LU 分解 三角方程组的解法

$$A \to L - I + U$$

```
for j = 1:n
    A(j+1:n,j) = A(j+1:n,j) / A(j,j);
    A(j+1:n,j+1:n) -= A(j+1:n,j) * A(j,j+1:n);
end
```

时间复杂度：

$$\sum_{j=1}^{n-1}[n-j+2(n-j)^2] = \frac{2}{3}n^3 + o(n^3) \to \ O(n^3)$$

LU 分解 三角方程组的解法 $Ux = b$

```
for j=n:-1:1
    b(j) -= u(j,j+1:n)*b(j+1:n);
    b(j) /= u(j,j);
end
```

Basic Linear Algebra Subprograms(BLAS)

层级	意义	时间复杂度	案例		
1	Vector – vector	$O(n)$	$\alpha \leftarrow y^T x$	$y \leftarrow x\alpha + y$	$\|\|x\|\|$
2	Matrix – vector	$O(n^2)$	$y \leftarrow Ax + y$	$A \leftarrow A + xy^T$	$x \leftarrow U^{-1}x$
3	Matrix – matrix	$O(n^3)$	$C \leftarrow AB + C$	$A \leftarrow A + XY^T$	$X \leftarrow U^{-1}X$

操作	运算耗时($\boldsymbol{OP}$)	读写耗时($\boldsymbol{IO}$)	效率$\frac{\boldsymbol{OP}}{\boldsymbol{IO}}$
$\boldsymbol{y \leftarrow x\alpha + y}$	$2n$	$3n$	$\frac{2}{3}$
$\boldsymbol{y \leftarrow Ax + y}$	$2n^2$	n^2	2
$\boldsymbol{C \leftarrow AB + C}$	$2n^3$	$3n^2$	$\frac{2}{3}n$

- 优化方向：提高缓存的利用率。

高斯消元

$$PA = LU$$

Step 1 进行行/列变换，使得矩阵A的前m行满秩(即得到矩阵PA)

- **选主元法：全主元/列主元**

通过行列变换，使得每行主元为绝对值最大的元素。

优势：选取过小的主元将造成之后的行在进行消元的时候加上一个过大数量级的数，选主元法可以有效避免这一缺陷带来的精度误差。（避免主元$|a_{j,j}| < eps.$）

Step 2 作LU分解，求解。

形式化地，分为三步（1）$A = LU$；（2）$Ly = b$；（3）$Ux = y.$

选主元方法	全主元	列主元
效率		

[高斯消元作列变换]

说明：P为行变换矩阵；Q为列变换矩阵。

$$PAQ(Q^{-1}x) = Pb$$

记$y = Q^{-1}x$，则

$$LUy = Pb$$

解得

$$y = U^{-1}(L^{-1}Pb)$$

即

$$x = Qy = QU^{-1}(L^{-1}Pb)$$

[性质]

若方程$Ax = b$可以化为$A_1A_2 \dots A_kx = b$，其中A_i均为容易求逆的矩阵，则原方程的解是容易求的。

容易求逆的矩阵有：

(1) 行列变换阵P, Q

(2) 酉阵$Q^*Q = I \Longrightarrow Q^{-1} = Q^*$

舍入误差分析

定义. 实数x的浮点数表示为$fl(x)$, 满足

$$fl(x) = x(1+\delta), |\delta| \leq \boldsymbol{u}$$

其中$\boldsymbol{u}$是机器精度, 满足

$$\boldsymbol{u} = \begin{cases} \frac{1}{2}\beta^{1-t}, \text{用舍入法} \\ \beta^{1-t}, \text{用截断法} \end{cases}.$$

定理. 若$|\delta_i| \leq \boldsymbol{u}$且$n\boldsymbol{u} \leq 0.01$, 则有

$$1 - n\boldsymbol{u} \leq \prod_{i=1}^{n}(1+\delta_i) \leq 1 + 1.01n\boldsymbol{u}$$

证明. 采用 Taylor 展开，利用$1 + x \leq e^x \leq 1 + x + \frac{0.01}{2}xe^x \leq 1 + 1.01x$证明. 证明的详细过程详见 P60.

例题. 设x, y是两个由浮点数构成的n维向量. 试估计$|fl(x^T y) - x^T y|$的上界.

解. 令

$$S_k = fl\left(\sum_{i=1}^{k} x_i y_i\right)$$

则由定义得

$$S_1 = x_1 y_1(1+\gamma_1), |\gamma_1| \leq \boldsymbol{u}$$

$$S_k = fl[S_{k-1} + fl(x_k y_k)] = [S_{k-1} + x_k y_k(1+\gamma_k)](1+\delta_k), |\delta_k|, |\gamma_k| \leq \boldsymbol{u}$$

于是

$$fl(x^T y) = S_n = \sum_{i=1}^{n} x_i y_i(1+\gamma_i)\prod_{j=i}^{n}(1+\delta_j) = \sum_{i=1}^{n}(1+\varepsilon_i)x_i y_i$$

其中

$$1 + \varepsilon_i = (1+\gamma_i)\prod_{j=i}^{n}(1+\delta_j)$$

令$\delta_1 = 0$, 即有

$$|fl(x^T y) - x^T y| \leq \sum_{i=1}^{n}|\varepsilon_i||x_i y_i| \leq 1.01n\boldsymbol{u}\sum_{i=1}^{n}|x_i y_i|$$

向前误差分析（forward error）

定义. 若$\frac{\|f(x+\Delta x)-f(x)\|}{\|f(x)\|}$(或$\frac{\|fl(y)-y\|}{\|y\|}$)较小,则称该问题为良态问题；否则称为病态问题（可用于加密）.

推论. 计算连续函数$f(x)$的问题大概率是良态问题；计算不连续函数$f(x)$的问题大概率为病态问题.

例题. 设

$$\beta=\sum_{i=1}^{n}\alpha_i$$

则

$$\frac{|fl(\beta)-\beta|}{\sum_{i=1}^{n}|\alpha_i|}\le\gamma_{n-1}$$

其中γ可由算法控制，但$\frac{\sum_{i=1}^{n}|\alpha_i|}{|\beta|}$取决于问题是否病态.

这表明$\frac{|fl(\beta)-\beta|}{\sum_{i=1}^{n}|\alpha_i|}$与$\frac{|fl(\beta)-\beta|}{|\beta|}$不一定相等. 故计算相对误差时应为

$$\frac{|fl(\beta)-\beta|}{|\beta|}\le\frac{\sum_{i=1}^{n}|\alpha_i|}{|\beta|}\gamma_{n-1}=\kappa\gamma_{n-1}$$

这被称为**向前误差分析**(即分析因变量误差).

解线性方程组的误差

在$\|\Delta A\|<\|A^{-1}\|^{-1}$的前提下(即扰动$\Delta A$较小时),考虑问题

$$(A+\Delta A)(x+\Delta x)=b+\Delta b$$

由于$I+A^{-1}\Delta A$是非奇异的,故有

$$\Delta x=(I+A^{-1}\Delta A)^{-1}A^{-1}(\Delta b-\Delta Ax)$$

由$(I+A^{-1}\Delta A)^{-1}$的级数展开

$$(I+A^{-1}\Delta A)^{-1}=\sum_{i=0}^{\infty}(-1)^i(A^{-1}\Delta A)^i$$

得到

$$\|(I+A^{-1}\Delta A)^{-1}\|\le\sum_{i=0}^{\infty}\|A^{-1}\Delta A\|^i\le\frac{1}{1-\|A^{-1}\|\|\Delta A\|}$$

于是绝对误差

$$\|\Delta x\|\le\frac{\|A^{-1}\|}{1-\|A^{-1}\|\|\Delta A\|}(\|\Delta b\|+\|\Delta A\|\|x\|)$$

相对误差

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

故当$\|A\|$确定时, $\|A^{-1}\|$越大, 问题越病态. （这里也可以看出条件数$\kappa(A)=\|A^{-1}\|\|A\|$越大, 问题越病态）

定义. 条件数$\kappa(A)=\|A\|\|A^{-1}\|$. 这一数值是下面两式的上界.

绝对条件数$\kappa_a=\overline{\lim_{\Delta x\to 0}}\frac{\|f(x+\Delta x)-f(x)\|}{\|\Delta x\|}$.

相对条件数$\kappa_r=\kappa_a\frac{\|x\|}{\|f(x)\|}$.

推论. 若A是奇异阵, 则$\kappa(A)=+\infty$.

推论. 对A做奇异值分解

$$A=U\Sigma V^*=\sum_{i=1}^{n}u_i\sigma_i v_i^*$$

则

$$\kappa_2(A)=\|A\|_2\|A^{-1}\|_2=\frac{\sigma_1}{\sigma_n}$$

向后误差分析(backward error)

现考虑问题

$$\hat{y}=fl[f(x)]=f(x+\Delta x)$$

于是

$$\|\Delta y\|\leq\kappa_a\|\Delta x\|$$
$$\frac{\|\Delta y\|}{\|y\|}\leq\kappa_r\frac{\|\Delta x\|}{\|x\|}$$

这一操作将向前误差分析转变为**向后误差分析**(即分析自变量误差).

例题. 设$\beta=y^Tx$, 对$f(\beta)$做向后误差分析.

解. $\Delta\beta=(y+\Delta y)^T(x+\Delta x)-y^Tx=\Delta y^Tx+y^T\Delta x+\Delta y^T\Delta x$

于是

$$\left|\frac{\Delta\beta}{\beta}\right|\leq\left|\frac{(\Delta y^Tx+y^T\Delta x+\Delta y^T\Delta x)}{y^Tx}\right|\leq\frac{\left\|\begin{bmatrix}\Delta y\\ \Delta x\end{bmatrix}\right\|\left\|\begin{bmatrix}x\\ y\end{bmatrix}\right\|}{\left\|\begin{bmatrix}x\\ y\end{bmatrix}\right\|\left\|\begin{bmatrix}y\\ x\end{bmatrix}\right\|}\cdot\frac{\left\|\begin{bmatrix}x\\ y\end{bmatrix}\right\|\left\|\begin{bmatrix}y\\ x\end{bmatrix}\right\|}{y^Tx}$$

即

$$\left|\frac{\Delta\beta}{\beta}\right| \leq \frac{\|x\|^2+\|y\|^2}{y^Tx}\cdot\frac{\left\|\begin{bmatrix}\Delta y\\ \Delta x\end{bmatrix}\right\|}{\left\|\begin{bmatrix}y\\ x\end{bmatrix}\right\|}$$

与

$$\frac{\|\Delta y\|}{\|y\|} \leq \kappa_r \frac{\|\Delta x\|}{\|x\|}$$

对比, 有

$$\kappa_r = \frac{\|x\|^2+\|y\|^2}{y^Tx}$$

且

$$fl[f(\beta)] = \sum_{i=1}^{n} x_i y_i \prod_{j=1}^{n+1-i}(1+\theta_{ij}) = \hat{y}^T x = (y+\Delta y)^T x$$

例题. 解下三角方程组$Lx=b$, 即

$$\begin{bmatrix}L_{11} & 0\\ L_{21} & L_{22}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} = \begin{bmatrix}b_1\\ b_2\end{bmatrix}$$

解.

$$x_1 = \frac{b_1}{L_{11}} \Rightarrow \hat{x}_1 = \frac{b_1}{L_{11}}(1+\gamma_1) = \frac{b_1}{\frac{L_{11}}{1+\gamma_1}}$$

$$x_2 = \frac{b_2 - L_{21}x_1}{L_{22}} \Rightarrow \hat{x}_2 = \frac{\left(b_2 - \hat{L}_{21}\hat{x}_1\right)(1+\delta_2)}{L_{22}}(1+\gamma_2) = \frac{b_2 - \hat{L}_{21}\hat{x}_1}{\frac{L_{22}}{(1+\delta_2)(1+\gamma_2)}}$$

则

$$\hat{L} = \begin{bmatrix}\frac{L_{11}}{1+\gamma_1} & 0\\ \frac{L_{21}}{1+\gamma_{21}} & \frac{L_{22}}{(1+\gamma_2)(1+\delta_2)}\end{bmatrix}$$

例题. 解线性方程组$Ax=b$.

解. 设$A=LU$, 则有

$$\left(L+\hat{L}\right)\left(U+\hat{U}\right)\hat{x} = b$$

也即

$$\left(A+\Delta A+\Delta L\hat{U}+L\Delta\hat{U}+\Delta\hat{L}\Delta\hat{U}\right)\hat{x} = b$$

令$\Delta A = \Delta A+\Delta L\hat{U}+L\Delta\hat{U}+\Delta\hat{L}\Delta\hat{U}$, 则

$$|\Delta A| \leq \left|\hat{L}\right|\left|\hat{U}\right|(\gamma_n+\gamma_n+\gamma_n+\gamma_n^2) = \gamma_n(3+\gamma_n)\left|\hat{L}\right|\left|\hat{U}\right| \leq \gamma_{3n}\left|\hat{L}\right|\left|\hat{U}\right|$$

从而

$$\|\Delta A\|_\infty \le \gamma_{3n}\|\hat{L}\|_\infty\|\hat{U}\|_\infty$$

其中控制$\|\hat{L}\|_\infty \le n$. 由于$\|\hat{U}\|_\infty = O(2^n)$, 控制$\|\hat{U}\|_\infty$可以通过计算增长因子动态控制.

