法的研究兴趣, 包括各种改进和推广, 如求解对称不定线性方程组的 MINRES 迭代法 (1975) 和 SYMMLQ 迭代法 (1975) [83], 求解一般线性方程组的 GMRES 迭代法 (1986) [94], QMR 迭代法 (1991) [41], BiCGSTAB 迭代法 (1992) [107], 等等. 目前, 带预处理的 Krylov 子空间迭代法已成为求解大规模稀疏线性方程组的主流方法.

本讲介绍常用的基本迭代法, 关于 Krylov 子空间迭代法, 我们将在下一讲介绍.

关于线性方程组基本迭代法的相关参考资料

- ▶ Varga, *Matrix Iterative Analysis*, 1962, 2nd edition, 2000. [109]
- ▶ Young, *Iterative Solution of Large Linear Systems*, 1971. [118]
- ▶ Golub and Van Loan, *Matrix Computations*, 1983, 4th edition, 2013. [49]
- ▶ Axelsson, *Iterative Solution Methods*, 1994. [8]
- ▶ 徐树方, *矩阵计算的理论与方法*, 1995. [122]
- ▷ Saad and van der Vorst, *Iterative solution of linear systems in the 20th century*, 2000. [97]
- ▷ Saad, *Iterative methods for linear systems of equations: A brief historical journey*, 2019. [96]

随着矩阵规模的增大, 直接法的运算量也随之快速增长. 对于大规模的线性方程组, 由于运算量太大, 往往会采用迭代法.

当直接求解 $Ax = b$ 比较困难时, 我们可以求解一个比较容易求解的近似等价方程组 $Mx = b$, 其中 M 可以看作是 A 在某种意义下的近似. 设 $Mx = b$ 的解为 $x^{(1)}$. 易知它与原方程组的解 $x_* = A^{-1}b$ 之间的差距满足

$$A(x_* - x^{(1)}) = b - Ax^{(1)}.$$

如果 $x^{(1)}$ 已经满足精度要求, 即非常接近真解 x_* , 则可以停止计算, 否则需要修正. 记 $\Delta x \triangleq x_* - x^{(1)}$, 则 Δx 满足方程 $A\Delta x = b - Ax^{(1)}$. 但由于直接求解该方程组比较困难 (与求解原方程组一样困难), 因此我们还是通过求解近似方程组

$$M\Delta x^{(1)} = b - Ax^{(1)},$$

得到一个修正量 $\Delta x^{(1)}$. 于是修正后的近似解为

$$x^{(2)} = x^{(1)} + \Delta x^{(1)} = x^{(1)} + M^{-1}(b - Ax^{(1)}).$$

如果 $x^{(2)}$ 已经满足精度要求, 则停止计算, 否则继续按以上的方式进行修正, 即求解 $M\Delta x^{(2)} =$



$b - Ax^{(2)}$ 得到修正量 $\Delta x^{(2)}$, 然后加到 $x^{(2)}$ 上得到 $x^{(3)}$:

$$x^{(3)} = x^{(2)} + \Delta x^{(2)} = x^{(2)} + M^{-1}(b - Ax^{(2)}).$$

不断重复以上步骤, 于是, 我们就得到一个序列

$$x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots$$

满足以下递推关系

$$x^{(k+1)} = x^{(k)} + M^{-1}(b - Ax^{(k)}), \quad k = 1, 2, \dots$$

这就构成了一个迭代法.

如何选择矩阵 M

通常, 构造一个好的迭代法, 需要考虑以下两点:

- (1) 以 M 为系数矩阵的线性方程组要比原线性方程组更容易求解;
- (2) M 应该是 A 的一个很好的近似, 而且迭代序列 $\{x^{(k)}\}$ 要快速收敛.

常用的基本迭代法主要包括:

- Richardson 迭代法
- Jacobi 迭代法
- Gauss-Seidel (G-S) 迭代法
- 超松弛 (SOR, Successive Over-Relaxation) 迭代法
- 对称超松弛 (SSOR, Symmetric SOR) 迭代法
- 加速超松弛 (AOR, Accelerated Over-Relaxation) 迭代法
- 多项式迭代法
- 交替方向 (ADI, Alternating Direction Implicit) 迭代法

6.1 矩阵分裂与迭代法

首先给出矩阵分裂的定义.

定义 6.1 (矩阵分裂 Matrix Splitting) 设 $A \in \mathbb{R}^{n \times n}$ 非奇异, 称

$$A = M - N \tag{6.1}$$

为 A 的一个矩阵分裂, 其中 M 非奇异.

考虑线性方程组

$$Ax = b, \tag{6.2}$$

其中 $A \in \mathbb{R}^{n \times n}$ 非奇异. 迭代法的基本思想: 给定一个迭代初始值 $x^{(0)}$, 通过一定的迭代格式生成一个迭代序列 $\{x^{(k)}\}_{k=0}^{\infty}$, 使得

$$\lim_{k \rightarrow \infty} x^{(k)} = x_* \triangleq A^{-1}b.$$



给定一个矩阵分裂 (6.1), 则原方程组 (6.2) 就等价于 $Mx = Nx + b$. 于是我们就可以构造出以下的迭代格式

$$Mx^{(k+1)} = Nx^{(k)} + b, \quad k = 0, 1, 2, \dots,$$

或

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b \triangleq Gx^{(k)} + g, \quad k = 0, 1, 2, \dots, \quad (6.3)$$

其中 $G \triangleq M^{-1}N$ 称为 **迭代矩阵**. 这就是基于矩阵分裂 (6.1) 的迭代法. 易知, 选取不同的 M , 就可以构造出不同的迭代法.

6.1.1 Jacobi 迭代法

将矩阵 A 写成

$$A = D - L - U,$$

其中 D 为 A 的对角线部分, $-L$ 和 $-U$ 分别为 A 的严格下三角和严格上三角部分.

在矩阵分裂 $A = M - N$ 中取 $M = D$, $N = L + U$, 则可得 **Jacobi 迭代法**:

$$x^{(k+1)} = D^{-1}(L + U)x^{(k)} + D^{-1}b, \quad k = 0, 1, 2, \dots \quad (6.4)$$

对应的迭代矩阵为

$$G_J = D^{-1}(L + U).$$

写成分量形式即为

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right), \quad i = 1, 2, \dots, n.$$

 由于 Jacobi 迭代法中 $x_i^{(k+1)}$ 的更新顺序与 i 无关, 即可以按顺序 $i = 1, 2, \dots, n$ 计算, 也可以按顺序 $i = n, n-1, \dots, 2, 1$ 计算, 或者乱序计算. 因此 Jacobi 迭代法非常适合并行计算.

算法 6.1 Jacobi 迭代法

- 1: Given an initial guess $x^{(0)}$
 - 2: **while** not converge **do**
 - 3: **for** $i = 1$ to n **do**
 - 4: $x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$
 - 5: **end for**
 - 6: **end while**
-

我们有时也将 Jacobi 迭代格式写为

$$x^{(k+1)} = x^{(k)} + D^{-1}(b - Ax^{(k)}) = x^{(k)} + D^{-1}r_k, \quad k = 0, 1, 2, \dots,$$



其中 $r_k \triangleq b - Ax^{(k)}$ 是 k 次迭代后的残量. 这表明, $x^{(k+1)}$ 是通过对 $x^{(k)}$ 做一个修正得到的.

6.1.2 Gauss-Seidel 迭代法

在分裂 $A = M - N$ 中取 $M = D - L$, $N = U$, 即可得 **Gauss-Seidel (G-S) 迭代法**:

$$x^{(k+1)} = (D - L)^{-1}Ux^{(k)} + (D - L)^{-1}b. \quad (6.5)$$

对应的迭代矩阵为

$$G_{GS} = (D - L)^{-1}U.$$

将 G-S 迭代法改写为

$$Dx^{(k+1)} = Lx^{(k+1)} + Ux^{(k)} + b,$$

即可得分量形式

$$\begin{aligned} x_i^{(k+1)} &= \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right), \\ &= x_i^{(k)} + \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^n a_{ij}x_j^{(k)} \right), \quad i = 1, 2, \dots, n. \end{aligned}$$

算法 6.2 Gauss-Seidel 迭代法

```

1: Given an initial guess  $x^{(0)}$ 
2: while not converge do
3:   for  $i = 1$  to  $n$  do
4:      $x_i^{(k+1)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$ 
5:   end for
6: end while

```

 G-S 迭代法的主要优点是充分利用了已经获得的最新数据.

 但在 G-S 迭代法中, 未知量的更新必须按自然顺序进行, 因此不适合并行计算.

6.1.3 SOR 迭代法

在 G-S 迭代法的基础上, 我们可以通过引入一个松弛参数 ω 来加快收敛速度. 这就是 SOR (Successive Overrelaxation) 方法 [117]. 该方法的基本思想是将 G-S 迭代法中的第 $k+1$ 步近似解与第 k 步近似解做一个加权平均, 从而给出一个新的近似解, 即

$$x^{(k+1)} = (1 - \omega)x^{(k)} + \omega D^{-1} \left(Lx^{(k+1)} + Ux^{(k)} + b \right). \quad (6.6)$$

整理后即为

$$x^{(k+1)} = (D - \omega L)^{-1}((1 - \omega)D + \omega U)x^{(k)} + \omega(D - \omega L)^{-1}b,$$



其中 ω 称为 **松弛参数** (relaxation parameter).

- 当 $\omega = 1$ 时, SOR 即为 G-S 迭代法,
- 当 $\omega < 1$ 时, 称为 **低松弛** (under relaxation) 迭代法,
- 当 $\omega > 1$ 时, 称为 **超松弛** (over relaxation) 迭代法.

在大多数情况下, 当 $\omega > 1$ 时会取得比较好的收敛效果.

 SOR 迭代法曾经在很长一段时间内是科学计算中求解线性方程组的首选方法.

SOR 的迭代矩阵为

$$G_{\text{SOR}} = (D - \omega L)^{-1} ((1 - \omega)D + \omega U),$$

对应的矩阵分裂为

$$M = \frac{1}{\omega}D - L, \quad N = \frac{1 - \omega}{\omega}D + U.$$

由 (6.6) 可知 SOR 迭代法的分量形式为

$$\begin{aligned} x_i^{(k+1)} &= (1 - \omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right) \\ &= x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^n a_{ij}x_j^{(k)} \right) \end{aligned}$$

算法 6.3 求解线性方程组的 SOR 迭代法

```

1: Given an initial guess  $x^{(0)}$  and parameter  $\omega$ 
2: while not converge do
3:   for  $i = 1$  to  $n$  do
4:      $x_i^{(k+1)} = (1 - \omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$ 
5:   end for
6: end while

```

 SOR 迭代法最大的优点是引入了松弛参数 ω : 通过选取适当的 ω 就可以大大提高方法的收敛速度. 但是 SOR 迭代法最大的难点就是如何选取最优的参数.

6.1.4 SSOR 迭代法

将 SOR 迭代法中的 L 和 U 相互交换位置, 则可得迭代格式

$$x^{(k+1)} = (D - \omega U)^{-1} ((1 - \omega)D + \omega L) x^{(k)} + \omega(D - \omega U)^{-1}b.$$

将这个迭代格式与 SOR 相结合, 就可以得到下面的两步迭代法

$$\begin{cases} x^{(k+\frac{1}{2})} = (D - \omega L)^{-1} [(1 - \omega)D + \omega U] x^{(k)} + \omega(D - \omega L)^{-1}b, \\ x^{(k+1)} = (D - \omega U)^{-1} [(1 - \omega)D + \omega L] x^{(k+\frac{1}{2})} + \omega(D - \omega U)^{-1}b. \end{cases}$$



这就是 **对称超松弛 (SSOR)** 迭代法, 相当于将 L 与 U 同等看待, 交替做两次 SOR 迭代法.

消去中间迭代向量 $x^{(k+\frac{1}{2})}$, 可得

$$x^{(k+1)} = G_{\text{SSOR}}x^{(k)} + g,$$

其中迭代矩阵

$$G_{\text{SSOR}} = (D - \omega U)^{-1}[(1 - \omega)D + \omega L](D - \omega L)^{-1}[(1 - \omega)D + \omega U].$$

对应的矩阵分裂为

$$\begin{aligned} M &= \frac{1}{\omega(2 - \omega)} [D - \omega(L + U) + \omega^2 LD^{-1}U] \\ &= \frac{1}{\omega(2 - \omega)} (D - \omega L)D^{-1}(D - \omega U), \\ N &= \frac{1}{\omega(2 - \omega)} [(1 - \omega)D + \omega L]D^{-1}[(1 - \omega)D + \omega U]. \end{aligned}$$

 对于某些特殊问题, SOR 迭代法不收敛, 但仍然可能构造出收敛的 SSOR 迭代法.

 一般来说, SOR 收敛速度要快于 SSOR, 但 SOR 的收敛速度对参数 ω 更敏感.

算法 6.4 SSOR 迭代法

```

1: Given an initial guess  $v^{(0)}$  and parameter  $\omega$ 
2: while not converge do
3:   for  $i = 1$  to  $n$  do
4:      $x_i^{(k+\frac{1}{2})} = (1 - \omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+\frac{1}{2})} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right)$ 
5:   end for
6:   for  $i = n$  to  $1$  do
7:      $x_i^{(k+1)} = (1 - \omega)x_i^{(k+\frac{1}{2})} + \frac{\omega}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+\frac{1}{2})} - \sum_{j=i+1}^n a_{ij}x_j^{(k+1)} \right)$ 
8:   end for
9: end while

```

6.1.5 AOR 迭代法

Hadjidimos 于 1978 年提出了 **加速超松弛 (Accelerated Over-Relaxation, AOR)** 迭代法, 迭代矩阵为

$$G_{\text{AOR}} = (D - \gamma L)^{-1}[(1 - \omega)D + (\omega - \gamma)L + \omega U],$$

其中 γ 和 ω 为松弛参数. 对应的矩阵分裂为

$$M = \frac{1}{\omega}(D - \gamma L), \quad N = \frac{1}{\omega}[(1 - \omega)D + (\omega - \gamma)L + \omega U].$$

易知:

- (1) 当 $\gamma = \omega$ 时, AOR 即为 SOR 迭代法;
- (2) 当 $\gamma = \omega = 1$ 时, AOR 即为 G-S 迭代法;



(3) 当 $\gamma = 0, \omega = 1$ 时, AOR 即为 Jacobi 迭代法.

✎ AOR 迭代法中含有两个参数. 因此在理论上, 通过选取合适的参数, AOR 迭代法会收敛得更快. 但也是因为含有两个参数, 使得参数的选取变得更加困难, 因此较少使用.

✎ 与 SSOR 类似, 我们也可以定义 SAOR 迭代法.

6.1.6 Richardson 迭代法

Richardson 迭代法是一类形式非常简单的算法, 其迭代格式为

$$x^{(k+1)} = x^{(k)} + \omega(b - Ax^{(k)}), \quad k = 0, 1, 2, \dots$$

它可以看作是基于一矩阵分裂的迭代法:

$$M = \frac{1}{\omega}I, \quad N = \frac{1}{\omega}I - A.$$

对应的迭代矩阵为

$$G_R = I - \omega A.$$

如果在每次迭代时选取不同的参数, 即

$$x^{(k+1)} = x^{(k)} + \omega_k(b - Ax^{(k)}), \quad k = 0, 1, 2, \dots,$$

则有

$$r_{k+1} = b - Ax^{(k+1)} = (I - \omega_k A)r_k,$$

因此可得

$$r_{k+1} = (I - \omega_k A)(I - \omega_{k-1} A) \cdots (I - \omega_0 A)r_0.$$

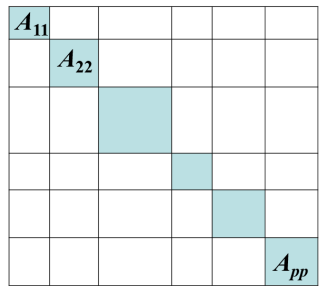
这就构成了 **多项式** 迭代法.

6.1.7 分块迭代法

如果 A 的对角线中出现零, 则 Jacobi, G-S, SOR 等方法就不再有定义. 此时我们可以采用分块迭代格式. 另外, 分块迭代法也能提升算法的计算效率.

将 A 写成如下的分块形式 (如右图所示):

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1p} \\ A_{21} & A_{22} & \cdots & A_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ A_{p1} & A_{p2} & \cdots & A_{pp} \end{bmatrix},$$



则相应的分块 Jacobi, 分块 G-S 和分块 SOR 迭代法分别为:

- 分块 Jacobi 迭代:

$$A_{ii}x_i^{(k+1)} = b_i - \sum_{j=1, j \neq i}^p A_{ij}x_j^{(k)}, \quad i = 1, 2, \dots, p.$$



- 分块 Gauss-seidel 迭代:

$$A_{ii}x_i^{(k+1)} = b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{(k+1)} - \sum_{j=i+1}^p A_{ij}x_j^{(k)}, \quad i = 1, 2, \dots, p.$$

- 分块 SOR 迭代:

$$x_i^{(k+1)} = (1 - \omega)x_i^{(k)} + \omega A_{ii}^{-1} \left(b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{(k+1)} - \sum_{j=i+1}^p A_{ij}x_j^{(k)} \right),$$

$$i = 1, 2, \dots, p.$$

 分块迭代法采用更多的 2 级 BLAS 运算, 因此更有利于发挥现代计算机的性能.

6.2 收敛性分析

6.2.1 基本概念

首先给出向量序列收敛的定义.

定义 6.2 (向量序列的收敛) 设 $\{x^{(k)}\}_{k=0}^{\infty}$ 是 $\mathbb{R}^n$ (或 $\mathbb{C}^n$) 中的一个向量序列. 如果存在向量 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$ (或 $\mathbb{C}^n$) 使得

$$\lim_{k \rightarrow \infty} x_i^{(k)} = x_i, \quad i = 1, 2, \dots, n,$$

其中 $x_i^{(k)}$ 表示 $x^{(k)}$ 的第 i 个分量, 则称 $\{x^{(k)}\}$ (按分量) 收敛到 x , 即 x 是 $x^{(k)}$ 的极限, 记为

$$\lim_{k \rightarrow \infty} x^{(k)} = x.$$

类似地, 我们可以给出矩阵序列收敛的定义.

定义 6.3 (矩阵序列的收敛) 设 $\{A^{(k)} = [a_{ij}^{(k)}]\}_{k=0}^{\infty}$ 是 $\mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 中的一个矩阵序列. 如果存在矩阵 $A = [a_{ij}] \in \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$) 使得

$$\lim_{k \rightarrow \infty} a_{ij}^{(k)} = a_{ij}, \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, n,$$

则称 $A^{(k)}$ 收敛到 A , 即 A 是 $A^{(k)}$ 的极限, 记为

$$\lim_{k \rightarrow \infty} A^{(k)} = A.$$

关于向量序列和矩阵序列的收敛性, 我们有下面的基本判别方法 [124].

定理 6.1 设向量序列 $\{x^{(k)}\}_{k=0}^{\infty} \subset \mathbb{R}^n$ (或 $\mathbb{C}^n$), 矩阵序列 $\{A^{(k)} = [a_{ij}^{(k)}]\}_{k=0}^{\infty} \subset \mathbb{R}^{m \times n}$ (或 $\mathbb{C}^{m \times n}$), 则

- (1) $\lim_{k \rightarrow \infty} x^{(k)} = x \iff \lim_{k \rightarrow \infty} \|x^{(k)} - x\| = 0$, 其中 $\|\cdot\|$ 为任一向量范数;



- (2) $\lim_{k \rightarrow \infty} A^{(k)} = A \iff \lim_{k \rightarrow \infty} \|A^{(k)} - A\| = 0$, 其中 $\|\cdot\|$ 为任一矩阵范数;
 (3) $\lim_{k \rightarrow \infty} A^{(k)} = 0 \iff \lim_{k \rightarrow \infty} A^{(k)}x = 0, \forall x \in \mathbb{R}^n$ (或 $\mathbb{C}^n$).

由定理 6.1 和引理 1.39, 我们可以立即得到下面的结论.

定理 6.2 设矩阵 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), 则 $\lim_{k \rightarrow \infty} A^k = 0$ 当且仅当 $\rho(A) < 1$. (板书)

下面给出谱半径与范数之间的一个重要关系式, 有时也用来定义矩阵的谱半径.

定理 6.3 设 $A \in \mathbb{R}^{n \times n}$ (或 $\mathbb{C}^{n \times n}$), 则对任意矩阵范数 $\|\cdot\|$, 有

$$\rho(A) = \lim_{k \rightarrow \infty} \|A^k\|^{\frac{1}{k}}.$$

(板书)

下面给出基本迭代法收敛的定义.

定义 6.4 考虑基本迭代法 (6.3), 如果对任意的初始向量 $x^{(0)}$, 都有

$$\lim_{k \rightarrow \infty} x^{(k)} \rightarrow x_*,$$

则称基本迭代法 (6.3) 是**收敛**的, 否则就称其为**发散**的.

 基于矩阵分裂的迭代法, 其收敛性取决于迭代矩阵的谱半径.

下面考虑基本迭代法 (6.3) 的收敛性. 首先给出一个迭代法收敛的充分条件.

引理 6.4 若存在算子范数 $\|\cdot\|$, 使得 $\|G\| < 1$, 则迭代法 (6.3) 收敛.

(板书)

 事实上, 由习题 1.10 可知, 引理 6.4 条件中的算子范数可以改为任意矩阵范数.

 我们记 $e^{(k)} \triangleq x^{(k)} - x_*$ 为第 k 步迭代解 $x^{(k)}$ 的**误差向量**.

定理 6.5 (基本收敛性定理) 对任意初始向量 $x^{(0)}$, 迭代法 (6.3) 收敛的充要条件是 $\rho(G) < 1$.

(板书)

定义 6.5 设 G 是迭代矩阵, 则迭代法 (6.3) 的**平均收敛速度**定义为

$$R_k(G) \triangleq -\ln \|G^k\|^{\frac{1}{k}},$$



渐进收敛速度 定义为

$$R(G) \triangleq \lim_{k \rightarrow \infty} R_k(G) = -\ln \rho(G).$$

 平均收敛速度与迭代步数和所用的范数有关, 但渐进收敛速度只依赖于迭代矩阵的谱半径.

 数值分析中关于 **收敛速度** 的定义:

设点列 $\{\varepsilon_k\}_{k=1}^{\infty}$ 收敛, 且 $\lim_{k \rightarrow \infty} \varepsilon_k = 0$. 若存在一个有界常数 $0 < c < \infty$, 使得

$$\lim_{k \rightarrow \infty} \frac{|\varepsilon_{k+1}|}{|\varepsilon_k|^p} = c,$$

则称点列 $\{\varepsilon_k\}$ 是 **p 次 (渐进) 收敛** 的. 若 $1 < p < 2$ 或 $p = 1$ 且 $c = 0$, 则称点列是**超线性收敛**的.

定理 6.6 考虑迭代法 (6.3). 如果存在某个算子范数 $\|\cdot\|$ 使得 $\|G\| = q < 1$, 则

- (1) $\|x^{(k)} - x_*\| \leq q^k \|x^{(0)} - x_*\|$;
- (2) $\|x^{(k)} - x_*\| \leq \frac{q}{1-q} \|x^{(k)} - x^{(k-1)}\|$;
- (3) $\|x^{(k)} - x_*\| \leq \frac{q^k}{1-q} \|x^{(1)} - x^{(0)}\|$.

(板书)

 一般来说, 好的迭代法应该满足 [23]:

- (1) $\rho(G)$ 很小;
- (2) 以 M 为系数矩阵的线性方程组比较容易求解.

6.2.2 不可约对角占优矩阵

这里我们考虑 A 是 (行) 严格对角占优或不可约 (行) 对角占优情形.

定理 6.7 设 $A \in \mathbb{R}^{n \times n}$, 若 A 严格对角占优, 则 Jacobi 迭代法和 G-S 迭代法都收敛, 且

$$\|G_{GS}\|_{\infty} \leq \|G_J\|_{\infty} < 1.$$

(板书)

定理 6.8 设 $A \in \mathbb{R}^{n \times n}$, 若 A 不可约对角占优, 则 Jacobi 迭代法和 G-S 迭代法都收敛. 进一



步, 若 A 是非负矩阵, 则

$$\rho(G_{GS}) < \rho(G_J) < 1.$$

(留作课外自习, 需利用非负矩阵的相关知识, 可参见 [109])

 **思考:** 若 $A \in \mathbb{R}^{n \times n}$ 严格对角占优且非负, 则是否有结论 $\rho(G_{GS}) \leq \rho(G_J)$?

 上述定理中的结论对一般矩阵并不成立: 对某些矩阵, Jacobi 迭代法收敛, 但 G-S 迭代法却不一定收敛, 见思考题 6.23.

关于 SOR 迭代法, 我们有下面的结论.

定理 6.9 设 $A \in \mathbb{R}^{n \times n}$, 若 A 严格对角占优且 $0 < \omega \leq 1$, 则 SOR 迭代法收敛.

(板书)

定理 6.10 设 $A \in \mathbb{R}^{n \times n}$, 若 A 不可约对角占优且 $0 < \omega \leq 1$, 则 SOR 迭代法收敛.

(留作课外自习)

 **注意:** 当 A 是严格列对角占优或不可约列对角占优时, 也有相应的结论成立.

6.2.3 对称正定矩阵

首先给出 Richardson 迭代法的收敛性.

定理 6.11 设 $A \in \mathbb{R}^{n \times n}$ 是对称正定矩阵, λ_1 和 λ_n 分别是 A 的最大和最小特征值, 则 Richardson 迭代法收敛当且仅当

$$0 < \omega < \frac{2}{\lambda_1}.$$

另外, Richardson 迭代法的最优参数为

$$\omega_* = \arg \min_{\omega} \rho(G_R) = \frac{2}{\lambda_1 + \lambda_n},$$

即当 $\omega = \omega_*$ 时, 迭代矩阵的谱半径达到最小, 并且有

$$\rho(G_R) = \begin{cases} 1 - \omega\lambda_n, & \text{if } \omega \leq \omega_*, \\ \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} = \frac{\kappa(A) - 1}{\kappa(A) + 1}, & \text{if } \omega = \omega_*, \\ \omega\lambda_1 - 1, & \text{if } \omega \geq \omega_*. \end{cases}$$

(留作练习)

下面考虑 Jacobi, Gauss-Seidel 和 SOR 的收敛性, 先介绍两个引理.



引理 6.12 设 $A \in \mathbb{C}^{n \times n}$ 是 Hermite 的, 矩阵分裂 $A = M - N$, 其中 M 非奇异, 则 $M^* + N$ 也是 Hermite 的, 且对任意 $x \in \mathbb{C}^n$ 有

$$x^* Ax - \tilde{x}^* A \tilde{x} = u^* (M^* + N) u,$$

其中 $\tilde{x} = M^{-1}Nx$, $u = x - \tilde{x}$.

(板书)

引理 6.13 设 $A \in \mathbb{R}^{n \times n}$ 对称, 矩阵分裂 $A = M - N$ 满足 $M^T + N$ 正定, 则 $\rho(M^{-1}N) < 1$ 当且仅当 A 是正定的.

(板书)

我们首先给出 SOR 迭代法收敛的一个必要条件. 这个结果是由 Kahan 在其博士论文中给出的.

定理 6.14 (Kahan [69]) 对于 SOR 迭代法, 有 $\rho(G_{\text{SOR}}) \geq |1 - \omega|$, 故 SOR 迭代法收敛的必要条件是 $0 < \omega < 2$.

(板书)

定理 6.15 设 $A \in \mathbb{R}^{n \times n}$ 对称正定.

- (1) 若 $2D - A$ 正定, 则 Jacobi 迭代法收敛.
- (2) 若 $0 < \omega < 2$, 则 SOR 迭代法和 SSOR 迭代法收敛.
- (3) G-S 迭代法收敛.

(留作课外自习, 直接利用前面的引理即可)

定理 6.16 设 $A \in \mathbb{R}^{n \times n}$ 对称, 且 D 正定.

- (1) 若 Jacobi 迭代法收敛, 则 A 和 $2D - A$ 都正定;
- (2) 若存在 $\omega \in (0, 2)$ 使得 SOR (或 SSOR) 迭代法收敛, 则 A 正定;
- (3) 若 G-S 迭代法收敛, 则 A 正定.

