

Lecture 13 大型稀疏矩阵的特征值问题 — 2025.12.08

教授：邵美悦

Scribe: 路人甲

目录

1 子空间迭代法	1
2 Lanczos 过程	4
2.1 Krylov 子空间方法	4
2.2 Lanczos 过程	5
2.2.1 朴素的 Lanczos 过程	5
2.2.2 收缩 (Deflation)	7
2.2.3 Krylov-Schur 方法	9

1 子空间迭代法

大型稀疏矩阵的特征值计算是比较困难的。如果我们使用传统的特征值算法（如 QR 算法）计算 Hermite 矩阵 A 的谱分解 $A = Q\Lambda Q^*$ ，即使 A 很稀疏，酉矩阵 Q 仍然有可能是稠密矩阵——这就使得我们很难将所有特征值和特征向量都计算出来并存储下来。仅举一例：

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

这个矩阵我们非常熟悉，它是两个 Frobenius 友阵的和。它的特征向量稠密到没有一个非零元。所以想要把所有特征值和特征向量都计算出来是很困难的。假如真有这样的需求，我们也只好假装不知道它是稀疏矩阵，直接使用传统的特征值算法来计算。但是在很多实际问题中，我们并不需要所有的特征值和特征向量，而只需要其中的一部分。这里的“一部分”也有难易之分：计算最大和

最小的几个特征值和对应的特征向量相对容易，这些特征值被称作极值特征值；而计算中间的特征值和对应的特征向量则要困难得多。我们接下来就来讨论一些计算大型稀疏矩阵部分特征值和特征向量的方法。

回顾一下我们之前学过的特征值算法——乘幂法或许比较适合用来计算大型稀疏矩阵的最大特征值——因为乘幂法的主要操作就是矩阵-向量乘法，而大型稀疏矩阵的矩阵-向量乘法可以很高效地实现。那么最小的特征值呢？我们只需要实现做一步位移 $\tilde{A} = A + \theta I$ ，使得最小特征值的模超过最大特征值，我们就可以再度利用乘幂法将 $\lambda_1(A) + \theta$ 算出来，最后减去 θ 就能够得到最小特征值了。所以在数学本质上，最小特征值和最大特征值并没有什么区别；当然做了位移可能会带来一定的数值误差，但通常不会很大，我们就不做考虑了。

但是乘幂法最多只能算出两端的特征值，对于第二个、第三个的特征值就无能为力了。因此，我们需要子空间迭代法来计算多个特征值。子空间迭代法可以看作是乘幂法在高维空间的推广。假设矩阵 A 的特征值满足：

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_5| \gg |\lambda_6| \geq \cdots \geq |\lambda_n|.$$

特征值整体按降序排列，但在 λ_5 和 λ_6 之间有一个大间隔 (gap)。现在，假如我们想要计算前三个特征值，我们可以随机生成一个 $n \times 3$ 的矩阵 X_0 ，然后不断迭代：

$$\begin{aligned} Z_{k+1} &= AX_k, \\ Z_{k+1} &= X_{k+1}R_{k+1} \quad (\text{QR 分解}). \end{aligned}$$

经过多次迭代后，矩阵 X_k 的列空间就会收敛到矩阵 A 的前 3 个特征向量所张成的子空间上。然后我们可以通过 Rayleigh 商来近似计算出对应的特征值。

但是，这样计算出来的特征值往往收敛得比较慢。这是一个线性收敛的算法，收敛速度取决于 $|\lambda_4/\lambda_3|$ 。假如 λ_4 和 λ_3 贴得很近，算法就会收敛得非常慢。注意到 λ_5 和 λ_6 之间有一个很大的间隔，我们不妨一次性算五个特征值，这样前三个特征值的收敛速度就取决于 $|\lambda_6/\lambda_3|$ 。这是一个小量，算法的会收敛得特别快。