定义 **增长因子**$\rho = \frac{\|\hat{U}\|_\infty}{\|A\|_\infty}$. 这是一个后验量, 用于动态监控$\|\hat{U}\|_\infty$.

[tip] 全选主元/列选主元的优化方法均为了降低ρ的值, 进而提高算法的数值稳定性.

定义 刻画算法受舍入误差影响大小的量为**数值稳定性**. 一般情况下, 采用$\frac{\|\delta f(x)\|}{\|f(x)\|}$作为数值稳定性的量化值.

一些定义

概念	定义
正交阵	满足$A^TA=I$的矩阵A
酉阵	满足$A^*A=I$的矩阵A
正定阵	$\forall x$,满足$x^TAx>0$的矩阵A
对称阵	满足$A^T=A$的矩阵A
Hermite 矩阵	满足$A^*=A$的矩阵A
正规矩阵	满足$A^*A=AA^*$的矩阵A
幂等矩阵	满足$A^2=A$的矩阵A

Cholesky 分解

对于对称正定阵A,存在对角元均为正数的下三角阵L,使得

$$A=LL^*$$

对于一般的$Ax=b$,可以作变换

$$A^*Ax=A^*b$$

从而可以转化为 Cholesky 分解的适用形式.(具体算法参见**作业 2-题目 2-算法 1**)

对于 Cholesky 分解,其效率为

$$O\left(\frac{1}{3}n^3\right)$$

其增长因子为

$$\rho=\frac{\|L\|\|L^*\|}{\|A\|}=\frac{\|L\|_2^2}{\|L^*L\|_2}=1$$

其数值稳定性为

$$\kappa_2(A^*A)=\left(\kappa_2(A)\right)^2$$

故当A不是对称正定阵时,强行使用 Cholesky 分解法在数值稳定性上并不占优,大多数情况下选用 LU 分解更优.

但当A是对称正定阵时,Cholesky 分解相比 LU 分解具有更高的效率和更优的数值稳定性.

改进 Cholesky 分解(LDL^T分解)

具体算法参加**作业 2-题目 2-算法 2.**

QR 分解

由于LU分解无法保证条件数不变,故考虑使用正交分解以维持原矩阵条件数不变.现希望得到$A=QR$,其中$Q^*Q=I$.则考虑$Ax=b$,只需得$QRx=b$,即

$$Rx=Q^*b$$

由 Cholesky 分解,得$A^*A=R^*R$且$A^*A=LL^*$.

由于$Q=AR^{-1}$,故

$$Q^*Q=R^{-*}A^*AR^{-1}=(A^*A)(R^*R)^{-1}=I$$

于是Q是酉阵,从而A存在QR分解.

算法 1 Givens(1958)——旋转置零

核心思想:选择酉矩阵T_{ij},把列向量的*子向量* 通过旋转变换到基向量上,从而达到置零元素的目的.例如

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{G(3,4)} \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{G(2,3)} \begin{bmatrix} x & x & x \\ x & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{G(1,2)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix}$$

$$\xrightarrow{G(3,4)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{G(2,3)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \end{bmatrix} \xrightarrow{G(3,4)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \end{bmatrix} = R$$

更形式化地,我们最终将得到(以下令$T_{p,q}=G(p,q)$)

$$\prod_{i=1}^{n-1}\prod_{j=i+1}^{n} T_{ij}\,A=R$$

下面考虑构造每一个旋转矩阵$T_{p,q}$.特殊地,对于2×1的向量$a=\begin{pmatrix}4\\2\end{pmatrix}$,我们希望将其转化为$a'=\begin{pmatrix}r\\0\end{pmatrix}$.这一变换可被表示为

$$a'=\begin{pmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{pmatrix}a$$

其中,$\cos\theta=\frac{4}{\sqrt{4^2+2^2}}$,$\sin\theta=\frac{2}{\sqrt{4^2+2^2}}$.

更一般地,旋转矩阵T_{pq}的形式为

$$
\boldsymbol{T}_{pq}=\begin{pmatrix}
1 & & & & & & & & \\
& \ddots & & & & & & & \\
& & 1 & & & & & & \\
& & & \bar{c} & & \bar{s} & & & \\
& & & & 1 & & & & \\
& & & & & \ddots & & & \\
& & & & & & 1 & & \\
& & & -s & & c & & & \\
& & & & & & & 1 & \\
& & & & & & & & \ddots \\
& & & & & & & & & 1
\end{pmatrix}
\begin{matrix}
\\ \\ \\ p\text{行} \\ \\ \\ \\ q\text{行} \\ \\ \\
\end{matrix}
$$

$$
\begin{matrix}
\\ \\ \\ p\text{列} & & q\text{列} & & \\ \\ \\ \\ \\
\end{matrix}
$$

其中,$c=\cos\theta_{pq}=\frac{a_{qp}}{\sqrt{a_{qp}^2+a_{pp}^2}}, s=\sin\theta_{pq}=\frac{a_{pp}}{\sqrt{a_{qp}^2+a_{pp}^2}}$.

Givens 变换尤其利好于稀疏矩阵的 QR 分解,这是因为每一步 Givens 变换仅影响矩阵每列中的两个元素(**注意：**这意味着每一步 Givens 变换均可能引发对其它列的污染,见 hw4:T5).

$$
A([i,k],:)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T A([i,k],:)
$$

每一步变换的耗时为$6n$.算法的总时间复杂度为

$$
T(n)=\frac{4}{3}n^3+O(n^3)
$$

算法 2 Householder (1958)——镜面反射

核心思想：选择酉矩阵Q_k, 使得第k列对角以下引入零元素, 且不保持先前引入的零元素不变. 此时得到的$Q_n \dots Q_2 Q_1 A = R$为上三角阵. 例如

$$\underset{A}{\begin{bmatrix} X & X & X \\ X & X & X \\ X & X & X \\ X & X & X \\ X & X & X \end{bmatrix}} \overset{Q_1}{\rightarrow} \underset{Q_1 A}{\begin{bmatrix} X & X & X \\ 0 & X & X \\ 0 & X & X \\ 0 & X & X \\ 0 & X & X \end{bmatrix}} \overset{Q_2}{\rightarrow} \underset{Q_1 Q_2 A}{\begin{bmatrix} X & X & X \\ 0 & X & X \\ 0 & 0 & X \\ 0 & 0 & X \\ 0 & 0 & X \end{bmatrix}} \overset{Q_3}{\rightarrow} \underset{Q_1 Q_2 Q_3 A}{\begin{bmatrix} X & X & X \\ 0 & X & X \\ 0 & 0 & X \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}$$

具体来说, 每个Q_k的形式均形如

$$Q_k = \begin{bmatrix} I & 0 \\ 0 & F \end{bmatrix}$$

其中I为$(k-1)\times(k-1)$的单位阵, F为$(n-k+1)\times(n-k+1)$的酉阵. 在此算法中, 称F为 Householder reflector. 令$x = \begin{bmatrix} x_k \\ x_{k+1} \\ x_{k+2} \\ \vdots \\ x_n \end{bmatrix}_{(n-k+1)\times 1}$, 则$F$应当满足

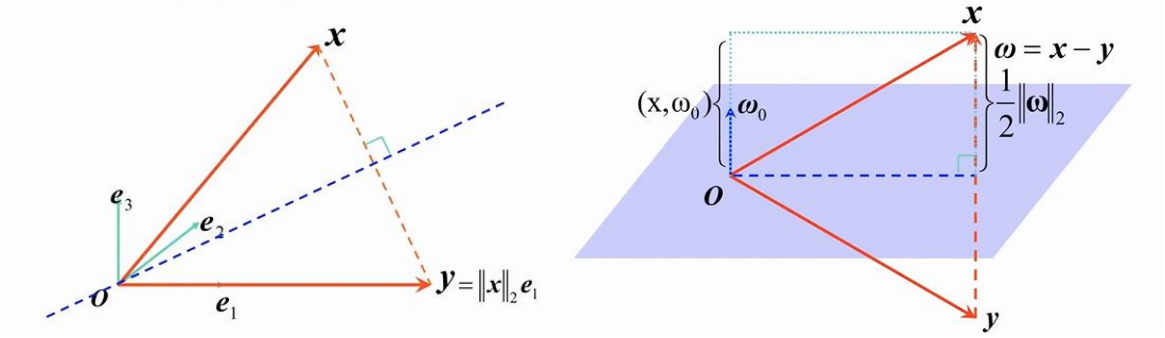
$$F_{(n-k+1)\times(n-k+1)} \begin{bmatrix} x_k \\ x_{k+1} \\ x_{k+2} \\ \vdots \\ x_n \end{bmatrix}_{(n-k+1)\times 1} = \begin{bmatrix} \|x\|_2 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{(n-k+1)\times 1} = \|x\|_2 e_1$$

这实际上是将x映射到$\|x\|_2 e_1$上.

对于任意向量x, 将其投影到平面H的投影为(其中P被称为投影算子)

$$Px = \left(I - \frac{vv^*}{v^*v}\right)x$$

而 Householder 算法实际上在做镜面反射(因为需要保持投影向量的模和原向量相同), 即如图



满足的式子包括

$$\begin{cases} \omega_0 = \dfrac{\omega}{\|\omega\|_2} \\ \omega = x - y \\ \|\omega\|_2 = 2\langle x, \omega_0 \rangle = 2\omega_0^* x \end{cases}$$

从而得到

$$x - y = 2(\omega_0^* x)\omega_0 = 2\omega_0(\omega_0^* x) = \frac{2\omega(\omega^* x)}{\|\omega\|_2^2} = \frac{2\omega\omega^*}{\omega^*\omega}x$$

因此

$$y = \left(I - \frac{2\omega\omega^*}{\omega^*\omega}\right)x$$

其中, $\omega = x + sgn(x_k)\|x\|_2 e_1$. 为避免$x$过于接近$e_1$时产生有效数字的损失, 作等价变形

$$\omega = \frac{x^2 - \|x\|_2^2}{x + \|x\|_2} = \frac{-1}{x + \|x\|_2}\sum_{i=k+1}^{n} x_i^2$$

对于Q_k, 时间复杂度为$8(n-k+1)^2$. 从而该算法的总时间复杂度为

$$T(n) = \sum_{i=1}^{n-1} 8i^2 = \frac{8}{3}n^3 + O(n^3)$$

算法的实现代码为

Algorithm (Householder QR)

for $k = 1 : n$ (sum over columns)

$\boldsymbol{x} = A(k:m, k)$

$\boldsymbol{v}_k = \boldsymbol{x} + \text{sign}(\boldsymbol{x}(1))\|\boldsymbol{x}\|_2 \boldsymbol{e}_1$

$\boldsymbol{v}_k = \boldsymbol{v}_k / \|\boldsymbol{v}_k\|_2$

$A(k:m, k:n) = A(k:m, k:n) - 2\boldsymbol{v}_k\,(\boldsymbol{v}_k^* A(k:m, k:n))$

end

定义. **镜射向量**$\omega_0 = \frac{\omega}{\|\omega\|_2}$.

定义. Householder **矩阵**$H(\omega) = I - \frac{2\omega\omega^*}{\omega^*\omega} = I - 2\omega_0\omega_0^*$. 上面的$F$即为$H(\omega)$.

性质. （1）对称性 $H(\omega)^T = H(\omega)$

（2）正交性 $H(\omega)^{-1} = H(\omega)^T$

（3）酉矩阵 $H(\omega)^{-1} = H(\omega)^*$

（4）模不变 $\|H(\omega)x\|_2 = \|x\|_2$

（5）$H(\omega)$的特征值为$n-1$个1和一个-1, 即$\det\big(H(\omega)\big) = -1$.

算法 3 Classical Gram-Schmidt 变换(CGS)

核心思想：逐列正交归一化.

