

关于稀疏矩阵特征值问题的 Arnoldi 算法与 Lanczos 算法的数值特性分析

吴开亮

数学科学学院 1101110030

目录

1	Arnoldi 算法及其数值特性分析	2
1.1	问题的提出	2
1.2	Arnoldi 方法与重正交化	2
1.3	数值特性分析	4
1.3.1	计算误差分析	4
1.3.2	正交性分析	6
1.3.3	收敛性分析	9
2	Lanczos 算法及其数值特性分析	16
2.1	问题的提出	16
2.2	Lanczos 方法及其完全重正交化	16
2.3	数值特性分析	18
2.3.1	计算误差分析	18
2.3.2	正交性分析	20
2.3.3	收敛性分析	23
2.3.4	最小 Ritz 值收敛性与改进方法	25
3	奇异值分解的 Lanczos 算法	29
3.1	问题的提出	29
3.2	算法的设计思路	29
3.3	双对角化 Lanczos 过程	29
3.4	计算部分奇异值的双对角化 Lanczos 算法	30
3.5	数值特性分析	31
3.5.1	奇异值收敛性分析	31
3.5.1	算法的性能分析	33
4	结语	34

1 Arnoldi 算法及其数值特性分析

1.1 问题的提出

先用 MATLAB 命令

```
>> load west0479
```

```
>> A = west0479
```

产生一个非对称的稀疏矩阵 A 。然后,对给定的自然数 k ,将重正交化的 Arnoldi 算法应用到 A 上产生 $Q_k, H_k, q_{k+1}, h_{k+1,k}$ 。

(1) 对 $k = 2, 3, \dots, 50$, 计算 $\nu_k = \|AQ_k - Q_k H_k - h_{k+1,k} q_{k+1} e_k^T\|_2$, 并且做图, 观察其变化趋势, 进而说明 Arnoldi 分解的计算误差随着 k 的变化规律。

(2) 对 $k = 2, 3, \dots, 50$, 计算 $\xi_k = \|Q_k^T Q_k - I_k\|_2$, 并且做图, 观察其变化趋势, 进而说明 Q_k 的正交性随着 k 增加的损失情况。

(3) 用 MATLAB 函数 **eig** 分别计算 A 和 H_{50} 的全部特征值, 并且将它们都标注在同一复平面上, 然后观察 H_{50} 那些特征值已经收敛到 A 的特征值。

1.2 Arnoldi 方法与重正交化

给定一个矩阵 $A \in R^{n \times n}$, 一个向量 $q_1 \in R^n$ ($\|q_1\|_2 = 1$) 和一个正整数 k , 下面的算法计算一个长度为 k 的 Arnoldi 分解

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T.$$

算法 1.1 Arnoldi 方法

```
for  $j = 1 : k$   
     $w = Aq_j$   
    for  $i = 1 : j$   
         $h_{ij} = q_i^T w; w = w - h_{ij} q_i$   
    end  
     $h_{j+1,j} = \|w\|_2$   
    if  $h_{j+1,j} = 0$     stop  
    else     $q_{j+1} = w / h_{j+1,j}$   
    end  
end
```

注意到，上述算法只在计算 w 时用到了 A 与一个向量做乘积，这使得该算法可充分利用 A 的稀疏性和其所具有的特殊结构。此外，若 Arnoldi 过程中途中断，即计算过程中出现了 $h_{j+1,j} = 0$ ，则表明 $\mathcal{K}_j(A, q_1)$ 已经是 A 的不变子空间。在很多情况下，这对我们是有益的。

需要指出的是，相关研究的数值结果表明，算法 1.1 在实际使用时， q_j 之间的正交性很快就会损失掉。解决这一问题的一种方法就是在计算过程中使用重正交化技术，即在算法中再重复执行一次 Gram-Schmidt 正交化过程。这样就得到了如下算法。

给定一个矩阵 $A \in R^{n \times n}$ ，一个向量 $q_1 \in R^n$ ($\|q_1\|_2 = 1$) 和一个正整数 k ，下面的算法计算一个长度为 k 的 Arnoldi 分解

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T.$$

并且再计算过程中使用重正交化技术。

算法 1.2 重正交化 Arnoldi 方法

```

for  $j = 1 : k$ 
     $w = Aq_j$ 
    for  $i = 1 : j$ 
         $h_{ij} = q_i^T w; w = w - h_{ij} q_i$ 
    end
    for  $i = 1 : j$ 
         $s = q_i^T w; h_{ij} = h_{ij} + s; w = w - sq_i$ 
    end
     $h_{j+1,j} = \|w\|_2$ 
    if  $h_{j+1,j} = 0$ 
        stop
    else
         $q_{j+1} = w / h_{j+1,j}$ 
    end
end
end

```

Arnoldi 方法是最经典的一类 Krylov 方法，我们希望计算的是 A 在某一指定范围内的少数几个特征值及其对应的特征向量，其中 A 是大型的稀疏矩阵。将算

法 1.2 与著名的 Rayleigh-Ritz 方法以及重启技术相结合就可以得到求稀疏矩阵部分特征值和对应特征向量的 Arnoldi 算法

给定一个 n 阶实矩阵 A ，下面的算法设计 A 的少数几个特征对。

1. 选择初始初始向量 v ，并令 $q_1 = v / \|v\|_2$ 。
2. 利用算法 1.2 产生一个长度为 k 的 Arnoldi 分解

$$A Q_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T.$$

3. 计算 H_k 的特征值，并选择其中若干个满足要求的记为

$$\mu_1, \mu_2, \dots, \mu_\ell$$

4. 计算 μ_j 所对应的特征向量 y_j ，并形成 Ritz 向量 $u_j = Q_k y_j, j = 1, 2, \dots, \ell$ 。
5. 如果不满足要求，则增加 k ，在返回到步 2，重复以上各步。

该算法只涉及矩阵乘向量 $y = Ax$ ，因此可充分利用 A 的稀疏性和 A 所具有的特殊结构。

设 $H_k y = \mu y$ ，在 Arnoldi 分解的两边右乘 y 可得

$$A Q_k y = Q_k H_k y + \beta_k q_{k+1} e_k^T y.$$

于是

$$\|A(Q_k y) - \mu(Q_k y)\|_2 = |\beta_k| |e_k^T y|.$$

由此可知，我们可以由 $|\beta_k| |e_k^T y|$ 的大小来决定 $(\mu, Q_k y)$ 是否可以作为 A 的近似特征对。

1.3 数值特性分析

1.3.1 计算误差分析

为便于分析，我们按照题意，定义 Arnoldi 分解的计算误差如下。

定义 1.1 设 Arnoldi 分解迭代的次数为 k ，定义此时得到的长度为 k 的 Arnoldi 分解

$$A Q_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T$$

计算误差为

$$\nu_k = \|A Q_k - Q_k H_k - h_{k+1,k} q_{k+1} e_k^T\|_2.$$

定义 1.1 定义的计算误差是用来衡量分解过程中产生的误差。

算例 1.1 先用 MATLAB 命令

```
>> load west0479
```

```
>> A = west0479
```

产生一个非对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将重正交化的 Arnoldi 算法应用到 A 上产生 $Q_k, H_k, q_{k+1}, h_{k+1,k}$ 。

然后我们对 $k = 2, 3, \dots, 50$ ，计算 $\nu_k = \|A Q_k - Q_k H_k - h_{k+1,k} q_{k+1} e_k^T\|_2$ ，并且做图，观察其变化趋势，进而说明 Arnoldi 分解的计算误差随着 k 的变化规律。

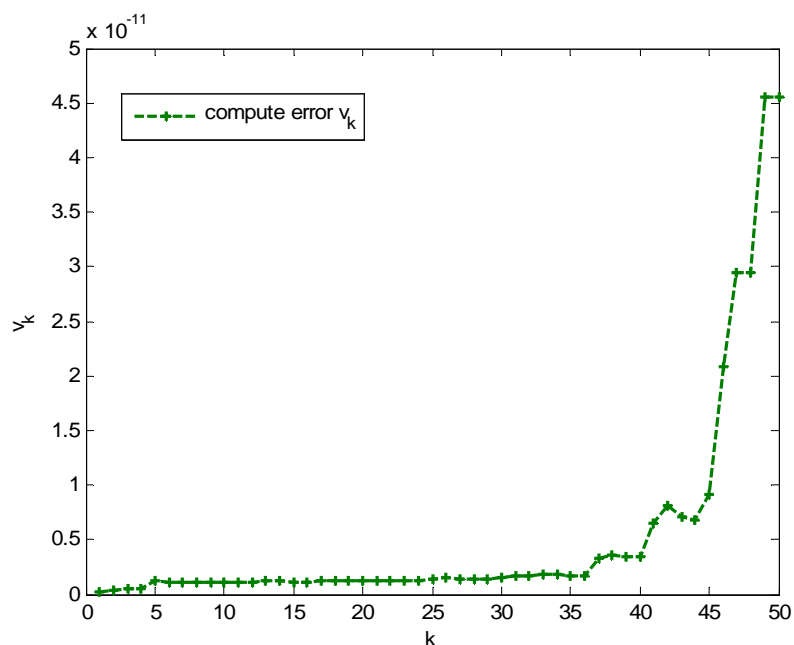


图 1.1 Arnoldi 分解的计算误差随着 k 的变化规律

图 1.1 给出了 Arnoldi 分解的计算误差随着 k 的变化规律。从图中可以看出，计算误差 ν_k 随着 k 的增加呈增长趋势：在 $k = 2 \sim 40$ 之间， ν_k 增长很缓慢；在 $k = 40 \sim 50$ 之间， ν_k 增长很迅速，几乎与指数增长相当。

为进一步分析 Arnoldi 分解的计算误差随着 k 的变化趋势，我们计算了 $k \geq 50$ 时的计算误差 ν_k ，绘制其随着 k 的变化曲线如图 1.2 所示。

从图 1.2 我们可以看出，当 $k \geq 55$ 时的计算误差 ν_k 的增长趋于平缓，直到 $k = 425$ 附近出现一个大的奇点，之后又趋于平缓。

此外，我们得到本问题的 Arnoldi 分解的计算误差始终控制在 $[0, 2.5 \times 10^{-10}]$ 范围内，这说明我们的计算精度还是比较高的，能用于实际问题的精确计算。

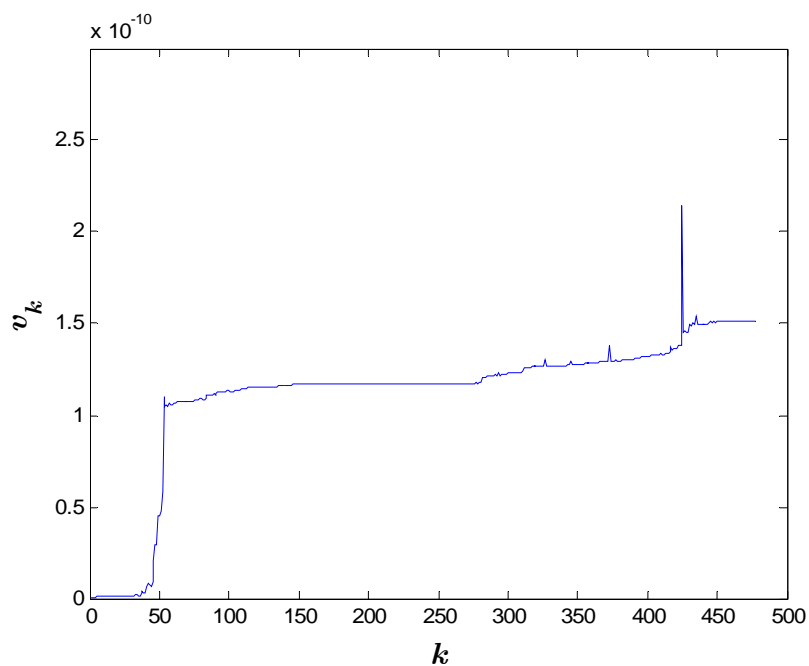


图 1.2 Arnoldi 分解的计算误差随着 k 的变化曲线

1.3.2 正交性分析

前面我们提到，对于未进行重正交化的 Arnoldi 方法， q_j 之间的正交性很快就会损失掉，而重正交化技术就是针对这一问题提出的。为衡量正交性（即 q_j 之间的正交程度），我们定义了 Arnoldi 分解的正交性误差如下。