但在实际计算中可能还是会有一些问题：特征值要怎么算？特征值自然是通过 Rayleigh 商来计算的。但是，我们最终得到的矩阵 X 很可能并不是特征向量构成的矩阵，只是满足 $\text{span}(X) = \text{span}(Q)$ 。换句话说，我们只是得到了特征子空间的一组基，但这些基向量本身**并不是特征向量的良好近似**，可能含有其他方向的成分。我们还需要进一步处理，才能得到比较准确的特征值。一种常用的方法是 Rayleigh-Ritz 投影。假设 $X \in \mathbb{C}^{n \times m}$ 。我们考虑 $x \in \text{span}(X)$ ，我们希望它是特征向量，即

$$Ax = xR(A, x) = x \cdot \frac{x^* Ax}{x^* x}.$$

既然 $x \in \text{span}(X)$ 且 X 是正交归一的矩阵，我们不妨设 $x = Xy$ ，其中 $y \in \mathbb{C}^m$ 是坐标向量。对于任意给定的一个向量 x ，上式几乎不可能满足，肯定会产生一个残差 $r = Ax - x\lambda$ 。现在，我们希望这个残差尽可能地小，该怎么办呢？我们引入 Galerkin 条件： $r \perp \text{span}(X)$ 。正交性保证了残

量在我们所关心的子空间内“看起来”为零，这在某种程度上也算将残差最小化了。于是，我们有

$$X^*r = X^*(Ax - x\lambda) = X^*AXy - y\lambda = 0.$$

于是，我们只要解

$$(X^*AX)y = y\lambda.$$

矩阵 $X^*AX \in \mathbb{C}^{m \times m}$ 的特征值是非常好解的：因为它是一个小矩阵，我们可以直接用 QR 算法或分治法把特征值算出来。或许有人会疑问：矩阵乘法 X^*AX 我们能承受得起吗？答案是可以的。 X 一般是小矩阵， m 一般远远小于 n ，可视为常数；于是 AX 的复杂度就是 m 倍的稀疏矩阵乘向量的复杂度，再和 X 作内积，复杂度也可控。因此，这个开销我们是可以接受的。解出 y 后回代 $x = Qy$ 就得到了 λ 对应的特征向量。我们把 λ 称为 Ritz 值， x 称为 Ritz 向量， (λ, x) 称为 Ritz 对。我们如何判断 Ritz 对的质量呢？最直接的检验方法就是把残差算出来，如果残差比较小，我们就认为我们算得足够好了。

从另一个角度，Rayleigh–Ritz 投影相当于把 A 投影到子空间 $\text{span}(X)$ 上，然后在这个子空间内以 X 为基求解特征值。得到的结果和上面是一样的。

综上，带有 Rayleigh–Ritz 投影的子空间迭代法的迭代步骤如下：

$$\begin{aligned} Z_{k+1} &= AX_k \\ Z_{k+1} &= Q_{k+1}R_{k+1} && \text{QR 分解} \\ Q_{k+1}^*AZ_{k+1} &= U_{k+1}\Theta_{k+1}U_{k+1}^* && \text{谱分解} \\ X_{k+1} &= Q_{k+1}U_{k+1}. \end{aligned}$$

使用这种算法的好处在于：使用传统的子空间迭代法可能无法将靠得很近的特征值分开，但使用 Rayleigh–Ritz 投影后，因为我们对小矩阵用的是 QR 算法，这些靠得很近的特征值就可以被很好地分开了。如图 1 所示，矩阵 A 有两个非常靠近的特征值 $\lambda_1 = 20$ ， $\lambda_2 = 19.9$ 。使用不带 Rayleigh–Ritz 投影的子空间迭代法的残差会停留在 10^{-3} 级别，而带有 Rayleigh–Ritz 投影的子空间迭代法的残差则可以降到 10^{-14} 级别。

但是，又一个问题在于，我们事先并不知道特征值，也无法确定哪里会有间隔，我们怎么可能知道要算多少个特征值才能跨过一个间隔呢？尽管有时候我们可以利用对称性估计特征值的分布，但更多时候我们是无法知道的。那这个时候我们又该怎么办呢？如果一定要使用子空间迭代法的话，那我们只好多算几个，赌它一定跨过间隔了。