对于任意矩阵A,令其列向量展开为$A=[x_1,x_2,\dots,x_n]$,则可通过 Gram-Schmidt 变换方法获得其一组正交基$[v_1,v_2,\dots,v_n]$.具体来说,即将x_j减去其在前$j-1$个基向量上的分量作为第j个基向量,依次类推即得一组正交基.形式化地,有

$$v_j=x_j-\sum_{i=1}^{j-1}\frac{v_i^*x_j}{\|v_i\|_2^2}v_i$$

并规定$v_1=x_1$.进一步做归一化处理,即得

$$u_j=\frac{v_j}{\|v_j\|_2}$$

代回前式得

$$\boxed{v_j=x_j-\sum_{i=1}^{j-1}(u_i^*x_j)u_i}$$

从而可以反解得

$$x_j=\|v_j\|_2u_j+\sum_{i=1}^{j-1}(u_i^*x_j)u_i$$

故可得A的QR分解为

$$A=[x_1,x_2,\dots,x_n]=[u_1,u_2,\dots,u_n]\begin{bmatrix}\|v_1\|_2 & u_1^*x_2 & \cdots & u_1^*x_n\\ 0 & \|v_2\|_2 & \cdots & u_2^*x_n\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \|v_n\|_2\end{bmatrix}=QR$$

算法的实现代码为

Algorithm (Classical Gram-Schmidt)

for $j=1:n$

$\boldsymbol{v}_j=\boldsymbol{a}_j$

for $i=1:(j-1)$

$r_{ij}=\boldsymbol{q}_i^*\boldsymbol{a}_j$

$\boldsymbol{v}_j=\boldsymbol{v}_j-r_{ij}\boldsymbol{q}_i$

end

$r_{jj}=\|\boldsymbol{v}_j\|_2$

$\boldsymbol{q}_j=\boldsymbol{v}_j/r_{jj}$

end

算法 4 Modified Gram-Schmidt 变换(MGS)

对算法 3 中带框的式子做变形,得

$$v_j = x_j - x_j \sum_{i=1}^{j-1} (u_i u_i^*)$$

令正交投影算子$P_j = I - Q_{j-1}Q_{j-1}^*$,其中$Q_{j-1} = [u_1, u_2, \ldots, u_{j-1}]$,则

$$v_j = P_j x_j$$

在 CGS 中,每一步投影过程就相当于做一次正交投影$P_j = I - Q_{j-1}Q_{j-1}^*$.然而在 MGS 中,由于$u_i, u_j$正交,每一步投影过程相当于做一次正交投影算子$P_j$的等价定义(此结论可采用数学归纳法证明)

$$P_j = I - Q_{j-1}Q_{j-1}^* = I - \sum_{i=1}^{j-1} u_i u_i^* = \prod_{i=1}^{j-1} (I - u_i u_i^*)$$

两种算法的投影过程差异如下

- CGS 的第 k 步
  (a) 计算 $r_{ik} = \langle a_k, q_i \rangle, \quad i = 1, \ldots, k-1.$
  (b) 计算 $\tilde{q}_k = a_k - r_{1k}q_1 - r_{2k}q_2 - \cdots - r_{k-1,k}q_{k-1}.$
  (c) 计算 $r_{kk} = \|\tilde{q}_k\|, \quad q_k = \tilde{q}_k / r_{kk}.$
- MGS 的第 k 步
  (a) 计算 $\tilde{q}_k = a_k.$
  (b) 计算 $r_{ik} = \langle \tilde{q}_k, q_i \rangle, \quad \tilde{q}_k = \tilde{q}_k - r_{ik}q_i, \quad i = 1, \ldots, k-1.$
  (c) 计算 $r_{kk} = \|\tilde{q}_k\|, \quad q_k = \tilde{q}_k / r_{kk}.$

MGS 相对于 CGS 有更优秀的数值稳定性,且重复两遍 MGS 的数值稳定性甚至可以更高.故在实际应用中,常常在一列正交化结束后对之前的所有列进行重正交化,以修正数值误差带来的影响(详见 hw5/codes/qr_mgs2.m & hw5/codes/qr_cgs2.m).

最小二乘问题

设$b \notin Range(A)$, 要求最小化$\|b-Ax\|_2$. 一般而言, 有$A \in \mathbb{C}^{m\times n}, rank(A)=n$.

算法 1 对A作奇异值分解, 得

$$A = U\Sigma V^*$$

于是$Ax=b$可以转化为$\Sigma V^* x = U^* b$, 令$y = V^* x, c = U^* b$, 则

$$\Sigma y = c$$

也即

$$\begin{bmatrix}\Sigma_* & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}y_1\\ y_2\end{bmatrix} = \begin{bmatrix}c_1\\ c_2\end{bmatrix}$$

故当$c_2 \neq 0$时, $b \notin Range(A)$, 此时考虑最小二乘问题, 有

$$\|b-Ax\|_2^2 = \|c-\Sigma y\|_2^2 = \left\|\begin{bmatrix}c_1-\Sigma_* y_1\\ c_2\end{bmatrix}\right\|_2^2 = \|c_1-\Sigma_* y_1\|_2^2 + \|c_2\|_2^2 \geq \|c_2\|_2^2$$

即得

$$\|b-Ax\|_2 \geq \|c_2\|_2$$

注解 此算法的问题在于进行奇异值分解的代价过大.

算法 2 对A作QR分解$A = QR = Q_* R_* \left(Q = [Q_* \quad Q_\perp], R = \begin{bmatrix}R_*\\ 0\end{bmatrix}\right)$, 得

$$\|b-Ax\|_2^2 = \|Q^* b - Rx\|_2^2$$

上式用到了 2 范数的正交不变性. 设$c = Q^* b = \begin{bmatrix}c_1\\ c_2\end{bmatrix}$, 则

$$\|Q^* b - Rx\|_2^2 = \left\|\begin{bmatrix}c_1\\ c_2\end{bmatrix} - \begin{bmatrix}R_*\\ 0\end{bmatrix}x\right\|_2^2 = \|c_1 - R_* x\|_2^2 + \|c_2\|_2^2 \geq \|c_2\|_2^2$$

即得

$$\|b-Ax\|_2 \geq \|c_2\|_2$$

取等时, 有

$$x = R_*^{-1} c_1$$

注解 由于QR分解是较容易做的, 故此算法明显优于算法 1.

算法 2 补充(几何意义) 下面考虑计算$\|b-Ax\|_2 \geq \|c_2\|_2$取等条件的几何意义.

设$Range(A) = span\{q_1, \dots, q_m\}$, 则该空间的投影算子为

$$P_A = QQ^* = \sum_{i=1}^{m} q_i q_i^*$$

设$b = b_* + b_\perp$, 其中$b_* = Ax_* \in Range(A), b_\perp \perp Range(A)$, 则

$$\begin{cases} b_* = \sum_{i=1}^{m} q_i \alpha_i \\ b_\perp = \sum_{i=m+1}^{n} q_i \alpha_i \end{cases}$$

由于$\langle q_i, q_j \rangle = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}$, 故有

$$P_A b = \sum_{i=1}^{m} q_i q_i^* \sum_{j=1}^{n} q_j \alpha_j = \sum_{i=1}^{m} q_i \alpha_i = b_*$$

则$P_A Ax = P_A b \Rightarrow R_* x = Q_*^* b$, 即得$\left(Q = [Q_* \quad Q_\perp], R = \begin{bmatrix} R_* \\ 0 \end{bmatrix}\right)$

$$x = R^{-1}(Q_*^* b)$$

也即

$$x = (Q_*^* A)^{-1} Q_*^* b$$

此时

$$\|b - Ax\|_2^2 = \|b_* + b_\perp - Ax\|_2^2 = \|A(x_* - x) + b_\perp\|_2^2 = \|A(x_* - x)\|_2^2 + \|b_\perp\|_2^2 \geq \|b_\perp\|_2$$

注解 由于Q是一组正交基, 故上式的数值稳定性优于$x = (A^* A)^{-1} A^* b$.

在实际解决问题的过程中, 对A做 QR 分解得$A = QR$, 于是有

$$x = (A^* A)^{-1} A^* b = R^{-1} Q b$$

即先计算 QR 分解$(T_1 = 2mn^2)$, 然后计算$z = Qb (T_2 = 2mn)$, 最后解线性方程组$Rx = z$, $(T_3 = n^2)$.算法的时间复杂度为

$$T = 2mn^2$$

定义 最小二乘问题的解x可被表示为$x = A^+ b$. 其中A^+被称为 Moore-Penrose 广义逆右逆. 其定义式为

$$A^+ = (A^* A)^{-1} A^*$$

注解 由于最小二乘问题的解可以被写为$x = A^+ b$, 故我们定义法方程

$$A^* Ax = A^* b$$

则最小二乘问题的解即该法方程的解.

带约束的最小范数问题

要求在$Cx = d$的条件下最小化$\|x\|_2$. 一般而言, 有$C \in \mathbb{C}^{p \times n}, x \in \mathbb{C}^{m \times 1} (p < m)$. 这一问题的解为

$$x = C^* (CC^*)^{-1} d$$

证明. 事实上,因为此时满足

$$Cx = CC^*(CC^*)^{-1}d = d$$

且对于任意满足$Cx' = d$的$x'(x' \neq x)$,均有

$$\|x'\|_2^2 = \|x + x' - x\|_2^2 = \|x\|_2^2 + 2x^*(x' - x) + \|x' - x\|_2^2 = \|x\|_2^2 + \|x' - x\|_2^2 \geq \|x\|_2^2$$

其中由于$Cx' = Cx = d$,有

$$x^*(x' - x) = d^*(CC^*)^{-1}C(x' - x) = 0$$

在实际解决问题的过程中,对C^*做 QR 分解得$C^* = QR$,于是有

$$x = C^*(CC^*)^{-1}d = QR^{-*}d$$

即先计算 QR 分解$(T_1 = 2p^2n)$,然后计算解线性方程组$R^*z = d(T_2 = p^2)$,最后得到$x = Qz(T_1 = 2pn)$. 算法的时间复杂度为

$$T = 2p^2n$$

定义 带约束的最小范数问题的解x可被表示为$x = A^+b$. 其中A^+被称为 Moore-Penrose **广义逆左逆**. 其定义式为

$$A^+ = A^*(AA^*)^{-1}$$

性质 （1） $AA^+A = A$

（2）$A^+AA^+ = A^+$

（3）$(AA^+)^* = AA^+$

（4）$(A^+A)^* = A^+A$

Krylov Arnoldi 算法(1951)——迭代扩展单位正交阵Q

核心思想：在一个维度较小的子空间中寻找近似解$A^{-1}b = p(A)b$.

定义 设$A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n$, 定义$A$和$b$生成的 Krylov subspace 为

$$K_m(A,b) = span\{b, Ab, \dots, A^{m-1}b\} \subseteq \mathbb{R}^n$$

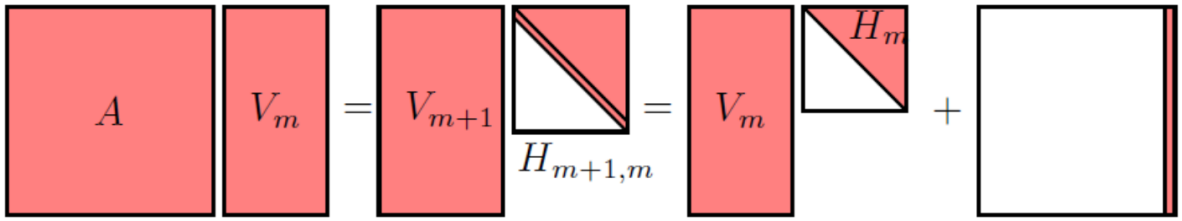
对 Krylov 子空间做单位正交化, 即得一组正交基, 于是

$$K_k(A,b) = span\{q_1, q_2, \dots, q_k\}$$

这一过程被称为 **Arnoldi 过程**, 其中q_i被称为 **Arnoldi 向量**.

Arnoldi 过程实际上可以看作下面的分解(如下图, $\widetilde{H}_k$被称为 Hessenberg 矩阵, 事实上是R阵向左平移一位)

$$AQ_k = Q_{k+1}\widetilde{H}_k = Q_kH_k + q_{k+1}\beta_k e_k^*$$



其中, $\widetilde{H}_k \in \mathbb{C}^{(k+1)\times k}$是一个多一斜线的上三角阵, $H_k \in \mathbb{C}^{k\times k}$是一个上三角阵. 算法的实现代码如左下

$v_1 = r/\|r\|_2$
for $j = 1$ to m **do**
  $z = Av_j$
  for $i = 1$ to j **do** % MGS 正交化过程
    $h_{i,j} = (v_i, z), \quad z = z - h_{i,j}v_i$
  end for
  $h_{j+1,j} = \|z\|_2$ % if $h_{j+1,j} = 0$, break, endif
  $v_{j+1} = z/h_{j+1,j}$
end for

Set $v_0 = 0$ and $\beta_0 = 0$
$v_1 = r/\|r\|_2$
for $j = 1$ to m **do**
  $z = Av_j$
  $\alpha_j = (v_j, z)$
  $z = z - \alpha_j v_j - \beta_{j-1}v_{j-1}$
  $\beta_j = \|z\|_2$
  if $\beta_j = 0$ **then** break, **end if**
  $v_{j+1} = z/\beta_j$
end for

特别地, 当A是 Hermite 矩阵时, 有

$$Q_k^*AQ_k = Q_k^*Q_kH_k + Q_k^*q_{k+1}\beta_k e_k^* = H_k$$

故H_k是一个 Hermite 三对角阵, 此时记之为T_k. 于是 Krylov Arnoldi 迭代过程可以进一步写为

$$AQ_k = Q_kT_k + q_{k+1}\beta_k e_k^*$$

这一特殊情况下的优化算法称为 Lanczos **算法**(1950). 算法的实现代码如右上

[**注**] 由于此算法已经认为T_k是三对角阵, 对于零元不再加以计算, 故该算法对舍入误差更加敏感, 数值稳定性可能较差.

秩亏损条件下的最小二乘问题——列选主元的 QR 分解

说明：A亏秩意味着给定的数据点存在重复.

每次把范数最大的列交换到子矩阵中的第一列, 做列选主元 QR 分解

$$A\Pi = QR$$

则得到的R矩阵的梯形矩阵, 即(其中$R' = [R_* \quad 0]\tilde{Q}$)

$$R = \begin{bmatrix} R' \\ 0 \end{bmatrix}$$

由于列选主元效率很低, 故除非事先知道亏秩, 并希望避免奇异值分解, 否则一般不使用该算法.