(板书, 只证明 (1), 结论 (2) 和 (3) 利用引理 6.13 即可)

推论 6.17 设 $A \in \mathbb{R}^{n \times n}$ 对称且 D 正定, 则

- (1) Jacobi 收敛的充要条件是 A 和 $2D - A$ 都正定;
- (2) G-S 收敛的充要条件是 A 正定;
- (3) SOR 收敛的充要条件是 A 正定且 $\omega \in (0, 2)$.

6.2.4 相容次序矩阵

针对一类特殊的矩阵, Jacobi 迭代矩阵和 SOR 迭代矩阵的特征值之间存在一种特殊的关系式, 根据这个关系式, 在某些场合下就可以确定 SOR 的最优参数的取值.



定义 6.6 设 $A \in \mathbb{R}^{n \times n}$ 的对角线元素全不为零, 记 $\tilde{L} = D^{-1}L$, $\tilde{U} = D^{-1}U$, 即 $A = D(I - \tilde{L} - \tilde{U})$. 若矩阵 $G(\alpha) \triangleq \alpha\tilde{L} + \frac{1}{\alpha}\tilde{U}$ 的特征值与 α 无关 ($\alpha \neq 0$), 则称 A 是 **相容次序矩阵** (consistently ordered matrix).

下面我们讨论一类特殊的相容次序矩阵. 首先给出一个引理.

引理 6.18 设 $B \in \mathbb{R}^{n \times n}$ 具有下面的结构

$$B = \begin{bmatrix} 0 & B_{12} \\ B_{21} & 0 \end{bmatrix}, \quad \text{其中 } B_{12} \in \mathbb{R}^{k \times (n-k)}, B_{21} \in \mathbb{R}^{(n-k) \times k}, \quad 0 \leq k \leq n.$$

令 B_L 和 B_U 分别表示 B 的下三角和上三角部分, 则下面的结论成立:

- (1) 若 μ 是 B 的特征值, 则 $-\mu$ 也是 B 的特征值;
- (2) $B(\alpha)$ 的特征值与 α 无关, 其中

$$B(\alpha) = \alpha B_L + \frac{1}{\alpha} B_U, \quad \alpha \neq 0.$$

(板书)

该结论对复矩阵也成立.

由引理 6.18 中结论 (2) 可知, $B(\alpha) + \beta I$ 的特征值也与 α 无关, 其中 β 为任意常数.

定义 6.7 设 $A \in \mathbb{R}^{n \times n}$, 如果存在一个置换矩阵 P , 使得

$$PAP^T = \begin{bmatrix} D_1 & F \\ E & D_2 \end{bmatrix}, \quad (6.7)$$

其中 D_1, D_2 为对角矩阵, 则称 A 具有 **性质 A**.

性质 A 意味着指标集 $\mathbb{Z}_n = \{1, 2, \dots, n\}$ 可以分成两个互不相交的子集之和, 在同一个子集内的未知量相互独立, 因此在迭代更新时可以并行计算. (也就是说, $\mathbb{Z}_n = \mathcal{S} \cup \mathcal{T}$, $\mathcal{S} \cap \mathcal{T} = \emptyset$, 当 $i \neq j$ 且 $i, j \in \mathcal{S}$ 或 $i, j \in \mathcal{T}$ 时, 有 $a_{ij} = 0$.)

相容次序矩阵和性质 A 有不同的定义方式, 我们这里采用的是 Demmel (1997) [23] 中的定义. 其他定义可参见 Varga (1962, 2000) [109] 或 Young (1971) [118]. 事实上, 根据 Axelsson (1994) [8, page 239] 中的说法, Yong (1950) [117] 也是按照定义 6.6 和定义 6.7 来定义相容次序矩阵和性质 A 的.

根据引理 6.18, 我们可以直接得到下面的结论.

定理 6.19 设 A 的对角线元素全不为零, 若 A 具有性质 A, 则存在置换矩阵 P , 使得 PAP^T 是相容次序矩阵.



如果矩阵 A 本身就具有 (6.7) 的形式, 则 A 就是相容次序矩阵.

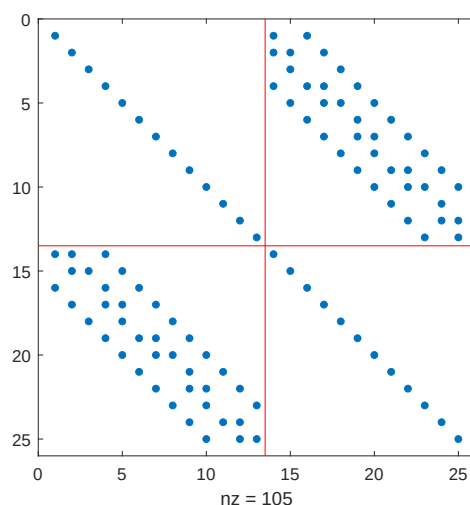
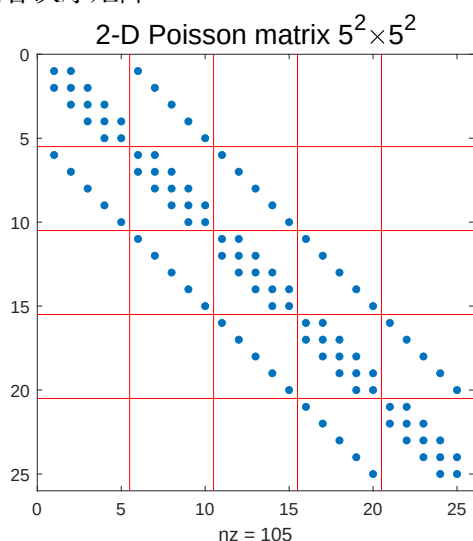
事实上, (6.7) 可推广到分块三对角矩阵情形, 见下面的例子.

例 6.1 设 D_i 是非奇异的对角矩阵, 则任意块三对角矩阵

$$\begin{bmatrix} D_1 & A_1 & & \\ B_1 & \ddots & \ddots & \\ & \ddots & \ddots & A_{N-1} \\ & & B_{N-1} & D_N \end{bmatrix}$$

都是相容次序矩阵.

(留作练习, 参见 [118])



用有限差分法离散的二维偏微分方程 (左图是网格点自然排序, 右图是红黑排序)

定理 6.20 设 A 是相容次序矩阵且 $\omega \neq 0$, 则下列命题成立

(1) Jacobi 迭代矩阵 G_J 的特征值正负成对出现;

(2) 若 μ 是 G_J 的一个特征值且 λ 满足

$$(\lambda + \omega - 1)^2 = \lambda \omega^2 \mu^2, \quad (6.8)$$

则 λ 是 SOR 迭代矩阵 G_{SOR} 的特征值;

(3) 反之, 若 $\lambda \neq 0$ 是 G_{SOR} 的一个特征值且 μ 满足 (6.8), 则 μ 是 G_J 的特征值.

(板书)

推论 6.21 设 A 是相容次序矩阵. 若 G_J 的特征值全部为实数, 则 SOR 迭代法收敛的充要条件是 $0 < \omega < 2$ 且 $\rho(G_J) < 1$.

(留作练习)

推论 6.22 若 A 是相容次序矩阵, 则 $\rho(G_{\text{GS}}) = \rho(G_J)^2$, 即当 Jacobi 迭代法收敛时, Gauss-Seidel 迭代法比 Jacobi 迭代法快一倍.

下面是关于 SOR 迭代法的最优参数的取值 [117].

定理 6.23 设 A 是相容次序矩阵. 若 G_J 的特征值全部为实数, 且 $\rho_J \triangleq \rho(G_J) < 1$, 则 SOR 迭代法的最优参数为

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho_J^2}},$$

此时

$$\rho(G_{SOR}) = \omega_{opt} - 1 = \frac{\rho_J^2}{\left(1 + \sqrt{1 - \rho_J^2}\right)^2}.$$

进一步, 有

$$\rho(G_{SOR}) = \begin{cases} \omega - 1, & \omega_{opt} \leq \omega \leq 2 \\ 1 - \omega + \frac{1}{2}\omega^2\rho_J^2 + \omega\rho_J\sqrt{1 - \omega + \frac{1}{4}\omega^2\rho_J^2}, & 0 < \omega \leq \omega_{opt} \end{cases}.$$

(留作课外自习, 直接求解等式 (6.8), 分情况讨论即可, 可参见 [118, page 173])

6.3 应用: Poisson 方程求解

Poisson 方程是一类典型的偏微分方程, 经过差分离散后, 转化为一个线性方程组. 我们以这个线性方程组为例, 讨论 Jacobi, G-S 和 SOR 迭代法的相关性质.

6.3.1 一维 Poisson 方程

首先介绍一维 Poisson 方程的离散. 考虑如下带 Dirichlet 边界条件的一维 Poisson 方程

$$\begin{cases} -\frac{d^2u(x)}{dx^2} = f(x), & 0 < x < 1, \\ u(0) = a, u(1) = b, \end{cases} \quad (6.9)$$

其中 $f(x)$ 是给定的函数, $u(x)$ 是需要计算的未知函数.

差分离散

取步长 $h = \frac{1}{n+1}$, 网格节点为 $x_i = ih, i = 0, 1, 2, \dots, n+1$. 我们采用二阶中心差分来近似二阶导数, 即

$$-\frac{d^2u(x)}{dx^2} \Big|_{x_i} = \frac{2u(x_i) - u(x_{i-1}) - u(x_{i+1}))}{h^2} + O\left(h^2 \left\| \frac{d^4u}{dx^4} \right\|_{\infty}\right), \quad i = 1, 2, \dots, n.$$

将其代入 (6.9), 舍去高阶项后得到 Poisson 方程在 x_i 点的近似离散方程

$$-u_{i-1} + 2u_i - u_{i+1} = h^2 f_i,$$

其中 $f_i = f(x_i)$, u_i 为 $u(x_i)$ 的近似. 令 $i = 1, 2, \dots, n$, 则可得 n 个线性方程. 写成矩阵形式为

$$T_n u = f, \quad (6.10)$$



其中

$$T_n = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \triangleq \text{tridiag}(-1, 2, -1), \quad u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix}, \quad f = \begin{bmatrix} f_1 + u_0 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n + u_{n+1} \end{bmatrix}. \quad (6.11)$$

由边界条件可知 $u_0 = a, u_{n+1} = b$.

系数矩阵 T_n 的性质

易知 T_n 是三对角对称矩阵, 因此其特征值都是实数. 事实上, 我们有下面的结论.

引理 6.24 T_n 的特征值和对应的特征向量分别为

$$\lambda_k = 2 - 2 \cos \frac{k\pi}{n+1},$$

$$z_k = \sqrt{\frac{2}{n+1}} \left[\sin \frac{k\pi}{n+1}, \sin \frac{2k\pi}{n+1}, \dots, \sin \frac{nk\pi}{n+1} \right]^T, \quad k = 1, 2, \dots, n,$$

即 $T_n = Z\Lambda Z^T$, 其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 是对角矩阵, $Z = [z_1, z_2, \dots, z_n]$ 是正交矩阵.

(留作课外自习, 直接代入验证即可)

由此可知, T_n 是对称正定的. 更一般地, 我们有下面的结论.

推论 6.25 设 $T = \text{tridiag}(a, b, c) \in \mathbb{R}^{n \times n}$, $ac > 0$, 则 T 的特征值为

$$\lambda_k = b - 2\sqrt{ac} \cos \frac{k\pi}{n+1}, \quad k = 1, 2, \dots, n,$$

对应的特征向量为 z_k , 其第 j 个分量为

$$z_k(j) = \left(\frac{a}{c}\right)^{\frac{j}{2}} \sin \frac{jk\pi}{n+1}.$$

特别地, 若 $a = c = 1$, 则对应的单位特征向量为

$$z_k = \sqrt{\frac{2}{n+1}} \left[\sin \frac{k\pi}{n+1}, \sin \frac{2k\pi}{n+1}, \dots, \sin \frac{nk\pi}{n+1} \right]^T.$$

(留作练习)

由引理 6.24 可知, T_n 的最大特征值为

$$2 \left(1 - \cos \frac{n\pi}{n+1} \right) = 4 \sin^2 \frac{n\pi}{2(n+1)} \approx 4,$$

最小特征值为

$$2 \left(1 - \cos \frac{\pi}{n+1} \right) = 4 \sin^2 \frac{\pi}{2(n+1)} \approx \left(\frac{\pi}{n+1} \right)^2.$$

因此, 当 n 很大时, T_n 的谱条件数约为

$$\kappa_2(T_n) \approx \frac{4(n+1)^2}{\pi^2}.$$



矩阵 T_n 有时也称为 **二阶差分矩阵**, 可以写成一阶差分矩阵的乘积, 即 $T_n = DD^T$, 其中

$$D = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n \times (n+1)}.$$

其中 D 是 **一阶差分矩阵**. 需要注意的是, D 不是方阵, 所以无法像 LU 分解那样直接用来求解 $T_n u = f$, 但通过适当变换, 可构造出 **变系数追赶法**, 感兴趣的读者可参见 [?].

6.3.2 二维 Poisson 方程

考虑二维 Poisson 方程的离散. 同样是带 Dirichlet 边界条件, 即

$$\begin{cases} -\Delta u(x, y) = -\frac{\partial^2 u(x, y)}{\partial x^2} - \frac{\partial^2 u(x, y)}{\partial y^2} = f(x, y), & (x, y) \in \Omega, \\ u(x, y) = u_0(x, y), & (x, y) \in \partial\Omega, \end{cases} \quad (6.12)$$

其中 $\Omega = [0, 1] \times [0, 1]$ 为求解区域, $\partial\Omega$ 表示 Ω 的边界.

五点差分离散

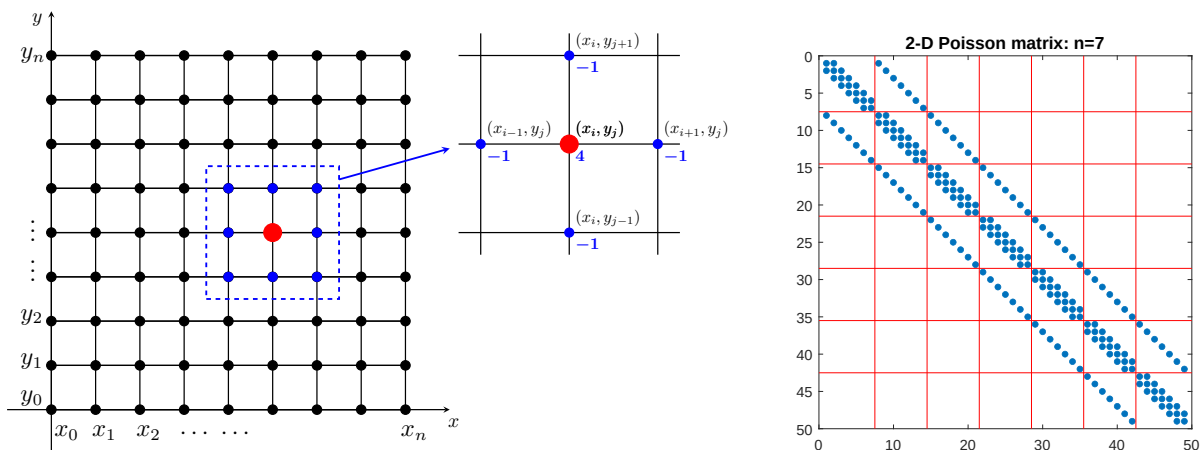
为了简单起见, 我们在 x -方向和 y -方向取相同的步长 $h = \frac{1}{n+1}$, 网格节点为 $x_i = ih$, $y_j = jh$, $i, j = 0, 1, 2, \dots, n+1$. 在 x -方向和 y -方向同时采用二阶中心差分近似, 可得

$$\begin{aligned} -\frac{\partial^2 u(x, y)}{\partial x^2} \Big|_{(x_i, y_j)} &\approx \frac{2u(x_i, y_j) - u(x_{i-1}, y_j) - u(x_{i+1}, y_j)}{h^2}, \\ -\frac{\partial^2 u(x, y)}{\partial y^2} \Big|_{(x_i, y_j)} &\approx \frac{2u(x_i, y_j) - u(x_i, y_{j-1}) - u(x_i, y_{j+1})}{h^2}. \end{aligned}$$

代入 (6.12) 后, 就可以得到二维 Poisson 方程在 (x_i, y_j) 点的近似离散方程

$$4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = h^2 f_{i,j},$$

其中 $f_{i,j} = f(x_i, y_j)$, $u_{i,j}$ 为 $u(x_i, y_j)$ 的近似. 这个离散格式可以用下面的图 (左图) 来描述.



按自然顺序排列以上方程 (即先沿 x -方向从左到右, 然后沿 y 方向从下往上), 可得矩阵形式

$$T_N u = h^2 f, \quad (6.13)$$

其中

$$T_N \triangleq I \otimes T_n + T_n \otimes I, \quad u = [u_{1,1}, \dots, u_{n,1}, u_{1,2}, \dots, u_{n,2}, \dots, u_{1,n}, \dots, u_{n,n}]^T.$$

这里 $\otimes$ 表示 Kronecker 乘积, T_n 为一维 Poisson 方程离散后的系数矩阵, 即 (6.11). 上图 (右图) 为 $N = 7^2$ 时的矩阵示例图.

需要注意的是, 系数矩阵 T_N 与网格点的排序有关, 不同的排序方式会得到不同的 T_N .

类似地, 如果对三维 Poisson 方程进行中心差分离散, 则对应的系数矩阵为

$$T_n \otimes I \otimes I + I \otimes T_n \otimes I + I \otimes I \otimes T_n.$$

系数矩阵 T_N 的性质

定理 6.26 设 $T_n = Z \Lambda Z^T$, 其中 $Z = [z_1, z_2, \dots, z_n]$ 为正交阵, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 为对角阵, 则 T_N 的特征值分解为

$$T_N = (Z \otimes Z)(I \otimes \Lambda + \Lambda \otimes I)(Z \otimes Z)^T,$$

即 T_N 的特征值为 $\lambda_i + \lambda_j$, 对应的特征向量为 $z_i \otimes z_j$, $i, j = 1, 2, \dots, n$.

由于 T_N 对称正定, 所以其条件数为

$$\kappa(T_N) = \frac{\lambda_{\max}(T_N)}{\lambda_{\min}(T_N)} = \frac{1 - \cos \frac{n\pi}{n+1}}{1 - \cos \frac{\pi}{n+1}} = \frac{\sin^2 \frac{n\pi}{2(n+1)}}{\sin^2 \frac{\pi}{2(n+1)}} \approx \frac{4(n+1)^2}{\pi^2}.$$

故当 n 越来越大时, $\kappa(T_N)$ 也越来越大, 即 T_N 越来越病态.

求解二维离散 Poisson 方程的 Jacobi 迭代法

算法 6.5 求解二维离散 Poisson 方程的 Jacobi 迭代法

```

1: Given an initial guess  $u^{(0)} \in \mathbb{R}^N$ 
2: while not converge do
3:   for  $i = 1$  to  $n$  do
4:     for  $j = 1$  to  $n$  do
5:        $u_{i,j}^{(k+1)} = \frac{1}{4} \left( h^2 f_{i,j} + u_{i+1,j}^{(k)} + u_{i-1,j}^{(k)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k)} \right)$ 
6:     end for
7:   end for
8: end while

```



求解二维离散 Poisson 方程的 G-S 迭代法

我们可以直接用 Gauss-Seidel 迭代法对基于自然顺序的离散 Poisson 方程 (6.13) 进行数值求解, 即

```

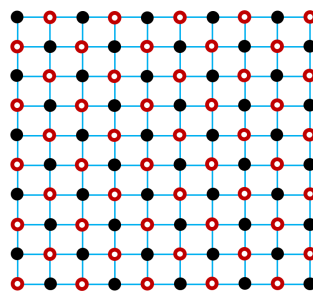
1: Given an initial guess  $u^{(0)} \in \mathbb{R}^N$ 
2: while not converge do
3:   for  $i = 1$  to  $n$  do
4:     for  $j = 1$  to  $n$  do
5:        $u_{i,j}^{(k+1)} = \frac{1}{4} \left( h^2 f_{i,j} + u_{i+1,j}^{(k)} + u_{i-1,j}^{(k+1)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k+1)} \right)$ 
6:     end for
7:   end for
8: end while

```

但此时迭代解的更新必须按自然顺序进行 (即更新 $u_{i,j}^{(k+1)}$ 时必须先更新 $u_{i-1,j}^{(k+1)}$ 和 $u_{i,j-1}^{(k+1)}$ 的值), 因此不适合并行计算.

下面我们介绍一种适合并行计算的排序方法: **红黑排序**, 即将二维网格点依次做红黑记号, 如右图所示.

在计算过程中, 对 $u_{i,j}^{(k+1)}$ 的值进行更新时, 我们可以先更新红色节点上的值, 此时所使用的只是黑色节点的数据, 然后再更新黑色节点上的值, 这时使用的是红色节点的数据. 由于在更新红点时, 各个红点之间是相互独立的, 因此可以并行计算. 同样, 在更新黑点时, 各个黑点之间也是相互独立的, 因此也可以并行计算.



算法 6.6 求解二维离散 Poisson 方程的红黑排序 G-S 迭代法

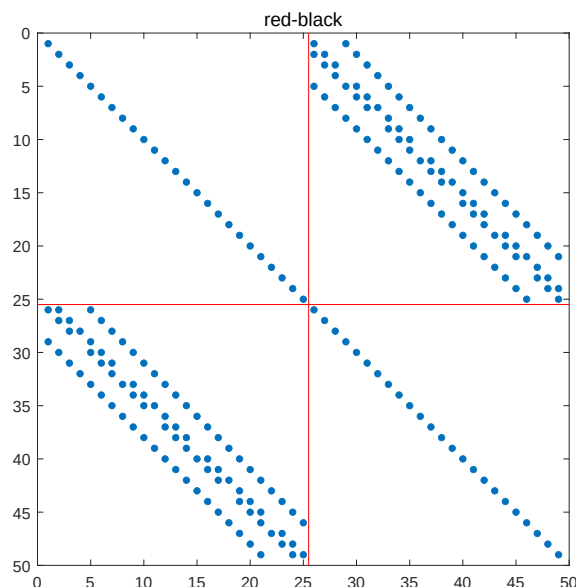
```

1: Given an initial guess  $u^{(0)} \in \mathbb{R}^N$ 
2: while not converge do
3:   for  $(i, j)$  为红色节点 do
4:      $u_{i,j}^{(k+1)} = \frac{1}{4} \left( h^2 f_{i,j} + u_{i+1,j}^{(k)} + u_{i-1,j}^{(k)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k)} \right)$  % 使用旧数据
5:   end for
6:   for  $(i, j)$  为黑色节点 do
7:      $u_{i,j}^{(k+1)} = \frac{1}{4} \left( h^2 f_{i,j} + u_{i+1,j}^{(k+1)} + u_{i-1,j}^{(k+1)} + u_{i,j+1}^{(k+1)} + u_{i,j-1}^{(k+1)} \right)$  % 使用新数据
8:   end for
9: end while

```

我们记基于红黑排序所得到的系数矩阵为 T_N^{RB} , 它可以写成 2×2 分块形式, 且两个对角块都是对角矩阵, 如下图所示 ($N = 7^2$)





事实上, T_N^{RB} 可以通过对 T_N 进行行置换和列置换得到, 即存在置换矩阵 P 使得

$$T_N^{\text{RB}} = PT_N P^T.$$

求解二维离散 Poisson 方程的 SOR 迭代法

算法 6.7 求解二维离散 Poisson 方程的 SOR 迭代法

- 1: Given an initial guess $u^{(0)} \in \mathbb{R}^N$ and a parameter ω
- 2: **while** not converge **do**
- 3: **for** $i = 1$ to n **do**
- 4: **for** $j = 1$ to n **do**
- 5: $u_{i,j}^{(k+1)} = (1 - \omega)u_{i,j}^{(k)} + \frac{\omega}{4} \left(h^2 f_{i,j} + u_{i+1,j}^{(k)} + u_{i-1,j}^{(k+1)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k+1)} \right)$
- 6: **end for**
- 7: **end for**
- 8: **end while**

上面的 SOR 算法是基于自然排序, 基于红黑排序的 SOR 算法请读者自己完成.

例 6.2 已知二维 Poisson 方程

$$\begin{cases} -\Delta u(x, y) = -1, & (x, y) \in \Omega \\ u(x, y) = \frac{x^2 + y^2}{4}, & (x, y) \in \partial\Omega \end{cases}$$

其中 $\Omega = (0, 1) \times (0, 1)$. 该方程的解析解是 $u(x, y) = \frac{x^2 + y^2}{4}$. 用五点差分格式离散后得到一个线性方程组.



- (1) 分别用 Jacobi, G-S 和 SOR 迭代法计算该方程组的解.
- (2) 分别用 SOR 和 SSOR 迭代法求解该方程组, 观察参数 ω 对算法收敛的影响.

 在编写程序时, 我们不需要生成和存放系数矩阵. 此外, 我们用矩阵形式来存放未知量 u_{ij} , 是为了与算法描述相一致, 便于计算矩阵向量乘积,

6.3.3 收敛性分析

对于基本迭代法, 其收敛的充要条件是迭代矩阵的谱半径小于 1. 当谱半径不可求时, 我们可以根据迭代矩阵的范数来判断, 即如果迭代矩阵的某个范数小于 1, 则迭代法收敛.

本小节考虑求解二维离散 Poisson 方程的 Jacobi, G-S 和 SOR 的收敛性. 考虑这些方法的收敛性, 只需研究相应的迭代矩阵的谱半径即可. 对于二维离散 Poisson 方程, 系数矩阵为

$$A = T = I \otimes T_n + T_n \otimes I.$$

故 Jacobi 迭代法的迭代矩阵为

$$G_J = D^{-1}(L + U) = (4I)^{-1}(4I - T) = I - T/4. \quad (6.14)$$

由于 T 的特征值为

$$\lambda_i + \lambda_j = 2 \left(1 - \cos \frac{\pi i}{n+1} \right) + 2 \left(1 - \cos \frac{\pi j}{n+1} \right) = 4 - 2 \left(\cos \frac{\pi i}{n+1} + \cos \frac{\pi j}{n+1} \right),$$

所以 G_J 的特征值为

$$1 - (\lambda_i + \lambda_j)/4 = \frac{1}{2} \left(\cos \frac{\pi i}{n+1} + \cos \frac{\pi j}{n+1} \right).$$

故

$$\rho(G_J) = \frac{1}{2} \max_{i,j} \left\{ \left| \cos \frac{\pi i}{n+1} + \cos \frac{\pi j}{n+1} \right| \right\} = \cos \frac{\pi}{n+1} < 1,$$

即 Jacobi 迭代法是收敛的.

注意当 n 越来越大时, $\kappa(T) \rightarrow \infty$, 即 T 越来越病态, 此时 $\rho(G_J) \rightarrow 1$, 即 Jacobi 迭代法收敛越来越慢.

 通常, 问题越病态就越难求解, 舍入误差对解的影响也越大.

关于 G-S 和 SOR, 我们有下面的结论.

定理 6.27 设 G_{GS} 和 G_{SOR} 分别表示求解基于红黑排序的二维 Poisson 方程的 G-S 和 SOR 的迭代矩阵, 则有

$$\rho(G_{GS}) = \rho(G_J)^2 = \cos^2 \frac{\pi}{n+1} < 1$$

$$\rho(G_{SOR}) = \frac{\cos^2 \frac{\pi}{n+1}}{\left(1 + \sin \frac{\pi}{n+1} \right)^2} < 1, \quad \omega = \frac{2}{1 + \sin \frac{\pi}{n+1}}.$$



在上述结论中, SOR 中的 ω 是最优参数, 即此时的 $\rho(G_{\text{SOR}})$ 最小. 由 Taylor 公式可知, 当 n 很大时, 有

$$\rho(G_J) = \cos \frac{\pi}{n+1} \approx 1 - \frac{\pi^2}{2(n+1)^2} = 1 - O\left(\frac{1}{n^2}\right),$$

$$\rho(G_{\text{SOR}}) = \frac{\cos^2 \frac{\pi}{n+1}}{\left(1 + \sin \frac{\pi}{n+1}\right)^2} \approx 1 - \frac{2\pi}{n+1} = 1 - O\left(\frac{1}{n}\right).$$

由于当 n 很大时有

$$\left(1 - \frac{1}{n}\right)^k \approx 1 - \frac{k}{n} = 1 - \frac{kn}{n^2} \approx \left(1 - \frac{1}{n^2}\right)^{kn},$$

即 SOR 迭代 k 步后误差的减小量与 Jacobi 迭代 kn 步后误差减小量差不多. 因此, 对于二维离散 Poisson 方程, 当 SOR 取最优参数时, 收敛速度大约是 Jacobi 的 n 倍.