但更坏的是，如果特征值比较聚集，不存在明显的间隔，但又没有聚集到可以完全齐同视之的地步，又该怎么办呢？我们只好另求他法了。

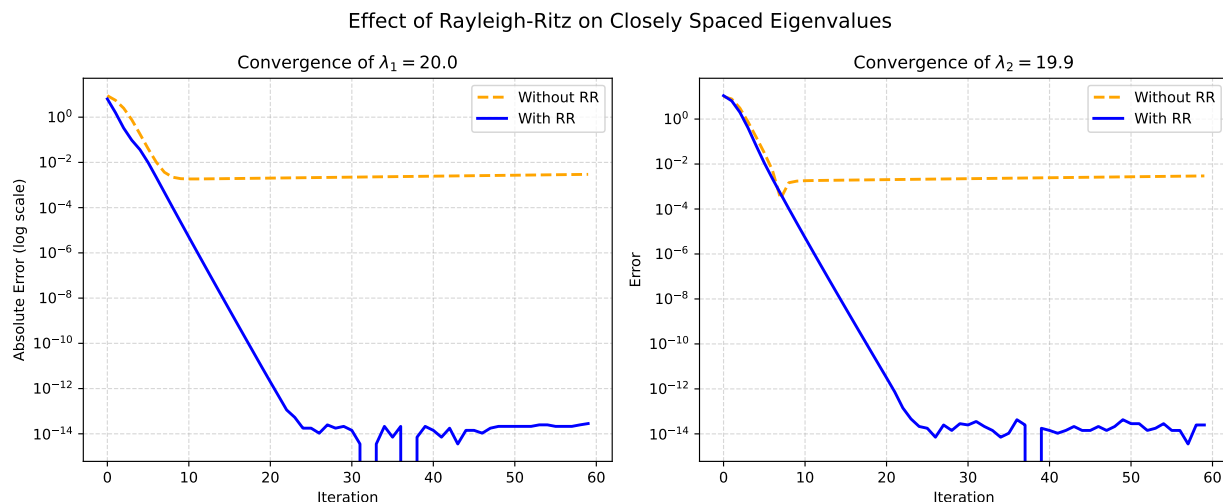


图 1: Rayleigh–Ritz 投影对靠得很近的特征值的分离效果

2 Lanczos 过程

2.1 Krylov 子空间方法

我们回到乘幂法。仔细想来，乘幂法是一个比较短视的算法：它每次只看当前的向量 x_k ，然后计算下一个向量 x_{k+1} 。于是，前 k 步得到的向量张成的空间是

$$\text{span}\{x_0, x_1, \dots, x_{k-1}\} = \text{span}\{x_0, Ax_0, \dots, A^{k-1}x_0\} = \mathcal{K}_k(A, x_0).$$

按理来说，既然这是一个 Krylov 子空间，我们完全可以在 $\mathcal{K}_n(A, x_0)$ 中找到特征向量，因为由 Cayley–Hamilton 定理， A^n 乃至更高次的矩阵幂都可以由 I, A, \dots, A^{n-1} 线性组合得到。可是，由于乘幂法只利用了上一步的信息 x_{k-1} ，只在 Ax_{k-1} 这一个方向搜索，使得乘幂法一般在 n 步以内无法收敛。我们的改进思路便是每一步都在 Krylov 子空间 $\mathcal{K}_k(A, x_0)$ 中寻找最优解。

一个最直接的优化思路便是：我们每一步都利用 Rayleigh–Ritz 投影计算出 Krylov 子空间的 Ritz 对，作为当前最优解；如果最优解足够好，我们就停止迭代；否则继续。那么正交基要怎么找呢？自然是利用 Arnoldi 过程来构造正交基。具体地

$$AQ_k = Q_{k+1}\tilde{H}_k = Q_k H_k + \text{rank1}.$$

于是

$$Q_k^* A Q_k = H_k + Q_k^* (\text{rank1}) = H_k.$$

于是，我们只需对 H_k 求解特征值问题 $H_k y = y\lambda$ 。

在第 k 步迭代中，我们得到 $Q_k^* A Q_k = H_k = U_k \Theta_k U_k^*$ ，其中 Θ_k 是上三角阵。我们令 $X_k = Q_k U_k$ ，然后计算其残量 $R_k = AX_k - X_k \Theta_k$ 判断是否停机。这和前面是一模一样的。