岭回归

考察

$$\min_x \|b - Ax\|_2^2 + \mu\|x\|_2^2 \ (\mu > 0)$$

可转化为标准的最小二乘问题

$$\min_x \left\| \begin{bmatrix} A \\ \sqrt{\mu} I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2$$

增广最小二乘法

设残差$r = b - Ax$, 则最小二乘问题$\min_x \|b - Ax\|_2$相当于求函数

$$f(x) = (b - Ax)^*(b - Ax)$$

的极小值解. 对其求导, 即得

$$\frac{df}{dx} = -2A^*(b - Ax)$$

极小值点处导数为 0, 即

$$A^*r = 0$$

于是求解$\min_x \|b - Ax\|_2$可转化为求解线性方程组

$$\begin{bmatrix} I & A \\ A^* & 0 \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

注解 也可以从法方程为$A^*r = 0$的角度理解. 两种方法本质上是一样的, 但增广系统法可以把问题转化为求解对称系统的解, 对病态问题有更好的数值表现.

带约束的最小二乘问题

要求在$Cx = d$的条件下最小化$\|Ax - b\|_2$. 一般有$C \in \mathbb{C}^{p\times n}, A \in \mathbb{C}^{m\times n}(p < m)$.

解法 1 这一问题的解为增广线性系统

$$\begin{bmatrix} A^*A & C^* \\ C & 0 \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} A^*b \\ d \end{bmatrix}$$

的解. 其中向量$z \in \mathbb{C}^{p\times 1}$.

证明. 事实上, 因为对于其它满足$Cx' = d$的x'来说, 有

$$\begin{aligned} \|Ax' - b\|_2^2 &= \|A(x' - x) + Ax - b\|_2^2 \\ &= \|A(x' - x)\|_2^2 + \|Ax - b\|_2^2 + 2(x' - x)^*A^*(Ax - b) \end{aligned}$$

由于$A^*Ax + C^*z = A^*b \Rightarrow A^*(Ax - b) = -C^*z$, 故

$$2(x' - x)^*A^*(Ax - b) = -2(x' - x)^*C^*z = -2[C(x' - x)]^*z = 0$$

从而

$$\|Ax' - b\|_2^2 = \|A(x' - x)\|_2^2 + \|Ax - b\|_2^2 \geq \|Ax - b\|_2^2$$

算法 1.1 LU 分解

设$H = A^*A(T_1 = mn^2), c = A^*b(T_2 = 2mn)$, 通过 LU 分解解线性方程组

$$\begin{bmatrix} H & C^* \\ C & 0 \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}$$

即可$\left(T_3 = \frac{2}{3}(p + n)^3\right)$. 总时间复杂度为

$$T = \frac{2}{3}(p + n)^3$$

算法 1.2 QR 分解

设$w = z - d$, 则法方程化为

$$\begin{bmatrix} A^*A + C^*C & C^* \\ C & 0 \end{bmatrix}\begin{bmatrix} x \\ w \end{bmatrix} = \begin{bmatrix} A^*b \\ d \end{bmatrix}$$

做 QR 分解

$$\begin{bmatrix} A \\ C \end{bmatrix} = QR = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R$$

则法方程化为

$$\begin{bmatrix} R^*R & R^*Q_2^* \\ Q_2R & 0 \end{bmatrix}\begin{bmatrix} x \\ w \end{bmatrix} = \begin{bmatrix} R^*Q_1^*b \\ d \end{bmatrix}$$

对第一行左右同乘R^{-*}, 并令$y = Rx$, 即得

$$\begin{bmatrix} I & Q_2^* \\ Q_2 & 0 \end{bmatrix}\begin{bmatrix} y \\ w \end{bmatrix} = \begin{bmatrix} Q_1^*b \\ d \end{bmatrix}$$

解法 2 这一问题的解为增广线性系统

$$\begin{bmatrix} I & A \\ A^* & 0 \\ 0 & C \end{bmatrix}\begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \\ d \end{bmatrix}$$

的解. 其中r表示残差$b - Ax$.

证明. 设 Lagrange 函数

$$L(x,\lambda) = \frac{1}{2}(\|Ax - b\|_2^2 + \lambda\|Cx - d\|_2^2)$$

令

$$\begin{cases} \dfrac{\partial L}{\partial x} = A^*(Ax - b) + \lambda C^*(Cx - d) = 0 \\ \|Cx - d\|_2^2 = 0 \end{cases}$$

即得条件极值处的x应满足所证增广线性系统

解法 3 对$Cx = d$做高斯消元, 得到

$$[R \quad C_2]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = d'$$

其中, 矩阵$[R \quad C_2]$是一个梯形矩阵. 于是解得

$$x_1 = R^{-1}(d' - C_2x_2)$$

按$[x_1 \quad x_2]$的规模大小将A分块为$[A_1 \quad A_2]$, 则

$$\min_{\mathrm{Cx=d}}\|b - Ax\|_2 = \min\|A_1x_1 + A_2x_2 - b\|_2$$

$$= \min\left\|\left((A_2 - A_1R^{-1}C_2)x_2 - (b - A_1R^{-1}d')\right)\right\|_2$$

解法 4 对C^*做 QR 分解, 得$C^* = QR = Q\begin{bmatrix} R_* \\ 0 \end{bmatrix}$, 并令$Qx = y$, 则

$$[R_* \quad 0]\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = d$$

于是最小二乘问题化为

$$\min_{\mathrm{Cx=d}}\|b - AQ^*y\|_2$$

令$AQ^* = [\hat{A}_1 \quad \hat{A}_2]$, 则上式进一步化为

$$\min_{\mathrm{x}}\left\|b - \hat{A}_1y_1 - \hat{A}_2y_2\right\|_2$$

其中$R_*y_1 = d \Rightarrow y_1 = R_*^{-1}d$, 于是上式进一步化为

$$\min_{\mathrm{y_2}}\left\|\left(b - \hat{A}_1R_*^{-1}d\right) - \hat{A}_2y_2\right\|_2$$

特征值问题

• **知λ求x** 解线性方程组

$$(A-\lambda I)x=0$$

例(Google Pagerank) 对数据库内的网站进行排序.

在存在指向链接的两个网站之间连接上一条有向边,认为入度大的节点网站为重要性更大的网站,设该图的概率加权邻接矩阵为A_0.

由于用户操作分为(1)直接键入网址访问网站;(2)点击页内链接访问网站.则考虑Markov Chain:

$$A=(1-\alpha)ee^T+\alpha A_0(0<\alpha<1)$$

其中$e=(1,1,\dots,1)^T$,其表示可以任意访问任意网站.设初始状态为$x_0=\frac{e}{n}$,则需要考虑

$$x_n=\lim_{n\to\infty}(A^T)^n x$$

这里的x_n即最终所有网站的权重,故可据此进行网站的权重排序.此式的不动点为

$$A^Tx=x$$

即需要求A^T对应于$\lambda=1$的特征向量.即解线性方程组(采用 GTH **算法**:避免扰动以精确解病态问题)

$$(A^T-I)x=0$$

[注] GTH 算法可以极小化向前误差,能做到$|fl(x_i)\cdot x_i|\le O(u)\cdot x_i$.

• **知x求λ** 把$Ax=x\lambda$转化为法方程$x^*Ax=x^*xA$,即得

$$\lambda=\frac{x^*Ax}{x^*x}$$

定义 Rayleigh **商**:$\frac{x^*Ax}{x^*x}$.

• $\boldsymbol{x,\lambda}$均未知

求模最大的特征值对应的特征向量：乘幂法(power method)

当A可对角化时, 设$A = P\Lambda P^{-1}$, 并令$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$. 设初始向量$x_0$为

$$x_0 = Pc = \sum_{i=1}^{n} c_i p_i$$

则$Ax_0 = P\Lambda c = \sum_{i=1}^{n} p_i \lambda_i c_i$, 则

$$x_k = A^k x_0 = \sum_{i=1}^{n} \lambda_i^k c_i p_i = \lambda_1^k \sum_{i=1}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^k c_i p_i \approx \lambda_1^k c_1 p_1$$

当A不可对角化时, 设

$$A = P \begin{bmatrix} \lambda_1 & 0 \\ 0 & A_2 \end{bmatrix} P^{-1}$$

其中A_2是不可对角化的, 于是$\rho\left(\frac{A_2}{\lambda_1}\right) \leq \left|\frac{\lambda_2}{\lambda_1}\right| < 1 \Rightarrow \left(\frac{A_2}{\lambda_1}\right)^k \to 0$, 故

$$x_k = A^k x_0 \approx \lambda_1^k P \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} P^{-1} x_0 = \lambda_1^k p_1 (q_1^T x_0)$$

其中称q_1^T为A的**左乘特征向量**, 即满足$q_1^T A = \lambda q_1^T \Rightarrow A^T q_1 = \lambda q_1$.

考虑到直接计算λ_1^k会造成数值溢出, 故在迭代$x_{n+1} = Ax_n$时, **对$\boldsymbol{x_{n+1}}$做归一化处理.**

$\boldsymbol{x_k}$最终收敛到$\boldsymbol{x}$, 算法的收敛速度取决于$\left|\frac{\lambda_2}{\lambda_1}\right|$. 即当$|\lambda_1| = |\lambda_2|$时, 乘幂法将失效.

定义 特征值的**条件数**$\sigma(\lambda)$用于刻画特征值的分离程度, 也即特征向量的数值稳定性, 例如

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1+\varepsilon & 0 \\ 0 & 1 \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = I + \frac{1}{2}\begin{bmatrix} \varepsilon & \varepsilon \\ \varepsilon & \varepsilon \end{bmatrix}$$

的特征对为$\left(1+\varepsilon, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right), \left(1, \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right)$,

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 1+\varepsilon \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = I + \frac{1}{2}\begin{bmatrix} \varepsilon & -\varepsilon \\ -\varepsilon & \varepsilon \end{bmatrix}$$

的特征对为$\left(1+\varepsilon, \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right), \left(1, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$, 这表明特征值$\lambda$的轻微扰动对矩阵$A$的影响不大(注: 矩阵的轻微扰动将导致特征值的巨大变化, 见 Jordan 标准型), 但将导致对应特征向量x差异巨大. 也即使用乘幂法求特征向量时难以处理重特征值的情况.

求模最小的特征值对应的特征向量：反幂法(inverse iteration)

使用迭代$x_{n+1} = A^{-1}x_n$, 过程完全类似乘幂法.

求模最接近μ的特征值对应的特征向量：带原点位移的反幂法(shift-and-invert)

使用迭代$x_{n+1} = (A - \mu I)^{-1}x_n$, 过程完全类似乘幂法.

Rayleigh 商迭代法(Rayleigh quotient iteration)

即对反幂法做进一步优化,具体迭代过程为:

$$x_{k+1} \leftarrow (A-\mu_k I)^{-1}x_k$$
$$\mu_{k+1} \leftarrow \frac{x_{k+1}^* A x_{k+1}}{\|x_{k+1}\|_2^2}$$

相对于之前三个算法的**线性收敛**速度($\|r_{k+1}\| \le \gamma\|r_k\|(0<\gamma<1)$),Rayleigh 商迭代法的收敛速度是**二次收敛**($\|r_{k+1}\| \le C\|r_k\|^2(0<C<1)$).

缺陷:

(1)若μ_0极度不准,则 Rayleigh 商迭代法可能收敛至其它特征值.故 Rayleigh 商迭代法的全局收敛性不优.

(2)Rayleigh 商迭代法在每一步迭代中都需要解一个新的线性方程组,而反幂法的每一步迭代可以共用同一LU分解,计算量远小于 Rayleigh 商迭代法.

(3)当迭代接近收敛时,$A-\mu_k I$将接近病态,数值稳定性差.但幸运的是这一不稳定性体现在解向量的模不准,而解向量的方向偏差不大,故每一步迭代后都对解向量做归一化即可消除此舍入误差.例如,设

$$A^{-1}=\begin{bmatrix}10^3 & 0\\ 0 & 1\end{bmatrix}$$

则

$$A\Delta x=\begin{bmatrix}10^3\Delta x_1\\ \Delta x_2\end{bmatrix}$$

即扰动后得到的特征向量仅模长变化,而方向几乎没有受到影响.

事实上,对A做 Schur 分解,得$A=QTQ^*$.进一步对T做对角化$T=P\Lambda P^{-1}$,得到

$$A=(QP)\Lambda(QP)^{-1}$$

于是

$$(A-\mu I)^{-1}=Q(T-\mu I)^{-1}Q^*$$

下面考虑计算上三角阵T对应于λ_i的特征向量x_i.我们发现

$$T-\mu_i I=\begin{bmatrix}\times & \times & \times & \times & \times & \times\\ & \times & \times & \times & \times & \times\\ & & 0 & \times & \times & \times\\ & & & \times & \times & \times\\ & & & & \times & \times\\ & & & & & \times\end{bmatrix}$$

于是对其做其对应的特征向量即

$$x_i=[\times \quad \times \quad 1 \quad 0 \quad 0 \quad 0]^T$$

故求解上三角阵T的特征向量x_i,只需令$x_{ii}=1$,然后令$x_{ij}=0(j>i)$,回代解出

$x_{ij}(j<i)$即可.

