Lecture 1 - Linear Solver

Xian-Liang Hu

School of Mathematical Sciences, Zhejiang University, CHINA.

http://www.mathweb.zju.edu.cn:8080/xlhu/sc.html

1. Introduction

2. Linear Solver

3. Sparse Matrix

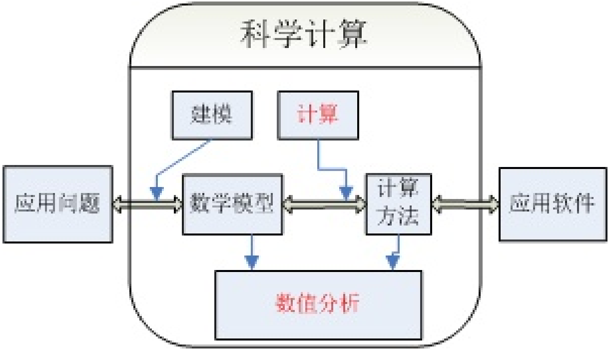
1. Introduction

2. Linear Solver

3. Sparse Matrix

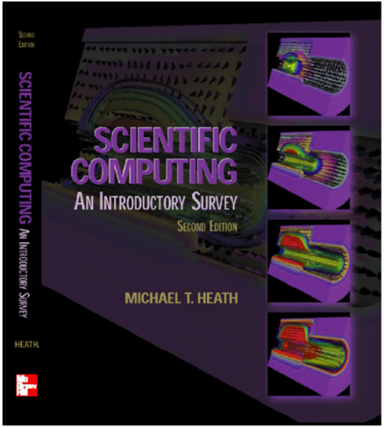
关于科学计算...

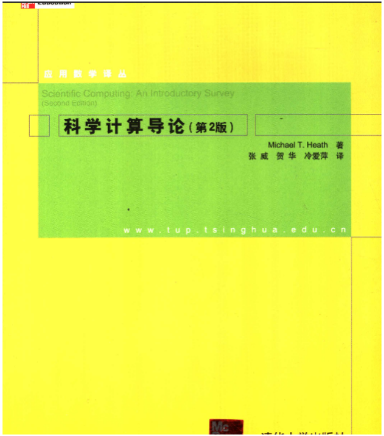
现代科学技术三种研究方法：理论推导、科学实验和科学计算



参考: 第三种科学方法: 计算机时代的科学计算

参考教材


SCIENTIFIC COMPUTING
AN INTRODUCTORY SURVEY
SECOND EDITION
MICHAEL T. HEATH


应用数学译丛
Scientific Computing: An Introductory Survey
(Second Edition)
科学计算导论(第2版)
Michael T. Heath 著
张威 贺华 冷爱萍 译
www.tup.tsinghua.edu.cn

课程纲要

第x周	上课日期	课程内容安排
第一讲	5月7日	1.数值分析与课程介绍 - 2.线性方程组
第二讲	5月14日	3.线性最小二乘问题 - 4.特征值问题
第三讲	5月21日	5.方程求根 - 6.数值最优化 - 7.插值与逼近
第四讲	5月28日	8.数值积分 - 9.初值问题 - 10.边值问题
第五讲	6月4日	11.偏微分方程数值分析 - 椭圆型 - 五点差分m文件 - 抛物型 - 双曲型
第六讲	6月11日	12.有限元方法 - 13.几类应用问题简介
七	6月18日	考试（待定）

科学计算能解决的问题(Ref - Google & YouTube)

"计算xx 学"，包括

- 物理：结构力学,流体力学,波,热传导,电磁场,"东方超环"
- 化学(材料学)：电子结构,纳米结构
- 医学：(非损伤)三维成像
- 生物：核酸序列分析
- ...

从抽象的角度来讲，归结为

- 揭示用物质实验手段尚不能表现的科学奥秘和科学规律
- 工程科学家的研究成果——理论、方法和科学数据的总结

科学计算能解决的问题(Ref - Google & YouTube)

"计算xx 学"，包括

- 物理: 结构力学,流体力学,波,热传导,电磁场,"东方超环"
- 化学(材料学): 电子结构,纳米结构
- 医学: (非损伤)三维成像
- 生物: 核酸序列分析
- ...

从抽象的角度来讲，归结为

- 揭示用物质实验手段尚不能表现的科学奥秘和科学规律
- 工程科学家的研究成果——理论、方法和科学数据的总结

数值分析/数值计算

数值分析是科学计算的核心内容，基本课程包括：

- Numerical Linear Algebra(数值线性代数)
- Numerical Analysis/Approximation(数值分析/逼近)
- Numerical Solutions to PDE(微分方程数值解)

应用领域的实际要求：

- Multi-Physics - 表现为有多个控制方程或模型
- Multi-Scale - 真实解在不同尺度下有不同的现象
- Random - 须考虑不确定因素、数据缺失
- BigData - 海量数据
- ...

数值分析/数值计算

数值分析是科学计算的核心内容，基本课程包括：

- Numerical Linear Algebra(数值线性代数)
- Numerical Analysis/Approximation(数值分析/逼近)
- Numerical Solutions to PDE(微分方程数值解)

应用领域的实际要求：

- Multi-Physics - 表现为有多个控制方程或模型
- Multi-Scale - 真实解在不同尺度下有不同的现象
- Random - 须考虑不确定因素、数据缺失
- BigData - 海量数据
- ...

数值分析的局限性

利用数值分析做科学研究有很多优点，但也有局限性：

- 数学模型描述能力的有限性：模型误差
- 数值结果是离散的，需考虑收敛性：截断误差
- 数值算法的稳定性：舍入误差的传播控制

此外，计算规模依赖于计算机硬件的发展。

一般策略：Nearby Problem

1. 寻找与复杂问题同解或“相近”的问题，如：
   - 有限维代替无限维空间(有限和代替无穷级数)
   - 非线性问题的线性化
   - 低阶替代高阶(方程组转化、降维)
   - 用简单对象(多项式、正定矩阵)实现复杂数学运算
2. 针对nearby problem构造数值方法
3. 考虑前面两步的：

适定性、相容性、收敛性、稳定性

一般策略：Nearby Problem

1. 寻找与复杂问题同解或“相近”的问题，如：
   - 有限维代替无限维空间(有限和代替无穷级数)
   - 非线性问题的线性化
   - 低阶替代高阶(方程组转化、降维)
   - 用简单对象(多项式、正定矩阵)实现复杂数学运算
2. 针于nearby problem构造数值方法
3. 考虑前面两步的：

适定性、相容性、收敛性、稳定性

一般策略：Nearby Problem

1. 寻找与复杂问题同解或“相近”的问题，如：
   - 有限维代替无限维空间(有限和代替无穷级数)
   - 非线性问题的线性化
   - 低阶替代高阶(方程组转化、降维)
   - 用简单对象(多项式、正定矩阵)实现复杂数学运算
2. 针对nearby problem构造数值方法
3. 考虑前面两步的：

适定性、相容性、收敛性、稳定性

适定性

已知$F(x,d)=0$，其中d为给定参数，x为未知数，则：

- 正问题：F,d给定，求x;
- 反问题：x,F给定，求d；或者x,d给定，求F。

所谓适定性，就是研究问题是否存在连续依赖于d的解x?

Example

求方程$p(x)=x^4-x^2(2\alpha-1)+\alpha(\alpha-1)$的实根。

适定性

已知$F(x,d)=0$，其中d为给定参数，x为未知数，则：

- 正问题：F,d给定，求x;
- 反问题：x,F给定，求d；或者x,d给定，求F。

所谓适定性，就是研究问题是否存在连续依赖于d的解x?