2.2 Lanczos 过程

2.2.1 朴素的 Lanczos 过程

上述基于 Arnoldi 过程的 Krylov 子空间方法虽然能逐步逼近特征值，但它有一个很大的缺点：每一步都要存储所有的 k 个正交基向量。随着迭代步数 k 的增加，存储开销会越来越大，最终可能会超出内存限制。对于非对称矩阵，我们没什么好方法，只能受着；但对于 Hermite 矩阵，由于生成的上 Hessenberg 矩阵实际上是三对角矩阵，我们可以利用这个性质来节省存储空间。这便是 Lanczos 过程。

Lanczos 过程和上面的方法没有本质区别，但它的收敛性更好：它的极值特征值会快速收敛，而且每步递推只需要前两步的向量。下面我们逐一进行解释。

首先，我们来解释一下为什么 Lanczos 过程只需要前两步的向量。由于 A 是 Hermite 矩阵，Arnoldi 过程生成的上 Hessenberg 矩阵是三对角阵 \tilde{T}_k ，如果我们设

$$\tilde{T}_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \alpha_k & \\ & & & \beta_k & \end{bmatrix}_{(k+1) \times k},$$

那么，Arnoldi 过程的递推关系为

$$Aq_j = \beta_{j-1}q_{j-1} + \alpha_jq_j + \bar{\beta}_jq_{j+1}.$$

我们可以把它改写成

$$\bar{\beta}_jq_{j+1} = Aq_j - \alpha_jq_j - \beta_{j-1}q_{j-1}.$$

这说明，计算 q_{j+1} 只需要用到 q_j 和 q_{j-1} ，不需要用到更早之前的向量了。这大大节省了存储空间。

极值特征值收敛比较快。一个比较直观的解释是 $\mathcal{K}_k(A, x_0) = \mathcal{K}_k(A + \theta I, x_0)$ ，而由前面的讨论，取正的比较大的 θ 可以让最大的几个特征值快速收敛；取负的比较大的特征值可以让最小的几个特征值快速收敛。因此，矩阵 A 的两端的特征值在 Lanczos 过程下都快速收敛。这是比较粗糙的分析。更精细的收敛性分析比较复杂，本课程不作要求。但是中间的特征值收敛性就没有保证了，可能会非常慢。

有关特征值，顺道提一嘴：在 Lanczos 过程中，我们需要计算 $Q_k^* A Q_k = T_k$ 的特征值。那么这个特征值会怎么运动呢？它不会“离谱”，即不会离开原有的谱。这是因为 Cauchy 交错定理保证了 T_k 的特征值会受到 A_k 的控制。比如原矩阵 A 的最大特征值是 100，在子空间 T_k 上算出的特征