子空间迭代法——同时求出模较大特征值对应的特征向量

考虑同时迭代p个正交规范向量. 即设初始正交向量矩阵

$$X_0 = \begin{bmatrix} x_1^{(0)} & x_2^{(0)} & \dots & x_p^{(0)} \end{bmatrix}$$

则有迭代过程$X_{k+1} = AX_k$, 其中对A做对角化$A = P\Lambda P^{-1}$, 并令$X_0 = PC$, 则有

$$A^k x_j^{(0)} = \lambda_1^k \sum_{i=1}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^k c_{ij} p_i$$

当λ_p与λ_1差距不大时, 有$span\ A^k\begin{bmatrix} x_1^{(0)} & x_2^{(0)} & \dots & x_p^{(0)} \end{bmatrix} \approx span\ \{p_1, p_2, \dots, p_k\}$. 对迭代终点$X_k$做正交化分解即可得前$p$个特征向量. (或, 得到$r$个特征向量和$p-r$个零向量, 这意味着有$r$个特征值的模较大, 其余特征值远小于$\lambda_1$)

正交迭代算法——子空间迭代法的优化

设p个正交规范向量组成向量矩阵. 特别地, 令初始正交向量矩阵为$X_0 = I$. 在迭代的过程中, 为了防止子空间X在迭代过程中各列均收敛到最大特征值对应的特征向量, 故我们每一步迭代均对X做 QR 分解, 即

$$X_{k+1} = AX_k = Q_{k+1}R_{k+1}$$

并做赋值

$$X_{k+1} \leftarrow Q_{k+1}$$

于是我们有

$$Q_{k+1}R_{k+1} = AQ_k$$

从$k = 0$处开始迭代, 即

$$A = Q_1 R_1$$

$$A^2 = AQ_1R_1 = Q_2(R_2R_1)$$

$$A^3 = AQ_2(R_2R_1) = Q_3(R_3R_2R_1)$$

于是我们得到收敛处A^n满足

$$A^n = Q_n(R_nR_{n-1}\dots R_1) = Q_nT$$

这样我们就可以同时获得矩阵A的r个模较大的特征值. 在某些时候, 收敛结果可能是A的一个 Schur 分解.

QR 迭代算法

核心思想：相似矩阵不改变矩阵的数值性质. 故可以通过迭代相似矩阵求得初始矩阵的特征值.

令初始矩阵$A_0 = A$, 往后的每一步迭代中, 都执行

$$A_k = Q_{k+1}R_{k+1}$$

$$A_{k+1} = R_{k+1}Q_{k+1}$$

于是有

$$R_kQ_k = Q_{k+1}R_{k+1}$$
$$A_{k+1} = Q_{k+1}^* A_k Q_{k+1}$$

故有以下结论

（1）每一步迭代结果均与原矩阵A酉相似，即

$$A_k = R_kQ_k = Q_k^*(Q_kR_k)Q_k = Q_k^*A_{k-1}Q_k = (Q_1Q_2 \dots Q_k)^*A(Q_1Q_2 \dots Q_k)$$

（2）迭代结果可直接求出A^k正交分解，即

$$A^2 = Q_1R_1Q_1R_1 = Q_1Q_2R_2R_1$$
$$A^3 = Q_1R_1Q_1R_1Q_1R_1 = Q_1Q_2R_2Q_2R_2R_1 = Q_1Q_2Q_3R_3R_2R_1$$
$$A^k = (Q_1Q_2 \dots Q_k)(R_k \dots R_2R_1)$$

记$\tilde{Q}_k = Q_1Q_2 \dots Q_k$，$\tilde{R}_k = R_k \dots R_2R_1$，则上面的结论可以被表示为

（1）$A_k = \tilde{Q}_k^* A \tilde{Q}_k$.

（2）$A^k = \tilde{Q}_k\tilde{R}_k$.

根据以上结论可以证明得到重要结论(证明见【NA】基于 QR 分解的特征值迭代法 qr 迭代，画出特征值作为 qr 因子分解数的函数和半对数曲线-CSDN 博客)：

$$\lim_{k\to\infty} A_k = R$$

其中R为上三角阵，且$R_{ii} = \lambda_i$. 结合之前的结论（1），说明 QR 迭代的结果实际上是对矩阵A进行的 Schur 分解.

此算法的收敛速度由$\|\Lambda_k\|_\infty \le C \max\limits_{j\in[1,n-1]} \left|\frac{\lambda_{j+1}}{\lambda_j}\right|^k$决定，当$r_n = \left|\frac{\lambda_n}{\lambda_{n-1}}\right| \approx 1$时收敛极慢，故可以仿照反幂法的优化，采取原点位移策略进行优化，即迭代矩阵$A - \mu I$，从而使得收敛速度$r_n' = \left|\frac{\lambda_n - s}{\lambda_{n-1} - s}\right|$变得尽可能大.

缺陷 算法求解单个特征值的时间复杂度为$O(n^4)$(每一步迭代的效率均为$O(n^3)$，总迭代次数为n次)，计算所有特征值所需要的时间复杂度为$O(n^5)$.

这是由于迭代过程中，每一次 QR 分解都需要大量计算，同时每一次完全的矩阵相乘RQ也需要大量计算. 在之后的 Francis 算法中，我们将针对此问题做进一步优化.

QR 迭代的优化算法

算法 1 Rutishauser 算法(1950s)

令初始矩阵$A_0 = A$,往后的每一步迭代中,都执行

$$A_k = L_k R_k$$

$$A_{k+1} = R_k L_k$$

于是有

$$A_{k+1} = L_k^{-1} A_k L_k$$

缺陷在于这一算法的代价极大(因为 LU 分解必然需要选主元),且收敛速度慢.

算法 2 Francis 算法(1961)

通过基于 Householder 变换的酉变换,可以将初始矩阵$A_0 = A$酉相似变换为

$$Q^* A_0 Q = H_0 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix}$$

其中H_0是一个上 Hessenberg 矩阵. 事实上,该操作也可被视作一次 Arnoldi 变换

$$A_0 Q = Q H_0$$

接下来的迭代过程中,每一步均执行

$$H_k = Q_k R_k$$

$$H_{k+1} = R_k Q_k$$

此处迭代求单个特征值的时间复杂度是$O(n^3)$(每一步迭代的效率均为$O(n^2)$,总迭代次数为n次),它远优于原始的 QR 迭代算法.

若已知一个(近似的)特征值μ,则存在更快收敛的迭代(原点位移):

$$H_k - \mu I = Q_k R_k$$

$$H_{k+1} = R_k Q_k + \mu I$$

此时仍然有

$$H_{k+1} = R_k Q_k + \mu I = Q_k^* (H_k - \mu I) Q_k + \mu I = Q_k^* H_k Q_k$$

这里的迭代结果将为

$$H_{k+1} = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \varepsilon & \varepsilon + \mu \end{bmatrix}$$

即此时通过一步迭代即可得到特征值.

进一步地,我们有基于 Rayleigh 商迭代的优化算法:考虑对$(H-\mu I)^{-T}$中μ对应特征值λ的特征向量做迭代,由于初始条件下有近似

$$H^T=\begin{bmatrix}\times & \times & & & & \\ \times & \times & \times & & & \\ \times & \times & \times & \times & & \\ \times & \times & \times & \times & \times & \\ \times & \times & \times & \times & \times & \varepsilon \\ \times & \times & \times & \times & \times & \varepsilon+\mu\end{bmatrix}$$

当$\varepsilon=0$时,$\lambda=\mu$对应的特征向量即$b=(0\quad 0\quad 0\quad 0\quad 0\quad 1)^T$. 换言之,$b$事实上是特征值$\lambda$的近似特征向量. 在实际操作中,我们事先选定初始向量$x_0=b$(如上定义),然后求得近似特征值$\mu_0=\frac{b^THb}{\|b\|_2^2}$. 由此开始 Rayleigh 商迭代,具体迭代过程为

$$x_{k+1}\leftarrow(H-\mu_kI)^{-T}x_k$$
$$\mu_{k+1}\leftarrow\frac{x_{k+1}^*Hx_{k+1}}{\|x_{k+1}\|_2^2}$$

由于$x_{k+1}\leftarrow(H-\mu_kI)^{-T}x_k$其实是在解线性方程组

$$(H-\mu_kI)^Tx_{k+1}=x_k$$

解该线性方程组可通过对$H-\mu_kI$做 QR 分解,得到$R_k^TQ_k^Tx=b$后再进一步消元求解.

这一迭代算法可以在常数步内收敛,则求解单个特征值的时间复杂度为$O(n^2)$(每一步迭代的效率均为$O(n^2)$,总迭代次数为常数次).

若要进一步迭代计算其余特征值,可以在之前的迭代基础上,对左上角规模为$(n-1)\times(n-1)$子阵做相似的迭代(即排除已迭代出答案的行). 这样计算所有特征值所需要的时间复杂度为$O(n^3)$.

Francis 进一步优化:隐式 QR 迭代

我们考虑不计算每一步迭代的 QR 分解$A_k=Q_kR_k$,而直接由A_k迭代至A_{k+1}.

定理(Implicit Q Theorem) 设$H=Q^TAQ\in\mathbb{R}^{n\times n}$是一个不可约上 Hessenberg 阵,其中$Q\in\mathbb{R}^{n\times n}$是正交矩阵,则$Q$的第 2 列至第$n$列均由$Q$的第一列唯一确定(至多相差一个符号).

对于朴素的 QR 迭代算法,每一步迭代过程记为

$$A_{k+1}=Q_k^TA_kQ_k$$

则相应的隐式 QR 迭代为

$$\tilde{A}_{k+1}=\tilde{Q}_k^TA_k\tilde{Q}_k$$

其中,由上面的定理,我们有$\tilde{Q}_k=WQ_k$,其中$W=diag(1,\pm1,\dots,\pm1)$. 于是

$$\tilde{A}_{k+1}\tilde{Q}_k^TA_k\tilde{Q}_k=W^TQ_k^TA_kQ_kW=W^TA_{k+1}W$$

即$\tilde{A}_{k+1}$与A_{k+1}的对角线元素相等,非对角线元素至多相差一个符号. 这表明在迭代过程中,我们只需要找到任意一个正交阵$\tilde{Q}_k$,使得$\tilde{Q}_k^TH\tilde{Q}_k$仍为不可约上 Hessenberg 阵即可.

由案例可知$\tilde{Q}_k = G_1 G_2 \dots G_{n-1}$, 其第一列为

$$[c_1 \quad s_1 \quad 0 \quad \dots \quad 0]^T$$

将其取为H的第一列$[a_{11} \quad a_{21} \quad 0 \quad \dots \quad 0]^T$归一化后的向量, 则$\tilde{Q}_k$的第一列与$Q_k$的第一列相同. 后续一种可行的构造是使用$n-2$次 Givens 变换进行构造, 案例见 slides_MC04_eig_nonsymm.pdf 中 P58-P64.

• 若要加入原点位移优化, 只需将$\tilde{Q}_k$的第一列取为$[a_{11}-\mu \quad a_{21} \quad 0 \quad \dots \quad 0]^T$单位化后的向量即可.

这一优化可以使每一步迭代的运算量降至$6n^2 + O(n)$.

原点位移优化:Francis 双位移策略

当λ是复数时, 原点位移优化将失效.

考虑$A \in \mathbb{R}^{n\times n}$的一个特征对$(x+iy) = (x+iy)(\alpha+i\beta)$($x, y$是线性无关的), 显然复数特征值/特征向量是成对(共轭)出现的. 于是

$$A[x+iy \quad x-iy] = [x+iy \quad x-iy]\begin{bmatrix}\alpha+i\beta & \\ & \alpha-i\beta\end{bmatrix}$$

进而有

$$A[x \quad y]\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix} = [x \quad y]\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}\begin{bmatrix}\alpha+i\beta & \\ & \alpha-i\beta\end{bmatrix}$$

即

$$A[x \quad y] = [x \quad y]\left(\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}\begin{bmatrix}\alpha+i\beta & \\ & \alpha-i\beta\end{bmatrix}\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}^{-1}\right)$$

其中$X = \begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}\begin{bmatrix}\alpha+i\beta & \\ & \alpha-i\beta\end{bmatrix}\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}^{-1} \in \mathbb{R}^{2\times 2}$. 这样就将复数域上的特征问题转化为实矩阵的特征问题.

在实际操作中, 设μ是复特征值λ的一个近似, 则$\bar{\mu}$也是$\bar{\lambda}$的一个近似. 我们在迭代过程中, 对μ和$\bar{\mu}$交替迭代位移, 即

$$H_k - \mu I = Q_k R_k$$
$$H_{k+1} = R_k Q_k + \mu I$$
$$H_{k+1} - \bar{\mu} I = Q_{k+1} R_{k+1}$$
$$H_{k+2} = R_{k+1} Q_{k+1} + \bar{\mu} I$$

于是有

$$H_{k+2} = Q_{k+1}^* H_{k+1} Q_{k+1} = (Q_k Q_{k+1})^* H_k (Q_k Q_{k+1})$$

其中$Q_k Q_{k+1}$可以是一个实阵. 事实上, 由于

$$Q_k (H_{k+1} - \bar{\mu} I) Q_k^* = Q_k R_k + (\mu - \bar{\mu}) I = H_k - \bar{\mu} I$$

故

$$(H_k - \mu I)(H_k - \bar{\mu} I) = Q_k (H_{k+1} - \bar{\mu} I) Q_k^* (H_k - \mu I) = (Q_k Q_{k+1})(R_{k+1} R_k)$$

即

$$H_k^2 - 2Re(\mu)H_k + |\mu|^2 I = (Q_k Q_{k+1})(R_{k+1}R_k)$$

从而上式是实矩阵$H_k^2 - 2Re(\mu)H_k + |\mu|^2 I$的一个 QR 分解, 故$Q_k Q_{k+1}$可以是一个实阵.

