数值代数基础

文再文

北京大学北京国际数学研究中心

http://bicmr.pku.edu.cn/~wenzw/optbook.html

提纲

向量范数的定义

定义

令记号$\|\cdot\|:\mathbb{R}^n\to\mathbb{R}^+$是一种非负函数，如果它满足:

- 正定性: 对于$\forall v\in\mathbb{R}^n$, 有$\|v\|\geqslant 0$, 且$\|v\|=0\Leftrightarrow v=0_{n\times 1}$;
- 齐次性: 对于$\forall v\in\mathbb{R}^n$和$\alpha\in\mathbb{R}$, 有$\|\alpha v\|=|\alpha|\,\|v\|$;
- 三角不等式: 对于$\forall v,w\in\mathbb{R}^n$, 均成立$\|v+w\|\leqslant\|v\|+\|w\|$.

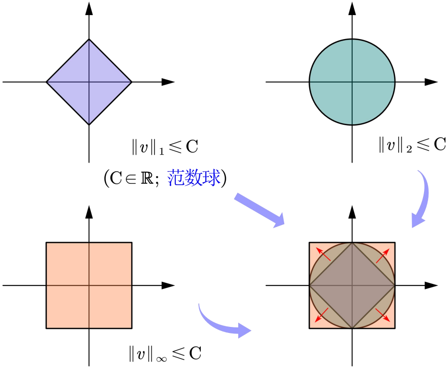
则称$\|\cdot\|$是定义在向量空间$\mathbb{R}^n$上的**向量范数**.

最常用的向量范数即我们熟知的ℓ_p范数(其中$p\geqslant 1$):

$$\|v\|_p=\left(\sum_{i=1}^{n}|v_i|^p\right)^{\frac{1}{p}};\quad \|v\|_\infty=\max_{1\leqslant j\leqslant n}|v_j|.$$

柯西不等式: 设$a,b\in\mathbb{R}^n$, 则$|a^{\mathrm{T}}b|\leqslant\|a\|_2\,\|b\|_2$, 且等号成立的条件是$a$与$b$线性相关.

向量范数的定义



容易看出, $p=\infty$时, 有关"最大值"的定义要求向量的分量是有限的. 在一般化的空间中, 这一要求很可能不成立, 此时我们只需将"最大值"更换成"上确界"即可.

向量范数度量的是v与零点之间的距离. 在实际应用时, 我们通常使用$p=1,2,\infty$的情形, 即分别使用$\|v\|_1,\|v\|_2,\|v\|_\infty$度量$v$在不同意义下的距离, 这是因为它们具有鲜明的度量特征.

左图是它们各自的范数球实例, 请想一想不同范数所度量的距离分别具有怎样的特征? 这些特征分别适用于度量什么情形?

矩阵范数

矩阵范数可以由向量范数的定义推广得到。常见的矩阵范数有：

- $\|A\|_1 = \sum_{i,j} |A_{ij}|$
- $\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2} = \sqrt{\mathbf{Tr}\left(AA^{\mathrm{T}}\right)}$
- 算子范数是一类特殊的矩阵范数，它由向量范数诱导得到:

$$\|A\|_{(m,n)} = \max_{x\in\mathbb{R}^n, \|x\|_{(n)}=1} \|Ax\|_{(m)}\,.$$

  - $p = 1$时，$\|A\|_{p=1} = \max\limits_{\|x\|_1=1} \|Ax\|_1 = \max\limits_{1\leqslant j\leqslant n} \sum_{i=1}^m |a_{ij}|$.
  - $p = 2$时，$\|A\|_{p=2} = \max\limits_{\|x\|_2=1} \|Ax\|_2 = \sqrt{\lambda_{\max}(A^{\mathrm{T}}A)}$，又称为$A$的谱范数.
  - $p = \infty$时，$\|A\|_{p=\infty} = \max\limits_{\|x\|_\infty=1} \|Ax\|_\infty = \max\limits_{1\leqslant i\leqslant m} \sum_{j=1}^n |a_{ij}|$.

- 核范数定义为

$$\|A\|_* = \sum_{i=1}^{r} \sigma_i,$$

其中$\sigma_i(i = 1, \cdots, r)$为$A$的所有非零奇异值, $r = \textbf{rank}\,(A)$.

- 矩阵A, B的内积定义为

$$\langle A, B\rangle = \mathrm{Tr}\left(AB^{\mathrm{T}}\right) = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}b_{ij}.$$

- 柯西不等式: 设$A, B \in \mathbb{R}^{m\times n}$, 则

$$|\langle A, B\rangle| \leqslant \|A\|_F \|B\|_F,$$

等号成立当且仅当A和B线性相关.