 需要指出的是, 对于一般线性方程组, 上述结论不一定成立.

由于 $\rho(G_{\text{GS}}) = \rho(G_J)^2$, 因此, 对于二维离散 Poisson 方程, G-S 的收敛速度大约是 Jacobi 的 2 倍.

 事实上, 当 n 很大时, 这三个方法的收敛速度都很慢.

例 6.3 同例 6.2, 分别对不同的 n , 观察 Jacobi, Gauss-Seidel 和 SOR 迭代法的收敛情况.

 事实上, 由于二维离散 Poisson 方程的系数矩阵 T_N 是不可约对角占优的, 因此根据定理 6.8, Jacobi 和 G-S 都收敛. 另外, T_N 是对称正定的, 故当 $0 < \omega < 2$ 时, SOR 迭代法收敛.

6.3.4 快速求解方法

如果已经知道矩阵 A 的特征值分解 $A = X\Lambda X^{-1}$, 则 $Ax = b$ 的解可表示为

$$x = A^{-1}b = X\Lambda^{-1}X^{-1}b.$$

如果 A 是正规矩阵, 即 X 是酉矩阵, 则

$$x = A^{-1}b = X\Lambda^{-1}X^*b.$$

 一般来说, 我们不会采用这种特征值分解的方法来解线性方程组, 因为计算特征值分解通常比解线性方程组更困难. 但在某些特殊情况下, 我们可以由此得到快速方法.

考虑二维离散 Poisson 方程

$$Tu = h^2 f, \quad (6.15)$$



其中

$$T = I \otimes T_n + T_n \otimes I, \quad T_n = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{n \times n}.$$

由定理 6.26 可知

$$T = (Z \otimes Z)(I \otimes \Lambda + \Lambda \otimes I)(Z \otimes Z)^T,$$

其中 $Z = [z_1, z_2, \dots, z_n]$ 是正交矩阵. 这里

$$z_k = \sqrt{\frac{2}{n+1}} \left[\sin \frac{k\pi}{n+1}, \sin \frac{2k\pi}{n+1}, \dots, \sin \frac{nk\pi}{n+1} \right]^T, \quad k = 1, 2, \dots, n.$$

所以, 方程 (6.15) 的解为

$$u = T^{-1}h^2f = [(Z \otimes Z)(I \otimes \Lambda + \Lambda \otimes I)^{-1}(Z \otimes Z)^T]h^2f.$$

因此, 主要的运算是 $Z \otimes Z$ 与向量的乘积, 以及 $(Z \otimes Z)^T$ 与向量的乘积. 而这些乘积可以通过快速 Sine 变换来实现.

快速 Fourier 变换

快速 Fourier 变换 (FFT) 是用来计算离散 Fourier 变换 (**DFT**) 矩阵与向量乘积的一种快速方法.

设 $x = [x_0, x_1, \dots, x_{n-1}]^T \in \mathbb{C}^n$, 其 DFT 定义为 $y = \text{DFT}(x) = [y_0, y_1, \dots, y_{n-1}]^T \in \mathbb{C}^n$, 其中

$$y_k = \sum_{j=0}^{n-1} \omega_n^{kj} x_j, \quad k = 0, 1, \dots, n-1.$$

这里

$$\omega_n = e^{-\frac{2\pi i}{n}} = \cos(2\pi/n) - i \sin(2\pi/n)$$

是 1 的一个 n 次本原根 (primitive n -th root of unity), i 是虚部单位.

 这里说 ω_n 是 primitive n -th root of unity 是指 $\omega_n^n = 1$ 且

$$\omega_n^k \neq 1, \quad k = 1, 2, \dots, n-1.$$

构造矩阵 $F_n = [f_{kj}] \in \mathbb{C}^{n \times n}$, 其中 $f_{kj} = \omega_n^{kj} = e^{-\frac{2kj\pi i}{n}} = \cos(2kj\pi/n) - i \sin(2kj\pi/n)$, 即

$$F_n = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)^2} \end{bmatrix},$$

则有

$$y = \text{DFT}(x) = F_n x.$$

我们称矩阵 F_n 为 DFT 矩阵. 易知 DFT 矩阵具有以下性质:

- (1) F_n 是对称矩阵 (但不是 Hermitian);



(2) $F_n^* F_n = nI$, 所以 $\frac{1}{\sqrt{n}} F_n$ 是酉矩阵.

相应的离散 Fourier 反变换定义为 $x = \text{IDFT}(y)$, 其中

$$x_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega_n^{-jk} y_k, \quad j = 0, 1, \dots, n-1.$$

写成矩阵形式为

$$x = \frac{1}{n} F^* y.$$

DFT 和 IDFT 满足下面的性质:

$$\text{IDFT}(\text{DFT}(x)) = x,$$

$$\text{DFT}(\text{IDFT}(y)) = y.$$

 在 MATLAB 中, 计算 DFT 和 IDFT 的函数分别为 `fft` 和 `ifft`, 即: `y=fft(x)`, `x=ifft(y)`.
(测试代码见 `FFT_test.m`)

离散 Sine 变换

离散 Sine 变换 (DST) 有多种定义, 我们这里只介绍与求解 Poisson 方程有关的一种定义, 其它定义可参见 [14].

设 $x = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$, 其离散 Sine 变换定义为 $y = \text{DST}(x) = [y_1, y_2, \dots, y_n]^T \in \mathbb{R}^n$, 其中

$$y_k = \sum_{j=1}^n x_j \sin\left(\frac{kj\pi}{n+1}\right), \quad k = 1, 2, \dots, n.$$

对应的离散 Sine 反变换记为 IDST, 即 $x = \text{IDST}(y)$, 其中

$$x_j = \frac{2}{n+1} \sum_{k=1}^n y_k \sin\left(\frac{jk\pi}{n+1}\right), \quad j = 1, 2, \dots, n.$$

DST 和 IDST 满足下面的性质:

$$\text{IDST}(\text{DST}(x)) = x,$$

$$\text{DST}(\text{IDST}(y)) = y.$$

 在 MATLAB 中, 计算 DST 和 IDST 的函数分别为 `dst` 和 `idst`, 即: `y=dst(x)`, `x=idst(y)`.
(测试代码见 `DST_test.m`)

Possion 方程与 DST

我们首先考虑矩阵 Z 与一个任意给定向量 b 的乘积. 设 $y = Zb$, 则

$$y_k = \sum_{j=1}^n Z(k, j) b_j = \sqrt{\frac{2}{n+1}} \sum_{j=1}^n b_j \sin\left(\frac{kj\pi}{n+1}\right) = \sqrt{\frac{2}{n+1}} [\text{DST}(b)]_k,$$



其中 $[\text{DST}(b)]_k$ 表示第 k 个分量. 因此, 乘积 $y = Zb$ 可以通过 DST 来实现. 类似地, 乘积 $y = Z^T b = Z^{-1}b$ 可以通过离散 Sine 反变换 IDST 实现, 即

$$y = Z^T b = Z^{-1}b = \left(\sqrt{\frac{2}{n+1}} \right)^{-1} \text{IDST}(b).$$

所以对于一维离散 Poisson 方程, 其解为

$$u = T_n^{-1}(h^2 f) = (Z\Lambda^{-1}Z^T)(h^2 f) = h^2 Z\Lambda^{-1}Z^T f = h^2 \text{DST}(\Lambda^{-1} \text{IDST}(b)).$$

可以通过公式 $T_n = Z\Lambda Z^T$ 来计算一维离散 Poisson 矩阵的特征值: 记 $[\alpha_1, \alpha_2, \dots, \alpha_n]^T$ 和 $[\beta_1, \beta_2, \dots, \beta_n]^T$ 分别为 $Z^T T_n$ 和 Z^T 的第一列, 则 T_n 的特征值为

$$\lambda_i = \frac{\alpha_i}{\beta_i}.$$

对应的 MATLAB 代码为: `Lam=idst([2,-1,zeros(1,n-2)]') ./ idst(eye(n,1))`

而对于二维离散 Poisson 方程, 我们需要计算 $(Z \otimes Z)b$ 和 $(Z^T \otimes Z^T)b$. 它们对应的是二维离散 Sine 变换和二维离散 Sine 反变换.

设 $b = [b_1^T, b_2^T, \dots, b_n^T]^T \in \mathbb{R}^{n^2}$, 其中 $b_k \in \mathbb{R}^{n \times n}$. 令 $B = [b_1, b_2, \dots, b_n] \in \mathbb{R}^{n \times n}$, 则由 Kronecker 乘积的性质可知

$$(Z \otimes Z)b = (Z \otimes Z)\text{vec}(B) = \text{vec}(ZBZ^T) = \text{vec}\left((Z(ZB)^T)^T\right).$$

因此, 我们仍然可以使用 DST 来计算 $(Z \otimes Z)b$. 类似地, 我们可以使用 IDST 来计算 $(Z^T \otimes Z^T)b$.

算法 6.8 二维离散 Poisson 方程的快速方法

- 1: 计算 $b = h^2 f$
- 2: $B = \text{reshape}(b, n, n)$
- 3: $B_1 = (Z^T B)^T = (\text{IDST}(B))^T$
- 4: $B_2 = (Z^T B_1)^T = (\text{IDST}(B_1))^T$
- 5: $b_1 = (I \otimes \Lambda + \Lambda \otimes I)^{-1} \text{vec}(B_2)$
- 6: $B_3 = \text{reshape}(b_1, n, n)$
- 7: $B_4 = (ZB_3)^T = (\text{DST}(B_3))^T$
- 8: $B_5 = (ZB_4)^T = (\text{DST}(B_4))^T$
- 9: $u = \text{reshape}(B_5, n^2, 1)$

6.4 加速方法

当迭代解 $x^{(0)}, x^{(1)}, x^{(2)}, \dots, x^{(k)}$ 已经计算出来后, 我们可以对其进行组合, 得到一个新的近似解, 这样就可以对原方法进行加速.

6.4.1 外推技术

设原迭代格式为

$$x^{(k+1)} = Gx^{(k)} + g. \quad (6.16)$$



由 $x^{(k)}$ 和 $x^{(k+1)}$ 加权组合后可得新的近似解

$$x^{(k+1)} = (1 - \omega)x^{(k)} + \omega(Gx^{(k)} + g), \quad (6.17)$$

其中 ω 是参数. 这种加速方法就称为 **外推方法**.

 如果原迭代格式是 Gauss-Seidel 迭代, 虽然 SOR 与外推方法 (6.17) 都是通过加权进行加速, 但格式是不一样的: 前者是在单个分量计算出来后就进行加权, 而后者则是当所有分量都计算出来后才加权.

为了使得迭代格式 (6.17) 尽可能快地收敛, 需要选择 ω 使得其迭代矩阵 $G_\omega \triangleq (1 - \omega)I + \omega G$ 的谱半径尽可能地小. 假设 G 的特征值都是实数, 且最大特征值和最小特征值分别为 λ_1 和 λ_n . 于是

$$\rho(G_\omega) = \max_{\lambda \in \sigma(G)} |(1 - \omega) + \omega\lambda| = \max\{|1 - \omega + \omega\lambda_1|, |1 - \omega + \omega\lambda_n|\}.$$

定理 6.28 设 G 的特征值都是实数, 其最大和最小特征值分别为 λ_1 和 λ_n , 且 $1 \notin [\lambda_n, \lambda_1]$, 则

$$\omega_* = \arg \min_{\omega} \rho(G_\omega) = \frac{2}{2 - (\lambda_1 + \lambda_n)},$$

此时

$$\rho(G_{\omega_*}) = 1 - |\omega_*|d,$$

其中 d 是 1 到 $[\lambda_n, \lambda_1]$ 的距离, 即当 $\lambda_n \leq \lambda_1 < 1$ 时, $d = 1 - \lambda_1$, 当 $\lambda_1 \geq \lambda_n > 1$ 时, $d = \lambda_n - 1$.

(板书)

由定理 6.28 可知, $\rho(G_{\omega_*}) = 1 - |\omega_*|d$, 且当 $\omega_* \neq 1$ 时, 外推迭代 (6.17) 比原迭代法收敛要更快一些.

 最优参数依赖于原迭代矩阵 G 的特征值, 因此实用性不强. 在实际应用时可以估计特征值所在的区间 $[\alpha, \beta]$, 然后用 α, β 来代替 λ_n 和 λ_1 .

JOR 方法

对 Jacobi 迭代进行外推加速, 则可得 JOR (Jacobi over-relaxation) 方法:

$$\begin{aligned} x^{(k+1)} &= (1 - \omega)x^{(k)} + \omega(D^{-1}(L + U)x^{(k)} + D^{-1}b) \\ &= x^{(k)} + \omega D^{-1}(b - Ax^{(k)}), \quad k = 0, 1, 2, \dots \end{aligned}$$

定理 6.29 设 A 对称正定. 若

$$0 < \omega < \frac{2}{\rho(D^{-1}A)},$$

则 JOR 方法收敛.



6.4.2 Chebyshev 多项式加速

本节对外推技巧进行推广.

假定通过迭代格式 (6.16) 已经计算出 $x^{(0)}, x^{(1)}, \dots, x^{(k)}$, 下面考虑如何将这些近似解进行组合, 以便得到更精确的近似解.

记 $\varepsilon_k = x^{(k)} - x_*$ 为第 k 步迭代解的误差, 则有

$$\varepsilon_k = G\varepsilon_{k-1} = G^2\varepsilon_{k-2} = \cdots = G^k\varepsilon_0.$$

设 $\tilde{x}^{(k)}$ 为 $x^{(0)}, x^{(1)}, \dots, x^{(k)}$ 的一个线性组合, 即

$$\tilde{x}^{(k)} = \alpha_0 x^{(0)} + \alpha_1 x^{(1)} + \cdots + \alpha_k x^{(k)}, \quad (6.18)$$

其中 α_i 为待定系数, 且满足 $\sum_{i=0}^k \alpha_i = 1$. 于是

$$\tilde{x}^{(k)} - x_* = \alpha_0 \varepsilon_0 + \alpha_1 G\varepsilon_0 + \cdots + \alpha_k G^k \varepsilon_0 \triangleq p_k(G)\varepsilon_0, \quad (6.19)$$

其中 $p_k(t) = \sum_{i=0}^k \alpha_i t^i$ 为 k 次多项式, 且满足 $p_k(1) = 1$.

我们希望通过适当选取参数 α_i , 使得 $\tilde{x}^{(k)} - x_*$ 尽可能地小, 即使得 $\tilde{x}^{(k)}$ 收敛到 x_* 速度远远快于 $x^{(k)}$ 收敛到 x_* 速度. 这种加速方法就称为**多项式加速**或**半迭代法** (semi-iterative method).

例 6.4 设 $p_n(t)$ 为 G 的特征多项式, 令 $\tilde{p}_n(t) \triangleq p_n(t)/p_n(1)$, 则 $\tilde{p}_n(1) = 1$ 且 $\tilde{p}_n(G) = 0$, 所以选取 α_i 为 $\tilde{p}_n$ 的系数, 则 $\tilde{x}^{(n)} - x_* = 0$. 但这种选取方法不实用, 原因是:

- (1) $\tilde{p}_n(t)$ 的系数并不知道;
- (2) 我们通常希望收敛所需的迭代步数 $\ll n$.

下面讨论参数 α_i 的较实用的选取方法. 由 (6.19) 可知

$$\|\tilde{x}^{(k)} - x_*\|_2 = \|p_k(G)\varepsilon_0\|_2 \leq \|p_k(G)\|_2 \cdot \|\varepsilon_0\|_2.$$

因此我们需要求解下面的极小化问题

$$\min_{p \in \mathbb{P}_k, p(1)=1} \|p(G)\|_2, \quad (6.20)$$

其中 $\mathbb{P}_k$ 表示所有次数不超过 k 的多项式组成的集合. 一般来说, 这个问题是非常困难的. 但在一些特殊情况下, 我们可以给出其 (近似) 最优解.

假设迭代矩阵 G 是对称矩阵, 即 G 存在特征值分解

$$G = Q\Lambda Q^T,$$

其中 Λ 是对角矩阵, 且对角线元素都是实的, Q 是正交矩阵. 于是有

$$\begin{aligned} \min_{p \in \mathbb{P}_k, p(1)=1} \|p(G)\|_2 &= \min_{p \in \mathbb{P}_k, p(1)=1} \|p(\Lambda)\|_2 \\ &= \min_{p \in \mathbb{P}_k, p(1)=1} \max_{1 \leq i \leq n} \{|p(\lambda_i)|\} \\ &\leq \min_{p \in \mathbb{P}_k, p(1)=1} \max_{\lambda \in [\lambda_n, \lambda_1]} \{|p(\lambda)|\}, \end{aligned} \quad (6.21)$$



其中 λ_1, λ_n 分别表示 G 的最大和最小特征值. 这是带归一化条件的多项式最佳一致逼近问题 (与零的偏差最小). 该问题的解与著名的 Chebyshev 多项式有关.

 由于所有算子范数 $\|p_k(G)\|$ 的下确界是 $\rho(p_k(G))$, 因此, 一种较实用的选取方法是使得 $p_k(G)$ 的谱半径尽可能地小.

Chebyshev 多项式

Chebyshev 多项式是一类很重要的正交多项式, 在函数逼近, 函数插值, 数值积分等方面都有着重要的应用.

Chebyshev 多项式 $T_k(t)$ 可以通过下面的递归方式来定义:

$$\begin{aligned} T_0(t) &= 1, \quad T_1(t) = t, \\ T_k(t) &= 2tT_{k-1}(t) - T_{k-2}(t), \quad k = 2, 3, \dots, \end{aligned} \quad (6.22)$$

也可以直接由下面的式子定义

$$T_k(t) = \begin{cases} \cos(k \arccos t), & |t| \leq 1 \\ \cosh(k \operatorname{arccosh} t), & |t| > 1 \end{cases},$$

其中 $\cosh$ 为双曲余弦, 即 $\cosh(t) = \frac{e^t + e^{-t}}{2}$.

引理 6.30 Chebyshev 多项式具有下面的性质:

- (1) $T_k(1) = 1$;
- (2) $T_k(t)$ 的首项系数为 2^{k-1} ;
- (3) $T_k(-t) = (-1)^k T_k(t)$, 即 $T_{2k}(t)$ 只含偶次项, $T_{2k+1}(t)$ 只含奇次项;
- (4) 当 $|t| \leq 1$ 时 $|T_k(t)| \leq 1$; 当 $|t| > 1$ 时 $|T_k(t)| > 1$;
- (5) $T_k(t) = 0$ 的解为 $t_i = \cos \frac{(2i-1)\pi}{2k}$, $i = 1, \dots, k$;
- (6) $T_k(t)$ 有 $n-1$ 个极值点: $t_i = \cos \frac{k\pi}{n}$, $i = 1, 2, \dots, k-1$;

下面的结论表明, 在所有首项系数为 1 的 k 次多项式中, $p_k(t) = \frac{1}{2^{k-1}} T_k(t)$ 在 $[-1, 1]$ 上与零的偏差是最小的 (在无穷范数意义下).

定理 6.31 设 $p_k(t) = \frac{1}{2^{k-1}} T_k(t)$, 则

$$\max_{-1 \leq t \leq 1} |p_k(t)| \leq \max_{-1 \leq t \leq 1} |p(t)|, \quad \forall p(t) \in \tilde{\mathbb{P}}_k,$$



其中 $\tilde{\mathbb{P}}_k$ 表示所有首项系数为 1 的 k 次多项式组成的集合, 即

$$\|p_k(t)\|_\infty = \min_{p(t) \in \tilde{\mathbb{P}}_k} \|p(t)\|_\infty.$$

这里的范数 $\|\cdot\|_\infty$ 是 $C[-1, 1]$ 上的无穷范数, 即 $\|f\|_\infty = \max_{x \in [-1, 1]} |f(x)|$, $f(x) \in C[-1, 1]$.

该性质可用于计算首项系数非零的 n 次多项式在 $[-1, 1]$ 上的 $n-1$ 次最佳一致逼近多项式. 利用这个性质, 我们可以采用 Chebyshev 多项式的零点作为节点进行多项式插值, 以使得插值的总体误差达到最小化.

Chebyshev 另外一个重要性质是下面的最小最大性质.

定理 6.32 设 $\eta \in \mathbb{R}$ 满足 $|\eta| > 1$, 则下面的最小最大问题

$$\min_{p(t) \in \mathbb{P}_k, p(\eta)=1} \max_{-1 \leq t \leq 1} |p(t)|$$

的唯一解为

$$\tilde{T}_k(t) \triangleq \frac{T_k(t)}{T_k(\eta)}.$$

通过简单的仿射变换, 该定理的结论可以推广到一般区间.

定理 6.33 设 $\alpha, \beta, \eta \in \mathbb{R}$ 满足 $\alpha < \beta$ 且 $|\eta| \notin [\alpha, \beta]$. 则下面的最小最大问题

$$\min_{p(t) \in \mathbb{P}_k, p(\eta)=1} \max_{\alpha \leq x \leq \beta} |p(t)|$$

的唯一解为

$$\hat{T}_k(t) \triangleq \frac{T_k\left(\frac{2t - (\beta + \alpha)}{\beta - \alpha}\right)}{T_k\left(\frac{2\eta - (\beta + \alpha)}{\beta - \alpha}\right)},$$

其中 $\mathbb{P}_k$ 表示所有次数不超过 k 的实系数多项式组成的集合.

Chebyshev 加速方法

考虑迭代格式 (6.16), 即 $x^{(k+1)} = Gx^{(k)} + g$, 我们以一种简单情形为例, 演示 Chebyshev 加速方法的推导过程, 即假定:

- (1) 迭代矩阵 G 对称;
- (2) G 的特征值 $\lambda(G) \in [-\rho, \rho]$, 其中 $\rho = \rho(G) < 1$.

于是最小最大问题 (6.21) 就转化为

$$\min_{p \in \mathbb{P}_k, p(1)=1} \max_{\lambda \in [-\rho, \rho]} \{|p(\lambda)|\}.$$

由于 $1 \notin [-\rho, \rho]$, 根据定理 6.33, 上述问题的解为

$$p_k(t) = \frac{T_k(t/\rho)}{T_k(1/\rho)}.$$



下面考虑 $\tilde{x}^{(k)}$ 的计算. 我们无需先计算出 $x^{(0)}, x^{(1)}, \dots, x^{(k)}$, 然后再通过线性组合 (6.18) 来计算 $\tilde{x}^{(k)}$. 事实上, 我们可以通过 Chebyshev 多项式的三项递推公式 (6.22), 由 $\tilde{x}^{(k-1)}$ 和 $\tilde{x}^{(k-2)}$ 直接计算出 $\tilde{x}^{(k)}$. 这样一方面可以得到一个收敛速度更快的迭代格式, 另一方面可以节约存储, 因为无需存储所有的 $\tilde{x}^{(i)}$. 下面给出具体的推导过程.

令 $\mu_k = \frac{1}{T_k(1/\rho)}$, 即 $T_k(1/\rho) = \frac{1}{\mu_k}$, 由三项递推公式 (6.22) 可得

$$\frac{1}{\mu_k} = \frac{2}{\rho} \frac{1}{\mu_{k-1}} - \frac{1}{\mu_{k-2}}.$$

所以

$$\begin{aligned} \tilde{x}^{(k)} - x_* &= p_k(G) \varepsilon_0 = \mu_k T_k(G/\rho) \varepsilon_0 \\ &= \mu_k \left[\frac{2G}{\rho} T_{k-1}(G/\rho) - T_{k-2}(G/\rho) \right] \varepsilon_0 \\ &= \mu_k \left[\frac{2G}{\rho} \frac{1}{\mu_{k-1}} p_{k-1}(G) \varepsilon_0 - \frac{1}{\mu_{k-2}} p_{k-2}(G) \varepsilon_0 \right] \\ &= \mu_k \left[\frac{2G}{\rho} \frac{1}{\mu_{k-1}} (\tilde{x}^{(k-1)} - x_*) - \frac{1}{\mu_{k-2}} (\tilde{x}^{(k-2)} - x_*) \right]. \end{aligned}$$

整理后可得

$$\tilde{x}^{(k)} = \frac{2\mu_k}{\mu_{k-1}} \frac{G}{\rho} \tilde{x}^{(k-1)} - \frac{\mu_k}{\mu_{k-2}} \tilde{x}^{(k-2)} + d_k,$$

其中

$$\begin{aligned} d_k &= x_* - \frac{2\mu_k}{\mu_{k-1}} \frac{G}{\rho} x_* + \frac{\mu_k}{\mu_{k-2}} x_* \\ &= x_* - \frac{2\mu_k}{\mu_{k-1}} \frac{x_* - g}{\rho} + \frac{\mu_k}{\mu_{k-2}} x_* \\ &= \mu_k \left(\frac{1}{\mu_k} - \frac{2}{\rho \mu_{k-1}} + \frac{1}{\mu_{k-2}} \right) x_* + \frac{2\mu_k g}{\mu_{k-1} \rho} = \frac{2\mu_k g}{\mu_{k-1} \rho}. \end{aligned}$$

由此, 我们可以得到迭代格式 (6.16) 的 Chebyshev 加速算法.

算法 6.9 Chebyshev 加速算法

- 1: Set $\mu_0 = 1, \mu_1 = \rho = \rho(G), \tilde{x}^{(0)} = x^{(0)}, k = 1$
 - 2: compute $\tilde{x}^{(1)} = Gx^{(0)} + g$
 - 3: **while** not converge **do**
 - 4: $k = k + 1$
 - 5: $\mu_k = \left(\frac{2}{\rho} \frac{1}{\mu_{k-1}} - \frac{1}{\mu_{k-2}} \right)^{-1}$
 - 6: $\tilde{x}^{(k)} = \frac{2\mu_k}{\mu_{k-1}} \frac{G}{\rho} \tilde{x}^{(k-1)} - \frac{\mu_k}{\mu_{k-2}} \tilde{x}^{(k-2)} + \frac{2\mu_k}{\mu_{k-1} \rho} g$
 - 7: **end while**
-

 该方法的每步迭代中只有一次矩阵向量乘积, 故方法每个迭代步的整体运算量与原迭代格



式的每个迭代步的运算量基本相当.

 设 $\lambda(G) \in [\alpha, \beta]$, 且 $-1 < \alpha \leq \beta < 1$, 则我们也可以构造出相应的 Chebyshev 加速方法.

注记

在前面的分析中, 我们要求 G 是对称的. 事实上, 由于 $\tilde{x}^{(k)} - x_* = p_k(G)\varepsilon_0$, 因此我们只要极小化 $\rho(p_k(G))$ 即可. 所以我们只需假定 G 的特征值是实数, 这样 $p_k(G)$ 的特征值就是 $p_k(\lambda_i)$, 其中 λ_i 为 G 的特征值. 此时我们需要求解的极小化问题为

$$\min_{p \in \mathbb{P}_k, p(1)=1} |\rho(p(G))| = \min_{p \in \mathbb{P}_k, p(1)=1} \max_{1 \leq i \leq n} \{|p(\lambda_i)|\} \leq \min_{p \in \mathbb{P}_k, p(1)=1} \max_{\lambda \in [\lambda_n, \lambda_1]} \{|p(\lambda)|\}.$$

该问题的解可以通过 Chebyshev 多项式来表示.