定义 1.2 设 Arnoldi 分解迭代的次数为 k ，定义此时得到的长度为 k 的 Arnoldi 分解

$$A Q_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T$$

计算误差为

$$\xi_k = \|Q_k^T Q_k - I_k\|_2.$$

算例 1.2

先用 MATLAB 命令

```
>> load west0479
```

```
>> A = west0479
```

产生一个非对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将重正交化的 Arnoldi 算法应用到 A 上产生 $Q_k, H_k, q_{k+1}, h_{k+1,k}$ 。

我们对 $k = 2, 3, \dots, 50$ ，计算正交性误差 $\xi_k = \|Q_k^T Q_k - I_k\|_2$ ，并且做图，观察其变化趋势，进而说明 Q_k 的正交性随着 k 增加的损失情况。

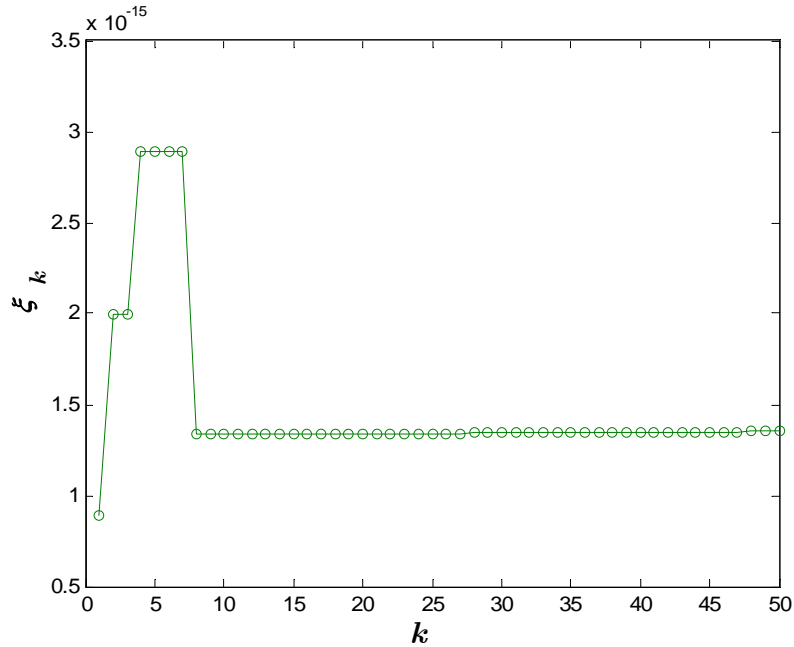


图 1.3 重正交化的 Arnoldi 算法产生 Q_k 的正交性随着 k 增加的损失情况

图 1.3 给出了重正交化的 Arnoldi 分解的正交性误差 ξ_k 随着 k 的变化规律。从图中可以看出，正交性误差 ξ_k 随着 k 的增加呈稳定趋势：在 $k = 2 \sim 5$ 之间， ξ_k 略有增长；在 $k = 5 \sim 8$ 之间，增长趋于平缓；在 $k = 8 \sim 10$ 之间，略有下降；此后，正交性误差 ξ_k 一直处于几乎不变的状态。

此外，我们得到本问题的 Arnoldi 分解的计算误差始终控制在 $[0, 3 \times 10^{-15}]$ 范围内，这说明我们的产生的 Q_k 的正交性是非常高的，能用于实际问题的精确计算。

下面我们用重正交化与未重正交化的 Arnoldi 算法的结果进行正交误差的对比分析，来讨论和论证重正交化的功效。

图 1.4 给出了未重正交化的 Arnoldi 分解的正交性误差 ξ_k 随着 k 的变化规律。从图中可以看出，正交性误差 ξ_k 随着 k 的增加一直呈增长趋势。

图 1.5 给出了重正化与未重正化的 Arnoldi 算法的正交误差的对比曲线，通过观察，重正化的作用是显著的。通过本数值实验，我们对重正化的作用有了一个更加清晰的了解和理解。

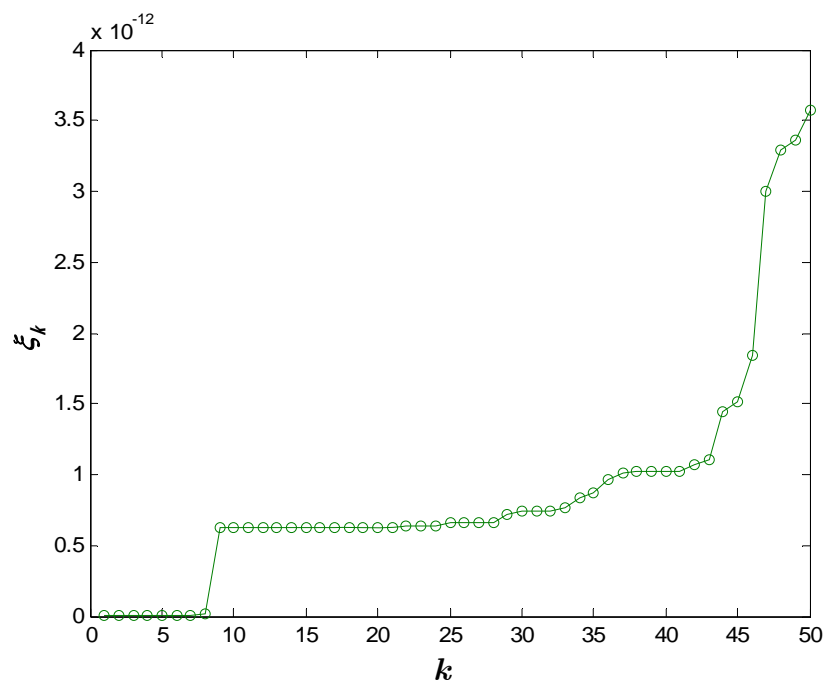


图 1.4 未重正化的 Arnoldi 算法产生 Q_k 的正交性随着 k 增加的损失情况

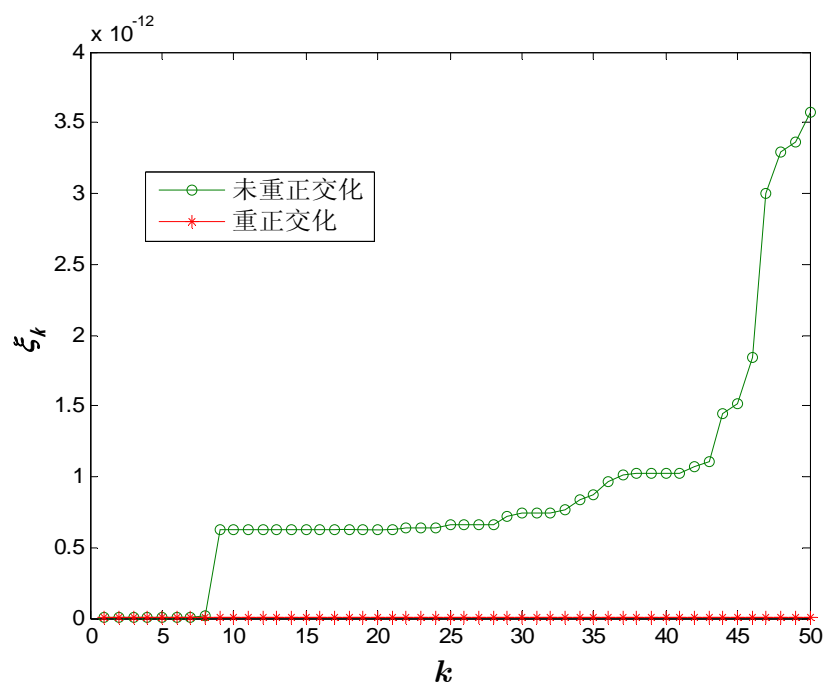


图 1.5 重正化与未重正化的 Arnoldi 算法的正交误差的对比分析

1.3.3 收敛性分析

在这一小节，我们用数值算例讨论 Arnoldi 算法的收敛性。

算例 1.3

先用 MATLAB 命令

```
>> load west0479
```

```
>> A = west0479
```

产生一个非对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将重正交化的 Arnoldi 算法应用到 A 上产生 $Q_k, H_k, q_{k+1}, h_{k+1,k}$ 。

我们用 MATLAB 函数 **eig** 分别计算 A 和 H_{50} 的全部特征值，并且将它们都标注在同一复平面上，然后观察 H_{50} 那些特征值已经收敛到 A 的特征值。

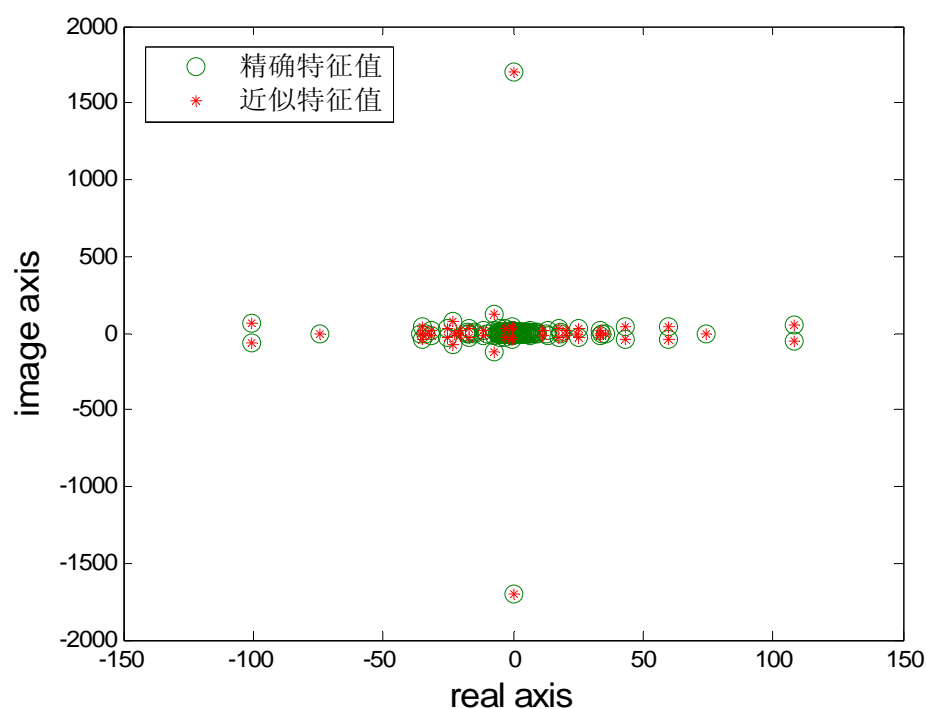


图 1.6 复平面上矩阵 A 和 H_{50} 的全部特征值

图 1.6 显示了矩阵 A 和 H_{50} 的全部特征值在复平面上的位置。请注意观察， H_{50} 的很多特征值已经收敛到了 A 的特征值。具体地说，离原点较远，即模较大的特征值均已经收敛到了 A 的特征值；而离原点较近，即模较小的特征值的收敛还不明显。