Example

求方程$p(x)=x^4-x^2(2\alpha-1)+\alpha(\alpha-1)$的实根。

条件数

即研究扰动问题

$$F(x+\delta_x, d+\delta_d)=0.$$

若$\exists \eta_0=\eta_0(d)>0$以及$k_0=k_0(\eta_0)>0$，使得

$$\|\delta_x\| \le k_0\|\delta_d\|,$$

$\forall \|\delta_d\| \le \eta_0$，即有$\|x-x_n\| \le k_0\|d-d_n\|$。则该问题的条件数为

$$K_{abs}(d)=\sup_{\delta_d \in D}\frac{\|\delta_x\|}{\|\delta_d\|}$$

Example

$Ax=b$，其中A非奇异，则可以得到条件数$K(d)=\|A\|\|A^{-1}\|$

条件数

即研究扰动问题

$$F(x+\delta_x, d+\delta_d) = 0.$$

若$\exists \eta_0 = \eta_0(d) > 0$以及$k_0 = k_0(\eta_0) > 0$，使得

$$\|\delta_x\| \leq k_0 \|\delta_d\|,$$

$\forall \|\delta_d\| \leq \eta_0$，即有$\|x - x_n\| \leq k_0 \|d - d_n\|$。则该问题的条件数为

$$K_{abs}(d) = \sup_{\delta_d \in D} \frac{\|\delta_x\|}{\|\delta_d\|}$$

Example

$Ax = b$，其中A非奇异，则可以得到条件数$K(d) = \|A\| \|A^{-1}\|$

相容性

考虑近似求解方法:

$$F_n(x_n, d_n) = 0, n \geq 1.$$

则希望使得在$n \to \infty$时，$x_n \to x$，即能够收敛于精确解。但只要$F_n \to F, d_n \to d$，即

$$\|F_n(x, d) - F(x, d)\| \leq \varepsilon.$$

该条件相比较而言更弱，因此称为相容性条件。

Example

求$f(x) = 0$根相容于$F_n(x_n, x_{n-1}; f) = x_n - x_{n-1} + \frac{f(x_{n-1})}{f'(x_{n-1})} = 0$,故

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

因此，讨论$F_n(*, *; f)$的一般形式具有普遍意义。

相容性

考虑近似求解方法:

$$F_n(x_n, d_n) = 0, n \geq 1.$$

则希望使得在$n \to \infty$时，$x_n \to x$，即能够收敛于精确解。但只要$F_n \to F, d_n \to d$，即

$$\|F_n(x, d) - F(x, d)\| \leq \varepsilon.$$

该条件相比较而言更弱，因此称为相容性条件。

Example

求$f(x) = 0$根相容于$F_n(x_n, x_{n-1}; f) = x_n - x_{n-1} + \frac{f(x_{n-1})}{f'(x_{n-1})} = 0$,故

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

因此，讨论$F_n(*, *; f)$的一般形式具有普遍意义。

收敛性-举例

Theorem

令$f \in C[a,b], \epsilon > 0$。则存在多项式$p(x)$使$\|f-p\|_\infty \leq \varepsilon$。

即证明存在一系列多项式$p_n(x)$，使$\|f-p_n\|_{C^m[a,b]} \overset{n\to\infty}{\to} 0$.

Proof.

$\|f-p_n\|_{C^m[a,b]} = \max\limits_{0\leq j\leq m} \|f^{(j)}\|_\infty$，其中$f \in C^m[a,b]$。

由$f \in C^m[a,b]$，则对于$0 \leq j \leq m, f^{(j)} \in C[a,b]$

则由上述定理可知，存在多项式$p_{n,j}$，使得$\|f^{(j)} - p_{n,j}\| \leq \varepsilon$

因此，$\max\limits_{0\leq j\leq m} \|f^{(j)} - p_{n,j}\| \leq \varepsilon$，即存在一个多项式序列，使得$\|f-p_n\|_{C^m[a,b]} \to 0$，当$n \to \infty$。

$\square$

稳定性I

- 适定性+ 数值方法收敛⇒ 稳定性

Proof.

$$
\begin{aligned}
\|\delta_{x_n}\| =& \|x_n(d+\delta_{d_n}) - x_n(d)\| \\
=& \|x_n(d+\delta_{d_n}) - x(d+\delta_{d_n})\| + \|x(d+\delta_{d_n}) - x(d)\| \\
& + \|x(d) - x_n(d)\|
\end{aligned}
$$

则由收敛性可知

$$\|\delta_{x_n}\| \le \varepsilon/2 + k_0\|\delta_{d_n}\| \to 0$$

从而可知解是稳定的。 □

- 对适定问题而言，相容性条件+ 数值方法稳定性
  $\Rightarrow$ 收敛性+ 解的稳定性

Proof.

$$
\begin{aligned}
\|x(d+\delta_{d_n})-x_n(d+\delta_{d_n})\| &\le \|x(d+\delta_{d_n})-x(d)\| \\
&\quad +\|x(d)-x_n(d)\|
\end{aligned}
$$

由适定性和相容性可知$\|x(d+\delta_{d_n})-x_n(d+\delta_{d_n})\| \to 0$。
即数值方法是收敛的，从而可知解是稳定的。 $\square$

计算数学经典文献13篇
By L. N. Trefethen

V.S. 计算机科学界评选的二十世纪最伟大的10大算法

- Cooley & Tukey (1965) the Fast Fourier Transform
- Courant, Friedrichs & Lewy (1928) finite difference methods for PDE
- Householder (1958) QR factorization of matrices
- Curtiss & Hirschfelder (1952) stiffness of ODEs; BD formulas
- de Boor (1972) calculations with B-splines
- Courant (1943) finite element methods for PDE
- Golub & Kahan (1965) the singular value decomposition
- Brandt (1977) multigrid algorithms

- Hestenes & Stiefel (1952) the conjugate gradient iteration
- Fletcher & Powell (1963)optimization via quasi-Newton updates
- Wanner, Hairer & Norsett (1978) order stars and applications to ODE
- Karmarkar (1984) interior pt. methods for linear prog
- Greengard & Rokhlin (1987) multipole methods for particles

Trefethen's Remark:

We were struck by how young many of the authors were when they wrote these papers (averageage: 34), and by how short an influential paper can be (Householder: 3.3 pages, Cooley & Tukey: 4.4)

冯康科学计算奖(FengKang Prize)