SSOR 方法的 Chebyshev 加速

假定 A 对称, 且对角线元素都为正数, 则 SSOR 迭代矩阵为

$$G_{\text{SSOR}} = (D - \omega U)^{-1}[(1 - \omega)D + \omega L](D - \omega L)^{-1}[(1 - \omega)D + \omega U].$$

由于 A 对称, 即 $L = U^T$, 故

$$\begin{aligned} & (D - \omega U)G_{\text{SSOR}}(D - \omega U)^{-1} \\ &= [(1 - \omega)D + \omega L](D - \omega L)^{-1}[(1 - \omega)D + \omega L^T](D - \omega L^T)^{-1} \\ &= [(2 - \omega)D(D - \omega L)^{-1} - I][(2 - \omega)D(D - \omega L^T)^{-1} - I] \\ &= I - (2 - \omega)D[(D - \omega L)^{-1} + (D - \omega L^T)^{-1}] + (2 - \omega)^2 D(D - \omega L)^{-1}D(I - \omega L^T)^{-1}. \end{aligned}$$

又 D 的对角线元素全是正的, 则

$$\begin{aligned} & D^{-1/2}(D - \omega U)G_{\text{SSOR}}(D - \omega U)^{-1}D^{1/2} \\ &= I - (2 - \omega)D^{-1/2}[(D - \omega L)^{-1} + (D - \omega L^T)^{-1}]D^{1/2} \\ &\quad + (2 - \omega)^2 D^{-1/2}(D - \omega L)^{-1}D(I - \omega L^T)^{-1}D^{1/2}. \end{aligned}$$

这是一个对称矩阵, 故 G_{SSOR} 具有实特征值, 所以我们可以对其实行 Chebyshev 加速. 但我们需要估计 G_{SSOR} 的谱半径.

 若存在矩阵 W 使得 $W^{-1}AW$ 是对称矩阵, 则称 A 是可对称化的, 即 A 相似于一个对称矩阵.

6.5 交替方向法与 HSS 迭代法

6.5.1 多步迭代法

设 $A = M_1 - N_1 = M_2 - N_2$ 是 A 的两个矩阵分裂, 则可以构造迭代格式

$$\begin{cases} M_1 x^{(k+\frac{1}{2})} = N_1 x^{(k)} + b, \\ M_2 x^{(k+1)} = N_2 x^{(k+\frac{1}{2})} + b, \end{cases} \quad k = 0, 1, 2, \dots \quad (6.23)$$



这就是两步迭代法, 对应的分裂称为二重分裂. 易知, 两步迭代格式 (6.23) 的迭代矩阵为

$$G = M_2^{-1}N_2M_1^{-1}N_1.$$

因此, 其收敛的充要条件是 $\rho(M_2^{-1}N_2M_1^{-1}N_1) < 1$.

类似地, 我们可以推广到多步迭代法. 设 l 是一个正整数, 则 A 的 l 重分裂为

$$A = M_1 - N_1 = M_2 - N_2 = \cdots = M_l - N_l,$$

相应的多步迭代法为

$$\begin{cases} M_1x^{(k+\frac{1}{l})} = N_1x^{(k)} + b, \\ M_2x^{(k+\frac{2}{l})} = N_2x^{(k+\frac{1}{l})} + b, \\ \cdots \\ M_lx^{(k+1)} = N_lx^{(k+\frac{l-1}{l})} + b, \end{cases} \quad k = 0, 1, 2, \dots$$

6.5.2 交替方向法

交替方向法 (Alternating Direction Implicit, **ADI**) 是由 Peaceman 和 Rachford [87] 于 1955 年提出, 用于计算偏微分方程的数值解, 因此也称为 PR 方法. 其本质上也可以看成是一个两步迭代法.

设 $A = A_1 + A_2$, 则 ADI 迭代格式为

$$\begin{cases} (\alpha I + A_1)x^{(k+\frac{1}{2})} = (\alpha I - A_2)x^{(k)} + b, \\ (\alpha I + A_2)x^{(k+1)} = (\alpha I - A_1)x^{(k+\frac{1}{2})} + b, \end{cases} \quad k = 0, 1, 2, \dots, \quad (6.24)$$

其中 $\alpha \in \mathbb{R}$ 是迭代参数. 易知 ADI 方法的迭代矩阵为

$$G_{\text{ADI}} = (\alpha I + A_2)^{-1}(\alpha I - A_1)(\alpha I + A_1)^{-1}(\alpha I - A_2).$$

它相似于

$$\tilde{G} \triangleq (\alpha I - A_1)(\alpha I + A_1)^{-1}(\alpha I - A_2)(\alpha I + A_2)^{-1}.$$

所以 ADI 迭代 (6.24) 收敛的充要条件是

$$\rho(\tilde{G}) < 1.$$

若 A 对称正定, 且 A_1 和 A_2 中有一个是对称正定, 另一个是对称半正定, 则有下面的收敛定理.

定理 6.34 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $A = A_1 + A_2$, 其中 A_1 和 A_2 中有一个是对称正定, 另一个是对称半正定, 则对任意正数 $\alpha > 0$, 有 $\rho(\tilde{G}) < 1$, 即 ADI 迭代法 (6.24) 收敛.

(板书)

6.5.3 HSS 方法

HSS 方法 全称为 **H**ermitian and **S**kew-Hermitian **S**plitting method, 是由 Bai, Golub 和 Ng [10] 于 2003 年提出.



设 $A = H + S$, 其中 H 和 S 分别是 A 的对称与反对称 (斜对称, Skew-Hermite) 部分, 即

$$H = \frac{A + A^T}{2}, \quad S = \frac{A - A^T}{2}.$$

该分裂就称为 HS 分裂, 即 HSS.

 任何一个矩阵都具有 HS 分裂. 如果 $A \in \mathbb{C}^{n \times n}$, 则取共轭转置.

类似于 ADI 方法, 我们可得下面的 HSS 方法

$$\begin{cases} (\alpha I + H)x^{(k+\frac{1}{2})} = (\alpha I - S)x^{(k)} + b, \\ (\alpha I + S)x^{(k+1)} = (\alpha I - H)x^{(k+\frac{1}{2})} + b, \end{cases} \quad k = 0, 1, 2, \dots \quad (6.25)$$

易知, HSS 方法的迭代矩阵为

$$G_{\text{HSS}} = (\alpha I + S)^{-1}(\alpha I - H)(\alpha I + H)^{-1}(\alpha I - S).$$

同样, 它相似于

$$\tilde{G} \triangleq (\alpha I - H)(\alpha I + H)^{-1}(\alpha I - S)(\alpha I + S)^{-1}.$$

我们首先考察矩阵 $(\alpha I - S)(\alpha I + S)^{-1}$. 事实上, 它是一个酉矩阵 (见习题 (5.6)). 因此

$$\|(\alpha I - S)(\alpha I + S)^{-1}\|_2 = 1.$$

由于 H 是对称矩阵, 因此

$$\|(\alpha I - H)(\alpha I + H)^{-1}\|_2 = \max_{\lambda \in \sigma(H)} \left| \frac{\alpha - \lambda}{\alpha + \lambda} \right|.$$

定理 6.35 设 $A \in \mathbb{R}^{n \times n}$ 正定, 则对任意正数 $\alpha > 0$, 有 $\rho(\tilde{G}) < 1$, 即 HSS 迭代法 (6.25) 收敛.

(板书)

参数 α 的选取

为了达到最快收敛效果, 我们希望迭代矩阵的谱半径越小越好. 但是在一般情况下, 谱半径很难计算或估计. 因此要极小化谱半径是非常困难的, 或者说不可能的. 此时, 我们能做的往往是极小化它的一个上界.

由前面的分析可知

$$\rho(G_{\text{HSS}}) = \rho(\tilde{G}) \leq \max_{\lambda \in \sigma(H)} \left| \frac{\alpha - \lambda}{\alpha + \lambda} \right| \triangleq \sigma(\alpha).$$

下面考虑 $\sigma(\alpha)$ 的极小值点. 设 H 的最大和最小特征值分别为 $\lambda_{\max}(H)$ 和 $\lambda_{\min}(H)$, 则我们有下面的结论.

定理 6.36 [10] 设 $A \in \mathbb{R}^{n \times n}$ 正定, 则极小极大问题

$$\min_{\alpha > 0} \max_{\lambda_{\min}(H) \leq \lambda \leq \lambda_{\max}(H)} \left| \frac{\alpha - \lambda}{\alpha + \lambda} \right|$$



的解为

$$\alpha_* = \sqrt{\lambda_{\max}(H)\lambda_{\min}(H)}.$$

此时

$$\sigma(\alpha_*) = \frac{\sqrt{\lambda_{\max}(H)} - \sqrt{\lambda_{\min}(H)}}{\sqrt{\lambda_{\max}(H)} + \sqrt{\lambda_{\min}(H)}} = \frac{\sqrt{\kappa(H)} - 1}{\sqrt{\kappa(H)} + 1}.$$

 HSS 自从被提出来以后, 很多学者将其进行了推广, 如 PSS, NSS, AHSS, PHSS 等, 感兴趣的读者可以参考相关文献.

6.6 课后习题

练习 6.1 已知 $A = \begin{bmatrix} a & 1 & 3 \\ 1 & a & 2 \\ -3 & 2 & a \end{bmatrix} \in \mathbb{R}^{3 \times 3}$, 且 Jacobi 方法收敛, 求 a 的取值范围.

练习 6.2 写出求解线性方程组 $Ax = b$ 的 Jacobi, GS 和 SOR 的分量形式, 并讨论它们的收敛性, 其中 $A = \begin{bmatrix} 5 & -3 & 2 \\ -3 & 6 & -1 \\ 2 & -1 & 3 \end{bmatrix}$, $b = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$.

练习 6.3 若存在非奇异矩阵 W , 使得 $W(I - G)W^{-1}$ 是对称正定矩阵, 则称迭代法

$$x^{(k+1)} = Gx^{(k)} + g, \quad k = 0, 1, 2, \dots$$

是**可对称化**的. 现假定上述迭代法是可对称化的, 试证

- (1) G 的特征值全部为实数, 且都小于 1;
- (2) 存在 γ , 使得下面的**外推迭代法**收敛

$$x^{(k+1)} = (1 - \gamma)x^{(k)} + \gamma(Gx^{(k)} + g), \quad k = 0, 1, 2, \dots$$

练习 6.4 设 $A = \begin{bmatrix} 2 & 0 \\ 3 & 1 \end{bmatrix}$, 构造迭代法

$$x^{(k+1)} = x^{(k)} + \alpha(b - Ax^{(k)}), \quad k = 0, 1, 2, \dots$$

问: 当 $\alpha \in \mathbb{R}$ 取何值时, 该迭代法收敛, 什么时候收敛最快?

练习 6.5 设 $A \in \mathbb{R}^{n \times n}$. 证明: $\rho(A) < 1$ 的充要条件是 $I - A$ 非奇异, 且 $(I - A)^{-1}(I + A)$ 的特征值具有正实部.

练习 6.6 设 $A \in \mathbb{R}^{n \times n}$ 是对称三对角矩阵

$$A = \begin{bmatrix} a & b & & \\ b & \ddots & \ddots & \\ & \ddots & \ddots & b \\ & & b & a \end{bmatrix}.$$

计算矩阵 A 的特征值和特征向量.



练习 6.7 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $B \in \mathbb{R}^{n \times k}$ 列满秩, 试证明 $B^T A B$ 对称正定.

练习 6.8* 设 $A = D - E - E^* \in \mathbb{C}^{n \times n}$, 其中 D 是 Hermite 正定的. 令

$$G_\omega = (D - \omega E)^{-1}[(1 - \omega)D + \omega E^*],$$

其中 ω 是实数. 证明 $\rho(G_\omega) < 1$ 当且仅当

(1) A 是正定的, 且 $0 < \omega < 2$, 或者

(2) A 是负定的, 且 $\omega \notin [0, 2]$.

(提示: 利用和仿照引理 6.13 中的证明思路)

练习 6.9 设 $A \in \mathbb{R}^{n \times n}$ 严格对角占优, 证明 $\rho(G_{GS}) < 1$.

(注: 不能直接使用定理 6.7 以及其证明过程)

练习 6.10 设迭代格式 $x^{(k+1)} = Gx^{(k)} + g$, 其中 G 对称且 $\lambda(G) \in [\alpha, \beta]$, $-1 < \alpha \leq \beta < 1$. 试构造出相应的 Chebyshev 加速方法.

练习 6.11 (定理 6.36) 设 $\lambda_{\max} \geq \lambda_{\min} > 0$, 则最小最大问题

$$\min_{\alpha > 0} \max_{\lambda_{\min} \leq \lambda \leq \lambda_{\max}} \left| \frac{\alpha - \lambda}{\alpha + \lambda} \right|$$

的解为

$$\alpha_* = \sqrt{\lambda_{\min} \lambda_{\max}}.$$

练习 6.12* 设 $A \in \mathbb{R}^{n \times n}$ 是对称正定的. 证明: 对任意正实数 α , 都有

$$\rho((\alpha I + A)^{-1}(\alpha I - A)) < 1.$$

如果 A 只是正定呢?

练习 6.13* (定理 6.8) 设 $A \in \mathbb{R}^{n \times n}$, 若 A 是不可约对角占优, 证明: Jacobi 迭代法和 G-S 迭代法都收敛.

练习 6.14* 证明: 对任意矩阵 $A \in \mathbb{C}^{n \times n}$ 和任意正数 $\varepsilon > 0$, 存在非奇异矩阵 L , 使得

$$\|A\|_L \leq \rho(A) + \varepsilon,$$

其中 $\|A\|_L \triangleq \|LAL^{-1}\|_2$.

练习 6.15* 设 $A \in \mathbb{R}^{n \times n}$. 证明: $\rho(A) < 1$ 的充要条件是存在对称正定矩阵 $P \in \mathbb{R}^{n \times n}$ 使得 $P - APA^T$ 正定.

..... 以下为可选题

练习 6.16 证明 Richardson 迭代法的收敛性, 即定理 6.11.

练习 6.17* 设 $A \in \mathbb{R}^{n \times n}$, 如果矩阵分裂 $A = M - N$ 满足 $M^{-1} \geq 0$, $N \geq 0$, 则称这个分裂为**正则分裂**. 假定 $A^{-1} \geq 0$, 且 $A = M - N$ 是正则分裂, 试证明

$$\rho(M^{-1}N) = \frac{\rho(A^{-1}N)}{1 + \rho(A^{-1}N)} < 1.$$

练习 6.18* 设 $A \in \mathbb{R}^{n \times n}$ 严格对角占优且非负, 则结论 $\rho(G_{GS}) \leq \rho(G_J)$ 是否成立? 若成立则给



出证明, 若不成立则给出反例.

练习 6.19 初值的选取对迭代法的收敛速度也有着很大的影响, 请以 Jacobi 迭代法为例, 初值取何时收敛最快 (真解除外)? 什么初值收敛最慢?

练习 6.20* 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $B \in \mathbb{R}^{n \times m}$ 满秩, 其中 $n \geq m$. 记 A 的最大和最小特征值分别为 μ_1 和 μ_n , 记 B 的最大和最小奇异值分别为 σ_1 和 σ_m . 证明: $\mathcal{A} = \begin{bmatrix} A & B \\ B^\top & 0 \end{bmatrix}$ 的特征值满足

$$\lambda(\mathcal{A}) \in I^- \cup I^+,$$

其中

$$I^- = \left[\frac{1}{2} \left(\mu_n - \sqrt{\mu_n^2 + 4\sigma_1^2} \right), \frac{1}{2} \left(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2} \right) \right],$$

$$I^+ = \left[\mu_n, \frac{1}{2} \left(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2} \right) \right].$$

练习 6.21* 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $B \in \mathbb{R}^{n \times m}$ 满秩, $C \in \mathbb{R}^{m \times m}$ 对称半正定, 其中 $n \geq m$, 记 A 的最大和最小特征值分别为 μ_1 和 μ_n , C 的最大特征值为 ν_1 , B 的最大和最小奇异值分别为 σ_1 和 σ_m . 证明: $\mathcal{A} = \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}$ 的特征值满足

$$\lambda(\mathcal{A}) \in I^- \cup I^+,$$

其中

$$I^- = \left[\frac{1}{2} \left(\mu_n - \nu_1 - \sqrt{(\mu_n + \nu_1)^2 + 4\sigma_1^2} \right), \frac{1}{2} \left(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2} \right) \right],$$

$$I^+ = \left[\mu_n, \frac{1}{2} \left(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2} \right) \right].$$

练习 6.22* 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $B \in \mathbb{R}^{n \times m}$ 满秩, 其中 $n \geq m$. 记 A 的最小特征值为 μ_n . 证明: 如果 $\mu_n \geq 4\|S\|_2$, 其中 $S = B^\top A^{-1}B$, 则 $\tilde{\mathcal{A}} = \begin{bmatrix} A & B \\ -B^\top & 0 \end{bmatrix}$ 的特征值都是正实数.

练习 6.23* 试举例说明, 存在线性方程组 $Ax = b$, 使得 Jacobi 迭代法收敛, 但 G-S 方法不收敛.

..... 以下为实践题

练习 6.24 写出 Poisson 方程红黑排序的 SOR 和 SSOR 方法的迭代格式, 并编程实现. (以例 6.2 中的 Poisson 方程为例)



第七讲 Krylov 子空间迭代法

Krylov 子空间迭代法 (带预处理) 是当前求解大规模稀疏线性方程组的首选方法. 本讲主要介绍 Krylov 子空间迭代法的一般思想和步骤, 以及两个典型的 Krylov 子空间迭代法: 共轭梯度法 (Conjugate Gradient, CG) 和 GMRES (Generalized Minimal Residual, GMRES) 方法.

子空间迭代法的基本思想

基本思想: 在一个维数较低的子空间中寻找解析解的一个“最佳”近似.

一般步骤: 子空间迭代法的主要过程可以分解为下面三步:

- (1) 寻找合适的子空间;
- (2) 在该子空间中求“最佳近似解”;
- (3) 若这个近似解满足精度要求, 则停止计算;
否则, 重新构造一个新的子空间 (即更新子空间), 并返回第 (2) 步.

关键问题:

- (1) 如何选择和更新子空间; → Krylov 子空间
- (2) 如何在给定的子空间中寻找“最佳近似解”. → 投影或最佳近似

关于 Krylov 子空间迭代法的相关资料

- Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edition, 2003. [95]
- A. Greenbaum, *Iterative Methods for Solving Linear Systems*, 1997. [52]
- G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th, 2013. [49]
- J. Demmel, *Applied Numerical Linear Algebra*, 1997. [23]
- L.N. Trefethen and D. Bau, *Numerical Linear Algebra*, 1997. [103]

7.1 Krylov 子空间

设 $A \in \mathbb{R}^{n \times n}$, $r \in \mathbb{R}^n$, 则由 A 和 r 生成的 **Krylov 子空间** 定义为

$$\mathcal{K}_m(A, r) = \text{span}\{r, Ar, A^2r, \dots, A^{m-1}r\}.$$

 为了书写方便, 在不引起歧义的情况下, 本讲义有时会将 $\mathcal{K}_m(A, r)$ 简记为 $\mathcal{K}_m$.

Krylov 子空间迭代法就是在 $\mathcal{K}_m(A, r)$ 中寻找最佳近似解. 设解析解在 $\mathcal{K}_m$ 中的最佳近似为 $x^{(m)}$. 我们假定 $r, Ar, A^2r, \dots, A^{m-1}r$ 线性无关, 则 $\dim \mathcal{K}_m = m$. 令 $v_1, v_2, \dots, v_m$ 是 $\mathcal{K}_m$ 的

一组基, 则 $\mathcal{K}_m$ 中的任意向量 x 都可表示为

$$x = y_1 v_1 + y_2 v_2 + \cdots + y_m v_m = V_m y,$$

其中 $y = [y_1, y_2, \dots, y_m]^T$ 为线性表出系数, $V_m = [v_1, v_2, \dots, v_m]$. 于是, 寻找“最佳近似解” $x^{(m)}$ 就转化为下面两个子问题:

- (1) 寻找一组合适的基 $v_1, v_2, \dots, v_m$;
- (2) 求出 $x^{(m)}$ 在这组基下面的线性表出系数 $y^{(m)} = [y_1^{(m)}, y_2^{(m)}, \dots, y_m^{(m)}]^T$.

7.1.1 Arnoldi 过程

首先考虑基的选取. 假定 $r, Ar, A^2r, \dots, A^{m-1}r$ 线性无关, 因此它们就自然地组成 $\mathcal{K}_m$ 的一组基. 但为了确保算法的稳定性, 我们通常希望选取一组标准正交基. 这并不困难, 只需对向量组 $\{r, Ar, A^2r, \dots, A^{m-1}r\}$ 进行 Gram-Schmidt 单位正交化即可.

对这个过程略加修改, 就得到下面的 **Arnoldi 过程**.

算法 7.1 Arnoldi 过程

```

1:  $v_1 = r / \|r\|_2$ 
2: for  $j = 1$  to  $m - 1$  do
3:    $z = Av_j$ 
4:   for  $i = 1$  to  $j$  do    % MGS
5:      $h_{i,j} = (v_i, z)$ 
6:      $z = z - h_{i,j}v_i$ 
7:   end for
8:    $h_{j+1,j} = \|z\|_2$ 
9:   if  $h_{j+1,j} = 0$  then
10:    break
11:  end if
12:   $v_{j+1} = z / h_{j+1,j}$ 
13: end for
```

 注意: Arnoldi 过程不是直接对 $r, Ar, A^2r, \dots, A^{m-1}r$ 进行 Gram-Schmidt 单位正交化, 而是按递归方式依次生成 $v_1, Av_1, Av_2, Av_3, \dots$, 然后进行单位正交化.

可以证明, Arnoldi 过程生成的向量 $v_1, v_2, \dots, v_m$ 构成 $\mathcal{K}_m$ 的一组标准正交基.

引理 7.1 如果 Arnoldi 过程不中断, 则

$$\mathcal{K}_m = \text{span}\{v_1, v_2, \dots, v_m\}.$$

(留作练习, 归纳法)



记 $V_m = [v_1, v_2, \dots, v_m]$,

$$H_{m+1,m} = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,m} \\ h_{2,1} & h_{2,2} & h_{2,3} & \cdots & h_{2,m} \\ & h_{3,2} & h_{3,3} & \cdots & h_{3,m} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & h_{m,m} \\ & & & & h_{m+1,m} \end{bmatrix} \in \mathbb{R}^{(m+1) \times m},$$

则由 Arnoldi 过程可知

$$h_{j+1,j}v_{j+1} = Av_j - h_{1,j}v_1 - h_{2,j}v_2 - \cdots - h_{j,j}v_j,$$

即

$$Av_j = \sum_{i=1}^{j+1} h_{i,j}v_i = V_{m+1} \begin{bmatrix} h_{1,j} \\ \vdots \\ h_{j+1,j} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = V_{m+1} H_{m+1,m}(:, j).$$

所以有

$$AV_m = V_{m+1} H_{m+1,m} = V_m H_m + h_{m+1,m} v_{m+1} e_m^T, \quad (7.1)$$

其中 H_m 是由 $H_{m+1,m}$ 的前 m 行组成的矩阵, 即 $H_m = H_{m+1,m}(1:m, 1:m)$, $e_m = [0, \dots, 0, 1]^T \in \mathbb{R}^m$. 由于 V_m 是列正交矩阵, 上式两边同乘 V_m^T 可得

$$V_m^T AV_m = H_m. \quad (7.2)$$

等式 (7.1) 和 (7.2) 是 Arnoldi 过程的两个重要性质. 这两个性质也可以通过下图来表示.

需要指出的是, 如果 $r, Ar, A^2r, \dots, A^{m-1}r$ 线性相关, 则 Arnoldi 过程就会提前中断. 此时, 我们会得到一个不变子空间.

定理 7.2 如果 Arnoldi 过程在第 k 步时中断, 即 $h_{k+1,k} = 0$, 其中 $k < m$, 则有 $AV_k = V_k H_k$, 即 $\mathcal{K}_k$ 是 A 的一个不变子空间.

(留作课外自习, 直接利用等式 (7.1))



7.1.2 Lanczos 过程

如果 A 是对称矩阵, 则 H_m 为对称三对角矩阵. 为了书写方便, 将其改记为 T_m , 即

$$T_m = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \ddots & \ddots & & \\ & \ddots & \ddots & \beta_{m-1} & \\ & & \beta_{m-1} & \alpha_m & \end{bmatrix}. \quad (7.3)$$

与 Arnoldi 过程类似, 我们有下面的性质

$$AV_m = V_m T_m + \beta_m v_{m+1} e_m^T, \quad (7.4)$$

$$V_m^T AV_m = T_m. \quad (7.5)$$

考察关系式 (7.4) 两边的第 j 列可知

$$\beta_j v_{j+1} = Av_j - \alpha_j v_j - \beta_{j-1} v_{j-1}, \quad j = 1, 2, \dots,$$

这里我们令 $v_0 = 0$ 和 $\beta_0 = 0$. 这就是著名的**三项递推公式**.

根据这个三项递推公式, Arnoldi 过程可简化为下面的**Lanczos 过程**.

算法 7.2 Lanczos 过程

```

1: Set  $v_0 = 0$  and  $\beta_0 = 0$ 
2:  $v_1 = r / \|r\|_2$ 
3: for  $j = 1$  to  $m - 1$  do
4:    $z = Av_j$ 
5:    $\alpha_j = (v_j, z)$ 
6:    $z = z - \alpha_j v_j - \beta_{j-1} v_{j-1}$ 
7:    $\beta_j = \|z\|_2$ 
8:   if  $\beta_j = 0$  then
9:     break
10:  end if
11:   $v_{j+1} = z / \beta_j$ 
12: end for
```

 由算法可知, 在 Lanczos 过程中, 我们是通过三项递推公式来计算 v_{j+1} 的, 即只需将 Av_j 与 v_j 和 v_{j-1} 正交化即可, 而无需与前面所有的 v_i 都正交化, 因此运算复杂度比 Arnoldi 过程要小很多, 这就是矩阵的对称性质所带来的优势.

可以证明, 由 Lanczos 过程得到的向量组 $\{v_1, v_2, \dots, v_m\}$ 是单位正交的.



定理 7.3 设 $\{v_1, v_2, \dots, v_m\}$ 是由 Lanczos 过程得到的向量组, 则

$$(v_i, v_j) = \delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases} \quad i, j = 1, 2, \dots, m.$$

(留作练习)

7.1.3 Krylov 子空间方法一般格式

Krylov 子空间迭代法的一般过程

- (1) 令 $m = 1$
- (2) 构造 Krylov 子空间 $\mathcal{K}_m$ (通过 Arnoldi 过程或 Lanczos 过程构造一组标准正交基);
- (3) 找出仿射空间 $x^{(0)} + \mathcal{K}_m$ 中的“最佳近似”解;
- (4) 如果这个近似解满足精度要求, 则迭代结束; 否则令 $m \leftarrow m + 1$, 即将 Krylov 子空间的维数增加一维, 并返回第 (2) 步.

 在实际计算中, 为了充分利用迭代初值中所包含的有用信息, 我们通常是在仿射空间 $x^{(0)} + \mathcal{K}_m$ 中寻找“最佳近似解”.

算法 7.3 Krylov 子空间迭代算法

- 1: 选取初始向量 $x^{(0)}$
- 2: 计算 $r_0 = b - Ax^{(0)}$, $v_1 = r_0 / \|r_0\|_2$
- 3: 寻找“最佳近似解”: $x^{(1)} \in x^{(0)} + \mathcal{K}_1 = x^{(0)} + \text{span}\{v_1\}$
- 4: **if** $x^{(1)}$ 满足精度要求 **then**
- 5: 终止迭代
- 6: **end if**
- 7: **for** $m = 2$ to n **do**
- 8: 调用 Arnoldi 或 Lanczos 过程计算向量 v_m
- 9: 寻找“最佳近似解”: $x^{(m)} \in x^{(0)} + \mathcal{K}_m = x^{(0)} + \text{span}\{v_1, \dots, v_m\}$
- 10: **if** $x^{(m)}$ 满足精度要求 **then**
- 11: 终止迭代
- 12: **end if**
- 13: **end for**

7.1.4 最佳近似解

子空间的选取和更新问题可以通过 Krylov 子空间来解决. 下面需要考虑的是如何寻找方程组在仿射空间 $x^{(0)} + \mathcal{K}_m$ 中的“最佳近似”解 $x^{(m)}$.



首先, 我们必须给出“最佳”的定义, 即 $x^{(m)}$ 满足什么条件时才是“最佳”的, 不同的定义会导出不同的算法.

我们很自然地想到用近似解与真解之间的距离来衡量“最佳”, 即使得 $x^{(m)} - x_*$ 在某种意义上最小, 如 $\|x^{(m)} - x_*\|_2$ 达到最小. 但是由于 x_* 是未知的, 因此这种方式往往是不实用的.