提纲

1. 范数
2. 矩阵的结构与算法复杂度
3. 数值代数软件
4. 通过分解矩阵来求解线性方程
5. 分块消去法
6. 稀疏数值线性代数
7. 特征值分解与奇异值分解

矩阵结构与算法复杂度

对于$A \in \mathbb{R}^{n\times n}$，求解方程组$Ax = b$的时间代价为：

- 对于大多数方法，计算所需时间会以n^3的量级增长。
- 当A具有某种结构时（如带状矩阵、稀疏矩阵、常对角矩阵），计算量可以减少。

flop 计数

- flop (floating-point operation): 两个浮点数之间的一次加法、减法、乘法或除法
- 估计算法复杂度: 将计算所需的flops数量表示为一个关于问题规模的（多项式）函数，并简化成只保留最高项。
- 不能准确预测算法在现代计算机上的计算时间。
- 可作为复杂度的粗略估计

向量-向量运算 $(x, y \in \mathbb{R}^n)$

- 内积$x^T y$: $2n-1$ flops (当n较大时，可记为$2n$)
- 求和$x+y$, 标量乘法αx: n flops

矩阵-向量乘法 $y = Ax$, $A \in \mathbb{R}^{m \times n}$

- $m(2n-1)$ flops (当n较大时，可记为$2mn$)
- 当A为稀疏矩阵时，若有N个非零元素，则计算量为$2N$
- 当$A = UV^T, U \in \mathbb{R}^{m \times p}, V \in \mathbb{R}^{n \times p}$时，计算量为$2p(n+m)$

矩阵-矩阵乘法 $C = AB$, $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}$

- $mp(2n-1)$ flops (当n较大时，可记为$2mnp$)
- 当A 和（或）B 是稀疏矩阵时，运算量会少一些
- 当$m = p$ 且C是对称矩阵时，flops数量为$(1/2)m(m+1)(2n-1) \approx m^2 n$

提纲

Basic Linear Algebra Subroutines (BLAS)

BLAS实现了数值代数基本操作，这些操作按照运算量可划分为3个层次（*level*）：

- *level 1*, 1973-1977: $O(n)$ 向量操作: 向量加法、数乘、点乘、范数运算
- *level 2*, 1984-1986: $O(n^2)$ 矩阵-向量操作: 矩阵-向量乘法，三角矩阵-向量求解，矩阵的秩1更新与对称矩阵的秩2更新
- *level 3*, 1987-1990: $O(n^3)$ 矩阵-矩阵操作: 矩阵-矩阵乘法，三角矩阵-矩阵求解，低秩矩阵更新

BLAS的运算

Level 1	加法/数乘	$\alpha x, \quad \alpha x + y$
	点乘，范数	$x^T y, \quad \\|x\\|_2, \quad \\|x\\|_1$
Level 2	矩阵/向量乘积	$\alpha Ax + \beta y, \quad \alpha A^T x + \beta y$
	秩1更新	$A + \alpha xy^T, \quad A + \alpha xx^T$
	秩2更新	$A + \alpha xy^T + \alpha yx^T$
	三角矩阵-向量求解	$\alpha T^{-1} x, \quad \alpha T^{-T} x$
Level 3	矩阵/矩阵乘积	$\alpha AB + \beta C, \quad \alpha AB^T + \beta C$
		$\alpha A^T B + \beta C, \quad \alpha A^T B^T + \beta C$
	秩k更新	$\alpha AA^T + \beta C, \quad \alpha A^T A + \beta C$
	秩k更新	$\alpha A^T B + \alpha B^T A + \beta C$
	三角矩阵求解	$\alpha T^{-1} C, \quad \alpha T^{-T} C$

Level 1 BLAS 命名惯例

BLAS 的命名惯例受Fortran启发

cblas_	X	XXXX
前缀	数据类型	操作

数据类型:

s	单精度实数	d	双精度实数
c	单精度复数	z	双精度复数

操作:

axpy	$y \leftarrow \alpha x + y$	dot	$r \leftarrow x^T y$
nrm2	$r \leftarrow \|x\|_2 = \sqrt{x^T x}$	asum	$r \leftarrow \|x\|_1 = \sum_i \lvert x_i \rvert$

示例:

cblas_ddot　双精度实值内积运算

BLAS 命名惯例: Level 2/3

cblas_	X	XX	XXX
前缀	数据类型	结构	操作

矩阵结构:

tr	三角矩阵	tp	压缩三角矩阵	tb	带状三角矩阵
sy	对称矩阵	sp	压缩对称矩阵	sb	带状对称矩阵
hy	厄米特矩阵	hp	压缩厄米特矩阵	hn	带状厄米特矩阵
ge	一般矩阵			gb	带状一般矩阵

