第四讲　Krylov 子空间方法

- 投影方法
- Krylov 子空间与 Arnoldi 过程
- 一般线性方程组的 Krylov 子空间方法
- 对称线性方程的 Krylov 子空间方法
- 收敛性分析
- 基于双正交化过程的迭代方法
- 免转置迭代方法
- 正规方程的迭代方法

大规模稀疏线性方程组

$$Ax = b, \quad A \in \mathbb{R}^{n\times n},\ b \in \mathbb{R}^n.$$

Krylov 子空间方法是子空间方法的成功代表

首选方法 Krylov 子空间方法.

基本思想 在一个维数较小的子空间 $\mathcal{K} \subset \mathbb{R}^n$ 中寻找近似解.

这类方法也被看作是一种 **投影方法**, 即寻找真解在某个子空间中的投影 (可以是正交投影, 也可以是斜投影)

如果没有特别注明, 本讲内容都是在 实数域 中讨论.

1 投影方法

设 $\mathcal{K}$ 是 $\mathbb{R}^n$ 的一个子空间, 记 $m \triangleq \dim(\mathcal{K}) \ll n$

m 维空间的向量有 m 个自由度, 因此需要加 m 的限制条件

目标 在 $\mathcal{K}$ 中寻找真解的一个 "最佳" 近似.

定解条件 设置 m 个约束: 要求残量满足 m 个正交性条件, 即

$$r = b - A\tilde{x} \perp \mathcal{L} \tag{4.1}$$

Petrov-Galerkin 条件

Galerkin 条件: $\mathcal{L} = \mathcal{K}$

其中 $\tilde{x}$ 是近似解, $\mathcal{L}$ 是另一个 m 维子空间.

$\mathcal{L}$ 称为 约束空间

$\mathcal{K}$ 称为 搜索空间

不同的 $\mathcal{L}$ 对应不同的投影方法

当 $\mathcal{L} = \mathcal{K}$ 时, 我们称为 **正交投影法**, 否则称为 **斜投影法**

投影方法的数学描述

$$\text{find } \tilde{x} \in \mathcal{K} \quad \text{such that} \quad b - A\tilde{x} \perp \mathcal{L} \tag{4.2}$$

若给定初值 $x^{(0)} \in \mathbb{R}^n$, 则改用仿射空间 $x^{(0)} + \mathcal{K}$, 即

$$\text{find } \tilde{x} \in x^{(0)} + \mathcal{K} \quad \text{such that} \quad b - A\tilde{x} \perp \mathcal{L}. \tag{4.3}$$

好的初值一般都包含有价值的信息

事实上, 如果将 $\tilde{x}$ 写成: $\tilde{x} = x^{(0)} + \hat{x}$, 其中 $\hat{x} \in \mathcal{K}$, 则 (4.3) 就等价于

$$\text{find } \hat{x} \in \mathcal{K} \quad \text{such that} \quad r_0 - A\hat{x} \perp \mathcal{L}, \tag{4.4}$$

求解 $\tilde{x}$ 等价于求解 $\hat{x}$

其中 $r_0 = b - Ax^{(0)}$ 是初始残量. 这与 (4.2) 的形式是一样的.

如何计算近似解

设 $\{v_1, v_2, \ldots, v_m\}$ 和 $\{w_1, w_2, \ldots, w_m\}$ 分别是 $\mathcal{K}$ 和 $\mathcal{L}$ 一组基.

$W \in \mathbb{R}^{n\times m}, V \in \mathbb{R}^{n\times m}$

令 $V = [v_1, v_2, \ldots, v_m], \quad W = [w_1, w_2, \ldots, w_m]$.

由于 $\tilde{x} \in x^{(0)} + \mathcal{K}$, 因此存在向量 $y \in \mathbb{R}^m$ 使得 $\tilde{x} = x^{(0)} + Vy$

$\hat{x} \triangleq \tilde{x} - x^{(0)} = Vy$

由正交性条件 (4.4) 可知 $r_0 - AVy \perp w_i,\ i = 1, 2, \ldots, m$, 即

$$W^\intercal AVy = W^\intercal r_0 .$$

在实际计算中, $W^\intercal AV$ 通常可以直接形成, 无需另外计算.

若 $W^\intercal AV$ 非奇异, 则解得 $y = (W^\intercal AV)^{-1} W^\intercal r_0$. 因此

$$\tilde{x} = x^{(0)} + V(W^\intercal AV)^{-1} W^\intercal r_0 .$$

近似解 $\tilde{x}$ 存在唯一的条件是矩阵 $W^{\intercal}AV$ 非奇异.

定理 1 如果 $A, \mathcal{K}$ 和 $\mathcal{L}$ 满足下面条件之一

(1) A 正定且 $\mathcal{L} = \mathcal{K}$;
(2) A 非奇异且 $\mathcal{L} = A\mathcal{K}$,

则矩阵 $W^{\intercal}AV$ 非奇异.

证明留作练习

如何实施投影方法

在实施投影方法时, 我们需要考虑下面两个问题:

- 怎样选择搜索空间 $\mathcal{K}$ 和约束空间 $\mathcal{L}$?
- 如果找到的近似解 $\tilde{x}$ 达不到精度要求, 则需要更换搜索空间 $\mathcal{K}$ 和约束空间 $\mathcal{L}$, 如何更新?

目前能够很好地解决上面两个问题的方法就是 **Krylov 子空间方法**

2 Krylov 子空间与 Arnoldi 过程

Krylov 子空间

设 $A \in \mathbb{R}^{n\times n}, r \in \mathbb{R}^n$, 我们称

$$\mathcal{K}_m(A, r) \triangleq \text{span}\{r, Ar, \ldots, A^{m-1}r\} \subseteq \mathbb{R}^n$$

是由 A 和 r 生成的 **Krylov 子空间**. 为书写方便, 通常简记为 $\mathcal{K}_m$.

Krylov 子空间方法就是在 Krylov 子空间中寻找近似解

几个基本性质:

- Krylov 子空间是嵌套的, 即: $\mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \cdots \subseteq \mathcal{K}_m \subseteq \cdots$
- $\mathcal{K}_m$ 的维数不超过 m
- $\mathcal{K}_m(A, r) = \left\{x = p(A)r : p \text{ 为次数不超过 } m-1 \text{ 的多项式}\right\}$

约束子空间 $\mathcal{L}$ 的选取

目前被广泛采用的选取方法有:

- $\mathcal{L} = \mathcal{K}$: 如 FOM, CG, SYMMLQ
- $\mathcal{L} = A\mathcal{K}$: 如 MINRES, GMRES
- $\mathcal{L} = \mathcal{K}(A^{\intercal}, r)$: 如 BiCG

通过对这些方法进行变形与改进, 可以构造更多的子空间方法.

Arnoldi 过程: 计算 $\mathcal{K}_m$ 的一组正交基

算法 2.1 基于 Gram-Schmidt 正交化的 Arnoldi 过程

给定非零向量 r, 计算 $v_1 = r/\|r\|_2$
for $j = 1, 2, \ldots, m-1$ **do**
  $w_j = Av_j$
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} = (w_j, v_i)$
  end for
  $w_j = w_j - \sum_{i=1}^{j} h_{ij} v_i$
  $h_{j+1,j} = \|w_j\|_2$
  if $h_{j+1,j} = 0$ **then**
    break
  end if
  $v_{j+1} = w_j / h_{j+1,j}$
end for

做计算时一定要注意任何可能存在的 “中断”.

如果到第 k ($k < m$) 步时有 $h_{k+1,k} = 0$, 则算法将 **提前终止**. 此时 Av_k 必定可由 $v_1, v_2, \ldots, v_k$ 线性表出 (不考虑浮点运算误差)

向量 v_i 称为 **Arnoldi 向量**

需要注意的是, 在算法中, 我们是用 A 乘以 v_j, 然后与之前的 Arnoldi 向量正交, 而不是计算 $A^j r$. 事实上, 它们是等价的.

引理 1 设算法 2.1 不提前终止 (即 $h_{k+1,k} \neq 0, k = 1, 2, \ldots, m-1$), 则

$$v_j \in \mathcal{K}_j(A, r), \quad j = 1, 2, \ldots, m.$$

(板书)

$\mathcal{K}_j(A, r) = \operatorname{span}\{v_1, \ldots, v_j\}$

根据上面的引理, 我们可以立即得到下面的结论.