设立于1994年9月,为纪念冯康先生对中国计算数学事业所做的杰出贡献。旨在奖励在科学计算领域作出突出贡献的**45岁**及以下海内外**中国**青年科学家，每两年颁发一次。更多信息请访问官方网站

http://lsec.cc.ac.cn/fengkangprize/index.html

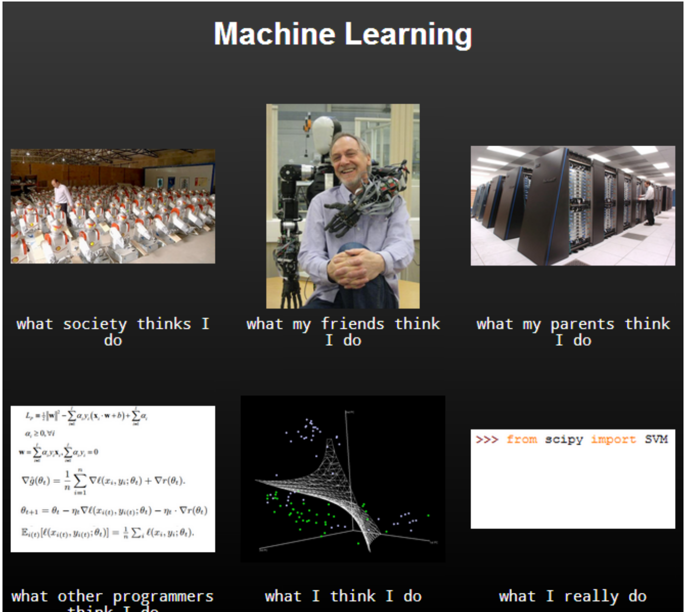
科学计算研究新趋势

1. 与具体应用结合形成新的交叉学科,比如计算流体力学,计算空气动力学,计算物理,计算化学,计算生物学等
2. 科学计算强调为新的学科发展作出贡献,被认为是除科学实验和理论分析之外的第三种研究手段
3. 材料和生物学中的计算问题是研究热点,纳米或原子尺度的多物里过程建模和多尺度计算方法是主要研究手段
4. Data-driven machine learning algorithm(数据驱动的机器学习)

科学计算研究新趋势

1. 与具体应用结合形成新的交叉学科,比如计算流体力学,计算空气动力学,计算物理,计算化学,计算生物学等
2. 科学计算强调为新的学科发展作出贡献,被认为是除科学实验和理论分析之外的第三种研究手段
3. 材料和生物学中的计算问题是研究热点,纳米或原子尺度的多物里过程建模和多尺度计算方法是主要研究手段
4. Data-driven machine learning algorithm(数据驱动的机器学习)

However ... , eh, before that,



1. Introduction

2. Linear Solver

3. Sparse Matrix

问题：线性代数方程$Ax = b$

大多数科学计算应用经过建模和数值离散之后，都可归结为求解

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \vdots &= \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{cases}$$

Crame法则可容易地计算低阶问题. 可能的来源:

- 数学模型的数值离散
- 数据处理算法
- 其他数学建模过程...

问题：线性代数方程$Ax = b$

大多数科学计算应用经过建模和数值离散之后，都可归结为求解

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \vdots &= \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{cases}$$

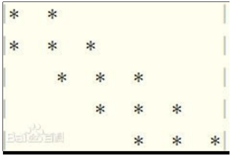
Crame法则可容易地计算低阶问题. 可能的来源:

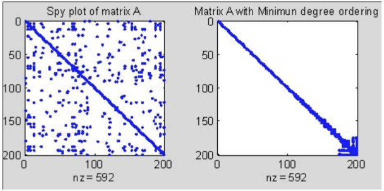
- 数学模型的数值离散
- 数据处理算法
- 其他数学建模过程...

系数矩阵A

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

- (共轭)转置,正交(酉)矩阵,Jordan分解,分块矩阵,特征值(谱)
- 稀疏性(Sparsity)





矩阵的几个度量：谱半径/范数/条件数

- 谱半径: $\rho(A) = \max|\lambda(A)|$，即绝对值最大的特征值
- 矩阵范数,如:
  - $$\|A\|_1 = \max_{1\le j\le n}\sum_{i=1}^{m}|a_{ij}|$$
    $$\|A\|_\infty = \max_{1\le i\le n}\sum_{j=1}^{m}|a_{ij}|$$
  - 谱范数$\|A\|_2 = \sqrt{\lambda_{max}(A^TA)}$
  - 任何矩阵范数满足$\|A\| \ge \rho(A)$
- 条件数: $\kappa(A) = \|A\|\cdot\|A^{-1}\|$
  - 若取L_2范数$\|\cdot\|_2$，则$\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$,最大和最小奇异值
  - 若A是正规矩阵($A^TA = AA^T$),则$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$
  - 若A是酉矩阵($A^TA = AA^T = I$),则$\kappa(A) = 1$

矩阵的几个度量：谱半径/范数/条件数

- 谱半径: $\rho(A) = \max|\lambda(A)|$，即绝对值最大的特征值
- 矩阵范数,如:
  - $$\|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^{m} |a_{ij}|$$
    $$\|A\|_\infty = \max_{1\le i\le n} \sum_{j=1}^{m} |a_{ij}|$$
  - 谱范数$\|A\|_2 = \sqrt{\lambda_{max}(A^T A)}$
  - 任何矩阵范数满足$\|A\| \ge \rho(A)$
- 条件数: $\kappa(A) = \|A\| \cdot \|A^{-1}\|$
  - 若取L_2范数$\|\cdot\|_2$，则$\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$,最大和最小奇异值
  - 若A是正规矩阵$(A^T A = AA^T)$,则$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$
  - 若A是酉矩阵$(A^T A = AA^T = I)$,则$\kappa(A) = 1$

矩阵的几个度量：谱半径/范数/条件数

- 谱半径: $\rho(A) = \max|\lambda(A)|$，即绝对值最大的特征值
- 矩阵范数,如:
  - $\|A\|_1 = \max\limits_{1\le j\le n}\sum\limits_{i=1}^{m}|a_{ij}|$

    $\|A\|_\infty = \max\limits_{1\le i\le n}\sum\limits_{j=1}^{m}|a_{ij}|$
  - 谱范数$\|A\|_2 = \sqrt{\lambda_{max}(A^TA)}$
  - 任何矩阵范数满足$\|A\| \ge \rho(A)$
- 条件数: $\kappa(A) = \|A\| \cdot \|A^{-1}\|$
  - 若取L_2范数$\|\cdot\|_2$，则$\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$,最大和最小奇异值
  - 若A是正规矩阵($A^TA = AA^T$),则$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$
  - 若A是酉矩阵($A^TA = AA^T = I$),则$\kappa(A) = 1$

常用求解方法

1. 直接求解方法：Gauss消元法
2. 经典迭代法,如Jacobi, Seidel, SOR等
3. 共轭梯度Conjugate Gradient法及其变形(A对称适用)
4. 极小化残量GMRES法及其变形(A非对称适用)
5. 其他Krylov子空间迭代法
6. 实用技术，如Precondition, Multigrid 等

常用求解方法

1. 直接求解方法：Gauss消元法
2. 经典迭代法,如Jacobi, Seidel, SOR等
3. 共轭梯度Conjugate Gradient法及其变形(A对称适用)
4. 极小化残量GMRES法及其变形(A非对称适用)
5. 其他Krylov子空间迭代法
6. 实用技术，如Precondition, Multigrid 等

常用求解方法

1. 直接求解方法：Gauss消元法
2. 经典迭代法,如Jacobi, Seidel, SOR等
3. 共轭梯度Conjugate Gradient法及其变形(A对称适用)
4. 极小化残量GMRES法及其变形(A非对称适用)
5. 其他Krylov子空间迭代法
6. 实用技术，如Precondition, Multigrid 等

常用求解方法

1. 直接求解方法：Gauss消元法
2. 经典迭代法,如Jacobi, Seidel, SOR等
3. 共轭梯度Conjugate Gradient法及其变形(A对称适用)
4. 极小化残量GMRES法及其变形(A非对称适用)
5. 其他Krylov子空间迭代法
6. 实用技术，如Precondition, Multigrid 等

直接法(Gauss消元)

