数值线性代数与算法

第四章 线性方程组的 Krylov 子空间迭代法




Back
Close

本章着重介绍求解大型稀疏线性方程组的 Krylov 子空间方法. 考虑线性方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4.1}$$

式中: 矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和向量 $\boldsymbol{b} \in \mathbb{R}^n$ 是已经给定的, 而 $\boldsymbol{x} \in \mathbb{R}^n$ 是待求的未知向量. 取初始向量 $\boldsymbol{x}_0 \in \mathbb{R}^n$, 则解方程组 (4.1) 等价于求解

$$\boldsymbol{A}\boldsymbol{z} = \boldsymbol{r}_0, \quad \boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{z}, \quad \boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0. \tag{4.2}$$

所谓求解线性方程组 (4.2) 的 Krylov 子空间方法, 就是先构造一个 Krylov 子空间

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\}; \tag{4.3}$$

然后设法求一个 $\boldsymbol{z}_k \in \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使其在某种意义下是方程组 (4.2) 解的最佳逼近, 然后得到原方程组 (4.1) 的近似解 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{z}_k$.

因此, 具体实现这一思想的关键是如何定量地刻画“最佳逼近”. 目前最常用的最佳性标准主要有两类: 一类是残量极小化标准; 另一类是残量正交化标准.

残量极小化标准是指求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得

$$\|\boldsymbol{r}_k\|_2 = \min\{\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\}, \qquad (4.4)$$

式中: $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 为 $\boldsymbol{x}_k$ 的残差向量 (简称残量).

本章主要介绍由这一标准导出的七种方法: 求解对称正定线性方程组的共轭梯度法 (CG 方法), 求解对称不定线性方程组的极小残量法 (MINRES 方法) 和 SYMMLQ 方法, 求解非对称线性方程组的广义极小残量法 (GMRES 方法)、拟极小残量法 (QMR 方法)、LSQR 方法和广义共轭残量法等.

Back

Close

残量正交化标准是指求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得

$$\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k \perp \mathcal{L}_k, \tag{4.5}$$

式中: $\mathcal{L}_k$ 是另一选定的 k 维子空间, 称为约束空间.

$\mathcal{L}_k$ 通常有三种典型取法: ① $\mathcal{L}_k = \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$; ② $\mathcal{L}_k = \boldsymbol{A}\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$; ③ $\mathcal{L}_k = \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{r}_0)$. 式 (4.5) 常称为 Galerkin 条件. 这里主要介绍这方面的三类方法: 双共轭梯度法 (BiCG 方法)、共轭梯度平方法 (CGS 方法) 和稳定化双共轭梯度法 (BICGSTAB 方法).

Krylov 子空间方法的一个最显著的特点就是整个计算过程只涉及矩阵 $\boldsymbol{A}$ 与某些向量的乘积, 这使得在算法执行过程中可以充分利用 $\boldsymbol{A}$ 的特殊性. 因此, 这种类型的线性方程组的求解方法就特别适用于大型稀疏线性方程组的求解. 当然, 对某些具有某种特殊结构的大型稠密线性方程组也是适用的.

Back

Close

§4.1 共轭梯度法

本节介绍求解对称正定线性方程组的一类最著名的 Krylov 子空间方法— 共轭梯度法.

§4.1.1 基本 CG 方法

设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 $\boldsymbol{b} \in \mathbb{R}^n$ 已经给定, 其中 $\boldsymbol{A}$ 是对称正定的. 对于给定初始向量 $\boldsymbol{x}_0 \in \mathbb{R}^n$, 求一个 $\boldsymbol{z} \in \mathbb{R}^n$ 满足式 (4.2), 便可得到方程组 (4.1) 的解 $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{z}$.

由于 $\boldsymbol{A}$ 是对称正定的, 所以在 $\mathbb{R}^n$ 上可以定义两种向量范数:

$$\|\boldsymbol{v}\|_{\boldsymbol{A}} = \sqrt{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}} \quad \text{和} \quad \|\boldsymbol{v}\|_{\boldsymbol{A}^{-1}} = \sqrt{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{v}}, \quad \boldsymbol{v} \in \mathbb{R}^n. \tag{4.6}$$

考虑由式 (4.2) 的系数矩阵 $\boldsymbol{A}$ 和右端项 $\boldsymbol{r}_0$ 所产生的 Krylov 子空间 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$. 并记式 (4.1) 的真解为 $\boldsymbol{x}^*$ 即 $\boldsymbol{x}^* = \boldsymbol{A}^{-1}\boldsymbol{b}$. 考虑求一

个 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使其在某种条件下是 $\boldsymbol{x}^*$ 的最佳逼近. 下面的定理是关于这种最佳性的几种等价表述.

定理 4.1 符号如上所述, 则下面三条等价:

(1) 向量 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 满足

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_{\boldsymbol{A}} = \min\{\|\boldsymbol{x} - \boldsymbol{x}^*\|_{\boldsymbol{A}} : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\} \quad (\text{误差极小}). \tag{4.7}$$

(2) 向量 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 满足

$$\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\|_{\boldsymbol{A}^{-1}} = \min\{\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_{\boldsymbol{A}^{-1}} : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\} \quad (\text{残量极小}). \tag{4.8}$$

(3) 向量 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 满足

$$\boldsymbol{z}^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k) = 0 \ (\text{即 } \boldsymbol{r}_k \perp \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)), \ \forall\, \boldsymbol{z} \in \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) \ (\text{残量正交}). \tag{4.9}$$




Back
Close

注 4.1 定理 4.1 说明, 求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 使其在 $\|\cdot\|_{\boldsymbol{A}}$ 范数下与真解 $\boldsymbol{x}^*$ 的距离最小就等价于其残量 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 在 $\|\cdot\|_{\boldsymbol{A}^{-1}}$ 范数下达到最小, 也等价于其残量 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 与 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 是正交的.

共轭梯度法基本迭代格式的推导需要利用对称 Lanczos 正交化过程. 假设已求得一个以 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$ 为初始向量的长度 k 的 Lanczos 分解 (见算法 2.13):

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k\boldsymbol{T}_k + \beta_k \boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}}, \tag{4.10}$$

式中: $\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k = \boldsymbol{I}_k$, $\boldsymbol{T}_k = \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k$ 为如下形式的对称三对角矩阵

$$\boldsymbol{T}_k = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-1} & \alpha_{k-1} & \beta_k \\ & & & \beta_k & \alpha_k \end{bmatrix},$$

且 $\boldsymbol{v}_{k+1}^{\mathrm{T}}\boldsymbol{V}_k = \boldsymbol{0}$, $\|\boldsymbol{v}_{k+1}\|_2 = 1$. 这里假定初始向量 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$, $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$. 经过 Lanzcos 正交化后, 得到的 $\boldsymbol{V}_k = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k]$, 其列向量已构成了 Krylov 子空间 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的一组标准正交基, 即有

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\} = \mathcal{R}(\boldsymbol{V}_k). \qquad (4.11)$$

现在希望求一个 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 使得式 (4.9) 成立. 由式 (4.11)

Back

Close

可知, 这就相当于求 $\boldsymbol{z}_k \in \mathbb{R}^k$, 使得

$$\boldsymbol{V}_k^{\mathrm{T}}(\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{z}_k) = \boldsymbol{V}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k) = \boldsymbol{0}, \tag{4.12}$$

式中: $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$.

注意到 $\boldsymbol{T}_k = \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k$ 和 $\boldsymbol{r}_0 = \|\boldsymbol{r}_0\|_2\boldsymbol{v}_1$, 由式 (4.12), 得

$$\boldsymbol{T}_k\boldsymbol{z}_k = \beta\boldsymbol{e}_1^{(k)}, \tag{4.13}$$

式中: $\beta = \|\boldsymbol{r}_0\|_2$, $\boldsymbol{e}_1^{(k)}$ 表示第 1 个分量为 1、其余分量为 0 的 k 维列向量.

注意到 $\boldsymbol{A}$ 对称正定蕴涵着 $\boldsymbol{T}_k$ 也是正定的, 便知 $\boldsymbol{T}_k$ 的 $\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}$ 分解存在, 设其为

$$\boldsymbol{T}_k = \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k^{\mathrm{T}}, \tag{4.14}$$

式中: $\boldsymbol{L}_k$ 为单位下三角矩阵; $\boldsymbol{D}_k = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_k)$, $\delta_i > 0$,

$i = 1, 2, \cdots, k$. $\boldsymbol{T}_k$ 的三对角结构蕴涵着 $\boldsymbol{L}_k$ 必有如下形状:

$$\boldsymbol{L}_k = \begin{bmatrix} 1 & & & \\ \gamma_1 & 1 & & \\ & \ddots & \ddots & \\ & & \gamma_{k-1} & 1 \end{bmatrix}. \tag{4.15}$$

利用分解 (4.14), 线性方程组 (4.13) 的解可以表示为

$$\boldsymbol{z}_k = \boldsymbol{L}_k^{-\mathrm{T}} \boldsymbol{D}_k^{-1} \boldsymbol{L}_k^{-1} (\beta \boldsymbol{e}_1^{(k)}),$$

从而有

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}} \boldsymbol{D}_k^{-1} \boldsymbol{L}_k^{-1} (\beta \boldsymbol{e}_1^{(k)}). \tag{4.16}$$

再令

$$\widetilde{\boldsymbol{P}}_k = [\tilde{\boldsymbol{p}}_1, \tilde{\boldsymbol{p}}_2, \cdots, \tilde{\boldsymbol{p}}_k] = \boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}},$$

$$\boldsymbol{z}_k = (\zeta_1, \zeta_2, \cdots, \zeta_k)^{\mathrm{T}} = \boldsymbol{D}_k^{-1} \boldsymbol{L}_k^{-1} (\beta \boldsymbol{e}_1^{(k)}),$$

Back

Close

则式 (4.16) 又可写为

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \widetilde{\boldsymbol{P}}_k \boldsymbol{z}_k. \tag{4.17}$$

注意到

$$\left[\begin{array}{c:c} \boldsymbol{T}_k & \beta_k \boldsymbol{e}_k \\ \hdashline \beta_k \boldsymbol{e}_k^{\mathrm{T}} & \alpha_{k+1} \end{array}\right] = \boldsymbol{T}_{k+1} := \boldsymbol{L}_{k+1} \boldsymbol{D}_{k+1} \boldsymbol{L}_{k+1}^{\mathrm{T}}$$

$$= \left[\begin{array}{c:c} \boldsymbol{L}_k & \boldsymbol{0} \\ \hdashline \gamma_k \boldsymbol{e}_k^{\mathrm{T}} & 1 \end{array}\right] \left[\begin{array}{c:c} \boldsymbol{D}_k & \boldsymbol{0} \\ \hdashline \boldsymbol{0} & \delta_{k+1} \end{array}\right] \left[\begin{array}{c:c} \boldsymbol{L}_k & \boldsymbol{0} \\ \hdashline \gamma_k \boldsymbol{e}_k^{\mathrm{T}} & 1 \end{array}\right]^{\mathrm{T}},$$

式中: $\boldsymbol{e}_k$ 表示第 k 个分量为 1、其余分量为 0 的 k 维列向量. 于是

有

$$\begin{aligned}
\widetilde{\boldsymbol{P}}_{k+1} &= \boldsymbol{V}_{k+1}\boldsymbol{L}_{k+1}^{-\mathrm{T}} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] \left[\begin{array}{c|c} \boldsymbol{L}_k & \boldsymbol{0} \\ \hline \gamma_k \boldsymbol{e}_k^{\mathrm{T}} & 1 \end{array}\right]^{-\mathrm{T}} \\
&= [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] \left[\begin{array}{c|c} \boldsymbol{L}_k^{-\mathrm{T}} & -\gamma_k \boldsymbol{L}_k^{-\mathrm{T}} \boldsymbol{e}_k^{(k)} \\ \hline \boldsymbol{0} & 1 \end{array}\right] \\
&= [\boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}}, \boldsymbol{v}_{k+1} - \gamma_k \boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}} \boldsymbol{e}_k^{(k)}] \\
&= [\widetilde{\boldsymbol{P}}_k, \boldsymbol{v}_{k+1} - \gamma_k \widetilde{\boldsymbol{P}}_k \boldsymbol{e}_k^{(k)}] := [\widetilde{\boldsymbol{P}}_k, \widetilde{\boldsymbol{p}}_{k+1}].
\end{aligned}$$

由此可得

$$\widetilde{\boldsymbol{p}}_{k+1} = \boldsymbol{v}_{k+1} - \gamma_k \widetilde{\boldsymbol{P}}_k \boldsymbol{e}_k^{(k)} = \boldsymbol{v}_{k+1} - \gamma_k \widetilde{\boldsymbol{p}}_k, \tag{4.18}$$

$$
\begin{aligned}
\boldsymbol{z}_{k+1} &= \boldsymbol{D}_{k+1}^{-1}\boldsymbol{L}_{k+1}^{-1}(\beta\boldsymbol{e}_1^{(k+1)}) \\
&= \left[\begin{array}{c:c} \boldsymbol{D}_k^{-1} & \boldsymbol{0} \\ \hdashline \boldsymbol{0} & \delta_{k+1}^{-1} \end{array}\right] \left[\begin{array}{c:c} \boldsymbol{L}_k^{-1} & \boldsymbol{0} \\ \hdashline -\gamma_k \boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{L}_k^{-1} & 1 \end{array}\right] (\beta\boldsymbol{e}_1^{(k+1)}) \\
&= \begin{bmatrix} \boldsymbol{D}_k^{-1}\boldsymbol{L}_k^{-1}(\beta\boldsymbol{e}_1^{(k)}) \\ \zeta_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_k \\ \zeta_{k+1} \end{bmatrix},
\end{aligned}
$$

式中: $\zeta_{k+1} = -\beta\gamma_k\delta_{k+1}^{-1}\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{L}_k^{-1}\boldsymbol{e}_1^{(k)}$. 从而有

$$
\boldsymbol{x}_{k+1} = \boldsymbol{x}_0 + \widetilde{\boldsymbol{P}}_{k+1}\boldsymbol{z}_{k+1} = \boldsymbol{x}_0 + [\widetilde{\boldsymbol{P}}_k, \widetilde{\boldsymbol{p}}_{k+1}] \begin{bmatrix} \boldsymbol{z}_k \\ \zeta_{k+1} \end{bmatrix} = \boldsymbol{x}_k + \zeta_{k+1}\widetilde{\boldsymbol{p}}_{k+1}. \tag{4.19}
$$

由式 (4.19), 得

$$\begin{aligned} \boldsymbol{r}_{k+1} &= \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{k+1} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k - \zeta_{k+1}\boldsymbol{A}\widetilde{\boldsymbol{p}}_{k+1} \\ &= \boldsymbol{r}_k - \zeta_{k+1}\boldsymbol{A}\widetilde{\boldsymbol{p}}_{k+1}. \end{aligned} \tag{4.20}$$

这样, 综合上面的推导, 得如下三个基本的迭代公式:

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \zeta_{k+1}\widetilde{\boldsymbol{p}}_{k+1}, \\ \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \zeta_{k+1}\boldsymbol{A}\widetilde{\boldsymbol{p}}_{k+1}, \\ \widetilde{\boldsymbol{p}}_{k+1} = \boldsymbol{v}_{k+1} - \gamma_k\widetilde{\boldsymbol{p}}_k. \end{cases} \tag{4.21}$$

下面设法消去式 (4.21) 第 3 式中的 $\boldsymbol{v}_{k+1}$. 由 $\boldsymbol{r}_0 \in \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 可知, $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k \in \mathcal{K}_{k+1}(\boldsymbol{A}, \boldsymbol{r}_0)$, 从而 $\boldsymbol{r}_k$ 可以由 $\boldsymbol{V}_{k+1}$ 的列向量线性表出, 即

$$\boldsymbol{r}_k = \widetilde{\gamma}_1\boldsymbol{v}_1 + \widetilde{\gamma}_2\boldsymbol{v}_2 + \cdots + \widetilde{\gamma}_k\boldsymbol{v}_k + \widetilde{\gamma}_{k+1}\boldsymbol{v}_{k+1}.$$




Back
Close

又由于 $\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{r}_k = \boldsymbol{0}$, 故 $\widetilde{\gamma}_i = 0,\ i = 1, 2, \cdots, k$, 从而

$$\boldsymbol{r}_k = \widetilde{\gamma}_{k+1}\boldsymbol{v}_{k+1}. \tag{4.22}$$

由式 (4.22) 可知 $|\widetilde{\gamma}_{k+1}| = \|\boldsymbol{r}_k\|_2$, 不妨设 $\widetilde{\gamma}_{k+1} = \|\boldsymbol{r}_k\|_2$, 并定义

$$\boldsymbol{p}_{k+1} = \|\boldsymbol{r}_k\|_2\widetilde{\boldsymbol{p}}_{k+1}, \tag{4.23}$$

则由式 (4.21) 和式 (4.22), 得

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + \frac{\zeta_{k+1}}{\|\boldsymbol{r}_k\|_2}\boldsymbol{p}_{k+1},\\
\boldsymbol{r}_{k+1} &= \boldsymbol{r}_k - \frac{\zeta_{k+1}}{\|\boldsymbol{r}_k\|_2}\boldsymbol{A}\boldsymbol{p}_{k+1},\\
\boldsymbol{p}_{k+1} &= \boldsymbol{r}_k - \frac{\|\boldsymbol{r}_k\|_2\gamma_k}{\|\boldsymbol{r}_{k-1}\|_2}\boldsymbol{p}_k.
\end{aligned}$$

令

$$\alpha_{k+1} = \frac{\zeta_{k+1}}{\|\boldsymbol{r}_k\|_2}, \quad \beta_k = -\frac{\|\boldsymbol{r}_k\|_2\gamma_k}{\|\boldsymbol{r}_{k-1}\|_2},$$

即得

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_{k+1}\boldsymbol{p}_{k+1}, \\ \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_{k+1}\boldsymbol{A}\boldsymbol{p}_{k+1}, \\ \boldsymbol{p}_{k+1} = \boldsymbol{r}_k + \beta_k \boldsymbol{p}_k. \end{cases} \tag{4.24}$$

下面导出不涉及分解式 (4.14) 的信息计算 α_{k+1} 和 β_k 的公式. 为此, 先导出向量组 $\{\boldsymbol{p}_i\}$ 和 $\{\boldsymbol{r}_i\}$ 所具有的基本性质.

由 $\widetilde{\boldsymbol{P}}_k$ 的定义, 可以立即导出

$$\widetilde{\boldsymbol{P}}_k^{\mathrm{T}} \boldsymbol{A} \widetilde{\boldsymbol{P}}_k = \boldsymbol{L}_k^{-1} \boldsymbol{V}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}} = \boldsymbol{D}_k.$$

这表明 $\widetilde{\boldsymbol{p}}_i$ 之间是 $\boldsymbol{A}$ 正交的 (或称为 $\boldsymbol{A}$ 共轭的), 即有

$$\widetilde{\boldsymbol{p}}_i^{\mathrm{T}} \boldsymbol{A} \widetilde{\boldsymbol{p}}_j = 0, \quad i \neq j.$$

于是有

$$\boldsymbol{p}_i^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_j = 0, \quad i \neq j. \tag{4.25}$$




Back
Close

此外, 由 $\widetilde{\boldsymbol{P}}_k$ 的定义和式 (4.23) 蕴涵着

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathcal{R}(\boldsymbol{V}_k) = \mathcal{R}(\widetilde{\boldsymbol{P}}_k) = \mathrm{span}\{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_k\}, \tag{4.26}$$

即 $\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_k$ 是 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的一组 $\boldsymbol{A}$ 正交基, 再由式 (4.22) 和 $\boldsymbol{v}_i$ 的相互正交性, 又可以导出

$$\boldsymbol{r}_i^{\mathrm{T}} \boldsymbol{r}_j = 0, \quad i \neq j. \tag{4.27}$$

即残量是相互正交的. 这样, 再由 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathcal{R}(\boldsymbol{V}_k)$ 便有

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}\}, \tag{4.28}$$

即 $\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}$ 构成了 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的一组正交基.

在式 (4.24) 的第 3 式两边左乘 $\boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A}$ 并利用式 (4.25), 得

$$0 = \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_{k+1} = \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{r}_k + \beta_k \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k.$$

于是有

$$\beta_k = -\frac{\boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{r}_k}{\boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k}. \tag{4.29}$$

再在式 (4.24) 的第 2 式两边左乘 $\boldsymbol{r}_k^{\mathrm{T}}$ 并利用式 (4.27), 得

$$0 = \boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{r}_{k+1} = \boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{r}_k - \alpha_{k+1} \boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_{k+1}.$$

因此又有

$$\alpha_{k+1} = \frac{\boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{r}_k}{\boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_{k+1}}. \tag{4.30}$$

这样就得到了不需要分解式 (4.14) 的信息就可计算式 (4.24) 中的系数 β_k 和 α_{k+1} 的公式.

事实上, 利用式 (4.25) 和式 (4.27) 还可导出更有效的计算公式. 首先在式 (4.24) 的第 3 式两边左乘 $\boldsymbol{p}_{k+1}^{\mathrm{T}} \boldsymbol{A}$, 并利用式 (4.25), 得

$$\boldsymbol{p}_{k+1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_{k+1} = \boldsymbol{p}_{k+1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{r}_k = \boldsymbol{r}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_{k+1}. \tag{4.31}$$

Back

Close

将式 (4.24) 第 2 式中的 k 用 $k-1$ 替换, 有

$$\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{A}\boldsymbol{p}_k.$$

将上式两边分别左乘 $\boldsymbol{r}_k^{\mathrm{T}}$ 和 $\boldsymbol{r}_{k-1}^{\mathrm{T}}$, 并利用式 (4.27), 得

$$\boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k = -\alpha_k \boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k, \tag{4.32}$$

$$\boldsymbol{r}_{k-1}^{\mathrm{T}}\boldsymbol{r}_{k-1} = \alpha_k \boldsymbol{r}_{k-1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k = \alpha_k \boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k, \tag{4.33}$$

这里式 (4.33) 的第 2 个等号利用了式 (4.31).

现将式 (4.31) 代入式 (4.30), 得

$$\alpha_{k+1} = \frac{\boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k}{\boldsymbol{p}_{k+1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_{k+1}}. \tag{4.34}$$

再将式 (4.32)、式 (4.33) 与式 (4.29) 相结合, 有

$$\beta_k = \frac{\boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k}{\boldsymbol{r}_{k-1}^{\mathrm{T}}\boldsymbol{r}_{k-1}}. \tag{4.35}$$

显然, 这两个计算公式比原来的计算公式减少了内积和矩阵乘向量的次数, 因此更有效.

综合上述讨论, 可得共轭梯度法 (简称 CG 方法) 求解对称正定线性方程组的基本迭代格式如下.

选取初值 $\boldsymbol{x}_0$, 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_1 = \boldsymbol{r}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

$$k = k + 1;$$

$$\alpha_k = \frac{(\boldsymbol{r}_{k-1}, \boldsymbol{r}_{k-1})}{(\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}; \quad \boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k \boldsymbol{p}_k;$$

$$\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{A}\boldsymbol{p}_k;$$

$$\beta_k = \frac{(\boldsymbol{r}_k, \boldsymbol{r}_k)}{(\boldsymbol{r}_{k-1}, \boldsymbol{r}_{k-1})}; \quad \boldsymbol{p}_{k+1} = \boldsymbol{r}_k + \beta_k \boldsymbol{p}_k;$$

end

为了优化算法程序 (考虑计算量和存储量), 通常写成如下形式.

算法 4.1 (CG 方法) 给定对称正定方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 和 $\varepsilon>0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2\leqslant\varepsilon$, 其中 $\boldsymbol{r}_k=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; $\boldsymbol{r}_0=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_1=\boldsymbol{r}_0$; $\rho_0=\boldsymbol{r}_0^{\mathrm{T}}\boldsymbol{r}_0$; $k=0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2>\varepsilon)$

$k=k+1$;

$\boldsymbol{z}_k=\boldsymbol{A}\boldsymbol{p}_k$; $\alpha_k=\rho_{k-1}/\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{p}_k$;

$\boldsymbol{x}_k=\boldsymbol{x}_{k-1}+\alpha_k\boldsymbol{p}_k$; $\boldsymbol{r}_k=\boldsymbol{r}_{k-1}-\alpha_k\boldsymbol{z}_k$;

$\rho_k=\boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k$; $\beta_k=\rho_k/\rho_{k-1}$;

$\boldsymbol{p}_{k+1}=\boldsymbol{r}_k+\beta_k\boldsymbol{p}_k$;

end

算法 4.1 的迭代终止准则也可以用 $\rho_k\leqslant\varepsilon\rho_0$, 其中 $\varepsilon>0$ 是给

定的允许误差. CG 方法的 MATLAB 程序如下:

```
%共轭梯度法程序-mcg.m
function [x,iter,time,res,resvec]=mcg(A,b,x,max_it,tol)
%输入:系数阵A,右端量b,初始值x,容许误差tol,最大迭代数max_it
%输出:解向量x,迭代次数iter,CPU时间time,
%      终止时相对残差模res,相对残差模向量resvec
tic;r=b-A*x;p=r;rho=r'*r;
mr=sqrt(rho);iter=0;
while (iter<max_it)
    iter=iter+1;
    z=A*p; alpha=rho/(z'*p);
    x=x+alpha*p; r=r-alpha*z;
    rho1=r'*r; beta=rho1/rho;
```




Back
Close

```
        p=r+beta*p;
        res=sqrt(rho1)/mr;
        resvec(iter)=res;
        if (res<tol), break; end
        rho=rho1;
    end
    time=toc;
```

例 4.1 假定线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的系数矩阵 $\boldsymbol{A}$ 和右端项 $\boldsymbol{b}$ 分别为

$$\boldsymbol{A}=\begin{bmatrix} 1 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & n \end{bmatrix}, \quad \boldsymbol{b}=\boldsymbol{A}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}.$$

显然, 此方程组的真解为 $\boldsymbol{x}^* = (1, 1, \cdots, 1)^{\mathrm{T}}$. 应用共轭梯度法求解该线性方程组, 取 $n = 1000$, 迭代 193 步后得到的近似解 $\widehat{\boldsymbol{x}} = \boldsymbol{x}_{193}$ 满足

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 3.7417 \times 10^{-8},$$

迭代过程的收敛轨迹如图 4.1 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差.

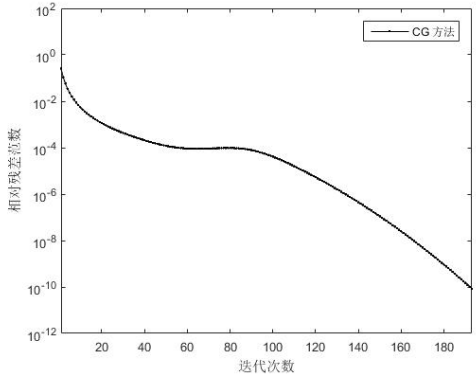


图 4.1 CG 算法的收敛特性

§4.1.2 收敛性分析

在没有误差的情况下, 由式 (4.27) 可知, 必存在一个 $\ell \leqslant n$, 使得 $\boldsymbol{r}_\ell = \boldsymbol{0}$, 即有 $\boldsymbol{x}^* = \boldsymbol{x}_\ell$. 换言之, 算法在有限步可得到方程组 (4.1) 的精确解. 从这个意义上而言, 共轭梯度法是一种直接法. 但实际使用时, 一般是将其作为一种迭代法来用的, 其原因有二: 一是在有误差的计算机上, 有限步终止的结论已经不再成立; 二是即使出现了 $\boldsymbol{r}_\ell = \boldsymbol{0}$, 但如果 ℓ 非常大, 也是难以忍受的, 因为实际应用中所考虑的问题其阶数 n 是十分巨大的.

下面介绍共轭梯度法 (作为一种迭代法) 的收敛性和收敛速度. 由于收敛性分析需要利用 Chebyshev 多项式的性质. 在第 3 章已

给出 k 次 Chebyshev 多项式 $C_k(t)$ 的定义:

$$C_k(t) = \begin{cases} \cos(k \arccos t), & |t| \leqslant 1, \\ \cosh(k \operatorname{arccosh} t), & |t| > 1. \end{cases} \tag{4.36}$$

下面再回顾一下 Chebyshev 多项式的一些最常用的性质.

定理 4.2 设 $C_k(t)$ 是由式 (4.36) 定义的 Chebyshev 多项式, 则它具有如下性质:

(1) $C_k(t)$ 具有递推式

$$C_0(t) = 1;\ C_1(t) = t;\ C_{k+1}(t) = 2tC_k(t) - C_{k-1}(t),\ k = 1, 2, \cdots.$$

(2) 对于 $|t| > 1$, $C_k(t)$ 有表达式

$$C_k(t) = \frac{1}{2}\left[\left(t + \sqrt{t^2 - 1}\right)^k + \left(t + \sqrt{t^2 - 1}\right)^{-k}\right]. \tag{4.37}$$

(3) 对任意的 $t \in [-1,1]$, 有 $|C_k(t)| \leqslant 1$, 且

$$C_k(t_i^{(k)}) = (-1)^i, \quad t_i^{(k)} = \cos\frac{i\pi}{k}, \quad i = 0, 1, \cdots, k,$$

即 $C_k(t)$ 在 $[-1,1]$ 上刚好有 $k+1$ 个极值点, 在这些点上交错地取 1 和 -1.

(4) 对任意的 $s > 1$, 有

$$\min_{p\in\mathcal{P}_k,\, p(s)=1} \max_{t\in[-1,1]} |p(t)| = \frac{1}{C_k(s)}, \tag{4.38}$$

式中:

$$\mathcal{P}_k = \{p : p \text{ 是次数不超过 } k \text{ 的实系数多项式}\},$$

而且上述极小极大问题有唯一解

$$p(t) = \frac{C_k(t)}{C_k(s)}. \tag{4.39}$$

证明 (1) 对于 $|t| \leqslant 1$, 令 $\theta = \arccos t$, 利用余弦函数的和差化积公式

$$\cos\alpha + \cos\beta = 2\cos\frac{\alpha+\beta}{2}\cos\frac{\alpha-\beta}{2},$$

有

$$\begin{aligned} C_{k+1}(t) + C_{k-1}(t) &= \cos(k+1)\theta + \cos(k-1)\theta \\ &= 2\cos k\theta\cos\theta = 2tC_k(t). \end{aligned}$$

对于 $|t| > 1$, 令 $\theta = \operatorname{arccosh}(t)$, 利用双曲余弦函数的定义

$$\cosh(\theta) = \frac{\mathrm{e}^{\theta} + \mathrm{e}^{-\theta}}{2},$$

有

$$
\begin{aligned}
C_{k+1}(t) + C_{k-1}(t) &= \cosh(k+1)\theta + \cosh(k-1)\theta \\
&= \frac{\mathrm{e}^{(k+1)\theta} + \mathrm{e}^{-(k+1)\theta}}{2} + \frac{\mathrm{e}^{(k-1)\theta} + \mathrm{e}^{-(k-1)\theta}}{2} \\
&= \mathrm{e}^{k\theta} \cdot \frac{\mathrm{e}^{\theta} + \mathrm{e}^{-\theta}}{2} + \mathrm{e}^{-k\theta} \cdot \frac{\mathrm{e}^{\theta} + \mathrm{e}^{-\theta}}{2} \\
&= 2 \cdot \frac{\mathrm{e}^{\theta} + \mathrm{e}^{-\theta}}{2} \cdot \frac{\mathrm{e}^{k\theta} + \mathrm{e}^{-k\theta}}{2} \\
&= 2\cosh\theta\cosh(k\theta) = 2tC_k(t).
\end{aligned}
$$

(2) 对于 $|t| > 1$, 令 $\theta = \operatorname{arccosh}(t)$, 有 $t = \cosh(\theta) = \mathrm{e}^{\theta} + \mathrm{e}^{-\theta}/2$, 即

$$
\left(\mathrm{e}^{\theta}\right)^2 - 2t\mathrm{e}^{\theta} + 1 = 0.
$$

得

$$\mathrm{e}^{\theta}=\frac{2t+\sqrt{4t^2-4}}{2}=t+\sqrt{t^2-1}.$$

于是, 有

$$C_k(t)=\cosh(k\theta)=\frac{1}{2}\left(\mathrm{e}^{k\theta}+\mathrm{e}^{-k\theta}\right)=\frac{1}{2}\left[\left(\mathrm{e}^{\theta}\right)^k+\left(\mathrm{e}^{\theta}\right)^{-k}\right]$$
$$=\frac{1}{2}\left[\left(t+\sqrt{t^2-1}\right)^k+\left(t+\sqrt{t^2-1}\right)^{-k}\right].$$

(3) $\forall\, t\in[-1,1]$, $|C_k(t)|\leqslant 1$ 成立是显然的. 直接验证, 得

$$C_k(t_i^{(k)})=\cos(k\arccos t_i^{(k)})=\cos(i\pi)=(-1)^i.$$

(4) 注意到 $s>1$, $p(t)$ 是 k 次多项式且 $p(s)=1$. 记

$$\alpha=1/C_k(s)>0,\quad p_{\max}=\max_{-1\leqslant t\leqslant 1}|p(t)|.$$

若 $p_{\max} < \alpha$, 则

$$(-1)^i \left(\alpha\, C_k(t_i^{(k)}) - p(t_i^{(k)})\right) = (-1)^i \alpha (-1)^i - (-1)^i p(t_i^{(k)})$$
$$= \alpha - (-1)^i p(t_i^{(k)}) \geqslant \alpha - p_{\max} > 0.$$

这表明多项式 $\psi = \alpha\, C_k(t) - p(t)$ 在 $t_i^{(k)}$ $(i = 0, 1, \cdots, k)$ 处有交替的正负号. 所以它在 $[-1, 1]$ 中有 k 个零点. 但 $s \notin [-1, 1]$ 也是 k 次多项式 $\alpha\, C_k(t) - p(t)$ 的零点, 从而 k 次多项式 $\alpha\, C_k(t) - p(t)$ 有 $k+1$ 个零点, 矛盾. 故必有

$$\max_{-1 \leqslant t \leqslant 1} |p(t)| \geqslant \alpha = \frac{1}{C_k(s)}.$$

若取 $p(t) = \dfrac{C_k(t)}{C_k(s)}$, 则

$$\max_{-1 \leqslant t \leqslant 1} |p(t)| = \max_{-1 \leqslant t \leqslant 1} \left|\frac{C_k(t)}{C_k(s)}\right| = \frac{\max\limits_{-1 \leqslant t \leqslant 1} |C_k(t)|}{C_k(s)} = \frac{1}{C_k(s)}.$$




Back
Close

由此可得

$$\min_{p\in\mathcal{P}_k,\,p(s)=1}\ \max_{t\in[-1,1]}|p(t)|=\frac{1}{C_k(s)}.$$

证毕. □

推论 4.1 设 $a<b<1$ (或 $1<a<b$). 若 $p\in\mathcal{P}_k$ 且 $p(1)=1$. 则

$$\min_{p\in\mathcal{P}_k,\,p(1)=1}\ \max_{t\in[a,b]}|p(t)|=\frac{1}{C_k(w(1))},\tag{4.40}$$

式中:

$$w(t)=\begin{cases}\dfrac{2t-a-b}{b-a}, & a<b<1,\\[2ex] \dfrac{2t-a-b}{a-b}, & 1<a<b.\end{cases}$$

且上述极小极大问题的唯一解为

$$p(t)=\frac{C_k(w(t))}{C_k(w(1))}.$$

现在考虑 CG 方法的收敛性分析. 由定理 4.1 和前面的推导过程可知, 共轭梯度法第 k 步得到的近似解 $\boldsymbol{x}_k$ 满足

$$\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_{\boldsymbol{A}} = \min\left\{\|\boldsymbol{x} - \boldsymbol{x}^*\|_{\boldsymbol{A}} : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\right\}.$$

由 Krylov 子空间的性质可知, $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的充分必要条件是存在 $\varphi \in \mathcal{P}_{k-1}$ 使得 $\boldsymbol{x} = \boldsymbol{x}_0 + \varphi(\boldsymbol{A})\boldsymbol{r}_0$, 这里 $\mathcal{P}_{k-1}$ 表示次数不超过 $k-1$ 的多项式的全体. 于是, 对任意的 $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 有

$$\boldsymbol{x}^* - \boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} - \boldsymbol{x}_0 - \varphi(\boldsymbol{A})\boldsymbol{r}_0 = \boldsymbol{A}^{-1}\psi(\boldsymbol{A})\boldsymbol{r}_0,$$

式中: $\psi(t) = 1 - t\varphi(t)$.

设 $\boldsymbol{A}$ 的谱分解为 $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}$, 其中 $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_n]$ 是正交矩阵, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$, $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n > 0$. 现将

$\boldsymbol{r}_0$ 按 $\boldsymbol{A}$ 的特征向量展开, 有

$$\boldsymbol{r}_0 = \eta_1\boldsymbol{u}_1 + \eta_2\boldsymbol{u}_2 + \cdots + \eta_n\boldsymbol{u}_n = \sum_{i=1}^{n}\eta_i\boldsymbol{u}_i.$$

这样, 便有

$$\begin{aligned}
\|\boldsymbol{x}^* - \boldsymbol{x}\|_{\boldsymbol{A}}^2 &= \|\boldsymbol{A}^{-1}\psi(\boldsymbol{A})\boldsymbol{r}_0\|_{\boldsymbol{A}}^2 = \big(\boldsymbol{A}^{-1}\psi(\boldsymbol{A})\boldsymbol{r}_0, \psi(\boldsymbol{A})\boldsymbol{r}_0\big) \\
&= \left(\sum_{i=1}^{n}\eta_i\boldsymbol{A}^{-1}\psi(\boldsymbol{A})\boldsymbol{u}_i, \sum_{j=1}^{n}\eta_j\psi(\boldsymbol{A})\boldsymbol{u}_j\right) \\
&= \left(\sum_{i=1}^{n}\eta_i\psi(\lambda_i)\lambda_i^{-1}\boldsymbol{u}_i, \sum_{j=1}^{n}\eta_j\psi(\lambda_j)\boldsymbol{u}_j\right) \\
&= \sum_{i=1}^{n}\eta_i^2\psi^2(\lambda_i)\lambda_i^{-1} \leqslant \max_{1\leqslant i\leqslant n}\psi^2(\lambda_i)\sum_{i=1}^{n}\lambda_i^{-1}\eta_i^2 \\
&= \max_{1\leqslant i\leqslant n}\psi^2(\lambda_i)\|\boldsymbol{x}^* - \boldsymbol{x}_0\|_{\boldsymbol{A}}^2,
\end{aligned}$$

从而有

$$\frac{\|\boldsymbol{x}^* - \boldsymbol{x}\|_{\boldsymbol{A}}}{\|\boldsymbol{x}^* - \boldsymbol{x}_0\|_{\boldsymbol{A}}} \leqslant \min_{\psi \in \mathcal{P}_k^0} \max_{1 \leqslant i \leqslant n} |\psi(\lambda_i)|, \tag{4.41}$$

式中: $\mathcal{P}_k^0 = \{\psi \in \mathcal{P}_k : \psi(0) = 1\}$. 令

$$a = \min_{1 \leqslant i \leqslant n} \lambda_i = \lambda_n, \quad b = \max_{1 \leqslant i \leqslant n} \lambda_i = \lambda_1,$$

则由推论 4.1, 知极小极大问题

$$\min_{\psi \in \mathcal{P}_k^0} \max_{\lambda \in [a,b]} |\psi(\lambda)|$$

有唯一的解

$$\widehat{\psi}(\lambda) = \frac{C_k\Big(\dfrac{b + a - 2\lambda}{b - a}\Big)}{C_k\Big(\dfrac{b + a}{b - a}\Big)},$$

而且有

$$\min_{\psi\in\mathcal{P}_k^0}\max_{\lambda\in[a,b]}|\psi(\lambda)|=\frac{1}{C_k\Big(\dfrac{b+a}{b-a}\Big)}, \tag{4.42}$$

这里假设了 $1<a<b$. 将式 (4.41) 和式 (4.42) 相结合, 可证明如下结果.

定理 4.3 共轭梯度法产生的近似解 $\boldsymbol{x}_k$ 满足

$$\frac{\|\boldsymbol{x}_k-\boldsymbol{x}^*\|_{\boldsymbol{A}}}{\|\boldsymbol{x}_0-\boldsymbol{x}^*\|_{\boldsymbol{A}}}\leqslant\frac{1}{C_k\Big(1+\dfrac{2}{\kappa-1}\Big)}, \tag{4.43}$$

式中: C_k 为 k 阶 Chebyshev 多项式, $\kappa=\kappa_2(\boldsymbol{A})=\|\boldsymbol{A}\|_2\cdot\|\boldsymbol{A}^{-1}\|_2=b/a$.

注 4.2 由定理 4.2 的结论 (2), 得

$$\left[C_k\left(1+\frac{2}{\kappa-1}\right)\right]^{-1}=\left[C_k\left(\frac{b+a}{b-a}\right)\right]^{-1}=\frac{2\sigma^k}{1+\sigma^{2k}}, \tag{4.44}$$

式中: $\sigma=(\sqrt{b}-\sqrt{a})/(\sqrt{b}+\sqrt{a})$. 由式 (4.44) 可证

$$\left[C_k\left(1+\frac{2}{\kappa-1}\right)\right]^{-1}\leqslant 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k. \tag{4.45}$$

由此可知, 系数矩阵的条件数 κ 越小, 共轭梯度法收敛就越快.

§4.1.3 预处理 CG 方法

由式 (4.43), 在实际使用共轭梯度法时, 通常需对方程组 (4.1) 作预处理, 使其系数矩阵的谱相对集中, 这就是所谓的预处理共轭梯度法 (PCG 方法). 简单来说, 预处理共轭梯度法就是预先选择一个适当的对称正定矩阵 $\boldsymbol{M}$, 使得矩阵 $\boldsymbol{M}^{-1}\boldsymbol{A}$ 的谱相对集中. 设 $\boldsymbol{M}$ 有分解 $\boldsymbol{M}=\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$, 则方程组 (4.1) 经过预处理后变为

$$(\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\widetilde{\boldsymbol{x}}=\boldsymbol{L}^{-1}\boldsymbol{b},\quad \boldsymbol{x}=\boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{x}}. \tag{4.46}$$

该方程组的系数矩阵 $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}$ 仍为对称正定矩阵. 通过 $\boldsymbol{M}$ 和 $\boldsymbol{L}$ 的选取, 可以使得 $\widetilde{\boldsymbol{A}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}$ 比 $\boldsymbol{A}$ 有更好的条件数. 所以预处理后的方程组 (4.46) 比原方程组更容易计算. 现在对方程组 (4.46) 使用共轭梯度法:

取 $\widetilde{\boldsymbol{x}}_0$, 令 $\widetilde{\boldsymbol{r}}_0 = \boldsymbol{L}^{-1}\boldsymbol{b} - (\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\widetilde{\boldsymbol{x}}_0$, $\widetilde{\boldsymbol{p}}_1 = \widetilde{\boldsymbol{r}}_0$, 对 $k = 1, 2, \cdots$, 计算出

$$\widetilde{\alpha}_k = \frac{\widetilde{\boldsymbol{r}}_{k-1}^{\mathrm{T}}\widetilde{\boldsymbol{r}}_{k-1}}{\widetilde{\boldsymbol{p}}_k^{\mathrm{T}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k}, \quad \widetilde{\boldsymbol{x}}_k = \widetilde{\boldsymbol{x}}_{k-1} + \widetilde{\alpha}_k\widetilde{\boldsymbol{p}}_k, \quad \widetilde{\boldsymbol{r}}_k = \widetilde{\boldsymbol{r}}_{k-1} - \widetilde{\alpha}_k\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k,$$

$$\widetilde{\beta}_k = \frac{\widetilde{\boldsymbol{r}}_k^{\mathrm{T}}\widetilde{\boldsymbol{r}}_k}{\widetilde{\boldsymbol{r}}_{k-1}^{\mathrm{T}}\widetilde{\boldsymbol{r}}_{k-1}}, \quad \widetilde{\boldsymbol{p}}_{k+1} = \widetilde{\boldsymbol{r}}_k + \widetilde{\beta}_k\widetilde{\boldsymbol{p}}_k.$$

为了在公式中使用原来变量, 定义 $\boldsymbol{x}_k = \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{x}}_k$, 则 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k = \boldsymbol{L}\widetilde{\boldsymbol{r}}_k$. 再令 $\boldsymbol{p}_k = \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{p}}_k$, $\boldsymbol{z}_k = \boldsymbol{M}^{-1}\boldsymbol{r}_k$, 就得到了预处理的共轭梯度法, 具体步骤如下:

Back

Close

步 1, 给定 $\boldsymbol{x}_0 \in \mathbb{R}^n$. 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\boldsymbol{z}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$, $\boldsymbol{p}_1 = \boldsymbol{z}_0$.

步 2, 对 $k = 1, 2, \cdots$, 计算

$$\alpha_k = \frac{\boldsymbol{z}_{k-1}^{\mathrm{T}}\boldsymbol{r}_{k-1}}{\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k}, \ \boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k\boldsymbol{p}_k,$$

$$\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k\boldsymbol{A}\boldsymbol{p}_k, \ \boldsymbol{z}_k = \boldsymbol{M}^{-1}\boldsymbol{r}_k, \tag{4.47}$$

$$\beta_k = \frac{\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{r}_k}{\boldsymbol{z}_{k-1}^{\mathrm{T}}\boldsymbol{r}_{k-1}}, \ \boldsymbol{p}_{k+1} = \boldsymbol{z}_k + \beta_k\boldsymbol{p}_k. \tag{4.48}$$

为了优化算法程序, 写成如下形式.