操作:

mv	$y \leftarrow \alpha Ax + \beta y$	sv	$x \leftarrow A^{-1}x$ (triangular only)
r	$A \leftarrow A + xx^T$	r2	$A \leftarrow A + xy^T + yx^T$
mm	$C \leftarrow \alpha AB + \beta C$	r2k	$C \leftarrow \alpha AB^T + \alpha BA^T + \beta C$

示例:

cblas_dtrmv	双精度实值三角矩阵与向量乘积
cblas_dsyr2k	双精度实值对称、秩2k矩阵更新

高效运用BLAS

用一个高层次的BLAS程序来代替多次调用低层次程序达到同样效果

$$A \leftarrow A + \sum_{i=1}^{k} x_i y_i^T, \quad A \in \mathbb{R}^{m\times n}, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}^n$$

两种选择: 调用k次层次2的程序`cblas_dger`

$$A \leftarrow A + x_1 y_1^T, \quad \dots \quad A \leftarrow A + x_k y_k^T$$

或调用一次层次3的程序`cblas_dgemm`

$$A \leftarrow A + XY^T, \quad X = [x_1 \dots x_k], \quad Y = [y_1 \dots y_k]$$

后者表现会更好

BLAS是必要的吗?

为什么用BLAS可以简化你的程序?

$$A \leftarrow A + XY^T, \quad A \in \mathbb{R}^{m\times n}, x_i \in \mathbb{R}^{m\times p}, y_i \in \mathbb{R}^{n\times p}$$

$$A_{ij} \leftarrow A_{ij} + \sum_{k=1}^{p} X_{ik} Y_{jk}$$

```
void matmutadd( int m, int n, int p, double* A,
        const double* X, const double* Y ) {
  int i, j, k;
  for ( i = 0 ; i < m ; ++i )
    for ( j = 0 ; j < n ; ++j )
      for ( k = 0 ; k < p ; ++k )
        A[i + j * n] += X[i + k * p] * Y[j + k * p];
}
```

BLAS是必要的吗？

- 优化调试好的BLAS程序会比你自己写的版本速度更快——通常会快10倍或更多。
- BLAS会根据你的中央处理器和高速缓存大小来选择合适的区块大小，以达到最优的变现。
- ATLAS (automatically tuned linear algebra software)

  `http://math-atlas.sourceforge.net`

  使用自动代码生成和测试方法，为特定的计算机优化BLAS库。

Linear Algebra PACKage (LAPACK)

LAPACK包含了求解线性系统和进行常见矩阵分解的程序。

- 首次发行: February 1992; 最新版本(3.0): May 2000
- 取代了原有的EISPACK 和LINPACK
- 支持与BLAS相同的数据类型（单/双精度实/复值）和矩阵结构（对称矩阵、带状矩阵等）
- 用BLAS作为程序的内核
- 其程序可以划分为三类: 辅助子程序（auxiliary）、计算子程序（computational）和驱动子程序（driver）

LAPACK 辅助和计算子程序

辅助子程序主要完成的是一些底层的操作

- 获取当前机器的机器精度和寄存器大小.
- 生成均匀分布或高斯分布的随机数.
- 使用低层次BLAS 运算完成矩阵分解.

LAPACK 计算子程序完成单一、特定任务

- 矩阵分解: $LU, LL^T/LL^H, LDL^T/LDL^H, QR, LQ, QRZ$, 广义$QR$ 和RQ
- 对称/厄米特矩阵和非对称矩阵的特征值分解
- 三对角矩阵的特征值分解和二对角矩阵的奇异值分解.
- 奇异值分解
- 广义特征值和奇异值分解

LAPACK 驱动子程序

驱动子程序调用一系列计算子程序解决标准的线性代数问题，例如：

- 线性方程: $AX = B$
- 线性最小二乘: $\text{minimize}_x \|b - Ax\|_2$
- 线性最小范数:

$$\begin{array}{ll} \text{minimize}_x & \|c - Ax\|_2 \\ \text{subject to} & Bx = d \end{array} \qquad \begin{array}{ll} \text{minimize}_y & \|y\|_2 \\ \text{subject to} & d = Ax + By \end{array}$$

- 实对称矩阵的特征值分解.