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 10 \\ 2 & 3 & 1 & 1 & 15 \\ 3 & -1 & 2 & -1 & 3 \\ 4 & 1 & -3 & 2 & 5 \end{pmatrix} \xrightarrow{(1)} \begin{pmatrix} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & -1 & -1 & -5 \\ 0 & -4 & -1 & -4 & -27 \\ 0 & -3 & -7 & -2 & -35 \end{pmatrix}$$

$$\xrightarrow{(2)} \begin{pmatrix} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & -1 & -1 & -5 \\ 0 & 0 & -5 & -8 & -47 \\ 0 & 0 & -10 & -5 & -50 \end{pmatrix} \xrightarrow{(3)} \begin{pmatrix} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & -1 & -1 & -5 \\ 0 & 0 & -5 & -8 & -47 \\ 0 & 0 & 0 & 11 & 44 \end{pmatrix}$$

向量化实现脚本(Script)

```
function x = gauss(A, b);
% A is a matrix sized n x n, anb b is a vector with length n;
n = size(A,1);
for k = 1:n-1
    A(k,:) = A(k,:)/A(k,k);
    for j = k+1:n
        factor = -A(j,k)/A(k,k);
        A(j,k:end) = A(j,k:end) + factor*A(k,k:end);
        b(j) = b(j) + factor*b(k);
    end
end
b(n) = b(n)/A(n,n);
for k = n:-1:2
  b(1:k-1) = b(1:k-1) - A(1:k-1,k)*b(k);
end
```

算法分析

1. 时间复杂度：考虑算法执行过程中所做的乘除法次数。一步消元过程所做的乘除次数为
$$\sum_{k=1}^{n}(n-k+1)^2 = \frac{1}{6}n(n+1)(2n+1).$$
回代过程所做的乘法次数为
$$\sum_{k=1}^{n}(n-k) = \frac{1}{2}n(n+1).$$
综上,高斯消去法所需总的乘除法次数为
$$\sum_{k=1}^{n}(n-k+1)^2 + \sum_{k=1}^{n}(n-k) = \frac{1}{3}n(n^2+3n-1) \approx \frac{1}{3}n^3. \tag{1}$$
2. 空间复杂度：不需额外存储空间、**原址**存储分解结果
3. 稳定性：当$|a_{kk}| \approx 0$时可导致浮点数溢出,用、**条件数**$\kappa(A)$衡量

算法分析

1. 时间复杂度：考虑算法执行过程中所做的乘除法次数。一步消元过程所做的乘除次数为

$$\sum_{k=1}^{n}(n-k+1)^2=\frac{1}{6}n(n+1)(2n+1).$$

回代过程所做的乘法次数为

$$\sum_{k=1}^{n}(n-k)=\frac{1}{2}n(n+1).$$

综上,高斯消去法所需总的乘除法次数为

$$\sum_{k=1}^{n}(n-k+1)^2+\sum_{k=1}^{n}(n-k)=\frac{1}{3}n(n^2+3n-1)\approx\frac{1}{3}n^3. \tag{1}$$

2. 空间复杂度：不需额外存储空间、**原址**存储分解结果
3. 稳定性：当$|a_{kk}|\approx 0$时可导致浮点数溢出,用、**条件数**$\kappa(A)$衡量

算法分析

1. 时间复杂度：考虑算法执行过程中所做的乘除法次数。一步消元过程所做的乘除次数为
$$\sum_{k=1}^{n}(n-k+1)^2 = \frac{1}{6}n(n+1)(2n+1).$$
回代过程所做的乘法次数为
$$\sum_{k=1}^{n}(n-k) = \frac{1}{2}n(n+1).$$
综上,高斯消去法所需总的乘除法次数为
$$\sum_{k=1}^{n}(n-k+1)^2 + \sum_{k=1}^{n}(n-k) = \frac{1}{3}n(n^2+3n-1) \approx \frac{1}{3}n^3. \qquad (1)$$
2. 空间复杂度：不需额外存储空间、**原址**存储分解结果
3. 稳定性：当$|a_{kk}| \approx 0$时可导致浮点数溢出,用、**条件数**$\kappa(A)$衡量

Gauss消元- 矩阵LU分解

若对矩阵A的Gauss消去过程时稳定的,则存在(上*下)三角分解

$$A = LU.$$

```
function [L, U] = lu_primer(A);
% A is a matrix sized n x n, anb b is a vector with length n;
n = size(A,1);
for k = 1:n-1
A(k+1:n,k) = A(k+1:n,k)/A(k,k);          % L_{kj}, j=k+1,...,n
for i = k+1:n
A(i,k+1:end) = A(i,k+1:end) + A(i,k)*A(k,k+1:end);
end
end
L = tril(A); U = triu(A) + diag(A);
```

$$A = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 3 & 3 & 12 & 6 \\ 2 & 4 & -1 & 2 \\ 4 & 2 & 1 & 1 \end{bmatrix} \overset{k=1}{\longrightarrow} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & -6 & -7 & -3 \end{bmatrix}$$

$$\overset{k=2}{\longrightarrow} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & 2 & -19 & -9 \end{bmatrix} \overset{k=3}{\longrightarrow} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & 2 & \frac{19}{5} & -9 \end{bmatrix}$$

由最后一步的结果可以分离出L和U分别为:

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{3}{2} & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 2 & 2 & \frac{19}{5} & 1 \end{bmatrix} \quad U = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 0 & -3 & 6 & 3 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & -9 \end{bmatrix}$$

$$A=\begin{bmatrix}2&4&4&2\\3&3&12&6\\2&4&-1&2\\4&2&1&1\end{bmatrix}\xrightarrow{k=1}\begin{bmatrix}2&4&4&2\\\frac{3}{2}&-3&6&3\\1&0&-5&0\\2&-6&-7&-3\end{bmatrix}$$

$$\xrightarrow{k=2}\begin{bmatrix}2&4&4&2\\\frac{3}{2}&-3&6&3\\1&0&-5&0\\2&2&-19&-9\end{bmatrix}\xrightarrow{k=3}\begin{bmatrix}2&4&4&2\\\frac{3}{2}&-3&6&3\\1&0&-5&0\\2&2&\frac{19}{5}&-9\end{bmatrix}$$

由最后一步的结果可以分离出L和U分别为:

$$L=\begin{bmatrix}1&0&0&0\\\frac{3}{2}&1&0&0\\1&0&1&0\\2&2&\frac{19}{5}&1\end{bmatrix}\qquad U=\begin{bmatrix}2&4&4&2\\0&-3&6&3\\0&0&-5&0\\0&0&0&-9\end{bmatrix}$$

$$A = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 3 & 3 & 12 & 6 \\ 2 & 4 & -1 & 2 \\ 4 & 2 & 1 & 1 \end{bmatrix} \xrightarrow{k=1} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & -6 & -7 & -3 \end{bmatrix}$$

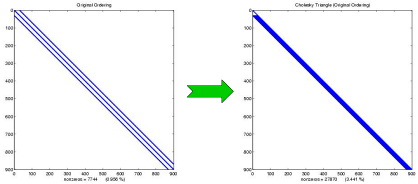
$$\xrightarrow{k=2} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & 2 & -19 & -9 \end{bmatrix} \xrightarrow{k=3} \begin{bmatrix} 2 & 4 & 4 & 2 \\ \frac{3}{2} & -3 & 6 & 3 \\ 1 & 0 & -5 & 0 \\ 2 & 2 & \frac{19}{5} & -9 \end{bmatrix}$$

由最后一步的结果可以分离出L和U分别为:

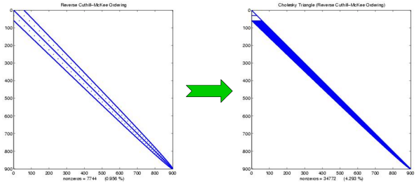
$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{3}{2} & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 2 & 2 & \frac{19}{5} & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 0 & -3 & 6 & 3 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & -9 \end{bmatrix}$$