算法 4.2 (PCG 方法) 给定对称正定方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, 预处理子 $\boldsymbol{M}$ 和 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{z}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$;

$\boldsymbol{p}_1 = \boldsymbol{z}_0;\ \rho_0 = \boldsymbol{z}_0^{\mathrm{T}}\boldsymbol{r}_0;\ k = 0;$

while $(\|\boldsymbol{r}_k\|_2/\|r_0\|_2 > \varepsilon)$

$k = k + 1;$

$\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{p}_k;\ \alpha_k = \rho_{k-1}/\boldsymbol{u}_k^{\mathrm{T}}\boldsymbol{p}_k;$

$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k\boldsymbol{p}_k;\ \boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k\boldsymbol{u}_k;$

$\boldsymbol{z}_k = \boldsymbol{M}^{-1}\boldsymbol{r}_k;\ \rho_k = \boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{r}_k;\ \beta_k = \rho_k/\rho_{k-1};$

$\boldsymbol{p}_{k+1} = \boldsymbol{z}_k + \beta_k\boldsymbol{p}_k;$

end

算法 4.2 仅与预处理矩阵 $\boldsymbol{M}$ 有关, 而与 $\boldsymbol{L}$ 无关. 当 $\boldsymbol{M}$ 为单位阵时, 它就是没有经过预处理的共轭梯度法. 与共轭梯度法相比较, 预处理共轭梯度法仅增加了式 (4.47) 的工作量, 即每一步需要求一个以预处理矩阵 $\boldsymbol{M}$ 为系数矩阵的方程组得到 $\boldsymbol{z}_k$. 算法 4.2

每一步的主要工作量是作一次矩阵 $\boldsymbol{A}$ 与向量的乘法操作和一次 $\boldsymbol{M}^{-1}$ 与向量的乘法操作.

下面讨论预处理矩阵 $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ 的选取. $\boldsymbol{M}$ 取法之一是 $\boldsymbol{A}$ 的不完全 Cholesky 分解, 即 $\boldsymbol{A} = \widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^{\mathrm{T}} + \boldsymbol{R}$, 然后取 $\boldsymbol{M} := \boldsymbol{L}\widehat{\boldsymbol{L}}^{\mathrm{T}}$, 其中 $\boldsymbol{R} = \boldsymbol{A} - \boldsymbol{M}$ 满足某种稀疏模式, 使得它比较小.

如果对 $\boldsymbol{A}$ 进行分裂, 即 $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$, 则需要判断古典迭代格式中的 $\boldsymbol{M}$ 是否满足要求, 即是否是对称正定阵. 对于 Jacobi 迭代, $\boldsymbol{M}$ 为 $\boldsymbol{A}$ 的对角元所组成的对角矩阵, 它显然是对称正定的, 因此可以作为预处理矩阵. 这样的预处理称为 Jacobi 预处理. 对于 Gauss–Seidel 和 SOR 迭代, 其分裂格式中的 $\boldsymbol{M}$ 都是下三角矩阵, 不是对称正定阵, 因此不能作为预处理矩阵. 而对于 SSOR 迭代, 其分裂格式中的 $\boldsymbol{M}$ 是对称正定的, 因此可以作为预处理矩阵, 称其为 SSOR 预处理矩阵.

Back

Close

给出预处理共轭梯度法的 MATLAB 程序如下:

```
%预处理共轭梯度法程序-pcg.m
function [x,iter,time,res,resvec]=pcg(A,b,x,M,max_it,tol)
%输入:系数矩阵A,右端向量b,初始向量x,预处理子M,
%      容许误差tol,最大迭代次数max_it
%输出:解向量x,迭代次数iter,CPU时间time,
%      终止时相对残差模res,相对残差模向量resvec
tic; r=b-A*x; z=M\r; p=z;
rho=z'*r; mr=norm(r); iter=0;
while (iter<max_it)
    iter=iter+1;
    u=A*p; alpha=rho/(p'*u);
    x=x+alpha*p; r=r-alpha*u;
```

```
        z=M\r; rho1=z'*r;
        beta=rho1/rho; p=z+beta*p;
        res=norm(r)/mr;
        resvec(iter)=res;
        if (res<tol), break; end
        rho=rho1;
    end
    time=toc;
```

为了对预处理的效果有一个更直观更深刻的印象, 下面再给出一个数值算例.

例 4.2 考虑例 4.1 中的线性方程组, 如果取预处理矩阵 $\boldsymbol{M}$ 为 $\boldsymbol{A}$ 的对角元构成的对角矩阵, 即 $\boldsymbol{M} = \mathrm{diag}(1, 2, \cdots, n)$. 实验中取 $n = 1000$, 则预处理共轭梯度法在 12 步后得到的近似解

$\widehat{\boldsymbol{x}} = \boldsymbol{x}_{12}$ 就满足 $\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 3.7305 \times 10^{-9}$. 迭代的收敛轨迹如图 4.2 所示, 其中横坐标表示迭代步数 k, 纵坐标表示相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差.

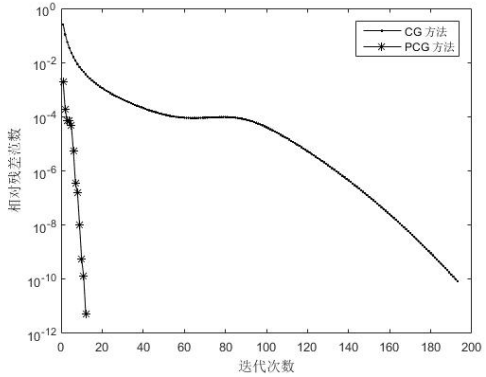


图 4.2 PCG 法和 CG 法的收敛特性

与例 4.1 的数值结果相比较可知, 虽然这里的预处理矩阵 $\boldsymbol{M}$ 选得非常简单, 但其加速效果是显著的. 为什么会这样呢? 因为预处理后矩阵 $\boldsymbol{M}^{-1}\boldsymbol{A}$ 的特征值分布很集中, 故其数值效果非常好.

§4.1.4 CGNR 方法和 CGNE 方法

1. CGNR 方法

任何对称不定或非对称非奇异的方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 都可转化为一个对称正定的方程组 (即法方程)

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}. \tag{4.49}$$

对方程组 (4.49) 应用 CG 法, 就导出了下面的关于法方程残差的共轭梯度法 (CGNR).

算法 4.3 (CGNR 方法) 给定系数矩阵 $\boldsymbol{A}$, 右端向量 $\boldsymbol{b}$, 初始向量 $\boldsymbol{x}_0$ 和容许误差限 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

$$\alpha_k = \frac{(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_k, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_k)}{(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k \boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k)};$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k; \quad \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k;$$

$$\beta_k = \frac{(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_{k+1}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_{k+1})}{(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_k, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_k)};$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k;$$

$$k = k + 1;$$

end

算法 4.4 每次迭代需要计算三次矩阵与向量的乘法: $\boldsymbol{A}\boldsymbol{p}_k$, $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k$ 和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_{k+1}$. 注意到, CGNR 方法是极小化误差

$$\|\boldsymbol{e}_k\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}} = \|\boldsymbol{x}_k - \boldsymbol{x}^*\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}},$$

式中: $\boldsymbol{x}^*$ 为精确解, 满足 $\boldsymbol{x}^* = \boldsymbol{A}^{-1}\boldsymbol{b}$; $\|\boldsymbol{e}_k\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}}$ 为仿射空间

$$\begin{aligned}&\boldsymbol{x}_0 + \mathrm{span}\left\{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, \cdots, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{k-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0\right\}\\ &:= \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0)\end{aligned}$$

上关于残量 $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 的 2–范数. 因此, 可以比照 CG 方法的收敛性分析技巧对 CGNR 方法进行相应的收敛性分析.

2. CGNE 方法

对称不定或非对称非奇异的方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 还可转化为如下形式的对称正定的方程组, 即

$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{b},\ \boldsymbol{x} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}. \tag{4.50}$$

在 "$\boldsymbol{y}$" 空间中, 对方程组 (4.50) 应用 CG 法, 得到如下形式:

选取初值 $\boldsymbol{y}_0$, 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}_0$; $\boldsymbol{p}_0 = \boldsymbol{r}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

$$\alpha_k = \frac{(\boldsymbol{r}_k, \boldsymbol{r}_k)}{(\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k)} = \frac{(\boldsymbol{r}_k, \boldsymbol{r}_k)}{(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k)};$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \alpha_k \boldsymbol{p}_k; \quad \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k;$$

$$\beta_k = \frac{(\boldsymbol{r}_{k+1}, \boldsymbol{r}_{k+1})}{(\boldsymbol{r}_k, \boldsymbol{r}_k)}; \quad \boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k;$$

$k = k + 1;$

end

作变量替换 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}_k \to \boldsymbol{x}_k$, $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k \to \boldsymbol{p}_k$, 再化简上述有关式子则得到下面的关于法方程误差的共轭梯度法 (CGNE).

算法 4.4 (CGNE 方法) 给定初始向量 $\boldsymbol{x}_0$ 和容许误差限 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

$$\alpha_k = \frac{(\boldsymbol{r}_k, \boldsymbol{r}_k)}{(\boldsymbol{p}_k, \boldsymbol{p}_k)}; \quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k;$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k; \quad \beta_k = \frac{(\boldsymbol{r}_{k+1}, \boldsymbol{r}_{k+1})}{(\boldsymbol{r}_k, \boldsymbol{r}_k)};$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{A}^{\mathrm{T}} \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k;$$

$$k = k + 1;$$

end

算法 4.4 每次迭代需要计算两次矩阵与向量的乘法: $\boldsymbol{A}\boldsymbol{p}_k$ 和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_{k+1}$. 此外, CGNE 方法是极小化误差

$$\|\widetilde{\boldsymbol{e}}_k\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}} = \|\boldsymbol{y}_k - \boldsymbol{y}^*\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}},$$

式中: $\boldsymbol{y}^*$ 满足 $\boldsymbol{y}^* = (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{b}$; $\|\widetilde{\boldsymbol{e}}_k\|_{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}}$ 为仿射空间

$$\begin{aligned}\boldsymbol{x}_k &\in \boldsymbol{x}_0 + \mathrm{span}\{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, \cdots, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{k-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0\} \\ &\equiv \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0)\end{aligned}$$

上的 2–范数 $\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_2$.

一般来说, CGNR 和 CGNE 这两种方法的困难或者说缺点在于矩阵的条件数变成了原来的平方, 然而, 在有些情况下它们是很有效的. 可参看文献 [25] 中的论述.

例 4.3 考虑方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$, 其中

$$\boldsymbol{A}=\begin{bmatrix} 12 & 3 & 2 & & & \\ -3 & 12 & 3 & 2 & & \\ -2 & -3 & 12 & 3 & 2 & \\ & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & -2 & -3 & 12 & 3 \\ & & & -2 & -3 & 12 \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad \boldsymbol{b}=\boldsymbol{A}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \\ 1 \end{bmatrix} \in \mathbb{R}^{n}.$$

取 $n=1000$, 分别将 CGNR 和 CGNE 方法应用到该线性方程组上, 两个迭代法均在 10 步后收敛 ($\varepsilon=10^{-10}$), 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和 $\widetilde{\boldsymbol{x}}$ 与真解 $\boldsymbol{x}^*$ 之间的绝对值误差分别为

$$\|\widehat{\boldsymbol{x}}-\boldsymbol{x}^*\|_2=3.4705\times10^{-10}, \quad \|\widetilde{\boldsymbol{x}}-\boldsymbol{x}^*\|_2=3.4515\times10^{-10}.$$

残量分别为

$$\|\boldsymbol{b}-\boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 4.5764\times 10^{-9}, \quad \|\boldsymbol{b}-\boldsymbol{A}\widetilde{\boldsymbol{x}}\|_2 = 4.6018\times 10^{-9}.$$

迭代过程的收敛轨迹如图 4.3 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

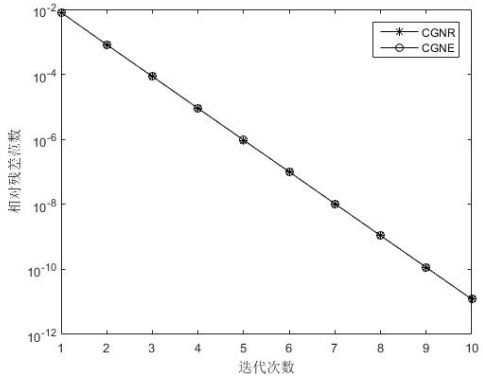


图 4.3 CGNR 和 CGNE 方法的收敛特性

§4.2 广义极小残量法

广义极小残量法是目前求解大型稀疏非对称线性方程组的最常用方法之一, 在文献中常称为 GMRES 方法 (Generalized Minimal RESidual Method). 本节主要介绍这一算法的详细计算步骤及其收敛性理论.

§4.2.1 GMRES 方法

考虑如下的线性方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4.51}$$

式中: 矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和向量 $\boldsymbol{b} \in \mathbb{R}^n$ 是给定的, 而 $\boldsymbol{x} \in \mathbb{R}^n$ 是待求的未知向量. 这里假定系数矩阵 $\boldsymbol{A}$ 是非奇异的大型稀疏矩阵, 而




Back
Close

且 $\boldsymbol{A}^{\mathrm{T}} \neq \boldsymbol{A}$. 广义极小残量法是求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得

$$\|\boldsymbol{r}_k\|_2 = \min\{\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\}, \tag{4.52}$$

式中: $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$, 即求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得残量 $\boldsymbol{r}_k$ 的 2–范数达到最小.

由于现在 $\boldsymbol{A}$ 是非对称的, 所以只能借助 $\boldsymbol{A}$ 的 Arnoldi 正交化过程极小化问题 (4.52). 回顾一下 Arnoldi 正交化过程 (见算法 2.11):

选取初始向量 $\boldsymbol{v}_1$, 使得 $\|\boldsymbol{v}_1\|_2 = 1$;

for $j = 1 : k$

for $i = 1 : j$

$h_{ij} = (\boldsymbol{A}\boldsymbol{v}_j, \boldsymbol{v}_i)$;

$$\widetilde{\boldsymbol{v}}_{j+1} = \boldsymbol{A}\boldsymbol{v}_j - \sum_{i=1}^{j} h_{ij}\boldsymbol{v}_i; \tag{4.53}$$

$$h_{j+1,j} = \|\widetilde{\boldsymbol{v}}_{j+1}\|_2; \;\; \boldsymbol{v}_{j+1} = \widetilde{\boldsymbol{v}}_{j+1}/h_{j+1,j};$$

end

end

不难发现, 式 (4.53) 可以写成

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k\boldsymbol{H}_k + \beta_k\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} = \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{H}}_{k+1,k}, \tag{4.54}$$

式中: $\boldsymbol{V}_{k+1} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] \in \mathbb{R}^{n\times(k+1)}$ 满足 $\boldsymbol{V}_{k+1}^{\mathrm{T}}\boldsymbol{V}_{k+1} = \boldsymbol{I}_{k+1}$; 矩阵

$$\widetilde{\boldsymbol{H}}_{k+1,k} = \begin{bmatrix} \boldsymbol{H}_k \\ \beta_k\boldsymbol{e}_k^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$$

为上 Hessenberg 矩阵, 其中 $\beta_k = h_{k+1,k}$,

$$\boldsymbol{H}_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & h_{2k} \\ & h_{32} & \cdots & h_{3k} \\ & \ddots & & \vdots \\ & & h_{k,k-1} & h_{k,k} \end{bmatrix}.$$

注意到 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathcal{R}(\boldsymbol{V}_k)$, 而且对任意的 $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 有

$$\begin{aligned} \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 &= \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{z}\|_2 = \|\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{z}\|_2 \\ &= \|\beta\boldsymbol{V}_{k+1}\boldsymbol{e}_1^{(k+1)} - \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{H}}_{k+1,k}\boldsymbol{z}\|_2 \\ &= \|\beta\boldsymbol{e}_1^{(k+1)} - \widetilde{\boldsymbol{H}}_{k+1,k}\boldsymbol{z}\|_2, \end{aligned}$$

式中: $\beta = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{e}_1^{(k+1)}$ 表示第 1 个分量为 1、其余分量均为 0 的




$k+1$ 维列向量. 由此可知, 极小化问题 (4.52) 等价于求 $\boldsymbol{z}_k \in \mathbb{R}^k$, 使得

$$\|\beta \boldsymbol{e}_1^{(k+1)} - \widetilde{\boldsymbol{H}}_{k+1,k}\boldsymbol{z}_k\|_2 = \min\{\|\beta \boldsymbol{e}_1^{(k+1)} - \widetilde{\boldsymbol{H}}_{k+1,k}\boldsymbol{z}\|_2 : \boldsymbol{z} \in \mathbb{R}^k\}. \tag{4.55}$$

一旦这样的 $\boldsymbol{z}_k$ 已经求得, 则所需的 $\boldsymbol{x}_k$ 就为 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$.

最小二乘问题 (4.55) 也可以用 $\widetilde{\boldsymbol{H}}_{k+1,k}$的 QR 分解来求解. 由于 $\widetilde{\boldsymbol{H}}_{k+1,k}$ 是上 Hessenberg 矩阵, 可以计算 k 个 Givens 变换

$$\boldsymbol{G}_i = \mathrm{diag}\left(\boldsymbol{I}_{i-1}, \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix}, \boldsymbol{I}_{k-i}\right), \quad c_i^2 + s_i^2 = 1,$$

使得

$$(\boldsymbol{G}_k\boldsymbol{G}_{k-1}\cdots\boldsymbol{G}_2\boldsymbol{G}_1)\widetilde{\boldsymbol{H}}_{k+1,k} = \begin{bmatrix} \boldsymbol{R}_k \\ \boldsymbol{0} \end{bmatrix}, \tag{4.56}$$

Back

Close

式中: $\boldsymbol{R}_k$ 为非奇异的上三角矩阵 (因为 $\widetilde{\boldsymbol{H}}_{k+1,k}$ 的次对角元均不为零). 令

$$\boldsymbol{G} = \boldsymbol{G}_k \boldsymbol{G}_{k-1} \cdots \boldsymbol{G}_2 \boldsymbol{G}_1, \quad \begin{bmatrix} \boldsymbol{t}_k \\ \rho_k \end{bmatrix} = \boldsymbol{G}(\beta \boldsymbol{e}_1), \quad \boldsymbol{t}_k = (\tau_1, \tau_2, \cdots, \tau_k)^{\mathrm{T}}, \tag{4.57}$$

则 $\boldsymbol{G}$ 是 $k+1$ 阶正交矩阵, 而且直接计算, 有

$$\begin{cases} \tau_1 = \beta c_1, \\ \tau_i = (-1)^{i-1} \beta s_1 s_2 \cdots s_{i-1} c_i, \quad i = 2, 3, \cdots, k, \\ \rho_k = (-1)^k \beta s_1 s_2 \cdots s_k. \end{cases} \tag{4.58}$$

由此立即可得最小二乘问题 (4.55) 的解为

$$\boldsymbol{z}_k = \boldsymbol{R}_k^{-1} \boldsymbol{t}_k. \tag{4.59}$$

此外, 此时的残量范数为

$$\|\boldsymbol{r}_k\|_2 = \|\beta \boldsymbol{e}_1^{(k+1)} - \widetilde{\boldsymbol{H}}_{k+1,k}\boldsymbol{z}_k\|_2 = |\rho_k|. \tag{4.60}$$

综合上面的讨论, 可以将广义极小残量法 (GMRES) 的步骤总结如下:

(1) 令 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$, 产生一个长度为 k 的 Arnoldi 分解 (4.54).

(2) 利用 Givens 变换求 $\widetilde{\boldsymbol{H}}_{k+1,k}$ 的 QR 分解 (4.56), 并按式 (4.58) 求得向量 $\boldsymbol{t}_k$ 和数 ρ_k.

(3) 用回代法求解上三角方程组 $\boldsymbol{R}_k\boldsymbol{z}_k = \boldsymbol{t}_k$, 得 $\boldsymbol{z}_k$.

(4) 计算 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$.

(5) 若 $|\rho_k|/\beta < \varepsilon$ (事先给定的误差界), 则终止; 否则增加 k 的值, 重复上面的过程.

上述 GMRES 方法可写成如下便于编程的算法形式.

算法 4.5 (GMRES 方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 向量 $\boldsymbol{b} \in \mathbb{R}^n$ 和允许误差 $\varepsilon > 0$, 取初始向量 $\boldsymbol{x}_0$. 本算法计算 $\boldsymbol{x}_k \in \mathbb{R}^n$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

步 1, 计算初始残量 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\beta = \|\boldsymbol{r}_0\|_2$ 和初始正交化向量 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta$, $\boldsymbol{\xi} = \boldsymbol{e}_1 = (1, 0, \cdots, 0)^{\mathrm{T}}$. 置 $k := 1$.

步 2, 用 Arnoldi 过程计算 $\boldsymbol{v}_{k+1}$ 和 $h_{i,k}\,(i = 1, 2, \cdots, k+1)$. 将 Givens 变换 $\boldsymbol{G}_i$ 作用于矩阵 $\widetilde{\boldsymbol{H}}_{i+1,i}$ 的最后一列:

$$\begin{bmatrix} h_{i,k} \\ h_{i+1,k} \end{bmatrix} := \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix} \begin{bmatrix} h_{i,k} \\ h_{i+1,k} \end{bmatrix}, \quad i = 1, \cdots, k-1.$$

步 3，计算第 k 次 Givens 变换 $\boldsymbol{G}_k$ 中的 c_k 和 s_k：

$$c_k = \frac{h_{k,k}}{\sqrt{h_{k,k}^2 + h_{k+1,k}^2}}, \quad s_k = \frac{h_{k+1,k}}{\sqrt{h_{k,k}^2 + h_{k+1,k}^2}}, \tag{4.61}$$

$$\left(\tau_k = \frac{h_{k+1,k}}{h_{k,k}}, \quad c_k = \frac{1}{\sqrt{1+\tau_k^2}}, \quad s_k = c_k \tau_k \right).$$

步 4，对 $k+1$ 维向量 $\boldsymbol{\xi}$ 的最后两个元素和矩阵 $\widetilde{\boldsymbol{H}}_{k+1,k}$ 分别作用第 k 次 Givens 变换 $\boldsymbol{G}_k$，有

$$\begin{bmatrix} \xi_k \\ \xi_{k+1} \end{bmatrix} := \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \xi_k \\ 0 \end{bmatrix} = \begin{bmatrix} c_k \xi_k \\ -s_k \xi_k \end{bmatrix},$$

$$\begin{bmatrix} h_{k,k} \\ h_{k+1,k} \end{bmatrix} := \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} h_{k,k} \\ h_{k+1,k} \end{bmatrix} = \begin{bmatrix} c_k h_{k,k} + s_k h_{k+1,k} \\ 0 \end{bmatrix}. \tag{4.62}$$

步 5, 若 $|\rho_k|/\beta = |\xi_{k+1}|/\beta \leqslant \varepsilon$, 求解关于 $\boldsymbol{z}_k$ 的上三角方程组 $\boldsymbol{H}_{k,k}\boldsymbol{z}_k = [\beta\boldsymbol{\xi}]_{k\times 1}$, 计算近似解向量 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$, 停算; 否则, 置 $k := k+1$, 转步 2.

算法 4.5 的 MATLAB 程序如下:

```
%GMRES方法程序-mgmres.m
function [x,k,time,res,resvec,flag]=mgmres(A,b,x,max_it,tol)
tic; flag=0; r=b-A*x; beta=norm(r);  %计算残差
n=length(b); e1=zeros(n,1); e1(1)=1.0;
res=norm(r)/beta; resvec(1)=res;
V(:,1)=r/beta; xi=beta*e1; k=0;
while (k<=max_it)
    k=k+1; w=A*V(:,k);
    for i=1:k  %修正Arnoldi过程
```




Back
Close

```
    H(i,k)=w'*V(:,i);
    w=w-H(i,k)*V(:,i);
end
H(k+1,k)=norm(w);
if abs(H(k+1,k))/beta<tol,
    return;
else
    V(:,k+1)=w/H(k+1,k);
end
for i=1:k-1,
    temp=c(i)*H(i,k)+s(i)*H(i+1,k);
    H(i+1,k)=-s(i)*H(i,k)+c(i)*H(i+1,k);
    H(i,k)=temp;
```




```
    end
    [c(k),s(k),H(k,k)]=givens(H(k,k), H(k+1,k));%第k次Givens
    xi(k+1)=-s(k)*xi(k);
    xi(k)=c(k)*xi(k); H(k+1,k)=0.0;
    res=abs(xi(k+1))/beta;
    resvec(k+1)=res;
    if (res<=tol ),
        y=H(1:k,1:k)\xi(1:k); x=x+V(:,1:k)*y;
        break;  %跳出循环
    end
end
if (res>tol), flag=1; end;  %不收敛
time=toc;
```

例 4.4 考虑系数矩阵 $\boldsymbol{A}$ 和右端项 $\boldsymbol{b}$ 分别为

$$\boldsymbol{A}=\begin{bmatrix} 1 & -1 & 0 & \cdots & 0 & n \\ 1 & 2 & -1 & \ddots & & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & 1 & n-1 & -1 \\ -n & 0 & \cdots & 0 & 1 & n \end{bmatrix}, \quad \boldsymbol{b}=\boldsymbol{A}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \\ 1 \end{bmatrix}$$

的线性方程组 (4.51). 显然, 该方程组的真解为 $\boldsymbol{x}^*=(1,1,\cdots,1)^{\mathrm{T}}$. 取 $n=10^3$, 应用 GMRES 算法到该方程组上, 迭代 172 步后得到的近似解 $\widetilde{\boldsymbol{x}}=\boldsymbol{x}_{172}$ 满足

$$\|\widetilde{\boldsymbol{x}}-\boldsymbol{x}_*\|_2=1.1427\times 10^{-7}.$$

迭代过程的收敛轨迹如图 4.4 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

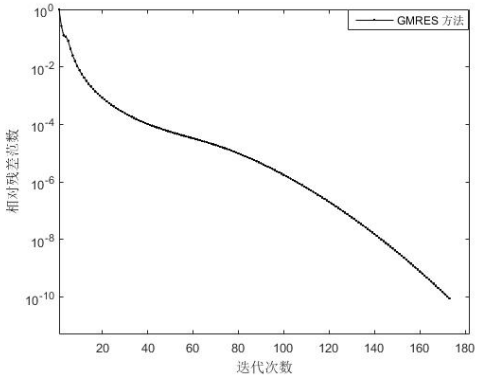


图 4.4 GMRES 方法的收敛特性

实际使用 GMRES 方法时的主要问题是 k 不能太大, 这是因为这一方法的内存需求量为 $O(kn)$, 而计算机的内存又是有限的, 这样就会出现近似解 $\boldsymbol{x}_k$ 还不满足精度要求, 而 k 已经不能再增加的情形. 解决这一问题的一个简单而行之有效的办法就是重开始技术. 它的基本思想是, 先选定一个不太大的正整数 m, 用 GMRES 方法产生 $\boldsymbol{x}_m$, 然后再以 $\boldsymbol{x}_m$ 作为初始向量重新开始. 这就是重开

始 GMRES 方法, 通常记为 GMRES(m) 方法, 具体计算过程可简述如下.

算法 4.6 (GMRES(m) 方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 向量 $\boldsymbol{b} \in \mathbb{R}^n$, 正整数 m, 初始向量 $\boldsymbol{x}_0$ 和允许误差 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_m \in \mathbb{R}^n$, 使得 $\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m\|_2/\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0\|_2 \leqslant \varepsilon$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\beta = \|\boldsymbol{r}_0\|_2$; $\rho_m := \beta$; $\boldsymbol{x}_m := \boldsymbol{x}_0$;

while $(\rho_m/\beta > \varepsilon)$

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m$; $\beta_0 = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta_0$;

以 $\boldsymbol{v}_1$ 为初始向量产生一个长度为 m 的 Arnoldi 分解

$$\boldsymbol{A}\boldsymbol{V}_m = \boldsymbol{V}_{m+1}\widetilde{\boldsymbol{H}}_{m+1,m};$$

计算 $\widetilde{\boldsymbol{H}}_{m+1,m}$ 的 QR 分解: $\widetilde{\boldsymbol{H}}_{m+1,m} = \boldsymbol{G}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{R}_m\\ \boldsymbol{0}\end{bmatrix}$;

按式 (4.58) 计算 $\boldsymbol{t}_m$ 和 ρ_m;

求解 $\boldsymbol{R}_m\boldsymbol{z}_m = \boldsymbol{t}_m$ 得到 $\boldsymbol{z}_m$;

计算 $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{V}_m\boldsymbol{z}_m$;

end

这一算法是目前求解大型稀疏非对称线性方程组的常用方法之一. 至于算法 4.6 中的 m 取多大为好, 现在还没有理论上的结果. 在理论上仅可以保证, 在系数矩阵 $\boldsymbol{A}$ 有正定的对称部分时, 对任意的 m, 算法 4.6 总是收敛的 (见后面的收敛性定理). 但对于一般情形, 并非对任意的 m 总能保证其收敛. 例如, 对线性方程组

$$\boldsymbol{A}\boldsymbol{x} := \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} := \boldsymbol{b},$$

利用 GMRES(1) 求解, 不论循环多少次, 总有 $\boldsymbol{x}_k = \mathbf{0}$, 而 $\|\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}\|_2 = \sqrt{2}$, 永远不会得到原方程组之真解满足要求的近似解. 但如果用 GMRES 求解, 则只需两步就可以得到该方程组的精确解.

GMRES(m) 的 MATLAB 程序如下:

```
%重开始GMRES方法-GMRES(m)-gmresm.m
function [x,out,int,time,res,resvec,flag]
            =gmresm(A,b,x,m,max_it,tol)
tic; flag=0; int=0; r= b-A*x;  %计算残差
beta=norm(r); res=norm(r)/beta; resvec(1)=res;
n=length(b); %m=restrt;
V(1:n,1:m+1)=zeros(n,m+1);
H(1:m+1,1:m)=zeros(m+1,m);
e1=zeros(n,1); e1(1)=1.0;
```

```
c(1:m)=zeros(m,1); s(1:m)=zeros(m,1);
for k=1:max_it
    V(:,1)=r/norm(r);
    xi=norm(r)*e1;
    for j=1:m  %用Arnoldi方法构造正交基
        w=A*V(:,j);
        for i=1:j
            H(i,j)=w'*V(:,i);
            w=w-H(i,j)*V(:,i);
        end
        H(j+1,j)=norm(w);
        if abs(H(j+1,j))/beta<tol,
            return;
```

```
else
    V(:,j+1)=w/H(j+1,j);
end
for i=1:j-1  %第i次Givens变换
    temp=c(i)*H(i,j)+s(i)*H(i+1,j);
    H(i+1,j)=-s(i)*H(i,j)+c(i)*H(i+1,j);
    H(i,j)=temp;
end
[c(j),s(j),H(j,j)]=givens(H(j,j),H(j+1,j));
%第j次Givens变换
xi(j+1)=-s(j)*xi(j);xi(j)=c(j)*xi(j);
H(j+1,j)=0.0;res=abs(xi(j+1))/beta;
resvec((k-1)*m+j+1)=res;
```

```
        if(res<=tol)
            y=H(1:j,1:j)\xi(1:j);
            x=x+V(:,1:j)*y;
            break;  %跳出内循环
        end
    end
    if (res<tol )
        out=k; int=j; break; %跳出外循环
    end
    y=H(1:m,1:m)\xi(1:m);
    x=x+V(:,1:m)*y;
    r=b-A*x ;
end
```

```
if (res>tol), flag=1; end;  %不收敛
time=toc;
```

例 4.5 用 GMRES(m) 方法求解例 4.4 的线性方程组. 表 4.1 列出了对于不同的 m, 达到收敛 (容许误差为 $\varepsilon = 10^{-10}$) 所需要的外迭代次数、内迭代次数、总迭代次数和 CPU 时间 (s). 从表 4.1

表 4.1 GMRES(m) 方法对 m 的依赖性

m	外迭代次数	内迭代次数	总迭代次数	CPU 时间	相对残差
10	47	3	463	0.2621	9.8273e-11
20	14	12	272	0.1488	9.1166e-11
30	9	8	248	0.1327	9.3534e-11
40	6	27	227	0.1267	9.4923e-11
50	5	19	219	0.1244	9.9472e-11
60	4	26	206	0.1203	9.9062e-11
GMRES	1	172	172	0.1651	8.8473e-11

中的数据可看出, 对于本例而言, 与 GMRES 方法相比, 对于比较




Back
Close

小的 m 值, GMRES(m) 方法并不占优势. 事实上, 前面已经指出, 只有在无法使用 GMRES 方法时 (如超出计算机的内存), 才考虑使用 GMRES(m) 方法, 并且 m 不宜取得太小.

此外, 对于不同的 m 值, 迭代过程的收敛轨迹如图 4.5 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

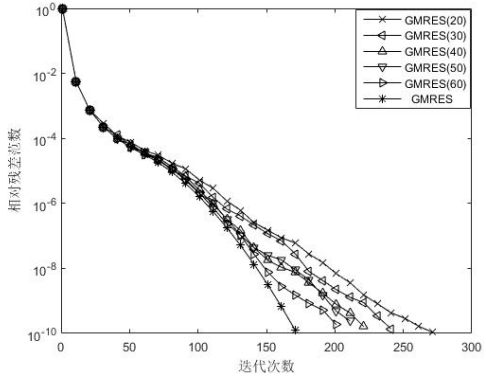


图 4.5 GMRES 方法的收敛特性

§4.2.2 预处理 GMRES 方法

回顾共轭梯度法求解方程组 $\boldsymbol{Ax} = \boldsymbol{b}$ 时预处理技术的重要性, 考虑预处理广义极小残量法 (简记为 PGMRES 方法). 因为共轭梯度法适用于对称正定矩阵 $\boldsymbol{A}$, 所以预处理矩阵 $\boldsymbol{M}$ 需要保持对称. 对于非对称矩阵 $\boldsymbol{A}$, 预处理矩阵 $\boldsymbol{M}$ 就没有必要是对称的. 此时方程组 $\boldsymbol{Ax} = \boldsymbol{b}$ 被预处理为

$$\boldsymbol{M}^{-1}\boldsymbol{Ax} = \boldsymbol{M}^{-1}\boldsymbol{b}. \tag{4.63}$$

这里 $\boldsymbol{M}$ 应该取为 $\boldsymbol{A}$ 的一个近似矩阵, 以保证用 GMRES 方法求解预处理方程组 (4.63) 比求解原方程组 $\boldsymbol{Ax} = \boldsymbol{b}$ 有更快的收敛速度.

由于 GMRES 方法只涉及到矩阵与向量的乘法, 所以要保证 $\boldsymbol{M}^{-1}$ 与某一个向量 $\boldsymbol{r}$ 的乘法 $\boldsymbol{z} = \boldsymbol{M}^{-1}\boldsymbol{r}$ 容易计算, 即 $\boldsymbol{Mz} = \boldsymbol{r}$




容易求解. 这启发选取 $\boldsymbol{M}$ 为 (块) 对角矩阵或 (块) 三角形矩阵, 如 Jacobi 预处理矩阵

$$\boldsymbol{M} = \boldsymbol{D} = \mathrm{diag}\{a_{11}, \cdots, a_{nn}\},$$

Gauss–Seidel 预处理矩阵

$$\boldsymbol{M} = \boldsymbol{D} - \boldsymbol{L}$$

以及 SOR 预处理矩阵

$$\boldsymbol{M} = \frac{1}{\omega}(\boldsymbol{D} - \omega\boldsymbol{L}),$$

式中:

$$\boldsymbol{L} = -\begin{bmatrix} 0 & & & \\ a_{21} & 0 & & \\ \vdots & \ddots & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}.$$




Back
Close

对方程组 (4.63) 用 GMRES 方法, 得到如下 PGMRES 方法.

算法 4.7 (PGMRES 方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 向量 $\boldsymbol{b} \in \mathbb{R}^n$, 预处理子 $\boldsymbol{M}$, 初始向量 $\boldsymbol{x}_0$ 和允许误差 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k \in \mathbb{R}^n$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

$\boldsymbol{r}_0 = \boldsymbol{M}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$; $\beta = \|\boldsymbol{r}_0\|_2$;

for $k = 1, 2, \cdots$

对 $\mathcal{K}_k(\boldsymbol{M}^{-1}\boldsymbol{A}, \boldsymbol{v}_1)$ 用 Arnoldi 分解计算 $\boldsymbol{V}_k = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k]$ 以及 $(k+1)\times k$ 阶矩阵 $\widetilde{\boldsymbol{H}}_{k+1,k}$.

计算 $\widetilde{\boldsymbol{H}}_{k+1,k}$ 的 QR 分解: $\widetilde{\boldsymbol{H}}_{k+1,k} = \boldsymbol{G}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{R}_k \\ \boldsymbol{0} \end{bmatrix}$;

按式 (4.58) 计算 $\boldsymbol{t}_k$ 和 ρ_k;

if $|\rho_k|/\beta \leqslant \varepsilon$

求解 $\boldsymbol{R}_k\boldsymbol{z}_k = \boldsymbol{t}_k$ 得到 $\boldsymbol{z}_k$;

计算 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$; 停算.

end

end

对方程组 (4.63) 用 GMRES(m) 得到如下算法:

算法 4.8 (PGMRES(m) 方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 向量 $\boldsymbol{b} \in \mathbb{R}^n$, 正整数 m, 预处理子 $\boldsymbol{M}$ 和允许误差 $\varepsilon > 0$, 选取初始向量 $\boldsymbol{x}_0 \in \mathbb{R}^n$. 本算法计算 $\boldsymbol{x}_k \in \mathbb{R}^n$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

步 1, 计算残量 $\boldsymbol{r}_0 = \boldsymbol{M}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$, $\beta = \|\boldsymbol{r}_0\|_2$ 和初始正交化向量 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta$.

步 2, 对 $\mathcal{K}_m(\boldsymbol{M}^{-1}\boldsymbol{A}, \boldsymbol{v}_1)$ 用 Arnoldi 分解计算

$$\boldsymbol{V}_m = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m]$$

以及 $(m+1)\times m$ 阶矩阵 $\widetilde{\boldsymbol{H}}_{m+1,m}$.

步 3, 计算满足极小化问题

$$\min\left\{\|\beta \boldsymbol{e}_1^{(m+1)} - \widetilde{\boldsymbol{H}}_{m+1,m}\boldsymbol{z}\|_2 : \boldsymbol{z}\in\mathbb{R}^m\right\}$$

的最小二乘解 $\boldsymbol{z}_m$, 令 $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{V}_m\boldsymbol{z}_m$. 若 $\boldsymbol{x}_m$ 达到精度要求, 停算.

步 4, 置 $\boldsymbol{x}_0 := \boldsymbol{x}_m$, 转步 1.

下面通过数值例子, 观察 PGMRES 方法的效果.

例 4.6 仍考虑例 4.4 的线性方程组. 取预处理矩阵 $\boldsymbol{M} = \mathrm{diag}(\boldsymbol{A})$, 应用 PGMRES 方法到该方程组上, 迭代 12 步后得到的

Back

Close

近似解 $\widetilde{\boldsymbol{x}} = \boldsymbol{x}_{12}$ 满足

$$\|\widetilde{\boldsymbol{x}} - \boldsymbol{x}_*\|_2 = 1.7407 \times 10^{-8}.$$

迭代过程的收敛轨迹如图 4.6 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

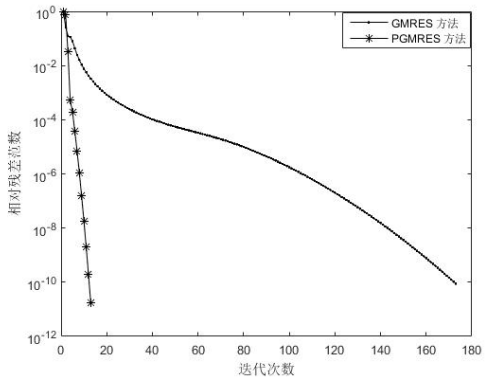


图 4.6 GMRES 和 PGMRES 的收敛特性

例 4.7 用 PGMRES(m) 方法求解例 4.4 的线性方程组, 取预处理矩阵 $\boldsymbol{M} = \mathrm{diag}(\boldsymbol{A})$. 表 4.2 列出了对于不同的 m, 达到收敛 (容许误差为 $\varepsilon = 10^{-10}$) 所需要的外迭代次数、内迭代次数、总迭代次数和 CPU 时间 (s).

表 4.2 PGMRES(m) 方法对 m 的依赖性

m	外迭代次数	内迭代次数	总迭代次数	CPU 时间	相对残差
3	21	3	63	0.1692	7.3458e-11
6	4	3	21	0.0449	3.8001e-11
9	2	5	14	0.0290	8.0750e-11
12	1	12	12	0.0238	1.7551e-11
PGMRES	1	12	12	0.0394	1.7551e-11

此外, 对于不同的 m 值, 迭代过程的收敛轨迹如图 4.7 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

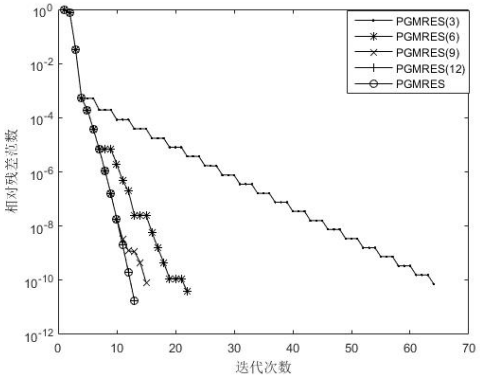


图 4.7 PGMRES(m) 方法对 m 的依赖性

§4.2.3 收敛性分析

前一节 CG 方法的收敛性都是关于对称矩阵的. 当系数矩阵非对称时, 对 Krylov 子空间方法 (比如 GMRES 方法) 的收敛性进行分析就会遇到很大的困难. 首先来看 GMRES 方法的一个重要性质.

定理 4.4 GMRES 方法不会发生中断.

证明 若 $h_{k+1,k} \neq 0$, 则计算过程直至第 k 步都不会中断. 事实上, 当 $h_{k+1,k} \neq 0$ 时, 由式 (4.61) 和式 (4.62), $\boldsymbol{R}_k$ 的对角元满足

$$r_{k,k} = h_{k,k} := c_k h_{k,k} + s_k h_{k+1,k} = \sqrt{h_{k,k}^2 + h_{k+1,k}^2} > 0.$$

故正交化可进行, 极小化问题可解.

由此可知, 只有当 $h_{k+1,k} = 0$ 时, 计算过程才在第 k 步中断, 此时向量 $\boldsymbol{v}_{k+1}$ 不能构造. 但此时成立

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k \boldsymbol{H}_k,$$

故 $\sigma(\boldsymbol{H}_k) \subset \sigma(\boldsymbol{A})$. 由于 $\boldsymbol{A}$ 非奇异, 故 $\boldsymbol{H}_k$ 也非奇异. 在第 k 步, 极

Back

Close

小化问题 (4.55) 变为

$$\|\beta \boldsymbol{v}_1 - \boldsymbol{A}\boldsymbol{V}_k \boldsymbol{z}\|_2 = \|\beta \boldsymbol{v}_1 - \boldsymbol{V}_k \boldsymbol{H}_k \boldsymbol{z}\|_2 = \left\|\boldsymbol{V}_k\left(\beta \boldsymbol{e}_1^{(k)} - \boldsymbol{H}_k \boldsymbol{z}\right)\right\|_2$$
$$= \|\beta \boldsymbol{e}_1^{(k)} - \boldsymbol{H}_k \boldsymbol{z}\|_2.$$

而 $\boldsymbol{H}_k$ 非奇异, 当 $\boldsymbol{z}_k = \boldsymbol{H}_k^{-1}(\beta \boldsymbol{e}_1^{(k)})$ 时, $\|\beta \boldsymbol{v}_1 - \boldsymbol{A}\boldsymbol{V}_k \boldsymbol{z}\|_2$ 达到极小值 0, 即 $\|\boldsymbol{r}_k\|_2 = 0$, 故

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{z}_k$$

是精确解, 换言之, GMRES 方法若在第 k 步中断, 则在这一步已得到精确解. 证毕. □

下面的定理是 GMRES 方法的另一个性质.

定理 4.5 设 $\{\boldsymbol{x}_i\}$ 是 GMRES 方法产生的迭代序列. 若 $\boldsymbol{x}_k$ 是精确解, 而 $\boldsymbol{x}_i\,(i < k)$ 不是精确解, 则算法在第 k 步中断.

证明 由假设, 有

$$\boldsymbol{r}_i \neq \mathbf{0}, \quad i = 1, 2, \cdots, k-1, \quad \text{而 } \boldsymbol{r}_k = \mathbf{0}.$$

但 $\|\boldsymbol{r}_k\|_2$ 是 $\widetilde{\boldsymbol{\xi}}_k = \beta \boldsymbol{V}_k \boldsymbol{e}_1^{(k+1)}$ 最后一个分量的绝对值, 即它是

$$\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & c_k & s_k \\ & & & -s_k & c_k \end{bmatrix} \begin{bmatrix} \\ \widetilde{\boldsymbol{\xi}}_{k-1} \\ \\ \\ 0 \end{bmatrix}$$

的第 $k+1$ 个分量的绝对值, 即

$$\|\boldsymbol{r}_k\|_2 = |s_k \boldsymbol{e}_k^{\mathrm{T}} \widetilde{\boldsymbol{\xi}}_{k-1}|. \tag{4.64}$$

但

$$|\boldsymbol{e}_k^{\mathrm{T}} \widetilde{\boldsymbol{\xi}}_{k-1}| = \|\boldsymbol{r}_{k-1}\|_2 \neq 0, \tag{4.65}$$

故 $s_k = 0$. 由于

$$s_k = \frac{h_{k+1,k}}{\sqrt{h_{k,k}^2 + h_{k+1,k}^2}},$$

故有 $h_{k+1,k} = 0$, 这样 $\widetilde{\boldsymbol{v}}_{k+1} = \mathbf{0}$ (由式 (4.53) 所定义), 算法中断. 证毕. □

由式 (4.64) 和式 (4.65) 容易得到下面的推论.