此外，从图 1.6 中我们还可以观察到特征值关于 x 轴对称，这是因为矩阵的实特征值均分布于 x 轴上，而复特征值互为共轭地成对出现，故关于 x 轴上下对称。

由于原点附近 A 的特征值较多，不易于观察。为进一步分析 H_{50} 在原点附近地收敛情况，我们将图 1.6 在原点附近进行局部放大，得到图 1.7。

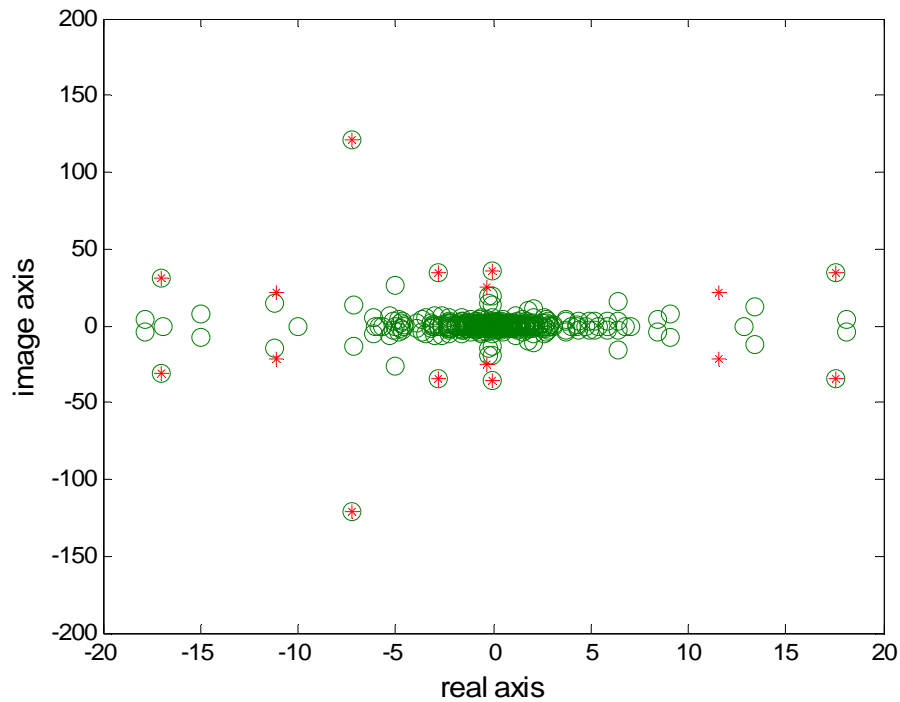


图 1.7 复平面上矩阵 A 和 H_{50} 在原点附近的特征值

通过图 1.7，进一步证实了我们上面的分析：离原点较远，即模较大的特征值均已经收敛到了 A 的特征值；而离原点较近，即模较小的特征值的收敛很慢。

算例 1.4

为进一步对重正交化的 Arnoldi 算法的收敛性进行分析，我们考虑本算例。
令

$$A = Q' \text{diag}(-24, -23, \dots, 24, 25) Q \in R^{50 \times 50}$$

其中 Q 是随机产生一个正交矩阵（用 matlab 命令 `orth(randn(n,n))`）。对于 k 从 1 到 30，我们应用重正交的 Arnoldi 算法于给定的矩阵 A 上。

图 1.8 显示的是每一步迭代得到的特征值与精确特征值之间的对比，其中横坐标表示迭代步数 k ，第 k 列的点表示 H_k 的所有特征值，最后一列的点表示 A 的精确特征值。

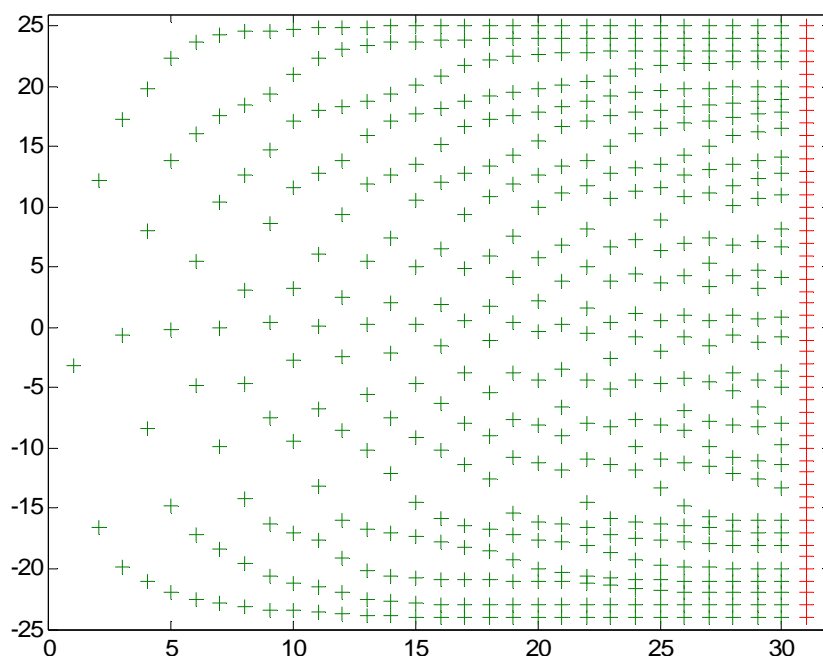


图 1.8 Arnoldi 算法的收敛特性分析

图 1.8 进一步的展示了重正交化的 Arnoldi 算法的收敛特性： $k = 10$ 时， H_{10} 就有两个特征值收敛到了 A 的两个最大特征值；而到了 $k = 21$ 时， H_{21} 就有 3 个最小特征值和 2 个最大特征值分别收敛到了 A 的 3 个最小特征值和 2 个最大特征值；而位于区间内部的有些特征值，直到 $k = 30$ 还没有 Ritz 值收敛到它们。

虽然算例 1.3 和 1.4 都仅仅是一些特殊矩阵，但是其所展示处的 Arnoldi 算法的收敛特性对一般的情形也是成立的，即外围特征值先收敛，内部特征值收敛特别缓慢。

1.3.4 Arnoldi 双纽线与几何收敛分析

以上收敛过程可以用在复平面内画出双纽线来作图示。双纽线是一曲线或曲线的集合

$$\{z \in \mathbb{C} : |p(z)| = C\}$$

其中 $p(z)$ 是一个多项式， C 为实常数。如果 $p(z)$ 取为 H_k 的特征多项式，则 C 取为

$$C = \left\| \sum_{j=1}^k q_j \right\|_2。$$

那么由 $\{z \in \mathbb{C} : |p(z)| = C\}$ 定义的曲线称为 **Arnoldi 双纽线**。当迭代数 k 增加时，这些双纽线的分支典型地围绕 A 的特征值出现，然后很快收缩到一个点，即特征值本身。

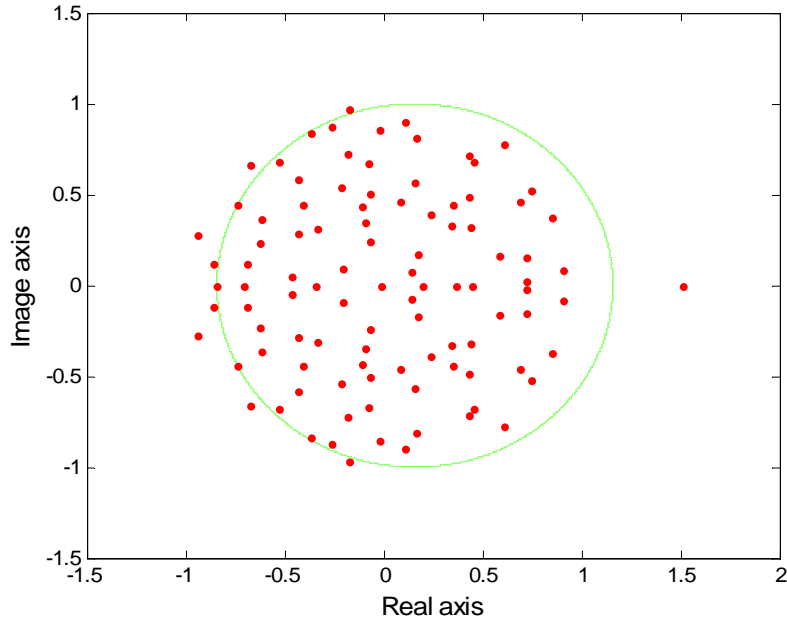


图 1.9 100×100 矩阵 A 的特征值，矩阵 A 的元素除 $(1, 1)$ 位置外均为随机的。

圆是 \mathbb{C} 中的单位圆，除分离特征值 $\lambda = 1.5099$ 外，所有的特征值在单位圆盘内附近地均匀分布

为了说明这些思想，设 A 为维数 $n = 100$ 的一个方阵，其元素来自于均值为 0，标准差为 $n^{-1/2} = 0.1$ 的实正态分布的独立随机数。由于 A 是实的，所以其特征值是由实数和复共轭对组成。标准差的选择是使得特征值在单位圆盘 $|z| \leq 1$ 内近似地均匀分布。但现在把对角元素 a_{11} 变为 1.5 来建立一个分离特征值，见图 1.9 所示。

把重正交化的 **Arnoldi** 算法应用到这一矩阵，初始向量取为随机向量。在 k 步，矩阵 H_k 已经形成，其特征多项式反映了迭代关于 A 的谱的当前信息。图

1.10(a)(b)(c)(d)分别画出了在 $k = 7, 8, 9, 10$ 的 Arnoldi 双纽线。在 $k = 1$ ，双纽线是一个精确的圆。但稍微受到分离特征值的影响。在 $k = 8$ 之前，圆开始在 λ 方向凸出。在 $k = 9$ 时，凸出部分断开并且出现了双纽线的一个新的分支。然后，这个分支继续下去，在其后的每一次迭代中收缩。即 Arnoldi 双纽线首先“吞下”分离特征值，在其后的迭代中，它以几何速率收缩到一个点。