其他形式的Lapack

并行与分布式计算

- Scalapack
  `http://www.netlib.org/scalapack/`
- Elemental
  `https://github.com/elemental/Elemental`

GPU:

- MAGMA
  `http://icl.cs.utk.edu/magma/`
- PLASMA
  `https://bitbucket.org/icl/plasma`

提纲

易解的线性方程

（块）对角矩阵 ($a_{ij} = 0$ if $i \neq j$): n flops

$$x = A^{-1}b = (b_1/a_{11}, \ldots, b_n/a_{nn})$$

下三角矩阵 ($a_{ij} = 0$ if $j > i$): n^2 flops

$$\begin{aligned}
x_1 &:= b_1/a_{11} \\
x_2 &:= (b_2 - a_{21}x_1)/a_{22} \\
x_3 &:= (b_3 - a_{31}x_1 - a_{32}x_2)/a_{33} \\
&\vdots \\
x_n &:= (b_n - a_{n1}x_1 - a_{n2}x_2 - \cdots - a_{n,n-1}x_{n-1})/a_{nn}
\end{aligned}$$

被称为前代法

上三角矩阵 ($a_{ij} = 0$ if $j < i$): n^2 flops 回代法

正交矩阵: $A^{-1} = A^T$

- $2n^2$ flops，对于一般的A 计算$x = A^T b$
- 当A 有结构时，运算量会更少，如$A = I - 2uu^T$（Household矩阵），其中$\|u\|_2 = 1$, 计算$x = A^T b = b - 2(u^T b)u$，只需要$4n$ flops

置换矩阵:

$$a_{ij} = \begin{cases} 1 & j = \pi_i \\ 0 & \text{otherwise} \end{cases}$$

其中$\pi = (\pi_1, \pi_2, ..., \pi_n)$ 是$(1, 2, ..., n)$的一个置换

- 乘法可以表示为: $Ax = (x_{\pi_1}, ..., x_{\pi_n})$
- 满足$A^{-1} = A^T$, 因此计算$Ax = b$ 的代价是0 flops

例子:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \qquad A^{-1} = A^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

通过因子分解来求解$Ax=b$

- 将A 分解为几个简单矩阵的乘积（通常为2到3个）

$$A = A_1 A_2 \cdots A_k$$

(A_i 是对角矩阵、上三角矩阵、下三角矩阵等)

- 通过求解k个“简单”的方程来计算$x = A^{-1}b = A_k^{-1} \cdots A_2^{-1} A_1^{-1} b$

$$A_1 x_1 = b, \qquad A_2 x_2 = x_1, \qquad \ldots, \qquad A_k x = x_{k-1}$$

分解矩阵的计算代价是求解方程组的主要代价

多个系数矩阵相同的方程

$$A x_1 = b_1, \qquad A x_2 = b_2, \qquad \ldots, \qquad A x_m = b_m$$

代价: 一次分解加m次求解

LU 分解

所有的非奇异矩阵A均可分解为

$$A = PLU$$

其中P 是置换矩阵, L 是下三角矩阵且对角元均为1, U 是上三角矩阵

代价: $(2/3)n^3$ flops

用LU分解来求解线性方程

给定 线性方程$Ax = b$, 其中A 非奇异.

1. LU 分解 将A 分解为$A = PLU$ ($(2/3)n^3$ flops).
2. 置换 求解$Pz_1 = b$ (0 flops).
3. 前代法 求解$Lz_2 = z_1$ (n^2 flops).
4. 回代法 求解$Ux = z_2$ (n^2 flops).

代价: $(2/3)n^3 + 2n^2 \approx (2/3)n^3$ （对于较大的n）

稀疏矩阵**LU**分解

$$A = P_1 L U P_2$$

- 增加一个置换矩阵P_2，来保证L和U的稀疏性(进而减少分解和求解代价)
- （启发式地）选择P_1 和P_2 来生成稀疏的L, U
- P_1 和P_2 的选择基于A的稀疏性模式和取值
- 代价通常低于$(2/3)n^3$; 确切的数值以复杂的形式取决于n、A中0元的个数、A的稀疏模式。

Cholesky分解

所有正定矩阵A可以被分解为

$$A = LL^T$$

其中L是下三角矩阵

代价: $(1/3)n^3$ flops

用Cholesky求解线性方程

给定 线性方程$Ax = b$, 其中$A \in \mathbb{S}^n_{++}$.