推论 4.2 GMRES 方法的残量范数 $\|\boldsymbol{r}_k\|_2$ 有表达式

$$\|\boldsymbol{r}_k\|_2 = \left(\prod_{i=1}^{k} |s_i|\right) \|\boldsymbol{r}_0\|_2. \tag{4.66}$$

定理 4.6 初始残量 $\boldsymbol{r}_0$ 的最小多项式次数是 k 的充分必要条件是 $\widetilde{\boldsymbol{v}}_{k+1} = \mathbf{0}$ 且 $\widetilde{\boldsymbol{v}}_i \neq \mathbf{0} (i = 1, 2, \cdots, k)$, 其中 $\widetilde{\boldsymbol{v}}_{i+1} (i = 1, 2, \cdots, k)$ 由式 (4.53) 所定义.

证明 若 $\boldsymbol{r}_0 = \|\boldsymbol{r}_0\|_2 \boldsymbol{v}_1$ 的最小多项式次数为 k, 则存在一个 k 次多项式 ϕ_k, 使得 $\phi_k(\boldsymbol{A})\boldsymbol{v}_1 = \boldsymbol{0}$, 且 ϕ_k 是次数最低者. 因此

$$\mathcal{K}_{k+1} = \mathrm{span}\{\boldsymbol{v}_1, \boldsymbol{A}\boldsymbol{v}_1, \cdots, \boldsymbol{A}^k\boldsymbol{v}_1\} = \mathcal{K}_k.$$

由于 $\widetilde{\boldsymbol{v}}_{k+1} \in \mathcal{K}_{k+1} = \mathcal{K}_k$, 且 $\widetilde{\boldsymbol{v}}_{k+1} \perp \mathcal{K}_k$, 故 $\widetilde{\boldsymbol{v}}_{k+1} = \boldsymbol{0}$. 另外, 若存在某个 $i\,(1 \leqslant i \leqslant k)$, 使 $\widetilde{\boldsymbol{v}}_i = \boldsymbol{0}$, 则存在一个 $i-1$ 次多项式 ϕ_{i-1} 使 $\phi_{i-1}(\boldsymbol{A})\boldsymbol{v}_1 = \boldsymbol{0}$, 这与 $\boldsymbol{v}_1$ 的最小多项式次数为 k 矛盾, 故 $\widetilde{\boldsymbol{v}}_i \neq \boldsymbol{0}$, $1 \leqslant i \leqslant k$.

反之, 设 $\widetilde{\boldsymbol{v}}_{k+1} = \boldsymbol{0}$ 且 $\widetilde{\boldsymbol{v}}_i \neq \boldsymbol{0}$, $i = 1, 2, \cdots, k$. 由 $\widetilde{\boldsymbol{v}}_{k+1} = \boldsymbol{0}$ 可知, 存在一个 k 次多项式 ϕ_k, 使 $\phi_k(\boldsymbol{A})\boldsymbol{v}_1 = \boldsymbol{0}$, 而且 k 是次数最小者. 否则, 当存在 $\phi_i\,(i < k)$ 使 $\phi_i(\boldsymbol{A})\boldsymbol{v}_1 = \boldsymbol{0}$ 时, 就有 $\mathcal{K}_{i+1} = \mathcal{K}_i$, 因此 $\widetilde{\boldsymbol{v}}_{i+1} = \boldsymbol{0}$, 这与 $\widetilde{\boldsymbol{v}}_i \neq \boldsymbol{0}\,(1 \leqslant i \leqslant k)$ 矛盾. 证毕. □

进一步, 由上述三个定理可以推出:

推论 4.3 GMRES 方法在第 k 步产生的解 $\boldsymbol{x}_k$ 是精确解的充分必要条件为下列诸等价条件:

(1) 算法在第 k 步中断.

(2) $h_{k+1,k} = 0$.

(3) $\widetilde{\boldsymbol{v}}_{k+1} = \mathbf{0}$.

(4) $\boldsymbol{r}_0$ 的最小多项式次数为 k.

推论 4.4 GMRES 方法得到的残量模序列 $\{\|\boldsymbol{r}_k\|_2\}$ 是单调下降的, 对于 n 阶方程组至多迭代 n 步即可得到精确解.

现在考虑 GMRES(m) 方法, 虽然它总能进行下去, 但可能不收敛. 当然, 当 m 充分大时是收敛的. 特别地, 当 $m = n$ 时, 一个重开始迭代即可得到精确解.

1. GMRES(m) 的收敛性定理

Back

Close

现在考虑 GMRES(m) 方法的误差估计. 利用 Krylov 子空间的性质, 注意到任意的 $\boldsymbol{x} = \boldsymbol{x}_0 + \varphi_{k-1}(\boldsymbol{A})\boldsymbol{r}_0$ (即 $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$), 可导出

$$\begin{aligned}\|\boldsymbol{r}_k\|_2 = \min \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 &= \min_{\varphi_{k-1}\in\mathcal{P}_{k-1}} \|\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_0 + \varphi_{k-1}(\boldsymbol{A})\boldsymbol{r}_0)\|_2 \\ &= \min_{\varphi_{k-1}\in\mathcal{P}_{k-1}} \|(\boldsymbol{I} - \boldsymbol{A}\varphi_{k-1}(\boldsymbol{A}))\boldsymbol{r}_0\|_2 = \min_{\psi\in\mathcal{P}_k^0} \|\psi(\boldsymbol{A})\boldsymbol{r}_0\|_2,\end{aligned} \tag{4.67}$$

式中: $\mathcal{P}_k^0$ 如式 (4.41) 中所定义.

定理 4.7 设 $\boldsymbol{A}$ 是正定的 (即 $\boldsymbol{A}$ 的对称部分是对称正定的), $m > 0$ 是任意给定的整数, 并假定在 GMRES(m) 中重新开始了 ℓ 次产生了近似解 $x_m^{(\ell)}$, 则有

$$\frac{\|\boldsymbol{r}_m^{(\ell)}\|_2}{\|\boldsymbol{r}_0\|_2} \leqslant \left[\sqrt{1 - \frac{\lambda_{\min}^2(\boldsymbol{M})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}}\right]^{\ell m}, \tag{4.68}$$

式中: $\boldsymbol{M}=\frac{1}{2}(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})$, 其他的记号如前所述.

证明 因为 $\boldsymbol{A}$ 是正定的, 即其对称部分

$$\boldsymbol{M}=\frac{1}{2}(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}}) \tag{4.69}$$

是对称正定的, 定义

$$\psi_\alpha(t)=(1+\alpha t)^m,\quad \alpha\in\mathbb{R}. \tag{4.70}$$

显然有 $\psi_\alpha(t)\in\mathcal{P}_m^0$. 下面先给出 $\min\limits_{\alpha}\|\psi_\alpha(\boldsymbol{A})\|_2$ 的上界估计.

现在任取一个 $\boldsymbol{u}\in\mathbb{R}^n, \|\boldsymbol{u}\|_2=1$ (即 $\boldsymbol{u}$ 是单位向量), 并记 $\widehat{\psi}_\alpha(t)=1+\alpha t$, 则有

$$\begin{aligned}\|\widehat{\psi}_\alpha(\boldsymbol{A})\boldsymbol{u}\|_2^2&=\boldsymbol{u}^{\mathrm{T}}(\boldsymbol{I}+\alpha\boldsymbol{A})^{\mathrm{T}}(\boldsymbol{I}+\alpha\boldsymbol{A})\boldsymbol{u}\\&=1+2\alpha\boldsymbol{u}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{u}+\alpha^2\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u},\end{aligned}$$




从而, 当 $\alpha \geqslant 0$ 时, 有

$$\|\widehat{\psi}_\alpha(\boldsymbol{A})\|_2 \geqslant \|\widehat{\psi}_\alpha(\boldsymbol{A})\boldsymbol{u}\|_2 \geqslant 1, \tag{4.71}$$

而当 $\alpha \leqslant 0$ 时, 有

$$\|\widehat{\psi}_\alpha(\boldsymbol{A})\boldsymbol{u}\|_2^2 \leqslant 1 + 2\alpha\lambda_{\min}(\boldsymbol{M}) + \alpha^2\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}), \tag{4.72}$$

式中: $\lambda_{\min}(\boldsymbol{M})$ 和 $\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})$ 分别为 $\boldsymbol{M}$ 的最小特征值和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的最大特征值.

注意到单位向量 $\boldsymbol{u}$ 的任意性, 式 (4.72) 蕴涵着当 $\alpha \leqslant 0$ 时, 有

$$\|\widehat{\psi}_\alpha(\boldsymbol{A})\|_2^2 \leqslant 1 + 2\alpha\lambda_{\min}(\boldsymbol{M}) + \alpha^2\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}). \tag{4.73}$$

不等式 (4.73) 的右边是关于 α 的二次函数, 在

$$\alpha = -\frac{\lambda_{\min}(\boldsymbol{M})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})} \leqslant 0$$

处达到最小值 $1-\lambda_{\min}^2(\boldsymbol{M})/\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})$, 从而有

$$\min_{\alpha<0}\|\widehat{\psi}_\alpha(\boldsymbol{A})\|_2^2 \leqslant 1-\frac{\lambda_{\min}^2(\boldsymbol{M})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}<1.$$

再注意到式 (4.71) , 有

$$\min_{\alpha\in\mathbb{R}}\|\widehat{\psi}_\alpha(\boldsymbol{A})\|_2 \leqslant \sqrt{1-\frac{\lambda_{\min}^2(\boldsymbol{M})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}}\equiv\kappa. \tag{4.74}$$

如果从 $\boldsymbol{x}_0$ 出发, 应用 GMRES 迭代 m 步得到 $\boldsymbol{x}_m$, 则对 $k=m$ 应用式 (4.67), 有

$$\begin{aligned}\|\boldsymbol{r}_m\|_2 &\leqslant \min_\alpha\|\psi_\alpha(\boldsymbol{A})\boldsymbol{r}_0\|_2 \leqslant \min_\alpha\|\psi_\alpha(\boldsymbol{A})\|_2\|\boldsymbol{r}_0\|_2\\ &\leqslant \min_\alpha\|\widehat{\psi}_\alpha(\boldsymbol{A})\|_2^m\|\boldsymbol{r}_0\|_2 \leqslant \kappa^m\|\boldsymbol{r}_0\|_2,\end{aligned} \tag{4.75}$$

其中最后一个不等式用到了式 (4.74).

Back

Close

假定以 $\boldsymbol{x}_m^{(1)} = \boldsymbol{x}_m$ 再重新开始, 用 GMRES 迭代产生 $\boldsymbol{x}_m^{(2)}$, 则由式 (4.75) 又有

$$\|\boldsymbol{r}_m^{(2)}\|_2 \leqslant \kappa^m \|\boldsymbol{r}_m^{(1)}\|_2 \leqslant \kappa^{2m} \|\boldsymbol{r}_0\|_2,$$

式中:

$$\boldsymbol{r}_m^{(2)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m^{(2)};\ \boldsymbol{r}_m^{(1)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m^{(1)} = \boldsymbol{r}_m.$$

如此可证, 若重新开始了 ℓ 次, 产生了 $\boldsymbol{x}_m^{(\ell)}$, 则有

$$\|\boldsymbol{r}_m^{(\ell)}\|_2 \leqslant \kappa^{\ell m} \|\boldsymbol{r}_0\|_2,$$

式中: $\boldsymbol{r}_m^{(\ell)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m^{(\ell)}$. 将 κ 的表达式代入上式即得定理的结论. 证毕. □

注 4.3 定理 4.7 表明, 如果系数矩阵 $\boldsymbol{A}$ 是正定的, 则对任意给定的正数 $m > 0$, GMRES (m) 总是收敛的.

2. 儒科夫斯基映射

要研究 GMRES 方法对更广一类线性方程组的收敛性, 需要借助复变函数中的儒科夫斯基 (Joukowski) 映射来导出复变元的 Chebyshew 多项式的一种易于使用的定义.

儒科夫斯基映射是指如下定义的从复平面到复平面的映射:

$$z = \frac{1}{2}(w + w^{-1}) \equiv J(w). \tag{4.76}$$

对任意的 $w = re^{i\theta}$, $r > 1$, 有

$$z = J(w) = x + \mathrm{i}\, y \equiv a_r \cos\theta + \mathrm{i}\, b_r \sin\theta,$$

式中:

$$a_r = \frac{1}{2}(r + r^{-1}), \quad b_r = \frac{1}{2}(r - r^{-1}). \tag{4.77}$$

从几何上看, J 将 w 平面内的圆

$$C_r = \{w = re^{i\theta} : 0 \leqslant \theta \leqslant 2\pi\},$$

Back

Close

映射到 z 平面的一个椭圆

$$E_r = \left\{ z = x + \mathrm{i}\, y : \frac{x^2}{a_r^2} + \frac{y^2}{b_r^2} = 1 \right\}.$$

显然, 这一映射也将 w 平面内的圆

$$C_{r-1} = \left\{ w = r^{-1} \mathrm{e}^{-\mathrm{i}\theta} : 0 \leqslant \theta \leqslant 2\pi \right\},$$

映射到了 E_r.

当 $r = 1$ 时, 有 $b_r = 0$. 此时 $z = J(w) = \cos\theta$, 即 J 把单位圆周映射到 z 平面内的线段 $[-1, 1]$. 当 θ 从 0 变到 π 时, z 从 1 变到 -1; 而当 θ 从 π 变到 2π 时, z 从 -1 变到 1.

当 C_r 连续地收缩到单位圆周 C_1 时, E_r 将连续地收缩到线段 $[-1, 1]$. 因此, 由 E_r 所包围的区域

$$E_r = \left\{ z = x + \mathrm{i}\, y : \frac{x^2}{a_r^2} + \frac{y^2}{b_r^2} \leqslant 1 \right\}$$

中的每一个点 z, 在圆环

$$C_{1,r} = \{w : 1 \leqslant |w| \leqslant r\}$$

中都存在一个点 w, 使得 $z = J(w)$, 即 $J(C_{1,r}) = E_r$.

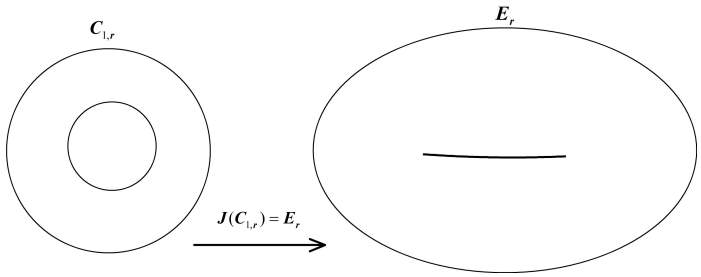


图 4.8 儒可夫斯基映射 $z = J(w)$

现在将 z 平面沿线段 $[-1, 1]$ 切开, 取定 $\sqrt{z^2 - 1}$ 的一个解析分支, 使得

$$w = z + \sqrt{z^2 - 1} \equiv J^{-1}(z). \tag{4.78}$$

刚好在单位圆外, 即 $|w| \geqslant 1$. 显然, J^{-1} 刚好将 z 平面内的椭圆 E_r

映射到 w 平面内的圆环 $C_{1,r}$, 而且有 $J^{-1}(E_r) = C_{1,r}$.

下面考察 z 平面内的一个中心位于 $z_c = x_c + \mathrm{i}\, y_c$ 的椭圆

$$E(a, b; z_c) = \left\{ z = x + \mathrm{i}\, y : \frac{(x - x_c)^2}{a^2} + \frac{(y - y_c)^2}{b^2} = 1 \right\},$$

这里 $0 < b < a$. 令 $d = \sqrt{a^2 - b^2}$, 即 d 为椭圆的半焦距. 现在作平移伸缩变换

$$\widetilde{z} = \frac{z - z_c}{d} \quad \left(\text{即} \quad \widetilde{x} = \frac{x - x_c}{d}, \quad \widetilde{y} = \frac{y - y_c}{d} \right),$$

则该变换将椭圆 $E(a, b; z_c)$ 变为中心在原点的椭圆

$$E_{\widetilde{r}} = \left\{ \widetilde{z} = \widetilde{x} + \mathrm{i}\, \widetilde{y} : \frac{\widetilde{x}^2}{a_{\widetilde{r}}^2} + \frac{\widetilde{y}^2}{b_{\widetilde{r}}^2} = 1 \right\},$$

式中: $\widetilde{r}$ 为

$$\frac{a}{d} = \frac{1}{2}(r + r^{-1})$$

的最大根

$$\widetilde{r} = \frac{a}{d} + \sqrt{\left(\frac{a}{d}\right)^2 - 1}\,, \tag{4.79}$$

$a_{\widetilde{r}}$ 和 $b_{\widetilde{r}}$ 是在式 (4.77) 中将 r 换作 $\widetilde{r}$ 而得到的.

再由儒科夫斯基映射的性质, 映射

$$\frac{z - z_c}{d} = \frac{w + w^{-1}}{2} \tag{4.80}$$

实现了区域

$$E(a, b; z_c) = \left\{ z = x + \mathrm{i}\, y : \frac{(x - x_c)^2}{a^2} + \frac{(y - y_c)^2}{b^2} \leqslant 1 \right\} \tag{4.81}$$

与圆环

$$C_{1,\widetilde{r}} = \left\{ w : 1 \leqslant |w| \leqslant \widetilde{r} \right\} \tag{4.82}$$

之间的点与点之间的一一对应关系.

3. 复变元的 Chebyshev 多项式

借助儒科夫斯基映射, 定义函数

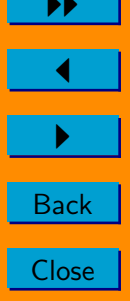
$$C_k(z) = \frac{1}{2}(w^k + w^{-k}), \tag{4.83}$$

式中:

$$z = \frac{1}{2}(w + w^{-1}).$$

由定义显然有

$$C_0(z) = \frac{1}{2}(w^0 + w^0) = 1,$$
$$C_1(z) = \frac{1}{2}(w + w^{-1}) = z.$$



而且容易导出

$$\begin{aligned} zC_k(z) &= \frac{1}{2}(w + w^{-1}) \cdot \frac{1}{2}(w^k + w^{-k}) \\ &= \frac{1}{4}\left[(w^{k+1} + w^{-(k+1)}) + (w^{k-1} + w^{-(k-1)})\right] \\ &= \frac{1}{2}\left[C_{k+1}(z) + C_{k-1}(z)\right], \end{aligned}$$

从而有

$$C_{k+1}(z) = 2zC_k(z) - C_{k-1}(z).$$

这表明由式 (4.83) 所定义的复变函数就是第一类 Chebyshev 多项式. 换句话说, 式 (4.83) 是复变元的 Chebyshev 多项式的又一种表达方式.

再由式 (4.78), 即知式 (4.83) 又可写为

$$C_k(z) = \frac{1}{2}\left[\left(z + \sqrt{z^2 - 1}\right)^k + \left(z + \sqrt{z^2 - 1}\right)^{-k}\right].$$




Back
Close

现在考虑 GMRES 方法的收敛性和误差估计. 设 $\boldsymbol{A}$ 是可对角化的, 即存在一个非奇异矩阵 $\boldsymbol{X} \in \mathbb{R}^{n\times n}$, 使得 $\boldsymbol{A} = \boldsymbol{X}\boldsymbol{\Lambda}\boldsymbol{X}^{-1}$, 其中 $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$. 注意到

$$\psi(\boldsymbol{A}) = \boldsymbol{X}\psi(\boldsymbol{\Lambda})\boldsymbol{X}^{-1}, \quad \psi(\boldsymbol{\Lambda}) = \mathrm{diag}\{\psi(\lambda_1), \psi(\lambda_2), \cdots, \psi(\lambda_n)\},$$

由式 (4.67), 得

$$\|\boldsymbol{r}_k\|_2 \leqslant \|\boldsymbol{X}\|_2 \cdot \|\boldsymbol{X}^{-1}\|_2 \min_{\psi\in\mathcal{P}_k^0} \max_{1\leqslant i\leqslant n} |\psi(\lambda_i)| \cdot \|\boldsymbol{r}_0\|_2. \tag{4.84}$$

再定义

$$\nu(\boldsymbol{A}) = \inf\{\|\boldsymbol{X}\|_2\|\boldsymbol{X}^{-1}\|_2 : \boldsymbol{A} = \boldsymbol{X}\boldsymbol{\Lambda}\boldsymbol{X}^{-1}\}, \tag{4.85}$$

则有

$$\|\boldsymbol{r}_k\|_2 \leqslant \nu(\boldsymbol{A}) \min_{\psi\in\mathcal{P}_k^0} \max_{1\leqslant i\leqslant n} |\psi(\lambda_i)| \cdot \|\boldsymbol{r}_0\|_2, \tag{4.86}$$

其中由式 (4.85) 所定义的数 $\nu(\boldsymbol{A})$ 被称为 $\boldsymbol{A}$ 的谱条件数.

下面通过选择特殊的多项式 $\psi \in \mathcal{P}_k^0$ 给出

$$\min_{\psi \in \mathcal{P}_k^0} \max_{1 \leqslant i \leqslant n} |\psi(\lambda_i)|$$

的尽可能小的上界估计. 由于这里的 λ_i 可能是复数, 因此比对称矩阵时的相应估计要困难得多, 需要用到复变元 Chebyshev 多项式的有关性质.

定理 4.8 设式 (4.51) 的系数矩阵 $\boldsymbol{A}$ 是可对角化的, 并满足

$$\lambda(\boldsymbol{A}) \subset E(a, b; z_c),$$

其中 $z_c = c$, 而且 $0 < b < a < c$, 并假设 GMRES 方法已经进行了 k 步得到了 $\boldsymbol{x}_k$, 则残量 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 满足

$$\frac{\|\boldsymbol{r}_k\|_2}{\|\boldsymbol{r}_0\|_2} \leqslant 2\nu(\boldsymbol{A}) \left(\frac{a+b}{c + \sqrt{c^2 + b^2 - a^2}} \right)^k, \tag{4.87}$$

这里的 $\nu(\boldsymbol{A})$ 由式 (4.85) 所定义.

证明 令 $d=\sqrt{a^2-b^2}$, 并定义

$$\widehat{\psi}(z)=C_k\Big(\frac{z-z_c}{d}\Big)/C_k\Big(\frac{-z_c}{d}\Big),$$

则有 $\widehat{\psi}\in\mathcal{P}_k^0$. 这样, 由式 (4.86), 有

$$\frac{\|\boldsymbol{r}_k\|_2}{\|\boldsymbol{r}_0\|_2}\leqslant\nu(\boldsymbol{A})\max_{\lambda\in\lambda(\boldsymbol{A})}|\widehat{\psi}(\lambda)|\leqslant\nu(\boldsymbol{A})\max_{z\in E(a,b,z_c)}|\widehat{\psi}(z)|. \tag{4.88}$$

由式 (4.83) 和式 (4.88), 有

$$C_k\Big(\frac{z-z_c}{d}\Big)=\frac{1}{2}(w^k+w^{-k}). \tag{4.89}$$

式中:

$$\frac{z-z_c}{d}=\frac{1}{2}(w+w^{-1}). \tag{4.90}$$

注意: 式 (4.90) 所定义的映射实现了区域 $E(a,b,z_c)$ 与圆环 $C_{1,\tilde{r}}$ 之间的点与点之间的一一对应关系, 有

$$\max_{z\in E(a,b,z_c)}|\widehat{\psi}(z)|=\max_{w\in C_{1,\tilde{r}}}\left|\frac{w^k+w^{-k}}{2C_k(-z_c/d)}\right|. \tag{4.91}$$

在式 (4.90) 中令 $z=0$, 得

$$-\frac{z_c}{d}=-\frac{c}{d}=\frac{1}{2}(w_0+w_0^{-1}). \tag{4.92}$$

解此方程, 并选择其模最大者, 即

$$w_0=-\frac{c}{d}-\sqrt{\left(\frac{c}{d}\right)^2-1}. \tag{4.93}$$

此外, 对任意的 $w=r\mathrm{e}^{\mathrm{i}\theta}$, 其中 $1\leqslant r\leqslant\widetilde{r}$, 有

$$w^k+w^{-k}=(r^k+r^{-k})\cos k\theta+\mathrm{i}\,(r^k-r^{-k})\sin k\theta,$$

从而有

$$\begin{aligned}|w^k+w^{-k}|^2&=(r^k+r^{-k})^2\cos^2 k\theta+(r^k-r^{-k})^2\sin^2 k\theta\\&=(r^k-r^{-k})^2+4\cos^2 k\theta.\end{aligned}$$

显然, 该函数在圆周 $w=r\mathrm{e}^{\mathrm{i}\theta}$ 上的最大值为

$$(r^k-r^{-k})^2+4=(r^k+r^{-k})^2,$$

从而

$$|w^k + w^{-k}| \leqslant r^k + r^{-k}, \quad w = r\mathrm{e}^{\mathrm{i}\theta}. \tag{4.94}$$

这样, 由式 (4.89), 式 (4.91), 式 (4.92) 和式 (4.94), 有

$$\max_{z\in E(a,b,z_c)} |\widehat{\psi}(z)| = \frac{\widetilde{r}^k + \widetilde{r}^{-k}}{|w_0^k + w_0^{-k}|}, \tag{4.95}$$

这里 $\widetilde{r}$ 由 (4.79) 定义.

此外, 由 $\widetilde{r}$ 的定义式 (4.79), w_0 的定义 (4.93) 以及 $d^2 = a^2 - b^2$, 得

$$\widetilde{r} = \frac{a+b}{d}, \quad w_0 = -\frac{c + \sqrt{c^2 + b^2 - a^2}}{d},$$

从而

$$
\begin{aligned}
&\frac{\widetilde{r}^k+\widetilde{r}^{-k}}{|w_0^k+w_0^{-k}|}\\
&=\frac{[(a+b)/d]^k+[(a+b)/d]^{-k}}{\left[(c+\sqrt{c^2+b^2-a^2})/d\right]^k+\left[(c+\sqrt{c^2+b^2-a^2})/d\right]^{-k}}\\
&=\left(\frac{a+b}{c+\sqrt{c^2+b^2-a^2}}\right)^k\frac{1+[d/(a+b)]^{2k}}{1+\left[(c+\sqrt{c^2+b^2-a^2})/d\right]^{-2k}}\\
&\leqslant\left(\frac{a+b}{c+\sqrt{c^2+b^2-a^2}}\right)^k\left[1+\left(\frac{a-b}{a+b}\right)^k\right]\\
&\leqslant 2\left(\frac{a+b}{c+\sqrt{c^2+b^2-a^2}}\right)^k. \qquad (4.96)
\end{aligned}
$$

将式 (4.96), 式 (4.95) 和式 (4.88) 结合起来即得定理的结论. 证毕.

□

本节介绍求解对称不定线性方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.97}$$

的极小残量法, 其中系数矩阵 $\boldsymbol{A}$ 是给定的对称不定矩阵 (其特征值有正有负), $\boldsymbol{b} \in \mathbb{R}^n$ 是给定的列向量, 而 $\boldsymbol{x} \in \mathbb{R}^n$ 是待求的未知向量.

若将 4.1 节所介绍的共轭梯度法应用到这类线性方程组上, 其投影方程组 (4.13) 的系数矩阵 $\boldsymbol{T}_k$ 就可能是奇异的, 从而导致算法失败 (即在没有求得式 (4.97) 的很好的近似解之前就发生中断). 基于这种情况, Paige 和 Saunders 于 1975 年在文献 [26] 中提出了一种克服这一困难的方法, 导出了求解对称不定线性方程组的 SYMMLQ 方法. 此外, 在该文献中还给出了求解这类方程组的

极小残量法 (MINRES). 本节着重介绍极小残量法, SYMMLQ 方法将在后面再做介绍.

§4.3.1 MINRES 方法

极小残量法的出发点是寻找一个 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得

$$\|\boldsymbol{r}_k\|_2 = \min\{\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 : \boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)\}, \tag{4.98}$$

这里 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 是残量, 而 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 是由矩阵 $\boldsymbol{A}$ 和向量 $\boldsymbol{r}_0$ 所生成的 Krylov 子空间.

假设已经求得一个以 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$ 为初始向量的长度为 k 的 Lanczos 分解

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{T}}_k, \tag{4.99}$$




Back
Close

式中:

$$\boldsymbol{V}_{k+1} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k, \boldsymbol{v}_{k+1}] \in \mathbb{R}^{n\times(k+1)}$$

满足 $\boldsymbol{V}_{k+1}^{\mathrm{T}}\boldsymbol{V}_{k+1} = \boldsymbol{I}_{k+1}$, 而 $\widetilde{\boldsymbol{T}}_k$ 是 $(k+1)\times k$ 阶的三对角矩阵, 即

$$\widetilde{\boldsymbol{T}}_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_k \\ & & & 0 & \beta_k \end{bmatrix}, \ \beta_i \neq 0. \tag{4.100}$$

注意到 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathcal{R}(\boldsymbol{V}_k)$, 则对任意的 $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 有

$$\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2=\|\boldsymbol{r}_0-\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{z}\|_2=\|\beta\boldsymbol{V}_{k+1}\boldsymbol{e}_1-\boldsymbol{V}_{k+1}\widetilde{\boldsymbol{T}}_k\boldsymbol{z}\|_2$$
$$=\|\boldsymbol{V}_{k+1}(\beta\boldsymbol{e}_1-\widetilde{\boldsymbol{T}}_k\boldsymbol{z})\|_2=\|\beta\boldsymbol{e}_1-\widetilde{\boldsymbol{T}}_k\boldsymbol{z}\|_2, \qquad (4.101)$$

式中: $\beta=\|\boldsymbol{r}_0\|_2$. 这样求 $\boldsymbol{x}_k\in\boldsymbol{x}_0+\mathcal{K}_k(\boldsymbol{A},\boldsymbol{r}_0)$ 满足式 (4.98) 就等价于求 $\boldsymbol{z}_k\in\mathbb{R}^k$, 使得

$$\|\beta\boldsymbol{e}_1-\widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k\|_2=\min\{\|\beta\boldsymbol{e}_1-\widetilde{\boldsymbol{T}}_k\boldsymbol{z}\|_2:\boldsymbol{z}\in\mathbb{R}^k\}. \qquad (4.102)$$

一旦这样的 $\boldsymbol{z}_k$ 求得, 则所寻求的 $\boldsymbol{x}_k$ 就是 $\boldsymbol{x}_k=\boldsymbol{x}_0+\boldsymbol{V}_k\boldsymbol{z}_k$.

极小化问题 (4.102) 是一个系数矩阵为三对角矩阵的最小二乘问题, 现在已有很多十分成熟的数值方法来求它的解. 这里将用 QR 分解来求解式 (4.102).

由于 $\widetilde{\boldsymbol{T}}_k$ 具有式 (4.100) 所示形状, 故可以计算 k 个 Givens 变

换 $\boldsymbol{G}_1, \boldsymbol{G}_2, \cdots, \boldsymbol{G}_k$, 使得

$$\boldsymbol{G}_k \boldsymbol{G}_{k-1} \cdots \boldsymbol{G}_2 \boldsymbol{G}_1 \widetilde{\boldsymbol{T}}_k = \begin{bmatrix} \boldsymbol{R}_k \\ \boldsymbol{0} \end{bmatrix}, \tag{4.103}$$

式中:

$$\boldsymbol{G}_i = \operatorname{diag}\left(\boldsymbol{I}_{i-1}, \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix}, \boldsymbol{I}_{k-i} \right) \in \mathbb{R}^{(k+1)\times(k+1)}, \quad c_i^2 + s_i^2 = 1,$$

$$\boldsymbol{R}_k = \begin{bmatrix} \gamma_1 & \delta_1 & \varepsilon_1 & & \\ & \gamma_2 & \delta_2 & \ddots & \\ & & \ddots & \ddots & \varepsilon_{k-2} \\ & & & \gamma_{k-1} & \delta_{k-1} \\ & & & & \gamma_k \end{bmatrix}, \tag{4.104}$$

而且 $\beta_i \neq 0$ 蕴涵着 $\gamma_i \neq 0,\ i = 1, 2, \cdots, k$, 从而 $\boldsymbol{R}_k$ 是非奇异的. 令

$$\boldsymbol{G} = \boldsymbol{G}_k \boldsymbol{G}_{k-1} \cdots \boldsymbol{G}_2 \boldsymbol{G}_1, \quad \begin{bmatrix} \boldsymbol{t}_k \\ \rho_k \end{bmatrix} = \boldsymbol{G}(\beta \boldsymbol{e}_1), \quad \boldsymbol{t}_k = (\tau_1, \tau_2, \cdots, \tau_k)^{\mathrm{T}}, \tag{4.105}$$

则 $\boldsymbol{G}$ 是 $k+1$ 阶正交矩阵, 且 $\tau_k,\ \rho_k$ 如式 (4.58) 所定义, 即

$$\begin{cases} \tau_1 = \beta c_1, \\ \tau_i = (-1)^{i-1} \beta s_1 s_2 \cdots s_{i-1} c_i, \quad i = 2, 3, \cdots, k, \\ \rho_k = (-1)^k \beta s_1 s_2 \cdots s_k. \end{cases} \tag{4.106}$$

这样, 利用分解式 (4.103) 和记号 (4.105), 有

$$\begin{aligned} \left\| \widetilde{\boldsymbol{T}}_k \boldsymbol{z} - \beta \boldsymbol{e}_1 \right\|_2^2 &= \left\| \boldsymbol{G}(\widetilde{\boldsymbol{T}}_k \boldsymbol{z} - \beta \boldsymbol{e}_1) \right\|_2^2 = \left\| \begin{bmatrix} \boldsymbol{R}_k \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{z} - \begin{bmatrix} \boldsymbol{t}_k \\ \rho_k \end{bmatrix} \right\|_2^2 \\ &= \left\| \boldsymbol{R}_k \boldsymbol{z} - \boldsymbol{t}_k \right\|_2^2 + \rho_k^2. \end{aligned}$$

对任意的 $\boldsymbol{z} \in \mathbb{R}^k$ 成立. 由此立即知道, 最小二乘问题 (4.102) 有唯一解, 即

$$\boldsymbol{z}_k = \boldsymbol{R}_k^{-1}\boldsymbol{t}_k, \tag{4.107}$$

而且有

$$\|\widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k - \beta\boldsymbol{e}_1\|_2 = |\rho_k|. \tag{4.108}$$

由式 (4.107) 求得 $\boldsymbol{z}_k$ 之后, 就可算出所求的 $\boldsymbol{x}_k$ 为 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$.

由式 (4.101) 和式 (4.108), 得

$$\|\boldsymbol{r}_k\|_2 = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\|_2 = \|\widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k - \beta\boldsymbol{e}_1\|_2 = |\rho_k|,$$

故在实际计算时, 可以用

$$|\rho_k|/\beta \leqslant \varepsilon \tag{4.109}$$

作为迭代终止的准则, 其中 $\varepsilon > 0$ 是给定的误差要求.




Back
Close

由式 (4.106) 可知, ρ_k 的值并不需要 $\boldsymbol{z}_k$ 和 $\boldsymbol{x}_k$ 的信息. 因此, 只需在 ρ_k 满足式 (4.109) 之后, 再去计算 $\boldsymbol{z}_k$ 和 $\boldsymbol{x}_k$ 即可.

至此, 已经给出了求解极小化问题 (4.98) 的具体方法, 然而这样做的一个最大缺点就是需要保存 $\boldsymbol{V}_k$ 的所有列向量, 随着 k 的增加, 存储量的需求会越来越大. 幸运的是, 在文献 [26] 中还给出了一种巧妙的方法来避免这种情形的出现. 这一方法的具体做法是不将 $\boldsymbol{z}_k$ 明确求出, 而是由 $\boldsymbol{z}_k$ 直接导出计算 $\boldsymbol{x}_k$ 的递推公式.

将式 (4.107) 代入 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$, 得

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{R}_k^{-1}\boldsymbol{t}_k = \boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{t}_k, \tag{4.110}$$

式中: $\boldsymbol{P}_k = \boldsymbol{V}_k\boldsymbol{R}_k^{-1}$. 这样, 只要将 $\boldsymbol{P}_k$ 算出, 就可以通过式 (4.110) 计算 $\boldsymbol{x}_k$. 令 $\boldsymbol{P}_k = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_k]$, 比较 $\boldsymbol{P}_k\boldsymbol{R}_k = \boldsymbol{V}_k$ 两边的每一列, 得

$$\gamma_1 \boldsymbol{p}_1 = \boldsymbol{v}_1,$$
$$\delta_1 \boldsymbol{p}_1 + \gamma_2 \boldsymbol{p}_2 = \boldsymbol{v}_2,$$
$$\varepsilon_{i-2}\boldsymbol{p}_{i-2} + \delta_{i-1}\boldsymbol{p}_{i-1} + \gamma_i \boldsymbol{p}_i = \boldsymbol{v}_i,\ i = 3, 4, \cdots, k.$$

由此可求得 $\boldsymbol{P}_k$ 的列向量为

$$\begin{cases} \boldsymbol{p}_1 = \boldsymbol{v}_1/\gamma_1, \\ \boldsymbol{p}_2 = (\boldsymbol{v}_2 - \delta_1 \boldsymbol{p}_1)/\gamma_2, \\ \boldsymbol{p}_i = (\boldsymbol{v}_i - \varepsilon_{i-2}\boldsymbol{p}_{i-2} - \delta_{i-1}\boldsymbol{p}_{i-1})/\gamma_i, \quad i = 3, 4, \cdots, k. \end{cases} \tag{4.111}$$

下面借助式 (4.106), 式 (4.110) 和式 (4.111) 导出计算 $\boldsymbol{x}_k$ 的递推公式.

首先注意, 若 Lanczos 分解的长度由 k 增加到 $k+1$, 则有

$$\widetilde{\boldsymbol{T}}_{k+1} = \left[\begin{array}{c|c} \widetilde{\boldsymbol{T}}_k & \widetilde{\boldsymbol{t}}_{k+1} \\ \hline \boldsymbol{0} & \beta_{k+1} \end{array}\right], \quad \widetilde{\boldsymbol{t}}_{k+1} = (0, \cdots, 0, \beta_k, \alpha_{k+1})^{\mathrm{T}},$$

于是, 有

$$\boldsymbol{R}_{k+1} = \left[\begin{array}{c|c} \boldsymbol{R}_k & \widetilde{\boldsymbol{r}}_{k+1} \\ \hline \boldsymbol{0} & \gamma_{k+1} \end{array}\right], \quad \widetilde{\boldsymbol{r}}_{k+1} = (0, \cdots, 0, \varepsilon_{k-1}, \delta_k)^{\mathrm{T}}, \tag{4.112}$$

式中:

$$\begin{aligned} &\varepsilon_{k-1} = s_{k-1}\beta_k, \quad \widetilde{\beta}_k = c_{k-1}\beta_k, \\ &\delta_k = c_k\widetilde{\beta}_k + s_k\alpha_{k+1}, \quad \widetilde{\alpha}_{k+1} = -s_k\widetilde{\beta}_k + c_k\alpha_{k+1}, \end{aligned} \tag{4.113}$$

上面的四个等式由如下 Givens 变换得到:

$$\widetilde{\boldsymbol{T}}_{k+1} = \left[\begin{array}{ccccccc:c} \alpha_1 & \beta_1 & & & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & \beta_{k-3} & \alpha_{k-2} & \beta_{k-2} & & & \\ & & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} & & \\ & & & & \beta_{k-1} & \alpha_k & \beta_k & \\ & & & & & \beta_k & \alpha_{k+1} & \\ \hdashline & & & & & & & \beta_{k+1} \end{array}\right] \xrightarrow{\boldsymbol{G}_1} \left[\begin{array}{ccccccc:c} \gamma_1 & \delta_1 & \varepsilon_1 & & & & & \\ 0 & \widetilde{\alpha}_2 & \widetilde{\beta}_2 & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & \beta_{k-3} & \alpha_{k-2} & \beta_{k-2} & & & \\ & & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} & & \\ & & & & \beta_{k-1} & \alpha_k & \beta_k & \\ & & & & & \beta_k & \alpha_{k+1} & \\ \hdashline & & & & & & & \beta_{k+1} \end{array}\right]$$

$$\xrightarrow{\boldsymbol{G}_2}\cdots\xrightarrow{\boldsymbol{G}_{k-2}}\left[\begin{array}{ccccccc:c}
\gamma_1 & \delta_1 & \varepsilon_1 & & & & & \\
 & \gamma_2 & \delta_2 & \varepsilon_2 & & & & \\
 & & \ddots & \ddots & \ddots & & & \\
 & & & \gamma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} & & \\
 & & & & \widetilde{\alpha}_{k-1} & \widetilde{\beta}_{k-1} & & \\
 & & & & \beta_{k-1} & \alpha_k & \beta_k & \\
 & & & & & \beta_k & \alpha_{k+1} & \\
\hdashline
 & & & & & & \beta_{k+1} &
\end{array}\right]$$

$$\xrightarrow{\boldsymbol{G}_{k-1}}\left[\begin{array}{ccccccc:c}
\gamma_1 & \delta_1 & \varepsilon_1 & & & & & \\
 & \gamma_2 & \delta_2 & \varepsilon_2 & & & & \\
 & & \ddots & \ddots & \ddots & & & \\
 & & & \gamma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} & & \\
 & & & & \gamma_{k-1} & \delta_{k-1} & \varepsilon_{k-1} & \\
 & & & & & \widetilde{\alpha}_k & \widetilde{\beta}_k & \\
 & & & & & \beta_k & \alpha_{k+1} & \\
\hdashline
 & & & & & & \beta_{k+1} &
\end{array}\right]\xrightarrow{\boldsymbol{G}_k}\left[\begin{array}{ccccccc:c}
\gamma_1 & \delta_1 & \varepsilon_1 & & & & & \\
 & \gamma_2 & \delta_2 & \varepsilon_2 & & & & \\
 & & \ddots & \ddots & \ddots & & & \\
 & & & \gamma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} & & \\
 & & & & \gamma_{k-1} & \delta_{k-1} & \varepsilon_{k-1} & \\
 & & & & & \gamma_k & \delta_k & \\
 & & & & & & \widetilde{\alpha}_{k+1} & \\
\hdashline
 & & & & & & \beta_{k+1} &
\end{array}\right],$$

即由

$$\begin{bmatrix} c_{k-1} & s_{k-1} \\ -s_{k-1} & c_{k-1} \end{bmatrix} \begin{bmatrix} 0 \\ \beta_k \end{bmatrix} = \begin{bmatrix} \varepsilon_{k-1} \\ \widetilde{\beta}_k \end{bmatrix},$$

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \widetilde{\beta}_k \\ \alpha_{k+1} \end{bmatrix} = \begin{bmatrix} \delta_k \\ \widetilde{\alpha}_{k+1} \end{bmatrix}$$

得到. 而 γ_{k+1} 是在确定第 $k+1$ 个 Givens 变换 $\boldsymbol{G}_{k+1}$ 时得到的, 即计算 c_{k+1} 和 s_{k+1}, 使得

$$\begin{bmatrix} c_{k+1} & s_{k+1} \\ -s_{k+1} & c_{k+1} \end{bmatrix} \begin{bmatrix} \widetilde{\alpha}_{k+1} \\ \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \gamma_{k+1} \\ 0 \end{bmatrix}. \tag{4.114}$$

由式 (4.106) 可知 $\boldsymbol{t}_{k+1} = (\boldsymbol{t}_k^{\mathrm{T}}, \tau_{k+1})^{\mathrm{T}}$, 其中

$$\tau_{k+1} = (-1)^k \beta s_1 s_2 \cdots s_k c_{k+1} = \rho_k c_{k+1}, \tag{4.115}$$

而

$$\rho_{k+1} = (-1)^{k+1}\beta s_1 s_2 \cdots s_k s_{k+1} = -\rho_k s_{k+1}. \tag{4.116}$$

由式 (4.111) 和式 (4.112), 有

$$\boldsymbol{P}_{k+1} = \boldsymbol{V}_{k+1}\boldsymbol{R}_{k+1}^{-1} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] \left[\begin{array}{c|c} \boldsymbol{R}_k^{-1} & \boldsymbol{s}_{k+1} \\ \hline \boldsymbol{0} & \gamma_{k+1}^{-1} \end{array}\right] = [\boldsymbol{P}_k, \boldsymbol{p}_{k+1}], \tag{4.117}$$

其中

$$\boldsymbol{p}_{k+1} = (\boldsymbol{v}_{k+1} - \varepsilon_{k-1}\boldsymbol{p}_{k-1} - \delta_k \boldsymbol{p}_k)/\gamma_{k+1}. \tag{4.118}$$

这是由于

$$\begin{aligned} \boldsymbol{I}_{k+1} = \boldsymbol{R}_{k+1}\boldsymbol{R}_{k+1}^{-1} &= \left[\begin{array}{c|c} \boldsymbol{R}_k & \widetilde{\boldsymbol{r}}_{k+1} \\ \hline \boldsymbol{0} & \gamma_{k+1} \end{array}\right] \left[\begin{array}{c|c} \boldsymbol{R}_k^{-1} & \boldsymbol{s}_{k+1} \\ \hline \boldsymbol{0} & \gamma_{k+1}^{-1} \end{array}\right] \\ &= \left[\begin{array}{c|c} \boldsymbol{I}_k & \boldsymbol{R}_k\boldsymbol{s}_{k+1} + \gamma_{k+1}^{-1}\widetilde{\boldsymbol{r}}_{k+1} \\ \hline \boldsymbol{0} & 1 \end{array}\right], \end{aligned}$$

由此可得

$$\boldsymbol{R}_k\boldsymbol{s}_{k+1}+\gamma_{k+1}^{-1}\widetilde{\boldsymbol{r}}_{k+1}=\boldsymbol{0}\Longrightarrow\boldsymbol{s}_{k+1}=-\gamma_{k+1}^{-1}\boldsymbol{R}_k^{-1}\widetilde{\boldsymbol{r}}_{k+1}.$$

于是

$$\begin{aligned}\boldsymbol{p}_{k+1}&=\boldsymbol{V}_k\boldsymbol{s}_{k+1}+\gamma_{k+1}^{-1}\boldsymbol{v}_{k+1}=-\gamma_{k+1}^{-1}\boldsymbol{V}_k\boldsymbol{R}_k^{-1}\widetilde{\boldsymbol{r}}_{k+1}+\gamma_{k+1}^{-1}\boldsymbol{v}_{k+1}\\&=\big(\boldsymbol{v}_{k+1}-\boldsymbol{P}_k\widetilde{\boldsymbol{r}}_{k+1}\big)/\gamma_{k+1}=(\boldsymbol{v}_{k+1}-\varepsilon_{k-1}\boldsymbol{p}_{k-1}-\delta_k\boldsymbol{p}_k)/\gamma_{k+1}.\end{aligned}$$

从而有

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_0+\boldsymbol{P}_{k+1}\boldsymbol{t}_{k+1}=\boldsymbol{x}_0+[\boldsymbol{P}_k,\boldsymbol{p}_{k+1}]\begin{bmatrix}\boldsymbol{t}_k\\ \tau_{k+1}\end{bmatrix}=\boldsymbol{x}_k+\tau_{k+1}\boldsymbol{p}_{k+1}.\tag{4.119}$$

这就有人们希望得到的 $\boldsymbol{x}_k$ 的递推公式. 这样, 每次迭代只需保存 $\boldsymbol{x}_k,\boldsymbol{p}_{k-1},\boldsymbol{p}_k,\boldsymbol{v}_k,\boldsymbol{v}_{k+1}$ 这五个向量即可.