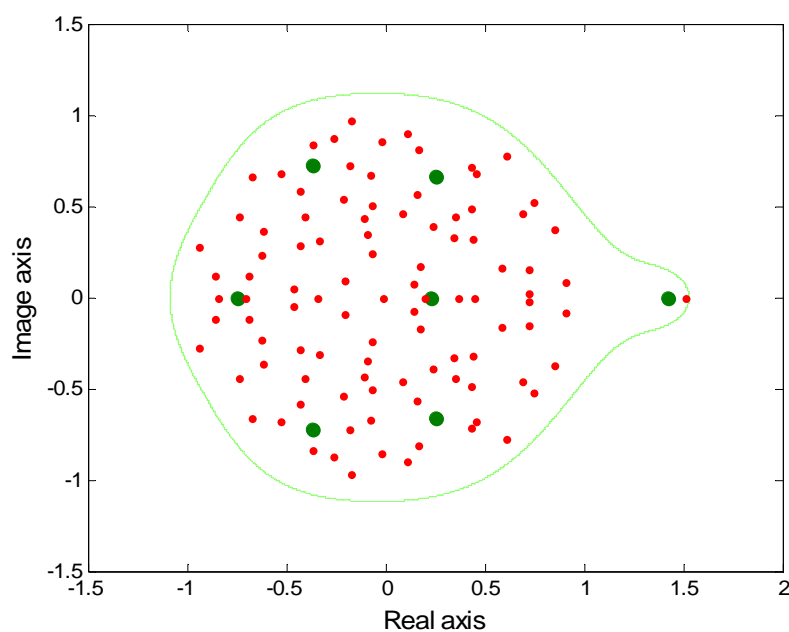


图 1.10(a) 对于同样的矩阵 A ，在 $k = 7$ 的 Arnoldi 双纽线

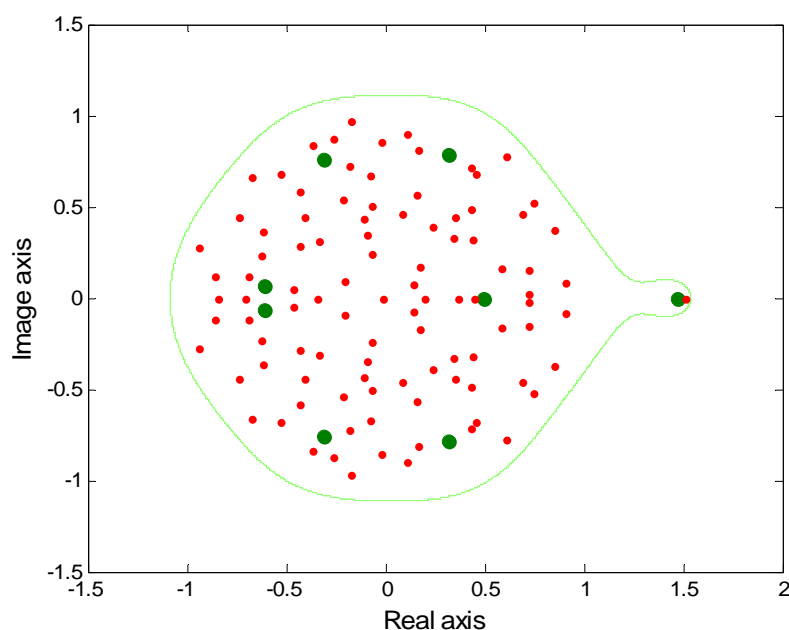


图 1.10(b) 对于同样的矩阵 A ，在 $k = 8$ 的 Arnoldi 双纽线

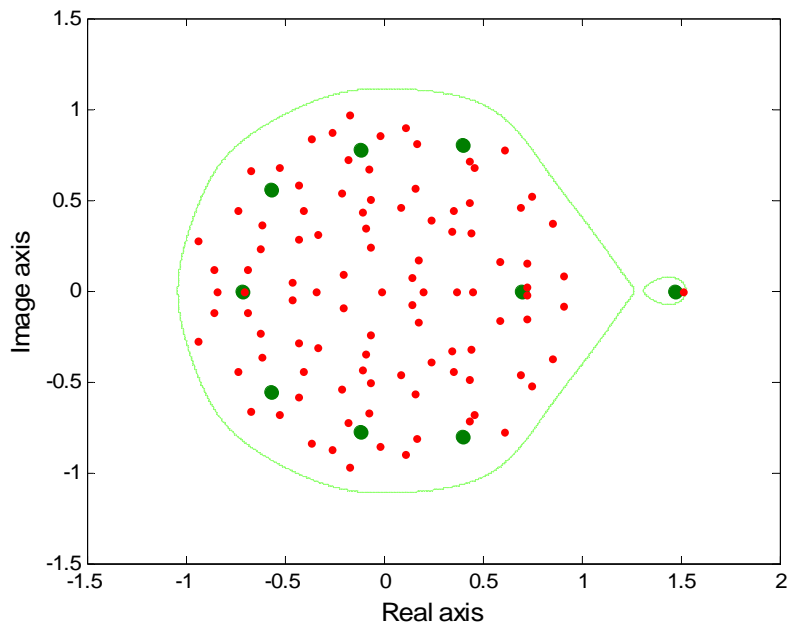


图 1.10(c) 对于同样的矩阵 A ，在 $k = 9$ 的 Arnoldi 双纽线

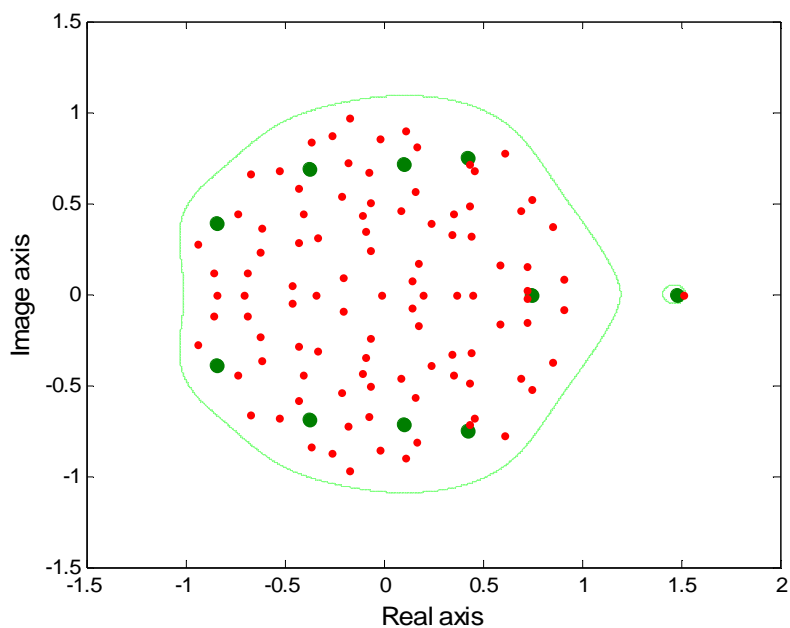


图 1.10(b) 对于同样的矩阵 A ，在 $k = 8$ 的 Arnoldi 双纽线

图 1.10 对于同样的矩阵 A ，在 $k = 7, 8, 9, 10$ 的 Arnoldi 双纽线，小的

点表示的是矩阵 A 的特征值；大的点是 H_k 的特征值。

在某些情况下，对于矩阵 A 的特征值，一些 Arnoldi 特征值估计的收敛是几何的。这些问题在当前是不能理解的，然后，用上面的数值算例，我们可以考察其收敛性，并给出部分说明。

从图 1.10 的一个图上，不能确定收敛速率超过精度的 1 或 2 位数字。对于如前同样的例子，用画出 $|\lambda^{(k)} - \lambda|$ 作为 k 的函数曲线的图 1.11 来填补这个差距，其中， $\lambda^{(k)}$ 是在 k 步上最靠近离群特征值 λ 的 Arnoldi 特征值估计。注意到第一件事情是 1000 来代替 100，那么这个图形也没有太多的不同。

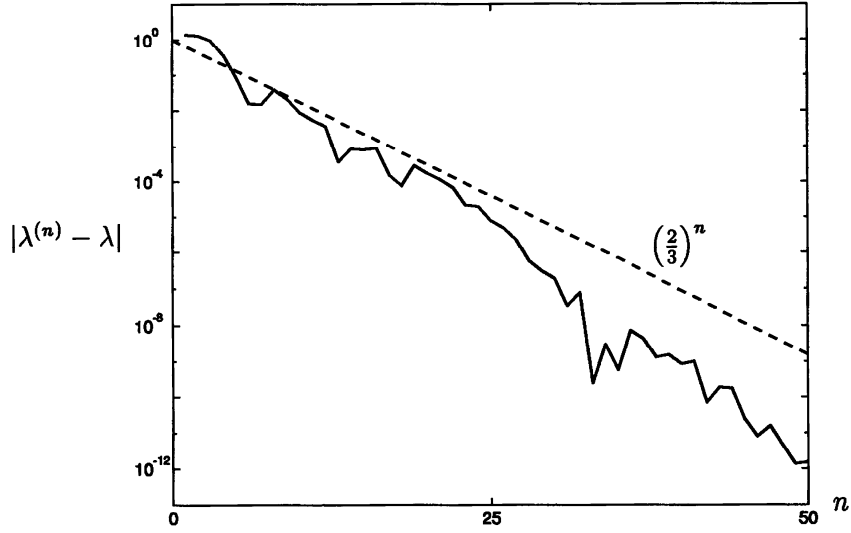


图 1.11 最右 Arnoldi 特征值估计的收敛性

可以更多的定量考察，不论怎样，对于最初几十次迭代，至少在图 1.10 中的收敛近似于速率

$$|\lambda^{(n)} - \lambda| \approx \left(\frac{2}{3}\right)^n.$$

这个性质的解释如下。在 A 的每一个特征值上， $p(z)$ 必须取一个出略的极小值。例如考虑候选多项式 $p(z) = z^{n-1}(z - \bar{\lambda})$ ，其中 $\bar{\lambda}$ 是接近 λ 的某个数。在单位圆盘内的 A 的每一个特征值上， $p(z)$ 是一阶的或者更小，在 $z = \lambda$ ，它有数量

$$p(\lambda) \approx \left(\frac{3}{2}\right)^n |\lambda^{(k)} - \lambda|$$

（如果 λ 恰好等于 $\frac{3}{2}$ ，则这是一个等式）。当 n 很大时， $\left(\frac{3}{2}\right)^n$ 非常大。对于这个数也是 1 阶的，为平衡它， $|\lambda - \bar{\lambda}|$ 必须取得足够小，即为 $\left(\frac{2}{3}\right)^n$ 。

另一个特点在图 1.11 中很明显。在最初几十次迭代之后，收敛开始加速。

2 Lanczos 算法及其数值特性分析

2.1 问题的提出

先用 MATLAB 命令

```
>> A = delsq (numgrid( 'H' ,40));
```

```
>> perm =symamd(A); A = A(perm,perm);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将完全重正交化的 Lanczos 算法应用到 A 上产生 $Q_k, T_k, q_{k+1}, \beta_k$ 。

(1) 对 $k = 2, 3, \dots, 50$ ，计算 $\nu_k = \|AQ_k - Q_k T_k - \beta_k q_{k+1} e_k^T\|_2$ ，并且做图，观察其变化趋势，进而说明 Lanczos 分解的计算误差随着 k 的变化规律。

(2) 对 $k = 2, 3, \dots, 50$ ，计算 $\xi_k = \|Q_k^T Q_k - I_k\|_2$ ，并且做图，观察其变化趋势，进而说明 Q_k 的正交性随着 k 增加的损失情况。

(3) 对 $k = 2, 3, \dots, 50$ ，计算 $\eta_k = \|\mu_k - \lambda_n\|_2$ （这里 μ_k 和 λ_n 分别表示 T_k 和 A 的最小特征值），并且做图，观察其变化趋势，进而说明最小 Ritz 值收敛情况。

(4) 将完全重正交化 Lanczos 算法应用到 A^{-1} 上产生 \tilde{T}_k 。然后，对 $k = 2, 3, \dots, 50$ ，计算 $\eta_k = \|\mu_{1,k}^{-1} - \lambda_n\|_2$ （这里 $\mu_{1,k}$ 和 λ_n 分别表示 \tilde{T}_k 的最大特征值和 A 的最小特征值），并且做图，观察其变化趋势，再与 (3) 的结果比较来说明它们的优劣。

2.2 Lanczos 方法及其完全重正交化

当矩阵 $A \in R^{n \times n}$ 是对称矩阵时，在 Arnoldi 分解中，其关于 Q_k 的 Rayleigh 商 $H_k = Q_k^T A Q_k$ 就是一个对称三对角矩阵。这样对应的 Arnoldi 分解就变成

$$A Q_k = Q_k T_k + \beta_k q_{k+1} e_k^T. \quad (2.1)$$

其中

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_k \end{bmatrix}.$$

此时，我们称分解（2.1）为一个长度为 k 的 **Lanczos** 分解。

比较（2.1）式两边的各列可得

$$\begin{aligned} Aq_1 &= \alpha_1 q_1 + \beta_1 q_2 \\ Aq_j &= \beta_{j-1} q_{j-1} + \alpha_j q_j + \beta_j q_{j+1}, j = 2, 3, \dots, k \end{aligned}$$

于是有

$$\begin{aligned} \alpha_1 &= q_1^T A q_1 \\ \beta_1 &= \|Aq_1 - \alpha_1 q_1\|_2, q_2 = (Aq_1 - \alpha_1 q_1) / \beta_1, \\ \alpha_j &= q_j^T A q_j, r_j = Aq_j - \alpha_j q_j - \beta_{j-1} q_{j-1}, \\ \beta_j &= \|r_j\|_2, q_{j+1} = r_j / \beta_j, j = 2, 3, \dots, k. \end{aligned}$$

这样我们就得到了如下的 **Lanczos** 算法。

给定一个对称矩阵 $A \in R^{n \times n}$ ，一个向量 $q_1 \in R^n$ ($\|q_1\|_2 = 1$) 和一个正整数 k ，

下面的算法计算一个长度为 k 的 **Lanczos** 分解

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

算法 2.1 Lanczos 方法