定理 2 如果算法 2.1 不提前终止, 则向量 $v_1, v_2, \ldots, v_m$ 构成 $\mathcal{K}_m$ 的一组标准正交基, 其中

$$\mathcal{K}_m(A, r) = \text{span}\{r, Ar, \ldots, A^{m-1}r\}.$$

Arnoldi 过程的重要性质

由 Arnoldi 过程可知 $h_{j+1,j}v_{j+1} = Av_j - \sum_{i=1}^{j} h_{ij}v_i$, 因此

$$w_j = w_j - \sum_{i=1}^{j} h_{ij}v_i$$
$$v_{j+1} = w_j / h_{j+1,j}$$

$$Av_j = h_{j+1,j}v_{j+1} + \sum_{i=1}^{j} h_{ij}v_i$$

$$= [v_1, v_2, \ldots, v_{j+1}] \begin{bmatrix} h_{1j} \\ h_{2j} \\ \vdots \\ h_{j+1,j} \end{bmatrix} = [v_1, \ldots, v_{j+1}, v_{j+2}, \ldots, v_{m+1}] \begin{bmatrix} h_{1j} \\ \vdots \\ h_{j+1,j} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$= V_{m+1}H_{m+1,m}(:, j),$$

其中 $V_{m+1} = [v_1, v_2, \ldots, v_{m+1}]$,

$$H_{m+1,m} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1,m-1} & h_{1,m} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2,m-1} & h_{2,m} \\ 0 & h_{32} & h_{33} & \cdots & h_{3,m-1} & h_{2,m} \\ 0 & 0 & h_{43} & \cdots & h_{4,m-1} & h_{4,m} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & h_{m,m-1} & h_{m,m} \\ 0 & 0 & 0 & \cdots & 0 & h_{m+1,m} \end{bmatrix} \in \mathbb{R}^{(m+1)\times m}$$

$w_j = Av_j$
for $i = 1, 2, \ldots, j$ **do**
  $h_{ij} = (w_j, v_i)$
end for
$w_j = w_j - \sum_{i=1}^{j} h_{ij} v_i$
$h_{j+1,j} = \|w_j\|_2$

这里 h_{ij} 是由 Arnoldi 过程所定义的.

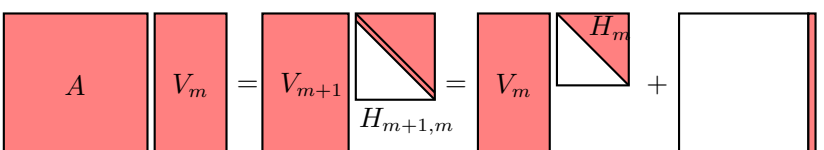
定理 3 设 $V_m = [v_1, v_2, \ldots, v_m]$, 则

$$AV_m = V_{m+1}H_{m+1,m} = V_m H_m + h_{m+1,m} v_{m+1} e_m^{\intercal} \tag{4.5}$$

$$V_m^{\intercal} A V_m = H_m \tag{4.6}$$

其中 $e_m = [0, \ldots, 0, 1]^{\intercal} \in \mathbb{R}^m$, $H_m = H_{m+1,m}(1:m, 1:m) \in \mathbb{R}^{m\times m}$, 即 H_m 是由 $H_{m+1,m}$ 的前 m 行组成的 Hessenberg 矩阵.

这两个公式在后面的算法推导过程中非常重要



如果 Arnoldi 方法提前终止, 则我们可以得到一个不变子空间.

定理 4 如果算法 2.1 在第 k $(k \leq m)$ 步终止, 即 $h_{k+1,k} = 0$, 则有

$$AV_k = V_k H_k,$$

即 $\mathcal{K}_k$ 是 A 的一个不变子空间.

一旦出现不变子空间, Arnoldi 过程就会被中断, 这是好事还是坏事?

考虑到数值稳定性, 通常使用 **修正的 Gram-Schmidt 过程 (MGS)**

算法 2.2 基于 MGS 的 Arnoldi 过程

1: 给定非零向量 r
2: $v_1 = r/\|r\|_2$
3: **for** $j = 1, 2, \ldots, m-1$ **do**
4: $\quad w_j = Av_j$
5: $\quad$ **for** $i = 1, 2, \ldots, j$ **do**
6: $\quad\quad h_{ij} = (w_j, v_i)$
7: $\quad\quad w_j = w_j - h_{ij}v_i$
8: $\quad$ **end for**
9: $\quad h_{j+1,j} = \|w_j\|_2$
10: $\quad$ **if** $h_{j+1,j} = 0$ **then**
11: $\quad\quad$ break
12: $\quad$ **end if**
13: $\quad v_{j+1} = w_j/h_{j+1,j}$
14: **end for**

$w_j = Av_j$
for $i = 1, 2, \ldots, j$ **do**
$\quad h_{ij} = (w_j, v_i)$
end for
$w_j = w_j - \sum_{i=1}^{j} h_{ij}v_i$

注记

基于 MGS 的 Arnoldi 过程与基于 CGS 的 Arnoldi 过程是等价的, 但 MGS 更稳定. 已有学者证明修正 Gram-Schmidt 过程是向后稳定的.

注记

但在某些极端情况下, 算法 2.2 仍不够可靠, 此时可以再做一次 MGS, 或者采用 Householder 变换.

3 一般线性方程组的 Krylov 子空间方法

给定一个迭代初始值 $x^{(0)} \in \mathbb{R}^n$, 令

$$\mathcal{K}_m = \text{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{m-1}r_0\},$$

其中 $r_0 = b - Ax^{(0)}$ 为初始残量.

下面介绍分别基于 $\mathcal{L} = \mathcal{K}_m$ 和 $\mathcal{L} = A\mathcal{K}_m$ 的投影方法:

- 完全正交方法: $\mathcal{L} = \mathcal{K}_m$
- 广义极小残量法: $\mathcal{L} = A\mathcal{K}_m$

3.1 完全正交方法 (FOM): $\mathcal{L} = \mathcal{K}_m$

寻找 $\tilde{x} \in x^{(0)} + \mathcal{K}_m$ **满足** $b - A\tilde{x} \perp \mathcal{K}_m$

由 Arnoldi 过程可知

$$V_m^\intercal A V_m = H_m \quad \text{且} \quad V_m^\intercal r_0 = V_m^\intercal(\beta v_1) = \beta e_1,$$

其中 $\beta = \|r_0\|_2,\ e_1 = [1, 0, \ldots, 0]^\intercal \in \mathbb{R}^m$.

由 $\tilde{x} \in x^{(0)} + \mathcal{K}_m$ **可知, 存在向量** $\tilde{y} \in \mathbb{R}^m$ **使得**

$$\tilde{x} = x^{(0)} + V_m \tilde{y}.$$

由正交性条件 $b - A\tilde{x} \perp \mathcal{K}_m$ **可得**

$$V_m^\intercal (b - A\tilde{x}) = 0,$$

矩阵 V_m 的列向量构成 $\mathcal{K}_m$ 的一组基

与 $\mathcal{K}_m$ 正交等价于与 $\mathcal{K}_m$ 的一组基正交

即

$$0 = V_m^\intercal(b - Ax^{(0)} - AV_m\tilde{y}) = V_m^\intercal r_0 - V_m^\intercal AV_m\tilde{y} = \beta e_1 - H_m\tilde{y}.$$

如果矩阵 H_m 非奇异, 则 $\tilde{y} = \beta H_m^{-1} e_1$, 所以

$$\tilde{x} = x^{(0)} + \beta V_m H_m^{-1} e_1$$

算法 3.1 完全正交方法 (FOM)

1: **给定初值** $x^{(0)}$，　**计算** $r_0 = b - Ax^{(0)}, \beta = \|r_0\|_2$ **和** $v_1 = r_0/\beta$
2: **for** $j = 1, 2, \ldots, m-1$ **do**
3: 　　$w_j = Av_j$
4: 　　**for** $i = 1, 2, \ldots, j$ **do**
5: 　　　　$h_{ij} = (w_j, v_i)$
6: 　　　　$w_j = w_j - h_{ij}v_i$
7: 　　**end for**
8: 　　$h_{j+1,j} = \|w_j\|_2$
9: 　　**if** $h_{j+1,j} = 0$ **then**
10: 　　　　$m = j$, break
11: 　　**end if**
12: 　　$v_{j+1} = w_j/h_{j+1,j}$
13: **end for**
14: **求解线性方程组** $H_m\tilde{y} = \beta e_1$　　**并计算近似解** $\tilde{x} = x^{(0)} + V_m\tilde{y}$

注记

由于子空间 $\mathcal{K}_m$ 的维数远远小于 n, 因此方程组 $H_m\tilde{y} = \beta e_1$ 可以通过直接法求解, 如列主元 Gauss 消去法.