综述上面的讨论, 就得到了如下的极小残量方法.

算法 4.9 (MINRES 方法) 给定 n 阶非奇异的实对称矩阵 $\boldsymbol{A}$, n 维向量 $\boldsymbol{b}$ 和允许误差 $\varepsilon > 0$. 本算法计算近似解向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\beta = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta$;

$\alpha_1 = \boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_1$; $\boldsymbol{u} = \boldsymbol{A}\boldsymbol{v}_1 - \boldsymbol{\alpha}_1\boldsymbol{v}_1$; $\beta_1 = \|\boldsymbol{u}\|_2$;

if $\beta_1 = 0$

$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \boldsymbol{r}_0/\alpha_1$; 结束

else

$\boldsymbol{v}_2 = \boldsymbol{u}/\beta_1$;

end

$c_0 = 1$; $s_0 = 0$; $\boldsymbol{p}_0 = \boldsymbol{0}$;

确定 $c_1 = \cos\theta_1$ 和 $s_1 = \sin\theta_1$, 使得

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ 0 \end{bmatrix};$$

$\boldsymbol{p}_1 = \boldsymbol{v}_1/\gamma_1;\ \rho_1 = -\beta s_1;\ \tau_1 = \beta c_1;$

$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \tau_1 \boldsymbol{p}_1;\ k = 1;$

while $(|\rho_k|/\beta > \varepsilon)$

$\alpha_{k+1} = \boldsymbol{v}_{k+1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{v}_{k+1};$

$\boldsymbol{u} = \boldsymbol{A}\boldsymbol{v}_{k+1} - \alpha_{k+1}\boldsymbol{v}_{k+1} - \beta_k \boldsymbol{v}_k;$

$\beta_{k+1} = \|\boldsymbol{u}\|_2;$

if $\beta_{k+1} \neq 0$

$\boldsymbol{v}_{k+2} = \boldsymbol{u}/\beta_{k+1};$

end

$\varepsilon_{k-1} = s_{k-1}\beta_k;\ \widetilde{\beta}_k = c_{k-1}\beta_k;$

$\delta_k = c_k \widetilde{\beta}_k + s_k \alpha_{k+1};\ \widetilde{\alpha}_{k+1} = -s_k \widetilde{\beta}_k + c_k \alpha_{k+1};$

Back

Close

确定 $c_{k+1} = \cos\theta_{k+1}$和 $s_{k+1} = \sin\theta_{k+1}$, 使得

$$\begin{bmatrix} c_{k+1} & s_{k+1} \\ -s_{k+1} & c_{k+1} \end{bmatrix} \begin{bmatrix} \widetilde{\alpha}_{k+1} \\ \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \gamma_{k+1} \\ 0 \end{bmatrix};$$

$\tau_{k+1} = \rho_k c_{k+1}$; $\rho_{k+1} = -\rho_k s_{k+1}$;

$\boldsymbol{p}_{k+1} = (\boldsymbol{v}_{k+1} - \varepsilon_{k-1}\boldsymbol{p}_{k-1} - \delta_k \boldsymbol{p}_k)/\gamma_{k+1}$;

$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \tau_{k+1}\boldsymbol{p}_{k+1}$;

$k = k + 1$;

end

MINRES 方法的 MATLAB 程序如下:

```
%MINRES方法的程序-mminres.m
function [x,iter,time,res,resvec]=mminres(A,b,x,max_it,tol)
%极小残量法求解对称不定方程组Ax=b
tic; r=b-A*x; b=norm(r); v=r/b; bt=b;
```

```
z=A*v; a=v'*z; v=z-a*v; b1=norm(v);
if (b1~=0), v1=v/b1; end
c0=1; s0=0; p0=zeros(length(b),1);
[c,s,gama]=givens(a,b1);  %Givens变换
p=v/gama; rho=-b*s; tau=b*c;
x=x+tau*p; iter=1;
while (iter<max_it)
    res=abs(rho)/bt; resvec(iter)=res;
    if (res<tol), break; end
    z=A*v1; a=v1'*z; v=z-a*v1-b1*v; b2=norm(v);
    if (b2~=0), v2=v/b2; end
    epsi=s0*b1; bh1=c0*b1;
    dta=c*bh1+s*a; ah=-s*bh1+c*a;
```

```
    [c1,s1,gama]=givens(ah,b2);  %Givens变换
    tau=rho*c1; rho=-rho*s1;
    p1=(v1-epsi*p0-dta*p)/gama;
    x=x+tau*p1; b1=b2; iter=iter+1;
    v=v1; v1=v2; p0=p; p=p1;
    c0=c; c=c1; s0=s; s=s1;
end
time=toc;
```

例 4.8 假设线性方程组 $\boldsymbol{Ax} = \boldsymbol{b}$, 其中矩阵 $\boldsymbol{A}$ 来自

http://www.cise.ufl.edu./research/sparse/matrices/PARSEC/SiNa.html,

它是一个 5743 阶的实对称不定稀疏矩阵, 右端项 $\boldsymbol{b}$ 为

$$\boldsymbol{b} = \boldsymbol{A} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^n.$$

显然, 该方程组的真解为 $\boldsymbol{x}^* = (1, 1, \cdots, 1)^{\mathrm{T}}$. 应用算法 4.9 到该线性方程组上, 第 224 步得到的 $\widehat{\boldsymbol{x}} = \boldsymbol{x}_{224}$ 满足

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 5.9139 \times 10^{-8},$$

迭代过程的收敛轨迹如图 4.9 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

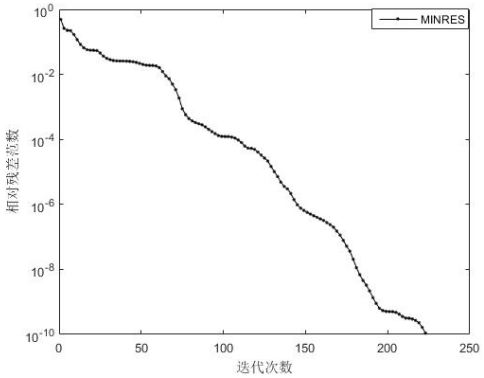


图 4.9 MINRES 方法的收敛特性

§4.3.2 PMINRES 方法

MINRES 方法 (算法 4.9) 与 4.1 节的共轭梯度法在形式上很不相同, 且不便于导出相应的预处理算法. 下面从另一个途径推导出与共轭梯度法形式类似的 MINRES 方法, 并由此导出预处理极小残量法 (简记为 PMINRES 方法) 的递推算法. 由前面的论述可知, 极小残量法就是求向量 $\boldsymbol{z}_k$ 及 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{z}_k$ 使得 $\|\boldsymbol{r}_k\|_2^2$ 达到

极小, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$. 由于 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$, 因此, 若记 $\boldsymbol{S}_k = \left[\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\right]$, 则 $\boldsymbol{x}_k$ 也可表示成

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{S}_k \boldsymbol{z}_k.$$

于是问题等价于求 $\boldsymbol{z}_k$ 使 $\|\boldsymbol{r}_k\|_2^2$ 达到极小. 注意到

$$\begin{aligned} \|\boldsymbol{r}_k\|_2^2 &= \big(\boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_k), \boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_k)\big) \\ &= \big(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{S}_k\boldsymbol{z}_k), \boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{S}_k\boldsymbol{z}_k\big). \end{aligned} \tag{4.120}$$

因为 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{A}^2$ 是对称正定矩阵, 因此可以定义一种新内积

$$(\boldsymbol{x}, \boldsymbol{z})_{\boldsymbol{A}^2} = \big(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}, \boldsymbol{z}\big),$$

因此

$$\|\boldsymbol{r}_k\|_2^2 = (\boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{S}_k\boldsymbol{z}_k, \boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{S}_k\boldsymbol{z}_k)_{\boldsymbol{A}^2}.$$




按照极小与正交关系定理, 可得 z_k 当且仅当满足条件 (即 $r_k \perp \mathcal{K}_k(A, r_0)$)

$$\left(x^* - x_0 - S_k z_k, A^i r_0\right)_{A^2} = 0, \quad i = 0, 1, \cdots, k-1.$$

若将 Krylov 子空间 $\mathrm{span}\{r_0, Ar_0, \cdots, A^{k-1}r_0\}$ 按内积 $(\cdot,\cdot)_{A^2}$ 标准正交化得 $p_0, p_1, \cdots, p_{k-1}$, 则有

$$P_k = \left[p_0, p_1, \cdots, p_{k-1}\right] = S_k R_k,$$

式中: R_k 为上三角矩阵, 它在 $r_0, Ar_0, \cdots, A^{k-1}r_0$ 线性无关的假设下是非奇异的, 且在对角元确定后是唯一确定的. 于是有

$$S_k z_k = P_k R_k^{-1} z_k.$$

记 $\alpha_k := R_k^{-1} z_k$, 则

$$x_k = x_0 + P_k \alpha_k.$$

注意到

$$\boldsymbol{P}_k\boldsymbol{\alpha}_k = \sum_{j=0}^{k-1} \alpha_j^{(k)} \boldsymbol{p}_j.$$

由

$$\left(\boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{P}_k\boldsymbol{\alpha}_k, \boldsymbol{p}_i\right)_{\boldsymbol{A}^2} = 0, \ i = 0, 1, \cdots, k-1$$

及 $\boldsymbol{p}_i$ 的 $(\cdot,\cdot)_{\boldsymbol{A}^2}$ 正交性, 得

$$\begin{aligned}\alpha_i^{(k)} &= \frac{(\boldsymbol{x}^* - \boldsymbol{x}_0, \boldsymbol{p}_i)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_i, \boldsymbol{p}_i)_{\boldsymbol{A}^2}} = \frac{(\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{A}\boldsymbol{x}_0, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)} \\ &= \frac{(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)} = \frac{(\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)}.\end{aligned}$$

注意到 $\alpha_i^{(k)}$ 计算公式与 k 无关, 于是可以得到简单的递推公式, 即

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{\alpha}_k = \boldsymbol{x}_0 + \sum_{i=0}^{k-1}\alpha_i\boldsymbol{p}_i$$

$$= \boldsymbol{x}_0 + \sum_{i=0}^{k-2}\alpha_i\boldsymbol{p}_i + \alpha_{k-1}\boldsymbol{p}_{k-1}$$

$$= \boldsymbol{x}_{k-1} + \alpha_{k-1}\boldsymbol{p}_{k-1}.$$

现在剩下的问题是如何求得标准正交向量组 $\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k-1}$. 按照共轭梯度法的思想, 取

$$\boldsymbol{p}_0 = \boldsymbol{r}_0, \quad \boldsymbol{p}_1 = \boldsymbol{A}\boldsymbol{p}_0 + \gamma_0^{(1)}\boldsymbol{p}_0.$$

要使 $(\boldsymbol{p}_0, \boldsymbol{p}_1)_{\boldsymbol{A}^2} = 0$, 必须

$$\gamma_0^{(1)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0)}{(\boldsymbol{A}\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0)}.$$

然后选取

$$\boldsymbol{p}_2 = \boldsymbol{A}\boldsymbol{p}_1 + \gamma_1^{(2)}\boldsymbol{p}_1 + \gamma_0^{(2)}\boldsymbol{p}_0.$$




Back
Close

利用 $\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2$ 的正交性, 分别用 $\boldsymbol{p}_1, \boldsymbol{p}_0$ 与上式两边作 $(\cdot,\cdot)_{\boldsymbol{A}^2}$ 内积, 得

$$\gamma_1^{(2)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_1, \boldsymbol{p}_1)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_1, \boldsymbol{p}_1)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_1, \boldsymbol{A}\boldsymbol{p}_1)}{(\boldsymbol{A}\boldsymbol{p}_1, \boldsymbol{A}\boldsymbol{p}_1)},$$
$$\gamma_0^{(2)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_1, \boldsymbol{p}_0)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_1, \boldsymbol{A}\boldsymbol{p}_0)}{(\boldsymbol{A}\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0)}.$$

同理, 令

$$\boldsymbol{p}_3 = \boldsymbol{A}\boldsymbol{p}_2 + \gamma_2^{(3)}\boldsymbol{p}_2 + \gamma_1^{(3)}\boldsymbol{p}_1 + \gamma_0^{(3)}\boldsymbol{p}_0.$$

分别用 $\boldsymbol{p}_2, \boldsymbol{p}_1, \boldsymbol{p}_0$ 与上式两边作 $(\cdot,\cdot)_{\boldsymbol{A}^2}$ 内积得

$$\gamma_2^{(3)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_2, \boldsymbol{p}_2)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_2, \boldsymbol{p}_2)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_2, \boldsymbol{A}\boldsymbol{p}_2)}{(\boldsymbol{A}\boldsymbol{p}_2, \boldsymbol{A}\boldsymbol{p}_2)},$$
$$\gamma_1^{(3)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_2, \boldsymbol{p}_1)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_1, \boldsymbol{p}_1)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_2, \boldsymbol{A}\boldsymbol{p}_1)}{(\boldsymbol{A}\boldsymbol{p}_1, \boldsymbol{A}\boldsymbol{p}_1)},$$

$$\gamma_0^{(3)} = -\frac{(\boldsymbol{A}\boldsymbol{p}_2, \boldsymbol{p}_0)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}} = -\frac{(\boldsymbol{p}_2, \boldsymbol{A}\boldsymbol{p}_0)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}}$$
$$= -\frac{(\boldsymbol{p}_2, \boldsymbol{p}_1 - \gamma_0^{(1)}\boldsymbol{p}_0)_{\boldsymbol{A}^2}}{(\boldsymbol{p}_0, \boldsymbol{p}_0)_{\boldsymbol{A}^2}} = 0.$$

依此类推, 有

$$\boldsymbol{p}_{i+1} = \boldsymbol{A}\boldsymbol{p}_i + \lambda_i \boldsymbol{p}_i + \mu_i \boldsymbol{p}_{i-1},$$
$$\lambda_i = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)}, \quad \mu_i = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})},$$
$$\mu_0 = 0, \ i = 0, 1, 2, \cdots .$$

综合上述, 即可得到形式与共轭梯度法类似的极小残量递推算法:




Back

Close

取 $\boldsymbol{p}_0 = \boldsymbol{r}_0$, $\mu_0 = 0$, 对 $k = 0, 1, 2, \cdots$ 进行下列计算直到满足终止条件:

$$\alpha_k = \frac{(\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_k)}{(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}, \quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k; \tag{4.121}$$

$$\lambda_k = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}{(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}, \quad \mu_k = -\frac{(\boldsymbol{A}^2\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_{k-1})}{(\boldsymbol{A}\boldsymbol{p}_{k-1}, \boldsymbol{A}\boldsymbol{p}_{k-1})},$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{A}\boldsymbol{p}_k + \lambda_k \boldsymbol{p}_k + \mu_k \boldsymbol{p}_{k-1}. \tag{4.122}$$

上述算法每次迭代需要作两次矩阵与向量乘法 $(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}(\boldsymbol{A}\boldsymbol{p}_k))$, 要存储四个向量 $\boldsymbol{x}_k, \boldsymbol{p}_k, \boldsymbol{p}_{k-1}, \boldsymbol{A}\boldsymbol{p}_k$, 计算量和存储量均为共轭梯度法的两倍.

回顾共轭梯度法中, 在求 $\boldsymbol{p}_{k+1}$ 时利用了 $\boldsymbol{r}_{k+1}$ 代替 $\boldsymbol{A}\boldsymbol{p}_k$ 来减少计算量和存储量. 在此, 也来分析一下 $\boldsymbol{r}_{k+1}$. 由

$$\left(\boldsymbol{x}^* - \boldsymbol{x}_0 - \boldsymbol{P}_k \boldsymbol{\alpha}_k, \boldsymbol{p}_i\right)_{\boldsymbol{A}^2} = 0, \quad i = 0, 1, \cdots, k-1,$$

可知

$$\left(\boldsymbol{x}^* - \boldsymbol{x}_k, \boldsymbol{p}_i\right)_{\boldsymbol{A}^2} = 0, \quad i = 0, 1, \cdots, k-1.$$

上式即

$$(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_i) = (\boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_k), \boldsymbol{A}\boldsymbol{p}_i) = 0, \quad i = 0, 1, \cdots, k-1. \quad (4.123)$$

但

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k, \quad (4.124)$$

即

$$\boldsymbol{r}_k = \boldsymbol{r}_{k+1} + \alpha_k \boldsymbol{A}\boldsymbol{p}_k,$$

故有

$$\|\boldsymbol{r}_k\|_2^2 = \|\boldsymbol{r}_{k+1}\|_2^2 + \alpha_k^2 \|\boldsymbol{A}\boldsymbol{p}_k\|_2^2.$$

因此, 只要 $\alpha_k \|\boldsymbol{A}\boldsymbol{p}_k\|_2 \neq 0$, 就必有

$$\|\boldsymbol{r}_{k+1}\|_2 < \|\boldsymbol{r}_k\|_2.$$

由式 (4.122), 有

$$\begin{aligned}\boldsymbol{p}_i &= \boldsymbol{A}\boldsymbol{p}_{i-1} + \lambda_{i-1}\boldsymbol{p}_{i-1} + \mu_{i-1}\boldsymbol{p}_{i-2} = \cdots \\ &= \boldsymbol{A}^i\boldsymbol{p}_0 + \sum_{s=0}^{i-1}\delta_s^{(i)}\boldsymbol{A}^s\boldsymbol{p}_0 = \boldsymbol{A}^i\boldsymbol{r}_0 + \sum_{s=0}^{i-1}\delta_s^{(i)}\boldsymbol{A}^s\boldsymbol{r}_0,\end{aligned}$$

得

$$\begin{aligned}\boldsymbol{r}_{i+1} &= \boldsymbol{r}_i - \alpha_i\boldsymbol{A}\boldsymbol{p}_i = \boldsymbol{r}_i - \alpha_i\boldsymbol{A}\Big(\boldsymbol{A}^i\boldsymbol{r}_0 + \sum_{s=0}^{i-1}\delta_s^{(i)}\boldsymbol{A}^s\boldsymbol{r}_0\Big) \\ &= \boldsymbol{r}_i - \alpha_i\sum_{s=0}^{i-1}\delta_s^{(i)}\boldsymbol{A}^{s+1}\boldsymbol{r}_0 - \alpha_i\boldsymbol{A}^{i+1}\boldsymbol{r}_0 \\ &= \cdots = \mathcal{P}_i(\boldsymbol{A})\boldsymbol{r}_0 - \alpha_i\boldsymbol{A}^{i+1}\boldsymbol{r}_0,\end{aligned}$$

式中: $\mathcal{P}_i(\boldsymbol{A})$ 为关于矩阵 $\boldsymbol{A}$ 的 i 次多项式. 由上式, 得

$$\big[\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}\big] = \big[\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\big]\widetilde{\boldsymbol{R}}_k = \boldsymbol{S}_k\widetilde{\boldsymbol{R}}_k, \quad (4.125)$$

式中: $\widetilde{\boldsymbol{R}}_k$ 为 k 阶上三角矩阵, 且其对角元为 $1, -\alpha_0, -\alpha_1, \cdots, -\alpha_{k-2}$.




Back
Close

若构造关于内积 $(\cdot,\cdot)_{A^2}$ 的正交化向量组 $\widetilde{\boldsymbol{p}}_0, \widetilde{\boldsymbol{p}}_1, \cdots, \widetilde{\boldsymbol{p}}_{k-1}$:

$$
\begin{aligned}
\widetilde{\boldsymbol{p}}_0 &= \boldsymbol{r}_0, \\
\widetilde{\boldsymbol{p}}_1 &= \boldsymbol{r}_1 + \gamma_0^{(1)}\widetilde{\boldsymbol{p}}_0, \\
\widetilde{\boldsymbol{p}}_2 &= \boldsymbol{r}_2 + \gamma_1^{(2)}\widetilde{\boldsymbol{p}}_1 + \gamma_0^{(2)}\widetilde{\boldsymbol{p}}_0, \\
&\vdots \\
\widetilde{\boldsymbol{p}}_i &= \boldsymbol{r}_i + \sum_{s=0}^{i-1}\gamma_s^{(i)}\widetilde{\boldsymbol{p}}_s = \boldsymbol{r}_i + \gamma_{i-1}^{(i)}\widetilde{\boldsymbol{p}}_{i-1} + \cdots + \gamma_1^{(i)}\widetilde{\boldsymbol{p}}_1 + \gamma_0^{(i)}\widetilde{\boldsymbol{p}}_0,
\end{aligned}
$$

式中:

$$
\gamma_s^{(i)} = -\frac{\left(\boldsymbol{r}_i, \widetilde{\boldsymbol{p}}_s\right)_{A^2}}{\left(\widetilde{\boldsymbol{p}}_s, \widetilde{\boldsymbol{p}}_s\right)_{A^2}}, \quad i = 0, 1, \cdots, k-1; \ \ s = 0, 1, \cdots, i-1.
$$

不失一般性, 设 $\widetilde{\boldsymbol{p}}_i = \omega_i \boldsymbol{p}_i$, 利用 $\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{A}$, 有

$$\begin{aligned}\left(\boldsymbol{r}_i, \widetilde{\boldsymbol{p}}_s\right)_{\boldsymbol{A}^2} &= \left(\boldsymbol{A}\boldsymbol{r}_i, \boldsymbol{A}\widetilde{\boldsymbol{p}}_s\right) = \left(\boldsymbol{A}\boldsymbol{r}_i, \omega_s \boldsymbol{A}\boldsymbol{p}_s\right) \\ &= \left(\boldsymbol{A}\boldsymbol{r}_i, \omega_s(\boldsymbol{p}_{s+1} - \lambda_s \boldsymbol{p}_s - \mu_s \boldsymbol{p}_{s-1}\right) \\ &= \left(\boldsymbol{r}_i, \omega_s \boldsymbol{A}\boldsymbol{p}_{s+1} - \omega_s \lambda_s \boldsymbol{A}\boldsymbol{p}_s - \omega_s \mu_s \boldsymbol{A}\boldsymbol{p}_{s-1}\right).\end{aligned}$$

由上式及式 (4.123), 不难发现, 当 $s+1 \leqslant i-1$, 即 $s \leqslant i-2$ 时, 有

$$\left(\boldsymbol{r}_i, \widetilde{\boldsymbol{p}}_s\right)_{\boldsymbol{A}^2} = 0,$$

即

$$\gamma_s^{(i)} = 0, \ s \leqslant i-2, \ i = 0, 1, \cdots, k-1.$$

于是有

$$\widetilde{\boldsymbol{p}}_i = \boldsymbol{r}_i + \gamma_{i-1}^{(i)} \widetilde{\boldsymbol{p}}_{i-1}, \ i = 0, 1, \cdots, k-1, \ \widetilde{\boldsymbol{p}}_{-1} = \boldsymbol{0}.$$




回顾式 (4.122), $\boldsymbol{p}_i$ 与 $\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-2}$ 有关, 可以看到, $\widetilde{\boldsymbol{p}}_i$ 只与 $\boldsymbol{r}_i$ 和 $\widetilde{\boldsymbol{p}}_{i-1}$ 有关. 现记 $\beta_{i-1} := \gamma_{i-1}^{(i)}$, 在不混淆的情况下, 再记 $\boldsymbol{p}_i := \widetilde{\boldsymbol{p}}_i$, 则有

$$\begin{aligned}
&\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_{i-1}\boldsymbol{p}_{i-1}, \ i = 1, 2, \cdots, k-1, \\
&\beta_{i-1} = \frac{(\boldsymbol{r}_i, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}}{(\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}}.
\end{aligned}$$

注意到, 对这一组 $\widetilde{\boldsymbol{p}}_0, \widetilde{\boldsymbol{p}}_2, \cdots, \widetilde{\boldsymbol{p}}_{k-1}$ (已记为 $\boldsymbol{p}_0, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{k-1}$), 也有

$$\left[\boldsymbol{p}_0, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{k-1}\right] = \left[\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\right]\widehat{\boldsymbol{R}}_k = \boldsymbol{S}_k\widehat{\boldsymbol{R}}_k, \quad (4.126)$$

式中: $\widehat{\boldsymbol{R}}_k$ 为上三角矩阵, 对角元也为 $1, -\alpha_0, -\alpha_1, \cdots, -\alpha_{k-2}$. 由式 (4.125) 和式 (4.126), 得

$$\left[\boldsymbol{r}_0, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_{k-1}\right] = \left[\boldsymbol{p}_0, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{k-1}\right]\widehat{\boldsymbol{R}}_k^{-1}\widetilde{\boldsymbol{R}}_k,$$

式中: $\widehat{\boldsymbol{R}}_k^{-1}\widetilde{\boldsymbol{R}}_k$ 仍为上三角矩阵, 且对角元均为 1. 于是有

$$\boldsymbol{r}_i = \boldsymbol{p}_i + \sum_{s=0}^{i-1} c_s \boldsymbol{p}_s,$$

及

$$\beta_{i-1} = \frac{(\boldsymbol{r}_i, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}}{(\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}} = \frac{(\boldsymbol{A}\boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} = \frac{\left(\boldsymbol{A}\boldsymbol{r}_i, \frac{\boldsymbol{r}_i - \boldsymbol{r}_{i-1}}{\alpha_{i-1}}\right)}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})}.$$

由于

$$(\boldsymbol{A}\boldsymbol{r}_i, \boldsymbol{r}_{i-1}) = \left(\boldsymbol{r}_i, \boldsymbol{A}\left(\boldsymbol{p}_{i-1} + \sum_{s=0}^{i-2} c_s \boldsymbol{p}_s\right)\right) = 0,$$

因此

$$\beta_{i-1} = \frac{(\boldsymbol{A}\boldsymbol{r}_i, \boldsymbol{r}_i)}{\alpha_{i-1}(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})}. \tag{4.127}$$

另外, 由 $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_{i-1}\boldsymbol{p}_{i-1}$ 及式 (4.123), 得

$$\begin{aligned}\alpha_{i-1} &= \frac{(\boldsymbol{x}_i - \boldsymbol{x}_{i-1}, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}}{(\boldsymbol{p}_{i-1}, \boldsymbol{p}_{i-1})_{\boldsymbol{A}^2}} = \frac{(\boldsymbol{A}(\boldsymbol{x}_i - \boldsymbol{x}_{i-1}), \boldsymbol{A}\boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} \\ &= \frac{(\boldsymbol{r}_{i-1} - \boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} = \frac{(\boldsymbol{r}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} \\ &= \frac{(\boldsymbol{A}\boldsymbol{r}_{i-1}, \boldsymbol{p}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} = \frac{(\boldsymbol{A}\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1} + \beta_{i-2}\boldsymbol{p}_{i-2})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})} \\ &= \frac{(\boldsymbol{A}\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1})}{(\boldsymbol{A}\boldsymbol{p}_{i-1}, \boldsymbol{A}\boldsymbol{p}_{i-1})}. \end{aligned} \tag{4.128}$$

将式 (4.128) 代入式 (4.127), 得

$$\beta_{i-1} = \frac{(\boldsymbol{A}\boldsymbol{r}_i, \boldsymbol{r}_i)}{(\boldsymbol{A}\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i-1})}.$$

于是可以得到极小残量法另一种形式的递推算法:

算法 4.10 (递推形式的 MINRES 方法) 给定 n 阶非奇异的实对称矩阵 $\boldsymbol{A}$, n 维向量 $\boldsymbol{b}$ 和允许误差 $\varepsilon > 0$. 本算法计算近似

解向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{r}_0$;

for $k = 0, 1, \cdots,$

$$\alpha_k = \frac{(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k)}{(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}; \quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k;$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k; \quad \beta_k = \frac{(\boldsymbol{A}\boldsymbol{r}_{k+1}, \boldsymbol{r}_{k+1})}{(\boldsymbol{A}\boldsymbol{r}_k, \boldsymbol{r}_k)};$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k; \quad \boldsymbol{A}\boldsymbol{p}_{k+1} = \boldsymbol{A}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{A}\boldsymbol{p}_k;$$

end

从形式上看, 算法 4.10 每一步迭代也需要计算两次矩阵与向量的乘法 $\boldsymbol{A}\boldsymbol{p}_k$ 和 $\boldsymbol{A}\boldsymbol{r}_{k+1}$, 实际上可从 $\boldsymbol{A}\boldsymbol{p}_{k+1} = \boldsymbol{A}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{A}\boldsymbol{p}_k$ 来计算 $\boldsymbol{A}\boldsymbol{p}_{k+1}$, 这样就只需要计算一次矩阵与向量的乘法. 此外, 在计算中需要保存五个向量 $\boldsymbol{x}_k$, $\boldsymbol{p}_k$, $\boldsymbol{r}_k$, $\boldsymbol{A}\boldsymbol{p}_k$ 和 $\boldsymbol{A}\boldsymbol{r}_k$. 递推形式的 MINRES 方法的 MATLAB 程序如下:




Back
Close

```
%递推形式的极小残量法-minres1.m
function [x,iter,time,res,resvec]=minres1(A,b,x,max_it,tol)
%极小残量法求解对称不定方程组Ax=b
tic; r=b-A*x; p=r; mr=norm(r);
u=A*p; z=A*r; iter=1;
while (iter<max_it)
    alpha=(r'*u)/(u'*u);
    x=x+alpha*p; r1=r-alpha*u;
    z1=A*r1; beta=(z1'*r1)/(z'*r);
    p=r1+beta*p; u=z1+beta*u;
    res=norm(r1)/mr; resvec(iter)=res;
    if (res<tol), break; end
    r=r1; z=z1; iter=iter+1;
```




Back
Close

```
end
time=toc;
```

现在考虑预处理的极小残量法. 选择一个适当的对称正定矩阵 $\boldsymbol{M}$, 设 $\boldsymbol{M}$ 有分解 $\boldsymbol{M}=\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$, 则方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 经过预处理后变为

$$\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{x}}=\widetilde{\boldsymbol{b}}, \tag{4.129}$$

式中:

$$\widetilde{\boldsymbol{A}}=\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}},\ \widetilde{\boldsymbol{x}}=\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x},\ \widetilde{\boldsymbol{b}}=\boldsymbol{L}^{-1}\boldsymbol{b}.$$

方程组 (4.129) 的系数矩阵 $\widetilde{\boldsymbol{A}}=\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}$ 仍为对称不定矩阵. 通过 $\boldsymbol{M}$ 和 $\boldsymbol{L}$ 的选取, 可以使得 $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}$ 比 $\boldsymbol{A}$ 有更好的条件数. 现在对方程组 (4.129) 使用极小残量法, 得

$$\widetilde{\boldsymbol{r}}_0 = \widetilde{\boldsymbol{b}} - \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{x}}_0,\ \widetilde{\boldsymbol{p}}_0 = \widetilde{\boldsymbol{r}}_0,$$

$$\alpha_k = \frac{(\widetilde{\boldsymbol{r}}_k, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k)}{(\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k)}, \quad \widetilde{\boldsymbol{x}}_{k+1} = \widetilde{\boldsymbol{x}}_k + \alpha_k\widetilde{\boldsymbol{p}}_k,$$

$$\widetilde{\boldsymbol{r}}_{k+1} = \widetilde{\boldsymbol{r}}_k - \alpha_k\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{p}}_k, \quad \beta_k = \frac{(\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{r}}_{k+1}, \widetilde{\boldsymbol{r}}_{k+1})}{(\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{r}}_k, \widetilde{\boldsymbol{r}}_k)},$$

$$\widetilde{\boldsymbol{p}}_{k+1} = \widetilde{\boldsymbol{r}}_{k+1} + \beta_k\widetilde{\boldsymbol{p}}_k, \quad k = 0, 1, \cdots.$$

为了在公式中使用原来变量, 作变换: $\boldsymbol{x}_k = \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{x}}_k,\ \boldsymbol{p}_k = \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{p}}_k,$ 则 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k = \boldsymbol{L}\widetilde{\boldsymbol{r}}_k.$

$$\alpha_k = \frac{\left(\boldsymbol{L}^{-1}\boldsymbol{r}_k, (\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\boldsymbol{L}^{\mathrm{T}}\boldsymbol{p}_k\right)}{(\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{p}}_k, \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{p}}_k)}$$
$$= \frac{(\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k)}{(\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)} = \frac{\left(\boldsymbol{M}^{-1}\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k\right)}{(\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)},$$

$$\widetilde{\boldsymbol{x}}_{k+1} = \widetilde{\boldsymbol{x}}_k + \alpha_k\widetilde{\boldsymbol{p}}_k \Longrightarrow \boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}_{k+1} = \boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}_k + \alpha_k\boldsymbol{L}^{\mathrm{T}}\boldsymbol{p}_k$$
$$\Longrightarrow \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k,$$

$$\boldsymbol{L}^{-1}\boldsymbol{r}_{k+1} = \boldsymbol{L}^{-1}\boldsymbol{r}_k - \alpha_k(\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\boldsymbol{L}^{\mathrm{T}}\boldsymbol{p}_k$$
$$\Longrightarrow \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{p}_k,$$

$$\beta_k = \frac{((\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\widetilde{\boldsymbol{r}}_{k+1}, \widetilde{\boldsymbol{r}}_{k+1})}{((\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}})\widetilde{\boldsymbol{r}}_k, \widetilde{\boldsymbol{r}}_k)} = \frac{(\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{r}}_{k+1}, \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{r}}_{k+1})}{(\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{r}}_k, \boldsymbol{L}^{-\mathrm{T}}\widetilde{\boldsymbol{r}}_k)}$$
$$= \frac{\left(\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_{k+1}, \boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_{k+1}\right)}{\left(\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_k, \boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_k\right)} = \frac{\left(\boldsymbol{A}(\boldsymbol{M}^{-1}\boldsymbol{r}_{k+1}), \boldsymbol{M}^{-1}\boldsymbol{r}_{k+1}\right)}{\left(\boldsymbol{A}(\boldsymbol{M}^{-1}\boldsymbol{r}_k), \boldsymbol{M}^{-1}\boldsymbol{r}_k\right)},$$

Back

Close

$$\widetilde{\boldsymbol{p}}_{k+1} = \widetilde{\boldsymbol{r}}_{k+1} + \beta_k \widetilde{\boldsymbol{p}}_k \implies \boldsymbol{L}^{\mathrm{T}}\boldsymbol{p}_{k+1} = \boldsymbol{L}^{-1}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{L}^{\mathrm{T}}\boldsymbol{p}_k$$

$$\implies \boldsymbol{p}_{k+1} = \boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k \implies \boldsymbol{p}_{k+1} = \boldsymbol{M}^{-1}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k.$$

这样, 就得到了预处理极小残量法的递推算法, 具体步骤为:

算法 4.11 (PMINRES 方法) 给定 n 阶非奇异的实对称矩阵 $\boldsymbol{A}$, n 维向量 $\boldsymbol{b}$, 允许误差 $\varepsilon > 0$ 和预处理矩阵 $\boldsymbol{M}$ (对称正定). 本算法计算近似解向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0 \in \mathbb{R}^n$; $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{z}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$;

$\boldsymbol{p}_0 = \boldsymbol{z}_0$; $\boldsymbol{u}_0 = \boldsymbol{A}\boldsymbol{p}_0$; $\boldsymbol{w}_0 = \boldsymbol{M}^{-1}\boldsymbol{u}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 > \varepsilon)$

$$\alpha_k = \frac{(\boldsymbol{z}_k, \boldsymbol{u}_k)}{(\boldsymbol{w}_k, \boldsymbol{u}_k)}; \quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k;$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{u}_k; \quad \boldsymbol{z}_{k+1} = \boldsymbol{M}^{-1}\boldsymbol{r}_k;$$

$$\beta_k = \frac{(\boldsymbol{A}\boldsymbol{z}_{k+1}, \boldsymbol{z}_{k+1})}{(\boldsymbol{A}\boldsymbol{z}_k, \boldsymbol{z}_k)}; \quad \boldsymbol{p}_{k+1} = \boldsymbol{z}_{k+1} + \beta_k \boldsymbol{p}_k;$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{A}\boldsymbol{p}_{k+1} = \boldsymbol{A}\boldsymbol{z}_{k+1} + \beta_k \boldsymbol{u}_k; \quad \boldsymbol{w}_{k+1} = \boldsymbol{M}^{-1}\boldsymbol{u}_{k+1};$$

$$k = k + 1;$$

end

注 4.4 算法 4.11 仅与预处理子 $\boldsymbol{M}$ 有关, 而与 $\boldsymbol{L}$ 无关. 当 $\boldsymbol{M}$ 为单位阵时, 它就是没有经过预处理的极小残量法. 与共轭梯度法相比较, 预处理极小残量法增加了计算 $\boldsymbol{z}_{k+1}$ 和 $\boldsymbol{w}_{k+1}$ 的工作量, 这可以通过求解两个系数矩阵都是 $\boldsymbol{M}$ 的方程组来实现, 即每一步需要求解 $\boldsymbol{M}\boldsymbol{z} = \boldsymbol{r}_{k+1}$ 和 $\boldsymbol{M}\boldsymbol{w} = \boldsymbol{A}\boldsymbol{p}_{k+1}$ 来得到 $\boldsymbol{z}_{k+1}$ 和 $\boldsymbol{w}_{k+1}$. 尽管如此, 若预处理子 $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ 是通过某种矩阵分解 (比如不完全 Cholesky 分解) 得到的, 此时 $\boldsymbol{L}$ 是下三角矩阵, 则 $\boldsymbol{M}\boldsymbol{z} = \boldsymbol{r}_{k+1}$

可转化为求解两个三角形方程组 $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{r}_{k+1}$ 和 $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{z} = \boldsymbol{y}$, 后者的运算量要比前者少得多.

PMINRES 方法的 MATLAB 程序如下:

```
%PMINRES方法程序-pminres.m
function [x,iter,time,res,resvec]=
                          pminres1(A,b,x,M1,M2,max_it,tol)
%PMINRES方法求解对称不定方程组Ax=b, 预处理子M=M1*M2
tic; n=length(b);
r=b-A*x; z=M2\(M1\r); p=z; mr=norm(r);
u=A*p; v=u; w=M2\(M1\u); iter=1;
while (iter<max_it)
    alpha=(z'*u)/(w'*u); x=x+alpha*p;
    r=r-alpha*u; z1=M2\(M1\r); v1=A*z1;
```




Back
Close

```
        beta=(v1'*z1)/(v'*z); p=z1+beta*p;
        u=v1+beta*u; w=M2\(M1\u);
        res=norm(r)/mr; resvec(iter)=res;
        if (res<tol), break; end
        z=z1; v=v1; iter=iter+1;
    end
    time=toc;
```

例 4.9 仍考虑例 4.8 中的线性方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$, 取预处理矩阵 $\boldsymbol{M}\approx\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ 为 $\boldsymbol{A}-\mu\boldsymbol{I}$ 的不完全 Cholesky 分解, 其中 $\mu=-1.0$ 是矩阵 $\boldsymbol{A}$ 的最小特征值的一个下界估计. 应用算法 4.11 到该线性方程组上, 第 79 步得到的 $\widehat{\boldsymbol{x}}=\boldsymbol{x}_{79}$ 满足

$$\|\widehat{\boldsymbol{x}}-\boldsymbol{x}^*\|_2=3.2414\times10^{-8}.$$

比较极小残量法和预处理极小残量法的数值表现, 迭代过程的收

Back

Close

敛轨迹如图 4.10 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

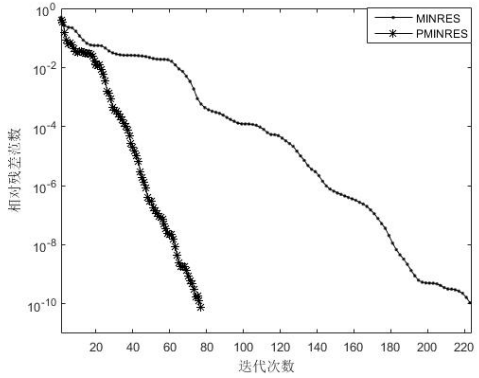


图 4.10 PMINRES 方法与 MINRES 方法的比较

相比于 MINRES 方法, PMINRES 方法尽管迭代次数得到了显著的下降, 从计算时间上看似乎没有优势, 原因可能是后者每一步需要求解两个以预处理子 $\boldsymbol{M}$ 为系数矩阵的线性方程组.

§4.3.3 收敛性分析

与共轭梯度法一样, 虽然从理论上讲极小残量法是一种直接方法, 但实际使用时是把它作为迭代法来使用的. 因此, 下面简要说明其收敛特性.

由 Krylov 子空间的性质, $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的充分必要条件是存在 $\varphi \in \mathcal{P}_{k-1}$ 使得 $\boldsymbol{x} = \boldsymbol{x}_0 + \varphi(\boldsymbol{A})\boldsymbol{r}_0$. 注意到 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, $\boldsymbol{x}^* = \boldsymbol{A}^{-1}\boldsymbol{b}$, 因此, 有

$$\boldsymbol{x}^* - \boldsymbol{x}_k = \boldsymbol{A}^{-1}\boldsymbol{b} - \boldsymbol{x}_0 - \varphi(\boldsymbol{A})\boldsymbol{r}_0 = \boldsymbol{A}^{-1}(\boldsymbol{I} - \boldsymbol{A}\varphi(\boldsymbol{A}))\boldsymbol{r}_0 = \boldsymbol{A}^{-1}\psi(\boldsymbol{A})\boldsymbol{r}_0,$$

式中: $\psi(t) = 1 - t\varphi(t)$ 满足 $\psi(0) = 1$.

设 $\boldsymbol{A}$ 的谱分解为 $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}$, 其中 $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n]$ 是正交矩阵, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$, $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n$. 现将 $\boldsymbol{r}_0$

按 $\boldsymbol{A}$ 的特征向量展开, 有

$$\boldsymbol{r}_0 = \eta_1\boldsymbol{v}_1 + \eta_2\boldsymbol{v}_2 + \cdots + \eta_n\boldsymbol{v}_n = \sum_{i=1}^{n}\eta_i\boldsymbol{v}_i.$$

于是, 由式 (4.120), 有

$$\begin{aligned}
\|\boldsymbol{r}_k\|_2^2 &= \big(\boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_k), \boldsymbol{A}(\boldsymbol{x}^* - \boldsymbol{x}_k)\big) = \big(\psi(\boldsymbol{A})\boldsymbol{r}_0, \psi(\boldsymbol{A})\boldsymbol{r}_0\big) \\
&= \left(\sum_{i=1}^{n}\eta_i\psi(\boldsymbol{A})\boldsymbol{v}_i, \sum_{j=1}^{n}\eta_j\psi(\boldsymbol{A})\boldsymbol{v}_j\right) \\
&= \left(\sum_{i=1}^{n}\eta_i\psi(\lambda_i)\boldsymbol{v}_i, \sum_{j=1}^{n}\eta_r\psi(\lambda_j)\boldsymbol{v}_j\right) \\
&= \sum_{i=1}^{n}\eta_i^2[\psi(\lambda_i)]^2 \leqslant \max_{1\leqslant i\leqslant n}[\psi(\lambda_i)]^2\sum_{i=1}^{n}\eta_i^2 \\
&= \max_{1\leqslant i\leqslant n}[\psi(\lambda_i)]^2\|\boldsymbol{r}_0\|_2^2.
\end{aligned}$$

由此可得

$$\|\boldsymbol{r}_k\|_2 \leqslant \min_{\psi\in\mathcal{P}_k^0} \max_{\lambda\in\lambda(\boldsymbol{A})} |\psi(\lambda)| \cdot \|\boldsymbol{r}_0\|_2, \tag{4.130}$$

式中: 集合 $\mathcal{P}_k^0$ 的定义为

$$\mathcal{P}_k^0 = \{\psi \in \mathcal{P}_k : \psi(0) = 1\}.$$

令

$$a = \min_{\lambda\in\lambda(\boldsymbol{A})} |\lambda|, \quad b = \max_{\lambda\in\lambda(\boldsymbol{A})} |\lambda|, \tag{4.131}$$

即 a 和 b 分别是 $\boldsymbol{A}$ 的特征值的最小模和最大模. 由 $\boldsymbol{A}$ 是非奇异的假定可知 $0 < a \leqslant b$. 下面假定 $a < b$, 当 $k = 2m$ 时, 特别取一个多项式

$$\widehat{\psi}(t) = C_m\left(\frac{b^2 + a^2 - 2t^2}{b^2 - a^2}\right) / C_m\left(\frac{b^2 + a^2}{b^2 - a^2}\right), \tag{4.132}$$

式中: C_m 为 m 次 Chebyshev 多项式, 即

$$C_m(t)=\frac{(t+\sqrt{t^2-1})^m+(t-\sqrt{t^2-1})^m}{2},$$

则 $\widehat{\psi}(t)$ 是一个 $2m$ 次多项式, 且 $\widehat{\psi}(0)=1$, 从而 $\widehat{\psi}\in\mathcal{P}_k^0$. 由 Chebyshev 逼近定理, $\widehat{\psi}(t)$ 在区间 $[-b,-a]\cup[a,b]$ 上均有

$$|\widehat{\psi}(t)|\leqslant 1/C_m\Big(1+\frac{2}{\kappa^2-1}\Big),\quad \forall\, t\in[-b,-a]\cup[a,b].$$

这样, 由式 (4.130) 即得

$$\frac{\|\boldsymbol{r}_{2m}\|_2}{\|\boldsymbol{r}_0\|_2}\leqslant\max_{\lambda\in\lambda(\boldsymbol{A})}|\widehat{\psi}(t)|=1/C_m\Big(\frac{b^2+a^2}{b^2-a^2}\Big)=1/C_m\Big(1+\frac{2}{\kappa^2-1}\Big),$$

式中: $\kappa=\|\boldsymbol{A}\|_2\|\boldsymbol{A}^{-1}\|_2=b/a$. 记 $\theta(\kappa)=1+\dfrac{2}{\kappa^2-1}$, 因 $\|\boldsymbol{r}_{2m+1}\|_2\leqslant\|\boldsymbol{r}_{2m}\|_2$, 故有

$$\frac{\|\boldsymbol{r}_k\|_2}{\|\boldsymbol{r}_0\|_2}\leqslant\frac{1}{C_{[k/2]}(\theta(\kappa))}.\tag{4.133}$$

当然式 (4.133) 的估计是十分粗糙的, 但也说明了极小残量法收敛得快慢与 $\boldsymbol{A}$ 的条件数有关. 事实上, 更精细的分析可以证明, 与共轭梯度法类似, 极小残量法的收敛快慢也与 $\boldsymbol{A}$ 的谱分布有关. 如果 $\boldsymbol{A}$ 的特征值除去少数几个外大多数集中在某数的附近, 则极小残量法将收敛得很快. 因此实际使用时也是将其与预处理技术相结合来使用的.