实用的“最佳”判别方式有:

- (1) 使得 $\|r_m\|_2 = \|b - Ax^{(m)}\|_2$ 最小, 即极小化残量 $r_m = b - Ax^{(m)}$. 这种方式是可行的, 当 A 对称时, 相应的算法即为 **MINRES 方法**, 当 A 不对称时, 相应的算法即为 **GMRES 方法**;
- (2) 若 A 对称正定, 我们也可以极小化残量的能量范数 $\|r_m\|_{A^{-1}}$, 相应的算法即为 **CG 方法**. 事实上, 我们有

$$\begin{aligned}\|r_m\|_{A^{-1}} &\triangleq \left(r_m^T A^{-1} r_m\right)^{\frac{1}{2}} = \left((b - Ax^{(m)})^T A^{-1} (b - Ax^{(m)})\right)^{\frac{1}{2}} \\ &= \left((A^{-1}b - x^{(m)})^T A (A^{-1}b - x^{(m)})\right)^{\frac{1}{2}} \\ &= \left((x_* - x^{(m)})^T A (x_* - x^{(m)})\right)^{\frac{1}{2}} \\ &= \|x_* - x^{(m)}\|_A.\end{aligned}$$

因此 CG 方法也可以看作是极小化能量范数 $\|x_* - x^{(m)}\|_A$ 的算法.

7.2 GMRES 方法

7.2.1 算法描述

GMRES 方法是目前求解大规模非对称线性方程组的最常用算法之一. 在该算法中, “最佳近似解”的判别法则为“使得 $\|r_m\|_2 = \|b - Ax^{(m)}\|_2$ 最小”, 即

$$x^{(m)} = \arg \min_{x \in x^{(0)} + \mathcal{K}_m} \|b - Ax\|_2. \quad (7.6)$$

下面我们就根据这个最优性条件来推导 GMRES 方法.

设迭代初始向量为 $x^{(0)}$, 则对任意向量 $x \in x^{(0)} + \mathcal{K}_m$, 可设 $x = x^{(0)} + V_m y$, 其中 $y \in \mathbb{R}^m$, 因此我们只需求出 y 即可. 直接计算可得

$$\begin{aligned}r &= b - Ax = b - A(x^{(0)} + V_m y) \\ &= r_0 - AV_m y \\ &= \beta v_1 - V_{m+1} H_{m+1, m} y \quad (\text{令 } \beta = \|r_0\|_2) \\ &= V_{m+1} (\beta e_1 - H_{m+1, m} y),\end{aligned}$$

其中 $\beta = \|r_0\|_2$. 由于 V_{m+1} 单位列正交, 所以

$$\|r\|_2 = \|V_{m+1}(\beta e_1 - H_{m+1, m} y)\|_2 = \|\beta e_1 - H_{m+1, m} y\|_2.$$

于是最优性条件 (7.6) 就等价于

$$y^{(m)} = \arg \min_{y \in \mathbb{R}^m} \|\beta e_1 - H_{m+1, m} y\|_2. \quad (7.7)$$



这是一个最小二乘问题. 由于 $H_{m+1,m}$ 是一个上 Hessenberg 矩阵, 而且通常 m 不是很大 (一般有 $m \ll n$), 所以我们可以用基于 Givens 变换的 QR 分解来求解.

由此, 我们就可以得到 GMRES 方法的基本框架, 见算法 7.4.

算法 7.4 GMRES 方法基本框架

```

1: 选取初值  $x^{(0)}$ , 停机准则  $\text{tol} > 0$  和最大迭代步数  $\text{itermax}$ 
2:  $r_0 = b - Ax^{(0)}$ ,  $\beta = \|r_0\|_2$ 
3:  $v_1 = r_0/\beta$ 
4: for  $j = 1$  to  $\text{itermax}$  do
5:    $w = Av_j$ 
6:   for  $i = 1$  to  $j$  do    % Arnoldi 过程
7:      $h_{i,j} = (v_i, w)$ 
8:      $w = w - h_{i,j}v_i$ 
9:   end for
10:   $h_{j+1,j} = \|w\|_2$ 
11:  if  $h_{j+1,j} = 0$  then    % 迭代中断, 算法收敛
12:     $m = j$ , break
13:  end if
14:   $v_{j+1} = w/h_{j+1,j}$ 
15:   $\text{relres} = \|r_j\|_2/\beta$     % 计算相对残量范数
16:  if  $\text{relres} < \text{tol}$  then % 算法收敛检测
17:     $m = j$ , break
18:  end if
19: end for
20: 求解最小二乘问题 (7.7), 得到  $y^{(m)}$ 
21:  $x^{(m)} = x^{(0)} + V_m y^{(m)}$ 

```

算法收敛准则

我们这里是根据相对残量的大小来判断算法是否收敛, 即

$$\frac{\|b - Ax^{(j)}\|_2}{\|b - Ax^{(0)}\|_2} < \text{tol}.$$

这也是求解线性方程组的迭代法普遍采用的收敛准则.

7.2.2 具体实施细节

算法 7.4 只是一个基本框架, 具体实施时需要解决以下两个问题:

- (1) 如何计算残量 $r_j \triangleq b - Ax^{(j)}$ 的范数? (算法第 15 步)
- (2) 如何求解最小二乘问题 (7.7)? (算法第 20 步)



这两个问题可以同时处理. 首先, 我们采用 QR 分解来求解最小二乘问题 (7.7). 设 $H_{m+1,m}$ 的 QR 分解为

$$H_{m+1,m} = Q_{m+1}^T R_{m+1,m},$$

其中 $Q_{m+1} \in \mathbb{R}^{(m+1) \times (m+1)}$ 是正交矩阵, $R_{m+1,m} \in \mathbb{R}^{(m+1) \times m}$ 是上三角矩阵, 则

$$\|\beta e_1 - H_{m+1,m} y\|_2 = \|\beta Q_{m+1} e_1 - R_{m+1,m} y\|_2 = \left\| \beta q_1 - \begin{bmatrix} R_m \\ 0 \end{bmatrix} y \right\|_2, \quad (7.8)$$

其中 $R_m \in \mathbb{R}^{m \times m}$ 是非奇异上三角矩阵 (这里假定 $H_{m+1,m}$ 不可约). 所以最小二乘问题 (7.7) 的解为

$$y^{(m)} = \beta R_m^{-1} q_1(1:m), \quad (7.9)$$

且

$$\|r_m\|_2 = \|b - Ax^{(m)}\|_2 = \|\beta e_1 - H_{m+1,m} y^{(m)}\|_2 = \beta |q_1(m+1)|, \quad (7.10)$$

其中 $q_1(m+1)$ 表示 q_1 的第 $m+1$ 个分量.

所以只要知道 $H_{m+1,m}$ 的 QR 分解, 则上述两个问题就都解决了. 而且由于 $H_{m+1,m}$ 是上 Hessenberg 矩阵, 因此可以采用 Givens 变换来计算, 非常高效.

由算法 7.4 可知, 最小二乘解只需计算一次 (即算法第 20 步), 但残量范数需要每次迭代都计算 (即算法第 15 步), 这意味着我们需要计算所有 $H_{j+1,j}$ 的 QR 分解, 计算成本显然很高. 幸运的是, 由于这些 Hessenberg 矩阵之间的嵌套关系 (即 $H_{j,j-1}$ 是 $H_{j+1,j}$ 的子矩阵), 我们可以利用递推方法来计算它们的 QR 分解.

$H_{j+1,j}$ 的 QR 分解的递推计算方法

(1) 当 $j=1$ 时, $H_{21} = \begin{bmatrix} h_{11} \\ h_{21} \end{bmatrix}$, 构造 Givens 变换 G_1 使得

$$G_1 H_{21} = \begin{bmatrix} * \\ 0 \end{bmatrix} = R_{21}, \quad \text{即} \quad H_{21} = G_1^T R_{21}.$$

(2) 假定已经知道 $H_{j,j-1}$ 的 QR 分解, 即存在 $G_1, G_2, \dots, G_{j-1}$, 使得

$$(G_{j-1} \cdots G_2 G_1) H_{j,j-1} = R_{j,j-1},$$

即

$$H_{j,j-1} = (G_{j-1} \cdots G_2 G_1)^T R_{j,j-1} \triangleq Q_j^T R_{j,j-1}.$$

 为了书写方便, 这里假定 G_i 的维数自动扩张, 以满足矩阵乘积的需要.



(3) 考虑 $H_{j+1,j}$ 的 QR 分解. 易知

$$H_{j+1,j} = \begin{bmatrix} H_{j,j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}, \quad \text{其中 } h_j = [h_{1,j}, h_{2,j}, \dots, h_{j,j}]^T.$$

所以有

$$\begin{bmatrix} Q_j & 0 \\ 0 & 1 \end{bmatrix} H_{j+1,j} = \begin{bmatrix} R_{j,j-1} & Q_j h_j \\ 0 & h_{j+1,j} \end{bmatrix} = \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \hat{h}_{j,j} \\ 0 & h_{j+1,j} \end{bmatrix},$$

其中 $\tilde{h}_{j-1}$ 是 $Q_j h_j$ 的前 $j-1$ 个元素组成的向量, $\hat{h}_{j,j}$ 是 $Q_j h_j$ 的最后一个元素. 构造 Givens 变换 G_j (对应上式矩阵中的最后一列的最后两个元素):

$$G_j = \begin{bmatrix} I_{j-1} & 0 & 0 \\ 0 & c_j & s_j \\ 0 & -s_j & c_j \end{bmatrix} \in \mathbb{R}^{(j+1) \times (j+1)},$$

其中

$$c_j = \frac{\hat{h}_{j,j}}{\tilde{h}_{j,j}}, \quad s_j = \frac{h_{j+1,j}}{\tilde{h}_{j,j}}, \quad \tilde{h}_{j,j} = \sqrt{\hat{h}_{j,j}^2 + h_{j+1,j}^2}.$$

令

$$Q_{j+1} = G_j \begin{bmatrix} Q_j & 0 \\ 0 & 1 \end{bmatrix},$$

则

$$Q_{j+1} H_{j+1,j} = G_j \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \hat{h}_{j,j} \\ 0 & h_{j+1,j} \end{bmatrix} = \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \tilde{h}_{j,j} \\ 0 & 0 \end{bmatrix} \triangleq R_{j+1,j}.$$

于是可得 $H_{j+1,j}$ 的 QR 分解 $H_{j+1,j} = Q_{j+1}^T R_{j+1,j}$.

计算过程细节分析

► 从 $H_{j,j-1}$ 的 QR 分解到 $H_{j+1,j}$ 的 QR 分解, 我们需要计算 $Q_j h_j$, 即将之前的 $j-1$ 个 Givens 变换作用到 $H_{j+1,j}$ 的最后一列的前 j 个元素上, 所以我们需要保留所有的 Givens 变换;

► 残量计算: $\|r_j\|_2 = |\beta q_1(j+1)| = |\beta Q_{j+1}(j+1, 1)|$, 即

$$G_j G_{j-1} \cdots G_2 G_1 (\beta e_1)$$

的最后一个分量的绝对值. 由于在计算 r_{j-1} 时就已经计算出 $G_{j-1} \cdots G_2 G_1 (\beta e_1)$, 因此这里只需做一次 Givens 变换即可;

► $y^{(m)}$ 的计算: 当相对残量满足精度要求时, 需要计算 $y^{(m)} = R_m^{-1} q_1(1:m)$, 而 q_1 即为 $G_m G_{m-1} \cdots G_2 G_1 (\beta e_1)$.

因此在实际计算时, 我们无需计算 Q , 只要保留所有 Givens 变换, 并计算 R (可以直接覆盖 H) 和 q_1 (另外用一个变量来存储).



算法 7.5 实用 GMRES 方法

```

1: 选取初值  $x^{(0)}$ , 停机准则  $\text{tol} > 0$  和最大迭代步数  $\text{itermax}$ 
2:  $r_0 = b - Ax^{(0)}$ ,  $\beta = \|r_0\|_2$ 
3:  $v_1 = r_0/\beta$ 
4:  $\xi = \beta e_1$     % 存储  $q_1$ 
5: for  $j = 1$  to  $\text{itermax}$  do
6:    $w = Av_j$ 
7:   for  $i = 1$  to  $j$  do    % Arnoldi 过程
8:      $h_{i,j} = (v_i, w)$ 
9:      $w = w - h_{i,j}v_i$ 
10:  end for
11:   $h_{j+1,j} = \|w\|_2$ 
12:  for  $i = 1$  to  $j - 1$  do    % 计算  $G_{j-1} \cdots G_2 G_1 H_{j+1,j}(1:j, j)$ 
13:    
$$\begin{bmatrix} h_{ij} \\ h_{i+1,j} \end{bmatrix} = \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix} \begin{bmatrix} h_{ij} \\ h_{i+1,j} \end{bmatrix}$$

14:  end for
15:  if  $h_{j+1,j} = 0$ , then break    % 迭代中断, 算法收敛
16:   $v_{j+1} = w/h_{j+1,j}$ 
17:  if  $|h_{jj}| > |h_{j+1,j}|$  then    % 构造 Givens 变换  $G_j$ 
18:     $\tau = h_{j+1,j}/h_{jj}$ ,  $c_j = 1/\sqrt{1+\tau^2}$ ,  $s_j = c_j\tau$ 
19:  else
20:     $\tau = h_{jj}/h_{j+1,j}$ ,  $s_j = 1/\sqrt{1+\tau^2}$ ,  $c_j = s_j\tau$ 
21:  end if
22:   $h_{jj} = c_j h_{jj} + s_j h_{j+1,j}$     % 计算  $G_j H_{j+1,j}(1:j, j)$ 
23:   $h_{j+1,j} = 0$ 
24:  
$$\begin{bmatrix} \xi_j \\ \xi_{j+1} \end{bmatrix} = \begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} \begin{bmatrix} \xi_j \\ 0 \end{bmatrix}$$
    % 计算  $G_j(\beta G_{j-1} \cdots G_2 G_1 e_1)$ 
25:   $\text{relres} = |\xi_{j+1}|/\beta$     % 计算相对残量范数
26:  if  $\text{relres} < \text{tol}$ , then break    % 算法收敛检测
27: end for
28:  $m = j$ 
29:  $y^{(m)} = H(1:m, 1:m) \backslash \xi(1:m)$     % 求最小二乘问题的解, 回代求解
30:  $x^{(m)} = x^{(0)} + V_m y^{(m)}$ 
31: if  $\text{relres} < \text{tol}$  then
32:   输出近似解及相关信息
33: else
34:   输出算法失败信息
35: end if

```



7.2.3 GMRES 算法的中断

在上面的 GMRES 算法中, 如果执行到某一步时有 $h_{k+1,k} = 0$ (即算法 7.5 的第 15 步), 则算法会中断 (breakdown). 如果出现这种中断, 则我们就找到了精确解 (不考虑舍入误差).

定理 7.4 设 $A \in \mathbb{R}^{n \times n}$ 非奇异且 $r_0 \neq 0$. 若 $h_{i+1,i} \neq 0, i = 1, 2, \dots, j-1$, 则 $h_{j+1,j} = 0$ 当且仅当 $x^{(j)}$ 是方程组的精确解. (不考虑舍入误差) (板书)

7.2.4 带重启的 GMRES 方法

由于随着迭代步数的增加 (即 Krylov 子空间的维数不断增加), GMRES 方法的每一步所需的运算量和存储量都会越来越大 (正交化过程中的内积计算和向量更新次数会不断增加, Krylov 子空间的基向量个数也会不断增加). 因此当迭代步数很大时, GMRES 方法的成本就很高, 实用性大大减弱.

目前通常采用的一个解决方法就是重启, 即事先设定一个重启迭代步数 k , 如 $k = 20$ 或 50 , 等等, 如果 GMRES 方法达到这个迭代步数时仍不收敛, 则计算出方程组在 $x^{(0)} + \mathcal{K}_k$ 中的最佳近似解 $x^{(k)}$, 然后令 $x^{(0)} = x^{(k)}$, 并重新开始新的 GMRES 迭代. 不断重复该过程, 直到收敛为止, 这就是 **带重启的 GMRES 方法**.

算法 7.6 GMRES(k): 带重启的 GMRES 方法

```

1: 设定重启步数  $k$  ( $\ll n$ )
2: 选取初值  $x^{(0)}$ , 停机准则  $\text{tol} > 0$  和最大迭代步数  $\text{itermax}$ 
3:  $r_0 = b - Ax^{(0)}, \beta = \|r_0\|_2$ 
4: for  $\text{iter}=1$  to  $\text{ceil}(\text{itermax}/k)$  do    % 外循环
5:    $v_1 = r_0/\beta$ 
6:    $\xi = \beta e_1$ 
7:   for  $j = 1$  to  $k$  do
8:     调用 GMRES 循环
9:   end for
10:   $m = j$ 
11:   $y^{(m)} = H(1:m, 1:m) \backslash \xi(1:m)$ 
12:   $x^{(m)} = x^{(0)} + V_m y^{(m)}$ 
13:  if  $\text{relres} < \text{tol}$ , then break    % 算法收敛检测
14:   $x^{(0)} = x^{(m)}$     % 重启 GMRES
15:   $r_0 = b - Ax^{(0)}, \beta = \|r_0\|_2$ 
16: end for
17: if  $\text{relres} < \text{tol}$  then
18:   输出近似解及相关信息
19: else
20:   输出算法失败信息

```



21: end if

几点注记

带重启的 GMRES 方法需要注意的问题:

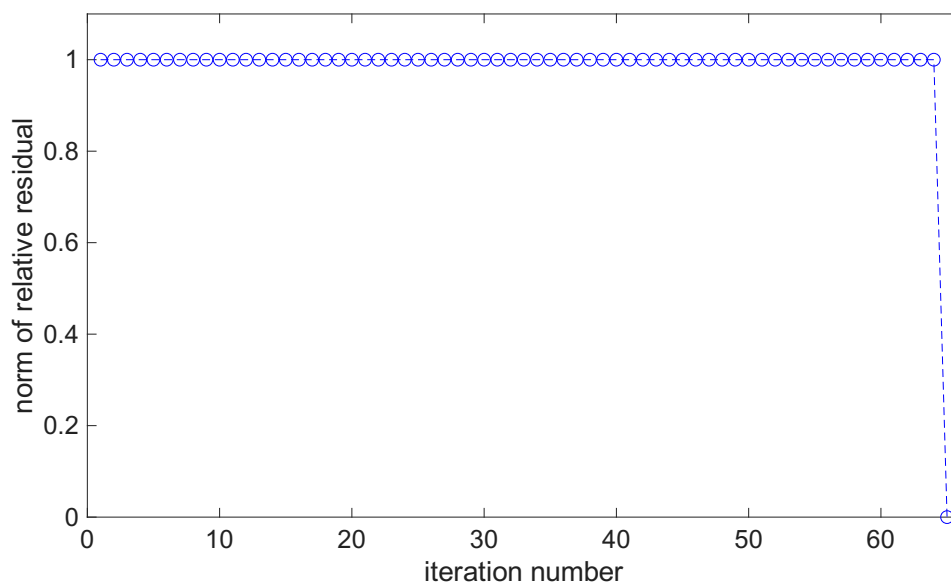
- (1) 如何选取合适的重启步数 k ? 一般只能依靠经验来选取;
- (2) 不带重启的 GMRES 方法能保证算法的收敛性, 但带重启的 GMRES 方法却不一定收敛, 有时可能出现停滞现象 (stagnation).

例 7.1 (带重启的 GMRES 方法: 停滞现象) 考虑线性方程组 $Ax = b$, 其中

$$A = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

下图绘出了 $n = 64$ 时的 (相对) 残量范数下降曲线.

(GMRES_restart_stagnation.m)



 对于该问题, 重启 GMRES 将导致算法失败.

例 7.2 考虑下面的带齐次 Dirichlet 边界条件的偏微分方程:

$$-\Delta u(x, y) + u_x(x, y) + u_y(x, y) + u(x, y) = f(x, y), \quad (7.11)$$

$$(x, y) \in \Omega \triangleq [0, 1] \times [0, 1]$$

$$u(x, 0) = u(x, 1) = u(0, y) = u(1, y) = 0, \quad 0 < x, y < 1.$$



二阶导数用五点差分离散, 一阶导数用中心差分离散, 步长 $h_x = h_y = 1/(N+1)$. 可得具体离散格式为

$$\frac{4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}}{h^2} + \frac{u_{i+1,j} - u_{i-1,j}}{2h} + \frac{u_{i,j+1} - u_{i,j-1}}{2h} + u_{i,j} = f_{i,j}.$$

对应的离散线性方程组为

$$[I \otimes T + T \otimes I + h(I \otimes D) + h(D \otimes I) + h^2 I]u = h^2 f,$$

其中 $T = \text{tridiag}(-1, 2, -1)$, $D = \text{tridiag}(-1/2, 0, 1/2)$.

在实际测试中, 我们假设解析解为

$$u(x, y) = xy(1-x)(1-y).$$

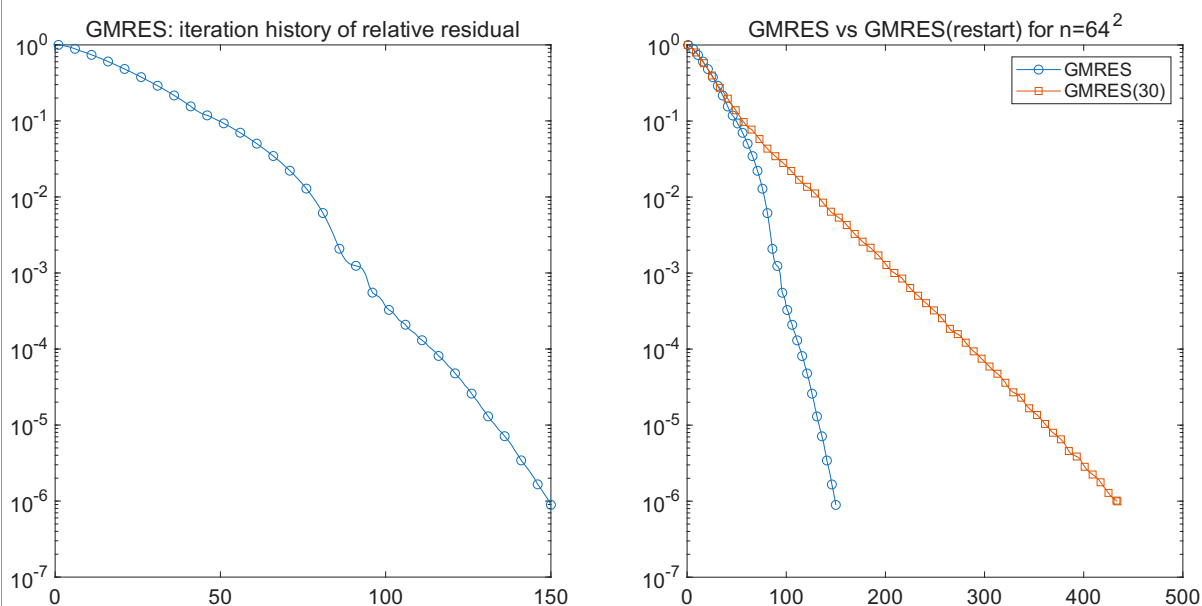
由此可以求出右端项

$$f(x, y) = (3-2x)(1-y)y + (3-2y)(1-x)x + x(1-x)y(1-y).$$

由于系数矩阵是非对称的, 因此用 GMRES 方法求解. 最大迭代步数设为 $\text{itermax} = N^2$, 迭代终止条件

$$\frac{\|r_k\|_2}{\|r_0\|_2} \leq \text{tol} \triangleq 10^{-6}.$$

下面左图是 $n = 64^2$ 时 GMRES 的相对残量下降曲线, 右图是 $n = 64^2$ 时的不带重启与带重启的 GMRES 收敛曲线. (GMRES_PDE_restart.m)



7.3 共轭梯度法

共轭梯度法 (Conjugate Gradient, CG) 是当前求解大规模对称正定线性方程组的首选方法. CG 算法可以从优化角度得出, 也可以从代数角度推导. 我们这里就从代数角度来介绍 CG 算法.



7.3.1 算法基本过程

CG 方法寻找最佳近似解的准则是极小化能量范数 $\|x^{(m)} - x_*\|_A$. 我们首先给出 CG 算法的一个重要性质, 然后根据这个性质来推导 CG 算法.

定理 7.5 设 A 对称正定, 则

$$x^{(m)} = \arg \min_{x \in x^{(0)} + \mathcal{K}_m} \|x - x_*\|_A \quad (7.12)$$

的充要条件是

$$x^{(m)} \in x^{(0)} + \mathcal{K}_m \quad \text{且} \quad b - Ax^{(m)} \perp \mathcal{K}_m. \quad (7.13)$$

(板书)

当 A 对称正定时, Arnoldi 过程就转化为 Lanczos 过程, 且

$$\begin{aligned} AV_m &= V_{m+1}T_{m+1,m} = V_mT_m + \beta_m v_{m+1}e_m^\top, \\ V_m^\top AV_m &= T_m, \end{aligned}$$

其中 $T_m = \text{tridiag}(\beta_i, \alpha_{i+1}, \beta_{i+1})$, 见 (7.3).

由定理 7.5 可知, CG 方法是在 $x^{(0)} + \mathcal{K}_m$ 中寻找最优近似解 $x^{(m)}$, 满足

$$b - Ax^{(m)} \perp \mathcal{K}_m. \quad (7.14)$$

根据这个性质, 我们就可以给出 CG 算法.

设 $x^{(m)} = x^{(0)} + V_my^{(m)}$, 其中 $y^{(m)} \in \mathbb{R}^m$. 由 (7.14) 可知

$$0 = V_m^\top(b - Ax^{(m)}) = V_m^\top(r_0 - AV_my^{(m)}) = V_m^\top(\beta v_1) - V_m^\top AV_my^{(m)} = \beta e_1 - T_my^{(m)}.$$

因此,

$$y^{(m)} = T_m^{-1}(\beta e_1).$$

于是可得

$$x^{(m)} = x^{(0)} + V_my^{(m)} = x^{(0)} + V_mT_m^{-1}(\beta e_1). \quad (7.15)$$

如果 $x^{(m)}$ 满足精度要求, 则计算结束. 否则令 $m \leftarrow m+1$, 继续下一步迭代. 这就是 CG 方法的基本过程.

算法细节分析

在实施 CG 算法时, 我们需要考虑下面两个问题.

- 最佳近似解由 (7.15) 给出, 因此需要计算 $T_m^{-1}(\beta e_1)$, 即求解一个以 T_m 为系数矩阵的线性方程组. 由于 A 对称正定, 所以 T_m 也对称正定, 而且通常 $m \ll n$, 因此我们可以用直接法 (Cholesky 分解或 LDL^T 分解) 求解. 另外, 每次迭代都要求解这种类型的方程组, 与 GMRES 方法类似, 根据 T_m 的嵌套性, 可以采用递推方法求解, 减少计算成本.
- 算法收敛准则: 我们仍然是基于相对残量范数的大小, 即

$$\frac{\|b - Ax^{(m)}\|_2}{\|b - Ax^{(0)}\|_2} < \text{tol}.$$



这意味着每次迭代都需要计算残量 $r_m = b - Ax^{(m)}$. 事实上, 在 CG 算法中, r_m 和 $x^{(m)}$ 是可以同时计算的.

根据以上分析, 下面给出 CG 算法的具体实施过程.

7.3.2 实用迭代格式

首先考虑 $x^{(m)}$ 的计算. 由于 A 是对称正定的, 因此我们可以借助三项递推公式来简化运算. 由于 T_m 对称正定, 所以存在 LDL^T 分解, 即 $T_m = L_m D_m L_m^T$. 于是

$$x^{(m)} = x^{(0)} + V_m z^{(m)} = x^{(0)} + V_m T_m^{-1} (\beta e_1^{(m)}) = x^{(0)} + (V_m L_m^{-T}) (\beta D_m^{-1} L_m^{-1} e_1^{(m)}),$$

其中 $e_1^{(m)}$ 表示 m 阶单位矩阵的第一列. 如果 $x^{(m)}$ 满足精度要求, 则计算结束. 否则我们需要计算 $x^{(m+1)}$, 即

$$x^{(m+1)} = x^{(0)} + V_{m+1} T_{m+1}^{-1} (\beta e_1^{(m+1)}) = x^{(0)} + (V_{m+1} L_{m+1}^{-T}) (\beta D_{m+1}^{-1} L_{m+1}^{-1} e_1^{(m+1)}).$$

这里假定 T_{m+1} 的 LDL^T 分解为 $T_{m+1} = L_{m+1} D_{m+1} L_{m+1}^T$.