事实上, 由于 H_m 是一个上 Hessenberg 矩阵, 我们也可以采用基于 Givens 变换的 QR 分解来求解. 这样更稳定.

注记

FOM 方法的一个难点是如何选取 m. 如果 m 太小, 则近似解 $\tilde{x}$ 可能达不到精度要求, 如果太大, 可能会多做一些无用功. 因此在实际计算时, 我们采用动态的方法, 即不断增长 m 的值. 当残量满足精度要求时就停止迭代. 因此, 在每一次迭代后, 我们都需要估计残量的范数.

残量估计

定理 5 设 $\tilde{x} \in x^{(0)} + \mathcal{K}_m$ 由算法 3.1 得到的近似解, 则残量表达式为

$$\tilde{r} \triangleq b - A\tilde{x} = -h_{m+1,m}\,(e_m^{\intercal}\tilde{y})\,v_{m+1}$$

其中 $\tilde{y} = \beta H_m^{-1} e_1$, $e_m = [0, \ldots, 0, 1]^{\intercal} \in \mathbb{R}^m$. 因此,

$$\|\tilde{r}\|_2 = h_{m+1,m}|e_m^{\intercal}\tilde{y}| = h_{m+1,m}|\tilde{y}(m)|$$

(板书)

$AV_m =$
$V_m H_m + h_{m+1,m} v_{m+1} e_m^{\intercal}$

注记

由定理 5 可知, 我们可以直接计算出残量 $\tilde{r}$ 的大小, 而无需在每次迭代中计算近似解 $\tilde{x}$. 当残量满足精度要求时, 我们再计算近似解.

注记

在不考虑舍入误差的情况下, 当 $m = n$ 时, 必有 $\|\tilde{r}\|_2 = 0$.

注记

如果 Arnoldi 过程提前终止, 即存在整数 k $(k < n)$ 使得 $h_{k+1,k} = 0$, 则由定理 5 可知, 此时 $\tilde{r} = 0$, 因此近似解 $\tilde{x}$ 就是方程组的精确解.

算法 3.2 实用 FOM 方法

1: 给定初值 $x^{(0)}$ 和 (相对) 精度要求 $\varepsilon > 0$
2: 计算 $r_0 = b - Ax^{(0)}$, $\beta = \|r_0\|_2$ 和 $v_1 = r_0/\beta$
3: **for** $j = 1, 2, \ldots$ **do**
4: $\quad w_j = Av_j$
5: $\quad$ **for** $i = 1, 2, \ldots, j$ **do**
6: $\quad\quad h_{ij} = (w_j, v_i)$
7: $\quad\quad w_j = w_j - h_{ij}v_i$
8: $\quad$ **end for**
9: $\quad h_{j+1,j} = \|w_j\|_2$
10: $\quad$ 求解方程 $H_j\tilde{y} = \beta e_1$
11: $\quad$ **if** $h_{j+1,j}|\tilde{y}(j)|/\beta < \varepsilon$ **then** % relative residual
12: $\quad\quad$ break
13: $\quad$ **end if**
14: $\quad v_{j+1} = w_j/h_{j+1,j}$
15: **end for**
16: 计算近似解 $\tilde{x} = x^{(0)} + V_j\tilde{y}$

3.2 广义极小残量法 (GMRES): $\mathcal{L} = A\mathcal{K}$

广义极小残量法 (Generalized Minimum Residual) **可描述为**

$$\text{寻找 } \tilde{x} \in x^{(0)} + \mathcal{K} \quad \text{满足} \quad b - A\tilde{x} \perp A\mathcal{K}. \tag{4.7}$$

GMRES 是当前求解大规模非对称线性方程组的首选方法

定理 6 设 $A \in \mathbb{R}^{n\times n}, \mathcal{K} = \mathcal{K}(A, r_0)$, 则 $\tilde{x}$ 是问题 (4.7) 的解当且仅当 $\tilde{x}$ 是下面的极小化问题的解

$$\min_{x \in x^{(0)}+\mathcal{K}} \|b - Ax\|_2, \tag{4.8}$$

即在仿射空间 $x^{(0)} + \mathcal{K}$ 中, $\tilde{x}$ 所对应的残量范数最小. (板书)

与 FOM 的推导过程不同, 我们需要用另外一种方式来推导 GMRES

这个最优性质是 GMRES 方法名称的由来

GMRES 的推导过程

任意向量 $x \in x^{(0)} + \mathcal{K}_m$ 都可表示为 $x = x^{(0)} + V_m y$, 其中 $y \in \mathbb{R}^m$, 故

$$\begin{aligned} b - Ax &= b - A(x^{(0)} + V_m y) \\ &= r_0 - AV_m y \\ &= \beta v_1 - V_{m+1} H_{m+1,m} y = V_{m+1}(\beta e_1 - H_{m+1,m} y), \end{aligned} \tag{4.9}$$

其中 $e_1 = [1, 0, \ldots, 0]^\intercal \in \mathbb{R}^{m+1}$. 又 V_{m+1} 的列向量标准正交的, 所以

$$\|b - Ax\|_2 = \|V_{m+1}(\beta e_1 - H_{m+1,m} y)\|_2 = \|\beta e_1 - H_{m+1,m} y\|_2. \tag{4.10}$$

因此, 极小化问题 (4.8) 的解可表示为

$$\tilde{x} = x^{(0)} + V_m \tilde{y}, \tag{4.11}$$

其中 $\tilde{y}$ 是下面最小二乘问题的解

$$\min_{y\in\mathbb{R}^m} \|\beta e_1 - H_{m+1,m}y\|_2 . \tag{4.12}$$

注记

与 FOM 方法中求解一个线性方程组不同的是, 在 GMRES 方法中, 我们是求解一个最小二乘问题.

最小二乘求解

一般 m 不会很大, 因此可以使用 QR 方法来求解 (4.12). 设 $H_{m+1,m} = Q_{m+1}^{\mathsf{T}} R_{m+1,m}$ 是 $H_{m+1,m}$ 的 QR 分解, 其中 $Q_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$ 是正交矩阵, $R_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ 是上三角矩阵. 于是

$$\begin{aligned}\|\beta e_1 - H_{m+1,m} y\|_2 &= \|\beta e_1 - Q_{m+1}^{\mathsf{T}} R_{m+1,m} y\|_2 \\ &= \|\beta Q_{m+1} e_1 - R_{m+1,m} y\|_2 = \left\| \beta q_1 - \begin{bmatrix} R_m \\ 0 \end{bmatrix} y \right\|_2,\end{aligned}$$

其中 q_1 是 Q_{m+1} 的第一列, $R_m = R_{m+1,m}(1:m,:) \in \mathbb{R}^{m\times m}$ 表示 $R_{m+1,m}$ 的前 m 行. 所以 y 可以通过求解下面的上三角方程组获得

$$R_m \tilde{y} = \beta q_1(1:m), \tag{4.13}$$

其中 $q_1(1:m)$ 表示 q_1 的前 m 个元素组成的向量.

残量计算

与 FOM 方法类似, 我们也可以直接得到残量范数的表达式.