§4.4 SYMMLQ 方法

SYMMLQ 方法与 MINRES 方法类似, 也是求解对称不定方程组的有效算法之一. 本节介绍这一算法的基本思想和导出过程, 并介绍与之关系密切的极小误差法, 以及这两个方法的收敛性及误差估计定理.

§4.4.1 SYMMLQ 方法

对于对称不定方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4.134}$$

令 $\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{x}_0$, 则式 (4.134) 可改写为

$$\boldsymbol{A}\boldsymbol{z} = \boldsymbol{r}_0, \quad \boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0.$$

若

$$\mathcal{S}_k = \text{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$$

张成一个 k 维子空间. 考虑二次泛函

$$J_k(\boldsymbol{z}_k) = (\boldsymbol{z} - \boldsymbol{z}_k)^{\mathrm{T}} \boldsymbol{A} (\boldsymbol{z} - \boldsymbol{z}_k), \tag{4.135}$$

式中:

$$\boldsymbol{z}_k = \boldsymbol{V}_k \boldsymbol{y}_k, \quad \boldsymbol{V}_k = \big[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\big].$$

那么二次泛函

$$J_k(\boldsymbol{z}_k) = \widetilde{J}_k(\boldsymbol{y}_k) = (\boldsymbol{A}^{-1}\boldsymbol{r}_0 - \boldsymbol{V}_k \boldsymbol{y}_k)^{\mathrm{T}} \boldsymbol{A} (\boldsymbol{A}^{-1}\boldsymbol{r}_0 - \boldsymbol{V}_k \boldsymbol{y}_k)$$

的驻点 $\boldsymbol{y}_k$ 满足

$$\boldsymbol{V}_k^{\mathrm{T}} (\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_k \boldsymbol{y}_k) = \boldsymbol{0},$$

或者

$$\boldsymbol{V}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{V}_k \boldsymbol{y}_k = \boldsymbol{V}_k^{\mathrm{T}} \boldsymbol{r}_0, \quad \boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{z}_k = \boldsymbol{x}_k + \boldsymbol{V}_k \boldsymbol{y}_k. \tag{4.136}$$




Back
Close

若取 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$, 则用对称 Lanczos 算法可得到 k 次 Krylov 子空间

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\}$$

的正交规范基 $\{\boldsymbol{v}_i\}_{i=1}^k$.

假设已经得到了一个长度为 k 的 Lanczos 分解 (2.36):

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k\boldsymbol{T}_k + \beta_{k+1}\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}}, \quad \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k = \boldsymbol{I}_k, \quad \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{v}_{k+1} = \boldsymbol{0}, \quad (4.137)$$

式中: 三对角矩阵 $\boldsymbol{T}_k$ 通常记为

$$\boldsymbol{T}_k \equiv \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-1} & \alpha_{k-1} & \beta_k \\ & & & \beta_k & \alpha_k \end{bmatrix}.$$




Back
Close

将上式代入式 (4.136) 则得到 Galerkin 方程组

$$\boldsymbol{T}_k \boldsymbol{y}_k = \beta_1 \boldsymbol{e}_1, \quad \boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{y}_k, \tag{4.138}$$

式中: $\beta_1 = \|\boldsymbol{r}_0\|_2$.

因为 $\boldsymbol{A}$ 对称不定, 所以三对角矩阵 $\boldsymbol{T}_k$ 也对称不定. 对这样的 $\boldsymbol{T}_k$ 进行 Cholesky 分解可能失效. 因此要寻找 $\boldsymbol{T}_k$ 更为稳定的分解, 即正交三角分解 (LQ 分解):

$$\boldsymbol{T}_k = \widetilde{\boldsymbol{L}}_k \boldsymbol{Q}_k, \ \boldsymbol{Q}_k^{\mathrm{T}} \boldsymbol{Q}_k = \boldsymbol{I}_k, \tag{4.139}$$

式中: $\widetilde{\boldsymbol{L}}_k$ 为下三角矩阵.

利用 LQ 分解式 (4.139) 来求解式 (4.138). 和前面的做法一样, 不是直接求出 $\boldsymbol{y}_k$, 而是寻找求 $\boldsymbol{x}_k$ 递推公式.

因 $\boldsymbol{T}_k$ 是对称三对角矩阵, $\boldsymbol{Q}_k$ 可写为乘积

$$\boldsymbol{Q}_k = \boldsymbol{G}_{k-1,k} \cdots \boldsymbol{G}_{2,3} \boldsymbol{G}_{1,2}, \tag{4.140}$$

式中: $\boldsymbol{G}_{i,i+1}\,(i=1,2,\cdots,k-1)$ 为 $(i,i+1)$ Givens 矩阵, 它与单位矩阵不同的是其 i 和 $i+1$ 行和列组成的 2 阶矩阵是

$$\boldsymbol{G}_{i,i+1}=\begin{bmatrix} c_i & s_i \\ s_i & -c_i \end{bmatrix},\quad c_i^2+s_i^2=1.$$

将式 (4.139) 代入式 (4.138), 得

$$\widetilde{\boldsymbol{L}}_k(\boldsymbol{Q}_k\boldsymbol{y}_k)=\beta_1\boldsymbol{e}_1,\ \boldsymbol{x}_k=\boldsymbol{x}_0+\boldsymbol{V}_k\boldsymbol{Q}_k^{\mathrm{T}}(\boldsymbol{Q}_k\boldsymbol{y}_k).$$

令

$$\widetilde{\boldsymbol{W}}_k=\boldsymbol{V}_k\boldsymbol{Q}_k^{\mathrm{T}},\ \widetilde{\boldsymbol{z}}_k=\boldsymbol{Q}_k\boldsymbol{y}_k, \tag{4.141}$$

则得到方程组和迭代向量公式

$$\widetilde{\boldsymbol{L}}_k\widetilde{\boldsymbol{z}}_k=\beta_1\boldsymbol{e}_1,\ \boldsymbol{x}_k=\boldsymbol{x}_0+\widetilde{\boldsymbol{W}}_k\widetilde{\boldsymbol{z}}_k. \tag{4.142}$$

由 $\boldsymbol{Q}_k$ 的特殊形式 (4.140), $\widetilde{\boldsymbol{W}}_k$ 和 $\widetilde{\boldsymbol{z}}_k$ 可表示为

$$\widetilde{\boldsymbol{W}}_k=\left[\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_{k-1},\widetilde{\boldsymbol{w}}_k\right], \tag{4.143}$$

$$\widetilde{\boldsymbol{z}}_k = \left[\zeta_1, \zeta_2, \cdots, \zeta_{k-1}, \widetilde{\zeta}_k\right]^{\mathrm{T}}. \tag{4.144}$$

而 LQ 分解式 (4.139) 中的 $\widetilde{\boldsymbol{L}}_k$ 和 $\boldsymbol{G}_{k,k+1}$ 的元素 c_k 和 s_k 可计算如下.

由式 (4.139), 公式

$$\widetilde{\boldsymbol{L}}_k = \boldsymbol{T}_k \boldsymbol{Q}_k^{\mathrm{T}} = \boldsymbol{T}_k \boldsymbol{G}_{1,2} \boldsymbol{G}_{2,3} \cdots \boldsymbol{G}_{k,k+1}$$

可表示为

$$\widetilde{\boldsymbol{L}}_k = \begin{bmatrix} \nu_1 & & & & \\ \delta_2 & \nu_2 & & & \\ \eta_3 & \delta_3 & \nu_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \eta_k & \delta_k & \widetilde{\nu}_k \end{bmatrix}. \tag{4.145}$$




现观察 $\widetilde{\boldsymbol{L}}_{k+1}$, 有

$$\widetilde{\boldsymbol{L}}_{k+1} = \boldsymbol{T}_{k+1}\boldsymbol{Q}_{k+1}^{\mathrm{T}} = \boldsymbol{T}_{k+1}\left[\begin{array}{c|c} \boldsymbol{Q}_k & \\ \hline & 1 \end{array}\right]\boldsymbol{G}_{k,k+1}$$

$$= \left[\begin{array}{cccc|cc} \nu_1 & & & & & \\ \delta_2 & \nu_2 & & & & \\ \eta_3 & \delta_3 & \nu_3 & & & \\ & \ddots & \ddots & \ddots & & \\ \hline & & \eta_k & \delta_k & \widetilde{\nu}_k & \beta_{k+1} \\ & & & \eta_{k+1} & \widetilde{\delta}_{k+1} & \alpha_{k+1} \end{array}\right] \left[\begin{array}{c|cc} \boldsymbol{I}_{k-1} & & \\ \hline & c_k & s_k \\ & s_k & -c_k \end{array}\right],$$

式中: c_k 和 s_k 由方程组

$$\widetilde{\nu}_k c_k + \beta_{k+1}s_k = \nu_k,\ \widetilde{\nu}_k s_k - \beta_{k+1}c_k = 0$$

确定. 由此得 (注意到 $\widetilde{\nu}_1 = \alpha_1,\ \widetilde{\delta}_2 = \beta_2$)

$$\nu_k = \sqrt{\widetilde{\nu}_k^2 + \beta_{k+1}^2},\ c_k = \frac{\widetilde{\nu}_k}{\nu_k},\ s_k = \frac{\beta_{k+1}}{\nu_k}, \tag{4.146}$$

以及

$$\delta_{k+1} = c_k\widetilde{\delta}_{k+1} + s_k\alpha_{k+1},\ \widetilde{\nu}_{k+1} = s_k\widetilde{\delta}_{k+1} - c_k\alpha_{k+1}, \tag{4.147}$$

$$\widetilde{\delta}_{k+2} = -c_k\beta_{k+2},\ \eta_{k+2} = s_k\beta_{k+2}. \tag{4.148}$$

由式 (4.143) 和式 (4.144), $\boldsymbol{x}_k$ 在式 (4.142) 中并不能递推地计算. 为此, 引进 $\boldsymbol{L}_k$, 它是 $\widetilde{\boldsymbol{L}}_k$ 以 ν_k 替换 $\widetilde{\nu}_k$ 而得. 相应地, 定义

$$\boldsymbol{W}_k = \big[\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k\big],\quad \boldsymbol{z}_k = \big[\zeta_1, \zeta_2, \cdots, \zeta_k\big]^{\mathrm{T}},$$

式中: $\boldsymbol{z}_k$ 满足三角方程组

$$\boldsymbol{L}_k\boldsymbol{z}_k = \beta_1\boldsymbol{e}_1. \tag{4.149}$$




Back
Close

由此得 (见 (4.146))

$$\zeta_k = \frac{\widetilde{\nu}_k}{\nu_k}\widetilde{\zeta}_k = c_k\widetilde{\zeta}_k. \tag{4.150}$$

最后, $\boldsymbol{W}_k$ 的列向量可按下列公式递推计算, 即

$$\left[\widetilde{\boldsymbol{w}}_k, \boldsymbol{v}_{k+1}\right]\begin{bmatrix} c_k & s_k \\ s_k & -c_k \end{bmatrix} = \left[\boldsymbol{w}_k, \widetilde{\boldsymbol{w}}_{k+1}\right],\ \widetilde{\boldsymbol{w}}_1 = \boldsymbol{v}_1. \tag{4.151}$$

若 $\widetilde{\nu}_k = 0$, 则 $\widetilde{\boldsymbol{L}}_k$ 是奇异的, 故式 (4.142) 不可解. 但由式 (4.146) 可知, 只要 $\beta_{k+1} \neq 0$, $\boldsymbol{L}_k$ 非奇异, 因而式 (4.149) 可解. 利用 $\boldsymbol{W}_k$ 和 $\boldsymbol{z}_k$ 递推地计算 $\widetilde{\boldsymbol{x}}_k$:

$$\widetilde{\boldsymbol{x}}_k = \boldsymbol{x}_0 + \boldsymbol{W}_k\boldsymbol{z}_k = \widetilde{\boldsymbol{x}}_{k-1} + \zeta_k\boldsymbol{w}_k. \tag{4.152}$$

由式 (4.142), 式 (4.143), 式 (4.144) 和式 (4.152), 有

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_0 + \widetilde{\boldsymbol{W}}_{k+1}\widetilde{\boldsymbol{z}}_{k+1} = \widetilde{\boldsymbol{x}}_k + \widetilde{\zeta}_{k+1}\widetilde{\boldsymbol{w}}_{k+1}. \tag{4.153}$$




式 (4.150) 只是给出了 ζ_k 与 $\widetilde{\zeta}_k$ 之间的关系. 还需要 ζ_k 或 $\widetilde{\zeta}_k$ ($k = 1, 2, \cdots$) 的计算公式. 由式 (4.149), 得

$$\begin{cases} \nu_1\zeta_1 = \beta_1 \Longrightarrow \zeta_1 = \dfrac{\beta_1}{\nu_1}, \\ \delta_2\zeta_1 + \nu_2\zeta_2 = 0 \Longrightarrow \zeta_2 = -\dfrac{\delta_2\zeta_1}{\nu_2}, \\ \eta_k\zeta_{k-2} + \delta_k\zeta_{k-1} + \nu_k\zeta_k = 0 \Longrightarrow \zeta_k = -\dfrac{\eta_k\zeta_{k-2} + \delta_k\zeta_{k-1}}{\nu_k}, \ k \geqslant 3. \end{cases} \tag{4.154}$$

求解式 (4.149) 有比式 (4.142) 更好的数值性态. 由式 (4.152) 可求得 $\{\widetilde{\boldsymbol{x}}_k\}$, 若需要 $\{\boldsymbol{x}_k\}$ 可按式 (4.153) 得到. 所描述的算法称为 SYMMLQ 方法.

算法 4.12 (SYMMLQ 方法) 给定 n 阶非奇异的实对称矩阵 $\boldsymbol{A}$, n 维向量 $\boldsymbol{b}$ 和允许误差 $\varepsilon > 0$. 本算法计算向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\beta_1 = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta_1$;

$\alpha_1 = \boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_1$; $\widetilde{\boldsymbol{v}}_2 = \boldsymbol{A}\boldsymbol{v}_1 - \alpha_1\boldsymbol{v}_1$; $\beta_2 = \|\widetilde{\boldsymbol{v}}_2\|_2$;

$\boldsymbol{v}_2 = \widetilde{\boldsymbol{v}}_2/\beta_2$; $\widetilde{\boldsymbol{w}}_1 = \boldsymbol{v}_1$;

$\widetilde{\nu}_1 = \alpha_1$; $\widetilde{\delta}_2 = \beta_2$; $\zeta_0 = 0$; $\eta_2 = 0$;

$\nu_1 = \sqrt{\alpha_1^2 + \beta_2^2}$; $c_1 = \widetilde{\nu}_1/\nu_1$; $s_1 = \beta_2/\nu_1$;

由 $\left[\widetilde{\boldsymbol{w}}_1, \boldsymbol{v}_2\right] \begin{bmatrix} c_1 & s_1 \\ s_1 & -c_1 \end{bmatrix} = \left[\boldsymbol{w}_1, \widetilde{\boldsymbol{w}}_2\right]$ 确定 $\boldsymbol{w}_1$ 和 $\widetilde{\boldsymbol{w}}_2$;

即 $\boldsymbol{w}_1 = c_1\widetilde{\boldsymbol{w}}_1 + s_1\boldsymbol{v}_2$; $\widetilde{\boldsymbol{w}}_2 = s_1\widetilde{\boldsymbol{w}}_1 - c_1\boldsymbol{v}_2$;

$\zeta_1 = \dfrac{\beta_1}{\nu_1}$; $\widetilde{\boldsymbol{x}}_1 = \boldsymbol{x}_0 + \zeta_1\boldsymbol{w}_1$; $k = 1$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 > \varepsilon)$

$\alpha_{k+1} = \boldsymbol{v}_{k+1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_{k+1}$;

$\widetilde{\boldsymbol{v}}_{k+2} = \boldsymbol{A}\boldsymbol{v}_{k+1} - \alpha_{k+1}\boldsymbol{v}_{k+1} - \beta_{k+1}\boldsymbol{v}_k$;

$\beta_{k+2} = \|\widetilde{\boldsymbol{v}}_{k+2}\|_2$; $\boldsymbol{v}_{k+2} = \widetilde{\boldsymbol{v}}_{k+2}/\beta_{k+2}$;

Back

Close

$$\widetilde{\nu}_{k+1} = s_k\widetilde{\delta}_{k+1} - c_k\alpha_{k+1};\ \ \nu_{k+1} = \sqrt{\widetilde{\nu}_{k+1}^2 + \beta_{k+2}^2};$$

$$c_{k+1} = \frac{\widetilde{\nu}_{k+1}}{\nu_{k+1}};\ \ s_{k+1} = \frac{\beta_{k+2}}{\nu_{k+1}};$$

$$\delta_{k+1} = c_k\widetilde{\delta}_{k+1} + s_k\alpha_{k+1};$$

$$\widetilde{\delta}_{k+2} = -c_k\beta_{k+2},\ \ \eta_{k+2} = s_k\beta_{k+2};$$

$$\zeta_{k+1} = -\frac{\eta_{k+1}\zeta_{k-1} + \delta_{k+1}\zeta_k}{\nu_{k+1}};\ \ \widetilde{\zeta}_{k+1} = \zeta_{k+1}/c_{k+1};$$

由 $\left[\widetilde{\boldsymbol{w}}_{k+1}, \boldsymbol{v}_{k+2}\right]\begin{bmatrix} c_{k+1} & s_{k+1} \\ s_{k+1} & -c_{k+1} \end{bmatrix} = \left[\boldsymbol{w}_{k+1}, \widetilde{\boldsymbol{w}}_{k+2}\right]$ 确定 $\boldsymbol{w}_{k+1}$ 和 $\widetilde{\boldsymbol{w}}_{k+2}$;

即 $\boldsymbol{w}_{k+1} = c_{k+1}\widetilde{\boldsymbol{w}}_{k+1} + s_{k+1}\boldsymbol{v}_{k+2};\ \ \widetilde{\boldsymbol{w}}_{k+2} = s_{k+1}\widetilde{\boldsymbol{w}}_{k+1} - c_{k+1}\boldsymbol{v}_{k+2}$;

$$\boldsymbol{x}_{k+1} = \widetilde{\boldsymbol{x}}_k + \widetilde{\zeta}_{k+1}\widetilde{\boldsymbol{w}}_{k+1};\ \ \widetilde{\boldsymbol{x}}_{k+1} = \widetilde{\boldsymbol{x}}_k + \zeta_{k+1}\boldsymbol{w}_{k+1};$$

Back

Close

$\boldsymbol{r}_{k+1} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{k+1}$;

$k = k + 1$;

end

例 4.10 仍考虑例 4.8 中的线性方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, 其中矩阵 $\boldsymbol{A}$ 来自

http://www.cise.ufl.edu./research/sparse/matrices/PARSEC/SiNa.html,

它是一个 5743 阶的实对称不定稀疏矩阵, 右端项 $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{e}$, 其中 $\boldsymbol{e} = (1, 1, \cdots, 1)^{\mathrm{T}}$. 显然, 该方程组的真解为 $\boldsymbol{x}^* = (1, 1, \cdots, 1)^{\mathrm{T}}$. 将 SYMMLQ 方法应用到该线性方程组上, 迭代在 224 步后收敛 ($\varepsilon = 10^{-10}$), 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的绝对值误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 2.8998 \times 10^{-8},$$

计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 2.1331 \times 10^{-8}.$$

将它与 MINRES 方法和 GMRES 方法进行了对比, 发现 SYMMLQ 方法与 MINRES 方法对本例有几乎完全等效的数值表现, 而 GMRES 方法则要略好一些. 迭代过程的收敛轨迹如图 4.11 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

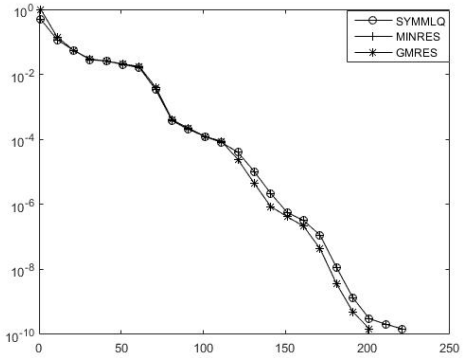


图 4.11 SYMMLQ 方法的收敛特性

§4.4.2 收敛性分析

对于方程组

$$\boldsymbol{A}\boldsymbol{z} = \boldsymbol{r}_0, \quad \boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0, \quad \boldsymbol{z} = \boldsymbol{x} - \boldsymbol{x}_0,$$

考虑二次泛函

$$J_k^{(k)}(\boldsymbol{z}_k) = (\boldsymbol{z} - \boldsymbol{z}_k)^{\mathrm{T}} \boldsymbol{A}^k (\boldsymbol{z} - \boldsymbol{z}_k), \tag{4.155}$$

式中: k 为非负整数; $\boldsymbol{z}_k = \boldsymbol{V}_k \boldsymbol{y}_k$, $\boldsymbol{V}_k = \left[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\right]$.

当 $k = 2$ 时, 式 (4.155) 中的泛函

$$\begin{aligned} J_k^{(2)}(\boldsymbol{z}_k) &= (\boldsymbol{z} - \boldsymbol{z}_k)^{\mathrm{T}} \boldsymbol{A}^2 (\boldsymbol{z} - \boldsymbol{z}_k) = \|\boldsymbol{A}(\boldsymbol{z} - \boldsymbol{z}_k)\|_2^2 \\ &= \|\boldsymbol{r}_0 - \boldsymbol{A}(\boldsymbol{x}_k - \boldsymbol{x}_0)\|_2^2 = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\|_2^2 = \|\boldsymbol{r}_k\|_2^2 \end{aligned}$$

是正定二次泛函. 极小化这一正定二次泛函就是 节中极小残量法 (MINRES) 的基本思想.




当 $k=1$ 时, 式 (4.155) 中的泛函为

$$\begin{aligned} J_k^{(1)}(\boldsymbol{z}_k) &= (\boldsymbol{z}-\boldsymbol{z}_k)^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_k) \\ &= (\boldsymbol{x}-\boldsymbol{x}_0-\boldsymbol{V}_k\boldsymbol{y}_k)^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{x}_0-\boldsymbol{V}_k\boldsymbol{y}_k). \end{aligned}$$

求上述泛函稳定点 $\boldsymbol{y}_k$:

$$\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{x}_0-\boldsymbol{V}_k\boldsymbol{y}_k)=\boldsymbol{V}_k^{\mathrm{T}}(\boldsymbol{r}_0-\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k)=\boldsymbol{0}$$

是本节 SYMMLQ 方法的出发点.

若在式 (4.155) 中取 $k=0$, 则有

$$J_k^{(0)}(\boldsymbol{z}_k)=(\boldsymbol{z}-\boldsymbol{z}_k)^{\mathrm{T}}(\boldsymbol{z}-\boldsymbol{z}_k)=\|\boldsymbol{x}-\boldsymbol{x}_k\|_2^2=\|\boldsymbol{\varepsilon}_k\|_2^2.$$

这也是一个正定二次泛函. 因而使 $J_k^{(0)}$ 极小化就是使误差 $\boldsymbol{\varepsilon}_k=\boldsymbol{x}-\boldsymbol{x}_k$ 的范数极小化.




Back
Close

为了使极小误差法可行, 取迭代所处的仿射子空间为

$$\boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{r}_0),$$

也就是说

$$\boldsymbol{z}_k = \boldsymbol{x}_k - \boldsymbol{x}_0 = \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k.$$

由此, $\boldsymbol{y}_k$ 是最小二乘问题

$$\min \|\boldsymbol{x} - \boldsymbol{x}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k\|_2 = \|\boldsymbol{A}^{-1}\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k\|_2$$

的解. 由法方程可知 $\boldsymbol{y}_k$ 满足线性方程组

$$\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}^2\boldsymbol{V}_k\boldsymbol{y}_k = \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{r}_0 = \beta_1\boldsymbol{e}_1, \quad \boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k. \tag{4.156}$$

由对称 Lanzcos 分解式 (4.137) 及 LQ 分解式 (4.139), 得

$$\begin{aligned}\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}^2\boldsymbol{V}_k &= (\boldsymbol{A}\boldsymbol{V}_k)^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{V}_k)\\ &= (\boldsymbol{V}_k\boldsymbol{T}_k + \beta_{k+1}\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}})^{\mathrm{T}}(\boldsymbol{V}_k\boldsymbol{T}_k + \beta_{k+1}\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}})\\ &= \boldsymbol{T}_k^2 + \beta_{k+1}^2\boldsymbol{e}_k\boldsymbol{e}_k^{\mathrm{T}} = (\widetilde{\boldsymbol{L}}_k\boldsymbol{Q}_k)(\widetilde{\boldsymbol{L}}_k\boldsymbol{Q}_k)^{\mathrm{T}} + \beta_{k+1}^2\boldsymbol{e}_k\boldsymbol{e}_k^{\mathrm{T}}\\ &= \widetilde{\boldsymbol{L}}_k\widetilde{\boldsymbol{L}}_k^{\mathrm{T}} + \beta_{k+1}^2\boldsymbol{e}_k\boldsymbol{e}_k^{\mathrm{T}} = \boldsymbol{L}_k\boldsymbol{L}_k^{\mathrm{T}},\end{aligned}$$

最后一个等式成立是因为 $\widetilde{\boldsymbol{L}}_k\widetilde{\boldsymbol{L}}_k^{\mathrm{T}}$ 与 $\boldsymbol{L}_k\boldsymbol{L}_k^{\mathrm{T}}$ 只是 (k,k) 处的那个元素不同, 前者是 $\eta_k^2+\delta_k^2+\widetilde{\nu}_k^2$, 而后者为 $\eta_k^2+\delta_k^2+\nu_k^2$. 注意到 $\nu_k^2=\widetilde{\nu}_k^2+\beta_{k+1}^2$, 即可得等式成立. 这样, 直接从 $\boldsymbol{T}_k$ 的 LQ 分解得到了 $\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}^2\boldsymbol{V}_k$ 的 Cholesky 分解. 代入式 (4.156), 需要求解如下线性方程组以得到 $\boldsymbol{y}_k$ 和 $\boldsymbol{x}_k$:

$$\boldsymbol{L}_k\boldsymbol{L}_k^{\mathrm{T}}\boldsymbol{y}_k = \beta_1\boldsymbol{e}_1, \quad \boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k. \tag{4.157}$$




由上述方式得到近似解序列 $\{\boldsymbol{x}_k\}$ 的方法通常称为极小误差法. 下面证明这个极小误差解序列 $\{\boldsymbol{x}_k\}$ 就是 SYMMLQ 方法中的 $\{\widetilde{\boldsymbol{x}}_k\}$ (见式 (4.152)).

定理 4.9 SYMMLQ 方法中的序列 $\{\widetilde{\boldsymbol{x}}_k\}$ 是在 Krylov 子空间 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{r}_0)$ 的极小误差逼近.

证明 由式 (4.157) 可知极小误差解 $\boldsymbol{x}_k$ 可表示为

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \beta_1 \boldsymbol{A}\boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}} \boldsymbol{L}_k^{-1} \boldsymbol{e}_1. \tag{4.158}$$

由式 (4.149) 和式 (4.152) 可知, $\widetilde{\boldsymbol{x}}_k$ 可表示为

$$\widetilde{\boldsymbol{x}}_k = \boldsymbol{x}_0 + \beta_1 \boldsymbol{W}_k \boldsymbol{L}_k^{-1} \boldsymbol{e}_1. \tag{4.159}$$

比较式 (4.158) 和式 (4.159) 右端, 只需证明

$$\boldsymbol{W}_k = \boldsymbol{A}\boldsymbol{V}_k \boldsymbol{L}_k^{-\mathrm{T}}, \tag{4.160}$$

即完成了定理的证明. 由式 (4.139) 和式 (4.141) 可知

$$\widetilde{\boldsymbol{W}}_{k+1}\widetilde{\boldsymbol{L}}_{k+1}^{\mathrm{T}} = \boldsymbol{V}_{k+1}(\widetilde{\boldsymbol{L}}_{k+1}\boldsymbol{Q}_{k+1})^{\mathrm{T}} = \boldsymbol{V}_{k+1}\boldsymbol{T}_{k+1}. \tag{4.161}$$

再由式 (4.137), 得

$$\widetilde{\boldsymbol{W}}_{k+1} = (\boldsymbol{A}\boldsymbol{V}_{k+1} - \beta_{k+2}\boldsymbol{v}_{k+2}\boldsymbol{e}_{k+1}^{\mathrm{T}})\widetilde{\boldsymbol{L}}_{k+1}^{-\mathrm{T}}. \tag{4.162}$$

比较式 (4.162) 前 k 列, 并注意到 $\widetilde{\boldsymbol{L}}_{k+1}^{-\mathrm{T}}$ 是上三角矩阵, 可得式 (4.160). 证毕. □

下面分析极小误差法的误差估计. 令

$$\widetilde{\mathcal{P}}_{k-1}^{0}(t) = 1 - t^2\mathcal{P}_{k-1}(t)$$

表示次数不超过 $k+1$, 在 $t=0$ 时值等于 1 且一阶导数等于 0 的多项式集合. 因 $\boldsymbol{x}_k - \boldsymbol{x}_0 \in \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{r}_0)$, 故存在 $p_{k-1} \in \mathcal{P}_{k-1}$, 使得




$\boldsymbol{x}_k - \boldsymbol{x}_0 = p_{k-1}(\boldsymbol{A})\boldsymbol{A}\boldsymbol{r}_0$. 所以误差 $\boldsymbol{\varepsilon}_k = \boldsymbol{x}^* - \boldsymbol{x}_k$ 满足

$$\begin{aligned}
\|\boldsymbol{\varepsilon}_k\|_2^2 &= \min_{p_{k-1}\in\mathcal{P}_{k-1}} \big(\boldsymbol{\varepsilon}_0 - p_{k-1}(\boldsymbol{A})\boldsymbol{A}\boldsymbol{r}_0, \boldsymbol{\varepsilon}_0 - p_{k-1}(\boldsymbol{A})\boldsymbol{A}\boldsymbol{r}_0\big) \\
&= \min_{p_{k-1}\in\mathcal{P}_{k-1}} \big(\boldsymbol{\varepsilon}_0 - p_{k-1}(\boldsymbol{A})\boldsymbol{A}^2\boldsymbol{\varepsilon}_0, \boldsymbol{\varepsilon}_0 - p_{k-1}(\boldsymbol{A})\boldsymbol{A}^2\boldsymbol{\varepsilon}_0\big) \\
&= \min_{p_{k-1}\in\mathcal{P}_{k-1}} \big([\boldsymbol{I} - \boldsymbol{A}^2 p_{k-1}(\boldsymbol{A})]\boldsymbol{\varepsilon}_0, [\boldsymbol{I} - \boldsymbol{A}^2 p_{k-1}(\boldsymbol{A})]\boldsymbol{\varepsilon}_0\big) \\
&= \min_{\widetilde{p}_{k-1}\in\widetilde{\mathcal{P}}_{k-1}^0} \big(\widetilde{p}_{k-1}(\boldsymbol{A})\boldsymbol{\varepsilon}_0, \widetilde{p}_{k-1}(\boldsymbol{A})\boldsymbol{\varepsilon}_0\big) \\
&\leqslant \min_{\widetilde{p}_{k-1}\in\widetilde{\mathcal{P}}_{k-1}^0} \max_{\lambda\in\sigma(\boldsymbol{A})} \widetilde{p}_{k-1}^2(\lambda)\|\boldsymbol{\varepsilon}_0\|_2^2. \qquad (4.163)
\end{aligned}$$

因为 $\boldsymbol{A}$ 是对称不定的, 所以 $\sigma(\boldsymbol{A})$ 包含正的和负的特征值. 有下面的定理.

定理 4.10 若 $\boldsymbol{A}$ 的谱 $\sigma(\boldsymbol{A}) \subset [-b,-a]\cup[a,b]$, 其中 $0 < a <$

b. 则对极小误差迭代法成立

$$\|\boldsymbol{\varepsilon}_k\|_2 \leqslant \frac{2\widetilde{r}^{\widetilde{k}}}{1+\widetilde{r}^{2\widetilde{k}}}\|\boldsymbol{\varepsilon}_0\|_2, \tag{4.164}$$

式中:

$$\widetilde{r} = \frac{b-a}{b+a} = \frac{b/a-1}{b/a+1}, \quad \widetilde{k} = \left[\frac{k+1}{2}\right].$$

证明 取多项式

$$p_{k-1}(t) = \frac{C_{\widetilde{k}}(v(t))}{C_{\widetilde{k}}(v(0))},$$

式中: $C_{\widetilde{k}}(z)$ **为** $\widetilde{k} = [(k+1)/2]$ **次** Chebyshev **多项式, 且**

$$v(t) = 1 - \frac{2(t^2-a^2)}{b^2-a^2},$$

它将区间 $[-b,-a]\cup[a,b]$ **映射到** $[-1,1]$.




显然, $p_{k-1}(t)$ 的次数不超过 $k+1$, 它是偶函数, 且 $p_{k-1}(0)=1$. 故 $p_{k-1}(t)\in \widetilde{P}^0_{k-1}$. 由式 (4.163), 有

$$\|\boldsymbol{\varepsilon}_k\|_2 \leqslant \max_{t\in[-b,-a]\cup[a,b]}\left|\frac{C_{\widetilde{k}}(v(t))}{C_{\widetilde{k}}(v(0))}\right|\|\boldsymbol{\varepsilon}_0\|_2 = \frac{1}{C_{\widetilde{k}}(v(0))}\|\boldsymbol{\varepsilon}_0\|_2.$$

不难推得

$$\begin{aligned} C_{\widetilde{k}}(v(0)) &= C_{\widetilde{k}}\left(\frac{b^2+a^2}{b^2-a^2}\right) \\ &= \frac{1}{2}\left\{\left[\frac{b^2+a^2}{b^2-a^2}+\sqrt{\left(\frac{b^2+a^2}{b^2-a^2}\right)^2-1}\right]^{\widetilde{k}}+\left[\frac{b^2+a^2}{b^2-a^2}+\sqrt{\left(\frac{b^2+a^2}{b^2-a^2}\right)^2-1}\right]^{-\widetilde{k}}\right\} \\ &= \frac{1}{2}\left[\left(\frac{b+a}{b-a}\right)^{\widetilde{k}}+\left(\frac{b+a}{b-a}\right)^{-\widetilde{k}}\right] = \frac{1}{2}\left[\left(\frac{1}{\widetilde{r}}\right)^{\widetilde{k}}+\widetilde{r}^{\widetilde{k}}\right] = \frac{1+\widetilde{r}^{2\widetilde{k}}}{2\widetilde{r}^{\widetilde{k}}}. \end{aligned}$$

证毕. □

§4.5 拟极小残量法

在 4.2 节中, 为了解决广义极小残量法的存储问题而采用了重新开始的方法, 进而导出了实际应用中常用的重开始广义极小残量法 GMRES(m). 本节介绍另一种解决存储问题的方法—拟极小残量法, 在目前的文献中常简称为 QMRES 方法 (Quasi-Minimal RESidual Approach).

GMRES 方法的存储量主要来自计算 Krylov 子空间 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{b})$ 的正交基 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$. 由于要求这些 $\boldsymbol{v}_i\,(i = 1, 2, \cdots, k)$ 相互正交, 故必须在计算中将这些向量都存储起来. 因此要使其存储量降低, 一个自然的想法是放弃正交性要求, 而选择其他适当的基向量.

类比于 MINRES 方法, 自然希望计算 Krylov 子空间 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 的一组基向量 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$, 使得

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k\}, \quad \boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2, \tag{4.165}$$

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{T}}_k, \tag{4.166}$$

式中: $\boldsymbol{V}_{k+1} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k, \boldsymbol{v}_{k+1}]$ 列满秩, 而 $\widetilde{\boldsymbol{T}}_k$ 是 $(k+1)\times k$ 阶三对角矩阵.

如果能够找到一种方法来实现式 (4.165) 和式 (4.166) 中的分解, 则由于这样的基满足等式 $\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{T}}_k$, 向量 $\boldsymbol{v}_i$ 就可由三项递推公式来确定, 从而所需的存储量只有 $O(n)$ 而并非 GMRES 方法的 $O(kn)$. 然而, 这样一来, 所要解决的极小化问题 (4.52) 就转化为求 $\boldsymbol{z}_k \in \mathbb{R}^k$, 使得

$$\|\boldsymbol{r}_k\|_2 = \|\boldsymbol{V}_{k+1}(\beta\boldsymbol{e}_1 - \widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k)\|_2 = \min, \tag{4.167}$$

式中: $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{z}_k$, $\beta = \|\boldsymbol{r}_0\|_2$. 但由于此时的 $\boldsymbol{V}_{k+1}$ 列向量并不正交, 故求解这一极小化问题就有很大的困难. 注意到

$$\|\boldsymbol{r}_k\|_2 = \|\boldsymbol{V}_{k+1}(\beta\boldsymbol{e}_1 - \widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k)\|_2 \leqslant \|\boldsymbol{V}_{k+1}\|_2\|\beta\boldsymbol{e}_1 - \widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k\|_2,$$

如果 $\|\boldsymbol{V}_{k+1}\|_2$ 不是特别大, 则只需求得最小二乘问题

$$\min\left\{\|\beta\boldsymbol{e}_1-\widetilde{\boldsymbol{T}}_k\boldsymbol{z}\|_2:\boldsymbol{z}\in\mathbb{R}^k\right\} \tag{4.168}$$

的解 $\boldsymbol{z}_k$. 虽然 $\|\boldsymbol{r}_k\|_2$ 并没有达到最小, 但也会相对很小, 这就是拟极小化方法的基本想法. 总结起来, 主要有两步:

第 1 步, 计算 $\mathcal{K}_k(\boldsymbol{A},\boldsymbol{b})$ 之满足条件 (4.165) 和 (4.166) 的一组基.

第 2 步, 求解最小二乘问题 (4.168).

关键是第 1 步如何实现, 第 2 步可用与 4.3 节中完全一样的方法来求解. 这里用非对称 Lanczos 方法实现 $\mathcal{K}_k(\boldsymbol{A},\boldsymbol{r}_0)$ 基向量的计算问题.

§4.5.1 非对称 Lanczos 方法

非对称 Lanczos 方法是对称 Lanczos 方法的一种自然的推广, 它是同时计算 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{v}_1)$ 和 $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{w}_1)$ 的基向量 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$ 和 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k$, 使得

$$\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{v}_j = \begin{cases} 0, & i \neq j, \\ 1, & i = j. \end{cases} \tag{4.169}$$

式 (4.169) 称为双正交性条件.

设矩阵 $\boldsymbol{A}$ 和向量 $\boldsymbol{v}_1$ 已经给定, 并且假定 $\|\boldsymbol{v}_1\|_2 = 1$. 首先需选择一个 $\boldsymbol{w}_1 \in \mathbb{R}^n$, 使得 $\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{w}_1 = 1$. 如果没有什么特别的信息可以利用的话, 可以简单地选择 $\boldsymbol{w}_1 = \boldsymbol{v}_1$. 然后类比于对称 Lanczos 方法的计算过程, 也采用如下的三项递推方式, 即

$$\begin{cases} \beta_i \boldsymbol{v}_{i+1} = \boldsymbol{A}\boldsymbol{v}_i - \alpha_i \boldsymbol{v}_i - \gamma_{i-1}\boldsymbol{v}_{i-1}, \\ \gamma_i \boldsymbol{w}_{i+1} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_i - \alpha_i \boldsymbol{w}_i - \beta_{i-1}\boldsymbol{w}_{i-1}. \end{cases} \tag{4.170}$$

依次确定 $\alpha_i, \beta_i, \gamma_i, \boldsymbol{v}_{i+1}$ 和 $\boldsymbol{w}_{i+1}$. 这里假定 $\gamma_0\boldsymbol{v}_0 = \beta_0\boldsymbol{w}_0 = \boldsymbol{0}$, 即当 $i = 1$ 时, 有

$$\begin{aligned} \beta_1 \boldsymbol{v}_2 &= \boldsymbol{A}\boldsymbol{v}_1 - \alpha_1 \boldsymbol{v}_1, \\ \gamma_1 \boldsymbol{w}_2 &= \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_1 - \alpha_1 \boldsymbol{w}_1. \end{aligned}$$

由于希望计算得到的 $\boldsymbol{v}_2$ 和 $\boldsymbol{w}_2$ 满足 $\boldsymbol{v}_2 \perp \boldsymbol{w}_1$, $\boldsymbol{w}_2 \perp \boldsymbol{v}_1$, 故在上面的第 1 式两边左乘 $\boldsymbol{w}_1^{\mathrm{T}}$, 得

$$\alpha_1 = \boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_1. \tag{4.171}$$

此处利用了 $\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{v}_1 = 1$. 当然, 也可以在前面的第 2 式两边左乘 $\boldsymbol{v}_1^{\mathrm{T}}$,




得到 α_1 的表达式 (4.171). 一旦 α_1 确定, 便可计算

$$\beta_1 \boldsymbol{v}_2 = \widetilde{\boldsymbol{v}}_2 = \boldsymbol{A}\boldsymbol{v}_1 - \alpha_1 \boldsymbol{v}_1,$$
$$\gamma_1 \boldsymbol{w}_2 = \widetilde{\boldsymbol{w}}_2 = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_1 - \alpha_1 \boldsymbol{w}_1.$$

由于要求 $\boldsymbol{v}_2^{\mathrm{T}}\boldsymbol{w}_2 = 1$, 故必有

$$\beta_1\gamma_1 = \widetilde{\boldsymbol{v}}_2^{\mathrm{T}}\widetilde{\boldsymbol{w}}_2 = \omega_1.$$

如果 $\omega_1 = 0$, 则只好结束; 否则可选择两个数 β_1 和 γ_1 使上式成立. 可选择

$$\beta_1 = \sqrt{|\omega_1|}, \quad \gamma_1 = \omega_1/\beta_1. \tag{4.172}$$

这样一来, 所需的向量 $\boldsymbol{v}_2$ 和 $\boldsymbol{w}_2$ 即为

$$\boldsymbol{v}_2 = \widetilde{\boldsymbol{v}}_2/\beta_1, \quad \boldsymbol{w}_2 = \widetilde{\boldsymbol{w}}_2/\gamma_1. \tag{4.173}$$

假定已经确定了

$$\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_i; \quad \boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_i;$$

$$\alpha_1, \alpha_2, \cdots, \alpha_{i-1}; \quad \gamma_1, \gamma_2, \cdots, \gamma_{i-1}; \quad \beta_1, \beta_2, \cdots, \beta_{i-1},$$

下一步利用 $\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{v}_{i+1} = \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{v}_{i-1} = 0$ 和 $\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{v}_i = 1$, 从式 (4.170) 的第 1 式可导出

$$\alpha_i = \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_i. \tag{4.174}$$

然后, 计算

$$\begin{cases} \widetilde{\boldsymbol{v}}_{i+1} = \boldsymbol{A}\boldsymbol{v}_i - \alpha_i\boldsymbol{v}_i - \gamma_{i-1}\boldsymbol{v}_{i-1}, \\ \widetilde{\boldsymbol{w}}_{i+1} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_i - \alpha_i\boldsymbol{w}_i - \beta_{i-1}\boldsymbol{w}_{i-1}, \\ \omega_i = \widetilde{\boldsymbol{v}}_{i+1}^{\mathrm{T}}\widetilde{\boldsymbol{w}}_{i+1}. \end{cases} \tag{4.175}$$

若 $\omega_i = 0$, 则结束; 否则, 计算

$$\beta_i = \sqrt{|\omega_i|}, \quad \gamma_i = \omega_i/\beta_i, \quad \boldsymbol{v}_{i+1} = \widetilde{\boldsymbol{v}}_{i+1}/\beta_i, \quad \boldsymbol{w}_{i+1} = \widetilde{\boldsymbol{w}}_{i+1}/\gamma_i. \tag{4.176}$$

假定上述过程共进行了 k 步, 现在令

$$\boldsymbol{V}_k = \big[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\big], \quad \boldsymbol{W}_k = \big[\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k\big],$$

$$\boldsymbol{T}_k = \begin{bmatrix} \alpha_1 & \gamma_1 & & & \\ \beta_1 & \alpha_2 & \gamma_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \ddots & \gamma_{k-1} \\ & & & \beta_{k-1} & \alpha_k \end{bmatrix}. \tag{4.177}$$

则易从上面的推导过程归纳地证明关于非对称 Lanczos 迭代的如下基本性质:

(1) $\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k\boldsymbol{T}_k + \beta_k \boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}}$.