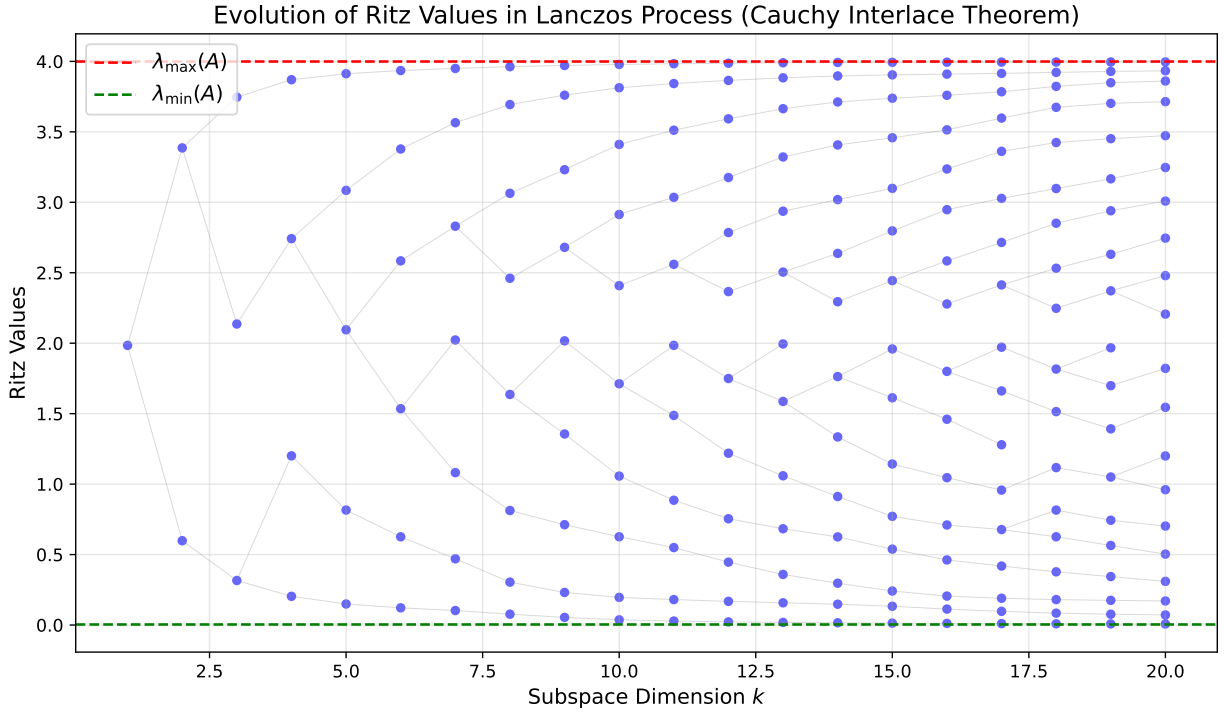


图 2: Lanczos 过程中 Ritz 值的变化

值有可能是 95，但绝无可能是 105. 更具体地，由于 T_k 实际上是 T_{k+1} 的子块，由 Cauchy 交错定理，迭代过程中谱的范围会越来越大，两端快速向矩阵 A 的极值特征值收敛，如图 2 所示。

在高等线性代数课上，我们还证明过一个结论：对任意方阵 A ，假如正交性误差 $\varepsilon = \|Q^*Q - I\|_2$ 很小，那么最终特征值的误差

$$|\lambda_i(Q^*AQ) - \lambda_i(A)| \leq |\lambda_i(A)| \cdot \varepsilon.$$

这说明，只要我们能够保证每一步的正交性误差足够小，最终的特征值误差也会很小。从另一个角度，如果正交性误差比较大，做 Rayleigh–Ritz 投影时，我们本应当计算 $Q^*AQy = Q^*Qy\lambda$ ，但由于 Q^*Q 不是单位阵，因此这个问题不能归为标准的特征值问题，我们却错误地将其当作标准的特征值问题来解了，这就会带来额外的误差。当然，如果我们在做 Rayleigh–Ritz 投影时直接求解广义特征值问题 $Q^*AQy = Q^*Qy\lambda$ ，那就不存在这个问题了，但这样做的计算开销会比较大。所以，不论是从理论上还是从实践上看，保持正交性都是非常重要的。

但不幸的是，Lanczos 算法的正交性维护得并不好。有理论表明，当正交性失去时，会有一个特征值收敛；但其他特征值我们就没法算了。如图 3 所示，我们生成了一个 50 阶矩阵，并可视化了正交性矩阵在迭代 30 步之后各个元素的大小，颜色越黄，误差越大。可以看到，正交性已经变得很差了：下三角部分的误差接近 10^{-1} ， $\|Q_k^*Q_k - I_k\|_F$ 更是达到了 10^0 量级。