定理 7 设 $\tilde{x} \in x^{(0)} + \mathcal{K}_m$ 是由 GMRES 方法所得到的近似解, 则

$$\|\tilde{r}\|_2 = \|b - A\tilde{x}\|_2 = \beta \, |q_1(m+1)|$$

其中 $q_1(m+1)$ 表示 $q_1 \in \mathbb{R}^{m+1}$ 的最后一个分量.

$$\begin{aligned}
\|\tilde{r}\|_2 &= \|b - A\tilde{x}\|_2 \\
&= \|\beta e_1 - H_{m+1,m}\tilde{y}\|_2 \\
&= \left\| \beta q_1 - \begin{bmatrix} R_m \\ 0 \end{bmatrix} \tilde{y} \right\|_2 \\
&= \beta \, |q_1(m+1)|.
\end{aligned}$$

算法 3.3 广义极小残量法 (GMRES)

1: **给定初值** $x^{(0)}$ **和 (相对) 精度要求** $\varepsilon > 0$
2: **计算** $r_0 = b - Ax^{(0)}$ **和** $\beta = \|r_0\|_2$
3: $v_1 = r_0/\beta$
4: **for** $j = 1, 2, \ldots$ **do**
5: $\quad w_j = Av_j$
6: $\quad$ **for** $i = 1, 2, \ldots, j$ **do** % Arnoldi process
7: $\quad\quad h_{ij} = (w_j, v_i)$
8: $\quad\quad w_j = w_j - h_{ij}v_i$
9: $\quad$ **end for**
10: $\quad h_{j+1,j} = \|w_j\|_2$
11: $\quad$ **if** $h_{j+1,j} = 0$ **then**
12: $\quad\quad m = j$, break
13: $\quad$ **end if**

14: $v_{j+1} = w_j / h_{j+1,j}$

15: **if** $\|\tilde{r}\|_2 / \beta < \varepsilon$ **then** % check convergence

16: $m = j$, break

17: **end if**

18: **end for**

19: 求解上三角线性方程组 (4.13)

20: 计算近似解 $\tilde{x} = x^{(0)} + V_m \tilde{y}$

算法的中断

在 GMRES 算法中, 如果第 k ($k < n$) 步时 $h_{k+1,k} = 0$, 则提前终止, 且

$$AV_k = V_k H_k, \quad \tilde{y} = H_k^{-1}(\beta e_1).$$

$\min \|\beta e_1 - H_{k+1,k} y\|_2$
$H_{k+1,k}$ 最后一行全为 0

因此

$$\begin{aligned} \|\tilde{r}\|_2 = \|b - A\tilde{x}\|_2 &= \|b - Ax^{(0)} - AV_k\tilde{y}\|_2 \\ &= \|r_0 - V_k H_k \tilde{y}\|_2 \\ &= \|\beta v_1 - V_k(\beta e_1)\|_2 \\ &= 0. \end{aligned}$$

这意味着 $\tilde{x}$ 就是原方程组的精确解. 因此, 这种提前终止是好事.

反之, 如果在第 k 步时有 $\tilde{r} = b - A\tilde{x} = 0$, 则必有 $h_{k+1,k} = 0$.

定理 8 设 $A \in \mathbb{R}^{n\times n}$ 非奇异, 则 GMRES 方法在第 k 步提前终止的充要条件是近似解即为精确解.

证明. 留作练习. □

3.3 GMRES 方法的实施细节

我们注意到, 在 GMRES 方法中, 计算残量的范数时需要计算 $H_{j+1,j}$ 的 QR 分解, 工作量太大!

在实际计算中, 我们采用递推方式和 Givens 变换, 在 $H_{j,j-1}$ 的 QR 分解基础上, 通过一次 Givens 变换得到 $H_{j+1,j}$ 的 QR 分解, 从而大大降低运算量.

下面我们详细描述具体的递推计算过程.

第一步

当 $j=1$ 时, 只需对 $H_{2,1}=\begin{bmatrix} h_{11} \\ h_{21} \end{bmatrix}$ 做一次 Givens 变换即可得 QR 分解

第二步

假定 $H_{j,j-1} \in \mathbb{R}^{j\times(j-1)}$ 的 QR 分解为

$$H_{j,j-1} = (G_{j-1}G_{j-2}\cdots G_1)^\intercal R_{j,j-1} = Q_j^\intercal \begin{bmatrix} R_{j-1} \\ 0 \end{bmatrix}_{j\times(j-1)}$$

为了确保矩阵乘积的相容性, 我们假定 G_i 的维数会根据需要自动增大, 即在右下角添加单位阵.

其中 $R_{j-1} \in \mathbb{R}^{(j-1)\times(j-1)}$ 是上三角矩阵, $Q_j = G_1G_2\cdots G_{j-1}$. 这里 G_i 都是 Givens 变换.

第三步

考虑 $H_{j+1,j}$ 的 QR 分解. 易知

$$H_{j+1,j} = \begin{bmatrix} H_{j,j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix},$$

其中 $h_j = [h_{1,j}, h_{2,j}, \ldots, h_{j,j}]^\intercal$ 是由 $H_{j+1,j}$ 最后一列的前 j 个元素构

成的向量. 于是

$$\begin{bmatrix} Q_j & 0 \\ 0 & 1 \end{bmatrix} H_{j+1,j} = \begin{bmatrix} Q_j & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} H_{j,j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}$$

$$= \begin{bmatrix} R_{j,j-1} & Q_j h_j \\ 0 & h_{j+1,j} \end{bmatrix} = \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \hat{h}_{j,j} \\ 0 & h_{j+1,j} \end{bmatrix},$$

其中 $\tilde{h}_{j-1}$ 是 $Q_j h_j$ 的前 $j-1$ 个元素构成的向量, $\hat{h}_{j,j}$ 是 $Q_j h_j$ 的最后一个元素. 下面我们利用 Givens 变换, 将 $h_{j+1,j}$ 化为 0. 计算

$$\tilde{h}_{j,j} = \sqrt{\hat{h}_{j,j}^2 + h_{j+1,j}^2}, \quad c_j = \frac{\hat{h}_{j,j}}{\tilde{h}_{j,j}}, \quad s_j = \frac{h_{j+1,j}}{\tilde{h}_{j,j}}.$$

构造 Givens 变换

$$G_j = \begin{bmatrix} I & 0 & 0 \\ 0 & c_j & s_j \\ 0 & -s_j & c_j \end{bmatrix} \in \mathbb{R}^{(j+1)\times(j+1)}.$$

令

$$Q_{j+1} = G_j \begin{bmatrix} Q_j & 0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{(j+1)\times(j+1)},$$

则

$$Q_{j+1}H_{j+1,j} = G_j \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \hat{h}_{j,j} \\ 0 & h_{j+1,j} \end{bmatrix} = \begin{bmatrix} R_{j-1} & \tilde{h}_{j-1} \\ 0 & \tilde{h}_{j,j} \\ 0 & 0 \end{bmatrix} \triangleq R_{j+1,j}.$$

所以 $H_{j+1,j} = Q_{j+1}^{\intercal} R_{j+1,j}$ 就是 $H_{j+1,j}$ 的 QR 分解.

注记

在上述过程中, 我们只需将 Givens 变换 $G_1, G_2, \ldots, G_j$ 依次作用到 $H_{j+1,j}$ 最后一列上, 就可以得到 R_j 的最后一列.

当残量充分小时, 我们再通过解方程 (4.13) 得到 $\tilde{y}$.

由 $Q_{m+1} = G_m G_{m-1} \cdots G_1$ 可知, $Q_{m+1}(\beta e_1)$ 可通过将 $G_1, G_2, \ldots, G_m$ 依次作用到向量 βe_1 上得到.

在 GMRES 的具体实现过程. 为节省存储, 我们将 R_m 存在 $H_{m+1,m}$ 中.