```

 $\beta_0 = 0; q_0 = 0$ 
for  $j = 1 : k$ 
     $w = Aq_j; \alpha_j = q_j^T w$ 
     $w = w - \alpha_j q_j - \beta_{j-1} q_{j-1}$ 
     $\beta_j = \|w\|_2$ 
    if  $\beta_j = 0$ 
        stop
    else
         $q_{j+1} = w / \beta_j$ 
    end
end

```

当然在实际应用时，该算法所产生的 q_j 之间的正交性很快就会损失掉。解决这一问题的一种方法仍然是在计算过程中使用重正交技术，即在算法中再重复执行一次 **Gram-Schmidt** 正交化过程。其实重正交化就是在算法 2.1 中的 $w = w - \alpha_j q_j - \beta_{j-1} q_{j-1}$ 之后加上一句

$$w = w - \sum_{i=1}^{j-1} (q_i^T w) q_i.$$

当然，在实际计算时，我们也可以用修正的 Gram-Schmidt 正交化方法来实现上式计算。

有时为了使算法所产生的 q_j 之间具有更好的正交性，也可以连续使用两次重正交化，即所谓的**完全重正交化方法**。

Lanczos 方法是目前计算大型稀疏实对称矩阵少数几个特征值和对应的特征向量的最常用方法之一，本质上就是 Arnoldi 算法应用到实对称矩阵上导出的。

给定一个 n 阶实对称矩阵 A ，下面的算法设计 A 的少数几个特征对。

1. 选择初始初始向量 v ，并令 $q_1 = v / \|v\|_2$ 。
2. 利用完全重正交的 Lanczos 算法产生一个长度为 k 的 Lanczos 分解

$$A Q_k = Q_k T_k + \beta_k q_{k+1} e_k^T.$$

3. 计算 T_k 的特征值，并选择其中若干个满足要求的记为 $\mu_1, \mu_2, \dots, \mu_\ell$
4. 计算 μ_j 所对应的特征向量 y_j ，并形成 Ritz 向量 $u_j = Q_k y_j, j = 1, 2, \dots, \ell$ 。
5. 如果不满足要求，则增加 k ，在返回到步 2，重复以上各步。

若 Rayleigh 商 T_k 的一个特征对 (μ, y) 使得 $|\beta_k| |e_k^T y|$ 很小，则 μ 就是 A 的某个特征值的很好近似。因此，通常在完全重正交化的 Lanczos 方法中我们就是以 $|\beta_k| |e_k^T y|$ 是否已经足够小来判断 Ritz 对 $(\mu, Q_k y)$ 是否可以作为 A 的近似特征对。

2.3 数值特性分析

2.3.1 计算误差分析

为便于分析，我们按照题意，定义 Lanczos 分解的计算误差如下。

定义 2.1 设 Lanczos 分解迭代的次数为 k ，定义此时得到的长度为 k 的 Lanczos 分解

$$A Q_k = Q_k T_k + \beta_k q_{k+1} e_k^T$$

计算误差为

$$\nu_k = \|A Q_k - Q_k T_k - \beta_k q_{k+1} e_k^T\|_2.$$

定义 2.1 定义的计算误差是用来衡量分解过程中产生的误差。

算例 2.1 先用 MATLAB 命令

```
>> A = delsq (numgrid( 'H' ,40));
```

```
>> perm =symamd(A); A = A(perm,perm);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将完全重正交的 Lanczos 算法应用到 A 上产生 $Q_k, T_k, q_{k+1}, \beta_k$ 。

然后我们对 $k = 2, 3, \dots, 50$ ，计算 $\nu_k = \|A Q_k - Q_k T_k - \beta_k q_{k+1} e_k^T\|_2$ ，并且做图，观察其变化趋势，进而说明 Lanczos 分解的计算误差随着 k 的变化规律。

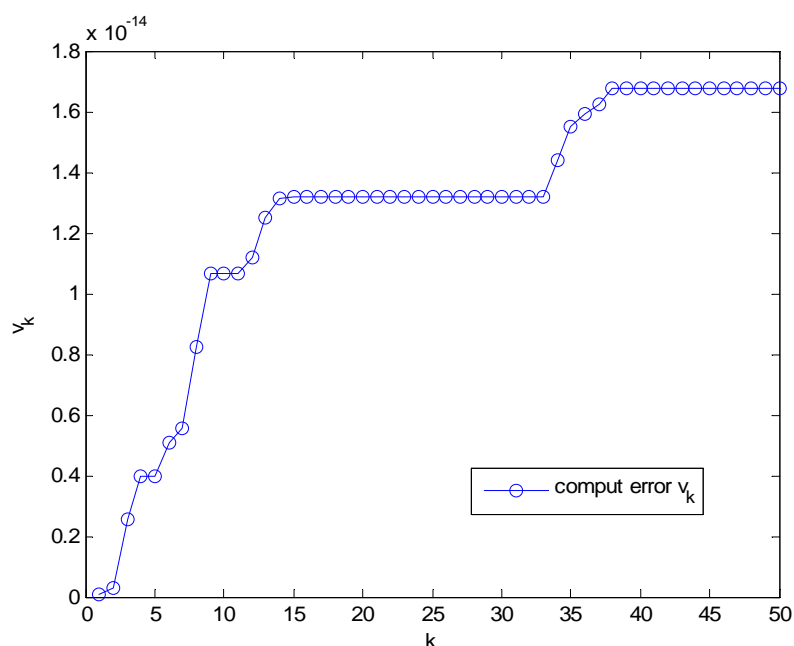


图 2.1 Lanczos 分解的计算误差随着 k 的变化规律

图 2.1 给出了 Lanczos 分解的计算误差随着 k 的变化规律。从图中可以看出，计算误差 ν_k 随着 k 的增加呈增长趋势：在 $k = 2 \sim 15$ 之间， ν_k 增长较迅速；在 $k = 15 \sim 50$ 之间， ν_k 增长很缓慢。

为进一步分析 Lanczos 分解的计算误差随着 k 的变化趋势，我们计算了 $k \geq 50$ 时的计算误差 ν_k ，绘制其随着 k 的变化曲线如图 2.2 所示。

从图 2.2 我们可以看出，当 $k \geq 55$ 时的计算误差 ν_k 的增长整体趋势平缓，呈现阶梯型增长。

此外，我们得到本问题的 Lanczos 分解的计算误差始终控制在 $[0, 1 \times 10^{-13}]$ 范围内，这说明我们的计算精度还是很高的，能用于实际问题的精确计算。

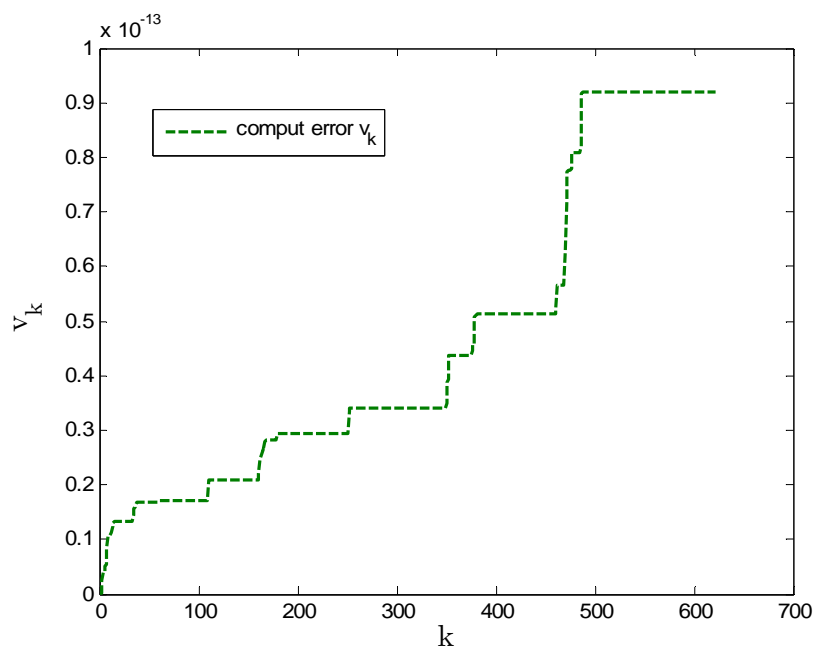


图 2.2 Lanczos 分解的计算误差随着 k 的变化曲线

2.3.2 正交性分析

前面我们提到，对于未进行重正交化的 Lanczos 方法， q_j 之间的正交性很快就会损失掉，而完全重正交化技术就是针对这一问题提出的。为衡量正交性（即 q_j 之间的正交程度），我们定义了 Lanczos 分解的正交性误差如下。