(2) $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{W}_k = \boldsymbol{W}_k\boldsymbol{T}_k^{\mathrm{T}} + \gamma_k \boldsymbol{w}_{k+1}\boldsymbol{e}_k^{\mathrm{T}}$.

(3) $\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{W}_k = \boldsymbol{I}_k,\ \boldsymbol{v}_{k+1}^{\mathrm{T}}\boldsymbol{W}_k = \boldsymbol{0},\ \boldsymbol{w}_{k+1}^{\mathrm{T}}\boldsymbol{V}_k = \boldsymbol{0},\ \boldsymbol{w}_{k+1}^{\mathrm{T}}\boldsymbol{v}_{k+1} = 1$.

(4) $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{v}_1) = \mathrm{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$.

(5) $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{w}_1) = \mathrm{span}\{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k\}$.

其实 (1) 和 (2) 就是前面算法的矩阵表示. (4) 和 (5) 是前三条的直接推论. 例如 (4), 由 (1) 归纳地可证 $\boldsymbol{v}_i \in \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{v}_1), i = 1, 2, \cdots, k$, 而 (3) 又蕴涵着 $\boldsymbol{V}_k$ 是列满秩的 (即 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k$ 是线性无关的), 从而必有 (4) 成立. 因此关键是 (3) 的证明.

下面用数学归纳法证明性质 (3). 首先由初值 $\boldsymbol{w}_1$ 的选择知, 自然有 $\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{v}_1 = 1$ 成立. 现在假设已经对自然数 i 证明了向量组 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_i$ 和 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_i$ 满足双正交性条件, 再来考虑 $i+1$ 的情形. 注意这里假定 $i + 1 \leqslant k + 1$.

首先由归纳法假设和 α_i 的定义, 有

$$\begin{aligned}\boldsymbol{v}_{i+1}^{\mathrm{T}}\boldsymbol{w}_i &= \frac{1}{\beta_i}(\boldsymbol{A}\boldsymbol{v}_i - \alpha_i\boldsymbol{v}_i - \gamma_{i-1}\boldsymbol{v}_{i-1})^{\mathrm{T}}\boldsymbol{w}_i \\ &= \frac{1}{\beta_i}\big(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_i - \alpha_i\big) = 0.\end{aligned}$$

对任意的 $l < i$, 有

$$\begin{aligned}\boldsymbol{v}_{i+1}^{\mathrm{T}}\boldsymbol{w}_l &= \frac{1}{\beta_i}\big(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_l - \alpha_i\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{w}_l - \gamma_{i-1}\boldsymbol{v}_{i-1}^{\mathrm{T}}\boldsymbol{w}_l\big) \\ &= \frac{1}{\beta_i}\big[\boldsymbol{v}_i^{\mathrm{T}}(\alpha_l\boldsymbol{w}_l + \beta_{l-1}\boldsymbol{w}_{l-1} + \gamma_l\boldsymbol{w}_{l+1}) - \gamma_{i-1}\boldsymbol{v}_{i-1}^{\mathrm{T}}\boldsymbol{w}_l\big].\end{aligned}$$

当 $l < i-1$ 时, 由归纳法假设立即由上式知 $\boldsymbol{v}_{i+1}^{\mathrm{T}}\boldsymbol{w}_l = 0$. 而当 $l = i-1$ 时, 直接计算有

$$\boldsymbol{v}_{i+1}^{\mathrm{T}}\boldsymbol{w}_{i-1} = \frac{1}{\beta_i}\left(\alpha_{i-1}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{w}_{i-1} + \beta_{i-2}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{w}_{i-2} + \gamma_{i-1}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{w}_i - \gamma_{i-1}\boldsymbol{v}_{i-1}^{\mathrm{T}}\boldsymbol{w}_{i-1}\right) = 0,$$

最后的等式用到了归纳法假设. 至于 $\boldsymbol{w}_{i+1}^{\mathrm{T}}\boldsymbol{v}_{i+1} = 1$ 可由计算过程

Back

Close

立即得到. 这表明 (3) 对 $i+1$ 也成立. 因此由归纳法原理知 (3) 成立.

综合上面的讨论, 可得下面的非对称 Lanczos 方法.

算法 4.13 (非对称 Lanczos 方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 向量 $\boldsymbol{v}_1 \in \mathbb{R}^n$ $(\|\boldsymbol{v}_1\|_2 = 1)$ 和正整数 k. 本算法计算形如 (4.177) 的三个矩阵 $\boldsymbol{V}_k, \boldsymbol{W}_k$ 和 $\boldsymbol{T}_k$, 以及向量 $\boldsymbol{v}_{k+1}, \boldsymbol{w}_{k+1}$ 和数 β_k, γ_k, 满足前面所述的 5 条性质.

选择向量 $\boldsymbol{w}_1 \in \mathbb{R}^n$ 满足 $\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{v}_1 = 1$;

$\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v}_1$; $\boldsymbol{w} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_1$;

for $i = 1 : k$

$\alpha_i = \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{v}$; $\boldsymbol{v} = \boldsymbol{v} - \alpha_i\boldsymbol{v}_i$; $\boldsymbol{w} = \boldsymbol{w} - \alpha_i\boldsymbol{w}_i$;

if $\|\boldsymbol{v}\|_2 = 0$ 或者 $\|\boldsymbol{w}\|_2 = 0$

stop

else

$\omega_i = \boldsymbol{v}^{\mathrm{T}}\boldsymbol{w}$;

end

if $\omega_i = 0$

stop

else

$\beta_i = \sqrt{|\omega_i|}$; $\gamma_i = \omega_i/\beta_i$;

$\boldsymbol{v}_{i+1} = \boldsymbol{v}/\beta_i$; $\boldsymbol{w}_{i+1} = \boldsymbol{w}/\gamma_i$;

end

$\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v}_{i+1} - \gamma_i\boldsymbol{v}_i$; $\boldsymbol{w} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_{i+1} - \beta_i\boldsymbol{w}_i$;

end

注 4.5 非对称 Lanczos 迭代的基本性质 (1) ∼ (3) 蕴涵着

$$\boldsymbol{T}_k = \boldsymbol{W}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k.$$

这表明 $\boldsymbol{T}_k$ 正好是 $\boldsymbol{A}$ 沿着 $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{w}_1)^{\perp}$ 到 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{v}_1)$ 上的投影. 当然, $\boldsymbol{T}_k^{\mathrm{T}}$ 也是 $\boldsymbol{A}$ 沿着 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{v}_1)^{\perp}$ 到 $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{w}_1)$ 上的投影. 有关投影的几何解释见注 4.6.

注 4.6 设 $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{n\times k}$ 满足 $\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{X} = \boldsymbol{I}_k$, 并记

$$\mathcal{X} = \mathcal{R}(\boldsymbol{X}), \quad \mathcal{Y} = \mathcal{R}(\boldsymbol{Y}), \quad \boldsymbol{P} = \boldsymbol{X}\boldsymbol{Y}^{\mathrm{T}},$$

则容易验证

$$\mathcal{R}(\boldsymbol{P}) = \mathcal{X}, \quad \mathcal{N}(\boldsymbol{P}) = \mathcal{Y}^{\perp}, \quad \boldsymbol{P}^2 = \boldsymbol{P},$$

这表明 $\boldsymbol{P}$ 是沿 $\mathcal{Y}^{\perp}$ 到 $\mathcal{X}$ 上的投影算子.

设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 将 $\boldsymbol{A}$ 视作 $\mathbb{R}^n$ 到 $\mathbb{R}^n$ 的线性算子, 并记其在 $\mathcal{X}$ 上的限制为 $\boldsymbol{A}|_{\mathcal{X}}$, 则 $\boldsymbol{P} \circ \boldsymbol{A}|_{\mathcal{X}}$ 就是从 $\mathcal{X}$ 到 $\mathcal{X}$ 的线性算子. 由于

$$\boldsymbol{P} \circ \boldsymbol{A}|_{\mathcal{X}}(\boldsymbol{X}\boldsymbol{v}) = \boldsymbol{X}\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}\boldsymbol{v}, \quad \forall\, \boldsymbol{v} \in \mathbb{R}^k,$$




故 $\boldsymbol{B}=\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 正好是算子 $\boldsymbol{P}\circ\boldsymbol{A}|_{\mathcal{X}}$ 在给定基 $\boldsymbol{X}$ 之下的矩阵表示. 因此, 通常就称 $\boldsymbol{B}$ 是 $\boldsymbol{A}$ 沿 $\mathcal{Y}^{\perp}$ 到 $\mathcal{X}$ 上的投影.

从实际应用的角度来看, 非对称 Lanczos 方法比 Arnoldi 方法有着很大的优势, 这主要体现在它仅需存储六个 n 维向量 (因为 $k\ll n$, 所以 $\boldsymbol{T}_k$ 的存储量是微不足道的), 并不随着 k 的增加而增加.

但是, 从另一个角度来看, 非对称 Lanczos 算法会发生中断, 即会出现 $\widetilde{\boldsymbol{v}}_{i+1}$ 和 $\widetilde{\boldsymbol{w}}_{i+1}$ 均不为零, 而 $\widetilde{\boldsymbol{w}}_{i+1}^{\mathrm{T}}\widetilde{\boldsymbol{v}}_{i+1}=0$ 的情形. 虽然在实际计算时出现 $\widetilde{\boldsymbol{w}}_{i+1}^{\mathrm{T}}\widetilde{\boldsymbol{v}}_{i+1}=0$ 的概率很小, 但 $|\widetilde{\boldsymbol{w}}_{i+1}^{\mathrm{T}}\widetilde{\boldsymbol{v}}_{i+1}|$ 很小的情况还是经常会遇到的, 此时 $|\gamma_i|$ 和 $|\beta_i|$ 之中有一个就会变得很小, 从而导致在计算过程中引进较大的误差. 解决这一问题的一种方法就是采用 Look–ahead 技术. Look–ahead 方式是基于对算法 4.13 的仔细观察而得到的. 在遇到中断的时候, 虽然不能给出 $\boldsymbol{v}_{i+1}$ 和

$\boldsymbol{w}_{i+1}$, 但却常常可以给出 $\boldsymbol{v}_{i+2}$ 和 $\boldsymbol{w}_{i+2}$. 这就使得 Lanczos 方法可以继续进行下去. Look–ahead 技术虽然解决了非对称 Lanczos 算法的中断问题, 然而付出的代价也是不容忽视的: 一是使得实现过程变得更加复杂; 二是得到的投影矩阵已经不是三对角矩阵, 致使 QMRES 方法的运算复杂性大为增加. 因此, 就求解线性方程组而言, 通常是放弃使用 Look–ahead, 而是在遇到中断时简单地采用重新开始的办法.

§4.5.2 QMRES 方法

从 $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$ 出发, 用算法 4.13 计算得到分解

$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{V}_k\boldsymbol{T}_k + \beta_k\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} = \boldsymbol{V}_{k+1}\widetilde{\boldsymbol{T}}_k \tag{4.178}$$




Back
Close

之后, QMRES 方法的下一步是求解最小二乘问题

$$\min \left\{ \|\beta \boldsymbol{e}_1 - \widetilde{\boldsymbol{T}}_k \boldsymbol{z}\|_2 : \boldsymbol{z} \in \mathbb{R}^k \right\}, \tag{4.179}$$

式中: $\beta = \|\boldsymbol{r}_0\|_2$. 一旦这样的 $\boldsymbol{z}_k$ 求得, 则所寻求的 $\boldsymbol{x}_k$ 就是 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{z}_k$.

极小化问题 (4.179) 是一个系数矩阵为三对角矩阵的最小二乘问题, 仍然用 QR 分解方法来求解.

由于 $\widetilde{\boldsymbol{T}}_k$ 具有式 (4.177) 所示的形状, 故可以计算 k 个 Givens 变换 $\boldsymbol{G}_1, \boldsymbol{G}_2, \cdots, \boldsymbol{G}_k$, 使得

$$\boldsymbol{G}_k \boldsymbol{G}_{k-1} \cdots \boldsymbol{G}_2 \boldsymbol{G}_1 \widetilde{\boldsymbol{T}}_k = \begin{bmatrix} \boldsymbol{R}_k \\ \boldsymbol{O} \end{bmatrix}, \tag{4.180}$$

Back

Close

式中:

$$\boldsymbol{G}_i = \mathrm{diag}\left(\boldsymbol{I}_{i-1}, \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix}, \boldsymbol{I}_{k-i}\right) \in \mathbb{R}^{(k+1)\times(k+1)}, \quad c_i^2 + s_i^2 = 1,$$

$$\boldsymbol{R}_k = \begin{bmatrix} \sigma_1 & \delta_1 & \varepsilon_1 & & \\ & \sigma_2 & \delta_2 & \ddots & \\ & & \ddots & \ddots & \varepsilon_{k-2} \\ & & & \sigma_{k-1} & \delta_{k-1} \\ & & & & \sigma_k \end{bmatrix}, \tag{4.181}$$

而且 $\beta_i \neq 0$ 蕴涵着 $\sigma_i \neq 0$, $i = 1, 2, \cdots, k$, 从而 $\boldsymbol{R}_k$ 是非奇异的.




令

$$\boldsymbol{G}=\boldsymbol{G}_k\boldsymbol{G}_{k-1}\cdots\boldsymbol{G}_2\boldsymbol{G}_1,\quad \begin{bmatrix}\boldsymbol{t}_k\\ \rho_k\end{bmatrix}=\boldsymbol{G}(\beta\boldsymbol{e}_1),\quad \boldsymbol{t}_k=(\tau_1,\tau_2,\cdots,\tau_k)^{\mathrm{T}}, \tag{4.182}$$

则 $\boldsymbol{G}$ 是 $k+1$ 阶正交矩阵, 而且直接计算有

$$\begin{aligned}&\tau_1=\beta c_1,\quad \tau_i=(-1)^{i-1}\beta s_1s_2\cdots s_{i-1}c_i,\quad i=2,3,\cdots,k,\\ &\rho_k=(-1)^k\beta s_1s_2\cdots s_k.\end{aligned} \tag{4.183}$$

这样, 利用式 (4.180) 和式 (4.182), 有

$$\begin{aligned}\|\widetilde{\boldsymbol{T}}_k\boldsymbol{z}-\beta\boldsymbol{e}_1\|_2^2&=\|\boldsymbol{G}(\widetilde{\boldsymbol{T}}_k\boldsymbol{z}-\beta\boldsymbol{e}_1)\|_2^2=\left\|\begin{bmatrix}\boldsymbol{R}_k\\ \boldsymbol{0}\end{bmatrix}\boldsymbol{z}-\begin{bmatrix}\boldsymbol{t}_k\\ \rho_k\end{bmatrix}\right\|_2^2\\ &=\|\boldsymbol{R}_k\boldsymbol{z}-\boldsymbol{t}_k\|_2^2+\rho_k^2\end{aligned}$$

对任意的 $\boldsymbol{z} \in \mathbb{R}^k$ 成立. 由此立即知道, 最小二乘问题 (4.179) 有唯一解

$$\boldsymbol{z}_k = \boldsymbol{R}_k^{-1}\boldsymbol{t}_k, \tag{4.184}$$

而且有

$$\|\widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k - \beta\boldsymbol{e}_1\|_2 = |\rho_k|. \tag{4.185}$$

由式 (4.184) 求得 $\boldsymbol{z}_k$ 之后, 就可算出所求的 $\boldsymbol{x}_k$ 为 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$.

由式 (4.167) 和式 (4.185), 得

$$\|\boldsymbol{r}_k\|_2 = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\|_2 \leqslant \|\boldsymbol{V}_{k+1}\|_2\|\beta\boldsymbol{e}_1 - \widetilde{\boldsymbol{T}}_k\boldsymbol{z}_k\|_2 = \|\boldsymbol{V}_{k+1}\|_2|\rho_k|,$$

故在实际计算时, 可以用

$$|\rho_k|/\beta \leqslant \varepsilon \tag{4.186}$$

作为迭代终止的准则, 其中 $\varepsilon > 0$ 是给定的误差要求.

由式 (4.183) 可知, ρ_k 的值并不需要 $\boldsymbol{z}_k$ 和 $\boldsymbol{x}_k$ 的信息. 因此, 只需在 ρ_k 满足式 (4.186) 之后, 再去计算 $\boldsymbol{z}_k$ 和 $\boldsymbol{x}_k$ 即可.

将式 (4.184) 代入 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{z}_k$, 得

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{R}_k^{-1}\boldsymbol{t}_k = \boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{t}_k, \tag{4.187}$$

式中: $\boldsymbol{P}_k = \boldsymbol{V}_k\boldsymbol{R}_k^{-1}$. 这样, 只要将 $\boldsymbol{P}_k$ 算出, 就可以通过式 (4.187) 来计算 $\boldsymbol{x}_k$. 令 $\boldsymbol{P}_k = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_k]$, 则比较 $\boldsymbol{P}_k\boldsymbol{R}_k = \boldsymbol{V}_k$ 两边的每一列, 得

$$\begin{aligned}
&\sigma_1\boldsymbol{p}_1 = \boldsymbol{v}_1,\\
&\delta_1\boldsymbol{p}_1 + \sigma_2\boldsymbol{p}_2 = \boldsymbol{v}_2,\\
&\varepsilon_{i-2}\boldsymbol{p}_{i-2} + \delta_{i-1}\boldsymbol{p}_{i-1} + \sigma_i\boldsymbol{p}_i = \boldsymbol{v}_i,\ i = 3, 4, \cdots, k.
\end{aligned}$$




由此可求得 $\boldsymbol{P}_k$ 的列向量为

$$\begin{cases} \boldsymbol{p}_1 = \boldsymbol{v}_1/\sigma_1, \\ \boldsymbol{p}_2 = (\boldsymbol{v}_2 - \delta_1\boldsymbol{p}_1)/\sigma_2, \\ \boldsymbol{p}_i = (\boldsymbol{v}_i - \varepsilon_{i-2}\boldsymbol{p}_{i-2} - \delta_{i-1}\boldsymbol{p}_{i-1})/\sigma_i, \quad i = 3, 4, \cdots, k. \end{cases} \tag{4.188}$$

下面借助式 (4.183), 式 (4.187) 和式 (4.188) 导出计算 $\boldsymbol{x}_k$ 的递推公式.

首先注意到, 若非对称 Lanczos 分解的长度由 k 增加到 $k+1$, 则有

$$\widetilde{\boldsymbol{T}}_{k+1} = \left[\begin{array}{c|c} \widetilde{\boldsymbol{T}}_k & \widetilde{\boldsymbol{t}}_{k+1} \\ \hline \boldsymbol{0} & \beta_{k+1} \end{array}\right], \quad \widetilde{\boldsymbol{t}}_{k+1} = (0, \cdots, 0, \gamma_k, \alpha_{k+1})^{\mathrm{T}},$$

于是有

$$\boldsymbol{R}_{k+1} = \left[\begin{array}{c|c} \boldsymbol{R}_k & \widetilde{\boldsymbol{r}}_{k+1} \\ \hline \boldsymbol{0} & \sigma_{k+1} \end{array}\right], \quad \widetilde{\boldsymbol{r}}_{k+1} = (0, \cdots, 0, \varepsilon_{k-1}, \delta_k)^{\mathrm{T}}, \qquad (4.189)$$

式中:

$$\varepsilon_{k-1} = s_{k-1}\gamma_k, \quad \widehat{\gamma}_k = c_{k-1}\gamma_k,$$

$$\delta_k = c_k\widehat{\gamma}_k + s_k\alpha_{k+1}, \quad \widetilde{\alpha}_{k+1} = -s_k\widehat{\gamma}_k + c_k\alpha_{k+1}.$$

上面的四个等式由如下 Givens 变换得到:

$$\widetilde{\boldsymbol{T}}_{k+1} = \left[\begin{array}{ccccccc:c} \alpha_1 & \gamma_1 & & & & & & \\ \beta_1 & \alpha_2 & \gamma_2 & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & \beta_{k-3} & \alpha_{k-2} & \gamma_{k-2} & & & \\ & & & \beta_{k-2} & \alpha_{k-1} & \gamma_{k-1} & & \\ & & & & \beta_{k-1} & \alpha_k & \gamma_k & \\ & & & & & \beta_k & \alpha_{k+1} & \\ \hdashline & & & & & & & \beta_{k+1} \end{array}\right] \xrightarrow{\boldsymbol{G}_1} \left[\begin{array}{ccccccc:c} \sigma_1 & \delta_1 & \varepsilon_1 & & & & & \\ & \widetilde{\alpha}_2 & \widehat{\gamma}_2 & & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & \alpha_{k-2} & \gamma_{k-2} & & & \\ & & & \beta_{k-2} & \alpha_{k-1} & \gamma_{k-1} & & \\ & & & & \beta_{k-1} & \alpha_k & \gamma_k & \\ & & & & & \beta_k & \alpha_{k+1} & \\ \hdashline & & & & & & & \beta_{k+1} \end{array}\right]$$

$$\xrightarrow{\boldsymbol{G}_2} \cdots \xrightarrow{\boldsymbol{G}_{k-2}} \left[\begin{array}{ccccc:c} \sigma_1 & \delta_1 & \varepsilon_1 & & & \\ & \sigma_2 & \delta_2 & \varepsilon_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \sigma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} \\ & & & & \widetilde{\alpha}_{k-1} & \widehat{\gamma}_{k-1} \\ & & & & \beta_{k-1} & \alpha_k & \gamma_k \\ & & & & & \beta_k & \alpha_{k+1} \\ \hdashline & & & & & & \beta_{k+1} \end{array}\right]$$

$$\xrightarrow{\boldsymbol{G}_{k-1}} \left[\begin{array}{ccccc:c} \sigma_1 & \delta_1 & \varepsilon_1 & & & \\ & \sigma_2 & \delta_2 & \varepsilon_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \sigma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} \\ & & & & \sigma_{k-1} & \delta_{k-1} & \varepsilon_{k-1} \\ & & & & & \widetilde{\alpha}_k & \widehat{\gamma}_k \\ & & & & & \beta_k & \alpha_{k+1} \\ \hdashline & & & & & & \beta_{k+1} \end{array}\right] \xrightarrow{\boldsymbol{G}_k} \left[\begin{array}{ccccc:c} \sigma_1 & \delta_1 & \varepsilon_1 & & & \\ & \sigma_2 & \delta_2 & \varepsilon_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \sigma_{k-2} & \delta_{k-2} & \varepsilon_{k-2} \\ & & & & \sigma_{k-1} & \delta_{k-1} & \varepsilon_{k-1} \\ & & & & & \sigma_k & \delta_k \\ & & & & & & \widetilde{\alpha}_{k+1} \\ \hdashline & & & & & & \beta_{k+1} \end{array}\right],$$

即由

$$\begin{bmatrix} c_{k-1} & s_{k-1} \\ -s_{k-1} & c_{k-1} \end{bmatrix} \begin{bmatrix} 0 \\ \gamma_k \end{bmatrix} = \begin{bmatrix} \varepsilon_{k-1} \\ \widehat{\gamma}_k \end{bmatrix},$$

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \widehat{\gamma}_k \\ \alpha_{k+1} \end{bmatrix} = \begin{bmatrix} \delta_k \\ \widetilde{\alpha}_{k+1} \end{bmatrix}$$

得到. 而 σ_{k+1} 是在确定第 $k+1$ 个 Givens 变换 $\boldsymbol{G}_{k+1}$ 时得到的, 即计算 $c_{k+1} = \cos\theta_{k+1}$ 和 $s_{k+1} = \sin\theta_{k+1}$, 使得

$$\begin{bmatrix} c_{k+1} & s_{k+1} \\ -s_{k+1} & c_{k+1} \end{bmatrix} \begin{bmatrix} \widetilde{\alpha}_{k+1} \\ \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \sigma_{k+1} \\ 0 \end{bmatrix}. \tag{4.190}$$

由式 (4.183) 可知 $\boldsymbol{t}_{k+1} = (\boldsymbol{t}_k^{\mathrm{T}}, \tau_{k+1})^{\mathrm{T}}$, 其中

$$\tau_{k+1} = (-1)^k \beta s_1 s_2 \cdots s_k c_{k+1} = \rho_k c_{k+1}, \tag{4.191}$$

而

$$\rho_{k+1} = (-1)^{k+1}\beta s_1 s_2 \cdots s_k s_{k+1} = -\rho_k s_{k+1}. \tag{4.192}$$

由式 (4.188) 和式 (4.189), 有

$$\begin{aligned}
\boldsymbol{P}_{k+1} &= \boldsymbol{V}_{k+1}\boldsymbol{R}_{k+1}^{-1} \\
&= [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}] \left[\begin{array}{c|c} \boldsymbol{R}_k^{-1} & -\boldsymbol{R}_k^{-1}\widetilde{\boldsymbol{r}}_{k+1}/\sigma_{k+1} \\ \hline \boldsymbol{0} & 1/\sigma_{k+1} \end{array}\right] \\
&= [\boldsymbol{P}_k, \boldsymbol{p}_{k+1}],
\end{aligned} \tag{4.193}$$

这里

$$\begin{aligned}
\boldsymbol{p}_{k+1} &= (\boldsymbol{v}_{k+1} - \boldsymbol{V}_k\boldsymbol{R}_k^{-1}\widetilde{\boldsymbol{r}}_{k+1})/\sigma_{k+1} = (\boldsymbol{v}_{k+1} - \boldsymbol{P}_k\widetilde{\boldsymbol{r}}_{k+1})/\sigma_{k+1} \\
&= (\boldsymbol{v}_{k+1} - \varepsilon_{k-1}\boldsymbol{p}_{k-1} - \delta_k\boldsymbol{p}_k)/\sigma_{k+1}.
\end{aligned} \tag{4.194}$$




从而有

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_0 + \boldsymbol{P}_{k+1}\boldsymbol{t}_{k+1} = \boldsymbol{x}_0 + \left[\boldsymbol{P}_k, \boldsymbol{p}_{k+1}\right] \begin{bmatrix} \boldsymbol{t}_k \\ \tau_{k+1} \end{bmatrix} = \boldsymbol{x}_k + \tau_{k+1}\boldsymbol{p}_{k+1}, \tag{4.195}$$

这就得到了 $\boldsymbol{x}_k$ 的递推公式.

算法 4.14 (QMRES 方法) 给定 n 阶非奇异的实矩阵 $\boldsymbol{A}$, n 维向量 $\boldsymbol{b}$, 初始向量 $\boldsymbol{x}_0$ 和允许误差 $\varepsilon > 0$. 本算法计算向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\beta = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{v}_1 = \boldsymbol{r}_0/\beta$;

取一个 $\boldsymbol{w}_1 \in \mathbb{R}^n$, 使得 $\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{v}_1 = 1$ (例如可取 $\boldsymbol{w}_1 = \boldsymbol{v}_1$);

$\alpha_1 = \boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_1$; $\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v}_1 - \alpha_1\boldsymbol{v}_1$; $\boldsymbol{w} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_1 - \alpha_1\boldsymbol{w}_1$;

$\omega = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{v}$;

if $\omega = 0$

$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \boldsymbol{r}_0/\alpha_1$; 结束

else

$\beta_1 = \sqrt{|\omega|}$; $\gamma_1 = \omega/\beta_1$;

$\boldsymbol{v}_2 = \boldsymbol{v}/\beta_1$; $\boldsymbol{w}_2 = \boldsymbol{w}/\gamma_1$;

end

确定 $c_1 = \cos\theta_1$ 和 $s_1 = \sin\theta_1$, 使得

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \sigma_1 \\ 0 \end{bmatrix};$$

$\boldsymbol{p}_1 = \boldsymbol{v}_1/\sigma_1$; $\rho_1 = -\beta s_1$; $\tau_1 = \beta c_1$;

$\boldsymbol{p}_0 = \boldsymbol{0}$; $c_0 = 1$; $s_0 = 0$; $\boldsymbol{x}_1 = \boldsymbol{x}_0 + \tau_1\boldsymbol{p}_1$; $k = 1$;

while $(|\rho_k| > \beta\varepsilon)$

$\alpha_{k+1} = \boldsymbol{w}_{k+1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_{k+1}$;

$\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v}_{k+1} - \alpha_{k+1}\boldsymbol{v}_{k+1} - \gamma_k\boldsymbol{v}_k$;

$\boldsymbol{w} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{w}_{k+1} - \alpha_{k+1}\boldsymbol{w}_{k+1} - \beta_k\boldsymbol{w}_k;$

$\omega_{k+1} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{v};$

if $\omega_{k+1} = 0$

stop

else

$\beta_{k+1} = \sqrt{|\omega_{k+1}|};\ \gamma_{k+1} = \omega_{k+1}/\beta_{k+1};$

$\boldsymbol{v}_{k+2} = \boldsymbol{v}/\beta_{k+1};\ \boldsymbol{w}_{k+2} = \boldsymbol{w}/\gamma_{k+1};$

end

$\varepsilon_{k-1} = s_{k-1}\gamma_k;\ \widehat{\gamma}_k = c_{k-1}\gamma_k;$

$\delta_k = c_k\widehat{\gamma}_k + s_k\alpha_{k+1};\ \widetilde{\alpha}_{k+1} = -s_k\widehat{\gamma}_k + c_k\alpha_{k+1};$

确定 $c_{k+1} = \cos\theta_{k+1}$和 $s_{k+1} = \sin\theta_{k+1}$, 使得

$$\begin{bmatrix} c_{k+1} & s_{k+1} \\ -s_{k+1} & c_{k+1} \end{bmatrix} \begin{bmatrix} \widetilde{\alpha}_{k+1} \\ \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \sigma_{k+1} \\ 0 \end{bmatrix};$$

Back

Close

$$\tau_{k+1} = \rho_k c_{k+1};\ \ \rho_{k+1} = -\rho_k s_{k+1};$$

$$\boldsymbol{p}_{k+1} = (\boldsymbol{v}_{k+1} - \varepsilon_{k-1}\boldsymbol{p}_{k-1} - \delta_k \boldsymbol{p}_k)/\sigma_{k+1};$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \tau_{k+1}\boldsymbol{p}_{k+1};$$

$$k = k + 1;$$

end

注 4.7 (1) 该算法只需存储 10 个 n 维向量即可, 并不随着 k 的增加而增加.

(2) 同样可以考虑预处理 QMRES 方法 (记为 PQMRES). 选定预处理矩阵 $\boldsymbol{M}$, 然后将 QMRES 方法应用到求解方程组

$$\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{M}^{-1}\boldsymbol{b}.$$

这只需要在算法 4.14 中将非对称 Lanczos 过程中的矩阵 $\boldsymbol{A}$ 替换成 $\boldsymbol{M}^{-1}\boldsymbol{A}$ 及初始残差 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$ 替换成 $\boldsymbol{r}_0 = \boldsymbol{M}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$ 即可.

例 4.11 给定线性方程组的系数矩阵 $\boldsymbol{A}$ 和右端项分别为

$$\boldsymbol{A} = \begin{bmatrix} 4 & -1 & & \\ -2 & 4 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -2 & 4 \end{bmatrix}, \quad \boldsymbol{b} = \boldsymbol{A} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix},$$

其中 $n = 1000$. 显然, 该方程组的真解为 $\boldsymbol{x}^* = (1, 1, \cdots, 1)^{\mathrm{T}}$. 应用 QMRES 方法到该方程组上 $(\varepsilon = 10^{-10})$, 迭代在 25 步后满足终止条件, 计算得到的近似解 $\widehat{\boldsymbol{x}} = \boldsymbol{x}_{25}$ 满足终止条件, 但

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 3.0263 \times 10^{-6}, \quad \|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 9.8572 \times 10^{-6}.$$

若选取预处理矩阵为 $\boldsymbol{M} = \mathrm{tril}(\boldsymbol{A})$ (即 $\boldsymbol{A}$ 的下三角部分). 应用 PQMRES 方法到该方程组上 $(\varepsilon = 10^{-10})$, 迭代在 12 步后满足终

止条件, 计算得到的近似解 $\widetilde{\boldsymbol{x}} = \boldsymbol{x}_{12}$ 满足终止条件, 但

$$\|\widetilde{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 3.4207 \times 10^{-5}, \quad \|\boldsymbol{b} - \boldsymbol{A}\widetilde{\boldsymbol{x}}\|_2 = 2.3047 \times 10^{-4}.$$

若将 GMRES 法应用于此例, 迭代在 40 步后满足终止条件, 计算得到的近似解 $\widetilde{\boldsymbol{x}} = \boldsymbol{x}_{40}$ 满足

$$\|\widetilde{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 1.5159 \times 10^{-9}, \quad \|\boldsymbol{b} - \boldsymbol{A}\widetilde{\boldsymbol{x}}\|_2 = 2.1777 \times 10^{-9}.$$

迭代过程的收敛轨迹如图 4.12 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量. 值得注意的是, Jacobi 预条件子对本例无效 (矩阵 $\boldsymbol{A}$ 的对角元素相同), Gauss–Seidel 预条件子虽然对迭代的加速很明显, 但在计算时间上没有任何优势.

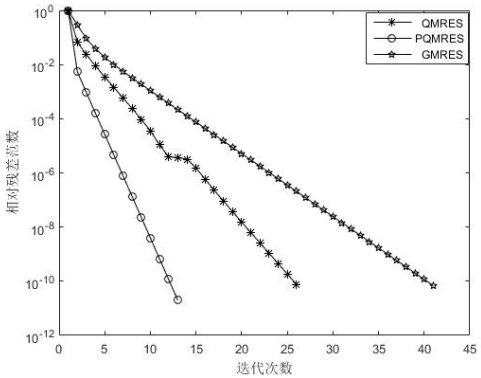


图 4.12 QMRES, PQMRES 和 GMRES 方法的收敛特性

Back
Close

§4.6 LSQR 方法

本节讨论用 LSQR 方法求解 n 阶非对称方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},$$

或当 $\boldsymbol{A} \in \mathbb{R}^{m\times n}\,(m > n)$ 时, 求解最小二乘问题

$$\min \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2. \tag{4.196}$$

这类方法基于 Lanczos 双对角化的思想, 是求解非对称方程组尤其是超定方程组 (线性最小二乘问题) 的一类十分有效的子空间方法.

§4.6.1 Lanczos 双对角化方法

设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}\,(m \geqslant n)$. 给定初始向量 $\boldsymbol{u}_1 \in \mathbb{R}^m$ 满足 $\|\boldsymbol{u}_1\|_2 = 1$. 双对角化方法产生 m 维向量组 $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots$ 和 n 维向量组 $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots$




如下:

$$\alpha_i \boldsymbol{v}_i = \boldsymbol{A}^{\mathrm{T}} \boldsymbol{u}_i - \beta_i \boldsymbol{v}_{i-1} \ (\beta_1 \boldsymbol{v}_0 = \boldsymbol{0}), \tag{4.197}$$

$$\beta_{i+1} \boldsymbol{u}_{i+1} = \boldsymbol{A} \boldsymbol{v}_i - \alpha_i \boldsymbol{u}_i, \quad i = 1, 2, \cdots, \tag{4.198}$$

式中: $\alpha_i, \beta_{i+1} \geqslant 0$, 且使 $\|\boldsymbol{u}_{i+1}\|_2 = \|\boldsymbol{v}_i\|_2 = 1$. 因此, 上述双对角方法可如下实现:

算法 4.15 (Lanczos 双对角化方法) 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{m \times n} (m \geqslant n)$, 初始向量 $\boldsymbol{u}_1 \in \mathbb{R}^m$ 满足 $\|\boldsymbol{u}_1\|_2 = 1$, $\boldsymbol{v}_0 = \boldsymbol{0}$ 和 $\beta_1 = 0$.

for $i = 1 : k$

$\widehat{\boldsymbol{v}}_i = \boldsymbol{A}^{\mathrm{T}} \boldsymbol{u}_i - \beta_i \boldsymbol{v}_{i-1}$;

$\alpha_i = \|\widehat{\boldsymbol{v}}_i\|_2$; $\boldsymbol{v}_i = \widehat{\boldsymbol{v}}_i / \alpha_i$;

$\widehat{\boldsymbol{u}}_{i+1} = \boldsymbol{A} \boldsymbol{v}_i - \alpha_i \boldsymbol{u}_i$;

$\beta_{i+1} = \|\widehat{\boldsymbol{u}}_{i+1}\|_2$; $\boldsymbol{u}_{i+1} = \widehat{\boldsymbol{u}}_{i+1} / \beta_{i+1}$;

end

假设 $\alpha_i, \beta_{i+1} \neq 0, i = 1, 2, \cdots, k$, 则上述 Lanczos 双对角化过程可进行到第 k 步, 且若令

$$\boldsymbol{U}_k = \big[\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k\big], \quad \boldsymbol{V}_k = \big[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\big],$$

$$\boldsymbol{L}_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \end{bmatrix}, \quad \widetilde{\boldsymbol{L}}_k = \begin{bmatrix} \boldsymbol{L}_k \\ \beta_{k+1}\boldsymbol{e}_k^{\mathrm{T}} \end{bmatrix}, \tag{4.199}$$

则式 (4.197) 和式 (4.198) 可表示为

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_k = \boldsymbol{V}_k\boldsymbol{L}_k^{\mathrm{T}}, \quad \boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{U}_k\boldsymbol{L}_k + \beta_{k+1}\boldsymbol{u}_{k+1}\boldsymbol{e}_k^{\mathrm{T}}. \tag{4.200}$$

由此, 得

$$\boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{U}_k\boldsymbol{L}_k + \beta_{k+1}\boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{u}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} = \boldsymbol{L}_k\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k. \tag{4.201}$$




Back
Close

若 $\alpha_{i+1} \neq 0$, 即式 (4.197) 对 $i = k+1$ 成立:

$$\alpha_{k+1}\boldsymbol{v}_{k+1} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_{k+1} - \beta_{k+1}\boldsymbol{v}_k, \tag{4.202}$$

则由上式和式 (4.200), 得

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_{k+1} = \boldsymbol{V}_k\widetilde{\boldsymbol{L}}_k^{\mathrm{T}} + \alpha_{k+1}\boldsymbol{v}_{k+1}\boldsymbol{e}_{k+1}^{\mathrm{T}}, \quad \boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{U}_{k+1}\widetilde{\boldsymbol{L}}_k. \tag{4.203}$$

由此, 得

$$\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_{k+1} = \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k\widetilde{\boldsymbol{L}}_k^{\mathrm{T}} + \alpha_{k+1}\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{v}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} = \widetilde{\boldsymbol{L}}_k^{\mathrm{T}}\boldsymbol{U}_{k+1}^{\mathrm{T}}\boldsymbol{U}_{k+1} \tag{4.204}$$

基于以上分析, 可得下面的定理.

定理 4.11 由 Lanczos 双对角化过程产生的两个向量组 $\{\boldsymbol{u}_i\}_{i=1}^k$ 和 $\{\boldsymbol{v}_i\}_{i=1}^k$ 是规范正交的, 即

$$\boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{U}_k = \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k = \boldsymbol{I}_k, \tag{4.205}$$

式中: $\boldsymbol{I}_k$ 为 k 阶单位矩阵.




Back
Close

证明 对 k 用归纳法. 当 $k=1$ 时, 结论显然为真. 设式 (4.205) 对某个 $k \geqslant 1$ 成立. 利用归纳法假设, 由式 (4.201) 的第 2 个等式可知 $\boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{u}_{k+1}=\boldsymbol{0}$, 即 $\boldsymbol{u}_{k+1}$ 与 $\boldsymbol{u}_i\,(i=1,2,\cdots,k)$ 均正交. 注意到 $\|\boldsymbol{u}_{k+1}\|_2=1$, 即得 $\boldsymbol{U}_{k+1}^{\mathrm{T}}\boldsymbol{U}_{k+1}=\boldsymbol{I}_{k+1}$. 再由式 (4.204) 的第 2 个等式可知 $\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{v}_{k+1}=\boldsymbol{0}$, 即知 $\boldsymbol{V}_{k+1}^{\mathrm{T}}\boldsymbol{V}_{k+1}=\boldsymbol{I}_{k+1}$. 证毕. □

由定理 4.11 可知, Lanczos 双对角化过程必在某个 $k \leqslant \min\{m,n\}$ 中断, 即或在式 (4.200) 中 $\beta_{k+1}\boldsymbol{u}_{k+1}=\boldsymbol{0}$, 或在式 (4.203) 中 $\alpha_{k+1}\boldsymbol{v}_{k+1}=\boldsymbol{0}$. 由此可知, 双对角化算法有两个中断状态:

(1) 由 $k\times k$ 阶矩阵 $\boldsymbol{L}_k$ 来表征 (见式 (4.200)):

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_k=\boldsymbol{V}_k\boldsymbol{L}_k^{\mathrm{T}},\quad \boldsymbol{A}\boldsymbol{V}_k=\boldsymbol{U}_k\boldsymbol{L}_k,\quad \boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{U}_k=\boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k=\boldsymbol{I}_k. \tag{4.206}$$

(2) 由 $(k+1)\times k$ 阶矩阵 $\widetilde{\boldsymbol{L}}_k$ 来表征 (见式 (4.203)):

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_{k+1}=\boldsymbol{V}_k\widetilde{\boldsymbol{L}}_k^{\mathrm{T}},\quad \boldsymbol{A}\boldsymbol{V}_k=\boldsymbol{U}_{k+1}\widetilde{\boldsymbol{L}}_k,$$
$$\boldsymbol{U}_{k+1}^{\mathrm{T}}\boldsymbol{U}_{k+1}=\boldsymbol{I}_{k+1},\quad \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k=\boldsymbol{I}_k. \tag{4.207}$$

定理 4.12 双对角化过程中断状态是 (1) 的充要条件是 $\boldsymbol{u}_1\in\mathcal{R}(\boldsymbol{A})$, 而中断状态是 (2) 的充要条件是 $\boldsymbol{u}_1\notin\mathcal{R}(\boldsymbol{A})$.

证明 考虑 $\widetilde{\boldsymbol{L}}_k$ 的奇异值分解. 注意到 $\widetilde{\boldsymbol{L}}_k$ 是列满秩的, 故有

$$\widetilde{\boldsymbol{L}}_k=\boldsymbol{P}_{k+1}\widetilde{\boldsymbol{\Sigma}}_k\boldsymbol{V}_k^{\mathrm{T}},\quad \boldsymbol{P}_{k+1}^{\mathrm{T}}\boldsymbol{P}_{k+1}=\boldsymbol{I}_{k+1},\quad \boldsymbol{V}_k^{\mathrm{T}}\boldsymbol{V}_k=\boldsymbol{I}_k, \tag{4.208}$$

式中: 矩阵 $\widetilde{\boldsymbol{\Sigma}}_k$ 为 $(k+1)\times k$ 阶对角阵,

$$\widetilde{\boldsymbol{\Sigma}}_k=\begin{bmatrix}\boldsymbol{\Sigma}_k\\ \boldsymbol{0}\end{bmatrix}=\left[\begin{array}{ccc}\sigma_1 & & \\ & \ddots & \\ & & \sigma_k\\ \hdashline 0 & \cdots & 0\end{array}\right],\ \sigma_i>0,\ i=1,2,\cdots,k.$$

若式 (4.206) 成立, 则结合式 (4.207), 得

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}=\boldsymbol{V}_k\boldsymbol{V}_k\widetilde{\boldsymbol{\Sigma}}_k^{\mathrm{T}},\quad \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{V}_k=\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\widetilde{\boldsymbol{\Sigma}}_k. \tag{4.209}$$

故有

$$\begin{aligned}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}&=\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{\Sigma}}_k^{\mathrm{T}},\\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{V}_k&=\boldsymbol{V}_k\boldsymbol{V}_k\widetilde{\boldsymbol{\Sigma}}_k^{\mathrm{T}}\widetilde{\boldsymbol{\Sigma}}_k.\end{aligned} \tag{4.210}$$

由式 (4.210) 可知, 当式 (4.207) 成立时, $\boldsymbol{A}$ 至少有 k 个奇异值 $\sigma_1,\sigma_2,\cdots,\sigma_k$ 非零, 且由式 (4.210) 的第 1 式可知

$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1})=\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{\Sigma}}_k^{\mathrm{T}}\boldsymbol{e}_{k+1}=\boldsymbol{0},$$

即 $\boldsymbol{A}$ 至少有一个奇异值为零, 且 $\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1}$ 是奇异向量. 再由式 (4.209) 的第 1 式可知

$$\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1})=\boldsymbol{V}_k\boldsymbol{V}_k\widetilde{\boldsymbol{\Sigma}}_k^{\mathrm{T}}\boldsymbol{e}_{k+1}=\boldsymbol{0},$$

Back

Close

因此, $\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1} \in \mathcal{N}(\boldsymbol{A}^{\mathrm{T}})$.