1. Cholesky分解 将A 分解为$A = LL^T$ ($(1/3)n^3$ flops).
2. 前代法 求解$Lz_1 = b$ (n^2 flops).
3. 回代法 求解$L^T x = z_1$ (n^2 flops).

代价: $(1/3)n^3 + 2n^2 \approx (1/3)n^3$ （对于较大的n）

稀疏**Cholesky** 分解

$$A = PLL^T P^T$$

- 增加一个置换矩阵P 来保证L 的稀疏性
- （启发式地）选择P 来生成稀疏的L
- P的选择仅基于A的稀疏模式(与稀疏LU不同)
- 代价通常低于$(1/3)n^3$; 确切的数值以复杂的形式取决于n、A中0元的个数、A的稀疏模式。

LDL^T 分解

所有非奇异对称矩阵A可被分解为

$$A = PLDL^TP^T$$

其中P 是置换矩阵, L 是下三角矩阵, D 是分块对角矩阵，其块的大小为1×1 或2×2

代价: $(1/3)n^3$

- 用LDL^T分解求解对称形式的线性方程，其代价为: $(1/3)n^3 + 2n^2 \approx (1/3)n^3$ （当n较大时）
- 对于稀疏矩阵A, 可以选择适当的P 来生成稀疏的L; 代价$\ll (1/3)n^3$

QR 分解

QR 分解又称正交分解. 对于“瘦高”形状的矩阵$A \in \mathbb{R}^{m\times n}$，其中$m \geq n$，QR 分解可以表示为

$$A = QR,$$

其中$Q \in \mathbb{R}^{m\times m}$为正交矩阵，$R \in \mathbb{R}^{m\times n}$是上三角矩阵，

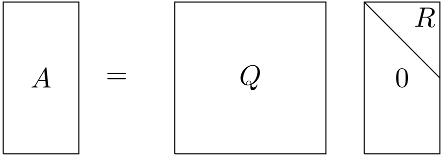


Figure: QR 分解

QR 分解

- R的下半部分都是零，当$m \gg n$时，这种分解方式极大地浪费存储空间
- Q的后$(m-n)$列是多余的,约化的QR 分解为

$$A = QR,$$

其中$Q \in \mathbb{R}^{m \times n}$ 满足$Q^{\mathrm{T}}Q = I$，$R \in \mathbb{R}^{n \times n}$ 为上三角矩阵，

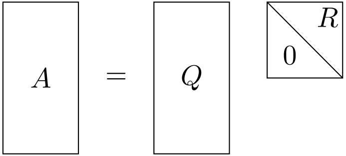


Figure: 约化的QR 分解

QR 分解

列满秩矩阵A，满足R对角元为正数的约化的QR分解是唯一的，有如下定理：

定理 (约化的QR分解的唯一性)

对于矩阵$A \in \mathbb{R}^{m\times n}(m \geq n)$，假设其列满秩，即$\mathrm{rank}(A) = n$，那么$A$有唯一的使得上三角矩阵$R$ 的对角元为正数（即$r_{ii} > 0$）的约化的QR 分解，

- QR分解可以通过Gram-Schmidt 正交化或Householder 三角化完成
- QR分解数值稳定性好，还可用于估计矩阵的秩和求解线性最小二乘问题.
- 计算代价大于LU分解，至少为$\mathcal{O}\left(\frac{4mn^2}{3}\right)$

提纲

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{1}$$

- 变量$x_1 \in \mathbb{R}^{n_1}$，$x_2 \in \mathbb{R}^{n_2}$；子矩阵$A_{ij} \in \mathbb{R}^{n_i \times n_j}$
- 若A_{11} 非奇异，可以消去x_1: $x_1 = A_{11}^{-1}(b_1 - A_{12}x_2)$;
  计算x_2，可以求解

$$(A_{22} - A_{21}A_{11}^{-1}A_{12})x_2 = b_2 - A_{21}A_{11}^{-1}b_1$$

用分块消去法求解线性方程

给定 一个非奇异的线性方程，并且A_{11}非奇异

1. 计算$A_{11}^{-1}A_{12}$ 和$A_{11}^{-1}b_1$.
2. 计算$S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ 和$\tilde{b} = b_2 - A_{21}A_{11}^{-1}b_1$.
3. 求解$Sx_2 = \tilde{b}$ 来确定x_2
4. 求解$A_{11}x_1 = b_1 - A_{12}x_2$ 来确定x_1

flop计数中的主要部分

- 第一步: $f + n_2 s$ (f 是分解A_{11}的代价; s 是求解的代价)
- 第二步: $2n_2^2 n_1$ (主要代价是A_{21} 和$A_{11}^{-1}A_{12}$的乘积)
- 第三步: $(2/3)n_2^3$

总代价: $f + n_2 s + 2n_2^2 n_1 + (2/3)n_2^3$

例子

- 对于一般的矩阵A_{11} ($f = (2/3)n_1^3$, $s = 2n_1^2$): 相比于标准方法没有任何改进

$$\#\text{flops} = (2/3)n_1^3 + 2n_1^2 n_2 + 2n_2^2 n_1 + (2/3)n_2^3 = (2/3)(n_1 + n_2)^3$$