稀疏矩阵排序技术- 减少“填零”

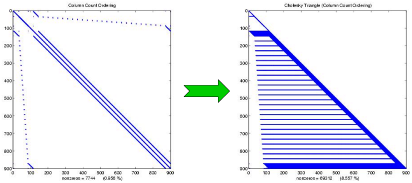
Original Ordering



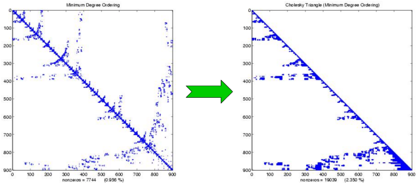
Reverse CMK Ordering



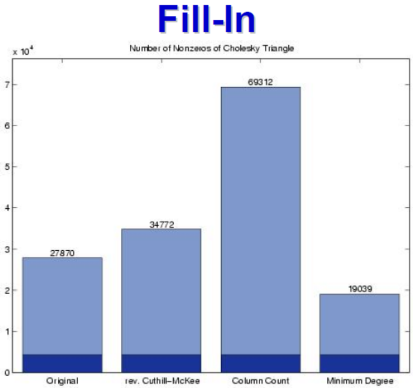
Column Count Ordering

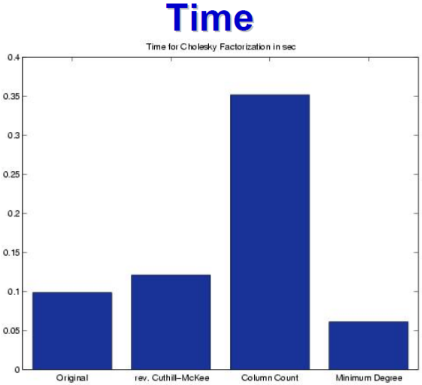


Minimum Degree Ordering



消元法performance





- 直接法在n较大时会耗费大量的时间和存储单元!

简单迭代法

迭代法具有的特点是速度快. 这就需要将线性方程组进行改写

$$\mathrm{x} = G\mathrm{x} + \mathrm{b}$$

那么,当$x^{(0)}$给定后,可以利用迭代格式

$$\mathrm{x}^{(k)} = G\mathrm{x}^{(k-1)} + \mathrm{b}, \quad \forall k = 1, 2, \cdots, n.$$

得到序列$\{x^{(k)}\}_{k=0}^{n}$.

- 序列的收敛只与算子G有关,而与初值$\mathrm{x}^{(0)}$的选取无关!
- 收敛性: 谱半径$\rho(G) < 1$

简单迭代法

迭代法具有的特点是速度快. 这就需要将线性方程组进行改写

$$\mathrm{x} = G\mathrm{x} + \mathrm{b}$$

那么,当$x^{(0)}$给定后,可以利用迭代格式

$$\mathrm{x}^{(k)} = G\mathrm{x}^{(k-1)} + \mathrm{b}, \quad \forall k = 1, 2, \cdots, n.$$

得到序列$\{x^{(k)}\}_{k=0}^{n}$.

- 序列的收敛只与算子G有关,而与初值$\mathrm{x}^{(0)}$的选取无关!
- 收敛性: 谱半径$\rho(G) < 1$

简单迭代法

迭代法具有的特点是速度快. 这就需要将线性方程组进行改写

$$\mathrm{x} = G\mathrm{x} + \mathrm{b}$$

那么,当$x^{(0)}$给定后,可以利用迭代格式

$$\mathrm{x}^{(k)} = G\mathrm{x}^{(k-1)} + \mathrm{b}, \quad \forall k = 1, 2, \cdots, n.$$

得到序列$\{x^{(k)}\}_{k=0}^{n}$.

- 序列的收敛只与算子G有关,而与初值$\mathrm{x}^{(0)}$的选取无关!
- 收敛性: 谱半径$\rho(G) < 1$

一个特例：Jacobi迭代法

对Ax=b

$$A=\begin{bmatrix}0 & & & & \\ a_{21} & 0 & & & \\ a_{31} & a_{32} & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ a_{n1} & a_{n2} & a_{n3} & \cdots & 0\end{bmatrix}+\begin{bmatrix}a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn}\end{bmatrix}+\begin{bmatrix}0 & a_{12} & a_{13} & \cdots & a_{1n} \\ & 0 & a_{23} & \cdots & a_{2n} \\ & & 0 & \cdots & a_{3n} \\ & & & \ddots & \vdots \\ & & & & 0\end{bmatrix}$$

下三角阵　　　　对角阵　　　　上三角阵

$$=L+D+U$$

$\because a_{ii}\neq 0$(Jacobi假设)　　$\therefore D$可逆

进一步，有：　（$L+D+U$）$x=b$

$$Dx=-(L+U)x+b$$

$$x=-D^{-1}(L+U)+D^{-1}b$$ →理论分析

Bai百科

为纪念普鲁士著名数学家雅可比

迭代格式de分量形式

- Jacobi(1845)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k-1)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

- Gauss(1823)-Seidel(1874)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

矩阵G对角占优时两类迭代均收敛。但是有例子表明:

- Gauss-Seidel 法收敛时，Jacobi 法可能不收敛;
- Jacobi 法收敛时，Gauss-Seidel 法也可能不收敛!

迭代格式de分量形式

- Jacobi(1845)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k-1)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

- Gauss(1823)-Seidel(1874)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

矩阵G对角占优时两类迭代均收敛。但是有例子表明:

- Gauss-Seidel 法收敛时，Jacobi 法可能不收敛;
- Jacobi 法收敛时，Gauss-Seidel 法也可能不收敛!

迭代格式de分量形式

- Jacobi(1845)

$$x_i^{(k)} = \frac{1}{G_{ii}}\Big(b_i - \sum_{j=1}^{i-1} G_{ij} x_j^{(k-1)} - \sum_{j=i+1}^{n} G_{ij} x_j^{(k-1)}\Big)$$

- Gauss(1823)-Seidel(1874)

$$x_i^{(k)} = \frac{1}{G_{ii}}\Big(b_i - \sum_{j=1}^{i-1} G_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij} x_j^{(k-1)}\Big)$$

矩阵G对角占优时两类迭代均收敛。但是有例子表明:

- Gauss-Seidel 法收敛时，Jacobi 法可能不收敛;
- Jacobi 法收敛时，Gauss-Seidel 法也可能不收敛!

迭代格式de分量形式

- Jacobi(1845)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k-1)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

- Gauss(1823)-Seidel(1874)

$$x_i^{(k)} = \frac{1}{G_{ii}}\left(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

矩阵G对角占优时两类迭代均收敛。但是有例子表明:

- Gauss-Seidel 法收敛时，Jacobi 法可能不收敛;
- Jacobi 法收敛时，Gauss-Seidel 法也可能不收敛!

迭代格式de分量形式

- Jacobi(1845)

$$x_i^{(k)} = \frac{1}{G_{ii}}\Big(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k-1)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\Big)$$

- Gauss(1823)-Seidel(1874)

$$x_i^{(k)} = \frac{1}{G_{ii}}\Big(b_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\Big)$$

矩阵G对角占优时两类迭代均收敛。但是有例子表明：

- Gauss-Seidel 法收敛时，Jacobi 法可能不收敛；
- Jacobi 法收敛时，Gauss-Seidel 法也可能不收敛!