若 $\boldsymbol{u}_1 \in \mathcal{R}(\boldsymbol{A})$, 则由式 (4.198) 可知 $\boldsymbol{u}_i \in \mathcal{R}(\boldsymbol{A}), \forall\ i$. 于是有 $\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1} \in \mathcal{R}(\boldsymbol{A})$. 但 $\mathcal{N}(\boldsymbol{A}^{\mathrm{T}})$ 是 $\mathcal{R}(\boldsymbol{A})$ 的正交补, 因此, $\boldsymbol{U}_{k+1}\boldsymbol{P}_{k+1}\boldsymbol{e}_{k+1} \notin \mathcal{N}(\boldsymbol{A}^{\mathrm{T}})$, 这就是说, 式 (4.207) 不可能成立. 所以必成立式 (4.206). 反之, 若 (1) 是中断状态, 即式 (4.206) 成立. 因 $\boldsymbol{L}_k$ 非奇异, 故 $\boldsymbol{U}_k = \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{L}_k^{-1}$, 即 $\boldsymbol{u}_i \in \mathcal{R}(\boldsymbol{A})$, $i = 1, 2, \cdots, k$. 证毕. □

§4.6.2 LSQR 算法

取 $\boldsymbol{u}_1 = \boldsymbol{r}_0/\beta_1$ ($\beta_1 = \|\boldsymbol{r}_0\|_2$) 作为双对角化算法 4.15 中的初始向量, 用以求解最小二乘问题 (4.196). 令

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{y}_k, \tag{4.211}$$

则由式 (4.203), 有

$$\begin{aligned}\boldsymbol{r}_k &= \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k = \boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}_k \\ &= \beta_1\boldsymbol{u}_1 - \boldsymbol{U}_{k+1}\widetilde{\boldsymbol{L}}_k\boldsymbol{y}_k = \boldsymbol{U}_{k+1}(\beta_1\boldsymbol{e}_1 - \widetilde{\boldsymbol{L}}_k\boldsymbol{y}_k).\end{aligned}$$

若令

$$\boldsymbol{s}_{k+1} = \beta_1\boldsymbol{e}_1 - \widetilde{\boldsymbol{L}}_k\boldsymbol{y}_k, \tag{4.212}$$

则

$$\boldsymbol{r}_k = \boldsymbol{U}_{k+1}\boldsymbol{s}_{k+1}. \tag{4.213}$$

因此极小化问题 (4.196) 等价于

$$\min \|\boldsymbol{s}_{k+1}\|_2 = \min \|\beta_1\boldsymbol{e}_1 - \widetilde{\boldsymbol{L}}_k\boldsymbol{y}_k\|_2. \tag{4.214}$$

利用正交化方法求解最小二乘问题 (4.214):

作 $(k+1)\times(k+1)$ 阶矩阵 $\left[\widetilde{\boldsymbol{L}}_k, \beta_1\boldsymbol{e}_1\right]$ 的 QR 分解:

$$\boldsymbol{G}_k\left[\widetilde{\boldsymbol{L}}_k, \beta_1\boldsymbol{e}_1\right] = \begin{bmatrix} \boldsymbol{R}_k & \boldsymbol{z}_k \\ & \widetilde{\zeta}_{k+1} \end{bmatrix} \equiv \left[\begin{array}{ccccc:c} \rho_1 & \theta_2 & & & & \zeta_1 \\ & \rho_2 & \theta_3 & & & \zeta_2 \\ & & \ddots & \ddots & & \vdots \\ & & & \rho_{k-1} & \theta_k & \zeta_{k-1} \\ \hdashline & & & & \rho_k & \zeta_k \\ & & & & & \widetilde{\zeta}_{k+1} \end{array}\right], \tag{4.215}$$

式中: $\boldsymbol{G}_k \equiv \boldsymbol{G}_{k,k+1}\cdots\boldsymbol{G}_{2,3}\boldsymbol{G}_{1,2}$ 为 Givens 变换的乘积, 用以消去 $\widetilde{\boldsymbol{L}}_k$ 的次对角元 $\beta_2, \beta_3, \cdots$.

向量 $\boldsymbol{y}_k$ 和 $\boldsymbol{s}_{k+1}$ 可分别由

$$\boldsymbol{R}_k\boldsymbol{y}_k = \boldsymbol{z}_k, \quad \boldsymbol{z}_k = (\zeta_1, \zeta_2, \cdots, \zeta_k)^{\mathrm{T}} \tag{4.216}$$

和

$$\boldsymbol{s}_{k+1} = \boldsymbol{G}_k^{\mathrm{T}}\begin{bmatrix} \boldsymbol{0} \\ \widetilde{\zeta}_{k+1} \end{bmatrix} \tag{4.217}$$

求得.

直接求解式 (4.216), $\boldsymbol{y}_k$ 与 $\boldsymbol{y}_{k-1}$ 一般并无公共元素, 注意到 $[\boldsymbol{R}_k, \boldsymbol{z}_k]$ 可由 $[\boldsymbol{R}_{k-1}, \boldsymbol{z}_{k-1}]$ 加入新的一行和一列得到. 由此, 式 (4.211) 与式 (4.216) 有效结合的一个途径是引进矩阵 $\boldsymbol{H}_k$:

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k \boldsymbol{R}_k^{-1} \boldsymbol{z}_k \equiv \boldsymbol{x}_0 + \boldsymbol{H}_k \boldsymbol{z}_k, \tag{4.218}$$

其中 $\boldsymbol{H}_k = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_k]$ 的列可由 $\boldsymbol{H}_k \boldsymbol{R}_k = \boldsymbol{V}_k$ 逐列求得: 令 $\boldsymbol{h}_0 = \boldsymbol{0}$, 则成立

$$\theta_k \boldsymbol{h}_{k-1} + \rho_k \boldsymbol{h}_k = \boldsymbol{v}_k,$$

由此可得到递推式

$$\boldsymbol{h}_k = \frac{1}{\rho_k}(\boldsymbol{v}_k - \theta_k \boldsymbol{h}_{k-1}). \tag{4.219}$$

则由式 (4.218), 得

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \sum_{i=1}^{k-1} \zeta_i \boldsymbol{h}_i + \zeta_k \boldsymbol{h}_k = \boldsymbol{x}_{k-1} + \zeta_k \boldsymbol{h}_k. \tag{4.220}$$

现在需要确定式 (4.215) 的 QR 分解. Givens 变换 $\boldsymbol{G}_{k,k+1}$ 作用于前面 $k-1$ 步变换过的矩阵 $\left[\widetilde{\boldsymbol{L}}_k, \beta_1 \boldsymbol{e}_1\right]$ 以消去 β_{k+1}, 这给出如下递推关系

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \widetilde{\rho}_k & 0 & \widetilde{\zeta}_k \\ \beta_{k+1} & \alpha_{k+1} & 0 \end{bmatrix} = \begin{bmatrix} \rho_k & \theta_{k+1} & \zeta_k \\ 0 & \widetilde{\rho}_{k+1} & \widetilde{\zeta}_{k+1} \end{bmatrix}, \quad (4.221)$$

其中 $\widetilde{\rho}_1 \equiv \alpha_1$, $\widetilde{\zeta}_1 \equiv \beta_1$. c_k, s_k 是 Givens 变换 $\boldsymbol{G}_{k,k+1}$ 中的元素. $\widetilde{\rho}_k$ 和 $\widetilde{\zeta}_k$ 是中间变量, 会被 ρ_k 和 ζ_k 所取代. 由 Givens 变换的定义, c_k, s_k 的选取应使得对向量 $[\widetilde{\rho}_k, \beta_{k+1}]^{\mathrm{T}}$ 作用后第 2 个分量为零, 且满足 $c_k^2 + s_k^2 = 1$. 为此, 必有

$$c_k = \frac{\widetilde{\rho}_k}{\sqrt{\widetilde{\rho}_k^2 + \beta_{k+1}^2}}, \quad s_k = \frac{\beta_{k+1}}{\sqrt{\widetilde{\rho}_k^2 + \beta_{k+1}^2}}. \quad (4.222)$$




由此, 式 (4.221) 即为

$$\begin{cases} \rho_k = \sqrt{\widetilde{\rho}_k^2 + \beta_{k+1}^2}, \ \theta_{k+1} = s_k \alpha_{k+1}, \ \zeta_k = c_k \widetilde{\zeta}_k, \\ \widetilde{\rho}_{k+1} = c_k \alpha_{k+1}, \ \widetilde{\zeta}_{k+1} = -s_k \widetilde{\zeta}_k, \ \widetilde{\rho}_1 \equiv \alpha_1, \ \widetilde{\zeta}_1 \equiv \beta_1. \end{cases} \tag{4.223}$$

此外, $\boldsymbol{G}_{k,k+1}$ 在式 (4.221) 中使用完之后即可丢弃而不必存储. 因此由 QR 分解算法产生 $\boldsymbol{R}_k$, $\boldsymbol{h}_k$ 和 $\widetilde{\zeta}_{k+1}$ 的工作量是很小的.

现将 LSQR 算法总结如下, 其中 $\boldsymbol{h}_k$ 用 $\boldsymbol{z}_k = \rho_k \boldsymbol{h}_k$ 代替.

算法 4.16 (LSQR 算法) 给定最小二乘问题 (4.196) 和容许误差限 $\varepsilon > 0$, 取初始向量 $\boldsymbol{x}_0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2 / \|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\beta_1 = \|\boldsymbol{r}_0\|_2$; $\boldsymbol{u}_1 = \boldsymbol{r}_0 / \beta_1$;

$\widehat{\boldsymbol{v}}_1 = \boldsymbol{A}^{\mathrm{T}} \boldsymbol{u}_1$; $\alpha_1 = \|\widehat{\boldsymbol{v}}_1\|_2$; $\boldsymbol{v}_1 = \widehat{\boldsymbol{v}}_1 / \alpha_1$;

$\boldsymbol{z}_1 = \boldsymbol{v}_1$; $\widetilde{\zeta}_1 = \beta_1$; $\widetilde{\rho}_1 = \alpha_1$; $k = 1$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

① 双对角化

$$\widehat{\boldsymbol{u}}_{k+1} = \boldsymbol{A}\boldsymbol{v}_k - \alpha_k \boldsymbol{u}_k;$$

$$\beta_{k+1} = \|\widehat{\boldsymbol{u}}_{k+1}\|_2;\ \boldsymbol{u}_{k+1} = \widehat{\boldsymbol{u}}_{k+1}/\beta_{k+1};$$

$$\widehat{\boldsymbol{v}}_{k+1} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_{k+1} - \beta_{k+1}\boldsymbol{v}_k;$$

$$\alpha_{k+1} = \|\widehat{\boldsymbol{v}}_{k+1}\|_2;\ \boldsymbol{v}_{k+1} = \widehat{\boldsymbol{v}}_{k+1}/\alpha_{k+1};$$

② 构造和使用正交变换

$$\rho_k = \sqrt{\widetilde{\rho}_k^2 + \beta_{k+1}^2};\quad c_k = \widetilde{\rho}_k/\rho_k;\ s_k = \beta_{k+1}/\rho_k;$$

$$\theta_{k+1} = s_k\alpha_{k+1};\ \widetilde{\rho}_{k+1} = c_k\alpha_{k+1};$$

$$\zeta_k = c_k\widetilde{\zeta}_k;\ \widetilde{\zeta}_{k+1} = -s_k\widetilde{\zeta}_k;$$

③ 更新 $\boldsymbol{x}_k$, $\boldsymbol{r}_k$ 和 $\boldsymbol{z}_k$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + (\zeta_k/\rho_k)\boldsymbol{z}_k;$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{k+1} = \boldsymbol{r}_k - (\zeta_k/\rho_k)\boldsymbol{A}\boldsymbol{z}_k;$$

$$\boldsymbol{z}_{k+1} = \boldsymbol{v}_{k+1} - (\theta_{k+1}/\rho_k)\boldsymbol{z}_k;$$

④ 置 $k = k + 1$;

end

注 4.8 由双对角化过程可知

$$\begin{aligned}\boldsymbol{v}_k &\in \mathrm{span}\{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_1, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_1, \cdots, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{k-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_1\} \\ &\equiv \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_1).\end{aligned}$$

因此, 由式 (4.211) 和 $\boldsymbol{u}_1 = \boldsymbol{r}_0/\beta_1$ 可知 LSQR 算法产生的迭代序列 $\boldsymbol{x}_k$ 满足

$$\begin{aligned}\boldsymbol{x}_k &\in \boldsymbol{x}_0 + \mathrm{span}\{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, \cdots, (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{k-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0\} \\ &\equiv \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0), \qquad (4.224)\end{aligned}$$

且使

$$\|\boldsymbol{r}_k\|_2 = \|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k\|_2 = \min\left\{\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2 : \boldsymbol{x} \in \boldsymbol{x}_0+\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{b})\right\}. \tag{4.225}$$

LSQR 方法的 MATLAB 程序如下:

```
%LSQR方法程序-mlsqr.m
function [x,k,time,res,resvec]=mlsqr(A,b,x,max_it,tol)
tic; r=b-A*x; mr=norm(r); u=r/mr;
v=A'*u; alpha=norm(v); v=v/alpha;
z=v; zetat=mr; rhot=alpha;
resvec(1)=1; k=0;
while (k<=max_it)
    k=k+1;
```

```
    u=A*v-alpha*u; beta=norm(u); u=u/beta;
    v=A'*u-beta*v; alpha=norm(v); v=v/alpha;
    rho=sqrt(rhot^2+beta^2);
    c=rhot/rho; s=beta/rho;
    theta=s*alpha; rhot=c*alpha;
    zeta=c*zetat; zetat=-s*zetat;
    x=x+(zeta/rho)*z; r=b-A*x;
    z=v-(theta/rho)*z;
    res=norm(r)/mr; resvec(k+1)=res;
    if (res<tol), break; end
end
time=toc;
```

例 4.12 假设线性方程组的系数矩阵 $\boldsymbol{A}$ 为

$$\boldsymbol{A} = \begin{bmatrix} 4 & -1 & & & \\ -2 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -2 & 4 & -1 \\ & & & -2 & 4 \end{bmatrix}.$$

再假设真解 $\boldsymbol{x}^*$ 是分量都为 1 的向量, 从而该方程组的右端项为 $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}^*$. 将 LSQR 算法应用到该线性方程组上, 迭代在 76 步后收敛 ($\varepsilon = 10^{-10}$), 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的绝对值误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 1.8330 \times 10^{-9},$$

但计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 2.5663 \times 10^{-9}.$$

此外, 将 GMRES 方法应用于求解此例, 迭代 40 步满足终止准则. 对于此例, 虽然 LSQR 方法的迭代次数比 GMRES 方法多了将近一倍, 但计算时间不到 GMRES 方法的一半. 迭代过程的收敛轨迹如图 4.13 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

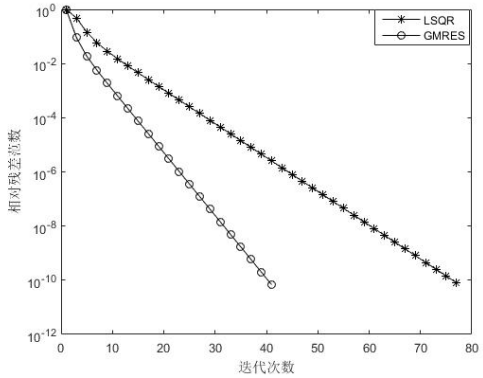


图 4.13 LSQR 算法的收敛特性

§4.7 广义共轭残量法

本节考虑求解线性方程组

$$\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b},\quad \boldsymbol{A}\in\mathbb{R}^{n\times n},\quad \boldsymbol{b}\in\mathbb{R}^{n} \tag{4.226}$$

的广义共轭残量法 (简记为 GCR 方法), 其中系数矩阵 $\boldsymbol{A}$ 是正定的, 即其对称部分 $\frac{1}{2}\left(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}}\right)$ 是对称正定的.

§4.7.1 GCR 方法

类似于对称正定方程组的共轭梯度法, 构造如下的广义共轭残量法.

算法 4.17 (GCR 方法) 给定方程组 (4.226) 和容许误差限 $\varepsilon>0$, 取初始向量 $\boldsymbol{x}_0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2\leqslant\varepsilon$, 其中 $\boldsymbol{r}_k=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{r}_0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

$k = k + 1$;

$$\alpha_k = \frac{(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k)}{(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)};$$

$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$; $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k$;

for $i = 0, 1, \cdots, k$

$$\beta_i^{(k)} = -\frac{(\boldsymbol{A}\boldsymbol{r}_{k+1}, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)};$$

end

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \sum_{i=0}^{k} \beta_i^{(k)} \boldsymbol{p}_i;$$

$$\boldsymbol{A}\boldsymbol{p}_{k+1} = \boldsymbol{A}\boldsymbol{r}_{k+1} + \sum_{i=0}^{k} \beta_i^{(k)} \boldsymbol{A}\boldsymbol{p}_i;$$

end

从算法 4.17 可知, 计算过程中需要存储每一步迭代的 $\boldsymbol{p}_i$ 和 $\boldsymbol{A}\boldsymbol{p}_i\,(i=0,1,\cdots,k)$. 其 MATLAB 程序如下:

```
%广义共轭残量法(GCR方法)程序-gcr.m
function [x,k,time,res,resvec]=gcr(A,b,x,max_it,tol)
tic; n=length(b); r=b-A*x; p=r; mr=norm(r);
resvec(1)=1; AP=zeros(n,max_it+1);
P=zeros(n,max_it+1);
AP(:,1)=A*p; P(:,1)=p; k=0;
while (k<max_it)
    k=k+1;
    alpha=(r'*AP(:,k))/(AP(:,k)'*AP(:,k));
    x=x+alpha*P(:,k);
    r=r-alpha*AP(:,k);
```

```
    Ar=A*r; s1=zeros(n,1); s2=zeros(n,1);
    for (i=1:k)
        b(i)=-(Ar'*AP(:,i))/(AP(:,i)'*AP(:,i));
        s1=s1+b(i)*P(:,i);
        s2=s2+b(i)*AP(:,i);
    end
    P(:,k+1)=r+s1;
    AP(:,k+1)=Ar+s2;
    res=norm(r)/mr; resvec(k+1)=res;
    if (res<tol), break; end
end
time=toc;
```

下面列出一些与共轭梯度法 (CG) 相类似的性质.




Back
Close

定理 4.13 设 $\boldsymbol{A}$ 正定, 即 $\boldsymbol{H}=\frac{1}{2}(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})$ 是对称正定的. 则由算法 4.17 产生的 $\{\boldsymbol{x}_k\}$, $\{\boldsymbol{r}_k\}$ 和 $\{\boldsymbol{p}_k\}$ 具有下列性质:

(1) $(\boldsymbol{A}\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_l)=0,\ k\neq l.$

(2) $(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_l)=0,\ k>l.$

(3) $(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_l)=(\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_l),\ k\leqslant l.$

(4) $(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k)=(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_k).$

(5) $(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_l)=0,\ k>l.$

(6) $\mathrm{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}=\mathrm{span}\{\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0, \cdots, \boldsymbol{A}^k\boldsymbol{p}_0\}$
$=\mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^k\boldsymbol{r}_0\}=\mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k\}.$

(7) 若 $\boldsymbol{r}_k\neq\boldsymbol{0}$, 则 $\boldsymbol{p}_k\neq\boldsymbol{0}$.

(8) $\boldsymbol{x}_{k+1}\in\mathcal{K}\equiv\boldsymbol{x}_0+\mathrm{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}$, 且使其残量范数 $f(\boldsymbol{x}_{k+1})\equiv\|\boldsymbol{r}_{k+1}\|_2=\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_{k+1}\|_2$ 在仿射空间 $\mathcal{K}$ 中达到极小.




Back
Close

证明 (1) 首先验证 $(\boldsymbol{Ap}_1, \boldsymbol{Ap}_0) = 0$ 成立. 事实上, 由 $\beta_i^{(k)}$ 更新规则, 有

$$(\boldsymbol{Ap}_1, \boldsymbol{Ap}_0) = \big(\boldsymbol{A}(\boldsymbol{r}_1 + \beta_0^{(0)}\boldsymbol{p}_0), \boldsymbol{Ap}_0\big) = \big(\boldsymbol{Ar}_1, \boldsymbol{Ap}_0\big) + \beta_0^{(0)}\big(\boldsymbol{Ap}_0, \boldsymbol{Ap}_0\big)$$
$$= \big(\boldsymbol{Ar}_1, \boldsymbol{Ap}_0\big) - \frac{(\boldsymbol{Ar}_1, \boldsymbol{Ap}_0)}{(\boldsymbol{Ap}_0, \boldsymbol{Ap}_0)}\big(\boldsymbol{Ap}_0, \boldsymbol{Ap}_0\big) = 0.$$

利用归纳法, 由 $\beta_i^{(k)}$ 的取法不难验证, 对任意的 $k \neq l$ 都有

$$(\boldsymbol{Ap}_k, \boldsymbol{Ap}_l) = 0.$$

(2) 对 k 使用归纳法. 当 $k = 0$ 时无需证明. 设 $k \leqslant s$ 时, 结论为真, 则

$$(\boldsymbol{r}_{s+1}, \boldsymbol{Ap}_l) = (\boldsymbol{r}_s, \boldsymbol{Ap}_l) - \alpha_s(\boldsymbol{Ap}_s, \boldsymbol{Ap}_l).$$

若 $s > l$, 由归纳法假设及结论 (1), 上式右端为 0. 若 $s = l$, 由 α_s 的表达式, 上式右端也为 0.

(3) 对 $k\,(k \leqslant l)$ 使用归纳法证明. 当 $k = 0$ 时, 结论平凡地成立. 设 $k = s < l$ 时结论成立, 则由结论 (1) 和归纳法假设, 有

$$\big(\boldsymbol{r}_{s+1}, \boldsymbol{A}\boldsymbol{p}_l\big) = \big(\boldsymbol{r}_s, \boldsymbol{A}\boldsymbol{p}_l\big) - \alpha_s\big(\boldsymbol{A}\boldsymbol{p}_s, \boldsymbol{A}\boldsymbol{p}_l\big) = \big(\boldsymbol{r}_s, \boldsymbol{A}\boldsymbol{p}_l\big) = \big(\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_l\big).$$

(4) 利用结论 (2), 有

$$(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k) = (\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_k) + \sum_{i=0}^{k-1} \beta_i^{(k-1)} (\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_i) = (\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_k).$$

(5) 由算法 4.17, 有

$$\boldsymbol{r}_l = \boldsymbol{p}_l - \sum_{i=0}^{l-1} \beta_i^{(l-1)} \boldsymbol{p}_i.$$

应用结论 (2), 对 $k > l$, 有

$$\big(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_l\big) = \big(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_l\big) - \sum_{i=0}^{l-1} \beta_i^{(l-1)} \big(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_i\big) = 0.$$




(6) 对 k 用归纳法证明. 当 $k=0$ 时, 结论显然成立. 设对 $k \leqslant s$ 结论成立, 即

$$\boldsymbol{r}_i, \boldsymbol{p}_i \in \operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0, \cdots, \boldsymbol{A}^s\boldsymbol{p}_0\}, \ i = 0, 1, \cdots, s.$$

由算法 4.17, 有

$$\boldsymbol{p}_{s+1} = \boldsymbol{r}_{s+1} + \sum_{i=0}^{s} \beta_i^{(s)} \boldsymbol{p}_i = \boldsymbol{r}_s - \alpha_s \boldsymbol{A}\boldsymbol{p}_s + \sum_{i=0}^{s} \beta_i^{(s)} \boldsymbol{p}_i,$$

故有

$$\boldsymbol{p}_{s+1} \in \operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0, \cdots, \boldsymbol{A}^{s+1}\boldsymbol{p}_0\}.$$

由此, 得

$$\operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{s+1}\} \subset \operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0, \cdots, \boldsymbol{A}^{s+1}\boldsymbol{p}_0\}.$$

但由结论 (1) 可知 $\{\boldsymbol{p}_i\}$ 线性无关, 故

$$\operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{s+1}\} = \operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_0, \cdots, \boldsymbol{A}^{s+1}\boldsymbol{p}_0\}.$$




同理, 可证

$$\operatorname{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k\} \subset \operatorname{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^k\boldsymbol{r}_0\}.$$

(7) 注意到 $\boldsymbol{H} = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}})$ 是对称正定的, 若 $\boldsymbol{r}_k \neq \boldsymbol{0}$, 则由结论 (4), 有

$$(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k) = (\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{r}_k) = (\boldsymbol{r}_k, \boldsymbol{H}\boldsymbol{r}_k) > 0,$$

故 $(\boldsymbol{r}_k, \boldsymbol{A}\boldsymbol{p}_k) \neq 0$, 由此 $\boldsymbol{p}_k \neq \boldsymbol{0}$.

(8) 由算法 4.17, 有

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_0 + \sum_{i=0}^{k} \alpha_i \boldsymbol{p}_i \in \boldsymbol{x}_0 + \operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}.$$

因此 $f^2(\boldsymbol{x}_{k+1}) = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{k+1}\|_2^2$ 是 $(\alpha_0, \alpha_1, \cdots, \alpha_k)$ 的二次泛函. 应




用结论 (1), 得

$$f^2(\boldsymbol{x}_{k+1}) = \left\| \boldsymbol{r}_0 - \sum_{i=0}^{k} \alpha_i \boldsymbol{A}\boldsymbol{p}_i \right\|_2^2$$

$$= \|\boldsymbol{r}_0\|_2^2 - 2\sum_{i=0}^{k} \alpha_i (\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_i) + \sum_{i=0}^{k} \alpha_i^2 (\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i).$$

由 $\dfrac{\partial f^2(\boldsymbol{x}_{k+1})}{\partial \alpha_i} = 0\,(i = 0, 1, \cdots, k)$, 得

$$\alpha_i = \frac{(\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)} = \frac{(\boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)}, \quad i = 0, 1, \cdots, k.$$

这就证明了 $f(\boldsymbol{x}_{k+1})$ 在仿射空间 $\mathcal{K}$ 中被极小化. 证毕. □

由定理 4.13 可推得 GCR 方法的有限迭代终止性质.

推论 4.5 设矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是正定的 (即其对称部分是对称正定的), 则 GCR 方法至多迭代 n 步就给出精确解.

证明 若当 $k \leqslant n-1$ 时有 $\boldsymbol{r}_k = \boldsymbol{0}$, 这表明 $\boldsymbol{x}_k$ 已是精确解. 若 $\boldsymbol{r}_k \neq \boldsymbol{0}$, $k = 0, 1, \cdots, n-1$, 则由定理 4.13 的结论 (7) 可知 $\boldsymbol{p}_k \neq \boldsymbol{0}$. 再由结论 (1) 知向量组 $\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{n-1}$ 线性无关, 因此 $\operatorname{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_{n-1}\} = \mathbb{R}^n$. 再由定理 4.13 的结论 (8), $\boldsymbol{x}_n$ 的残量范数有下述极小化性质:

$$\|\boldsymbol{r}_n\|_2 = \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_n\|_2 = \min_{\alpha_i} \left\| \boldsymbol{r}_0 - \sum_{k=0}^{n-1} \alpha_k \boldsymbol{A}\boldsymbol{p}_k \right\|_2 .$$

因 $\boldsymbol{A}$ 非奇异, 故 $\operatorname{span}\{\boldsymbol{A}\boldsymbol{p}_0, \boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{n-1}\} = \mathbb{R}^n$. 从而 $\|\boldsymbol{r}_n\|_2 = 0$, 故 $\boldsymbol{x}_n = \boldsymbol{A}^{-1}\boldsymbol{b}$. 证毕. □

现在考虑 GCR 方法作为迭代法时的迭代误差估计和收敛性. 令 $\mathcal{P}_k$ 是次数不超过 k 且满足 $p_k(0) = 1$ 的实系数多项式 $p_k(t)$ 的集合.

定理 4.14 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 正定, 即 $\boldsymbol{H} = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}})$ 是对称正定的. 则由算法 4.17 产生的残量序列 $\{\boldsymbol{r}_k\}$ 满足

$$\|\boldsymbol{r}_k\|_2 \leqslant \min_{p_k\in\mathcal{P}_k} \|p_k(\boldsymbol{A})\|_2\|\boldsymbol{r}_0\|_2 \leqslant \left[1 - \frac{\lambda_{\min}^2(\boldsymbol{H})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}\right]^{k/2} \|\boldsymbol{r}_0\|_2. \tag{4.227}$$

因此, GCR 方法收敛. 若 $\boldsymbol{A}$ 有完全特征向量集, 则

$$\|\boldsymbol{r}_k\|_2 \leqslant \kappa_2(\boldsymbol{T})M_k\|\boldsymbol{r}_0\|_2,$$

式中: $\boldsymbol{T}$ 为 $\boldsymbol{A}$ 的 Jordan 标准形的变换矩阵, 而

$$M_k = \min_{p_k\in\mathcal{P}_k} \max_{\lambda\in\sigma(A)} |p_k(\lambda)|.$$

进一步, 若 $\boldsymbol{A}$ 是规范的, 则

$$\|\boldsymbol{r}_k\|_2 \leqslant M_k\|\boldsymbol{r}_0\|_2.$$




Back
Close

证明 由算法 4.17 和定理 4.13 的结论 (6), GCR 方法产生的残量 $\boldsymbol{r}_k$ 可表示为 $\boldsymbol{r}_k = \widetilde{p}_k(\boldsymbol{A})\boldsymbol{r}_0$, $\widetilde{p}_k \in \mathcal{P}_k$. 而任意的 $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\}$, 其对应的残量 $\boldsymbol{r}$ 可表示为 $\boldsymbol{r} = p_k(\boldsymbol{A})\boldsymbol{r}_0$, $p_k \in \mathcal{P}_k$. 于是由定理 4.13 的结论 (8), 有

$$\|\boldsymbol{r}_k\|_2 = \min_{p_k \in \mathcal{P}_k} \|p_k(\boldsymbol{A})\boldsymbol{r}_0\|_2 \leqslant \min_{p_k \in \mathcal{P}_k} \|p_k(\boldsymbol{A})\|_2 \|\boldsymbol{r}_0\|_2,$$

这就证明了式 (4.227) 的第 1 个不等式.

现证明第 2 个不等式. 令

$$p_1(t) = 1 + \alpha t \in \mathcal{P}_1,$$

则 $p_1(t)^k \in \mathcal{P}_k$. 因此

$$\min_{p_k \in \mathcal{P}_k} \|p_k(\boldsymbol{A})\|_2 \leqslant \|p_1(\boldsymbol{A})^k\|_2 \leqslant \|p_1(\boldsymbol{A})\|_2^k. \tag{4.228}$$

下面估计 $\|p_1(\boldsymbol{A})\|_2$. 有

$$\begin{aligned}\|p_1(\boldsymbol{A})\|_2^2 &= \max_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{((\boldsymbol{I}+\alpha\boldsymbol{A})\boldsymbol{x},(\boldsymbol{I}+\alpha\boldsymbol{A})\boldsymbol{x}))}{(\boldsymbol{x},\boldsymbol{x})} \\ &= \max_{\boldsymbol{x}\neq\boldsymbol{0}} \left\{1+2\alpha\frac{(\boldsymbol{x},\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}+\alpha^2\frac{(\boldsymbol{A}\boldsymbol{x},\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}\right\}. \qquad (4.229)\end{aligned}$$

注意到

$$\frac{(\boldsymbol{A}\boldsymbol{x},\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}=\frac{(\boldsymbol{x},\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}\leqslant\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})=\|\boldsymbol{A}\|_2^2.$$

利用 $\boldsymbol{H}$ 的对称正定性, 有

$$\frac{(\boldsymbol{x},\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}=\frac{1}{2}\left[\frac{(\boldsymbol{x},\boldsymbol{A}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}+\frac{(\boldsymbol{x},\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}\right]=\frac{(\boldsymbol{x},\boldsymbol{H}\boldsymbol{x})}{(\boldsymbol{x},\boldsymbol{x})}\geqslant\lambda_{\min}(\boldsymbol{H})>0.$$

若选取 $\alpha<0$, 则由式 (4.229), 得

$$\|p_1(\boldsymbol{A})\|_2^2\leqslant 1+2\lambda_{\min}(\boldsymbol{H})\,\alpha+\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\,\alpha^2. \qquad (4.230)$$

上式右端当

$$\alpha = -\frac{\lambda_{\min}(\boldsymbol{H})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}$$

时达到极小. 将这个 α 值代入式 (4.230), 得

$$\|p_1(\boldsymbol{A})\|_2^2 \leqslant 1 - \frac{\lambda_{\min}^2(\boldsymbol{H})}{\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}.$$

由上式及式 (4.228) 即得式 (4.227) 的第 2 个不等式.

若 $\boldsymbol{A}$ 有完全特征向量集, 即 $\boldsymbol{A}$ 可对角化, 则 $\boldsymbol{\Lambda} = \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}$ 是对角矩阵, 则

$$\begin{aligned}\|\boldsymbol{r}_k\|_2 &= \min_{p_k\in\mathcal{P}_k} \|\boldsymbol{T}p_k(\boldsymbol{\Lambda})\boldsymbol{T}^{-1}\boldsymbol{r}_0\|_2 \leqslant \|\boldsymbol{T}\|_2\|\boldsymbol{T}^{-1}\|_2 \min_{p_k\in\mathcal{P}_k} \|p_k(\boldsymbol{\Lambda})\|_2\|\boldsymbol{r}_0\|_2 \\ &= \kappa_2(\boldsymbol{T}) \min_{p_k\in\mathcal{P}_k} \max_{\lambda\in\sigma(\boldsymbol{A})} |p_k(\lambda)| \cdot \|\boldsymbol{r}_0\|_2.\end{aligned}$$

进一步, 若 $\boldsymbol{A}$ 还是规范的, 则可使 $\kappa_2(\boldsymbol{T}) = 1$. 证毕. □




Back
Close

例 4.13 考虑正定方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$, 其中

$$\boldsymbol{A}=\begin{bmatrix} 12 & 3 & 2 & 1 & & & \\ -3 & 12 & 3 & 2 & 1 & & \\ -2 & -3 & \ddots & \ddots & \ddots & \ddots & \\ -1 & -2 & \ddots & \ddots & \ddots & \ddots & 1 \\ & \ddots & \ddots & \ddots & \ddots & 3 & 2 \\ & & -1 & -2 & -3 & 12 & 3 \\ & & & -1 & -2 & -3 & 12 \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad \boldsymbol{b}=\boldsymbol{A}\begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ 1 \\ 1 \end{bmatrix} \in \mathbb{R}^{n}.$$

取 $n=1000$, 将 GCR 方法应用到该线性方程组上, 迭代在 20 步后收敛 $(\varepsilon=10^{-10})$, 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的绝对值误差为

$$\|\widehat{\boldsymbol{x}}-\boldsymbol{x}^*\|_2=2.0725\times 10^{-9},$$

但计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b}-\boldsymbol{A}\widehat{\boldsymbol{x}}\|_2=2.7746\times 10^{-8}.$$




Back
Close

迭代过程的收敛轨迹如图 4.14 所示, 其中横坐标为迭代步数 k, 纵

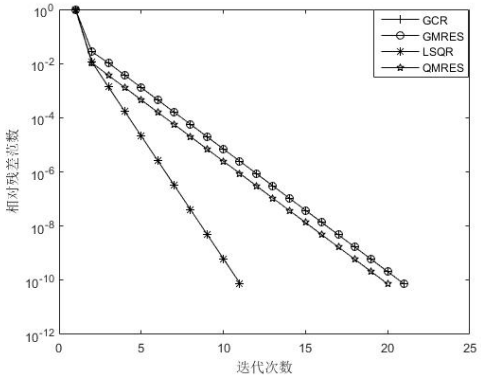


图 4.14 GCR 方法的收敛特性

坐标为 $\lg \|\boldsymbol{r}_k\|_2$, 其中 $\boldsymbol{r}_k$ 是第 k 步得到的残量. 此外, 还给出了 GCR 方法与 GMRES, LSQR 及 QMRES 方法的比较, 结果如表 4.3 所示.

从表中可以看出, 对于此例, GCR 算法不如 LSQR 算法有效.

表 4.3 GCR 方法的数值结果

算法	迭代次数	CPU时间	相对残差	绝对误差
GCR	20	0.0028	7.3094e-11	2.0725e-09
GMRES	20	0.0018	7.3094e-11	2.0725e-09
LSQR	10	0.0007	7.7475e-11	2.1967e-09
QMRES	19	0.0010	7.3094e-11	2.0725e-09

§4.7.2 GCR(m) 方法

注意到算法 4.17 (GCR 方法) 的执行需要存储所有的方向向量 $\boldsymbol{p}_k$. 如果在 GCR 方法中周期地重开始: 每 $m+1$ 次迭代, 用当前的第 i 次重开始的结果 $\boldsymbol{x}_{i(m+1)}$ 作为新的 $i+1$ 次重开始的初始向量, 这时只要存储 m 个方向向量. 这个重开始 CGR 方法称为 GCR(m). 下面给出重开始 GCR 方法的详细算法步骤.

算法 4.18 (GCR(m) 方法) 给定方程组 (4.226) 和容许误差限 $\varepsilon > 0$, 取初始向量 $\boldsymbol{x}_0$ 及重开始数 m. 本算法计算 $\boldsymbol{x}_k$, 使得

$\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{x}_1 = \boldsymbol{x}_0$; $\boldsymbol{r}_1 = \boldsymbol{r}_0$; $\boldsymbol{p}_1 = \boldsymbol{r}_0$; $k = 1$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\| > \varepsilon)$

for $i = 1 : m$

$$\alpha_i = \frac{(\boldsymbol{r}_i, \boldsymbol{A}\boldsymbol{p}_i)}{(\boldsymbol{A}\boldsymbol{p}_i, \boldsymbol{A}\boldsymbol{p}_i)};$$

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i; \quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i \boldsymbol{A}\boldsymbol{p}_i;$$

for $s = 1 : i$

$$\beta_s^{(i)} = -\frac{(\boldsymbol{A}\boldsymbol{r}_{i+1}, \boldsymbol{A}\boldsymbol{p}_s)}{(\boldsymbol{A}\boldsymbol{p}_s, \boldsymbol{A}\boldsymbol{p}_s)};$$

end

$$\boldsymbol{p}_{i+1} = \boldsymbol{r}_{i+1} + \sum_{s=0}^{i} \beta_s^{(i)} \boldsymbol{p}_s;$$

$$\boldsymbol{A}\boldsymbol{p}_{i+1} = \boldsymbol{A}\boldsymbol{r}_{i+1} + \sum_{s=0}^{i} \beta_i^{(i)} \boldsymbol{A}\boldsymbol{p}_s;$$

Back

Close

end

$\boldsymbol{x}_1 = \boldsymbol{x}_{m+1};\ \boldsymbol{r}_1 = \boldsymbol{r}_{m+1};\ \boldsymbol{p}_1 = \boldsymbol{p}_{m+1};\ \boldsymbol{Ap}_1 = \boldsymbol{Ap}_{m+1};$

$k = k + 1;$

end

GCR(m) 方法的 MATLAB 程序如下:

```
%GCR(m)方法程序-gcrm.m
function [x,out,int,time,res,resvec]
                    =gcrm(A,b,x,restrt,max_it,tol)
tic; n=length(b); m=restrt;
r=b-A*x; p=r; mr=norm(r); k=0;
AP=zeros(n,m+1); P=zeros(n,m+1);
AP(:,1)=A*p; P(:,1)=p; resvec(1)=1;
while (k<max_it)
```

```
k=k+1;
for i=1:m
    alpha=(r'*AP(:,i))/(AP(:,i)'*AP(:,i));
    x=x+alpha*P(:,i);
    r=r-alpha*AP(:,i);
    res=norm(r)/mr;
    resvec((k-1)*m+1+i)=res;
    if (res<tol), break;  end
    Ar=A*r; s1=zeros(n,1); s2=zeros(n,1);
    for (s=1:i)
        b(s)=-(Ar'*AP(:,s))/(AP(:,s)'*AP(:,s));
        s1=s1+b(s)*P(:,s);
        s2=s2+b(s)*AP(:,s);
```

```
            end
            P(:,i+1)=r+s1;
            AP(:,i+1)=Ar+s2;
        end
        if (res<tol),
             out=k; int=i;  break;
        end
        AP(:,1)=AP(:,m+1); P(:,1)=P(:,m+1);
    end
    time=toc;
```

例 4.14 仍考虑例 4.13 中的 $\boldsymbol{A}$ 和 $\boldsymbol{b}$. 将重开始 GCR 方法应用到该线性方程组上, 取重开始数 $m = 6$, 在外迭代第 4 步内迭代第 2 步后收敛 $(\varepsilon = 10^{-10})$, 计算得到的近似解 $\hat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间

的绝对值误差为

$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 2.0725 \times 10^{-9},$$

但计算解 $\hat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{x}}\|_2 = 2.7746 \times 10^{-8}.$$

迭代过程的收敛轨迹如图 4.15 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量.

若按公式

总迭代次数 = (外迭代次数 − 1) × 重开始数 + 内迭代数

计算, 本例的总迭代次数为 20, 这跟例 4.13 中 GCR 方法一致.

此外, 还比较了 GCR(m) 与 GMRES(m) 算法. 从图 4.15 可以看出, 对于此例, GCR(m) 算法跟 GMRES(m) 算法的数值效果相仿.

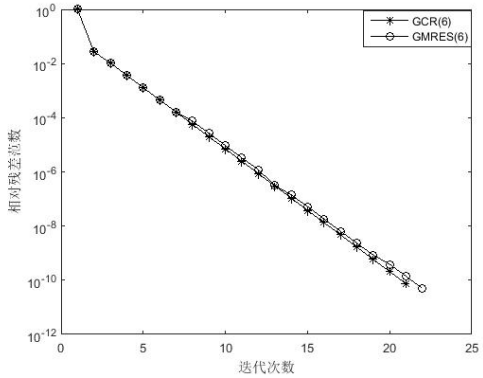


图 4.15 GCR(m) 方法的收敛特性

关于 GCR(m) 方法的收敛性, 有下面的结论.

定理 4.15 在定理 4.14 的假设下, 若 $\{\boldsymbol{r}_k\}$ 是 GCR(m) 的残量序列, 则

$$\|\boldsymbol{r}_{i(m+1)}\|_2 \leqslant \left[\min_{q_{m+1}\in\mathcal{P}_{m+1}} \|q_{m+1}(\boldsymbol{A})\|_2\right]^i \|\boldsymbol{r}_0\|_2, \tag{4.231}$$

由此, 得

$$\|\boldsymbol{r}_k\|_2 \leqslant \left[1 - \frac{\lambda_{\min}^2(\boldsymbol{H})}{\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})}\right]^{k/2} \|\boldsymbol{r}_0\|_2. \tag{4.232}$$

因此, GCR(m) 方法收敛.

证明 由定理 4.14, 有

$$\begin{aligned}\|\boldsymbol{r}_{i(m+1)}\|_2 &\leqslant \left[\min_{q_{m+1}\in\mathcal{P}_{m+1}} \|q_{m+1}(\boldsymbol{A})\|_2\right] \|\boldsymbol{r}_{(i-1)(m+1)}\|_2 \\ &\leqslant \cdots \leqslant \left[\min_{q_{m+1}\in\mathcal{P}_{m+1}} \|q_{m+1}(\boldsymbol{A})\|_2\right]^i \|\boldsymbol{r}_0\|_2.\end{aligned}$$

令 $k = i(m+1)+s$, $0 \leqslant s \leqslant m$, 再由定理 4.14, 有

$$\begin{aligned}\|\boldsymbol{r}_k\|_2 &\equiv \|\boldsymbol{r}_{i(m+1)+s}\|_2 \\ &\leqslant \left[1 - \frac{\lambda_{\min}^2(\boldsymbol{H})}{\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})}\right]^{s/2} \|\boldsymbol{r}_{i(m+1)}\|_2.\end{aligned} \tag{4.233}$$

由式 (4.231) 和定理 4.14, 得

$$\|\boldsymbol{r}_{i(m+1)}\|_2 \leqslant \left[1 - \frac{\lambda_{\min}(\boldsymbol{H})^2}{\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})}\right]^{i(m+1)/2} \|\boldsymbol{r}_0\|_2.$$

将上式代入式 (4.233) 即得式 (4.232). 证毕. □

§4.8 投影类方法

前几节所介绍的方法都是残量极小化类型的方法, 本节介绍第二类方法, 即残量正交化方法. 考虑如下的线性方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4.234}$$

式中: 矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和向量 $\boldsymbol{b} \in \mathbb{R}^n$ 是已经给定的, 而 $\boldsymbol{x} \in \mathbb{R}^n$ 是待求的未知向量. 残量正交化方法是求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$, 使得

$$\boldsymbol{r}_k \perp \mathcal{X}_k, \tag{4.235}$$

其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$, 而 $\mathcal{X}_k \subset \mathbb{R}^n$ 是一个适当选择的 k 维子空间. 通常称式 (4.235) 为 Galerkin 条件.

设

$$\mathcal{X}_k = \mathrm{span}\{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k\},$$
$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\},$$

并且记

$$\boldsymbol{V}_k = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k], \quad \boldsymbol{W}_k = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k],$$

则容易导出, 求 $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 满足式 (4.235) 就等价于求解线性方程组

$$\boldsymbol{H}_k \boldsymbol{z} = \boldsymbol{f}_k, \tag{4.236}$$

式中:

$$\boldsymbol{H}_k = \boldsymbol{W}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{V}_k, \quad \boldsymbol{f}_k = \boldsymbol{W}_k^{\mathrm{T}} \boldsymbol{r}_0. \tag{4.237}$$

从几何上来看, 当 $\boldsymbol{W}_k^{\mathrm{T}} \boldsymbol{V}_k = \boldsymbol{I}_k$ 时, $\boldsymbol{H}_k$ 正好是 $\boldsymbol{A}$ 沿 $\mathcal{X}_k^{\perp}$ 到 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 上的投影 (见注 4.6), 故称式 (4.236) 为*投影方程*, 而称这类方法为*投影类方法*.

通常 $k \ll n$, 故式 (4.236) 有多种方法来求解. 这样, 实现这一方法的关键是如何选择子空间 $\mathcal{X}_k$ 以及如何有效地计算 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$

和 $\mathcal{X}_k$ 的基向量.

本节取 $\mathcal{X}_k = \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0)$, $\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$, 并给出这方面的三类典型的方法: 双共轭梯度法 (BCG 方法)、共轭梯度平方法 (CGS 方法) 和稳定化共轭梯度法 (BCGSTAB 方法). BCG 方法是基础, 另外两个方法是通过对 BCG 方法的改进而得到的.

§4.8.1 BCG 方法

BCG 方法 (双共轭梯度法, Bi-Conjugate Gradient) 是在式 (4.235) 中取子空间 $\mathcal{X}_k = \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0)$ 而得到的一类投影方法. 当然, 可以利用非对称 Lanczos 方法来计算 $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0)$ 和 $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0)$ 的基, 然后以类似于 CG 方法的推导过程导出 BCG 的基本迭代格式. 为了更清楚地展示这类方法与共轭梯度法的联系, 这里采用 CG 方法的一种等价描述来导出 BCG 的迭代格式.