算法 3.4 实用的 GMRES 方法

1: 给定初值 $x^{(0)}$ 和 (相对) 精度要求 $\varepsilon > 0$
2: 计算 $r_0 = b - Ax^{(0)}$ 和 $\beta = \|r_0\|_2$
3: $v_1 = r_0/\beta$, $\xi = \beta e_1$
4: **for** $j = 1, 2, \ldots$ **do**
5: $\quad w_j = Av_j$
6: $\quad$ **for** $i = 1, 2, \ldots, j$ **do** % Arnoldi process
7: $\quad\quad h_{ij} = (w_j, v_i)$
8: $\quad\quad w_j = w_j - h_{ij}v_i$
9: $\quad$ **end for**
10: $\quad h_{j+1,j} = \|w_j\|_2$
11: $\quad$ **for** $i = 1, 2, \ldots, j-1$ **do** % Apply $G_{j-1}, \ldots, G_1$ to the last column of $H_{j+1,j}$

$$\begin{bmatrix} h_{i,j} \\ h_{i+1,j} \end{bmatrix} = \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix} \begin{bmatrix} h_{i,j} \\ h_{i+1,j} \end{bmatrix}$$

end for

if $h_{j+1,j} = 0$ **then**

set $m = j$ and break

end if

$v_{j+1} = w_j / h_{j+1,j}$

if $|h_{j,j}| > |h_{j+1,j}|$ **then** % Form the Givens rotation G_j

$c_j = \frac{1}{\sqrt{1+\tau^2}}, s_j = c_j\tau$ where $\tau = \frac{h_{j+1,j}}{h_{j,j}}$

else

$s_j = \frac{1}{\sqrt{1+\tau^2}}, c_j = s_j\tau$ where $\tau = \frac{h_{j,j}}{h_{j+1,j}}$

end if

$h_{j,j} = c_j h_{j,j} + s_j h_{j+1,j}, h_{j+1,j} = 0$ % Apply G_j to last column of $H_{j+1,j}$

24: $\begin{bmatrix} \xi_j \\ \xi_{j+1} \end{bmatrix} = \begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} \begin{bmatrix} \xi_j \\ 0 \end{bmatrix}$ % Apply G_j to the right-hand side

25: **if** $|\xi_{j+1}|/\beta < \varepsilon$ **then** % Check convergence: $\|\tilde{r}\|_2 = |\xi_{j+1}|$

26: set $m = j$ and break

27: **end if**

28: **end for**

29: 计算 $\tilde{y} = R_m^{-1}\xi^{(m)}$, 其中 $R_m = H(1:m, 1:m), \xi^{(m)} = \xi(1:m)$

30: 计算近似解 $\tilde{x} = x^{(0)} + V_m\tilde{y}$

注记

在 FOM 方法和 GMRES 方法中, 都是采用 MGS 来构造 Krylov 子空间的标准正交基, 在某些特殊情况下, 可能需要使用 Householder 变换来增加方法的稳定性.

注记

在不考虑舍入误差的情况下, 当 GMRES 方法迭代到 n 步时, 残量肯定为 0, 因此方法肯定收敛.

注记

对于大规模的线性方程组, 我们不可能迭代 n 步, 因为 GMRES 每一步迭代的运算量和存储量都会随着迭代步数的增加而增加, 所以当迭代步数增大时, 运算时间和存储量会大大增加, 这使得方法失去竞争性.

带重启的 GMRES

我们希望将迭代步数控制在一个能接受的范围内.

目前实现这一目标的通用做法是采用重启策略, 即给定一个最大迭代步数 m (通常远远小于 n, 如 20, 50 等) 如果 GMRES 迭代到 m 步后仍不收敛, 则计算出近似解 $x^{(m)} \in x^{(0)} + \mathcal{K}_m$. 然后将其作为新的迭代初值, 即令 $x^{(0)} = x^{(m)}$, 重新启动 GMRES 方法. 该过程可以重复执行, 直到找到满意的近似解为止.

算法 3.5 GMRES(m): 带重启的 GMRES 方法

1: 给定正整数 $m \ll n$, 初值 $x^{(0)}$ 和 (相对) 精度要求 $\varepsilon > 0$
2: 计算 $r_0 = b - Ax^{(0)}$ 和 $\beta = \|r_0\|_2$
3: $v_1 = r_0/\beta$
4: **for** $j = 1, 2, \ldots, m$ **do**
5: 执行 GMRES 方法 3.4 的第 5 步至第 27 步
6: **end for**
7: 计算 $\tilde{y} = R_m^{-1}\xi^{(m)}$, 其中 $R_m = H(1:m, 1:m)$, $\xi^{(m)} = \xi(1:m)$
8: 计算近似解 $\tilde{x} = x^{(0)} + V_m\tilde{y}$
9: **if** $|\xi_{m+1}|/\beta < \varepsilon$ **then**
10: stop
11: **end if**
12: 令 $x^{(0)} = x^{(m)}$, 返回第 2 步

注记

加入重启技术可以节省运算量和存储量, 但随之也会带来一些负面效应. 如收敛速度会减慢, 这是因为在重启方法时, 一些有用信息会被丢弃.

如果 A 不是正定矩阵, 则可能还会出现方法停滞不前的情形, 即残量范数很难减小. 此时即使总迭代步数已经达到 n 步, 或者甚至超过 n 步, 但方法仍然不收敛.

另外, 怎样选取合适的重启步数 m, 目前仍然没有很好的办法, 经验很重要!

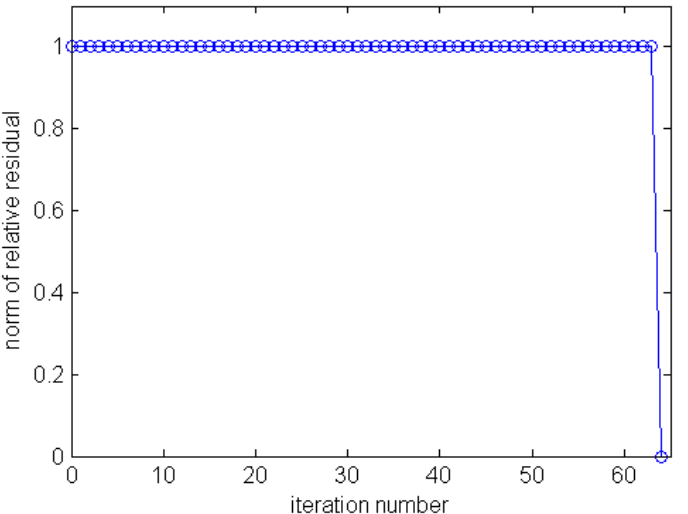
3.4 举例

由最优性可知, GMRES 残量范数 $\|r_k\|_2$ 递减, 但不一定严格递减. 在最坏的情况下, 可能会出现 $\|r_k\|_2$ 保持不变, 直到最后一步才降为 0.

例 1 考虑线性方程组 $Ax = b$, 其中

$$A = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

具体求解过程参见 MATLAB 程序 GMRES_Example01.m. 下图绘出了 $n = 64$ 时的 (相对) 残量范数下降曲线.


norm of relative residual
iteration number
0
0.2
0.4
0.6
0.8
1
0
10
20
30
40
50
60

第二个例子是关于偏微分方程的.

例 2 考虑下面的带齐次 Dirichlet 边界条件的偏微分方程:

$$
\begin{aligned}
&-\Delta u(x,y)+u_x(x,y)+u_y(x,y)+u(x,y)=f(x,y), \qquad (4.14)\\
&\quad (x,y)\in\Omega\triangleq[0,1]\times[0,1]\\
&\quad u(x,0)=u(x,1)=u(0,y)=u(1,y)=0, \quad 0<x,y<1.
\end{aligned}
$$

二阶导数用五点差分离散, 一阶导数用中心差分离散, 步长 $h_x = h_y = 1/(N+1)$. 可得具体离散格式为

$$
\begin{aligned}
&\frac{4u_{i,j}-u_{i+1,j}-u_{i-1,j}-u_{i,j+1}-u_{i,j-1}}{h^2}\\
&\qquad +\frac{u_{i+1,j}-u_{i-1,j}}{2h}+\frac{u_{i,j+1}-u_{i,j-1}}{2h}+u_{i,j}=f_{i,j}
\end{aligned}
$$

对应的离散线性方程组为

$$[I \otimes T + T \otimes I + h(I \otimes D) + h(D \otimes I) + h^2 I]u = h^2 f.$$

其中 $T = \text{tridiag}(-1, 2, -1)$, $D = \text{tridiag}(-1/2, 0, 1/2)$.