松弛(Relaxation)迭代–分量形式

在Gauss-Sediel基础上的一种加权修正,分两步

1.
$$\tilde{x}_i^{(k)} = \frac{1}{G_{ii}}\left(g_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

2.
$$\begin{aligned} x_i^{(k)} &= x_i^{(k-1)} + \omega(\tilde{x}_i^{(k)} - x_i^{(k-1)}) \\ &= (1-\omega)x_i^{(k-1)} + \omega\tilde{x}_i^{(k)} \end{aligned}$$

A对称正定时, 松弛迭代收敛$0 < \omega < 2$(depend on ω)。称

- $0 < \omega < 1$ under-relaxation (Liebmann,Richardson,etc., 19xx)
- $\omega > 1$ successive over-relaxation method(Young, 1950)

如何选取ω使矩阵谱半径达到最小? 一般取 $1.4 < \omega < 1.6$.

松弛(Relaxation)迭代–分量形式

在Gauss-Sediel基础上的一种加权修正,分两步

1.
$$\tilde{x}_i^{(k)} = \frac{1}{G_{ii}}\left(g_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

2.
$$\begin{aligned} x_i^{(k)} &= x_i^{(k-1)} + \omega(\tilde{x}_i^{(k)} - x_i^{(k-1)}) \\ &= (1-\omega)x_i^{(k-1)} + \omega\tilde{x}_i^{(k)} \end{aligned}$$

A对称正定时, 松弛迭代收敛$0 < \omega < 2$(depend on ω)。称

- $0 < \omega < 1$ under-relaxation (Liebmann,Richardson,etc., 19xx)
- $\omega > 1$ successive over-relaxation method(Young, 1950)

如何选取ω使矩阵谱半径达到最小? 一般取 $1.4 < \omega < 1.6$.

松弛(Relaxation)迭代–分量形式

在Gauss-Sediel基础上的一种加权修正,分两步

1.
$$\tilde{x}_i^{(k)} = \frac{1}{G_{ii}}\left(g_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

2.
$$\begin{aligned} x_i^{(k)} &= x_i^{(k-1)} + \omega(\tilde{x}_i^{(k)} - x_i^{(k-1)}) \\ &= (1-\omega)x_i^{(k-1)} + \omega\tilde{x}_i^{(k)} \end{aligned}$$

A对称正定时, 松弛迭代收敛$0 < \omega < 2$(depend on ω)。称

- $0 < \omega < 1$ under-relaxation (Liebmann,Richardson,etc., 19xx)
- $\omega > 1$ successive over-relaxation method(Young, 1950)

如何选取ω使矩阵谱半径达到最小? 一般取 $1.4 < \omega < 1.6$.

松弛(Relaxation)迭代–分量形式

在Gauss-Sediel基础上的一种加权修正,分两步

1.
$$\tilde{x}_i^{(k)} = \frac{1}{G_{ii}}\left(g_i - \sum_{j=1}^{i-1} G_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} G_{ij}x_j^{(k-1)}\right)$$

2.
$$\begin{aligned} x_i^{(k)} &= x_i^{(k-1)} + \omega(\tilde{x}_i^{(k)} - x_i^{(k-1)}) \\ &= (1-\omega)x_i^{(k-1)} + \omega\tilde{x}_i^{(k)} \end{aligned}$$

A对称正定时, 松弛迭代收敛$0 < \omega < 2$(depend on ω)。称

- $0 < \omega < 1$ under-relaxation (Liebmann,Richardson,etc., 19xx)
- $\omega > 1$ successive over-relaxation method(Young, 1950)

如何选取ω使矩阵谱半径达到最小? 一般取 $1.4 < \omega < 1.6$.

算例- 迭代法对比

Example

- The linear system $A\mathbf{x} = \mathbf{b}$ given by

$$\begin{aligned} 4x_1 + 3x_2 \quad\quad &= 24 \\ 3x_1 + 4x_2 - \quad x_3 &= 30 \\ - \quad x_2 + 4x_3 &= -24 \end{aligned}$$

  has the solution $(3, 4, -5)^t$.
- Compare the iterations from the Gauss-Seidel method and the SOR method with $\omega = 1.25$ using $\mathbf{x}^{(0)} = (1, 1, 1)^t$ for both methods.

Solution (1/3)

For each $k = 1, 2, \ldots,$ the equations for the Gauss-Seidel method are

$$\begin{aligned} x_1^{(k)} &= -0.75x_2^{(k-1)} + 6 \\ x_2^{(k)} &= -0.75x_1^{(k)} + 0.25x_3^{(k-1)} + 7.5 \\ x_3^{(k)} &= 0.25x_2^{(k)} - 6 \end{aligned}$$

and the equations for the SOR method with $\omega = 1.25$ are

$$\begin{aligned} x_1^{(k)} &= -0.25x_1^{(k-1)} - 0.9375x_2^{(k-1)} + 7.5 \\ x_2^{(k)} &= -0.9375x_1^{(k)} - 0.25x_2^{(k-1)} + 0.3125x_3^{(k-1)} + 9.375 \\ x_3^{(k)} &= 0.3125x_2^{(k)} - 0.25x_3^{(k-1)} - 7.5 \end{aligned}$$

Gauss-Seidel Iterations

k	0	1	2	3	$\cdots$	7
$x_1^{(k)}$	1	5.250000	3.1406250	3.0878906		3.0134110
$x_2^{(k)}$	1	3.812500	3.8828125	3.9267578		3.9888241
$x_3^{(k)}$	1	-5.046875	-5.0292969	-5.0183105		-5.0027940

SOR Iterations ($\omega = 1.25$)

k	0	1	2	3	$\cdots$	7
$x_1^{(k)}$	1	6.312500	2.6223145	3.1333027		3.0000498
$x_2^{(k)}$	1	3.5195313	3.9585266	4.0102646		4.0002586
$x_3^{(k)}$	1	-6.6501465	-4.6004238	-5.0966863		-5.0003486

观察与分析

为了达到小数点后第 7 位的精度，

- Gauss-Sediel:　　34步迭代
- SOR:　　　　14步迭代

Example (Exercises)

请给出一个你认为最优的ω

观察与分析

为了达到小数点后第 7 位的精度,

- Gauss-Sediel:　　34步迭代
- SOR:　　　　14步迭代

Example (Exercises)

请给出一个你认为最优的ω

1. Introduction

2. Linear Solver

3. Sparse Matrix

Krylov子空间迭代法

构造迭代

$$Kx_{i+1} = Kx_i + (b - Ax_i)$$

- K(来源于A. N. Krylov)是用**投影法**构造的 A 的近似
- 将复杂问题简化为易于计算的多步，在第 m 步构建子空间

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, \cdots, A^{m-1}r_0\}.$$

- Hestenes,Stiefel, Lanczos(Conjugate Gradient, 1950)

  \- 美国国家标准局数值分析研究所
- 根据 K 所属空间 $\mathcal{K}_m$ 的不同可构造不同类型的迭代法

Krylov子空间迭代法

构造迭代

$$Kx_{i+1} = Kx_i + (b - Ax_i)$$

- K(来源于A. N. Krylov)是用**投影法**构造的 A 的近似
- 将复杂问题简化为易于计算的多步，在第 m 步构建子空间

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, \cdots, A^{m-1}r_0\}.$$

- Hestenes,Stiefel, Lanczos(Conjugate Gradient, 1950)