定义 1.2 设 Lanczos 分解迭代的次数为 k ，定义此时得到的长度为 k 的 Lanczos 分解

$$A Q_k = Q_k T_k + \beta_k q_{k+1} e_k^T$$

计算误差为

$$\xi_k = \|Q_k^T Q_k - I_k\|_2.$$

算例 2.2

先用 MATLAB 命令

```
>> A = delsq(numgrid('H',40));
```

```
>> perm = symamd(A); A = A(perm,perm);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将完全重正交化的 Lanczos 算法应用到 A 上产生 $Q_k, T_k, q_{k+1}, \beta_k$ 。

我们对 $k = 2, 3, \dots, 50$ ，计算正交性误差 $\xi_k = \|Q_k^T Q_k - I_k\|_2$ ，并且做图，观察其变化趋势，进而说明 Q_k 的正交性随着 k 增加的损失情况。

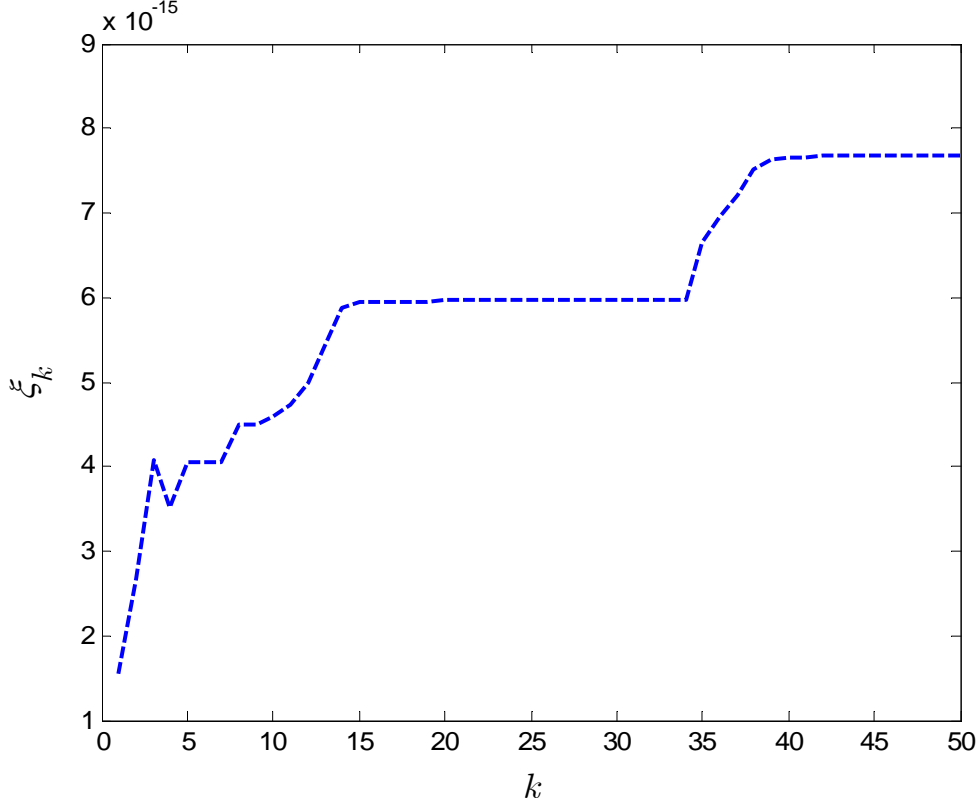


图 2.3 完全重正交化的 Lanczos 算法产生 Q_k 的正交性随着 k 增加的损失情况

图 2.3 给出了完全重正交化的 Lanczos 分解的正交性误差 ξ_k 随着 k 的变化规律。从图中可以看出，正交性误差 ξ_k 随着 k 的增加呈稳定增长趋势。此外，我们得到本问题的完全重正交化的 Lanczos 分解的计算误差始终控制在 $[0, 9 \times 10^{-15}]$ 范围内，这说明我们的产生的 Q_k 的正交性是非常高的，能用于实际问题的精确计算。

下面我们完全重正交化与未重正交化的 Lanczos 算法的结果进行正交误差的对比分析，来讨论和论证完全重正交化的功效。

图 2.4 给出了未重正交化的 Lanczos 分解的正交性误差 ξ_k 随着 k 的变化规律。从图中可以看出，正交性误差 ξ_k 随着 k 的增加一直呈几乎指数增长趋势。通过与图 2.3 对比可以看出，完全重正交化的功效是非常显著的。

图 2.5 给出了重正化与完全重正化的 Lanczos 算法的正交误差的对比曲线，通过观察，完全重正化的作用的是比重正化的作用要显著些。通过本数值实验，我们对重正化与完全重正化的作用有了一个更加清晰的了解和理解。

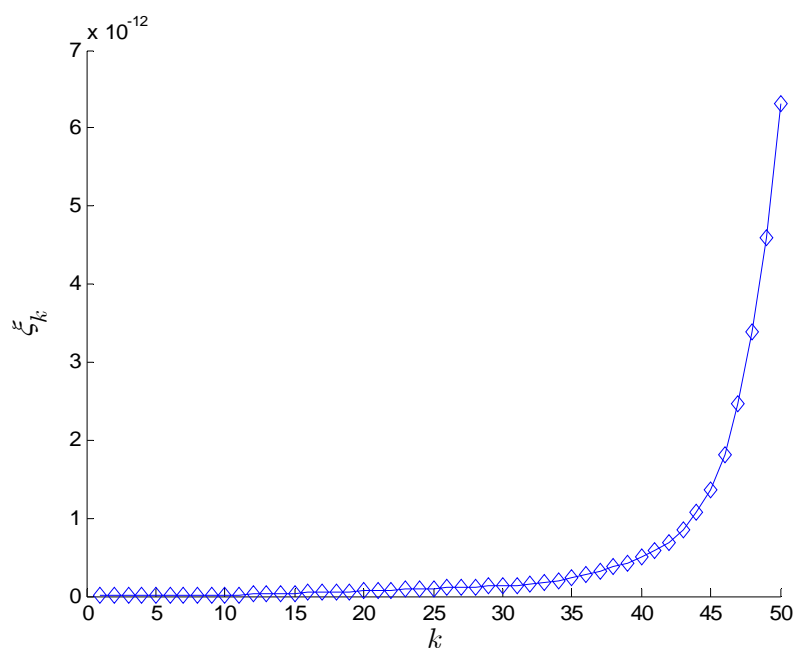


图 2.4 未重正化的 Lanczos 算法产生 Q_k 的正交性随着 k 增加的损失情况

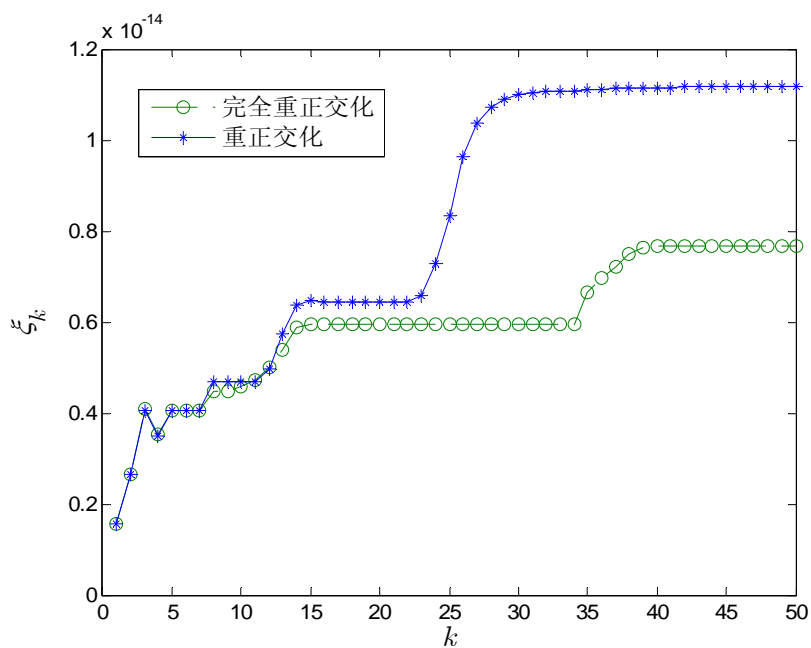


图 2.5 重正化与完全重正化的 Lanczos 算法的正交误差的对比分析

2.3.3 收敛性分析

在这一小节，我们用数值算例讨论 Lanczos 算法的收敛性。

算例 2.3

先用 MATLAB 命令

```
>> A = delsq (numgrid( 'H' ,40));  
>> perm =symamd(A); A = A(perm,perm);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将完全重正交化的 Lanczos 算法应用到 A 上产生 $Q_k, T_k, q_{k+1}, \beta_k$ 。

图 2.6 显示的是每一步迭代得到的特征值与精确特征值之间的对比，其中横坐标表示迭代步数 k ，第 k 列的点表示 T_k 的所有特征值，最后一列的点表示 A 的精确特征值。

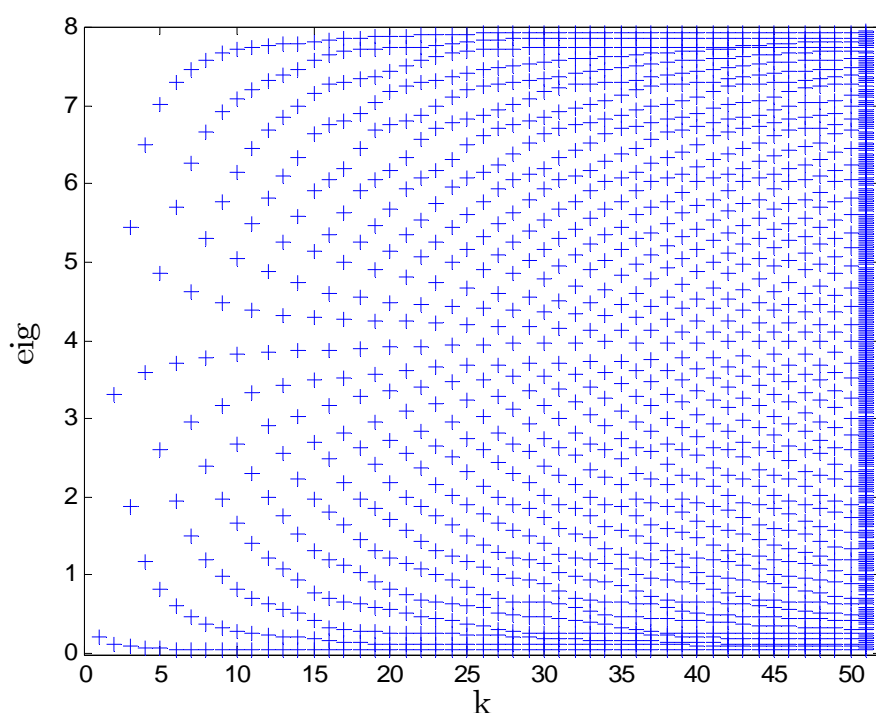


图 2.6 完全重正交化的 Lanczos 算法的收敛特性分析

图 2.6 展示了完全重正交化的 Lanczos 算法的收敛特性： $k = 20$ 时， T_{20} 就有四五个特征值收敛到了 A 的较大的特征值；而到了 $k = 30$ 时， T_{30} 就有 4 个最小

特征值和 4 个最大特征值分别收敛到了 A 的 4 个最小特征值和 4 个最大特征值；而位于区间内部的有些特征值，直到 $k = 40$ 还没有 Ritz 值收敛到它们。

算例 2.4

为进一步对完全重正交化的 Lanczos 算法的收敛性进行分析，我们考虑一个与教材中类似的算例。

令 $A \in R^{500 \times 500}$ 时一个随机产生的对角矩阵，其对角元服从正态分布。对于 k 从 1 到 50，我们应用完全重正交的 Lanczos 算法于给定的矩阵 A 上，计算出 T_k 的所有特征值，并标注在图 2.7 当中。这里横坐标表示迭代步数 k ，第 k 列的 “+” 表示 T_k 的所有特征值，最后一列的加号表示 A 的精确特征值。

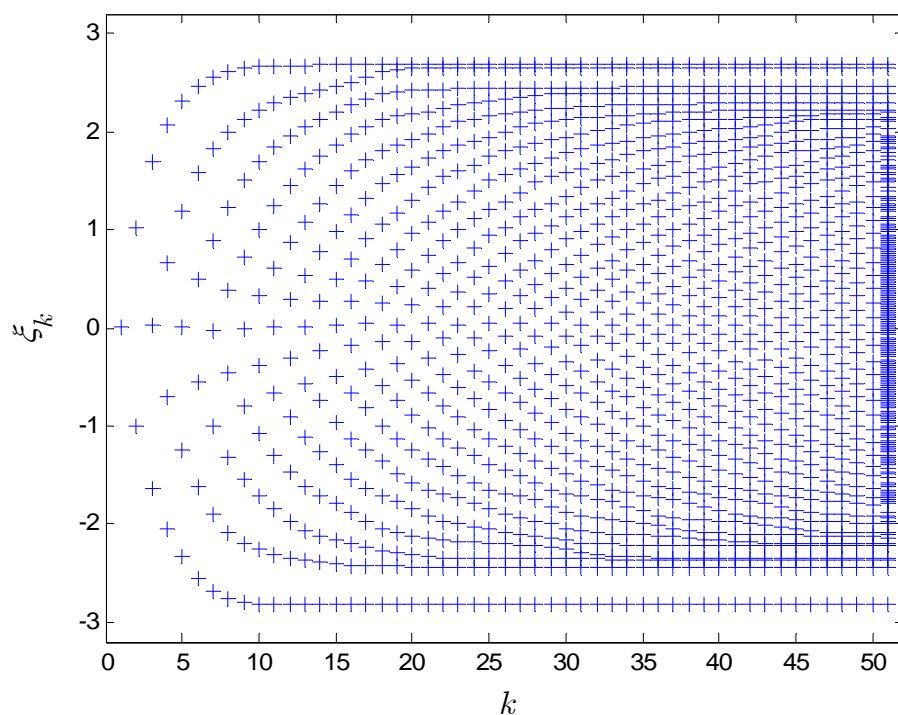


图 2.7 完全重正交化的 Lanczos 算法的收敛特性分析

从这个数值算例的计算结果可以看出：

- 两端的特征值（即最大特征值和最小特征值）收敛最快，而内部的特征值收敛较慢；

- T_k 的第 i 个最大（或最小）特征值单调增加（或减少）地收敛到 A 的第 i 个最大（或最小）特征值。

虽然算例 2.3 和 2.4 都仅仅是一些特殊矩阵，但是其所展示处的 Lanczos 算法的收敛特性对一般的情形也是成立的，即外围特征值先收敛，内部特征值收敛特别缓慢，这一点与 Arnoldi 方法的结论是类似的。