在实际测试中, 我们假设解析解为

$$u(x, y) = xy(1 - x)(1 - y).$$

由此可以求出右端项

$$f(x, y) = (3 - 2x)(1 - y)y + (3 - 2y)(1 - x)x + x(1 - x)y(1 - y).$$

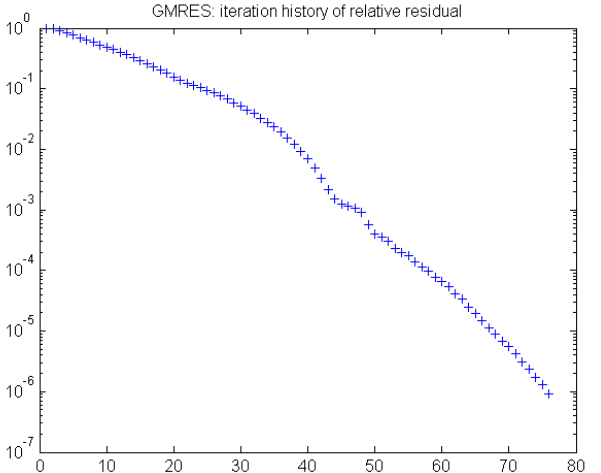
用 GMRES 方法求解. 最大迭代步数设为 IterMax $= N^2$, 迭代终止条

件

$$\frac{\|r_k\|_2}{\|r_0\|_2} \leq \text{tol} \triangleq 10^{-6}.$$

我们画出 $n = 32$ 时的相对残量下降曲线.

MATLAB 程序见 GMRES_Example02_PDE.m


GMRES: iteration history of relative residual
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
10^-6
10^-7
0
10
20
30
40
50
60
70
80

4 对称线性方程的 Krylov 子空间方法

4.1 Lanczos 过程

如果 A 是对称的, 则由 Arnoldi 过程的性质可知 $H_m = V_m^{\intercal} A V_m$, 因此 H_m 也对称. 又 H_m 是上 Hessenberg 矩阵, 所以 H_m 是一个对称三对角矩阵.

为了书写方便, 我们记 $\alpha_j = h_{j,j}$, $\beta_j = h_{j,j+1}$, 并将 H_m 改写为 T_k, 即

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

与 Arnoldi 过程类似, 我们有下面的性质

$$AV_k = V_k T_k + \beta_k v_{k+1} e_k^{\intercal}, \tag{4.15}$$
$$V_k^{\intercal} A V_k = T_k. \tag{4.16}$$

考察关系式 (4.15) 两边的第 j 列可知

$$\beta_j v_{j+1} = A v_j - \alpha_j v_j - \beta_{j-1} v_{j-1}, \quad j = 1, 2, \ldots,$$

这个三项递推公式是 CG 方法的核心, 它使得每一步迭代的运算量与迭代步数无关.

其中 $v_0 = 0$ 和 $\beta_0 = 0$.

基于这个三项递推公式, Arnoldi 过程可简化为 Lanczos 过程.

算法 4.1 Lanczos 过程

给定非零向量 r, **令** $v_0 = 0, \beta_0 = 0$
$v_1 = r/\|r\|_2$
for $j = 1, 2, \ldots$ **do**
    $w_j = Av_j - \beta_{j-1}v_{j-1}$
    $\alpha_j = (w_j, v_j)$
    $\tilde{w}_j = w_j - \alpha_j v_j$
    $\beta_j = \|\tilde{w}_j\|_2$
    if $\beta_j = 0$ **then**
        break
    end if
    $v_{j+1} = \tilde{w}_j/\beta_j$
end for

如果不考虑舍入误差, 则 Lanczos 过程生成的向量是相互正交的.

定理 9 设 $v_1, v_2, \ldots, v_k, \ldots$ 由 Lanczos 方法 4.1 生成, 则

$$(v_i, v_j) = \delta_{ij} \triangleq \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases} \qquad \text{for } i, j = 1, 2, \ldots. \tag{4.17}$$

(板书)

注记

该结论说明, 只需正交化相邻的三个向量, 因此只需存三个向量, 而且每次迭代的运算量也固定不变. 这就是 Lanczos 过程的优点.

实际计算中, 由于舍入误差影响, Lanczos 过程生成的向量可能会逐渐失去正交性. 这时需采取一些补救措施, 如全部重新正交化或部分重新正交化.

4.2 共轭梯度法 (CG)

首先考虑 $A \in \mathbb{R}^{n\times n}$ 是 **对称正定** 情形.

共轭梯度法 (Conjugate Gradient) 本质上与 FOM 是等价的, 即都是取 $\mathcal{L} = \mathcal{K}$. 但由于 A 对称, 因此可借助三项递推公式来简化运算.

CG 方法是当前求解对称正定线性方程组的首选方法.

方法描述

设 $x^{(0)}$ 是初始值, 在仿射空间 $x^{(0)} + \mathcal{K}_k$ 中寻找近似解 $x^{(k)}$, 满足

$$x^{(k)} \in x^{(0)} + \mathcal{K}_k \quad \text{且} \quad b - Ax^{(k)} \perp \mathcal{K}_k. \tag{4.18}$$

CG 方法就是根据这个性质来推导的.

方法推导

首先, 与 FOM 方法类似, 我们可得

$$x^{(k)} = x^{(0)} + V_k y_k, \qquad y_k = T_k^{-1}(\beta e_1^{(k)}), \tag{4.19}$$

其中 $V_k = [v_1, v_2, \ldots, v_k], \beta = \|r_0\|_2, e_1^{(k)} = [1, 0, \ldots, 0]^\intercal \in \mathbb{R}^k$.

因此需要求解一个以 T_k 为系数矩阵的线性方程组. 由于 A 对称正定, 故 T_k 也对称正定. 所以可以用 Cholesky 分解或 LDL$^\intercal$ 分解来求解.

设 T_k 的 LDL$^\intercal$ 分解为 $T_k = L_k D_k L_k^\intercal$. 于是可得

$$x^{(k)} = x^{(0)} + V_k y_k = x^{(0)} + V_k T_k^{-1}(\beta e_1^{(k)}) = x^{(0)} + (V_k L_k^{-\intercal})(\beta D_k^{-1} L_k^{-1} e_1^{(k)}).$$

如果 $x^{(k)}$ 满足精度要求, 则计算结束. 否则我们需要计算

$$x^{(k+1)} = x^{(0)} + V_{k+1} T_{k+1}^{-1}(\beta e_1^{(k+1)}) = x^{(0)} + (V_{k+1} L_{k+1}^{-\intercal})(\beta D_{k+1}^{-1} L_{k+1}^{-1} e_1^{(k+1)}),$$

这里假定 T_{k+1} 的 LDL$^\intercal$ 分解为 $T_{k+1} = L_{k+1}D_{k+1}L_{k+1}^\intercal$.

下面就考虑如何利用 **递推方式** 来计算 $x^{(k)}, k = 1, 2, \ldots$. 记

$$\tilde{P}_k \triangleq V_k L_k^{-\intercal} = [\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_k] \in \mathbb{R}^{n\times k},$$

$$\tilde{y}_k \triangleq \beta D_k^{-1} L_k^{-1} e_1^{(k)} = [\eta_1, \ldots, \eta_k]^\intercal \in \mathbb{R}^k.$$

我们首先证明下面的结论.