为什么会出现这种情况呢？这是因为 Lanczos 过程每一步都只用到了前两步的向量，每一步都产

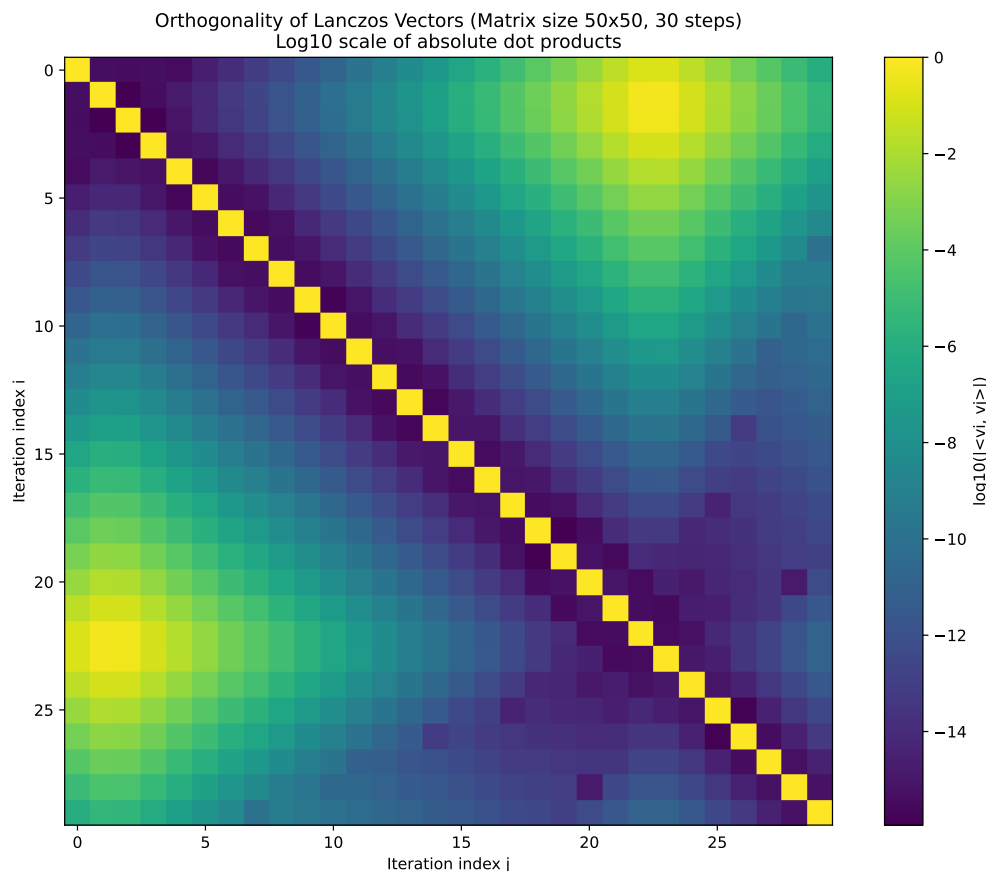


图 3: Lanczos 过程中正交性的丧失

生一些数值误差。这个三项递推不够稳定，误差不断放大，最终导致正交性丧失。正交性丧失后，算法在数值上“忘记”了它已经探索过的方向，Krylov 子空间的有效维数停滞不前，算法会重复收敛到已经找到的特征值上，生成的 Ritz 向量将是原本的已经找到的特征向量的线性组合，这将严重干扰我们对其他特征值的计算。

2.2.2 收缩 (Deflation)

那么我们该如何解决这个问题呢？首先，既然有特征值收敛了，我们就应当尽快把它们从计算中剔除出去，这样就可以避免它们对后续计算产生影响。这个过程被称作收缩 (deflation)。问题来了：怎样判断一个特征值是否已经收敛了呢？最直接的判断方法就是把残差算出来，如果残差足够小，我们就可以认为它已经收敛了。更好的方法是：我们在第 k 步迭代中得到了 $AQ_k = Q_{k+1}\tilde{T}_k$ ，其

中

$$\tilde{T}_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \alpha_k & \\ & & & & \beta_k \end{bmatrix}_{(k+1) \times k}.$$

如果 β_k 是一个小量, 那么 $AQ_k \approx Q_k T_k$, 这意味着我们已经找到了一个近似的不变子空间 Q_k , 对应的特征值就是 T_k 的特征值。具体地, 一方面

$$\begin{aligned} \|AQ_k - Q_k T_k\|_F &= \|AQ_k - Q_k U_k \Theta_k U_k^*\|_F \quad T_k = U_k \Theta_k U_k^* \\ &= \|AQ_k U_k - Q_k U_k \Theta_k\|_F \\ &= \|AX_k - X_k \Theta_k\|_F. \end{aligned}$$