2.3.4 最小 Ritz 值收敛性与改进方法

为描述最小 Ritz 值的收敛性，我们定义最小 Ritz 值误差为

定义 2.3 定义第 k 步的最小 Ritz 值误差为

$$\eta_k = \|\mu_k - \lambda_n\|_2$$

这里 μ_k 和 λ_n 分别表示 T_k 和 A 的最小特征值。

算例 2.5

先用 MATLAB 命令

```
>> A = delsq(numgrid('H',40));
>> perm = symamd(A); A = A(perm,perm);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将完全重正交化的 Lanczos 算法应用到 A 上产生 $Q_k, T_k, q_{k+1}, \beta_k$ 。

对 $k = 2, 3, \dots, 50$ ，计算 $\eta_k = \|\mu_k - \lambda_n\|_2$ ，并且做图，观察其变化趋势，进而说明最小 Ritz 值收敛情况。

绘制最小 Ritz 值收敛情况如图 2.8 所示。从图中，我们可以看出，最小 Ritz 值收敛得很快，几乎以指数形式收敛。

为描述得更加清晰些，我们在对数坐标系下绘图进行观察。如图 2.9 所示的是以 10 为底的对数坐标系的收敛曲线。

从图 2.9 中我们可以看出，对数坐标系下的误差曲线几乎为直线，所以我们断定最小 Ritz 值的误差几乎以指数形式收敛的。经过回归分析，可以得到对数坐标系下的误差对应的线性回归曲线为 $y = -0.1678x - 0.6271$ 。

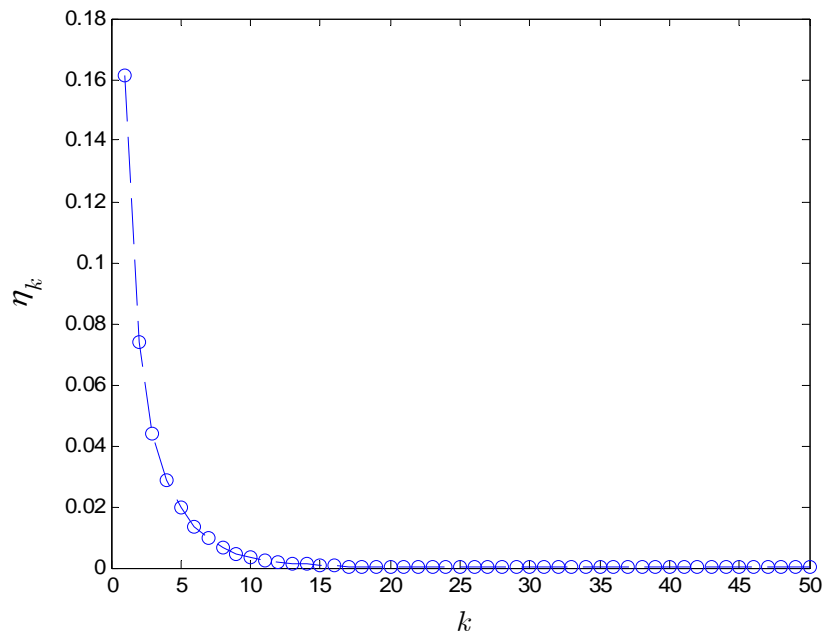


图 2.8 最小 Ritz 值的收敛性分析

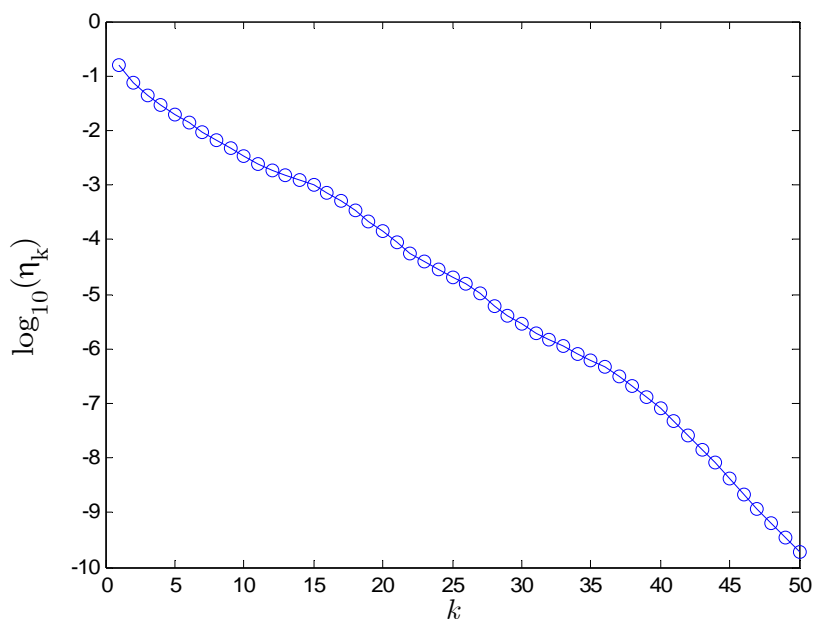


图 2.9 对数坐标系下最小 Ritz 值的收敛性

综上所述，Lanczos 算法最小 Ritz 值的误差几乎以指数形式收敛，对本问题的收敛率大约为 0.1678。

算例 2.6

下面将完全重正交化 Lanczos 算法应用到算例 2.5 中的矩阵的逆 A^{-1} 上产生 \tilde{T}_k 。

对 $k = 2, 3, \dots, 50$ ，计算 $\eta_k = \|\mu_{1,k}^{-1} - \lambda_n\|_2$ （这里 $\mu_{1,k}$ 和 λ_n 分别表示 \tilde{T}_k 的最大特征值和 A 的最小特征值），并且做图，观察其变化趋势，再与算例 2.5 中的结果比较来说明它们的优劣。

绘制最小 Ritz 值收敛情况如图 2.10 所示。从图中，我们可以看出，最小 Ritz 值收敛得很快，几乎以指数形式收敛甚至更快。

为描述得更加清晰些，我们在对数坐标系下绘图进行观察。如图 2.9 所示的是以 10 为底的对数坐标系的收敛曲线。

从图 2.11 中我们可以看出，对数坐标系下的误差曲线开始几乎为直线，后来几乎为 0，所以我们断定最小 Ritz 值的误差开始几乎以指数形式收敛的。经过回归分析，可以得到衰减阶段对数坐标系下的误差对应的线性回归曲线为 $y = -1.945x - 0.09977$ 。

综上所述，这种改进的算法最小 Ritz 值的误差几乎以指数形式收敛，对本问题的收敛率大约为 1.945。

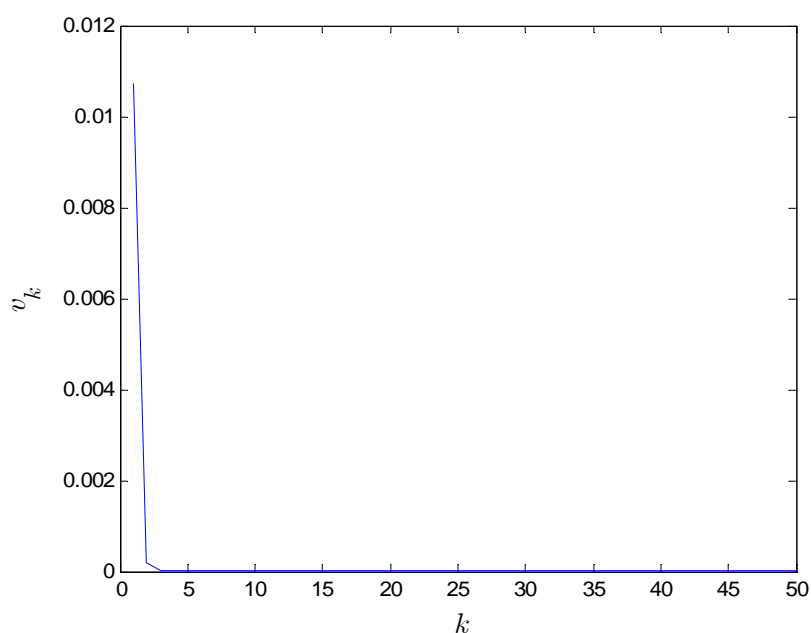


图 2.8 最小 Ritz 值的收敛性分析

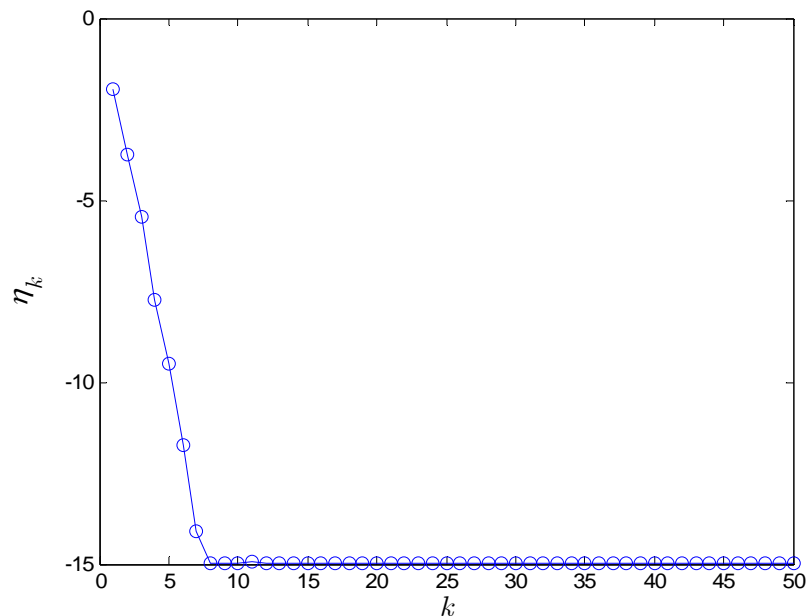


图 2.9 对数坐标系下最小 Ritz 值的收敛性

为定量分析，我们定义

定义 2.4 最小 Ritz 值的收敛率为

$$\eta_c = -\log(\eta_k / \eta_1)$$

则 Lanczos 算法应用于 A 上的最小 Ritz 值收敛率大约为 0.1678，应用于 A^{-1} 上的最小 Ritz 值收敛率大约为 1.945，改进后的算法收敛速度是改进前的 11.6 倍。这主要是因为 Lanczos 算法方法对于模较大的特征值收敛速度较快的缘故。此外，从收敛效果上看，改进后的算法在第 8 步就将误差 η_k 降到了 10^{-15} 数量级，而改进之前的算法直到第 50 步误差 η_k 还没有降到了 10^{-10} 数量级

3 奇异值分解的 Lanczos 算法

3.1 问题的提出

设 A 是一个 $m \times n$ 实矩阵。设计一种求 A 的几个最大奇异值和对应的奇异向量的 Lanczos 算法，并且用数值实验的方法来说明你所设计算法的数值特性。

3.2 算法的设计思路

设问题为：计算一个大规模矩阵 $A \in R^{m \times n}$ 的 k 个最大的奇异值及其对应的奇异向量，其中 k 要比 m, n 小得多。

不失一般性，假设 $m \geq n$ ，设 $\sigma_i, i = 1, 2, \dots, N$ 是 A 的奇异值， u_i, v_i 分别是对应的左右奇异向量，奇异值是从小到大的顺序排列的。称 (σ_i, u_i, v_i) 是矩阵 A 的一个奇异组。考虑下面的增广矩阵：

$$\tilde{A} = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \quad (3.1)$$

则 \tilde{A} 的特征值正好是 $\pm\sigma_i, i = 1, 2, \dots, N$ 和 $M - N$ 个零。 σ_i 和 $-\sigma_i$ 对应的特征向量分别为 $(u_i^T, v_i^T)^T$ 和 $(u_i^T, -v_i^T)^T$ ，零特征值对应的特征向量都具有 $(u_i^T, 0^T)^T$ 的形式。因此得到与原问题等价的问题：计算 \tilde{A} 的个最大 k (或最小) 的正特征值及其对应的特征向量。

3.3 双对角化 Lanczos 过程

首先给出双对角化 Lanczos 过程的算法。

算法 3.1 j 步双对角化 Lanczos 算法

1. 给定单位初始向量 q_1 ，令 $\beta_1 = 0, p_0 = 0$

2. 对 $i = 1, 2, \dots, j$ ，执行

$$(a) z_{2i} = Aq_i - \beta_i p_{i-1},$$

$$\alpha_i = \|z_{2i}\|, p_i = z_{2i} / \alpha_i,$$

$$(b) z_{2i+1} = Ap_i - \alpha_i p_i$$

$$\beta_{i+1} = \|z_{2i+1}\|, q_{i+1} = z_{2i+1} / \beta_{i+1},$$

令 $Q_m = (q_1, q_2, \dots, q_j)$, $P_m = (p_1, p_2, \dots, p_j)$, 则执行完上述双对角化 Lanczos 过程后有下面的矩阵等式:

$$A Q_j = P_j B_j \quad (3.2)$$

$$A^T P_j = Q_j B_j^T + \beta_{j+1} q_{j+1} e_j^T \quad (3.3)$$

其中

$$B_m = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ & \alpha_2 & \ddots & \\ & & \ddots & \beta_j \\ & & & \alpha_j \end{bmatrix}$$

它等价于以 $(0^T, q_1^T)^T$ 为初始向量, 对 \tilde{A} 执行 $2j$ 步 Lanczos 过程:

$$\tilde{A} V_{2j} = V_{2j} T_{2j} + \hat{z}_{2j+1} e_{2j}^T,$$

其中 V_{2j} 的 $2i-1$ 列为 $(0^T, q_1^T)^T$, 第 $2i$ 列为 $(p_1^T, 0^T)^T$,

$$\hat{z}_{2j+1} = (0^T, \beta_{j+1} q_{j+1}^T)$$

$$T_{2j} = \begin{bmatrix} 0 & \alpha_1 & & & \\ \alpha_1 & 0 & \beta_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_j & 0 & \alpha_j \\ & & & \alpha_j & 0 \end{bmatrix}$$

为对角元为零的对称三对角矩阵。在有限的精度运算下, P_j, Q_j 的正交性很快就会损失掉。解决这一问题的一种方法就是在计算过程中使用重正交技术, 即在算法中再重复执行一次 Gram-Schmidt 正交化过程。我们这里采用了 Larsen 提出的部分在正交化策略。

3.4 计算部分奇异值的双对角化 Lanczos 算法

假设 $(\tilde{\theta}_i, w_i, s_i)$ 是 B_j 的一个奇异组, 令

$$\tilde{u}_i = P_j w_i, \tilde{v}_i = Q_j s_i$$

由 (3.2) 和 (3.3) 我们可以得到

$$A^T \tilde{u}_i = \tilde{\theta}_i \tilde{v}_i + \beta_{j+1} e_j^T \tilde{u}_i q_{j+1}$$

如果 $\beta_{j+1} e_j^T \tilde{u}_i q_{j+1} = 0$, 则 $(\tilde{\theta}_i, \tilde{u}_i, \tilde{v}_i)$ 就是 A 的准确奇异值。如果用 $(\tilde{\theta}_i, \tilde{u}_i, \tilde{v}_i)$ 作为 A 的奇异组的近似, 则上述过程相当于一个 \tilde{A} 的不变子空间

$$E = \text{span} \left\{ \begin{pmatrix} P_j & 0 \\ 0 & Q_j \end{pmatrix} \right\}$$

上的正交投影, 即计算 $(\tilde{\theta}_i, \tilde{\varphi}_i) = (\tilde{\theta}_i, \frac{1}{\sqrt{2}} \begin{pmatrix} \tilde{u}_i \\ \tilde{v}_i \end{pmatrix})$ 满足

$$\begin{cases} \tilde{\varphi}_i \in E, \\ \tilde{A} \tilde{\varphi}_i - \tilde{\theta}_i \tilde{\varphi}_i \perp E. \end{cases}$$

称 $\tilde{\theta}_i$ 为 A 在 $\text{span}(P_j)$ 和 $\text{span}(Q_j)$ 上的 Ritz 值, \tilde{u}_i, \tilde{v}_i 对应的 Ritz 向量, 其残量为

$$\sqrt{\|A \tilde{v}_i - \tilde{\theta}_i \tilde{u}_i\|^2 + \|A^T \tilde{u}_i - \tilde{\theta}_i \tilde{v}_i\|^2} = \beta_{j+1} |e_j^T w_i|,$$

因此, 在方法收敛之前, 不需要显式地形成 \tilde{u}_i, \tilde{v}_i 。此外, 我们可以由 $\beta_{j+1} |e_j^T w_i|$ 的大小来决定 $(\tilde{\theta}_i, \tilde{u}_i, \tilde{v}_i)$ 是否可以作为 A 的近似特征对。

具体实现以上算法时, 我们采用了隐式重启技术。该算法只涉及矩阵乘向量 $y = Ax$, 因此可充分利用 A 的稀疏性和 A 所具有的特殊结构。

3.5 数值特性分析

3.5.1 奇异值收敛性分析

算例 3.1

先用 MATLAB 命令