下面考虑如何利用递推方式来计算 $x^{(m+1)}$. 记

$$\begin{aligned}\tilde{P}_m &\triangleq V_m L_m^{-T} = [\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_m] \in \mathbb{R}^{n \times m}, \\ z_m &\triangleq \beta D_m^{-1} L_m^{-1} e_1^{(m)} = [\eta_1, \dots, \eta_m]^T \in \mathbb{R}^m.\end{aligned}$$

由于 T_{m+1} 与 T_m 的嵌套关系 (即 T_m 是 T_{m+1} 的 m 阶顺序主子矩阵), 它们的 LDL^T 分解之间也存在递推关系, 由此我们可以给出下面的结论 (关于 $\tilde{P}_m$ 和 z_m 的递推关系).

引理 7.6 设 $\tilde{P}_m$ 和 y_m 由上面的式子所定义, 则下面的递推公式成立:

$$\begin{aligned}\tilde{P}_{m+1} &\triangleq V_{m+1} L_{m+1}^{-T} = [\tilde{P}_m, \tilde{p}_{m+1}], \\ z_{m+1} &\triangleq \beta D_{m+1}^{-1} L_{m+1}^{-1} e_1^{(m+1)} = [z_m^T, \eta_{m+1}]^T, \quad m = 1, 2, \dots\end{aligned}$$

(板书)

有了上面的性质, 我们就可以得到从 $x^{(m)}$ 到 $x^{(m+1)}$ 的递推公式

$$x^{(m+1)} = \tilde{P}_{m+1} y_{m+1} = [\tilde{P}_m, \tilde{p}_{m+1}] \begin{bmatrix} y_m \\ \eta_{m+1} \end{bmatrix} = x^{(m)} + \eta_{m+1} \tilde{p}_{m+1}.$$

为了判断近似解是否满足精度要求, 我们还需要计算残量. 通过直接计算可知, 残量满足下面的递推公式:

$$r_{m+1} = b - Ax^{(m+1)} = b - A(x^{(m)} + \eta_{m+1} \tilde{p}_{m+1}) = r_m - \eta_{m+1} A \tilde{p}_{m+1}.$$

另一方面, 我们有

$$\begin{aligned}r_m &= b - Ax^{(m)} = b - A(x^{(0)} + V_m z^{(m)}) \\ &= r_0 - AV_m z^{(m)} \\ &= \beta v_1 - V_m T_m z^{(m)} - \beta_m v_{m+1} e_m^T z^{(m)}\end{aligned}$$



$$= -\beta_m(e_m^T z^{(m)})v_{m+1},$$

即 r_m 与 v_{m+1} 平行. 记

$$r_m = \tau_m v_{m+1}, \quad m = 0, 1, 2, \dots, \quad (7.16)$$

其中

$$\tau_0 = \beta = \|r_0\|_2, \quad \tau_m = -\beta_m(e_m^T z^{(m)}), \quad m = 1, 2, \dots$$

引入变量

$$p_m \triangleq \tau_{m-1} \tilde{p}_m, \quad m = 1, 2, \dots$$

由 (??) 和 (7.16) 可知, $\{p_m\}$ 满足下面的递推公式:

$$p_{m+1} = \tau_m \tilde{p}_{m+1} = \tau_m(v_{m+1} - l_m \tilde{p}_m) = r_m + \mu_m p_m, \quad m = 1, 2, \dots, \quad (7.17)$$

其中 $\mu_m = -\frac{l_m \tau_m}{\tau_{m-1}}$. 于是我们就得到关于近似解和残量的递推计算公式

$$x^{(m+1)} = x^{(m)} + \eta_{m+1} \tilde{p}_{m+1} = x^{(m)} + \xi_{m+1} p_{m+1}, \quad (7.18)$$

$$r_{m+1} = r_m - \eta_{m+1} A \tilde{p}_{m+1} = r_m - \xi_{m+1} A p_{m+1}, \quad m = 1, 2, \dots, \quad (7.19)$$

其中 $\xi_{m+1} = \frac{\eta_{m+1}}{\tau_m}$.

下面考虑系数 ξ_{m+1} 和 μ_m 的计算方法. 首先给出下面的性质.

引理 7.7 对于共轭梯度法, 下面的结论成立:

- (1) $r_1, r_2, \dots, r_m$ 相互正交;
- (2) $p_1, p_2, \dots, p_m$ 相互 A -共轭 (或 A -正交), 即当 $i \neq j$ 时有 $p_i^T A p_j = 0$.

(板书)

下面给出 ξ_{m+1} 和 μ_m 的计算公式. 在等式 (7.17) 两边同时左乘 $p_{m+1}^T A$ 可得

$$p_{m+1}^T A p_{m+1} = p_{m+1}^T A r_m + \mu_m p_{m+1}^T A p_m = r_m^T A p_{m+1}.$$

再用 r_m^T 左乘方程 (7.19) 可得

$$0 = r_m^T r_{m+1} = r_m^T r_m - \xi_{m+1} r_m^T A p_{m+1}.$$

于是

$$\xi_{m+1} = \frac{r_m^T r_m}{r_m^T A p_{m+1}} = \frac{r_m^T r_m}{p_{m+1}^T A p_{m+1}}. \quad (7.20)$$

在等式 (7.17) 两边同时左乘 $p_m^T A$ 可得

$$0 = p_m^T A p_{m+1} = p_m^T A r_m + \mu_m p_m^T A p_m.$$

于是

$$\mu_m = -\frac{p_m^T A r_m}{p_m^T A p_m} = -\frac{r_m^T A p_m}{p_m^T A p_m}. \quad (7.21)$$



为了进一步减少运算量, 我们还可以将上式进行简化. 用 $r_{m+1}^\top$ 左乘方程 (7.19) 可得

$$r_{m+1}^\top r_{m+1} = r_{m+1}^\top r_m - \xi_{m+1} r_{m+1}^\top A p_{m+1} = -\frac{r_m^\top r_m}{p_{m+1}^\top A p_{m+1}} \cdot r_{m+1}^\top A p_{m+1}.$$

于是

$$r_{m+1}^\top A p_{m+1} = -\frac{r_{m+1}^\top r_{m+1}}{r_m^\top r_m} \cdot p_{m+1}^\top A p_{m+1}.$$

所以 $r_m^\top A p_m = -\frac{r_m^\top r_m}{r_{m-1}^\top r_{m-1}} \cdot p_m^\top A p_m$, 代入 (7.21) 可得

$$\mu_m = -\frac{r_m^\top A p_m}{p_m^\top A p_m} = \frac{r_m^\top r_m}{r_{m-1}^\top r_{m-1}}. \quad (7.22)$$

综上, 由 (7.22), (7.17), (7.20) 和 (7.18), (7.19) 就构成了 CG 算法的迭代过程, 即

$$\begin{aligned} p_{m+1} &= r_m + \mu_m p_m, \quad \text{其中} \quad \mu_m = \frac{r_m^\top r_m}{r_{m-1}^\top r_{m-1}}, \\ x^{(m+1)} &= x^{(m)} + \xi_{m+1} p_{m+1}, \\ r_{m+1} &= r_m - \xi_{m+1} A p_{m+1}, \quad \text{其中} \quad \xi_{m+1} = \frac{r_m^\top r_m}{p_{m+1}^\top A p_{m+1}}. \end{aligned}$$

注意, 以上递推公式是从 $m = 1$ 开始的. 因此 $m = 0$ 时的计算公式需要另外推导.

首先, 由 $\tilde{p}_1$ 的定义可知

$$\tilde{p}_1 = \tilde{P}_1 = V_1 L_1^{-T} = v_1.$$

因此

$$p_1 = \tau_0 \tilde{p}_1 = \beta v_1 = r_0.$$

其次, 由 Lanczos 过程可知 $T_1 = \alpha_1 = v_1^\top A v_1$. 注意到 $\beta = r_0^\top r_0$, 于是

$$x^{(1)} = x^{(0)} + V_1 T_1^{-1} (\beta e_1) = x^{(0)} + \frac{\beta}{v_1^\top A v_1} v_1 = x^{(0)} + \frac{r_0^\top r_0}{p_1^\top A p_1} p_1.$$

令 $\xi_1 = \frac{r_0^\top r_0}{p_1^\top A p_1}$, 则当 $m = 0$ 时关于 ξ_{m+1} 的递推公式 (7.20) 仍然成立 (注: 之前的计算公式 (7.20) 只对 $m \geq 1$ 有定义). 从而关于 $x^{(m+1)}$ 的递推公式 (7.18) 在 $m = 0$ 时也成立.

最后考虑残量 r_1 . 易知

$$r_1 = b - A x^{(1)} = b - A x^{(0)} - \frac{r_0^\top r_0}{p_1^\top A p_1} A p_1 = r_0 - \xi_1 A p_1,$$

即当 $m = 0$ 时关于 r_{m+1} 的递推公式也成立.

于是, 我们就可以给出下面的 CG 算法.

算法 7.7 共轭梯度法 (CG)

- 1: 选取初值 $x^{(0)}$, 停机准则 $\text{tol} > 0$ 和最大迭代步数 itermax
- 2: $r_0 = b - A x^{(0)}$, $\beta = \|r_0\|_2$
- 3: **for** $m = 1$ **to** itermax **do**



```

4:    $\rho = r_{m-1}^\top r_{m-1}$ 
5:   if  $m > 1$  then
6:        $\mu_{m-1} = \rho / \rho_0$ 
7:        $p_m = r_{m-1} + \mu_{m-1} p_{m-1}$ 
8:   else
9:        $p_m = r_0$ 
10:  end if
11:   $q_m = Ap_m$ 
12:   $\xi_m = \rho / (p_m^\top q_m)$ 
13:   $x^{(m)} = x^{(m-1)} + \xi_m p_m$ 
14:   $r_m = r_{m-1} - \xi_m q_m$ 
15:   $\text{relres} = \|r_m\|_2 / \beta$ 
16:  if  $\text{relres} < \text{tol}$ , then break    % 算法收敛
17:   $\rho_0 = \rho$ 
18: end for
19: if  $\text{relres} < \text{tol}$  then
20:     输出近似解及相关信息
21: else
22:     输出算法失败信息
23: end if

```

计算复杂度

CG 算法的每个迭代步的主要运算为一个矩阵向量乘积和两个向量内积;

7.4 收敛性分析

对于一般情形, Krylov 子空间迭代法的收敛性分析是比较困难的. 我们这里主要讨论对称正定情形 (CG 方法) 和正规情形 (GMRES 方法) 的收敛性.

7.4.1 CG 的收敛性

我们估计 $\|x^{(m)} - x_*\|_A^2$ 的大小.

记 $\mathbb{P}_k$ 为所有次数不超过 k 的多项式的集合. 对任意 $x \in x^{(0)} + \mathcal{K}_m$, 存在 $p(t) \in \mathbb{P}_{m-1}$, 使得

$$x = x^{(0)} + p(A)r_0.$$

于是有

$$x - x_* = \varepsilon_0 + p(A)(b - Ax^{(0)}) = \varepsilon_0 + p(A)(Ax_* - Ax^{(0)}) = (I - Ap(A))\varepsilon_0 \triangleq q(A)\varepsilon_0,$$

其中 $\varepsilon_0 = x^{(0)} - x_*$, 多项式 $q(t) = 1 - tp(t) \in \mathbb{P}_m$ 且 $q(0) = 1$. 所以

$$\|x - x_*\|_A^2 = \varepsilon_0^\top q(A)^\top A q(A) \varepsilon_0.$$



设 $A = Q\Lambda Q^T$ 是 A 的特征值分解, 其中 Q 是正交矩阵, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 是对角阵, 且 $\lambda_i > 0$. 记 $y = [y_1, y_2, \dots, y_n]^T \triangleq Q^T \varepsilon_0$, 由于 $x^{(m)} = \arg \min_{x \in x^{(0)} + \mathcal{K}_m} \|x - x_*\|_A$, 所以有

$$\begin{aligned}
 \|x^{(m)} - x_*\|_A^2 &= \min_{x \in x^{(0)} + \mathcal{K}_m} \|x - x_*\|_A^2 \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \varepsilon_0^T q(A)^T A q(A) \varepsilon_0 \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \varepsilon_0^T Q q(\Lambda)^T \Lambda q(\Lambda) Q^T \varepsilon_0 \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} y^T q(\Lambda)^T \Lambda q(\Lambda) y \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \sum_{i=1}^n y_i^2 \lambda_i q(\lambda_i)^2 \\
 &\leq \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} \{q(\lambda_i)^2\} \sum_{i=1}^n y_i^2 \lambda_i \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} \{q(\lambda_i)^2\} y^T \Lambda y \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} \{q(\lambda_i)^2\} \varepsilon_0^T A \varepsilon_0 \\
 &= \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} \{q(\lambda_i)^2\} \|\varepsilon_0\|_A^2,
 \end{aligned}$$

即

$$\frac{\|x^{(m)} - x_*\|_A^2}{\|x^{(0)} - x_*\|_A^2} \leq \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} \{q(\lambda_i)^2\}.$$

因此, 我们有下面的结论.

引理 7.8 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $x^{(m)}$ 是 CG 算法迭代 m 步后得到的近似解, 则

$$\frac{\|x^{(m)} - x_*\|_A}{\|x^{(0)} - x_*\|_A} \leq \min_{q \in \mathbb{P}_m, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)|. \quad (7.23)$$

由于 A 的特征值通常是不知道的, 因此不等式 (7.23) 右端通常难以计算. 但在许多情况下, 我们可以通过一些近似方法估计出 A 的最大特征值 λ_1 和最小特征值 λ_n . 假设 λ_1 和 λ_n 已知, 则不等式 (7.23) 右端可以放缩为

$$\min_{q \in \mathbb{P}_m, q(0)=1} \max_{\lambda_n \leq \lambda \leq \lambda_1} |q(\lambda)|. \quad (7.24)$$

由 Chebyshev 多项式的最佳逼近性质 (定理 6.33) 可知, 最小最大问题 (7.24) 的解为

$$q_*(t) = \frac{T_m \left(\frac{2t - (\lambda_1 + \lambda_n)}{\lambda_1 - \lambda_n} \right)}{T_m \left(-\frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} \right)},$$

其中 T_m 为 m 次 Chebyshev 多项式. 由于 $T_m(t) = (-1)^m T_m(t)$, 且当 $|t| \leq 1$ 时有 $|T_m(t)| \leq 1$, 所以

$$|q_*(t)| \leq \frac{1}{T_m \left(\frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} \right)}.$$



又当 $|t| > 1$ 时

$$T_m(t) = \frac{1}{2} \left[\left(t + \sqrt{t^2 - 1} \right)^m + \left(t + \sqrt{t^2 - 1} \right)^{-m} \right] \geq \frac{1}{2} \left(t + \sqrt{t^2 - 1} \right)^m,$$

所以

$$\begin{aligned} T_m \left(\frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} \right) &\geq \frac{1}{2} \left(\frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} + \sqrt{\frac{(\lambda_1 + \lambda_n)^2}{(\lambda_1 - \lambda_n)^2} - 1} \right)^m \\ &= \frac{1}{2} \left(\frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} + \frac{2\sqrt{\lambda_1 \lambda_n}}{\lambda_1 - \lambda_n} \right)^m \\ &= \frac{1}{2} \left(\frac{\sqrt{\lambda_1} + \sqrt{\lambda_n}}{\sqrt{\lambda_1} - \sqrt{\lambda_n}} \right)^m \\ &= \frac{1}{2} \left(\frac{\sqrt{\kappa(A)} + 1}{\sqrt{\kappa(A)} - 1} \right)^m, \end{aligned}$$

其中 $\kappa(A) = \frac{\lambda_1}{\lambda_n}$ 为 A 的 (谱) 条件数. 因此我们可以得到下面的收敛性定理.

定理 7.9 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $x^{(m)}$ 是 CG 算法迭代 m 步后得到的近似解, 则

$$\frac{\|x^{(m)} - x_*\|_A}{\|x^{(0)} - x_*\|_A} \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^m. \quad (7.25)$$

由此可知, CG 算法是整体线性收敛性的, 其收敛速度与系数矩阵 A 的条件数有关, 条件数越小则收敛速度越快.

7.4.2 CG 的超线性收敛性

如果我们能够获得 A 的更多的特征值信息, 则能得到更好的误差限 [8].

定理 7.10 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, 其特征值满足

$$0 < \lambda_n \leq \cdots \leq \lambda_{n+1-i} \leq b_1 \leq \lambda_{n-i} \leq \cdots \leq \lambda_{j+1} \leq b_2 \leq \lambda_j \leq \cdots \leq \lambda_1.$$

则当 $m \geq i + j$ 时有

$$\frac{\|x^{(m)} - x_*\|_A}{\|x^{(0)} - x_*\|_A} \leq 2 \left(\frac{b-1}{b+1} \right)^{m-i-j} \max_{\lambda \in [b_1, b_2]} \left\{ \prod_{k=n+1-i}^n \left(\frac{\lambda - \lambda_k}{\lambda_k} \right) \prod_{k=1}^j \left(\frac{\lambda_k - \lambda}{\lambda_k} \right) \right\}, \quad (7.26)$$

其中

$$b = \left(\frac{b_2}{b_1} \right)^{\frac{1}{2}} \geq 1.$$

由此可知, 当 b_1 与 b_2 非常接近时, b 趋向于 1, CG 算法迭代 $i + j$ 步后, 收敛就会非常快.

推论 7.11 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, 其特征值满足

$$0 < \delta \leq \lambda_n \leq \cdots \leq \lambda_{n+1-i} \leq 1 - \varepsilon \leq \lambda_{n-i} \leq \cdots \leq \lambda_{j+1} \leq 1 + \varepsilon \leq \lambda_j \leq \cdots \leq \lambda_1.$$



则当 $m \geq i + j$ 时有

$$\frac{\|x^{(m)} - x_*\|_A}{\|x^{(0)} - x_*\|_A} \leq 2 \left(\frac{1 + \varepsilon}{\delta} \right)^i \varepsilon^{m-i-j}. \quad (7.27)$$

 该结论表明: 如果 A 的特征值聚集在 1 附近 (即除了个别特征值外, 其他都在 1 的某个小邻域内), 则 CG 具有超线性收敛性.

7.4.3 GMRES 的收敛性

设 $x^{(k)}$ 是 GMRES 在 $x^{(0)} + \mathcal{K}_k(A, r_0)$ 中找到的最佳近似解, 我们考虑 $\|b - Ax^{(k)}\|_2$ 趋向于 0 的速度.

正规矩阵情形

设 A 是正规矩阵, 即

$$A = U\Lambda U^*,$$

其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 的对角线元素 $\lambda_i \in \mathbb{C}$ 为 A 的特征值.

设 $x \in x^{(0)} + \mathcal{K}_k(A, r_0)$, 则存在多项式 $p(t) \in \mathbb{P}_{k-1}$ 使得 $x = x^{(0)} + p(A)r_0$. 于是

$$b - Ax = b - Ax^{(0)} - Ap(A)r_0 = (I - Ap(A))r_0 \triangleq q(A)r_0, \quad (7.28)$$

其中 $q(t) = 1 - tp(t) \in \mathbb{P}_k$ 满足 $q(0) = 1$. 直接计算可知

$$\|b - Ax\|_2 = \|q(A)r_0\|_2 = \|Uq(\Lambda)U^*r_0\|_2 \leq \|U\|_2\|U^*\|_2\|q(\Lambda)\|_2\|r_0\|_2 = \|q(\Lambda)\|_2\|r_0\|_2.$$

又 Λ 是对角矩阵, 所以

$$\|q(\Lambda)\|_2 = \max_{1 \leq i \leq n} |q(\lambda_i)|.$$

设 $x^{(k)}$ 是近似解, 则由 GMRES 方法的最优性可知, $x^{(k)}$ 极小化残量的 2-范数. 因此,

$$\begin{aligned} \|b - Ax^{(k)}\|_2 &= \min_{x \in x^{(0)} + \mathcal{K}_k(A, r_0)} \|b - Ax\|_2 \\ &= \min_{q \in \mathbb{P}_k, q(0)=1} \|q(A)r_0\|_2 \\ &\leq \min_{q \in \mathbb{P}_k, q(0)=1} \|q(\Lambda)\|_2 \|r_0\|_2 \\ &= \|r_0\|_2 \min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)|. \end{aligned}$$

于是, 我们有下面的结论.

定理 7.12 设 $A \in \mathbb{R}^{n \times n}$ 是正规矩阵, $x^{(k)}$ 是 GMRES 算法迭代 k 步后得到的近似解, 则

$$\frac{\|b - Ax^{(k)}\|_2}{\|r_0\|_2} \leq \min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)|. \quad (7.29)$$

需要指出的是, 上界 (7.32) 是紧致的 [51, 67].



与 CG 算法类似, 我们也可以选取一个包含 A 的所有特征值的区域 $\Omega \subset \mathbb{C}$, 然后将定理 7.12 中的上界缩放为

$$\min_{q \in \mathbb{P}_k, q(0)=1} \max_{\lambda \in \Omega} |q(\lambda)|. \quad (7.30)$$

通常 Ω 必须是连通的, 否则 (7.30) 的求解非常困难, 即使是两个不相交区间的并都没法求解 [40]. 另外, Ω 不能包含原点.

我们首先考虑一种最简单的情形, 即 Ω 是复平面上的一个圆盘, 圆心为 $C(c, 0)$, 半径 $r > 0$. 这里假定 $r < c$, 因此原点不在 Ω 内.

定义复系数多项式 $q_k(z) = \left(\frac{c-z}{c}\right)^k$, 则 $q_k(z) \in \mathbb{P}_k$ 且 $q_k(0) = 1$. 所以

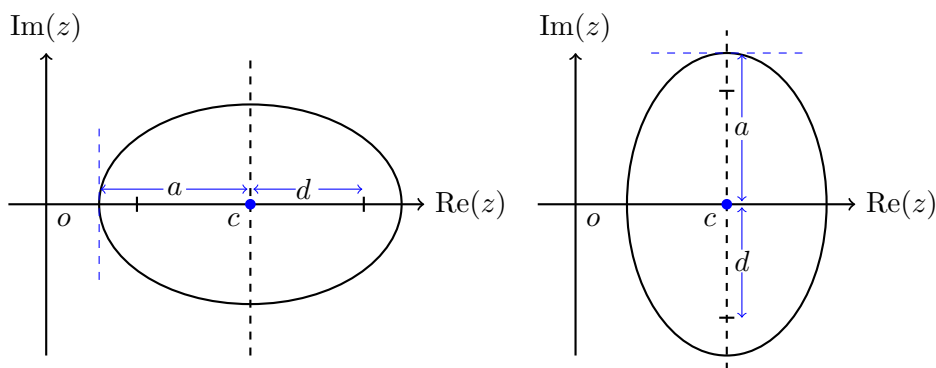
$$\min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)| \leq \max_{1 \leq i \leq n} |q_k(\lambda_i)| = \max_{1 \leq i \leq n} \left| \frac{(c - \lambda_i)^k}{c^k} \right| \leq \frac{r^k}{|c|^k}.$$

由此可见, 当 r 越小或 c 越大时, 右式趋向于 0 的速度就越快.

结论

设 A 是正规矩阵, 其特征值包含在一个圆盘内, 则圆盘的半径越小或者圆心离原点越远, 则 GMRES 收敛越快.

如果用椭圆来代替圆盘, 则可得出更紧凑的上界. 设 A 的特征值全部包含在椭圆 $E(c, d, a)$ 内, 其中 $d > 0$ 为焦距, $a > 0$ 为半长轴长, $C(c, 0)$ 为椭圆中心. 并且假定原点不在椭圆内. 如下图所示.



令 T_k 为 k 次复 Chebyshev 多项式, 则多项式

$$\tilde{T}_k(z) = \frac{T_k(\frac{c-z}{d})}{T_k(\frac{c}{d})}$$

就是极小极大问题

$$\min_{q \in \mathbb{P}_k, q(0)=1} \max_{\lambda \in E(c, d, a)} |q(\lambda)| \quad (7.31)$$

的渐进最优解. 于是

$$\min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)| \leq \min_{q \in \mathbb{P}_k, q(0)=1} \max_{\lambda \in E(c, d, a)} |q(\lambda)| \leq \max_{\lambda \in E(c, d, a)} |\tilde{T}_k(\lambda)|.$$

由复 Chebyshev 多项式的性质可知

$$\max_{\lambda \in E(c, d, a)} |\tilde{T}_k(\lambda)| = \frac{T_k(\frac{a}{d})}{|T_k(\frac{c}{d})|}.$$



因此

$$\min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)| \leq \frac{T_k(\frac{a}{d})}{|T_k(\frac{c}{d})|}.$$

所以我们可得下面的结论.

推论 7.13 设 $A \in \mathbb{R}^{n \times n}$ 是正规矩阵, $x^{(k)}$ 是由 GMRES 得到的近似解, 则

$$\frac{\|b - Ax^{(k)}\|_2}{\|r_0\|_2} \leq \frac{T_k(\frac{a}{d})}{|T_k(\frac{c}{d})|}. \quad (7.32)$$

虽然 $\tilde{T}_k(z)$ 通常不是极小极大问题 (7.31) 的解, 但上界 (7.32) 却往往能给出 GMRES 方法收敛速度的一个很好的估计 [77].

非正规情形

当 A 不是正规矩阵时, 在一般情况下, GMRES 方法的收敛性很难分析.

如果 $A \in \mathbb{R}^{n \times n}$ 是可对角化的, 即

$$A = X\Lambda X^{-1},$$

其中 $X \in \mathbb{C}^{n \times n}$ 非奇异, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 的对角线元素 $\lambda_i \in \mathbb{C}$ 为 A 的特征值, 则

$$\|b - Ax^{(k)}\|_2 = \min_{x \in x^{(0)} + \mathcal{K}_k(A, r_0)} \|b - Ax\|_2 = \min_{q \in \mathbb{P}_k, q(0)=1} \|q(A)r_0\|_2. \quad (7.33)$$

相类似地, 我们可以得到下面的结论 [94].

定理 7.14 设 $A = X\Lambda X^{-1}$, 其中 $X \in \mathbb{C}^{n \times n}$ 非奇异, Λ 是对角矩阵, $x^{(k)}$ 是 GMRES 方法得到的近似解, 则

$$\begin{aligned} \frac{\|b - Ax^{(k)}\|_2}{\|r_0\|_2} &\leq \|X\|_2 \|X^{-1}\|_2 \min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)| \\ &= \kappa(X) \min_{q \in \mathbb{P}_k, q(0)=1} \max_{1 \leq i \leq n} |q(\lambda_i)|, \end{aligned} \quad (7.34)$$

其中 $\kappa(X)$ 是 X 的谱条件数.

如果 A 接近正规, 则 $\kappa(X) \approx 1$. 此时上界 (7.34) 在一定程度上能描述 GMRES 的收敛速度. 当如果 X 远非正交, 则 $\kappa(X)$ 会很大, 此时该上界就失去了实际意义.

需要指出的是, 上面的分析并不意味着非正规矩阵就一定比正规矩阵收敛慢. 事实上, 对任意一个非正规矩阵, 总存在一个相应的正规矩阵, 使得 GMRES 方法的收敛速度是一样的 (假定初始残量相同) [5, 53, 54, 76].

虽然 GMRES 方法的收敛性与系数矩阵的特征值有关, 但显然并不仅仅取决于特征值的分布. 事实上, 我们有下面的结论.



定理 7.15 对于任意给定的特征值分布和一条不增的收敛曲线, 则总存在一个矩阵 A 和一个右端项 b , 使得 A 具有指定的特征值分布, 且 GMRES 方法的收敛曲线与给定的收敛曲线相同.

例 7.3 (见例 7.1) 考虑线性方程组 $Ax = b$ 其中

$$A = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{n-1} \end{bmatrix}, \quad b = e_1.$$

当 $a_0 \neq 0$ 时, A 非奇异. 易知, A 的特征值多项式为

$$p(x) = \lambda^n - a_{n-1}\lambda^{n-1} - a_{n-2}\lambda^{n-2} - \cdots - a_1\lambda - a_0.$$

方程组的精确解为

$$x = [-a_1/a_0, 1, 0, \dots, 0]^T.$$

以零向量为迭代初值, 则 GMRES 迭代到第 n 步时才收敛. (前 $n-1$ 步残量范数不变).