另一方面, 由于

$$AQ_k = Q_k T_k + q_{k+1} \beta_k e_k^*,$$

我们有

$$\|AQ_k - Q_k T_k\|_F = \|\beta_k q_{k+1} e_k^*\|_F = |\beta_k|.$$

于是,

$$\|AX_k - X_k \Theta_k\|_F = |\beta_k|.$$

这说明 β_k 越小, 残差越小, 特征值收敛得越好。因此, 我们可以用 β_k 来判断特征值是否收敛。如果 β_k 足够小, 我们就可以认为对应的特征值已经收敛了, 可以将其从计算中剔除出去。

第二个问题是: 我们该如何剔除已经收敛的特征值呢? 我们有两种做法。

第一种做法的基本思想是: 假设我们已经找到了一个特征对 (λ_1, x_1) . 考虑 A 的谱分解的算子形式 $A = \sum_{i=1}^n \lambda_i x_i x_i^*$. 既然第一个特征对已经找到了, 剩下的特征对就在其正交补空间内, 即我们的搜索空间应该要变成 $A - \lambda_1 x_1 x_1^*$. 这样就把第一个特征对给撇掉了。于是, 我们可以在每一步迭代中都对矩阵 A 做这样的修正, 把已经找到的特征对都剔除出去。假设在第 k 步迭代中, 我们已经找到了 r 个特征对 $(\lambda_1, x_1), \dots, (\lambda_r, x_r)$. 那么我们可以定义修正后的矩阵

$$A_k = A - \sum_{i=1}^r \lambda_i x_i x_i^*.$$

然后在每一步迭代中使用 A_k 代替 A 来进行计算。这样就可以避免已经找到的特征对对后续计算产生影响。这种做法称为显式收缩 (explicit deflation)。

第二种做法的基本思想是: 我们的搜索空间应该是已经找到的特征向量的正交补空间。假设我们已经找到了一个特征对 (λ_1, x_1) , 我们对 Krylov 子空间进行正交化, 确保新的搜索空间与 x_1 正交。

具体地，在每一步迭代中，我们可以使用 Gram-Schmidt 正交化方法，将新的向量与已经找到的特征向量正交化。这样就可以确保新的搜索空间不包含已经找到的特征向量。

当然，事实上我们只要对第一个新向量进行正交化就可以了。假如正交化后 $q_i = \sum_{i=2}^n x_i \alpha_i$ ，那么接下来的 Krylov 子空间会是

$$\text{span}\{q_i, Aq_i, \dots\} \subseteq \text{span}\{x_2, x_3, \dots, x_n\}.$$

这是因为 q_i 并不包含 x_1 的分量， Aq_i 不可能无中生有产生 x_1 的成分。这样，我们就可以避免已经找到的特征对后续计算产生影响。这种做法称为隐式收缩 (implicit deflation)。

如果我们有多个特征对，我们可以将新向量与所有已经找到的特征向量正交化。这样就可以确保新的搜索空间与所有已经找到的特征向量正交。

收缩策略能部分地缓解正交性丧失的问题，但并不能完全解决这个问题。为了更好地维护正交性，我们还需要一些其他技术。

2.2.3 Krylov-Schur 方法

当 $\|AQ_k - Q_k T_k\| = |\beta_k|$ 不够小时，这里面可能包含两个原因：一是特征值还没收敛，二是正交性丧失了。对于第二种情况， T_k 中混有已经收敛的特征值，我们需要将它们剔除出去。我们考虑

$$AQ_k = Q_k T_k + \beta_k q_{k+1} e_k^*.$$

设 $T_k = U_k \Theta_k U_k^*$ ，于是

$$A(Q_k U_k) = (Q_k U_k) \Theta_k + \beta_k q_{k+1} (e_k^* U_k).$$

残差项 $R_k = \beta_k q_{k+1} (e_k^* U_k)$ 可能是一个稠密矩阵，里面的数值可能也比较大。但是， R_k 内部可能包含一些小扰动项：比如前三列很小，而后面几列比较大，这说明前三个 Ritz 对已经收敛了。对应地， T_k 的前三个特征值也已经收敛了，我们就可以把它们剔除出去。这就是 Krylov-Schur 方法的基本思想。