于是为了使迭代在实数域上进行, 我们希望的迭代过程为

$$H_{k+2} = (Q_k Q_{k+1})^T H_k (Q_k Q_{k+1})$$

这样的迭代过程可以避免复数的运算. 在实际操作中, 令$Q_k Q_{k+1}$的第一列为$H_k^2 - 2Re(\mu)H_k + |\mu|^2 I$的第一列的归一化向量, 也即

$$\begin{bmatrix} a_{11}^2 + a_{12}a_{21} - 2Re(\mu)a_{11} + |\mu|^2 \\ a_{21}\big(a_{11} + a_{22} - 2Re(\mu)\big) \\ a_{21}a_{32} \\ 0 \\ \vdots \end{bmatrix}$$

的归一化向量, 并通过 MGS 扩展出第一个正交阵. 后续一种可行的构造是使用$n-3$次 Householder 变换和1次 Givens 变换进行构造, 案例见 slides_MC04_eig_nonsymm.pdf 中 P70-P77.

接下来考虑如何选取合适的μ. 选取H_k的右下角矩阵

$$\begin{bmatrix} H_k(n-1,n-1) & H_k(n-1,n) \\ H_k(n,n-1) & H_k(n,n) \end{bmatrix}$$

的复共轭特征值作为双位移(**Francis 位移**), 收敛结果为一个 2 阶子矩阵. 若该子阵的两个特征值均为实的, 则选取其中距离$H_k(n,n)$较近的特征值做单原点位移(**Wilkinson 位移**), 收敛结果为一个上三角矩阵. 这一算法具有二次收敛性, 即收敛一个特征值一般只需要迭代两步.

收敛的标志为$H_k(i+1,i)$是否趋向0, 实际操作中判断标准为

$$|H_k(i+1,i)| \le O(\omega)\big(H_k(i,i) + H_k(i+1,i+1)\big)$$

注 Francis QR 迭代并非对所有矩阵都收敛, 例如

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

特征值的换序问题

设$Q = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$, 考虑使变换

$$Q^T \begin{bmatrix} a & d \\ 0 & b \end{bmatrix} Q = \begin{bmatrix} b & \pm d \\ 0 & a \end{bmatrix}$$

成立, 于是可以很方便地解出c和s, 进而确定Q. (见 hw08-T4)

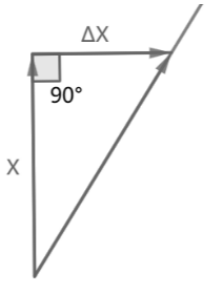
这表明相邻特征值可以做次序交换, 于是利用冒泡排序可以对特征值任意排序.

特征值的扰动分析

考虑对A施加一个扰动ΔA, 得到$\tilde{x} = x + \Delta x$, 由于特征向量的模长可以任意指定, 故可以对$\tilde{x}$取得适当的模长, 使得$x \perp \Delta x$. 展开$(A + \Delta A)(x + \Delta x) = (x + \Delta x)(\lambda + \Delta\lambda)$, 约去极小项, 即得

$$\Delta A x + A\Delta x \approx \Delta x\lambda + x\Delta\lambda$$

设λ对应的左乘特征向量为y^* $(y^*A = \lambda y^*)$, 对上式同时左乘y^*, 得



$$y^*\Delta A x \approx y^* x\Delta\lambda$$

其中规定$\|x\|_2 = \|y\|_2 = 1$. 于是我们得到特征值受扰动ΔA的影响为

$$\Delta\lambda = \frac{y^*\Delta A x}{y^* x} + O(\|\Delta A\|^2)$$

定理(Weyl Theorem) 设 Hermite 阵A, 以及 Hermite 的扰动ΔA, 则

$$|\lambda_i(A + \Delta A) - \lambda_i(A)| \le \|\Delta A\|_2$$

定理(Hoffman-Wielandt Theorem) 设 Hermite 阵A, 以及 Hermite 的扰动ΔA, 则

$$\sum_{i=1}^{n} |\lambda_i(A + \Delta A) - \lambda_i(A)|^2 \le \|\Delta A\|_F^2$$

以上两个定理表明奇异值是向后稳定的.

对称实阵的特征值问题

算法 1 Jacobi 算法(1845)

给定对称阵A, 设

$$off(A) = \sum_{i \ne j} |a_{ij}|^2 = \|A\|_F^2 - \|Diag(A)\|_F^2$$

由于 F 范数具有酉不变性, 故

$$off(Q^*AQ) = off(A) - 2|a_{i_0 j_0}|^2$$

这是单调递减的. 采用贪心的选主元策略, 每次都取模最大的非对角元素, 做正交相似变换把对应的2×2子阵变为对角阵, 从而将模最大元素置零, 这样将得到

$$|a_{i_0 j_0}|^2 \ge \frac{off(A)}{n^2 - n}$$

于是

$$off(Q_k^* \dots Q_1^* A Q \dots Q_k) \le \left(1 - \frac{1}{n^2 - n}\right)^k off(A) \to 0$$

事实上, 该算法具有二次收敛性.

正交相似变换的酉矩阵$Q=\begin{bmatrix}c & -s\\ s & c\end{bmatrix}$由下面的方程确定

$$\begin{bmatrix}c & s\\ -s & c\end{bmatrix}\begin{bmatrix}a_{ii} & a_{ij}\\ a_{ji} & a_{jj}\end{bmatrix}\begin{bmatrix}c & -s\\ s & c\end{bmatrix}=\begin{bmatrix}\lambda_i & 0\\ 0 & \lambda_j\end{bmatrix}$$

解得

$$\theta=\begin{cases}\pm\dfrac{\pi}{4} & ,a_{ii}=a_{jj}\\ \dfrac{\arctan\left(\dfrac{2a_{ij}}{a_{ii}-a_{jj}}\right)}{2} & ,a_{ii}\neq a_{jj}\end{cases}$$

缺陷 选主元操作代价极大.

算法 2 对称 QR 算法

Step 1 对A做酉相似变换$T=Q^*AQ$,其中T是一个三对角矩阵(见 hw10-T1).

Step 2 每步迭代采取 Wilkinson 隐式位移,即选取右下角2×2子阵中离$T(n,n)$较近的特征值作为位移参与迭代.迭代过程中,采用 Givens 变换对称地逐一消去次对角线之外的 bulge.例如,采用$G(3,4)$将下面矩阵的 bulge 向右下排一位:

$$\begin{bmatrix}\times & \times & & & & \\ \times & \times & \times & + & & \\ & \times & \times & \times & & \\ & + & \times & \times & \times & \\ & & & \times & \times & \times\\ & & & & \times & \times\end{bmatrix}$$

注 (1)该算法的时间复杂度与 Jacobi 算法一样同为$O(n^3)$,但实际上效率远优于 Jacobi 算法.

(2)实现细节参考前面的 Francis 隐式双位移,几乎是一样的.

算法 3 Divide & Conquer(分治算法)

接算法 2 的步骤一,可以把n维对称三对角阵二分为($xx^T=e_m+\rho e_{m+1}$表示秩 1 矩阵,$m=\frac{n}{2}$)

$$\begin{bmatrix}A_1 & 0\\ 0 & A_2\end{bmatrix}+xx^T$$

当二分得到的子阵A_1,A_2的阶数足够小时,直接应用 Francis 双位移,有

$$\begin{bmatrix}Q_1\Lambda_1Q_1^T & 0\\ 0 & Q_2\Lambda_2Q_2^T\end{bmatrix}+xx^T=\begin{bmatrix}Q_1 & \\ & Q_2\end{bmatrix}\left(\begin{bmatrix}\Lambda_1 & \\ & \Lambda_2\end{bmatrix}+yy^T\right)\begin{bmatrix}Q_1 & \\ & Q_2\end{bmatrix}^T$$

于是解决对称阵的特征值问题,相当于解决一个对角阵+秩 1 矩阵的特征值问题.

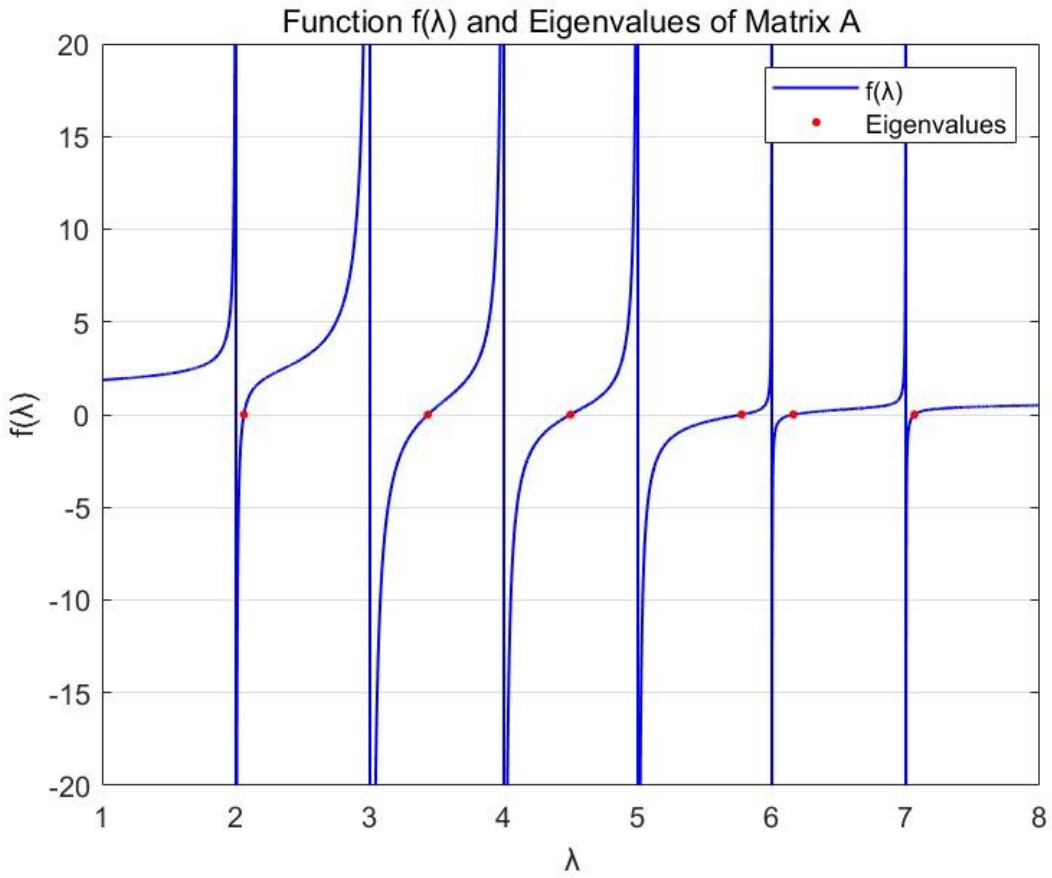
令$D=\lambda I-\Lambda$, 由于特征多项式

$$\det(D-yy^T)=\det\begin{bmatrix}1 & y^T\\ y & D\end{bmatrix}=(1-y^TD^{-1}y)\det D=\left(1-\sum_{i=1}^{n}\frac{y_i^2}{\lambda-\lambda_i}\right)\prod_{i=1}^{n}(\lambda-\lambda_i)=0$$

考虑函数

$$f(\lambda)=\sum_{i=1}^{n}\frac{y_i^2}{\lambda-\lambda_i}-1$$

的零点, 它的函数图像大致为(可以采用 Newton 迭代之类的算法求解)



奇异值分解

对不满秩的矩阵A做奇异值分解$A = U\Sigma V^*$, 有以下形式

$$A = [U \quad U_\perp]\begin{bmatrix}\Sigma\\0\end{bmatrix}V^*$$

通常可以省略$U_\perp$, 得到精简版的奇异值分解.

算法 1 对称 QR 算法

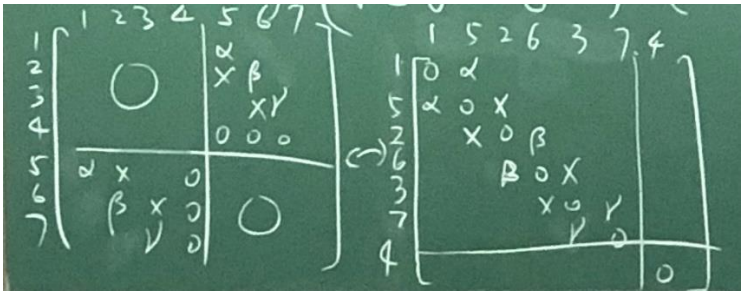
对任意矩阵A, 可通过一系列 Householder 变换, 使得$B = Q_1^* A Q_2$是一个上二对角矩阵. 由于B^*B是一个三对角矩阵, 且$BB^* = U\Sigma\Sigma^*U^*$, 对它做 Wilkinson 隐式位移即可得到其所有特征值, 也即A的奇异值的平方.