1. CG 方法的等价描述

为了便于参照, 先将求对称正定线性方程组的 CG 方法的迭代格式重新叙述如下:

(1) $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0;\ \boldsymbol{p}_{-1} = \boldsymbol{0};\ \rho_{-1} = 1.$

(2) $\rho_k = \boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k;\ \beta_k = \rho_k/\rho_{k-1};\ \boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{p}_{k-1};$

$\sigma_k = \boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k;\ \alpha_k = \rho_k/\sigma_k;\ \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k;$

$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{p}_k;\ k = 0, 1, 2, \cdots.$

若迭代进行到了 k 步, 则将产生三组向量:

① 近似解向量组 $\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k$;

② 残量组 $\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k$;

③ 方向向量组 $\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_k$.

它们具有如下的基本性质:

① $\boldsymbol{r}_i^{\mathrm{T}}\boldsymbol{r}_j = \rho_i\delta_{ij}$ (残量相互正交);

② $\boldsymbol{p}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_j = \sigma_i\delta_{ij}$ (方向向量相互 $\boldsymbol{A}$–共轭正交);

③ $\mathcal{K}_{k+1}(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k\} = \mathrm{span}\{\boldsymbol{p}_0, \boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}$.

由性质 ③ 可知, 必存在 $\varphi_k, \psi_k \in \mathcal{P}_k$, 使得

$$\boldsymbol{r}_k = \varphi_k(\boldsymbol{A})\boldsymbol{r}_0, \quad \boldsymbol{p}_k = \psi_k(\boldsymbol{A})\boldsymbol{r}_0.$$

若记 $\varphi_0(t) = 1$, $\psi_{-1}(t) = 0$, $\vartheta(t) = t$, 则由前面所述的 CG 方法的迭代格式可导出

$$\begin{cases} \varphi_{k+1} = \varphi_k - \alpha_k\vartheta\psi_k, \\ \psi_k = \varphi_k + \beta_k\psi_{k-1}, \quad k = 0, 1, 2, \cdots. \end{cases} \tag{4.238}$$

现在在线性空间 $\mathcal{P}_n$ 上定义双线性泛函

$$[\varphi, \psi] = \boldsymbol{r}_0^{\mathrm{T}}\varphi(\boldsymbol{A})\psi(\boldsymbol{A})\boldsymbol{r}_0, \quad \forall\ \varphi, \psi \in \mathcal{P}_n. \tag{4.239}$$

注意, 这样定义的二元泛函除不能由 $[\varphi,\varphi]=0$ 导出 $\varphi=0$ 之外, 它具有内积的其他所有性质.

利用式 (4.239) 容易导出 CG 方法迭代格式中的几个参数的表达式, 即

$$\rho_k=[\varphi_k,\varphi_k],\quad \sigma_k=[\psi_k,\vartheta\psi_k]. \tag{4.240}$$

将式 (4.238) 和式 (4.240) 相结合就得到了 CG 方法用多项式语言给出的等价描述:

$$(1)\ \ \varphi_0=1;\ \psi_{-1}=0;\ \rho_{-1}=1; \tag{4.241a}$$

$$(2)\ \ \rho_k=[\varphi_k,\varphi_k];\ \beta_k=\rho_k/\rho_{k-1}; \tag{4.241b}$$

$$\psi_k=\varphi_k+\beta_k\psi_{k-1}; \tag{4.241c}$$

$$\sigma_k=[\psi_k,\vartheta\psi_k];\ \alpha_k=\rho_k/\sigma_k; \tag{4.241d}$$

$$\varphi_{k+1}=\varphi_k-\alpha_k\vartheta\psi_k;\ k=0,1,2,\cdots. \tag{4.241e}$$

显然, 由上面的迭代格式产生的多项式系列满足

$$[\varphi_i, \varphi_j] = \rho_i \delta_{ij}, \quad [\psi_i, \vartheta\psi_j] = \sigma_i \delta_{ij}. \tag{4.242}$$

2. BCG 方法

类比于式 (4.239), 对给定的非奇异矩阵 $\boldsymbol{A}$ 以及向量 $\boldsymbol{r}_0$ 和 $\widetilde{\boldsymbol{r}}_0$, 在 $\mathcal{P}_n$ 上定义双线性泛函

$$\langle \varphi, \psi \rangle = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \varphi(\boldsymbol{A}) \psi(\boldsymbol{A}) \boldsymbol{r}_0, \quad \forall\, \varphi,\, \psi \in \mathcal{P}_n, \tag{4.243}$$

然后将式 (4.239) 中定义的双线性泛函 $[\cdot, \cdot]$ 换为现在定义的 $\langle \cdot, \cdot \rangle$, 再逐字照搬过来, 便有

$$(1) \quad \varphi_0 = 1;\ \psi_{-1} = 0;\ \rho_{-1} = 1; \tag{4.244a}$$

$$(2) \quad \rho_k = \langle \varphi_k, \varphi_k \rangle;\ \beta_k = \rho_k / \rho_{k-1}; \tag{4.244b}$$

$$\psi_k = \varphi_k + \beta_k \psi_{k-1}; \tag{4.244c}$$

$$\sigma_k = \langle \psi_k; \vartheta\psi_k \rangle;\ \alpha_k = \rho_k / \sigma_k; \tag{4.244d}$$

$$\varphi_{k+1} = \varphi_k - \alpha_k \vartheta\psi_k;\ k = 0, 1, 2, \cdots. \tag{4.244e}$$

当然, 也可用数学归纳法证明这样迭代产生的 $\{\varphi_i\}_{i=0}^k$ 和 $\{\psi_i\}_{i=0}^k$ 具有性质

$$\langle \varphi_i, \varphi_j \rangle = \rho_i \delta_{ij}, \quad \langle \psi_i, \vartheta \psi_j \rangle = \sigma_i \delta_{ij}. \tag{4.245}$$

现在定义

$$\boldsymbol{r}_k = \varphi_k(\boldsymbol{A})\boldsymbol{r}_0, \quad \widetilde{\boldsymbol{r}}_k = \varphi_k(\boldsymbol{A}^{\mathrm{T}})\widetilde{\boldsymbol{r}}_0, \tag{4.246}$$

$$\boldsymbol{q}_k = \psi_k(\boldsymbol{A})\boldsymbol{r}_0, \quad \widetilde{\boldsymbol{q}}_k = \psi_k(\boldsymbol{A}^{\mathrm{T}})\widetilde{\boldsymbol{r}}_0. \tag{4.247}$$

将式 (4.244) 改成向量形式, 便有

(1) $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\widetilde{\boldsymbol{r}}_0$ 满足 $\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$;

$\boldsymbol{q}_{-1} = \widetilde{\boldsymbol{q}}_{-1} = \boldsymbol{0}$; $\rho_{-1} = 1$;

(2) $\rho_k = \widetilde{\boldsymbol{r}}_k^{\mathrm{T}}\boldsymbol{r}_k;\ \beta_k = \rho_k/\rho_{k-1};$ (4.248a)

$$\boldsymbol{q}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{q}_{k-1}; \tag{4.248b}$$

$$\widetilde{\boldsymbol{q}}_k = \widetilde{\boldsymbol{r}}_k + \beta_k\widetilde{\boldsymbol{q}}_{k-1}; \tag{4.248c}$$

$$\sigma_k = \widetilde{\boldsymbol{q}}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_k;\ \alpha_k = \rho_k/\sigma_k; \tag{4.248d}$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{q}_k; \tag{4.248e}$$

$$\widetilde{\boldsymbol{r}}_{k+1} = \widetilde{\boldsymbol{r}}_k - \alpha_k\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_k;\ k = 0, 1, 2, \cdots. \tag{4.248f}$$

利用数学归纳法可证如下的结果.

定理 4.16 由式 (4.248) 产生的序列 $\{\boldsymbol{q}_k\}$, $\{\widetilde{\boldsymbol{q}}_k\}$, $\{\boldsymbol{r}_k\}$ 和 $\{\widetilde{\boldsymbol{r}}_k\}$ 具有如下性质:

(1) $\widetilde{\boldsymbol{r}}_k^{\mathrm{T}}\boldsymbol{r}_l = \boldsymbol{r}_k^{\mathrm{T}}\widetilde{\boldsymbol{r}}_l = \rho_k\delta_{kl}.$ (4.249a)

(2) $\widetilde{\boldsymbol{q}}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_l = \boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{A}\widetilde{\boldsymbol{q}}_l = \sigma_k\delta_{kl}.$ (4.249b)

(3) $\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{q}_0, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_{k-1}\} = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}\}.$ (4.249c)

(4) $\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0) = \mathrm{span}\{\widetilde{\boldsymbol{q}}_0, \widetilde{\boldsymbol{q}}_1, \cdots, \widetilde{\boldsymbol{q}}_{k-1}\} = \mathrm{span}\{\widetilde{\boldsymbol{r}}_0, \widetilde{\boldsymbol{r}}_1, \cdots, \widetilde{\boldsymbol{r}}_{k-1}\}.$ (4.249d)

证明 首先对 k 用归纳法证明式 (4.249a) 和式 (4.249b). 显然, 当 $k=0$ 时, 结论是平凡的. 故设 $k+1 \geqslant 1$. 假设结论对 k 成立. 由式 (4.248c) 和式 (4.248e), 得

$$\begin{aligned} \boldsymbol{r}_{k+1}^{\mathrm{T}} \widetilde{\boldsymbol{r}}_l &= \boldsymbol{r}_k^{\mathrm{T}} \widetilde{\boldsymbol{r}}_l - \alpha_k \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \widetilde{\boldsymbol{r}}_l \\ &= \boldsymbol{r}_k^{\mathrm{T}} \widetilde{\boldsymbol{r}}_l - \alpha_i \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} (\widetilde{\boldsymbol{q}}_l - \beta_l \widetilde{\boldsymbol{q}}_{l-1}). \end{aligned} \tag{4.250}$$

在式 (4.250) 中, 当 $l=k$ 时, 由 α_k 的表达式和归纳假设, 得

$$\begin{aligned} \boldsymbol{r}_{k+1}^{\mathrm{T}} \widetilde{\boldsymbol{r}}_k &= \boldsymbol{r}_k^{\mathrm{T}} \widetilde{\boldsymbol{r}}_k - \frac{\widetilde{\boldsymbol{r}}_k^{\mathrm{T}} \boldsymbol{r}_k}{\widetilde{\boldsymbol{q}}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{q}_k} (\boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \widetilde{\boldsymbol{q}}_k - \beta_k \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \widetilde{\boldsymbol{q}}_{k-1}) \\ &= \boldsymbol{r}_k^{\mathrm{T}} \widetilde{\boldsymbol{r}}_k - \frac{\boldsymbol{r}_k^{\mathrm{T}} \widetilde{\boldsymbol{r}}_k}{\widetilde{\boldsymbol{q}}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{q}_k} (\widetilde{\boldsymbol{q}}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{q}_k - \beta_k \widetilde{\boldsymbol{q}}_{k-1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{q}_k) = 0. \end{aligned}$$

当 $l<k$ 时, 由归纳法假设可得 $\boldsymbol{r}_{k+1}^{\mathrm{T}} \widetilde{\boldsymbol{r}}_l = 0$. 同理, 可证 $\widetilde{\boldsymbol{r}}_{k+1}^{\mathrm{T}} \boldsymbol{r}_l = 0\,(l \leqslant k)$. 这就证明了式 (4.249a).

Back

Close

由式 (4.248b) 和式 (4.248f), 得

$$\begin{aligned}\boldsymbol{q}_{k+1}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_l &= \boldsymbol{r}_{k+1}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_l + \beta_{k+1}\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_l \\ &= \boldsymbol{r}_{k+1}^{\mathrm{T}}\left(\frac{\widetilde{\boldsymbol{r}}_l - \widetilde{\boldsymbol{r}}_{l+1}}{\alpha_k}\right) + \beta_{k+1}\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_l. \qquad (4.251)\end{aligned}$$

在式 (4.251) 中, 当 $l = k$ 时, 由 α_k 和 β_{k+1} 的表达式及归纳假设, 得

$$\begin{aligned}\boldsymbol{q}_{k+1}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_k &= \boldsymbol{r}_{k+1}^{\mathrm{T}}\left(\frac{\widetilde{\boldsymbol{r}}_k - \widetilde{\boldsymbol{r}}_{k+1}}{\alpha_k}\right) + \beta_{k+1}\frac{\boldsymbol{r}_k^{\mathrm{T}}\widetilde{\boldsymbol{r}}_k}{\alpha_k} \\ &= \boldsymbol{r}_{k+1}^{\mathrm{T}}\left(\frac{\widetilde{\boldsymbol{r}}_k - \widetilde{\boldsymbol{r}}_{k+1}}{\alpha_k}\right) + \frac{\boldsymbol{r}_{k+1}^{\mathrm{T}}\widetilde{\boldsymbol{r}}_{k+1}}{\boldsymbol{r}_k^{\mathrm{T}}\widetilde{\boldsymbol{r}}_k}\frac{\boldsymbol{r}_k^{\mathrm{T}}\widetilde{\boldsymbol{r}}_k}{\alpha_k} \\ &= \frac{\boldsymbol{r}_{k+1}^{\mathrm{T}}\widetilde{\boldsymbol{r}}_k}{\alpha_k} = 0.\end{aligned}$$

当 $l < k$ 时, 利用式 (4.249a) 和归纳法假设可得 $\boldsymbol{q}_{k+1}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_l = 0$. 同理, 可证 $\widetilde{\boldsymbol{q}}_{k+1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_l = 0\,(l \leqslant k)$. 这就证明了式 (4.249b).

由于

$$\mathcal{K}_k(\boldsymbol{A}, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}\},$$

$$\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0) = \mathrm{span}\{\widetilde{\boldsymbol{r}}_0, \widetilde{\boldsymbol{r}}_1, \cdots, \widetilde{\boldsymbol{r}}_{k-1}\},$$

且向量组 $\{\boldsymbol{q}_0, \boldsymbol{q}_1, \cdots, \boldsymbol{q}_{k-1}\}$ 与 $\{\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}\}$ 及向量组 $\{\widetilde{\boldsymbol{q}}_0, \widetilde{\boldsymbol{q}}_1, \cdots, \widetilde{\boldsymbol{q}}_{k-1}\}$ 与 $\{\widetilde{\boldsymbol{r}}_0, \widetilde{\boldsymbol{r}}_1, \cdots, \widetilde{\boldsymbol{r}}_{k-1}\}$ 可以相互线性表出, 故容易得到式 (4.249c) 和式 (4.249d). 证毕. □

此外, 若定义

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{q}_k, \quad k = 0, 1, 2, \cdots,$$

则迭代式 (4.248e) 中的向量 $\boldsymbol{r}_k$ 刚好为 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$, 且由定理 4.16 的 (1) 和 (4) 可知

$$\boldsymbol{r}_k \perp \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0),$$

这表明 $\boldsymbol{x}_k$ 满足残量正交化条件式 (4.235). 综上所述, 就得到了如下算法.

算法 4.19 (BCG 方法) 给定线性方程组 (4.234), 初始向量 $\boldsymbol{x}_0$ 和 $\varepsilon > 0$. 本算法计算向量 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{q}_{-1} = \widetilde{\boldsymbol{q}}_{-1} = \boldsymbol{0}$; $\rho_{-1} = 1$;

选择 $\widetilde{\boldsymbol{r}}_0$ 满足 $\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 > \varepsilon)$

$$\rho_k = \widetilde{\boldsymbol{r}}_k^{\mathrm{T}}\boldsymbol{r}_k;\ \beta_k = \rho_k/\rho_{k-1};$$

$$\boldsymbol{q}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{q}_{k-1};\ \widetilde{\boldsymbol{q}}_k = \widetilde{\boldsymbol{r}}_k + \beta_k\widetilde{\boldsymbol{q}}_{k-1};$$

$$\sigma_k = \widetilde{\boldsymbol{q}}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_k;\ \alpha_k = \rho_k/\sigma_k;$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{q}_k;\ \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{q}_k;$$

$$\widetilde{\boldsymbol{r}}_{k+1} = \widetilde{\boldsymbol{r}}_k - \alpha_k\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{q}}_k;\ k = k+1;$$

end

注 4.9 若计算过程中出现 $\rho_{k-1}=0$ 或 $\sigma_k=0$, 但还没有满足收敛性条件, 则此时算法就发生中断. 此外, 如果希望同时求解对偶方程 $\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{x}}=\widetilde{\boldsymbol{b}}$, 则只需在上述算法中增加两句 $\widetilde{\boldsymbol{r}}_0=\widetilde{\boldsymbol{b}}-\boldsymbol{A}^{\mathrm{T}}\widetilde{\boldsymbol{x}}_0$ 和 $\widetilde{\boldsymbol{x}}_{k+1}=\widetilde{\boldsymbol{x}}_k+\alpha_k\widetilde{\boldsymbol{q}}_k$ 即可.

BCG 方法的 MATLAB 程序如下:

```
%BCG方法程序-bcg.m
function [x,k,time,res,resvec]=bcg(A,b,x,max_it,tol)
tic;n=length(b);
q=zeros(n,1);qt=q;rho=1;
r=b-A*x;mr=norm(r);k=0;
rt=ones(n,1);%rt的选取会影响收敛速度
%rt=r/mr; %rt=zeros(n,1);rt(1)=1;
while (k<max_it)
```

```
    k=k+1;
    res=norm(r)/mr;resvec(k)=res;
    if (res<tol), break; end
    rho1=rt'*r;beta=rho1/rho;
    q=r+beta*q;qt=rt+beta*qt;
    Aq=A*q; sigma=qt'*Aq;
    alpha=rho1/sigma;
    x=x+alpha*q;
    r=r-alpha*Aq;
    rt=rt-alpha*(A'*qt);
    rho=rho1;
end
time=toc;
```

例 4.15 假设线性方程组的系数矩阵 $\boldsymbol{A}$ 由 MATLAB 命令

$$gallery('lotkin', n), \quad n = 1000,$$

产生, 这是一个 Hilbert 矩阵 (即其 (i, j) 元素为 $1/(i + j - 1)$), 但其第 1 行的元素都换成了 1. 这个矩阵非常病态, 其条件数为 6.65×10^{21}. 再假设真解 $\boldsymbol{x}^*$ 是分量都为 1 的向量, 从而该方程组的右端项为 $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}^*$. 将 BCG 算法应用到该线性方程组上, 取算法中的 $\widetilde{\boldsymbol{r}}_0 = ones(n, 1)$. 迭代在 60 步后收敛 ($\varepsilon = 10^{-10}$), 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的绝对误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 0.6727,$$

但计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 2.6526 \times 10^{-8}.$$

迭代过程的收敛轨迹如图 4.16 所示, 其中横坐标为迭代步数 k, 纵

Back

Close

坐标为 $\lg \|\boldsymbol{r}_k\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残量. 请注意观察, 与前面几个方法相比, 此时 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$ 不再单调下降, 而是在振荡中下降着.

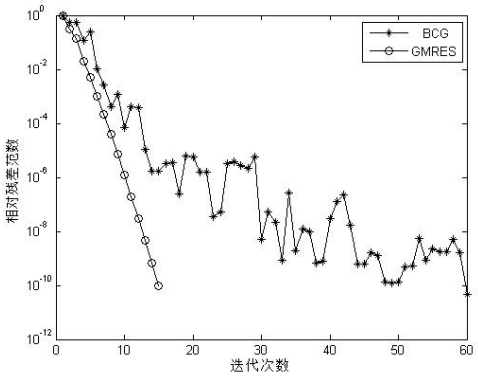


图 4.16 BCG 算法的收敛特性

§4.8.2 CGS 方法

CGS 方法 (共轭梯度平方法, Conjugate Gradient Squared) 是为了避免 BCG 方法中 $\boldsymbol{A}^{\mathrm{T}}$ 的出现而提出的.

仔细观察 BCG 的迭代格式就会发现, $\boldsymbol{A}^{\mathrm{T}}$ 只在计算 $\widetilde{\boldsymbol{r}}_k$ 时用到, 而这一量又在计算 ρ_k 和 σ_k 时用到. 再利用导出 BCG 迭代格式的多项式形式的迭代格式 (4.244) 立即可以看出, 这两个量的计算完全可以避开 $\boldsymbol{A}^{\mathrm{T}}$ 的使用. 事实上, 有

$$\begin{cases} \rho_k = \langle \varphi_k, \varphi_k \rangle = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \varphi_k(\boldsymbol{A}) \varphi_k(\boldsymbol{A}) \boldsymbol{r}_0 = \langle \varphi_0, \varphi_k^2 \rangle, \\ \sigma = \langle \psi_k, \vartheta \psi_k \rangle = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \psi_k(\boldsymbol{A}) \boldsymbol{A} \psi_k(\boldsymbol{A}) \boldsymbol{r}_0 = \langle \psi_0, \vartheta \psi_k^2 \rangle. \end{cases} \tag{4.252}$$

因此, 若定义

$$\boldsymbol{r}_k = \varphi_k^2(\boldsymbol{A}) \boldsymbol{r}_0, \quad \boldsymbol{q}_k = \psi_k^2(\boldsymbol{A}) \boldsymbol{r}_0, \tag{4.253}$$

则有

$$\rho_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_k, \quad \sigma_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{A}\boldsymbol{q}_k. \tag{4.254}$$

这样一来, 要用式 (4.253) 来高效地计算 $\boldsymbol{r}_k$ 和 $\boldsymbol{q}_k$, 关键是如何高效地计算 φ_k^2 和 ψ_k^2. 由式 (4.244) 立即可导出

$$\psi_k^2 = (\varphi_k + \beta_k \psi_{k-1})^2 = \varphi_k^2 + 2\beta_k \varphi_k \psi_{k-1} + \beta_k^2 \psi_{k-1}^2, \tag{4.255}$$

$$\varphi_{k+1}^2 = (\varphi_k - \alpha_k \vartheta \psi_k)^2 = \varphi_k^2 - 2\alpha_k \vartheta \varphi_k \psi_k + \alpha_k^2 \vartheta^2 \psi_k^2, \tag{4.256}$$

其中又涉及 $\varphi_k \psi_{k-1}$ 和 $\varphi_k \psi_k$. 再利用式 (4.244), 有

$$\varphi_{k+1} \psi_k = \varphi_k \psi_k - \alpha_k \vartheta \psi_k^2, \tag{4.257}$$

$$\varphi_k \psi_k = \varphi_k^2 + \beta_k \varphi_k \psi_{k-1}. \tag{4.258}$$

再定义

$$\boldsymbol{p}_k = \varphi_k(\boldsymbol{A}) \psi_{k-1}(\boldsymbol{A}) \boldsymbol{r}_0, \quad \boldsymbol{u}_k = \varphi_k(\boldsymbol{A}) \psi_k(\boldsymbol{A}) \boldsymbol{r}_0, \tag{4.259}$$

由式 (4.255)~式 (4.258) 得到如下递推公式:

$$\begin{cases} \boldsymbol{q}_k = \boldsymbol{r}_k + 2\beta_k \boldsymbol{p}_k + \beta_k^2 \boldsymbol{q}_{k-1}, \\ \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - 2\alpha_k \boldsymbol{A}\boldsymbol{u}_k + \alpha_k^2 \boldsymbol{A}^2 \boldsymbol{q}_k, \\ \boldsymbol{p}_{k+1} = \boldsymbol{u}_k - \alpha_k \boldsymbol{A}\boldsymbol{q}_k, \\ \boldsymbol{u}_k = \boldsymbol{r}_k + \beta_k \boldsymbol{p}_k. \end{cases} \tag{4.260}$$

由式 (4.260) 的第 1 式和第 4 式, 得

$$\boldsymbol{q}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{p}_k + \beta_k \boldsymbol{q}_{k-1}), \tag{4.261}$$

再由式 (4.260) 的第 2 式和第 3 式, 得

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}(\boldsymbol{u}_k + \boldsymbol{p}_{k+1}). \tag{4.262}$$

由式 (4.262) 可知, 若定义

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k(\boldsymbol{u}_k + \boldsymbol{p}_{k+1}), \tag{4.263}$$

则 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$ 正好是对应近似解的残量.

综合上面的讨论, 便有如下的共轭梯度平方法.

算法 4.20 (CGS 方法) 给定线性方程组 (4.234), 初始向量 $\boldsymbol{x}_0$ 和容许误差 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{q}_{-1} = \boldsymbol{p}_0 = 0$; $\rho_{-1} = 1$;

选择 $\widetilde{\boldsymbol{r}}_0$ 满足 $\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|r_0\|_2 > \varepsilon)$

$\rho_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_k$; $\beta_k = \rho_k/\rho_{k-1}$;

$\boldsymbol{u}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{p}_k$;

$\boldsymbol{q}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{p}_k + \beta_k\boldsymbol{q}_{k-1})$;

$\boldsymbol{q}_k = \boldsymbol{A}\boldsymbol{q}_k$; $\sigma_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{q}_k$;

$\alpha_k = \rho_k/\sigma_k$; $\boldsymbol{p}_{k+1} = \boldsymbol{u}_k - \alpha_k\boldsymbol{q}_k$;

$\boldsymbol{z}_k = \alpha_k(\boldsymbol{u}_k + \boldsymbol{p}_{k+1})$;

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{z}_k;\ \boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \boldsymbol{A}\boldsymbol{z}_k;$$

$$k = k + 1;$$

end

算法 4.20 确实避免了 $\boldsymbol{A}^{\mathrm{T}}$ 的出现, 每次迭代只需作两次系数矩阵 $\boldsymbol{A}$ 与向量的乘积. 由前面的推导过程可知, 算法 4.20 产生的近似解的残量为 $\boldsymbol{r}_k^{\mathrm{CGS}} = \varphi_k^2(\boldsymbol{A})\boldsymbol{r}_0$, 而 BCG 算法产生的近似解的残量为 $\boldsymbol{r}_k^{\mathrm{BCG}} = \varphi_k(\boldsymbol{A})\boldsymbol{r}_0$. 当 $\boldsymbol{r}_k^{\mathrm{BCG}}$ 趋向于零时, $\varphi_k(\boldsymbol{A})$ 就是一个收缩因子. 从这个意义上来讲, $\boldsymbol{r}_k^{\mathrm{CGS}}$ 的收敛到零的速度应该是 $\boldsymbol{r}_k^{\mathrm{BCG}}$ 的两倍. 但这个算法的缺点是 $\|\boldsymbol{r}_k^{\mathrm{CGS}}\|_2$ 随着 k 的增加会发生激烈的抖动.

例 4.16 仍然考虑例 4.15 中的 $\boldsymbol{A}$ 和 $\boldsymbol{b}$. 将 CGS 算法应用到该线性方程组上, 取算法中的 $\widetilde{\boldsymbol{r}}_0 = (1, 0, \cdots, 0)^{\mathrm{T}}$, 迭代在 142 步后

收敛 ($\varepsilon = 10^{-10}$), 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 0.8086,$$

但计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 7.0413 \times 10^{-8}.$$

迭代过程的收敛轨迹如图 4.17 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量. 此时 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$ 与 BCG 方法一样也是在剧烈的振荡中艰难地下降着, 但注意到此例反映不出前面分析中 CGS 的收敛速度大约是 BCG 的两倍, 或许是受 $\widetilde{\boldsymbol{r}}_0$ 选取的影响.

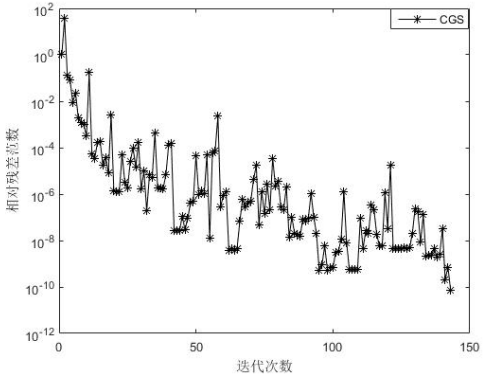


图 4.17 CGS 算法的收敛特性

§4.8.3 BCGSTAB 方法

BCGSTAB 方法 (稳定化双共轭梯度法, Bi-Conjugate Gradient Stablized) 是为了改进 CGS 方法之残量的范数剧烈抖动而提出的. 这一方法的基本思想是 CGS 方法的残量 $\boldsymbol{r}_k^{\mathrm{CGS}}$ 满足

$$\boldsymbol{r}_k^{\mathrm{CGS}} = \varphi_k(\boldsymbol{A})\boldsymbol{r}_k^{\mathrm{BCG}} = \varphi_k^2(\boldsymbol{A})\boldsymbol{r}_0, \tag{4.264}$$

可以考虑不用多项式 φ_k, 而是选择一个其他的 k 次多项式 $\widetilde{\varphi}_k$, 使

$$\boldsymbol{r}_k = \widetilde{\varphi}_k(\boldsymbol{A})\boldsymbol{r}_k^{\mathrm{BCG}} = \widetilde{\varphi}_k(\boldsymbol{A})\varphi_k(\boldsymbol{A})\boldsymbol{r}_0, \tag{4.265}$$

以期待这样选取的相对残差范数 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$ 的振荡性有所改进.

一种选择 $\widetilde{\varphi}_k$ 的方法是根据递推公式

$$\widetilde{\varphi}_0(t) = 1, \quad \widetilde{\varphi}_{k+1}(t) = (1 - \omega_{k+1}t)\widetilde{\varphi}_k(t), \tag{4.266}$$

式中: ω_{k+1} 为待定参数. BCGSTAB 方法正是利用这一参数的可选择性来改进其相对残差范数的振荡性的.

利用式 (4.244) 和式 (4.266), 可导出 $\widetilde{\varphi}_{k+1}\varphi_{k+1}$ 的递推计算公式

$$\begin{aligned}\widetilde{\varphi}_{k+1}\varphi_{k+1} &= (1 - \omega_{k+1}\vartheta)\widetilde{\varphi}_k(\varphi_k - \alpha_k\vartheta\psi_k)\\ &= (1 - \omega_{k+1}\vartheta)(\widetilde{\varphi}_k\varphi_k - \alpha_k\vartheta\widetilde{\varphi}_k\psi_k),\end{aligned} \tag{4.267}$$

$$\begin{aligned}\widetilde{\varphi}_k\psi_k &= \widetilde{\varphi}_k(\varphi_k + \beta_k\psi_{k-1})\\ &= \widetilde{\varphi}_k\varphi_k + \beta_k(1 - \omega_k\vartheta)\widetilde{\varphi}_{k-1}\psi_{k-1}.\end{aligned} \tag{4.268}$$

现定义

$$\boldsymbol{r}_k = \widetilde{\varphi}_k(\boldsymbol{A})\varphi_k(\boldsymbol{A})\boldsymbol{r}_0, \quad \boldsymbol{p}_k = \widetilde{\varphi}_k(\boldsymbol{A})\psi_k(\boldsymbol{A})\boldsymbol{r}_0, \tag{4.269}$$

则由式 (4.267) 和式 (4.268), 得

$$\boldsymbol{r}_{k+1} = (\boldsymbol{I} - \omega_{k+1}\boldsymbol{A})(\boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{p}_k), \tag{4.270}$$

$$\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k(\boldsymbol{I} - \omega_k\boldsymbol{A})\boldsymbol{p}_{k-1}. \tag{4.271}$$

下面来考虑 α_k 和 β_k 的计算问题. 由式 (4.244) 可知

$$\alpha_k = \frac{\rho_k}{\sigma_k}, \quad \beta_k = \frac{\rho_k}{\rho_{k-1}}, \tag{4.272}$$

式中:

$$\rho_k = \langle \varphi_k, \varphi_k \rangle, \quad \sigma_k = \langle \psi_k, \vartheta\psi_k \rangle. \tag{4.273}$$

由定理 4.16 可知

$$\varphi_k(\boldsymbol{A})\boldsymbol{r}_0 = \boldsymbol{r}_k^{\mathrm{BCG}} \perp \mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0),$$
$$\psi_k(\boldsymbol{A})\boldsymbol{r}_0 = \boldsymbol{q}_k^{\mathrm{BCG}} \perp \boldsymbol{A}^{\mathrm{T}}\mathcal{K}_k(\boldsymbol{A}^{\mathrm{T}}, \widetilde{\boldsymbol{r}}_0).$$

于是对任意的 $\psi \in \mathcal{P}_{k-1}$, 有

$$\langle \varphi_k, \psi \rangle = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \psi(\boldsymbol{A})\varphi_k\big(\boldsymbol{A}\big)\boldsymbol{r}_0 = \big(\psi(\boldsymbol{A}^{\mathrm{T}})\widetilde{\boldsymbol{r}}_0\big)^{\mathrm{T}}\big(\varphi_k(\boldsymbol{A})\boldsymbol{r}_0\big) = 0, \quad (4.274)$$

$$\langle \psi_k, \vartheta\psi \rangle = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \psi(\boldsymbol{A})\boldsymbol{A}\psi_k(\boldsymbol{A})\boldsymbol{r}_0 = \big(\boldsymbol{A}^{\mathrm{T}}\psi\big(\boldsymbol{A}^{\mathrm{T}}\big)\widetilde{\boldsymbol{r}}_0\big)^{\mathrm{T}}\big(\psi_k(\boldsymbol{A})\boldsymbol{r}_0\big) = 0. \quad (4.275)$$

设 φ_k 和 $\widetilde{\varphi}_k$ 的首项系数分别为 ξ_k 和 η_k, 则由式 (4.244) 和式 (4.266) 可知, 它们可递推地计算

$$\xi_{k+1} = -\alpha_k \xi_k, \quad \eta_{k+1} = -\omega_{k+1}\eta_k, \quad (4.276)$$

其中 $\xi_0 = \eta_0 = 1$, 这里用到了

$$\psi_k = \varphi_k + \beta_k \psi_{k-1}$$

蕴涵着 ψ_k 和 φ_k 有相同的首项系数. 这样

$$\psi = \psi_k - \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k \quad \text{和} \quad \varphi = \varphi_k - \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k$$

均为 $k-1$ 次多项式, 从而利用式 (4.274) 和式 (4.275) 有

$$\begin{aligned}
\rho_k &= \langle \varphi_k, \varphi_k \rangle = \langle \varphi_k, \varphi + \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k \rangle = \langle \varphi_k, \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k \rangle \\
&= \frac{\xi_k}{\eta_k}\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\widetilde{\varphi}_k(\boldsymbol{A})\varphi_k(\boldsymbol{A})\boldsymbol{r}_0 = \frac{\xi_k}{\eta_k}\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_k, \qquad (4.277) \\
\sigma_k &= \langle \psi_k, \vartheta\psi_k \rangle = \langle \psi + \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k, \vartheta\psi_k \rangle = \langle \frac{\xi_k}{\eta_k}\widetilde{\varphi}_k, \vartheta\psi_k \rangle \\
&= \frac{\xi_k}{\eta_k}\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{A}\widetilde{\varphi}_k(\boldsymbol{A})\psi_k(\boldsymbol{A})\boldsymbol{r}_0 = \frac{\xi_k}{\eta_k}\widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k. \qquad (4.278)
\end{aligned}$$

因此,

$$\alpha_k = \frac{\rho_k}{\sigma_k} = \frac{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_k}{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{A}\boldsymbol{p}_k}, \tag{4.279}$$

$$\beta_k = \frac{\rho_k}{\rho_{k-1}} = \frac{\xi_k}{\xi_{k-1}} \frac{\eta_{k-1}}{\eta_k} \frac{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_k}{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_{k-1}} = \frac{\alpha_{k-1}}{\omega_k} \frac{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_k}{\widetilde{\boldsymbol{r}}_0^{\mathrm{T}} \boldsymbol{r}_{k-1}}, \tag{4.280}$$

其中最后一个等式用到了式 (4.276).

到此为止 ω_{k+1} 仍然是自由的, 下面确定 ω_{k+1}. 为了符号简单起见, 记

$$\boldsymbol{s}_k = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k, \tag{4.281}$$

有

$$\boldsymbol{r}_{k+1} = (\boldsymbol{I} - \omega_{k+1}\boldsymbol{A})\boldsymbol{s}_k. \tag{4.282}$$

自然选择 ω_{k+1} 使

$$\|\boldsymbol{r}_{k+1}\|_2 = \min_{\omega} \|(\boldsymbol{I} - \omega \boldsymbol{A})\boldsymbol{s}_k\|_2.$$




Back
Close

解此最小二乘问题, 得

$$\omega_{k+1} = \frac{\boldsymbol{s}_k^{\mathrm{T}} \boldsymbol{A}\boldsymbol{s}_k}{\boldsymbol{s}_k^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{A}\boldsymbol{s}_k}. \tag{4.283}$$

正是 ω_{k+1} 的这种极小化取法 $\|\boldsymbol{r}_{k+1}\|_2$ 而使得的振荡性有所改善.

注意: 式 (4.282) 又可写为

$$\boldsymbol{r}_{k+1} = \boldsymbol{s}_k - \omega_{k+1}\boldsymbol{A}\boldsymbol{s}_k = \boldsymbol{r}_k - \alpha_k \boldsymbol{A}\boldsymbol{p}_k - \omega_{k+1}\boldsymbol{A}\boldsymbol{s}_k, \tag{4.284}$$

故可定义

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k + \omega_{k+1}\boldsymbol{s}_k. \tag{4.285}$$

这样定义 $\boldsymbol{x}_{k+1}$ 后, 其残量正好是式 (4.282) 所定义的 $\boldsymbol{r}_{k+1}$.

综合上面的讨论, 便得到如下的稳定化双共轭梯度算法.

算法 4.21 (BCGSTAB 方法) 给定线性方程组 (4.234), 初始向量 $\boldsymbol{x}_0$ 和 $\varepsilon > 0$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中




Back
Close

$\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k.$

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{r}_0$;

选择 $\widetilde{\boldsymbol{r}}_0$ 使 $\rho_0 = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2 > \|\boldsymbol{r}_0\|_2\varepsilon)$

$k = k + 1$;

$\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{p}_k$; $\sigma_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{u}_k$;

$\alpha_k = \rho_k/\sigma_k$; $\boldsymbol{s}_k = \boldsymbol{r}_k - \alpha_k\boldsymbol{u}_k$;

$\boldsymbol{q}_k = \boldsymbol{A}\boldsymbol{s}_k$; $\omega_{k+1} = (\boldsymbol{s}_k^{\mathrm{T}}\boldsymbol{q}_k)/(\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{q}_k)$;

$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k + \omega_{k+1}\boldsymbol{s}_k$;

$\boldsymbol{r}_{k+1} = \boldsymbol{s}_k - \omega_{k+1}\boldsymbol{q}_k$;

$\rho_{k+1} = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_{k+1}$;

$\beta_{k+1} = (\alpha_k\rho_{k+1})/(\omega_{k+1}\rho_k)$;

Back

Close

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_{k+1}(\boldsymbol{p}_k - \omega_{k+1}\boldsymbol{u}_k);$$

end

算法 4.21 也避免了 $\boldsymbol{A}^{\mathrm{T}}$ 的出现, 而且每次迭代也只需作两次矩阵向量乘法. 实际应用显示这种方法收敛得较快, 而且残量的范数较平稳, 是一种比较好的算法.

BCGSTAB 方法的 MATLAB 程序如下:

```
%BCGSTAB方法程序-bcgstab.m
function [x,k,time,res,resvec]=bcgstab(A,b,x,max_it,tol)
tic; r=b-A*x; p=r; mr=norm(r);
rt=ones(length(b),1); %rt=r;
%rt=zeros(length(b),1); rt(1)=1;
rho=rt'*r; k=0;
```

Back

Close

```
while (k<=max_it)
    k=k+1; u=A*p; sigma=rt'*u;
    alpha=rho/sigma; s=r-alpha*u;
    v=A*s; omega=(s'*v)/(v'*v);
    x=x+alpha*p+omega*s;
    r=s-omega*v; rho1=rt'*r;
    beta=(alpha*rho1)/(omega*rho);
    p=r+beta*(p-omega*u);
    res=norm(r)/mr;resvec(k)=res;
    if (res<tol), break; end
    rho=rho1;
end
time=toc;
```

例 4.17 仍然考虑例 4.15 中的 $\boldsymbol{A}$ 和 $\boldsymbol{b}$. 将 BCGSTAB 算法应用到该线性方程组上, 取算法中的 $\widetilde{\boldsymbol{r}}_0 = \mathrm{ones}(n,1)$, 迭代在 101 步后收敛 $(\varepsilon = 10^{-10})$, 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的绝对误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 0.9536,$$

但计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 7.8561 \times 10^{-8}.$$

迭代过程的收敛轨迹如图 4.18 所示, 其中横坐标为迭代步数 k, 纵坐标为相对残差 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$, 这里 $\boldsymbol{r}_k$ 是第 k 步得到的残差向量. 此时的收敛速度与 CGS 方法基本一样, 但是注意 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2$ 已经变为呈平稳下降.

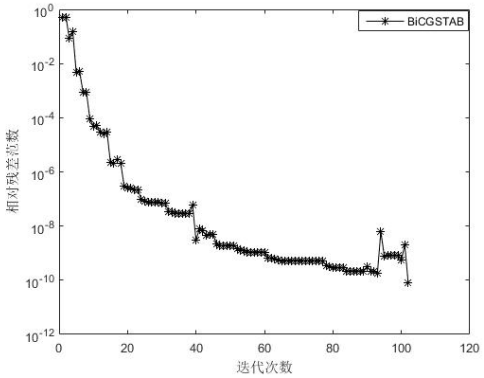


图 4.18 BCGSTAB 算法的收敛特性

最后, 关于预处理 BCGSTAB 算法. 对预处理方程组

$$\widehat{\boldsymbol{A}}\widehat{\boldsymbol{x}} = \widehat{\boldsymbol{b}}, \tag{4.286}$$

式中:

$$\widehat{\boldsymbol{A}} = \boldsymbol{M}_1^{-1}\boldsymbol{A}\boldsymbol{M}_2^{-1}, \quad \widehat{\boldsymbol{x}} = \boldsymbol{M}_2\boldsymbol{x}, \quad \widehat{\boldsymbol{b}} = \boldsymbol{M}_1^{-1}\boldsymbol{b}.$$

对预处理方程组 (4.286) 应用 BCGSTAB 算法, 再作代换:

$$\widehat{\boldsymbol{u}}_k = \boldsymbol{M}_1^{-1}\boldsymbol{u}_k,\ \widehat{\boldsymbol{s}}_k = \boldsymbol{M}_1^{-1}\boldsymbol{s}_k,\ \widehat{\boldsymbol{q}}_k = \boldsymbol{M}_1^{-1}\boldsymbol{q}_k,$$
$$\widehat{\boldsymbol{r}}_k = \boldsymbol{M}_1^{-1}\boldsymbol{r}_k,\ \widehat{\boldsymbol{p}}_k = \boldsymbol{M}_1^{-1}\boldsymbol{p}_k,\ \widehat{\boldsymbol{x}}_k = \boldsymbol{M}_2\boldsymbol{x}_k,\ \widehat{\widetilde{\boldsymbol{r}}}_0 = \boldsymbol{M}_1^{\mathrm{T}}\widetilde{\boldsymbol{r}}_0.$$

便得到如下预处理稳定化双共轭梯度算法.

算法 4.22 (PBCGSTAB 方法) 给定线性方程组 (4.234), 初始向量 $\boldsymbol{x}_0$ 和 $\varepsilon > 0$ 及预处理矩阵 $\boldsymbol{M} = \boldsymbol{M}_1\boldsymbol{M}_2$. 本算法计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k$.

选取 $\boldsymbol{x}_0$; 计算 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$; $\boldsymbol{p}_0 = \boldsymbol{r}_0$;

选择 $\widetilde{\boldsymbol{r}}_0$ 使 $\rho_0 = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$; $k = 0$;

while $(\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 > \varepsilon)$

$k = k + 1$;

由 $\boldsymbol{M}\boldsymbol{y}_k = \boldsymbol{p}_k$ 解得 $\boldsymbol{y}_k$;

$$\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{y}_k;\ \sigma_k = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{u}_k;$$

$$\alpha_k = \rho_k/\sigma_k;\ \boldsymbol{s}_k = \boldsymbol{r}_k - \alpha_k\boldsymbol{u}_k;$$

由 $\boldsymbol{M}\boldsymbol{z}_k = \boldsymbol{s}_k$ 解得 $\boldsymbol{z}_k$; $\boldsymbol{q}_k = \boldsymbol{A}\boldsymbol{z}_k$;

由 $\boldsymbol{M}_1[\boldsymbol{\xi}_k, \boldsymbol{\eta}_k] = [\boldsymbol{q}_k, \boldsymbol{s}_k]$ 解得 $[\boldsymbol{\xi}_k, \boldsymbol{\eta}_k]$;

$$\omega_{k+1} = (\boldsymbol{\xi}_k^{\mathrm{T}}\boldsymbol{\eta}_k)/(\boldsymbol{\xi}_k^{\mathrm{T}}\boldsymbol{\xi}_k);$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\boldsymbol{y}_k + \omega_{k+1}\boldsymbol{z}_k;$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{s}_k - \omega_{k+1}\boldsymbol{q}_k;$$

$$\rho_{k+1} = \widetilde{\boldsymbol{r}}_0^{\mathrm{T}}\boldsymbol{r}_{k+1};\ \beta_{k+1} = (\alpha_k\rho_{k+1})/(\omega_{k+1}\rho_k);$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_{k+1}(\boldsymbol{p}_k - \omega_{k+1}\boldsymbol{u}_k);$$

end

若考虑例 4.15 中的 $\boldsymbol{A}$ 和 $\boldsymbol{b}$. 将 PBCGSTAB 方法应用到该线性方程组上, 取算法中的 $\widetilde{\boldsymbol{r}}_0 = \boldsymbol{r}_0$, 预处理矩阵 $\boldsymbol{M} = \boldsymbol{M}_1\boldsymbol{M}_2$ 为矩阵

$\boldsymbol{A}$ 的不完全 LU 分解, 则只需迭代 1 步即满足终止准则 $(\varepsilon = 10^{-10})$, 计算得到的近似解 $\widehat{\boldsymbol{x}}$ 和真解 $\boldsymbol{x}^*$ 之间的误差为

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2 = 5.7303 \times 10^{-15},$$

计算解 $\widehat{\boldsymbol{x}}$ 的残量满足

$$\|\boldsymbol{b} - \boldsymbol{A}\widehat{\boldsymbol{x}}\|_2 = 6.0738 \times 10^{-15}.$$