引理 2 下面的递推公式成立:

$$\tilde{P}_{k+1} \triangleq V_{k+1}L_{k+1}^{-\intercal} = [\tilde{P}_k, \tilde{p}_{k+1}],$$
$$y_{k+1} \triangleq \beta D_{k+1}^{-1} L_{k+1}^{-1} e_1^{(k+1)} = [\tilde{y}_k^\intercal, \eta_{k+1}]^\intercal, \quad k = 1, 2, \ldots.$$

(板书)

有了上面的性质, 我们就可以得到从 $x^{(k)}$ 到 $x^{(k+1)}$ 的递推公式

$$x^{(k+1)} = \tilde{P}_{k+1}\tilde{y}_{k+1} = [\tilde{P}_k, \tilde{p}_{k+1}] \begin{bmatrix} \tilde{y}_k \\ \eta_{k+1} \end{bmatrix} = x^{(k)} + \eta_{k+1}\tilde{p}_{k+1}. \tag{4.20}$$

为了判断近似解的精度, 我们还需要计算残量. 通过直接计算可知

$$r_{k+1} = b - Ax^{(k+1)} = b - A(x^{(k)} + \eta_{k+1}\tilde{p}_{k+1}) = r_k - \eta_{k+1}A\tilde{p}_{k+1}. \tag{4.21}$$

另一方面, 我们有

$$\begin{aligned} r_k = b - Ax^{(k)} = b - A(x^{(0)} + V_k y_k) &= r_0 - AV_k y_k \\ &= \beta v_1 - V_k T_k y_k - \beta_k v_{k+1} e_k^{\intercal} y_k \\ &= -\beta_k (e_k^{\intercal} y_k) v_{k+1}, \end{aligned}$$

即 r_k 与 v_{k+1} 平行. 记

$$r_k = \tau_k v_{k+1}, \quad k = 0, 1, 2, \ldots, \tag{4.22}$$

其中

$$\tau_0 = \beta = \|r_0\|_2, \quad \tau_k = -\beta_k (e_k^{\intercal} y_k), \quad k = 1, 2, \ldots.$$

定义

$$p_k = \tau_{k-1} \tilde{p}_k, \qquad k = 1, 2, \ldots.$$

所以 $\{p_k\}$ 满足下面的递推公式:

$$p_{k+1} = \tau_k \tilde{p}_{k+1} = \tau_k(v_{k+1} - l_k \tilde{p}_k) = r_k + \mu_k p_k, \tag{4.23}$$

$$\tilde{p}_{k+1} = -l_k \tilde{p}_k + v_{k+1}$$
$$r_k = \tau_k v_{k+1}$$

其中 $\mu_k = -\dfrac{l_k \tau_k}{\tau_{k-1}}, k = 1, 2, \ldots.$

于是我们就得到下面的递推公式

$$r_{k+1} = r_k - \eta_{k+1} A \tilde{p}_{k+1} = r_k - \xi_{k+1} A p_{k+1} \tag{4.24}$$
$$x^{(k+1)} = x^{(k)} + \eta_{k+1} \tilde{p}_{k+1} = x^{(k)} + \xi_{k+1} p_{k+1}, \tag{4.25}$$

其中 $\xi_{k+1} = \dfrac{\eta_{k+1}}{\tau_k}, k = 0, 1, 2, \ldots.$

系数 ξ_{k+1} 和 μ_k 的计算方法

引理 3 下面的结论成立:

(1) $r_1, r_2, \ldots, r_k, \ldots$ 相互正交;

(2) $p_1, p_2, \ldots, p_k, \ldots$ 相互 A 共轭 (或 A 正交), 即当 $i \neq j$ 时有 $p_i^{\mathsf{T}} A p_j = 0$.

(板书)

$$p_{k+1} = \tau_k \tilde{p}_{k+1} = \tau_k (v_{k+1} - l_k \tilde{p}_k) = r_k + \mu_k p_k, \tag{4.26}$$

$$r_{k+1} = r_k - \eta_{k+1} A\tilde{p}_{k+1} = r_k - \xi_{k+1} A p_{k+1} \tag{4.27}$$

$$x^{(k+1)} = x^{(k)} + \eta_{k+1}\tilde{p}_{k+1} = x^{(k)} + \xi_{k+1} p_{k+1}, \tag{4.28}$$

在等式 (4.26) 两边同时左乘 $p_{k+1}^{\intercal} A$ 可得

$$p_{k+1}^{\intercal} A p_{k+1} = p_{k+1}^{\intercal} A r_k + \mu_k p_{k+1}^{\intercal} A p_k = r_k^{\intercal} A p_{k+1}.$$

再用 $r_k^{\intercal}$ 左乘方程 (4.27) 可得

$$0 = r_k^{\intercal} r_{k+1} = r_k^{\intercal} r_k - \xi_{k+1} r_k^{\intercal} A p_{k+1},$$

故

$$\xi_{k+1} = \frac{r_k^{\intercal} r_k}{r_k^{\intercal} A p_{k+1}} = \frac{r_k^{\intercal} r_k}{p_{k+1}^{\intercal} A p_{k+1}}. \tag{4.29}$$

等式 (4.26) 两边同时左乘 $p_k^{\mathsf{T}}A$ 可得

$$0 = p_k^{\mathsf{T}}Ap_{k+1} = p_k^{\mathsf{T}}Ar_k + \mu_k p_k^{\mathsf{T}}Ap_k,$$

故

$$\mu_k = -\frac{p_k^{\mathsf{T}}Ar_k}{p_k^{\mathsf{T}}Ap_k} = -\frac{r_k^{\mathsf{T}}Ap_k}{p_k^{\mathsf{T}}Ap_k}. \tag{4.30}$$

为了进一步减少运算量, 我们将上式进行改写. 用 r_{k+1}^{T} 左乘方程 (4.27) 可得

$$r_{k+1}^{\mathsf{T}}r_{k+1} = r_{k+1}^{\mathsf{T}}r_k - \xi_{k+1}r_{k+1}Ap_{k+1} = -\xi_{k+1}r_{k+1}^{\mathsf{T}}Ap_{k+1},$$

故

$$\xi_{k+1} = -\frac{r_{k+1}^{\mathsf{T}}r_{k+1}}{r_{k+1}^{\mathsf{T}}Ap_{k+1}}.$$

所以 $\xi_k = -\dfrac{r_k^{\intercal} r_k}{r_k^{\intercal} A p_k}$, 即 $r_k^{\intercal} A p_k = -r_k^{\intercal} r_k / \xi_k$, 代入 (4.30) 可得

$$\mu_k = -\frac{r_k^{\intercal} A p_k}{p_k^{\intercal} A p_k} = \frac{r_k^{\intercal} r_k}{p_k^{\intercal} A p_k} \cdot \frac{1}{\xi_k} = \frac{r_k^{\intercal} r_k}{p_k^{\intercal} A p_k} \cdot \frac{p_k^{\intercal} A p_k}{r_{k-1}^{\intercal} r_{k-1}} = \frac{r_k^{\intercal} r_k}{r_{k-1}^{\intercal} r_{k-1}}. \tag{4.31}$$

综合公式 (4.26), (4.27), (4.28) 和 (4.29), (4.31) 即可得 CG 的迭代格式:

$$\begin{aligned}
&p_{k+1} = r_k + \mu_k p_k, \quad \text{其中} \quad \mu_k = \frac{r_k^{\intercal} r_k}{r_{k-1}^{\intercal} r_{k-1}}, \\
&x^{(k+1)} = x^{(k)} + \xi_{k+1} p_{k+1}, \\
&r_{k+1} = r_k - \xi_{k+1} A p_{k+1}, \quad \text{其中} \quad \xi_{k+1} = \frac{r_k^{\intercal} r_k}{p_{k+1}^{\intercal} A p_{k+1}}.
\end{aligned}$$

注意, 以上递推公式是从 $k = 1$ 开始. $k = 0$ 时的公式需要另外推导.

$k = 0$ 时的计算公式

首先, 由 $\tilde{p}_1$ 的定义可知

$$\tilde{p}_1 = \tilde{P}_1 = V_1 L_1^{-\intercal} = v_1.$$

因此

$$p_1 = \tau_0 \tilde{p}_1 = \beta v_1 = r_0.$$

其次, 由 Lanczos 过程可知 $T_1 = \alpha_1 = v_1^\intercal A v_1$. 注意到 $\beta^2 = r_0^\intercal r_0$, 于是

$$x^{(1)} = x^{(0)} + V_1 T_1^{-1}(\beta e_1) = x^{(0)} + \frac{\beta}{v_1^\intercal A v_1} v_1 = x^{(0)} + \frac{r_0^\intercal r_0}{p_1^\intercal A p_1} p_1.$$

令 $\xi_1 = \dfrac{r_0^\intercal r_0}{p_1^\intercal A p_1}$, 则当 $k = 0$ 时关于 $x^{(k+1)}$ 的递推公式仍然成立.

(注: 之前的 ξ_{k+1} 计算公式 (4.29) 只对 $k \geq 1$ 有定义)

最后考虑残量. 易知

$$r_1 = b - Ax^{(1)} = b - Ax^{(0)} - \frac{r_0^{\mathsf{T}} r_0}{p_1^{\mathsf{T}} A p_1} A p_1 = r_0 - \xi_1 A p_1,$$

即当 $k = 0$ 时关于 r_{k+1} 的递推公式也成立.