- 当A_{11}具有结构时，分块消去法非常有用($f \ll n^3$)
  例如, 对角矩阵($f = 0$, $s = n_1$): $\#\text{flops} \approx 2n_2^2 n_1 + (2/3)n_2^3$

$$(A+BC)x = b$$

- $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}$, $C \in \mathbb{R}^{p\times n}$
- 假设A 有某种结构($Ax = b$ 解起来很方便)

首先将方程改写为

$$\begin{bmatrix} A & B \\ C & -I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

然后使用分块消去法：求解

$$(I + CA^{-1}B)y = CA^{-1}b,$$

然后求解$Ax = b - By$

该方法也证明了**SMW** 公式: 若A 和$A + BC$ 均非奇异，则

$$(A+BC)^{-1} = A^{-1} - A^{-1}B(I + CA^{-1}B)^{-1}CA^{-1}$$

例子: A 是对角矩阵, B, C 是稠密的

- 方法1: 计算$D = A + BC$, 然后求解$Dx = b$

  代价: $(2/3)n^3 + 2pn^2$

- 方法2 (借助SMW 公式): 求解

$$(I + CA^{-1}B)y = CA^{-1}b, \tag{2}$$

  然后计算$x = A^{-1}b - A^{-1}By$

  总代价由(2)决定: $2p^2n + (2/3)p^3$ (*i.e.*, 关于n是线性的)

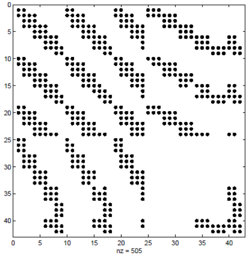
若$A \in \mathbb{R}^{p\times n}$，其中$p < n$, rank $A = p$,

$$\{x|Ax = b\} = \{Fz + \hat{x}|z \in \mathbb{R}^{n-p}\}$$

- $\hat{x}$ 是（任意）一个特解
- $F \in \mathbb{R}^{n\times(n-p)}$ 的列向量张成了A的零空间
- 有很多计算F的数值方法
  (QR 分解, rectangular LU分解, ...)

提纲

稀疏矩阵



- $A \in \mathbb{R}^{m\times n}$ 是稀疏的，如果它有“enough zeros that it pays to take advantage ot them” (J. Wilkinson)
- 通常意味着非零元个数n_{NZ}非常小: $n_{\mathrm{NZ}} \ll mn$

稀疏矩阵

稀疏矩阵可以节省时间和空间代价

- 使用双精度数存储$A \in \mathbb{R}^{m \times n}$
  - 稠密矩阵: $8mn$ 字节
  - 稀疏矩阵: $\approx 16n_{\mathrm{NZ}}$ 字节或更少, 取决于存储格式
- 计算$y \leftarrow y + Ax$
  - 稠密矩阵: mn flops
  - 稀疏矩阵: n_{NZ} flops
- 计算$x \leftarrow T^{-1}x, T \in \mathbb{R}^{n \times n}$ 是非奇异的三角矩阵:
  - 稠密矩阵: $n^2/2$ flops
  - 稀疏矩阵: n_{NZ} flops

稀疏矩阵的表示法

- 有很多种方法在使用
- 最简单(但是并不经常使用)的方法是将数据存储在一个列表中，列表的元素是(i, j, A_{ij})三元组。
- 列压缩格式（column compressed format）：将所有(A_{ij}, i)对按顺序放在一个数组中，再用指针记录每一列开始的位置，并将这些位置指针放在一个数组中
- 在高层次的工作中，也会使用其他的数据结构
- 可惜的是，目前没有一个统一的标准

稀疏BLAS?

然而现在并没有一个标准的稀疏矩阵BLAS库

- “官方”稀疏*BLAS*

  `http://www.netlib.org/blas/blast-forum`

  `http://math.nist.gov/spblas`

- C++: Boost uBlas, Matrix Template Library, SparseLib++
- 因特尔的数学核心函数库：MKL，支持串行和并行计算.
- Pyhton: SciPy, PySparse, CVXOPT

稀疏矩阵的分解

求解或分解稀疏矩阵系统的函数库

- 最容易理解的: SuiteSparse (Tim Davis)