算法 2 Golub-Kahan 算法

B阵的构造与算法 1 中提到的相同, 即B是上二对角矩阵, 且

$$\begin{bmatrix}0 & B\\B^* & 0\end{bmatrix} = \begin{bmatrix}0 & U\Sigma V^*\\V\Sigma^*U^* & 0\end{bmatrix} = \begin{bmatrix}U & \\ & V\end{bmatrix}\begin{bmatrix}0 & \Sigma\\\Sigma^* & 0\end{bmatrix}\begin{bmatrix}U & \\ & V\end{bmatrix}^*$$

对$\begin{bmatrix}0 & \Sigma\\\Sigma^* & 0\end{bmatrix}$进行行列重排, 可把矩阵的对角元变换为$\begin{bmatrix}0 & \alpha\\\alpha & 0\end{bmatrix}$型的特征子矩阵, 于是该矩阵的特征值即原始矩阵$A$的奇异值, 由于该矩阵的特征值成对出现, 故可采用 Francis 隐式双位移.



注 相较于对称 QR 算法, Golub-Kahan 算法的优势在于避免了直接计算B^*B, 从而有效减小了数值误差.

算法 3 单边 Jacobi 算法

即对$M = A^*A$实施隐式 Jacobi 算法来计算A的奇异值. 由于每一步 Jacobi 变换均为

$$A^TA \leftarrow J^TA^TAJ$$

故实际操作中只需计算AJ即可.

注 该算法的优越性在于避免了矩阵双对角化产生的数值误差, 缺陷在于时间复杂度大.

特征值/奇异值的应用

桥梁共振、图片压缩等

SEP/SVD & GEP/GSVD

SEP(特征值问题)	$Ax = x\lambda$
GEP(广义特征值问题)	$Ax = Bx\lambda$
SVD(奇异值问题)	$A = U\Sigma V^* \Leftrightarrow A^*Av = v\sigma^2$
GSVD(广义奇异值问题)	$A = U\Sigma_1 X^{-1}, B = V\Sigma_2 X^{-1} \Leftrightarrow A^*Aw = B^*Bw\sigma^2$

Sylvester 方程　　Bartels－Stewart 算法

解形如$AX - XB = C$的方程,该方程被称为 Sylvester 方程.该方程等价于

$$(I\otimes A - B^T\otimes I)vec\ X = vec\ C$$

直接解这一方程的时间复杂度是$O(m^3n^3)$的,进一步有基于实 Schur 分解的优化,即做分解$A = Q_1T_1Q_1^*, B = Q_2T_2^TQ_2^*$,则上面的方程化为

$$Q_1^*Q_1T_1(Q_1^*XQ_2) - (Q_1^*XQ_2)T_2^TQ_2^*Q_2 = Q_1^*CQ_1$$

即

$$T_1Y - YT_2^T = \tilde{C}$$

其中$Y = Q_1^*XQ_2, \tilde{C} = Q_1^*CQ_2$.算法的时间复杂度是$O(m^3 \vee n^3)$.

矩阵函数

算法 1　Schur-Parlett 算法

对于任意矩阵A,做实 Schur 分解得$A = QTQ^*$,则

$$f(A) = Qf(T)Q^*$$

令$F = f(T)$,则$F_{ii} = f(T_{ii})$,且由于矩阵函数的可交换性($Tf(T) = f(T)T$),可得

$$\begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}\begin{bmatrix} F_{11} & F_{12} \\ & F_{22} \end{bmatrix} = \begin{bmatrix} F_{11} & F_{12} \\ & F_{22} \end{bmatrix}\begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}$$

即有

$$T_{11}F_{12} + T_{12}F_{22} = F_{11}T_{12} + F_{12}T_{22}$$

上式可以化为 Sylvester 方程的形式

$$T_{11}F_{12} - F_{12}T_{22} = C$$

其中$C = F_{11}T_{12} - T_{12}F_{22}$.

算法 2　有理逼近算法

当T_{11}和T_{22}中含有相近特征值时,上述 Sylvester 方程条件数较大.我们可以通过 Schur 型特征值交换算法来使得相近的特征值位于同一对角块中,对于对角块使用有理函数逼近来计算,其余部分则可以使用上述 Schur-Parlett 算法计算.由于

$$f(\lambda) \approx r(\lambda) = \frac{p(\lambda)}{q(\lambda)}$$

这里$r(\lambda)$由某处的 Padé 逼近确定,则

$$f(T) \approx p(T)q(T)^{-1}$$

算法 3 scaling-and-squaring 算法(针对指数函数)

为了求解$f(A)=e^A$,对指数做二分,即有

$$e^A=\left(e^{\frac{A}{2^k}}\right)^{2^k}$$

在$\left\|\frac{A}{2^k}\right\|\to 0$时,对其做有理逼近,即有

$$e^{\frac{A}{2^k}}\approx r\left(e^{\frac{A}{2^k}}\right)$$

这一逼近可以使用 Padé 逼近或 Taylor 展开完成(Matlab 中有函数库 expm(A)完成这一算法).

下面考虑求解三种形式的矩阵函数问题,其中给定的$A\in\mathbb{C}^{n\times n}$均为大型稀疏对称阵.

- 求形如$f(A)v$的值. 由 Lanczos 过程,有

$$AQ_k=Q_kT_k+rank\ 1$$

其中选定$q_1=Q_ke_1=\frac{v}{\|v\|_2}$,则上式近似可认为

$$AQ_k\approx Q_kT_k$$

进一步,有

$$A^nQ_k\approx Q_kT_k^n$$

于是我们得到

$$f(A)Q_k\approx Q_kf(T_k)$$

此时对T_k做相似对角化$T_k=U_k\Theta_kU_k^*$,则$f(T_k)=U_kf(\Theta_k)U_k^*$,故

$$f(A)v=f(A)Q_ke_1\approx Q_kf(T_k)e_1\|v\|_2$$

- 求形如$v^*f(A)v$的值. 事实上,有

$$v^*f(A)v=\|v\|_2^2(e_1^*Q_k^*f(A)Q_ke_1)\approx\|v\|_2^2(e_1^*f(T_k)e_1)$$

- 求形如$u^*f(A)v$的值. 由极化恒等式,有

$$4u^*f(A)v=(u+v)^*f(A)(u+v)-(u-v)^*f(A)(u-v)$$

于是此问题转化为上面的$v^*f(A)v$型问题.

矩阵的存储格式

算法 0 直接存储

直接存储矩阵，存储效率为$O(mn)$.

算法 1 COO format

对于稀疏矩阵，考虑只记录非零元素的位置和值，即以下表存储矩阵

$rows$	$cols$	$values$
…	…	…

存储效率为$O(3z)$(z为矩阵中非零元素个数)

算法 2 CSC format

对 COO 存储格式进行进一步优化，按列顺序依次存储非零元素，这样就无需记录每个元素的列坐标，而是用一个指针标定每一列开始的位置即可. 例如，存储矩阵

$$\begin{bmatrix}1 & 0 & 2\\ 0 & 0 & 3\\ 4 & 5 & 6\end{bmatrix}$$

时，只需以下三个数组：

$$cols\ pointer：[0,2,3,6]$$
$$rows\ index：[0,2,2,0,1,2]$$
$$values：[1,4,5,2,3,6]$$

存储效率为$O(n+2z)$，一般情况下 CSC 格式略优于 COO 格式.

算法 3 CSR format

与 CSC 存储格式完全一致，只是改为存储$rows\ pointer$和$cols\ index$.

大规模矩阵问题

基本思路：转化为$x \to Ax$的问题(因为该操作可以尽可能保留矩阵的稀疏性质). 例：

1. **解线性方程组$Ax=b$**

$$x = A^{-1}b = p(A)b = \sum_{i=0}^{n-1} a_i A^i b$$

2. **解A的特征值**

由于$A = Q\Lambda Q^*$中Q有可能变为满矩阵，故尽可能避免矩阵分解，采用A的子空间近似(即乘幂法).

拉普拉斯算子

定义拉普拉斯算子$\Delta u = f$, 它的意义即为微分方程

$$\frac{\partial^2}{\partial x^2}u + \frac{\partial^2}{\partial y^2}u = f(x,y)$$

考虑u的二阶导的离散形式, 由于

$$u(x \pm h) = u(x) \pm u'(x)h + \frac{u''(x)}{2}h^2 + \cdots$$

故

$$u''(x) = \frac{u(x+h) + u(x-h) - 2u(x)}{h^2} + O(h^2)$$

在二维矩阵上, 有

$$u''(x,y) = \frac{u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) - 4u(x,y)}{h^2} + O(h^2)$$

表示矩阵为

$$A = \frac{1}{h^2}(I_n \otimes T_m + T_n \otimes I_m)$$

我们给定矩阵的边界元素$u|_{\partial u} = u_0$, 在$f = 0$(即$u''(x,y) = 0$)的特殊情况下, 有

$$u(x,y) = \frac{u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h)}{4}$$

则可以通过不断循环计算矩阵内元素

$$u^{(k+1)}(x,y) = \frac{u^{(k)}(x+h,y) + u^{(k)}(x-h,y) + u^{(k)}(x,y+h) + u^{(k)}(x,y-h)}{4}$$

最终即可使矩阵内元素收敛到同一值, 这符合热力学定律, 我们称该做法为**简单迭代法**.

分裂型迭代法

考虑解线性方程组$Ax = b$, 其中A是一个难以求逆的矩阵. 设$A = M - N$, 其中$M \approx A$被称为 preconditioner, 使得M是容易求逆的矩阵. 且$\rho(M^{-1}N) < 1$, 则原方程转化为

$$x = M^{-1}Nx + M^{-1}b$$

对x进行迭代

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b$$

x的收敛值即原方程$Ax = b$的解.

例. 给定对称正定阵A, 求$\lambda_1(A)$.

解. 反幂法:$x_{k+1} = A^{-1}x_k$. 设定参量$r_k = Ax_k - x_k\lambda_k$, 则

$$A^{-1}r_k = x_k - A^{-1}x_k\lambda_k$$

于是

$$x_{k+1} = A^{-1}x_k\lambda_k = x_k - A^{-1}r_k$$

做预条件处理$A \approx M$, 则

$$x_{k+1} = x_k - M^{-1}r_k$$

设$A=D-L-U$,其中D是对角阵,L是下三角阵,U是上三角阵(其中L,U的对角元均为零),则

算法 1 Jacobi 迭代法

此法中令$M=D$,迭代式为

$$Dx^{(k+1)}=(L+U)x^{(k)}+b$$

以$A\in\mathbb{C}^{3\times3}$为例,迭代式为

$$\begin{cases}x_1^{(k+1)}=\dfrac{b_1-a_{12}x_2^{(k)}-a_{13}x_3^{(k)}}{a_{11}}\\ x_2^{(k+1)}=\dfrac{b_1-a_{21}x_1^{(k)}-a_{23}x_3^{(k)}}{a_{22}}\\ x_3^{(k+1)}=\dfrac{b_1-a_{31}x_1^{(k)}-a_{32}x_2^{(k)}}{a_{33}}\end{cases}$$

下面说明保证$\rho\left(D^{-1}(L+U)\right)<1$的必要性. 若不然,则存在$|\lambda|\geq1$,使得

$$D^{-1}(L+U)x=x\lambda$$

可推得

$$(\lambda D-L-U)x=0$$

由于$\lambda\geq1$,故在A是严格对角占优的条件下,$\lambda D-L-U$是严格对角占优的,则上述方程只有唯一的零解$x=0$,即λ不是特征值,推出矛盾. 故此时使用 Jacobi 迭代法是错误的.

算法 2 Gauss-Seidel 迭代法

此法中令$M=D-L$,迭代式为

$$(D-L)x^{(k+1)}=Ux^{(k)}+b$$

以$A\in\mathbb{C}^{3\times3}$为例,迭代式为

$$\begin{cases}x_1^{(k+1)}=\dfrac{b_1-a_{12}x_2^{(k)}-a_{13}x_3^{(k)}}{a_{11}}\\ x_2^{(k+1)}=\dfrac{b_1-a_{21}x_1^{(k+1)}-a_{23}x_3^{(k)}}{a_{22}}\\ x_3^{(k+1)}=\dfrac{b_1-a_{31}x_1^{(k+1)}-a_{32}x_2^{(k+1)}}{a_{33}}\end{cases}$$

下面说明保证$\rho((D-L)^{-1}U)<1$的必要性. 若不然,则存在$|\lambda|\geq1$,使得

$$(D-L)^{-1}Ux=x\lambda$$

可推得

$$(\lambda D-\lambda L-U)x=0$$

由于$\lambda\geq1$,故在A是严格对角占优的条件下,$\lambda D-\lambda L-U$是严格对角占优的,则上述方程只有唯一的零解$x=0$,即λ不是特征值,推出矛盾. 故此时使用 G-S 迭代法是错误的.