由于 a_i 可以取任意实数, 这意味着 A 可以取任意特征值, 也就是说, 无论 A 取什么样的特征值, GMRES 都必须迭代到 n 步才收敛.

如果 A 不可以对角化, 我们在分析 GMRES 方法的收敛性时, 通常会想办法用一个新的极小化问题来近似原来的极小化问题 (7.33). 当然, 这个新的极小化问题应该是比较容易求解的.

事实上, 我们有

$$\begin{aligned} \frac{\|b - Ax^{(k)}\|_2}{\|r_0\|_2} &= \frac{\min_{q \in \mathbb{P}_k, q(0)=1} \|q(A)r_0\|_2}{\|r_0\|_2} \\ &\leq \max_{\|v\|_2=1} \min_{q \in \mathbb{P}_k, q(0)=1} \|q(A)v\|_2 \end{aligned} \quad (7.35)$$

$$\leq \min_{q \in \mathbb{P}_k, q(0)=1} \|q(A)\|_2. \quad (7.36)$$

不等式 (7.35) 右端代表的是在最坏情况下的 GMRES 收敛性, 而且是紧凑的, 即它是所能找到的不依赖于 r_0 的最好上界. 但我们仍然不清楚, 到底是 A 的那些性质决定着这个上界 [37].

可以证明, 当 A 是正规矩阵时, 上界 (7.35) 和 (7.36) 是相等的 [51, 67]. 但是, 对于大多数非正规矩阵而言, 这两者是否相等或者非常接近, 迄今仍不太清楚.

最后需要指出的是, 算法的收敛性也依赖于迭代初值和右端项. 所以定理 7.9, 7.12 和 7.14 中的上界描述的都是最坏情况下的收敛速度. 也就是说, 在实际计算中, 算法的收敛速度可能会比预想的要快得多.



7.5 预处理方法

Nothing will be more central to computational science in the next century than the art of transforming a problem that appears intractable into another whose solution can be approximated rapidly. For Krylov subspace matrix iterations, this is **preconditioning**.

— Trefethen & Bau III [103, page 319], 1997.

为了改善 krylov 迭代方法的收敛性与可靠性, 我们通常需要运用预处理技术 (preconditioning, 也称为预条件). 通俗地说, 预处理就是将原来难以求解的问题转化成一个等价的但比较容易求解的新问题. 预处理技术的研究是目前科学计算领域中的重要研究课题之一.

早在 1948 年, 著名数学家 Turing (也被称为计算机科学之父和人工智能之父) 就提出了预处理这个概念, 用于改善线性方程组的性态.

The experiences of Fox, Huskey, and Wilkinson prompted Turing to write a remarkable paper “Rounding-off errors in matrix processes” [104, 1948]. In this paper, Turing made several important contributions. He formulated the LU (actually, the LDU) factorization of a matrix, He used the word “preconditioning” to mean improving the condition of a system of linear equations (a term that did not come into popular use until the 1970s).

— Nicholas J. Higham [61, page 184], 2002.

随后, Evans 在 1968 年提出了类似的想法, 用于降低线性方程组的条件数.

To obtain improved convergence rates for the methods of successive displacement we require the coefficient matrix to have a P -condition number as small as possible. If this criterion is not satisfied, then it is advisable to prepare the system or “precondition” it beforehand.

— D. J. Evans [36], 1968.

7.5.1 预处理方法介绍

对于线性方程组而言, 预处理就是对系数矩阵进行适当的线性转换, 转换为一个新矩阵.

考虑线性方程组

$$Ax = b, \quad A \in \mathbb{R}^{n \times n} \text{ 非奇异}, \quad b \in \mathbb{R}^n. \quad (7.37)$$



如果在方程组的两边同时左乘一个非奇异矩阵 $P \in \mathbb{R}^{n \times n}$ 的逆, 则可得

$$P^{-1}Ax = P^{-1}b. \quad (7.38)$$

这个新方程组就是预处理后的方程组, P 就称为**预处理子 (preconditioner)**. 当 Krylov 子空间方法被用于求解 (7.38) 时, 就称为预处理 Krylov 子空间方法.

理论上讲, 任何一个非奇异矩阵都可以作为预处理子. 但一个好的预处理子 P 通常需满足下面两个要求:

- (1) 预处理后的线性方程组更容易求解;
- (2) 预处理子 P 的构造和使用所增加的额外计算成本较低.
- (1) $P^{-1}A$ 具有更好的特征值分布及/或更小的条件数;
- (2) $Pz = r$ 容易求解.

其中第一个要求是为了确保预处理后的线性方程组更容易求解, 也就是说, 选取的预处理方法是有效的, 通常要求 $P^{-1}A$ 具有更小的条件数和/或更好的特征值分布. 对于 Krylov 子空间方法而言, 第二个要求主要是指以 P 为系数矩阵的线性方程组很容易求解.

预处理方式 (7.38) 称为**左预处理**. 相应地, 我们可以在方程组两边同时右乘 P^{-1} , 这就是**右预处理**, 即

$$AP^{-1}u = b, \quad x = P^{-1}u. \quad (7.39)$$

另外, 我们也可以将 P 分解成两个矩阵的乘积, 即 $P = LR$. 于是我们可以用下面的方式对原方程组 (7.37) 进行预处理

$$L^{-1}AR^{-1}u = b, \quad x = R^{-1}u. \quad (7.40)$$

这就是**两边预处理**.

以上是三种常用的预处理方式. 这三种方式预处理后的系数矩阵分别为 $P^{-1}A$, AP^{-1} 和 $L^{-1}AR^{-1}$. 由于它们是相似的, 所以具有相同的特征值分布. 如果 A 是对称正定的, 则使用共轭梯度法求解时, 这三种方式的预处理效果基本上是一样的. 但对于非对称 (特别是非正规) 情形, 效果可能会相差很大.

在实际使用中, 该选取哪种预处理方式, 需要根据问题本身和所用的方法来确定. 如对于对称正定线性方程组的 CG 方法, 三种方式都可以, 而对于 GMRES 方法, 则选取右预处理比较合适. 一方面是实际使用时, 得到的残量范数与原方程组的残量范数是一样的, 另一方面是, 右预处理极小化的是原始残量范数, 而左预处理极小化的是预处理后的残量.

这里需要指出的是, 在实际求解预处理后的方程组时, 我们并不会显式地计算 P^{-1} (除非 P^{-1} 非常容易计算), 更不会显式地计算 $P^{-1}A$.

7.5.2 预处理 CG 方法

设 $A \in \mathbb{R}^{n \times n}$ 对称正定, 并假定预处理子 P 也是对称正定的. 为了保证预处理后的系数矩阵仍然是对称正定的, 我们考虑使用两边预处理方式. 设 P 的 Cholesky 分解为

$$P = LL^T.$$



于是我们得到下面的预处理方程组

$$L^{-1}A(L^T)^{-1}u = L^{-1}b, \quad x = (L^T)^{-1}u. \quad (7.41)$$

用 CG 方法求解上述方程组, 迭代 k 步后, 得到的近似解记为 $u^{(k)}$, **预处理残量** 记为 $\tilde{r}_k \triangleq L^{-1}b - L^{-1}A(L^T)^{-1}u^{(k)}$. 于是, 求解预处理方程组 (7.41) 的 CG 方法可描述如下:

算法 7.8 两边预处理 CG 方法

```

1: 给定初值  $x^{(0)}$ 
2: 计算  $r_0 = b - Ax^{(0)}$ 
3: 令  $\tilde{r}_0 = L^{-1}r_0$ ,  $\tilde{p}_1 = \tilde{r}_0$ 
4: for  $k = 1, 2, \dots$  do
5:    $\xi_k = \frac{(\tilde{r}_{k-1}, \tilde{r}_{k-1})}{(L^{-1}A(L^T)^{-1}\tilde{p}_k, \tilde{p}_k)}$ 
6:    $u^{(k)} = u^{(k-1)} + \xi_k \tilde{p}_k$ 
7:    $\tilde{r}_k = \tilde{r}_{k-1} - \xi_k L^{-1}A(L^T)^{-1}\tilde{p}_k$ 
8:    $\mu_k = \frac{(\tilde{r}_k, \tilde{r}_k)}{(\tilde{r}_{k-1}, \tilde{r}_{k-1})}$ 
9:    $\tilde{p}_{k+1} = \tilde{r}_k + \mu_k \tilde{p}_k$ 
10: end for
  
```

由于算法 7.8 中得到的是预处理后的近似解 $u^{(k)}$ 和预处理残量 $\tilde{r}_k$, 因此我们还需要考虑原方程的近似解和残量的计算.

我们对上述算法中的迭代公式进行适当的改写. 首先引入一个辅助变量 p_k :

$$p_k \triangleq (L^T)^{-1}\tilde{p}_k, \quad k = 1, 2, \dots$$

于是有

$$p_{k+1} = (L^T)^{-1}p_{k+1} = (L^T)^{-1}\tilde{r}_k + \mu_k(L^T)^{-1}\tilde{p}_k = (L^T)^{-1}\tilde{r}_k + \mu_k p_k.$$

又由 $\tilde{r}_k$ 的表达式可知, 原方程组的残量 $r_k = L\tilde{r}_k$. 因此可得到下面的递推公式:

$$\begin{aligned}
r_k &= L\tilde{r}_k = L(\tilde{r}_{k-1} - \xi_k L^{-1}A(L^T)^{-1}\tilde{p}_k) = r_{k-1} - \xi_k A p_k, \\
p_{k+1} &= (L^T)^{-1}\tilde{r}_k + \mu_k p_k = (L^T)^{-1}L^{-1}r_k + \mu_k p_k = P^{-1}r_k + \mu_k p_k, \\
x^{(k)} &= (L^T)^{-1}u^{(k)} = (L^T)^{-1}(u^{(k-1)} + \xi_k \tilde{p}_k) = x^{(k-1)} + \xi_k p_k,
\end{aligned}$$

其中

$$\begin{aligned}
\xi_k &= \frac{(L^{-1}r_{k-1}, L^{-1}r_{k-1})}{(L^{-1}A(L^T)^{-1}\tilde{p}_k, \tilde{p}_k)} = \frac{(r_{k-1}, (L^T)^{-1}L^{-1}r_{k-1})}{(A(L^T)^{-1}\tilde{p}_k, (L^T)^{-1}\tilde{p}_k)} = \frac{(r_{k-1}, P^{-1}r_{k-1})}{(A p_k, p_k)}, \\
\mu_k &= \frac{(L^{-1}r_k, L^{-1}r_k)}{(L^{-1}r_{k-1}, L^{-1}r_{k-1})} = \frac{(r_k, P^{-1}r_k)}{(r_{k-1}, P^{-1}r_{k-1})}.
\end{aligned}$$

记 $z_k \triangleq P^{-1}r_k$, 则可得下面的预处理 CG 方法.

算法 7.9 PCG: 预处理 CG 方法

```

1: 给定初值  $x^{(0)}$ , (相对) 精度要求  $\text{tol} > 0$  和最大迭代步数  $\text{itermax}$ 
2: 计算  $r_0 = b - Ax^{(0)}$  和  $\beta = \|r_0\|_2$ 
  
```



```

3: 令  $z_0 = P^{-1}r_0$ ,  $p_1 = z_0$ 
4: 计算  $\rho = r_0^T z_0$ 
5: for  $k = 1$  to itermax do
6:    $q_k = Ap_k$ 
7:    $\xi_k = \rho / (p_k^T q_k)$ 
8:    $x^{(k)} = x^{(k-1)} + \xi_k p_k$ 
9:    $r_k = r_{k-1} - \xi_k q_k$ 
10:  relres =  $\|r_k\|_2 / \beta$ 
11:  if relres < tol, then break    % 算法收敛
12:   $\rho_0 = \rho$ 
13:   $z_k = P^{-1}r_k$ 
14:   $\rho = r_k^T z_k$ 
15:   $\mu_k = \rho / \rho_0$ 
16:   $p_{k+1} = z_k + \mu_k p_k$ 
17: end for
18: if relres < tol then
19:   输出近似解及相关信息
20: else
21:   输出算法失败信息
22: end if

```

我们注意到, 矩阵 L 并没有出现在算法中, 这意味着我们并不需要对 P 做 Cholesky 分解. 在算法 7.9 中, 每步迭代都需要计算 $z_k = P^{-1}r_k$. 一般情况下, 都是通过求解方程组 $Pz_k = r_k$ 来实现, 除非 P^{-1} 非常容易得到, 或者是直接给出的.

事实上, PCG 方法 7.9 也可以从左预处理方式导出. 考虑左预处理方程组

$$P^{-1}Ax = P^{-1}b. \quad (7.42)$$

易知 $P^{-1}A$ 是正定的, 但通常不是对称的, 因此我们不能直接对 (7.42) 实施 CG 方法. 此时我们需要定义一个新的内积: P -内积, 即

$$(x, y)_P \triangleq (Px, y) = (x, Py). \quad (7.43)$$

由于 P 是对称正定的, 所以这个定义是有效的. 在该内积下, 有

$$(P^{-1}Ax, y)_P = (Ax, y) = (x, Ay) = (x, P(P^{-1}Ay)) = (x, P^{-1}Ay)_P,$$

即 $P^{-1}A$ 关于 P -内积是自伴随的. 也就是说, 在 P -内积意义下, $P^{-1}A$ 是“对称”的. 这时我们就可以用 CG 方法来求解预处理方程组 (7.42), 但需要将欧拉内积改为 P -内积.

引入辅助变量 $z_k \triangleq P^{-1}r_k$, 则求解预处理方程组 (7.42) 的 CG 方法可描述如下:

算法 7.10 基于 P -内积的 CG 方法



```

1: 给定初值  $x^{(0)}$ , (相对) 精度要求  $\text{tol} > 0$  和最大迭代步数  $\text{itermax}$ 
2: 计算  $r_0 = b - Ax^{(0)}$  和  $\beta = \|r_0\|_2$ 
3: 令  $z_0 = P^{-1}r_0$ ,  $p_1 = z_0$ 
4: for  $k = 1$  to  $\text{itermax}$  do
5:    $\xi_k = \frac{(z_{k-1}, z_{k-1})_P}{(P^{-1}Ap_k, p_k)_P} = \frac{(r_{k-1}, z_{k-1})}{(Ap_k, p_k)}$ 
6:    $x^{(k)} = x^{(k-1)} + \xi_k p_k$ 
7:    $r_k = r_{k-1} - \xi_k Ap_k$ 
8:    $\text{relres} = \|r_k\|_2 / \beta$ 
9:   if  $\text{relres} < \text{tol}$ , then break   % 算法收敛
10:   $z_k = z_{k-1} - \alpha_k P^{-1}Ap_k = P^{-1}r_k$ 
11:   $\mu_k = \frac{(z_k, z_k)_P}{(z_{k-1}, z_{k-1})_P} = \frac{(r_k, z_k)}{(r_{k-1}, z_{k-1})}$ 
12:   $p_{k+1} = z_k + \mu_k p_k$ 
13: end for

```

不难看出, 算法 7.10 和算法 7.9 是完全一样的.

类似地, 我们也可以从右预处理方式来推导 PCG 方法, 具体过程留作练习.

7.5.3 预处理 GMRES 方法

对于非对称 Krylov 子空间方法, 也有三种预处理方式. 但与对称正定情形不同的是, 这三种方式并不等价, 而且有时效果会相差很大.

如果采用左预处理方式, 则原方程组转化为

$$P^{-1}Ax = P^{-1}b. \quad (7.44)$$

记 r_k 为原线性方程组的残量, 即 $r_k = b - Ax^{(k)}$, 则预处理方程组 (7.44) 的残量为 $\tilde{r}_k = P^{-1}r_k$. 因此在对应的算法中, 停机准则是针对 $\tilde{r}_k$, 而不是真实的残量 r_k . 所以我们通常采用右预处理方式.

设 P 是预处理子, 则右预处理后的方程组为

$$AP^{-1}u = b, \quad x = P^{-1}u. \quad (7.45)$$

给定迭代初值 $x^{(0)}$, 则 $u^{(0)} = Px^{(0)}$. 相应的预处理初始残量为

$$\tilde{r}_0 = b - AP^{-1}u^{(0)} = b - Ax^{(0)} = r^{(0)},$$

即预处理残量与原方程组残量是一样的. 因此无需计算 $u^{(0)}$. 设 $u^{(m)} = u^{(0)} + V_m y_m$ 是迭代 m 步后的近似解, 则对应的原方程组近似解 $x^{(m)}$ 为

$$x^{(m)} = P^{-1}u^{(m)} = P^{-1}(u^{(0)} + V_m y_m) = x^{(0)} + P^{-1}V_m y_m,$$

原方程组的残量为

$$r_m = b - Ax^{(m)} = r_0 - AP^{-1}V_m y_m = \beta v_1 - V_{m+1} H_{m+1, m} y_m = V_{m+1}(\beta e_1 - H_{m+1, m} y_m).$$



由于 y_m 是最小二乘问题

$$\min_{y \in \mathbb{R}^m} \|\beta e_1 - H_{m+1,m} y\|_2$$

的解, 因此

$$\|r_m\|_2 = \|\beta e_1 - H_{m+1,m} y_m\|_2 = \beta \cdot |q_1(m+1)|.$$

这与不带预处理的 GMRES 方法是一样的. 因此在实际求解过程中我们可以直接得到原方程组残量的模, 而无需计算 $u^{(m)}$ 和 $x^{(m)}$. 这是右预处理方式与左预处理方式的主要区别.

右预处理 GMRES 方法具体描述如下:

算法 7.11 右预处理 GMRES 方法

```

1: 给定初值  $x^{(0)}$  和 (相对) 精度要求  $\text{tol} > 0$ 
2: 计算  $r_0 = b - Ax^{(0)}$  和  $\beta = \|r_0\|_2$ 
3: 令  $v_1 = \tilde{r}_0/\beta$ ,  $\xi = \beta e_1$ 
4: for  $j = 1, 2, \dots$  do
5:    $\tilde{w}_j = P^{-1}v_j$    % Apply preconditioning
6:    $w_j = A\tilde{w}_j$ 
7:   for  $i = 1, 2, \dots, j$  do
8:      $h_{ij} = (w_j, v_i)$ 
9:      $w_j = w_j - h_{ij}v_i$ 
10:  end for
11:   $h_{j+1,j} = \|w_j\|_2$ 
12:  for  $i = 1, 2, \dots, j-1$  do   % Apply  $G_{j-1}, \dots, G_1$  to the last column of  $H_{j+1,j}$ 
13:    
$$\begin{bmatrix} h_{i,j} \\ h_{i+1,j} \end{bmatrix} = \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix} \begin{bmatrix} h_{i,j} \\ h_{i+1,j} \end{bmatrix}$$

14:  end for
15:  if  $h_{j+1,j} = 0$  then
16:    set  $m = j$  and break
17:  end if
18:   $v_{j+1} = w_j/h_{j+1,j}$ 
19:  if  $|h_{j,j}| > |h_{j+1,j}|$  then   % Form the Givens rotation  $G_j$ 
20:     $c_j = \frac{1}{\sqrt{1+\tau^2}}$ ,  $s_j = c_j\tau$  where  $\tau = \frac{h_{j+1,j}}{h_{j,j}}$ 
21:  else
22:     $s_j = \frac{1}{\sqrt{1+\tau^2}}$ ,  $c_j = s_j\tau$  where  $\tau = \frac{h_{j,j}}{h_{j+1,j}}$ 
23:  end if
24:   $h_{j,j} = c_j h_{j,j} + s_j h_{j+1,j}$ ,  $h_{j+1,j} = 0$    % Apply  $G_j$  to last column of  $H_{j+1,j}$ 
25:  
$$\begin{bmatrix} \xi_j \\ \xi_{j+1} \end{bmatrix} = \begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} \begin{bmatrix} \xi_j \\ 0 \end{bmatrix}$$
   % Apply  $G_j$  to the right-hand side
26:  if  $|\xi_{j+1}|/\beta < \text{tol}$  then   % Check convergence:  $\|r_j\|_2 = |\xi_{j+1}|$ 

```



```

27:         set  $m = j$  and break
28:     end if
29: end for
30: 计算  $y^{(m)} = R_m^{-1}\xi^{(m)}$ , 其中  $R_m = H(1:m, 1:m)$ ,  $\xi^{(m)} = \xi(1:m)$ 
31: 计算近似解  $x^{(m)} = x^{(0)} + P^{-1}V_m y^{(m)}$ 

```

关于左、右预处理 GMRES, 我们有下面的最优性质.

定理 7.16 设 $x_L^{(k)}$ 和 $x_R^{(k)}$ 分别是左预处理 GMRES 方法和右预处理 GMRES 方法迭代 k 步后得到的近似解, 且迭代初始值均为 $x^{(0)}$. 则 $x_L^{(k)}$ 是 $\|P^{-1}(b - Ax)\|_2$ 在 $x^{(0)} + \mathcal{K}_k(P^{-1}A, P^{-1}r_0)$ 的极小值点, 而 $x_R^{(k)}$ 则是 $\|b - Ax\|_2$ 在同一个子空间中的极小值点.

7.5.4 预处理子构造

This process of “preconditioning” is essential to most successful applications of iterative methods.

— Trefethen & Bau III [103, page 313], 1997.

Finding a good preconditioner to solve a given sparse linear system is often viewed as a combination of art and science. Theoretical results are rare and some methods work surprisingly well, often despite expectations.

— Saad [95], 2003.

预处理能否取得成功的关键就是能否找到一个好的预处理子. 一个公认的事实是, 好的预处理子通常是与问题本身的特征密切相关的.

 既要有很好的通用性, 又要有很好的加速效果的预处理子往往是可遇而不可求的.

关于预处理技术的理论分析很少, 大多数情况下只能根据经验来构造. 尽管如此, 在实际应用中, 这些根据经验构造出来的预处理子往往能取得很好的数值效果, 有时甚至会大大出乎人们的意料.

 预处理方法主要是用来提升 Krylov 子空间迭代法的收敛速度, 因此, 对 Krylov 子空间迭代法的收敛性质的深刻理解对构造好的预处理方法有很大的帮助.



A preconditioner M is good if $M^{-1}A$ is not too far from normal and its eigenvalues are clustered.

— Trefethen & Bau III [103, page 314], 1997.

一般来说, 预处理子可以分为两大类

- (a) 代数预处理子 (Algebraic Preconditioner), 即仅仅根据所给的矩阵构造出来的预处理子.
- (b) 专用预处理子 (Problem-Specific Preconditioner), 即根据问题的物理背景所构造的预处理子.

显然, 由于专用预处理子充分利用了问题的物理背景知识, 所以它们往往具有很好的数值表现, 如多重网格, 区域分解, 快速变换等等. 但它们严重依赖于原问题的物理背景, 因此通用性较差.

我们这里只考虑代数预处理子, 即仅仅根据所给的系数矩阵来构造预处理方法. 这种预处理方法具有较好的通用性. 这类预处理子的一般来源有:

- 定常迭代法: 设有矩阵分裂 $A = M - N$, 则 M 可作为一个预处理子, 如 Jacobi, G-S, SOR, HSS 等;
- 直接法: 如不完全 LU 分解, 不完全 Cholesky 分解等;
- 近似逆: 即选取矩阵 P , 使得 $P^{-1} \approx A^{-1}$;
- 块状结构: 如基于 Schur 补的块对角矩阵, 块三角矩阵等.

基于矩阵分裂的预处理子

考虑线性方程组

$$Ax = b, \quad A \in \mathbb{R}^{n \times n},$$

对 A 做如下的矩阵分裂:

$$A = M - N \tag{7.46}$$

其中 M 非奇异, 则可以得到下面的迭代方法

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b = x^{(k)} + \left(M^{-1}b - M^{-1}Ax^{(k)}\right). \quad k = 0, 1, 2, \dots$$

这等价于求解下面的方程组

$$M^{-1}Ax = M^{-1}b. \tag{7.47}$$

这就是与矩阵分裂 (7.46) 相对应的 **预处理线性方程组**. 将 Krylov 子空间方法用于求解方程组 (7.47), 就得到预处理 Krylov 子空间方法. 矩阵 M 就是由矩阵分裂 (7.46) 所定义的预处理子.

理论上讲, 任何一个矩阵分裂都可以定义一个预处理子. 但为了使得预处理子能有很好的预处理效果, 往往需要其在一定意义下与 A 充分接近.

设 $A = D - L - U$, 其中 D , $-L$, $-U$ 分别是 A 的对角部分, 严格下三角部分和严格上三角部分, 并假定 D 非奇异. 则由我们之前讨论的定常迭代法, 可以立即得到下面的预处理子:



- Jacobi 预处理子, 即取 A 的对角部分作为预处理子:

$$P_J = D.$$

- G-S 预处理子, 即取 A 的下三角部分作为预处理子:

$$P_{GS} = D - L.$$

- SOR 预处理子, 即

$$P_{SOR} = \frac{1}{\omega} (\omega D - L).$$

- SSOR 预处理子, 即

$$P_{SSOR} = \frac{1}{\omega(2-\omega)} [(1-\omega)D + \omega L] D^{-1} [(1-\omega)D + \omega U].$$

由于 SSOR 对参数 ω 的取值不是很敏感, 因此我们通常令 $\omega = 1$, 对应的预处理子就是对称 Gauss-Seidel (SGS) 预处理子: 即

$$P_{SGS} = (D - L)D^{-1}(D - U).$$

- HSS 预处理子, 即

$$P_{HSS} = \frac{1}{2\alpha} (\alpha I + H)(\alpha I + S),$$

其中 $H = \frac{1}{2}(A + A^T)$, $S = \frac{1}{2}(A - A^T)$.

不完全 LU 分解

对于大规模稀疏线性方程组, 有一类比较常用的预处理方法是**不完全分解**. 不完全分解首先由 Buleev [15, 15] 于上个世纪五十年代提出, 当时主要用来求解五点差分方程 (2D) 和七点差分方程 (3D) [66], 取得了非常好的效果, 但没有任何理论结果. 之后有更多学者对不完全分解进行了研究, 如 Varga (1960) [108], Oliphant (1962) [82], 等等 (参见 [66]). 但真正的突破是 1977 年 Meijerink 和 van der Vorst [79] 将不完全 Cholesky 分解用作共轭梯度法的预处理子, 提出了不完全 Cholesky 分解共轭梯度法 (ICCG). 它的出现, 才使得共轭梯度法成为求解大规模对称正定线性方程组的首选方法. 这时离共轭梯度法的提出已经过去了整整 25 年. 同时, 这也标志着 Krylov 子空间方法逐渐替代定常迭代方法, 成为求解大规模稀疏线性方程组的主流方法. 从那以后, 关于不完全矩阵分解的研究工作越来越多, 直至今天, 仍然是科学与工程计算领域的一个核心研究课题.

不完全 LU 分解的基本思想就是在对系数矩阵 A 做 LU 分解时, 要求 L 和 U 满足一定的稀疏性, 比如与 A 具有相同的稀疏模式, 即 A 在 (i, j) ($i \neq j$) 位置上的元素为 0, 则 $L(i, j)$ 和 $U(i, j)$ 也必须为 0. 于是得到的不完全 LU 分解为

$$A = LU + R,$$

其中 R 是误差矩阵. 按这种方式定义的不完全 LU 分解记为 $ILU(0)$.

为了增加 LU 与 A 的近似度, 可以适当放宽对 L 和 U 的稀疏性要求, 即允许 L 和 U 中出现额外的非零元, 这些非零元称为**填充 (fill-in)**. 但为了控制预处理子 LU 的使用成本, 需要控制填充的数量. 于是学者们提出了各种限制填充出现的规则, 比如设定填充的大小或填充的个数,



即只有大于给定阈值的填充才保留, 或者每行填充的个数不能超过一个给定的阈值, 等等. 更多关于这方面的内容可以参见 [58, 95, 114].

其他预处理子

其他常用的预处理子有

- 针对稀疏矩阵的稀疏近似逆 (Sparse Approximate Inverse Preconditioners, SPAI)
- (代数) 多重网格
- 区域分解
- 低秩逼近, 等等

更多关于预处理的介绍可以参考 [12, 95, 111].