```
>> aA=randint(1,NN,[1 N]);
>> bA=randint(1,NN,[1 M]);
>> cA=randint(1,NN,[2 8]);
>> A=sparse(aA,bA,cA,N,M);
```

产生一个对称的稀疏矩阵 A 。然后，对给定的自然数 k ，将计算部分奇异值的双对角化 Lanczos 算法应用到 A 。

图 3.1 显示的是每一步迭代得到的奇异值与精确奇异值之间的对比，其中横坐标表示迭代步数 k ，第 k 列的点表示第 k 步产生的所有特征值，最后一列的点表示 A 的精确奇异值。

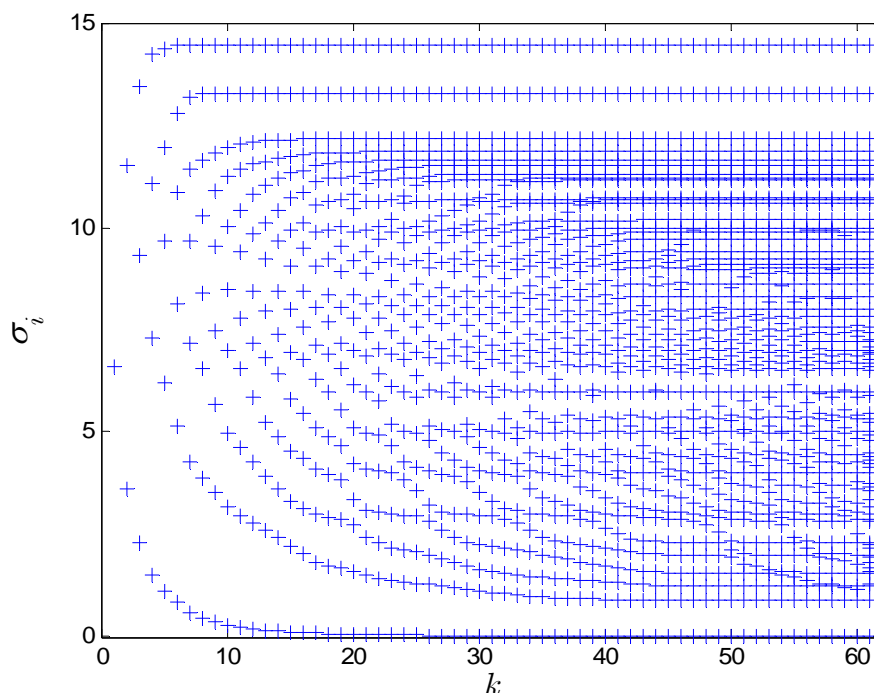


图 3.1 计算部分奇异值的双对角化 Lanczos 算法的收敛特性分析

图 3.1 展示了计算部分奇异值的双对角化 Lanczos 算法的收敛特性： $k = 8$ 时，就有两个 Ritz 值收敛到了 A 的最大的两个奇异值。

此外，通过图 3.1，从这个数值算例的计算结果可以看出：

- 最大的奇异值收敛最快，而较小的奇异值收敛较慢；
- T_k 的第 i 个最大（或最小）奇异值值单调增加（或减少）地收敛到 A 的第 i 个最大（或最小）特征值。

下面我们设计所设计的算法(LBD)与直接将 Lanczos 算法(LM)用于 $A^T A$ 的数值结果进行对比，以此来分析我们的算法的有效性。

在此前，我们设定停机准则为

$$stop = \max_{1 \leq i \leq k} \sqrt{\|A \tilde{v}_i - \tilde{\theta}_i \tilde{u}_i\|^2 + \|A^T \tilde{u}_i - \tilde{\theta}_i \tilde{v}_i\|^2}$$

如果

$$stop \leq tol \times \max\{\|B_j\|, 1\}$$

则

$$stop \Leftarrow \max_{1 \leq i \leq k} \frac{stop}{\|A\|_1}$$

当 $stop < tol$ 时停机。为了加速收敛性，Matlab 软件计算部分特征对的程序包 `eigs.m` 中，计算 $k+3$ 个特征对，使用 $m-(k+3)$ 个位移，我们的程序中也采取了这一策略。对于每个例子，算法均使用相同的按一致分布随机生成的单位初始向量， $tol=1.0 \times 10^{-6}$, $iter$ 表示重启的次数， $time$ 表示计算时间，单位为秒， $n.c.$ 表示重新启动 2000 次仍然步收敛。

3.5.1 算法的性能分析

算例 3.2

本算例分别计算了这些矩阵的最大的 3 个和 5 个奇异值及其对应的奇异向量，表 3.1 给出了计算结果。所有的计算结果都表明，LBD 方法要比 LM 方法要优越。

表 3.1 1850×712 阶，计算 k 个最大奇异组

	$k = 3$				$k = 5$			
	LM		LBD		LM		LBD	
j	iter	time	iter	time	iter	time	iter	time
10	825	162	413	44	n.c.	-	n.c.	-
15	177	95	80	26	320	142	164	41
20	79	81	53	30	118	113	89	48
25	46	79	41	37	65	104	47	42

4 结语

至此，我们完成了以下工作：

- 实现了非对称矩阵的重正交化的 **Arnoldi** 算法，并对计算误差的变化规律进行了分析，讨论了分解过程中的正交性损失情况，并用数值算例说明了重正交化过程的功效。最后绘制了特征值在复数平面上的分布，进行了收敛性分析。此外我们还利用双纽线理论进行了几何收敛率分析，直观地解释了收敛过程。
- 实现了对称矩阵的重正交化的 **Lanczos** 算法，并对计算误差的变化规律进行了分析，讨论了分解过程中的正交性损失情况，并用数值算例说明了重正交化过程的功效。绘制了特征值随迭代步数地变化曲线，进行了收敛性分析。最后我们对最小 **Ritz** 值的收敛性进行了详细地讨论，并说明了应用在 A^{-1} 时效果显著。
- 设计了一般矩阵的奇异值和对应奇异向量的 **Lanczos** 算法，通过数值实验对算法的收敛性和高效性进行了分析，说明了我们所设计算法的优越性。

总之，通过本次数值实验，我更加认识到了算法设计中的关键思想，即如何算得快，如何算得准。比如，一些重正交技术可大大的保持正交性；通过位移加速收敛；通过重启技术加速变准等等，这将会对我们在后面的学习和研究过程中帮助很大。