超松弛迭代 SOR

考虑迭代式

$$x^{(k+1)} = Bx^{(k)} + g = x^{(k)} + (B - I)x^{(k)} + g$$

即

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$$

进一步加入超松弛参数$\omega \in [1,2)$*(相应地,低松弛参数为*$\omega \in (0,1)$*)*,则

$$x^{(k+1)} = x^{(k)} + \omega \Delta x^{(k)}$$

在合适的参数下该迭代的收敛速度很快,SOR 优化下的 G-S 迭代为

$$x^{(k+1)} = x^{(k)} + \omega\left[(D - L)^{-1}Ux^{(k)} + (D - L)^{-1}b\right]$$

Krylov 子空间迭代

定义 Krylov 子空间$K_m(A,r) = span\{r, Ar, \dots, A^{m-1}r\}$. 考虑解线性方程组$Ax = b$,其中定义残差$r = b - Ax_0$,则

$$x = x_0 + A^{-1}r$$

又$A^{-1} = p(A)$,故

$$x = x_0 + p(A)r \in x_0 + K_n(A,r)$$

进一步地,我们希望降低 Krylov 子空间的迭代次数,即尽可能最小化k,使得

$$x_k \in x_0 + K_k(A,r)$$

为此有 **GMRES(Generalized minimal residual)算法(1985)**:

$$x_k \triangleq \arg\min\|b - Ax\|_2 \left(x \in x_0 + K_k(A,r)\right)$$

具体来说,设$K_k(A,r) = span\ Q_k$,则

$$x_k = x_0 + Q_k y_k$$

于是

$$\min_{\mathrm{x}}\|b - Ax\|_2 = \min_{\mathrm{y_k}}\|b - Ax_0 - AQ_k y_k\|_2 = \min_{\mathrm{y_k}}\|r - AQ_k y_k\|_2$$

由二范数的酉不变性,上式等价于

$$\min_{\mathrm{y_k}}\|Q_{k+1}^*(r - AQ_k y_k)\|_2$$

其中Q_k通过 Arnoldi 过程得到,即有$q_1 = \frac{r}{\|r\|_2}$,由正交性可得$Q_{k+1}^* r = e_1\|r\|_2$. 以及 Arnoldi 过程可导出性质$Q_{k+1}^* AQ_k = H_{k+1}$,故上式化为

$$\min_{\mathrm{y_k}}\|e_1\|r\|_2 - H_{k+1}y_k\|_2$$

即当$\|e_1\|r\|_2 - H_{k+1}y_k\|_2 < eps$时认为已经达到收敛. 此时可以确定$y_k$,进而解得$x_k$.

解这个最小二乘问题可以通过 Givens 变换求解,以四阶矩阵为例:

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ & * & * & * \\ & & * & * \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \|r\|_2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

做 Givens 变换,得到

$$\begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ & 0 & * & * \\ & & 0 & * \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} * \\ * \\ * \\ * \end{bmatrix}$$

其中最后一行无法变动,从最后一行起向前回代即可得到要求的y_k.

注 该算法实际上是用精度换时间.

GMRES 的 lucky breakdown

考虑恰好收敛到 0 的条件,有

$$AQ_k = Q_k H_k$$

结合$x_k = x_0 + Q_k y_k$,得到

$$b - Ax_k = b - Ax_0 - Q_k H_k y_k = r_0 - Q_k H_k y_k = 0$$

这表明

$$\begin{cases} r_0 = Q_k e_1 \|r_0\|_2 \\ y_k = H_k^{-1} e_1 \|r_0\|_2 \end{cases}$$

GMRES 的优化方向

1. 预条件

设$A = M - N$,其中$M \approx A$,则称M为A的**预条件因子**. 这时求解$Ax = b$相当于求

$$x = M^{-1}(b + Nx)$$

或相当于$(AM^{-1})(Mx) = b \Leftrightarrow (M^{-1}A)x = (M^{-1}b)$,则对$M^{-1}A$做 Krylov 子空间迭代,很大概率有

$$K(M^{-1}A) \ll K(A)$$

2. 重启动

在迭代到一定步数后,若仍未收敛,则选定此时的x为初始向量,重新开始一轮 Krylov 子空间迭代.

FOM 算法(1975)

优化 Arnoldi 过程为

$$AQ_k = Q_k H_k + rank1$$

则收敛时有

$$(b - Ax_k) \perp Q_k$$

MINRES 算法(1975)　　SYMMLQ 算法(1976)

当$A^* = A$时,Arnoldi 过程变为

$$AQ_k = Q_{k+1}T_{k+1}$$

在这种情况下,有 GMRES 的优化 MINRES 算法,以及 FOM 算法的优化 SYMMLQ 算法.

在 GMRES 算法中,考虑残差的平方

$$R^2 = \|b - Ax\|_2^2 = x^T A^T Ax - 2x^T A^T b + b^T b$$

取得最小值时,有

$$\frac{\partial R^2}{\partial x} = 2A^T Ax - 2A^T b = 0$$

令$x_* = A^{-1}b$,考察残差的等价形式

$$\phi(x) = \|x - x_*\|_A^2 = \langle x - x_*, x - x_* \rangle_A = \varphi(x) + \langle x_*, x_* \rangle_A$$

其中函数$\varphi(x)$定义为

$$\varphi(x) = x^T Ax - 2x^T b = \langle x, x \rangle_A - 2\langle x, A^{-1}b \rangle_A$$

于是要求$\phi(x)$的极小值,只需求$\varphi(x)$的极小值.

最速下降法(Steepest Descent)

对于$A^* = A \succ 0$,求二次函数$\varphi(x) = x^T Ax - 2x^T b$的极小值.考虑其梯度

$$\nabla f = grad\ f = \frac{\partial \varphi}{\partial x} = 2Ax - 2b$$

当A为对称正定阵时,二次函数$\varphi(x)$的极小值点必定存在,且恰为驻点.于是

$$f(x_0) = \min f(x) \Leftrightarrow \nabla f(x_0) = 2Ax_0 - 2b = 0$$

为通过迭代找到符合条件的x_0,选定初值x_0,并计算残差$r_0 = b - Ax_0$.在第k步迭代中,选定方向$p_k = -\nabla f = r_k$,做一步迭代

$$x_{k+1} = x_k + \alpha_k p_k$$

其中步长α_k由当前方向p_k的最低点决定,即有

$$\left.\frac{df}{d\alpha_k}\right|_{x_{k+1}} = \nabla f|_{x=x_{k+1}} \cdot p_k = \langle r_{k+1}, p_k \rangle = 0$$

由于$r_{k+1} = b - Ax_{k+1} = b - Ax_k - \alpha_k Ap_k = r_k - \alpha_k Ap_k$,故上式也即

$$\langle r_k - \alpha_k Ap_k, p_k \rangle = 0 \Longrightarrow \alpha = \frac{\langle r_k, p_k \rangle}{\langle p_k, p_k \rangle_A} = \frac{r_k^T r_k}{r_k^T Ar_k}$$

收敛过程中,由于

$$r_{k+1} = r_k - \alpha_k Ar_k = (I - \alpha_k A)r_k$$

下面考虑$g(\lambda) = 1 - \alpha\lambda$,则$g(A)$可以正交对角化,即

$$g(A) = Qg(\Lambda)Q^*$$

于是$\|r_{k+1}\|_2 = \|g(A)r_k\|_2 = \|g(\Lambda)r_k\|_2 \leq \|g(\Lambda)\|_2\|r_k\|_2$, 即我们希望取

$$\min_{\alpha} \max\{|g(\lambda_1)|, |g(\lambda_n)|\}$$

由于特征值的分布是一次函数, 则取等时, 有$1-\alpha\lambda_1 = -1+\alpha\lambda_n \Rightarrow \alpha = \frac{2}{\lambda_1+\lambda_n}$, 也即

$$\|g(\Lambda)\|_2 = 1-\alpha\lambda_1 = \frac{\lambda_n-\lambda_1}{\lambda_n+\lambda_1} = \frac{\kappa(A)-1}{\kappa(A)+1} \Rightarrow \|r_k\|_2 \leq \left(\frac{\kappa(A)-1}{\kappa(A)+1}\right)^k \|r_0\|_2$$

共轭梯度法(Conjugate Gradient)(1952)

在最速下降法中, 每一步迭代为$x_{k+1} = x_k + \alpha_k p_k$, 则

$$x_k = x_0 + \sum_{i=0}^{k-1} \alpha_i p_i$$

那么当p_i彼此A共轭时, 它们也彼此正交. 选定$x_0 = 0$, 于是我们有

$$x = \sum_{i=0}^{k-1} \alpha_i p_i \Rightarrow b = Ax = \sum_{i=0}^{k-1} \alpha_i A p_i$$

两边同乘p_k^T, 由p之间的正交性, 即得

$$\boxed{p_k^T b = \alpha_k p_k^T A p_k \Rightarrow \alpha_k = \frac{\langle p_k, b\rangle}{\langle p_k, p_k\rangle_A} = \frac{p_k^T b}{p_k^T A p_k}}$$

从而下面只需确定一组共轭向量组p_k即可. 初始时选定$p_0 = r_0$, 并设一组由残差组成的k维向量组$\{r_1, r_2, \ldots, r_k\}$, 那么在$A$内积下, 由 Gram-Schmidt 过程, 有

$$p_j = r_j - \sum_{i=0}^{j-1} \frac{\langle p_i, r_j\rangle_A}{\langle p_i, p_i\rangle_A} p_i$$

写成迭代形式, 即

$$p_{k+1} = r_{k+1} - \beta_k p_k$$

其中$\beta_k = \frac{\langle p_k, r_{k+1}\rangle_A}{\langle p_k, p_k\rangle_A}$. 由于实际实践过程中是在$k$维子空间中做近拟, 故不可直接使用上面的框中步长进行迭代. 而应当使用最速下降法中计算的迭代步长进行迭代, 即

$$\alpha_k = \frac{\langle r_k, p_k\rangle}{\langle p_k, p_k\rangle_A} = \frac{r_k^T p_k}{p_k^T A p_k}$$

$$x_{k+1} = x_k + \alpha_k p_k$$

通过一些稀奇古怪的推导(好像跟 Chebyshev 数列有关)得到

$$\|r_k\|_A \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k \|r_0\|_A$$

这表明共轭梯度法的收敛速度远快于最速下降法.

8 其他稀疏矩阵算法

8.1 特征值

本小节假设 $A = A^*$.

1. Rayleigh–Ritz 投影: 设有一组正交基 $V \in \mathbb{C}^{n\times k}$, 如果存在复数 θ, 向量 $\tilde{x} \neq 0$, 使得
$$V^*(A\tilde{x} - \theta\tilde{x}) = 0, \quad \tilde{x} = Vy,$$
则 θ 被称为 Ritz 值, $\tilde{x}$ 被称为 Ritz 向量.

2. Lanczos 算法: 该算法一般用于求解 A 模较大的特征值. 算法使用 Lanczos 过程计算得到 $AV_m = V_mT_m + \beta_m v_{m+1}e_m^*$, 然后计算 T_m 模较大的特征值 θ 以及对应的特征向量 y, 进而将 (θ, V_my) 作为矩阵 A 的近似特征对.

3. FEAST 算法: 该算法可以用于求解 A 在给定区域的特征值. 如图 1 所示, 求

图 1: FEAST 算法示意图

解区间 (α, β) 内的特征值, 先用一条曲线 Γ 包围 (α, β), 根据复变函数的知识可以得到
$$f(A) = \frac{1}{2\pi \mathrm{i}} \int_\Gamma (zI - A)^{-1}\,\mathrm{d}z = Q_{\text{in}}Q_{\text{in}}^*,$$
其中 Q_{in} 是 A 位于区间 (α, β) 内的特征值对应的特征向量. 围道积分的可以通过梯形积分公式或者 Gauss 积分公式来计算, 近似得到 $f(A)$ 后便可以通过子空间迭代得到特征值以及特征向量.

4. Davidson 算法: 设有一组初始正交基 V, 通过 Rayleigh–Ritz 投影得到一组 Ritz 值 Θ 和 Ritz 向量 Q, 算法计算残量
$$R = AQ - Q\Theta,$$
将 $[Q, R]$ 作为下一次迭代的初始向量.

5. PINVIT: 反幂法 $x_{k+1} = A^{-1}x_k$ 可以改写成
$$x_{k+1} = x_k - A^{-1}r_k, \quad r_k = Ax_k - \lambda_k x_k,$$
其中 λ_k 是第 k 步的 Rayleigh 商. PINVIT 用一个容易求解的矩阵 M 代替迭代式中的 A, 即
$$x_{k+1} = x_k - M^{-1}r_k.$$
当 M 满足一定条件时, 上述算法是收敛的.

8.2 最小二乘问题和奇异值算法

Paige–Saunders 二对角化: 设 u_1 为单位向量, 设 $\alpha_1 v_1 = A^* u_1$, 其中 α_1 是归一化系数, 则整个过程为

1. $\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i$
2. $\alpha_{i+1} v_{i+1} = A^* u_{i+1} - \beta_{i+1} v_i$

最后得到的向量满足

$$AV_k = U_{k+1} B_k, \quad A^* U_k = V_k \tilde{B}_k^*,$$

其中

$$B_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix},$$

$\tilde{B}_k$ 是 B_k 的前 k 行组成的矩阵. 上述过程可以视为对矩阵

$$\begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix}$$

初始向量选为 $[u_1^*, 0]^*$, 进行 Lanczos 过程.

求解最小二乘问题 $\min_x \|Ax - b\|_2$, 可以令 $u_1 = b/\|b\|_2$, 利用上述过程可以将问题转化为求解

$$\min_y \|AV_k y - b\|_2 = \min_y \|B_k y - \|b\| e_1\|_2.$$

若要求解 A 的奇异值, 则可以用 B_k 的奇异值代替.