  `http://www.cise.ufl.edu.research/sparse/SuiteSparse`

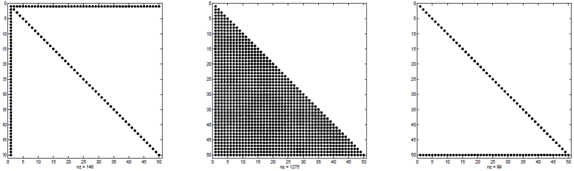
- 其他还有SuperLU, TAUCS, SPOOLES

- 通常包括

  -$A = PLL^TP^T$ Cholesky
  -$A = PLDL^TP^T$ 对于对称不定系统
  -$A = P_1LUP_2^T$ 对于一般的（非对称）矩阵
  P, P_1, P_2 是置换矩阵（permutations）或排列矩阵（*orderings*）

稀疏矩阵的排列

重新排列稀疏矩阵，会对因子矩阵的稀疏性有极其重要的影响



- 左图: 原版NW箭形矩阵的spy图像
- 中图: 不重排，进行Cholesky分解，因子矩阵的spy图像$(P = I)$
- 右图: 选择最优的排列方式，进行Cholesky分解，因子矩阵的spy图像(交换$1 \to n$)
- 选择产生最稀疏分解的排序的一般问题是很难的。
- 但是，一些简单的启发式方法非常有效
- 还有更多的排列方法，例如nested disection 也非常有效

符号分解

- 对于Cholesky分解，重排的选择仅基于A的稀疏格式，与具体值无关
- 分解可以被划分为两步: 符号 分解和数值 分解
  - 在求解多个线性系统时，如果他们有相同的稀疏格式，那么符号分解可以只做一次
  - 可以在选择重排方式时投入更多，因为在多次数值分解中，符号分解的代价会被摊销
- LDL^T分解的重排通常只能听天由命/按部就班/特事特办/根据数据（has to be done on the fly）

提纲

特征值分解

- 对于矩阵$A \in \mathbb{R}^{n\times n}$，若存在非奇异矩阵$X \in \mathbb{R}^{n\times n}$和对角矩阵$\Sigma \in \mathbb{R}^{n\times n}$，使得

$$A = X\Sigma X^{-1}.$$

  则称A可对角化，并称上式为A 的特征值分解.

- 矩阵Σ的对角元为A的特征值，矩阵X的每一列为其对应的特征向量.

很多优化问题往往是实对称矩阵. 实对称矩阵$A \in \mathcal{S}^n$的特征值分解有非常好的性质：

1. 实对称矩阵的所有特征值均为实数；
2. 实对称矩阵对应于不同特征值的特征向量是相互正交的；
3. 实对称矩阵可以正交对角化，即存在正交矩阵U，使得$U^{\mathrm{T}}AU$为对角矩阵.

实对称矩阵的特征值分解

- 对于$A \in \mathcal{S}^n$，有分解
$$A = U\Lambda U^{\mathrm{T}},$$
其中$U \in \mathbb{R}^{n\times n}$ 为正交矩阵，$\Lambda = \mathrm{Diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ 为对角矩阵并且λ_i，$i = 1, 2, \cdots, n$对应于A的特征值.

- 记$U = [u_1, u_2, \cdots, u_n]$，那么$u_i$ 为特征值λ_i 对应的特征向量，我们还可以将A写成如下秩1矩阵的和的形式（又称为外积形式）
$$A = \sum_{i=1}^{n} \lambda_i u_i u_i^{\mathrm{T}}.$$

- 特征值的计算有幂法、反幂法、QR方法、Jacobi方法、二分法等

SVD - Properties

Theorem: SVD

If A is a real m-by-n matrix, then there exits

$$U = [u_1, \ldots, u_m] \in \mathbb{R}^{m\times m} \text{ and } V = [v_1, \ldots, v_n] \in \mathbb{R}^{n\times n}$$

such that $U^T U = I$, $V^T V = I$ and

$$U^T AV = \mathrm{diag}(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m\times n}, \quad p = \min(m, n),$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$.

- Proof: Let $V_1 \in \mathbb{R}^{n\times r}$ has orthonormal columns, then exits $V_2 \in \mathbb{R}^{n\times(n-r)}$ such that $V = [V_1, V_2]$ is orthogonal.
- Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ be unit 2-norm vectors: $Ax = \sigma y$ with $\sigma = \|A\|_2$. Then exists $V_2 \in \mathbb{R}^{n\times(n-1)}$ and $U_2 \in \mathbb{R}^{m\times(m-1)}$ so $V = [x, V_2] \in \mathbb{R}^{n\times n}$ and $U = [y, U_2] \in \mathbb{R}^{m\times m}$ are orthogonal.