  \- 美国国家标准局数值分析研究所
- 根据 K 所属空间 $\mathcal{K}_m$ 的不同可构造不同类型的迭代法

Krylov子空间迭代法

构造迭代

$$Kx_{i+1} = Kx_i + (b - Ax_i)$$

- K(来源于A. N. Krylov)是用**投影法**构造的 A 的近似
- 将复杂问题简化为易于计算的多步，在第 m 步构建子空间

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, \cdots, A^{m-1}r_0\}.$$

- Hestenes,Stiefel, Lanczos(Conjugate Gradient, 1950)

  - 美国国家标准局数值分析研究所

- 根据 K 所属空间 $\mathcal{K}_m$ 的不同可构造不同类型的迭代法

Alexei Nikolaevich Krylov

1863-1945

Maritime Engineer

300 papers and books:
shipbuilding, magnetism,
artillery, math, astronomy

1890: Theory of oscillating
motions of the ship

1931: *Krylov subspace methods*

投影法的概念

Let A be an $n \times n$ real matrix and $\mathcal{K}$ and $\mathcal{L}$ be two m -dimensional subspaces of R_n. A projection technique onto the subspace $\mathcal{K}$ and orthogonal to $\mathcal{L}$ is a process described as

$$\text{Find } \tilde{x} \in x_0 + \mathcal{K}, \text{ such that } b - A\tilde{x} \perp \mathcal{L}$$

等价于

$$\text{Find } \tilde{x} = x_0 + \delta, \delta \in \mathcal{K}, \text{ such that } (r_0 - A\delta, \omega) = 0, \forall \omega \in \mathcal{L}$$

在Krylov子空间方法中,可以取 $\mathcal{L}_m = \mathrm{K}_m$ 或 $\mathcal{L}_m = A\mathrm{K}_m$.

投影法的概念

Let A be an $n \times n$ real matrix and $\mathcal{K}$ and $\mathcal{L}$ be two m -dimensional subspaces of R_n. A projection technique onto the subspace $\mathcal{K}$ and orthogonal to $\mathcal{L}$ is a process described as

$$\text{Find } \tilde{x} \in x_0 + \mathcal{K}, \text{ such that } b - A\tilde{x} \perp \mathcal{L}$$

等价于

$$\text{Find } \tilde{x} = x_0 + \delta, \delta \in \mathcal{K}, \text{ such that } (r_0 - A\delta, \omega) = 0, \forall \omega \in \mathcal{L}$$

在Krylov子空间方法中,可以取 $\mathcal{L}_m = \mathrm{K}_m$ 或 $\mathcal{L}_m = A\mathrm{K}_m$.

投影法的概念

Let A be an $n \times n$ real matrix and $\mathcal{K}$ and $\mathcal{L}$ be two m -dimensional subspaces of R_n. A projection technique onto the subspace $\mathcal{K}$ and orthogonal to $\mathcal{L}$ is a process described as

$$\text{Find } \tilde{x} \in x_0 + \mathcal{K}, \text{ such that } b - A\tilde{x} \perp \mathcal{L}$$

等价于

$$\text{Find } \tilde{x} = x_0 + \delta, \delta \in \mathcal{K}, \text{ such that } (r_0 - A\delta, \omega) = 0, \forall \omega \in \mathcal{L}$$

在Krylov子空间方法中,可以取 $\mathcal{L}_m = \mathrm{K}_m$ 或 $\mathcal{L}_m = A\mathrm{K}_m$.

实用投影法

1. 最速下降法(Steep Descent Method)

for $j = 0, 1, \ldots,$ *until convergence* **do**

$\quad \mathrm{r}_j = \mathrm{b} - A\mathrm{x}_j;$

$\quad \alpha_j = (\mathrm{r}_j, \mathrm{r}_j)/(A\mathrm{r}_j, \mathrm{r}_j);$

$\quad \mathrm{x}_{j+1} = \mathrm{x}_j + \alpha_j \mathrm{r}_j;$

end

2.极小化残量迭代法(Minimal Residual Iteration)

for $j = 0, 1, \ldots,$ *until convergence* **do**

$\quad [\ \mathrm{r}_j = \mathrm{b} - A\mathrm{x}_j;$

$\quad \alpha_j = (A\mathrm{r}_j, \mathrm{r}_j)/(A\mathrm{r}_j, A\mathrm{r}_j);$

$\quad \mathrm{x}_{j+1} = \mathrm{x}_j + \alpha_j \mathrm{r}_j;$

end

实用投影法

1. 最速下降法(Steep Descent Method)

for $j = 0, 1, \ldots,$ *until convergence* **do**

$\quad r_j = b - Ax_j;$

$\quad \alpha_j = (r_j, r_j)/(Ar_j, r_j);$

$\quad x_{j+1} = x_j + \alpha_j r_j;$

end

2.极小化残量迭代法(Minimal Residual Iteration)

for $j = 0, 1, \ldots,$ *until convergence* **do**

$\quad [\ r_j = b - Ax_j;$

$\quad \alpha_j = (Ar_j, r_j)/(Ar_j, Ar_j);$

$\quad x_{j+1} = x_j + \alpha_j r_j;$

end

利用Arnoldi算法构造正交子空间K_m

Choose a unit vector v_1;

for $j = 1, 2, \ldots, m$ **do**

  Compute $h_{ij} = (Av_j, v_i), \quad \forall i = 1, 2, \cdots, j$;

  Compute $u_j = Av_j - \sum_{i=1}^{j} h_{ij} v_i$;

  $h_{j+1,j} = \|u_j\|_2$;

  if $h_{j+1,j} == 0$ **then**

    Stop

  else

    $v_{j+1} = u_j / h_{j+1,j}$

  end

end

Arnoldi-Modified Gram-Schmidt正交化过程

Choose a unit vector v_1;

for $j = 1, 2, \ldots, m$ **do**

  Compute $\mathrm{u}_j = A\mathrm{v}_j$;

  for $i = 1, \ldots, j$ **do**

    $h_{i,j} = (\mathrm{u}_j, \mathrm{v}_i)$;

    $\mathrm{u}_j = \mathrm{u}_j - h_{i,j}\mathrm{v}_i$;

  end

  $h_{j+1,j} = \|\mathrm{u}_j\|_2$;

  if $h_{j+1,j} == 0$ **then**

    Stop

  else

    $\mathrm{v}_{j+1} = \mathrm{u}_j / h_{j+1,j}$

Sketch: Full Orthogonalization Method(FOM)

1 $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, and $v_1 = r_0/\beta$, $H_m = \{h_{i,j}\}_{i,j=1,2,\dots,m}$;

2 **for** $j = 1, 2, \dots, m$ **do**

3   Compute $u_j = Av_j$;

4   **for** $i = 1, \dots, j$ **do**

5         $h_{i,j} = (u_j, v_i)$;

6         $u_j = u_j - h_{i,j}v_i$;

7   **end**

8     $h_{j+1,j} = \|u_j\|_2$;

9   **if** $h_{j+1,j} == 0$ **then**

10   set $m = j$, and goto line 15

11   **else**

12   Compute $v_{j+1} = u_j/h_{j+1,j}$

Scheme A: Generalized Minimized Residual(GMRes)

1 $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, and $v_1 = r_0/\beta$, $H_m = \{h_{i,j}\}_{1\le i\le(m+1)}^{1\le j\le m}$;
2 **for** $j = 1, 2, \ldots, m$ **do**
3   Compute $u_j = Av_j$;
4   **for** $i = 1, \ldots, j$ **do**
5         $h_{i,j} = (u_j, v_i)$;
6         $u_j = u_j - h_{i,j}v_i$;
7   **end**
8     $h_{j+1,j} = \|u_j\|_2$;
9   **if** $h_{j+1,j} == 0$ **then**
10   set $m = j$, and goto line 15
11   **else**
12   Compute $v_{j+1} = u_j/h_{j+1,j}$