7.6 课后习题

练习 7.1 证明引理 7.1, 即

$$\mathcal{K}_m = \text{span}\{v_1, v_2, \dots, v_m\}.$$

(提示: 用归纳法证明 $v_k \in \mathcal{K}_k$, $k = 1, 2, \dots, m$ 即可)

练习 7.2 证明定理 7.3, 即由 Lanczos 过程生成的向量是相互正交的.

练习 7.3 证明定理 7.5 中的必要性.

练习 7.4 设 $H_{m+1,m}$ 不可约, 试证明 (7.8) 中的 R_m 非奇异.

练习 7.5 设 $A \in \mathbb{R}^{n \times n}$ 对称正定, 非零向量 $p_1, p_2, \dots, p_m$ 是 A -共轭的, 即当 $i \neq j$ 时有 $p_i^T A p_j = 0$.

试证明: $p_1, p_2, \dots, p_m$ 线性无关.

练习 7.6 用右预处理方式推导 PCG 算法.

练习 7.7 设 $A \in \mathbb{R}^{n \times n}$ 可对角化且特征值都是实数, 试证明: 存在某内积 $(\cdot, \cdot)$, 使得在此内积下 A 是自伴随的.

练习 7.8 设迭代格式 $x^{(k+1)} = Gx^{(k)} + g$, 其中 $G \in \mathbb{R}^{n \times n}$ 对称且其最小特征值 λ_n 和最大特征值 λ_1 满足 $-1 < \lambda_n < \lambda_1 < 1$. 试构造该迭代格式的 Chebyshev 加速迭代格式.

练习 7.9* 设迭代格式 $x^{(k+1)} = Gx^{(k)} + g$, 其中 G 对称且 $\lambda(G) \in [\alpha, \beta]$ ($\alpha \leq \beta$).

(1) 利用外推思想构造收敛的迭代格式.

(2) 构造新迭代格式的 Chebyshev 加速方法.

..... 以下为实践题

练习 7.10 编程实现 GMRES 方法.

练习 7.11 编程实现 CG 算法.



参考文献

- [1] J. O. Aasen, On the reduction of a symmetric matrix to tridiagonal form, *BIT Numerical Mathematics*, 11 (1971), 233–242, <https://doi.org/10.1007/BF01931804>.
- [2] Å. Björck, Solving linear least squares problems by gram-schmidt orthogonalization, *BIT Numerical Mathematics*, 7 (1967), 1–21, <https://doi.org/10.1007/BF01934122>.
- [3] Å. Björck, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics, 1996, <https://doi.org/10.1137/1.9781611971484>.
- [4] E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney and D. Sorensen, *LAPACK Users' Guide*, Software, Environments, and Tools, Society for Industrial and Applied Mathematics, 1999, <https://doi.org/10.1137/1.9780898719604>.
- [5] M. Arioli, V. Pták and Z. Strakoš, Krylov sequences of maximal length and convergence of gmres, *BIT Numerical Mathematics*, 38 (1998), 636–643, <https://doi.org/10.1007/BF02510405>.
- [6] C. Ashcraft, R. G. Grimes and J. G. Lewis, Accurate symmetric indefinite linear equation solvers, *SIAM Journal on Matrix Analysis and Applications*, 20 (1998), 513–561, <https://doi.org/10.1137/S0895479896296921>.
- [7] J. L. Aurentz, T. Mach, R. Vandebril and D. S. Watkins, Fast and backward stable computation of roots of polynomials, *SIAM Journal on Matrix Analysis and Applications*, 36 (2015), 942–973, <https://doi.org/10.1137/140983434>.
- [8] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, 1994, <https://doi.org/10.1017/CB09780511624100>.
- [9] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems*, Software, Environments, and Tools, Society for Industrial and Applied Mathematics, 2000, <https://doi.org/10.1137/1.9780898719581>.
- [10] Z.-Z. Bai, G. H. Golub and M. K. Ng, Hermitian and skew-hermitian splitting methods for non-hermitian positive definite linear systems, *SIAM Journal on Matrix Analysis and Applications*, 24 (2003), 603–626, <https://doi.org/10.1137/S0895479801395458>.
- [11] V. Barwell and A. George, A comparison of algorithms for solving symmetric indefinite systems of linear equations, *ACM Transactions on Mathematical Software (TOMS)*, 2 (1976), 242–251, <https://doi.org/10.1145/355694.355697>.
- [12] M. Benzi, Preconditioning techniques for large linear systems: A survey, *Journal of Computational Physics*, 182 (2002), 418–477, <https://doi.org/10.1006/jcph.2002.7176>.
- [13] D. A. Bini, P. Boito, Y. Eidelman, L. Gemignani and I. Gohberg, A fast implicit qr eigenvalue algorithm for companion matrices, *Linear Algebra and its Applications*, 432 (2010), 2006–2031, <https://doi.org/10.1016/j.laa.2009.08.003>.
- [14] V. Britanak, P. Yip and K. Rao, *Discrete Cosine and Sine Transforms: General Properties*,



- Fast Algorithms and Integer Approximations*, Academic Press, 2007.
- [15] N. I. Buleev, A numerical method for solving two- and three-dimensional diffusion equations, *Mathematical Collection*, 51 (1960), 227–238, <https://www.mathnet.ru/links/2538e6c2bb690618f324802f30315f62/sm4813.pdf>.
- [16] J. R. Bunch, Analysis of the diagonal pivoting method, *SIAM Journal on Numerical Analysis*, 8 (1971), 656–680, <https://doi.org/10.1137/0708061>.
- [17] J. R. Bunch and B. N. Parlett, Direct methods for solving symmetric indefinite systems of linear equations, *SIAM Journal on Numerical Analysis*, 8 (1971), 639–655, <https://doi.org/10.1137/0708060>.
- [18] J. R. Bunch and L. Kaufman, Some stable methods for calculating inertia and solving symmetric linear systems, *Mathematics of Computation*, 31 (1977), 163–179, <https://www.ams.org/mcom/1977-31-137/S0025-5718-1977-0428694-0/>.
- [19] S. Chandrasekaran, M. Gu, J. Xia and J. Zhu, A fast qr algorithm for companion matrices, in *Recent Advances in Matrix and Operator Theory* (eds. J. A. Ball, Y. Eidelman, J. W. Helton, V. Olshevsky and J. Rovnyak), Birkhäuser, Basel, 2007, 111–143.
- [20] S. H. Cheng, *Symmetric Indefinite Matrices: Linear System Solvers and Modified Inertia Problems*, PhD thesis, University of Manchester, 1998.
- [21] J. J. M. Cuppen, A divide and conquer method for the symmetric tridiagonal eigenproblem, *Numerische Mathematik*, 36 (1981), 177–195, <https://doi.org/10.1007/BF01396757>.
- [22] T. A. Davis, *Direct Methods for Sparse Linear Systems*, Fundamentals of Algorithms, Society for Industrial and Applied Mathematics, 2006, <https://doi.org/10.1137/1.9780898718881>.
- [23] J. W. Demmel, *Applied Numerical Linear Algebra*, Society for Industrial and Applied Mathematics, 1997, <https://doi.org/10.1137/1.9781611971446>.
- [24] J. Dongarra, M. Gates, A. Haidar, J. Kurzak, P. Luszczek, S. Tomov and I. Yamazaki, The singular value decomposition: Anatomy of optimizing an algorithm for extreme scale, *SIAM Review*, 60 (2018), 808–865, <https://epubs.siam.org/doi/10.1137/17M117732>.
- [25] J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, Software, Environments, and Tools, Society for Industrial and Applied Mathematics, 1998, <https://doi.org/10.1137/1.9780898719611>.
- [26] J. J. Dongarra and V. Eijkhout, Numerical linear algebra algorithms and software, *Journal of Computational and Applied Mathematics*, 123 (2000), 489–514, [https://doi.org/10.1016/S0377-0427\(00\)00400-3](https://doi.org/10.1016/S0377-0427(00)00400-3).
- [27] Z. Drmač and K. Veselić, New fast and accurate Jacobi SVD algorithm. I, *SIAM Journal on Matrix Analysis and Applications*, 29 (2008), 1322–1342, <https://doi.org/10.1137/050639193>.
- [28] Z. Drmač and K. Veselić, New fast and accurate Jacobi SVD algorithm. II, *SIAM Journal on Matrix Analysis and Applications*, 29 (2008), 1343–1362, <https://doi.org/10.1137/05063920X>.



- [29] A. A. Dubrulle, Householder transformations revisited, *SIAM Journal on Matrix Analysis and Applications*, 22 (2000), 33–40, <https://doi.org/10.1137/S0895479898338561>.
- [30] I. S. Duff, A. M. Erisman and J. K. Reid, *Direct Methods for Sparse Matrices*, Oxford University Press, 2017, <https://doi.org/10.1093/acprof:oso/9780198508380.001.0001>.
- [31] I. S. Duff and J. K. Reid, The multifrontal solution of indefinite sparse symmetric linear, *ACM Transactions on Mathematical Software*, 9 (1983), 302–325, <https://doi.org/10.1145/356044.356047>.
- [32] I. S. Duff, G. H. Golub, J. K. Reid, J. A. Scott and K. Turner, The factorization of sparse symmetric indefinite matrices, *IMA Journal of Numerical Analysis*, 11 (1991), 181–204, <https://doi.org/10.1093/imanum/11.2.181>.
- [33] J. Durbin, The fitting of time-series models, *Review of the International Statistical Institute*, 28 (1960), 233–244, <https://doi.org/10.2307/1401322>.
- [34] C. Eckart and G. Young, The approximation of one matrix by another of lower rank, *Psychometrika*, 1 (1936), 211–218, <https://doi.org/10.1007/BF02288367>.
- [35] M. Engeli, Th. Ginsburg, H. Rutishauser and E. Stiefel, *Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, Birkhäuser, Basel, 1959, <https://doi.org/10.1007/978-3-0348-7224-9>.
- [36] D. J. Evans, The use of pre-conditioning in iterative methods for solving linear equations with symmetric positive definite matrices, *IMA Journal of Applied Mathematics*, 4 (1968), 295–314, <https://doi.org/10.1093/imamat/4.3.295>.
- [37] V. Faber, W. Joubert, E. Knill and T. Manteuffel, Minimal residual method stronger than polynomial preconditioning, *SIAM Journal on Matrix Analysis and Applications*, 17 (1996), 707–729, <https://epubs.siam.org/doi/10.1137/S0895479895286748>.
- [38] D. K. Faddeev, V. N. Kublanovskaya and V. N. Faddeeva, Sur les systèmes Linéaires Algébriques de Matrices Rectangulaires et Mal-Conditionnées, *Colloques Internationaux du C.N.R.S.*, 165 (1968), 161–70.
- [39] K. V. Fernando and B. N. Parlett, Accurate singular values and differential qd algorithms, *Numerische Mathematik*, 67 (1994), 191–229, <https://doi.org/10.1007/s002110050024>.
- [40] B. Fischer, *Polynomial Based Iteration Methods for Symmetric Linear Systems*, Vieweg+Teubner Verlag, Wiesbaden, 1996, <https://doi.org/10.1007/978-3-663-11108-5>.
- [41] R. W. Freund and N. M. Nachtigal, QMR: A quasi-minimal residual method for non-Hermitian linear systems, *Numerische Mathematik*, 60 (1991), 315–339, <https://doi.org/10.1007/BF01385726>.
- [42] T. Fukaya, R. Kannan, Y. Nakatsukasa, Y. Yamamoto and Y. Yanagisawa, Shifted cholesky qr for computing the qr factorization of ill-conditioned matrices, *SIAM Journal on Scientific Computing*, 42 (2020), A477–A503, <https://doi.org/10.1137/18M1218212>.
- [43] N. Gastinel, *Linear Numerical Analysis*, Kershaw Publishing Co Ltd, 1983, <https://doi.org/10.1007/978-1-4612-1108-5>.



- [//doi.org/10.1007/978-3-322-85864-1](https://doi.org/10.1007/978-3-322-85864-1).
- [44] G. H. Golub and C. Reinsch, Singular value decomposition and least squares solutions, *Numerische Mathematik*, 14 (1970), 403–420, <https://doi.org/10.1007/BF02163827>.
- [45] G. H. Golub, History of numerical linear algebra: A personal view, 2007.
- [46] G. H. Golub and W. M. Kahan, Calculating the singular values and pseudo-inverse of a matrix, *Journal of the Society for Industrial and Applied Mathematics Series B Numerical Analysis*, 2 (1965), 205–224, <https://epubs.siam.org/doi/10.1137/0702016>.
- [47] G. H. Golub and D. P. O’Leary, Some history of the conjugate gradient and lanczos algorithms: 1948–1976, *SIAM Review*, 31 (1989), 50–102, <https://epubs.siam.org/doi/10.1137/1031003>.
- [48] G. H. Golub and H. A. van der Vorst, Eigenvalue computation in the 20th century, *Journal of Computational and Applied Mathematics*, 123 (2000), 35–65, [https://doi.org/10.1016/S0377-0427\(00\)00413-1](https://doi.org/10.1016/S0377-0427(00)00413-1).
- [49] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th edition, Johns Hopkins University Press, 2013, <https://doi.org/10.56021/9781421407944>.
- [50] J. F. Grcar, Mathematicians of gaussian elimination, *Notices of the AMS*, 58 (2011), 782–792.
- [51] A. Greenbaum and L. Gurvits, Max-min properties of matrix factor norms, *SIAM Journal on Scientific Computing*, 15 (1994), 348–358, <https://doi.org/10.1137/0915024>.
- [52] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, Frontiers in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, 1997, <https://doi.org/10.1137/1.9781611970937>.
- [53] A. Greenbaum, V. Pták and Z. Strakoš, Any nonincreasing convergence curve is possible for gmres, *SIAM Journal on Matrix Analysis and Applications*, 17 (1996), 465–469, <https://doi.org/10.1137/S0895479894275030>.
- [54] A. Greenbaum and Z. Strakoš, Matrices that generate the same krylov residual spaces, in *Recent Advances in Iterative Methods* (eds. G. Golub, M. Luskin and A. Greenbaum), Springer, New York, NY, 1994, 95–118.
- [55] M. Gu and S. C. Eisenstat, A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem, *SIAM Journal on Matrix Analysis and Applications*, 15 (1994), 1266–1276, <https://doi.org/10.1137/S089547989223924X>.
- [56] M. Gu and S. C. Eisenstat, A divide-and-conquer algorithm for the bidiagonal svd, *SIAM Journal on Matrix Analysis and Applications*, 16 (1995), 79–92, <https://epubs.siam.org/doi/10.1137/S0895479892242232>.
- [57] M. Gu and S. C. Eisenstat, A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem, *SIAM Journal on Matrix Analysis and Applications*, 16 (1995), 172–191, <https://epubs.siam.org/doi/abs/10.1137/S0895479892241287>.



- [58] I. Gustafsson, A class of first order factorization methods, *BIT Numerical Mathematics*, 18 (1978), 142–156, <https://doi.org/10.1007/BF01931691>.
- [59] M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *Journal of Research of the National Bureau of Standards*, 49 (1952), 409–435, <https://doi.org/10.6028/jres.049.044>.
- [60] N. J. Higham, Stability of the diagonal pivoting method with partial pivoting, *SIAM Journal on Matrix Analysis and Applications*, 18 (1997), 52–65, <https://doi.org/10.1137/S0895479895290371>.
- [61] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, Society for Industrial and Applied Mathematics, 2002, <https://doi.org/10.1137/1.9780898718027>.
- [62] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, 1991, <https://doi.org/10.1017/CB09780511840371>.
- [63] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd edition, Cambridge University Press, 2013.
- [64] A. S. Householder, Unitary triangularization of a nonsymmetric matrix, *Journal of the ACM (JACM)*, 5 (1958), 339–342, <https://doi.org/10.1145/320941.320947>.
- [65] A. S. Householder, *The Theory of Matrices in Numerical Analysis*, Blaisdell Publishing Company, New York, 1964.
- [66] V. P. Il'in, *Iterative Incomplete Factorization Methods*, Series on Soviet and East European Mathematics, World Scientific, Singapore, 1992, <https://doi.org/10.1142/1677>.
- [67] W. Joubert, A robust gmres-based adaptive polynomial preconditioning algorithm for nonsymmetric linear systems, *SIAM Journal on Scientific Computing*, 15 (1994), 427–439, <https://epubs.siam.org/doi/10.1137/0915029>.
- [68] W. Kahan, Numerical linear algebra, *Canadian Mathematical Bulletin*, 9 (1966), 757–801, <https://doi.org/10.4153/CMB-1966-083-2>.
- [69] W. M. Kahan, *Gauss-Seidel Methods of Solving Large Systems of Linear Equations*, PhD thesis, University of Toronto, 1958.
- [70] T. Kailath and A. H. Sayed, *Fast Reliable Algorithms for Matrices with Structure*, Society for Industrial and Applied Mathematics, 1999, <https://doi.org/10.1137/1.9781611971354>.
- [71] P. A. Knight, D. Ruiz and B. Uçar, A symmetry preserving algorithm for matrix scaling, *SIAM Journal on Matrix Analysis and Applications*, 35 (2014), 931–955, <https://doi.org/10.1137/110825753>.
- [72] D. Kressner, *Numerical Methods for General and Structured Eigenvalue Problems*, vol. 46 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg, 2005, <https://doi.org/10.1007/3-540-28502-4>.
- [73] C. Lanczos, Solution of systems of linear equations by minimized iterations, *Journal of*



- Research of the National Bureau of Standards*, 49 (1952), 33–53, <https://doi.org/10.6028/jres.049.006>.
- [74] N. Levinson, The wiener (root mean square) error criterion in filter design and prediction, *Journal of Mathematics and Physics*, 25 (1946), 261–278, <https://doi.org/10.1002/sapm1946251261>.
- [75] S. Li, M. Gu and B. N. Parlett, An improved dqds algorithm, *SIAM Journal on Scientific Computing*, 36 (2014), C290–C308, <https://doi.org/10.1137/120881087>.
- [76] J. Liesen, Computable convergence bounds for GMRES, *SIAM Journal on Matrix Analysis and Applications*, 21 (2000), 882–903, <https://epubs.siam.org/doi/10.1137/S0895479898341669>.
- [77] J. Liesen and P. Tichý, Behavior of cg and minres for symmetric tridiagonal toeplitz matrices, 2004, <https://www3.math.tu-berlin.de/preprints/files/2004LiTi.pdf>.
- [78] J. W. H. Liu, A partial pivoting strategy for sparse symmetric matrix decomposition, *ACM Transactions on Mathematical Software (TOMS)*, 13 (1987), 173–182, <https://dl.acm.org/doi/10.1145/328512.328525>.
- [79] J. A. Meijerink and H. A. Van Der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M -matrix, *Mathematics of Computation*, 31 (1977), 148–162, <https://doi.org/10.1090/S0025-5718-1977-0438681-4>.
- [80] E. H. Moore, On the reciprocal of the general algebraic matrix, *Bulletin of the American Mathematical Society*, 26 (1920), 394–395.
- [81] L. Neal and G. Poole, A geometric analysis of gaussian elimination. ii, *Linear Algebra and its Applications*, 173 (1992), 239–264, [https://doi.org/10.1016/0024-3795\(92\)90432-A](https://doi.org/10.1016/0024-3795(92)90432-A).
- [82] T. A. Oliphant, An extrapolation procedure for solving linear systems, *Quarterly of Applied Mathematics*, 20 (1962), 257–265, <https://doi.org/10.1090/qam/143350>.
- [83] C. C. Paige and M. A. Saunders, Solution of sparse indefinite systems of linear equations, *SIAM Journal on Numerical Analysis*, 12 (1975), 617–629, <https://epubs.siam.org/doi/10.1137/0712047>.
- [84] C. C. Paige, M. Rozložník and Z. Strakos, Modified gram-schmidt (mgs), least squares, and backward stability of mgs-gmres, *SIAM Journal on Matrix Analysis and Applications*, 28 (2006), 264–284, <https://epubs.siam.org/doi/10.1137/050630416>.
- [85] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, 1998, <https://doi.org/10.1137/1.9781611971163>.
- [86] B. N. Parlett and O. A. Marques, An implementation of the dqds algorithm (positive case), *Linear Algebra and its Applications*, 309 (2000), 217–259, [https://doi.org/10.1016/S0024-3795\(00\)00010-0](https://doi.org/10.1016/S0024-3795(00)00010-0).
- [87] D. W. Peaceman and H. H. Rachford Jr., The numerical solution of parabolic and elliptic differential equations, *Journal of the Society for Industrial and Applied Mathematics*, 3 (1955), 28–41, <https://epubs.siam.org/doi/abs/10.1137/0103003>.



- [88] R. Penrose, A generalized inverse for matrices, *Mathematical Proceedings of the Cambridge Philosophical Society*, 51 (1955), 406–413, <https://doi.org/10.1017/S0305004100030401>.
- [89] A. D. Polyanin and A. V. Manzhirov, *Handbook of Mathematics for Engineers and Scientists*, Chapman and Hall/CRC, New York, 2006, <https://doi.org/10.1201/9781420010510>.
- [90] J. K. Reid, On the method of conjugate gradients for the solution of large sparse systems of linear equations, in *Large Sparse Sets of Linear Equations*, Academic Press, New York, 1971, 283.
- [91] M. Rozložník, G. Shklarski and S. Toledo, Partitioned triangular tridiagonalization, *ACM Transactions on Mathematical Software (TOMS)*, 37 (2011), 38:1–38:16, <https://doi.org/10.1145/1916461.1916462>.
- [92] J. D. Rutter, A Serial Implementation of Cuppen’s Divide and Conquer Algorithm for the Symmetric Eigenvalue Problem, 1994, <https://digicoll.lib.berkeley.edu/record/137596>.
- [93] S. J. Leon, Å. Björck and W. Gander, Gram-schmidt orthogonalization: 100 years and more, *Numerical Linear Algebra with Applications*, 20 (2013), 492–532, <https://doi.org/10.1002/nla.1839>.
- [94] Y. Saad and M. H. Schultz, Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM Journal on Scientific and Statistical Computing*, 7 (1986), 856–869, <https://doi.org/10.1137/0907058>.
- [95] Y. Saad, *Iterative Methods For Sparse Linear Systems*, 2nd edition, Other Titles in Applied Mathematics, SIAM, Philadelphia, PA, 2003, <https://doi.org/10.1137/1.9780898718003>.
- [96] Y. Saad, Iterative methods for linear systems of equations: A brief historical journey, 2019, <http://arxiv.org/abs/1908.01083>.
- [97] Y. Saad and H. A. van der Vorst, Iterative solution of linear systems in the 20th century, *Journal of Computational and Applied Mathematics*, 123 (2000), 1–33, <https://www.sciencedirect.com/science/article/pii/S037704270000412X>.
- [98] J. A. Scott, Scaling and pivoting in an out-of-core sparse direct solver, *ACM Transactions on Mathematical Software*, 37 (2010), 19:1–19:23, <https://doi.org/10.1145/1731022.1731029>.
- [99] G. W. Stewart, *Matrix Algorithms: Volume 1: Basic Decompositions*, Society for Industrial and Applied Mathematics, 1998, <https://doi.org/10.1137/1.9781611971408>.
- [100] G. W. Stewart, *Matrix Algorithms: Volume II: Eigensystems*, Society for Industrial and Applied Mathematics, 2001, <https://doi.org/10.1137/1.9780898718058>.
- [101] G. W. Stewart and J.-g. Sun, *Matrix Perturbation Theory*, Academic Press, New York, 1990.
- [102] L. N. Trefethen, Numerical analysis, in *The Princeton Companion to Mathematics*, Princeton University Press, Edited by T. Gowers, J. Barrow-Green and I. Leader, 2008.
- [103] L. N. Trefethen and III. Bau David, *Numerical Linear Algebra*, Other Titles in Applied



- Mathematics, Society for Industrial and Applied Mathematics, 1997, <https://epubs.siam.org/doi/book/10.1137/1.9780898719574>.
- [104] A. Turing, Rounding-off errors in matrix processes, *The Quarterly Journal of Mechanics and Applied Mathematics*, 1 (1948), 287–308, https://www.academia.edu/46952338/Rounding_Off_Errors_in_Matrix_Processes.
- [105] H. W. Turnbull and A. C. Aitken, *An Introduction to the Theory of Canonical Matrices*, Blackie and Son, London, Glasgow, 1932, <https://scispace.com/papers/an-introduction-to-the-theory-of-canonical-matrices-haabykvuzh>.
- [106] M. Van Barel, R. Vandebril and P. Van Dooren, Computing the eigenvalues of a companion matrix, 2008, <http://bezout.dm.unipi.it/cortona08/slides/VanBarel.pdf>.
- [107] H. A. van der Vorst, Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, *SIAM Journal on Scientific and Statistical Computing*, 13 (1992), 631–644, <https://doi.org/10.1137/0913035>.
- [108] R. S. Varga, Factorizations and normalized iterative methods, in *Boundary Problems in Differential Equations*, University Wisconsin Press, Madison, 1960, 121–172, <https://www.osti.gov/biblio/4231859>.
- [109] R. S. Varga, *Matrix Iterative Analysis*, vol. 27 of Springer Series in Computational Mathematics, 2nd edition, Springer, Berlin, Heidelberg, 2000, <https://link.springer.com/10.1007/978-3-642-05156-2>.
- [110] J. Von Neumann and H. H. Goldstine, Numerical inverting of matrices of high order, *Bulletin of the American Mathematical Society*, 53 (1947), 1021–1099, <https://doi.org/10.1090/S0002-9904-1947-08909-6>.
- [111] A. J. Wathen, Preconditioning, *Acta Numerica*, 24 (2015), 329–376, <https://doi.org/10.1017/S0962492915000021>.
- [112] D. S. Watkins and L. Elsner, Convergence of algorithms of decomposition type for the eigenvalue problem, *Linear Algebra and its Applications*, 143 (1991), 19–47, [https://doi.org/10.1016/0024-3795\(91\)90004-G](https://doi.org/10.1016/0024-3795(91)90004-G).
- [113] D. S. Watkins, *The Matrix Eigenvalue Problem*, Society for Industrial and Applied Mathematics, 2007, <https://doi.org/10.1137/1.9780898717808>.
- [114] J. W. Watts, A conjugate gradient-truncated direct method for the iterative solution of the reservoir simulation pressure equation, *Society of Petroleum Engineers Journal*, 21 (1981), 345–353, <https://doi.org/10.2118/8252-PA>.
- [115] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Numerical Mathematics and Scientific Computation, Clarendon Press, Oxford University, Oxford, New York, 1965.
- [116] Y. Yamamoto, Y. Nakatsukasa, Y. Yanagisawa and T. Fukaya, Roundoff error analysis of the CholeskyQR2 algorithm, *Electronic Transactions on Numerical Analysis*, 44 (2015), 306–326.



- [117] D. M. Young, *Iterative Methods for Solving Partial Difference Equations of Elliptic Type*, PhD thesis, Harvard University, Cambridge, MA, USA, 1950.
- [118] D. M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971, <https://doi.org/10.1016/C2013-0-11733-3>.
- [119] 北京大学数学系, 高等代数 (第三版), 高等教育出版社, 2003.
- [120] 孙继广, 矩阵扰动分析, 科学出版社, 北京, 2001.
- [121] 张恭庆, 林源渠, 泛函分析讲义 (上册), 北京大学出版社, 北京, 1987.
- [122] 徐树方, 矩阵计算的理论与方法, 北京大学出版社, 北京, 1995.
- [123] 徐树方, 钱江, 矩阵计算六讲, 高等教育出版社, 北京, 2011.
- [124] 戴华, 矩阵论, 科学出版社, 2001.
- [125] 石钟慈, 第三种科学方法 — 计算机时代的科学计算, 清华大学出版社, 暨南大学出版社, 北京, 广州, 2000.
- [126] 胡家赣, 线性代数方程组的迭代解法, 科学出版社, 北京, 1991.
- [127] 蒋尔雄, 矩阵计算, 科学出版社, 北京, 2008.
- [128] 詹兴致, 矩阵论, 高等教育出版社, 北京, 2008.
- [129] 陈志明, 科学计算: 科技创新的第三种方法, 中国科学院院刊, 27 (2012), 161–166, <https://doi.org/10.3969/j.issn.1000-3045.2012.02.004>.
- [130] 魏木生, 李莹, 赵建立, 广义最小二乘问题的理论与计算, 第二版, 科学出版社, 北京, 2020.