- Then it can be proved that $U^T AV$ has the following structure

$$U^T AV = \begin{pmatrix} \sigma & w^T \\ 0 & B \end{pmatrix} \equiv A_1.$$

Since

$$\left\| A_1 \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\|_2^2 \geq (\sigma^2 + w^T w)^2,$$

we have $\|A_1\|_2^2 \geq (\sigma^2 + w^T w)$. But $\sigma^2 = \|A\|_2^2 = \|A_1\|_2^2$, and so we must have $w = 0$. An induction gives the proof.

Properties:

- $AV = U\Sigma$, $A^T U = V\Sigma^T$: $Av_i = \sigma u_i$, $A^T u_i = \sigma_i v_i$, $i = 1, \ldots, p$.
- $rank(A) = r$, $null(A) = span\{v_{r+1}, \ldots, v_n\}$, $ran(A) = span\{u_1, \ldots, u_r\}$
- $A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$
- $\|A\|_F^2 = \sigma_1^2 + \ldots + \sigma_p^2$, $\|A\|_2 = \sigma_1$

SVD - Best Low Rank Approximation

Theorem

Let the SVD of $A \in \mathbb{R}^{m \times n}$ be given in Theorem: SVD. If $k < r = rank(A)$ and $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$, then

$$\min_{rank(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}.$$

- Proof: Since $U^T A_k V = \mathrm{diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)$ it follows that $rank(A_k) = k$ and $U^T(A - A_k)V = \mathrm{diag}(0, \ldots, 0, \sigma_{k+1}, \ldots, \sigma_p)$. Hence $\|A - A_k\|_2 = \sigma_{k+1}$.
- Suppose $rank(B) = k$ for some $B \in \mathbb{R}^{m \times n}$. We can find orthonormal vectors $x_1, \ldots, x_{n-k}$ so $null(B) = span\{x_1, \ldots, x_{n-k}\}$. A dimension argument shows:

$$span\{x_1, \ldots, x_{n-k}\} \cap span\{v_1, \ldots, v_{k+1}\} \neq \{0\}$$

- Let z be a unit 2-norm vector in this intersection. Since $Bz = 0$ and

$$Az = \sum_{i=1}^{k+1} \sigma_i (v_i^T z) u_i,$$

we have

$$\|A - B\|_2^2 \geq \|(A - B)z\|_2^2 = \|Az\|_2^2 = \sum_{i=1}^{k+1} \sigma_i^2 (v_i^T z)^2 \geq \sigma_{k+1}^2$$

Comments:

- So zeroing small σ_i introduces less error
- How many σs to keep? Rule of thumb: keep 80-90% of 'energy' ($= \sum \sigma_i^2$)

奇异值分解

- 从定义得：$A^{\mathrm{T}}A = V\Sigma^{\mathrm{T}}\Sigma V^{\mathrm{T}}$，即$A^{\mathrm{T}}A$与$\Sigma^{\mathrm{T}}\Sigma$相似，求$A$的奇异值，等价于求$A^{\mathrm{T}}A$的特征值．
- 流程如下：
  1. 计算$A^{\mathrm{T}}A$；
  2. 计算$A^{\mathrm{T}}A$的特征值分解：$A^{\mathrm{T}}A = V\Lambda V^{\mathrm{T}}$；
  3. 对Λ所有对角元素开根号：$\Sigma = \sqrt{\Lambda}$；
  4. 求解正交矩阵U,使得$U\begin{bmatrix}\Sigma \\ 0\end{bmatrix} = AV$（可以利用QR分解）．
- 因为$A^{\mathrm{T}}A$损失了原矩阵A的部分信息，该方法可能数值不稳定.
- 优点是，当$n \ll m$时，仅涉及n阶小方阵的特征值分解

奇异值分解

- 另一种方法将这一问题转化为另一个矩阵的特征值分解.
- 对于矩阵A及其奇异值分解$A = U\Sigma V^{\mathrm{T}}$，构造矩阵

$$H = \begin{bmatrix} 0 & A^{\mathrm{T}} \\ A & 0 \end{bmatrix}, \quad X = \begin{bmatrix} V & V \\ U & -U \end{bmatrix},$$

那么

$$HX = X \begin{bmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{bmatrix}.$$

- 将A的奇异值分解问题转化为H的特征值分解问题.
- 该方法数值稳定，但增大了问题维数，需要更大的存储量和迭代次数.

计算主要特征（奇异）值和特征（奇异）向量

特征值对的计算

- ARPACK (eigs in matlab)
- LOBPCG
- Arrabit
- SLEPc

奇异值对的计算

- PROPACK, a good implementation is lansvd in `http://www.math.nus.edu.sg/~mattohkc/NNLS.html`
- LMSVD with warm-starting: `https://ww2.mathworks.cn/matlabcentral/fileexchange/46875-lmsvd-m`