具体地，若 $AQ_k = Q_{k+1} \tilde{T}_k$ ，我们对 \tilde{T}_k 作 Schur 分解：

$$\tilde{T}_k = V_k \begin{bmatrix} \times & & & & \\ & \times & & & \\ & & \times & & \\ & & & \ddots & \\ & & & & \times \\ \times & \varepsilon & \times & \cdots & \times & \times \end{bmatrix} V_k^*.$$

假如我们发现最后一行的某一个元素比较小，我们可以把它置零，利用置换矩阵把它对应的特征值换到前几列去。这样，我们就将已经收敛的特征值和未收敛的特征值分开了：

$$\tilde{T}_k = V_k \begin{bmatrix} \Lambda_1 & & \\ & \Lambda_2 & \\ 0 & \star & \beta_k \end{bmatrix} V_k^*.$$

对于未收敛的部分，我们先做列变换，把最后一行的残差项全部消成零，这样上方的对角阵 Λ_2 会被填满：

$$\tilde{T}_k = V_k \begin{bmatrix} \Lambda_1 & & \\ & \star & \\ 0 & 0 & \beta_k \end{bmatrix} V_k^*.$$

对这部分填满的 \star ，我们再做 Hessenberg 约化，把它变成三对角矩阵：

$$\tilde{T}_k = V_k \begin{bmatrix} \Lambda_1 & & \\ & T_{n_2} & \\ 0 & 0 & \beta_k \end{bmatrix} V_k^*.$$

接下来，我们只保留未收敛的部分 T_{n_2} ，并将其作为新的迭代矩阵继续迭代：

$$A(Q_k V_{n_2}) = (Q_k V_{n_2}) T_{n_2} + \beta_k q_{k+1} (e_k^* V_{n_2}).$$

通过这种方式，我们可以有效地剔除已经收敛的特征值，保持迭代过程的稳定性和效率。Krylov–Schur 方法是一种隐式重启的技术，它能够在保持 Krylov 子空间维数不变的情况下，剔除已经收敛的特征值，从而提高算法的收敛速度和数值稳定性。

我们上面都是已知特征向量，利用 Rayleigh–Ritz 投影来计算特征值。那么，假如特征向量的精度是 $O(\varepsilon)$ ，特征值的精度会是多少呢？对 Hermite 矩阵来说，有一个非常漂亮的结论：特征值的误差是特征向量误差的平方倍，即 $O(\varepsilon^2)$ 。

证明. 考虑 $\hat{x} = x + \Delta x$. 于是

$$\begin{aligned} \hat{\lambda} &= \frac{\hat{x}^* A \hat{x}}{\hat{x}^* \hat{x}} \\ &= \frac{(x + \Delta x)^* A (x + \Delta x)}{(x + \Delta x)^* (x + \Delta x)} \\ &= \frac{x^* \boxed{Ax} + \Delta x^* \boxed{Ax} + \boxed{x^* A} \Delta x + \Delta x^* A \Delta x}{(x + \Delta x)^* (x + \Delta x)} \\ &= \frac{x^* x \lambda + \Delta x^* x \lambda + \lambda x^* \Delta x + \Delta x^* A \Delta x}{(x + \Delta x)^* (x + \Delta x)} \\ &= \lambda + \frac{\Delta x^* (A - \lambda I) \Delta x}{(x + \Delta x)^* (x + \Delta x)} \\ &= \lambda + O(\|\Delta x\|^2). \end{aligned}$$

于是，若 $|\Delta x| = O(\varepsilon)$ ，则 $|\hat{\lambda} - \lambda| = O(\varepsilon^2)$. □

这个结论的意义在于，假如我们只要特征值，不要特征向量，那么我们可以提早收敛：比如特征向量的误差是 10^{-8} ，那么特征值的误差就已经是 10^{-16} 了，已经达到机器精度级别了，我们没必要再算下去了。