Scheme B: Conjugate Gradient

FOM在A对称时的特殊情形,利用Lanczos三项递推关系可将FOM简化成如下简洁的算法

Compute $\mathrm{r}_0 = \mathrm{b} - A\mathrm{x}_0$, $\mathrm{p}_0 = \mathrm{r}_0$;
for $j = 0, 1, \ldots,$*until convergence* **do**
  $\alpha_j = (\mathrm{r}_j, \mathrm{r}_j)/(A\mathrm{p}_j, \mathrm{p}_j)$;
  $\mathrm{x}_{j+1} = \mathrm{x}_j + \alpha_j \mathrm{p}_j$;
  $\mathrm{r}_{j+1} = \mathrm{r}_j - \alpha_j A\mathrm{p}_j$;
  $\beta_j = (\mathrm{r}_{j+1}, \mathrm{r}_{j+1})/(\mathrm{r}_j, \mathrm{r}_j)$;
  $\mathrm{p}_{j+1} = \mathrm{r}_{j+1} + \beta_j \mathrm{p}_j$;
end

CG的收敛性

令x_m 是执行第$m-$步Conjugate Gradient algorithm得到的近似解，并且x^*是精确解，那么

$$||x^* - x_m||_A \leq \left[\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right]^m ||x^* - x_0||_A$$

其中κ是矩阵A最大与最小特征值的比值，即所谓条件数。可见

- κ越接近1，则共轭梯度法收敛越快！

CG的预处理

设M是non-singular矩阵，并且$M^{-1}A$的条件数相对教小，则求解

$$(M^{-1}A)\mathrm{x} = M^{-1}\mathrm{b}$$

相对容易，或者

$$(AM^{-1})\mathrm{y} = \mathrm{b},$$

再求$M\mathrm{x} = \mathrm{y}$得到原方程的解。

- M对称正定for CG算法
- $M\mathrm{x} = \mathrm{y}$容易求解

for e.g., assume that $A = L + D + L^T$,
$M_1 = D^{-1}$;
$M_2 = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega L^T), 0 < \omega < 2.$

CG的预处理

设M是non-singular矩阵，并且$M^{-1}A$的条件数相对教小，则求解

$$(M^{-1}A)\mathrm{x} = M^{-1}\mathrm{b}$$

相对容易，或者

$$(AM^{-1})\mathrm{y} = \mathrm{b},$$

再求$M\mathrm{x} = \mathrm{y}$得到原方程的解。

- M对称正定for CG算法
- $M\mathrm{x} = \mathrm{y}$容易求解

for e.g., assume that $A = L + D + L^T$,

$M_1 = D^{-1}$;

$M_2 = \frac{1}{\omega(2-\omega)}(D+\omega L)D^{-1}(D+\omega L^T), 0 < \omega < 2.$

CG的预处理

设M是non-singular矩阵，并且$M^{-1}A$的条件数相对教小，则求解

$$(M^{-1}A)\mathrm{x} = M^{-1}\mathrm{b}$$

相对容易，或者

$$(AM^{-1})\mathrm{y} = \mathrm{b},$$

再求$M\mathrm{x} = \mathrm{y}$得到原方程的解。

- M对称正定for CG算法
- $M\mathrm{x} = \mathrm{y}$容易求解

for e.g., assume that $A = L + D + L^T$,

$M_1 = D^{-1}$;

$M_2 = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega L^T), 0 < \omega < 2.$

CG的预处理

设M是non-singular矩阵，并且$M^{-1}A$的条件数相对教小，则求解

$$(M^{-1}A)\mathrm{x} = M^{-1}\mathrm{b}$$

相对容易，或者

$$(AM^{-1})\mathrm{y} = \mathrm{b},$$

再求$M\mathrm{x} = \mathrm{y}$得到原方程的解。

- M对称正定for CG算法
- $M\mathrm{x} = \mathrm{y}$容易求解

for e.g., assume that $A = L + D + L^T$,
$M_1 = D^{-1}$;
$M_2 = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega L^T), 0 < \omega < 2.$

CG算法的其他常见形式

- ICCG： Incomplete Cholesky预处理的CG迭代
- BiCG： 双正交共轭梯度法
- BiCGstab： 稳定化的BiCG

附录1：一些著名参考教材

- Lloyd N. Trefethen & David Bau III: Numerical Linear Algebra
- R. S. Varga: Matrix Iterative Analysis(2nd Edition)
- Gene H Golub & van Loan: Matrix Computation
- James W. Demmel: Applied Numerical Linear Algebra
- Yousef Saad: Iterative methods for sparse systems(2nd edition)

附录1：一些著名参考教材

- Lloyd N. Trefethen & David Bau III: Numerical Linear Algebra
- R. S. Varga: Matrix Iterative Analysis(2nd Edition)
- Gene H Golub & van Loan: Matrix Computation
- James W. Demmel: Applied Numerical Linear Algebra
- Yousef Saad: Iterative methods for sparse systems(2nd edition)

附录2：数值线性代数软件包

- lapack：适用于小规模满矩阵，benchmark
- Spooles/SuperLU：基于矩阵直接分解法
- Petsc/Slepc 大规模稀疏矩阵算法,并行化
- hypre：多重网格预处理
- boomerAMG：代数多重网格
- Matlab/SciLab/Octave：集成环境，脚本语言

附录2：数值线性代数软件包

- lapack：适用于小规模满矩阵，benchmark
- Spooles/SuperLU：基于矩阵直接分解法
- Petsc/Slepc 大规模稀疏矩阵算法,并行化
- hypre：多重网格预处理
- boomerAMG：代数多重网格
- Matlab/SciLab/Octave：集成环境，脚本语言

附录2：数值线性代数软件包

- lapack：适用于小规模满矩阵，benchmark
- Spooles/SuperLU：基于矩阵直接分解法
- Petsc/Slepc 大规模稀疏矩阵算法,并行化
- hypre：多重网格预处理
- boomerAMG：代数多重网格
- Matlab/SciLab/Octave：集成环境，脚本语言

附录2：数值线性代数软件包

- lapack：适用于小规模满矩阵，benchmark
- Spooles/SuperLU：基于矩阵直接分解法
- Petsc/Slepc 大规模稀疏矩阵算法,并行化
- hypre：多重网格预处理
- boomerAMG：代数多重网格
- Matlab/SciLab/Octave：集成环境，脚本语言

开源库安装案例: Spooles 2.2

1. 下载最新版本，目前为2.2版本，更新慢;
2. 解压缩: tar -xvf spooles.2.2.tar.gz;
3. 修改目录下Make.inc中的编译命令，最简单是gcc;
4. 构建库: make lib.

开源库使用案例- A为14922阶的稀疏矩阵

求解线性方程

$$AX = Y$$

where A is square, large and sparse, and X and Y are dense matrices with one or more columns. 在Spooles 2.2 中

1. 内存中建立线性方程组
   1.1 Constructing an InpMtx object that holds the entries of A;
   1.2 Constructing a DenseMtx objcet that holds the entries of Y;
   1.3 Constructing a DenseMtx object to hold the entries of X.
2. 利用Spooles中的Bridge类进行求解:
   2.1 Initialization and setup step;
   2.2 Factorization step;
   2.3 Solution step

Thanks for your attentation!