综合公式 (4.26), (4.27), (4.28) 和 (4.29), (4.31) 即可得 CG 方法:

算法 4.2 共轭梯度法 (CG)

1: 给定初值 $x^{(0)}$, (相对) 精度要求 $\varepsilon > 0$ 和最大迭代步数 IterMax
2: 计算 $r_0 = b - Ax^{(0)}$ 和 $\beta = \|r_0\|_2$
3: **for** $k = 1$ to IterMax **do**
4: $\quad \rho = r_{k-1}^{\mathsf{T}} r_{k-1}$
5: $\quad$ **if** $k > 1$ **then**

$\mu_{k-1} = \rho/\rho_0$

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

else

$p_k = r_0$

end if

$q_k = Ap_k$

$\xi_k = \rho/(p_k^{\intercal} q_k)$

$x^{(k)} = x^{(k-1)} + \xi_k p_k$

$r_k = r_{k-1} - \xi_k q_k$

relres $= \|r_k\|_2/\beta$

if relres$< \varepsilon$ **then**

break

end if

$\rho_0 = \rho$

end for

21: **if** relres$<\varepsilon$ **then**

22: 　　输出近似解 $x^{(k)}$ 及相关信息

23: **else**

24: 　　输出算法失败信息

25: **end if**

编程练习：CG 方法求解二维 poisson 方程.

注记

在优化领域, CG 方法中的向量 p_k 称为**搜索方向**, 而 ξ_k 称为**步长**. 因此, 搜索方法是 A 共轭的, 这就是方法名称的由来.

注记

在 CG 方法中, 需要额外存储四个向量 x, p, Ap 和 r, 每步迭代的主要运算为一个矩阵向量乘积和两个内积.

CG 方法也具有最优性质.

定理 10 设 A 对称正定, 则

$$x^{(k)} = \arg \min_{x \in x^{(0)} + \mathcal{K}_k} \|x - x_*\|_A \tag{4.32}$$

当且仅当

$$x^{(k)} \in x^{(0)} + \mathcal{K}_k \quad \text{且} \quad b - Ax^{(k)} \perp \mathcal{K}_k. \tag{4.33}$$

(板书)

与 GMRES 方法不同的是, CG 方法极小化的是绝对误差的 A 范数.

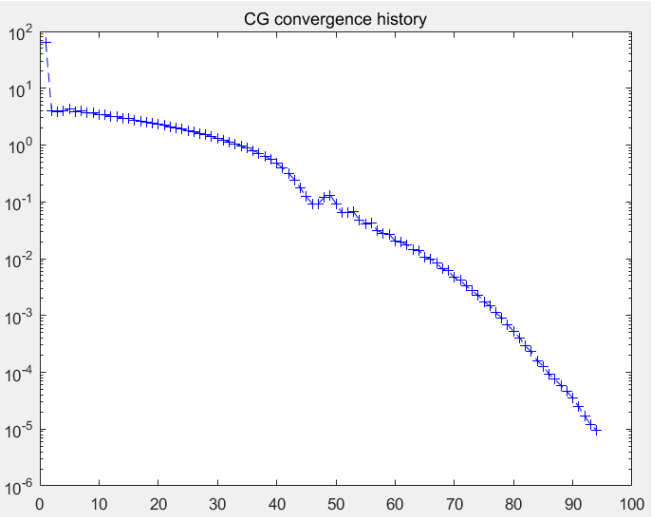
注记

如果 A 对称, 但不定, 此时 A 范数就没有定义, 因此最优性结论不再成立. 而从方法推导过程可知, 此时仍可以使用 CG 方法, 但由于 LDL^T 分解不一定存在, 因此方法可能会中断.

例 3 用 CG 方法求解线性方程组 $Ax = b$, 其中

$$A = B\otimes I + I\otimes B \in \mathbb{R}^{N^2\times N^2}, \quad B = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}_{N\times N}, \quad b = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{N^2\times 1}.$$

画出 $n = 64$ 时的相对残量下降曲线: Poisson_CG.m


CG convergence history
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0
10
20
30
40
50
60
70
80
90
100

To be continued ...

4.3 极小残量法 (MINRES)

这里假定 A 对称, 但不定. 此时 CG 方法不再适用.

与 GMRES 方法类似, 我们取约束空间为 $\mathcal{L}_k = A\mathcal{K}_k$, 即在仿射空间 $x^{(0)} + \mathcal{K}_k$ 中寻找近似解 $x^{(k)}$, 满足

$$x^{(k)} \in x^{(0)} + \mathcal{K}_k \quad \text{且} \quad b - Ax^{(k)} \perp A\mathcal{K}_k.$$

这就是 极小残量法 (MINRES). 它也可以看作是 GMRES 的对称情形, 即作用在对称方程组上的 GMRES 方法. 但由于 A 对称, 因此在计算 K_k 的正交基时, 可以采用 Lanczos 方法, 即三项递推方法, 从而节省运算量.

与 GMRES 方法类似, MINRES 具有下面的最优性质.

定理 11 设 $A \in \mathbb{R}^{n\times n}$ 是对称矩阵, $x^{(k)} \in x^{(0)} + \mathcal{K}_k$ 是 MINRES 方法迭代 k 步后得到的近似解, 则

$$x^{(k)} = \arg \min_{x\in x^{(0)}+\mathcal{K}_k} \|b - Ax\|_2,$$

即 $x^{(k)}$ 是残量在仿射空间 $x^{(0)} + \mathcal{K}_k$ 中的唯一最小值点.

由 GMRES 方法的推导过程可知, 当 MINRES 方法迭代 k 步之后, 近似解可表示为

$$x^{(k)} = x^{(0)} + V_k y_k \tag{4.34}$$

其中

$$y_k = \arg \min_{y\in\mathbb{R}^k} \|\beta e_1 - T_{k+1,k} y\|_2 \tag{4.35}$$

这里 $T_{k+1,k}$ 是由 Lanczos 方法生成的 $(k+1)\times k$ 的上 Hessenberg 矩阵

$$T_{k+1,k}=\begin{bmatrix}\alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \beta_{k-1} & \alpha_k \\ & & & \beta_k \end{bmatrix}.$$

与 CG 方法类似, 我们无需存储所有的基向量 v_i's. 事实上, $x^{(k)}$ 可以通过直接更新 $x^{(k-1)}$ 得到. 下面我们就给出推导过程.

最小二乘问题的求解

我们仍然用 QR 分解来求解最小二乘问题 (4.35). 设

$$T_{k+1,k}=Q_{k+1}R_{k+1,k}$$

是 $T_{k+1,k}$ 的 QR 分解, 其中 $Q_{k+1} \in \mathbb{R}^{(k+1)\times(k+1)}$ 是正交矩阵, $R_{k+1,k} \in \mathbb{R}^{(k+1)\times k}$ 是一个上三角矩阵.

由于 $T_{k+1,k}$ 是三对角的, 所以 $R_{k+1,k}$ 也只有三条对角线非零, 即

$$R_{k+1,k} = \begin{bmatrix} \tau_1^{(1)} & \tau_1^{(2)} & \tau_1^{(3)} & & & \\ & \tau_2^{(1)} & \tau_2^{(2)} & \tau_2^{(3)} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \tau_{k-2}^{(1)} & \tau_{k-2}^{(2)} & \tau_{k-2}^{(3)} \\ & & & & \tau_{k-1}^{(1)} & \tau_{k-1}^{(2)} \\ & & & & & \tau_k^{(1)} \\ 0 & & \cdots & & \cdots & 0 \end{bmatrix}_{(k+1)\times k} \triangleq \begin{bmatrix} R_k \\ 0 \end{bmatrix},$$

其中 R_k 是由 $R_{k+1,k}$ 的前 k 行组成的 k 阶矩阵. 于是, 我们有

$$\begin{aligned}\|\beta e_1 - T_{k+1,k}y\|_2 &= \|\beta e_1 - Q_{k+1}R_{k+1,k}\,y)\|_2 \\ &= \left\| Q_{k+1}\left(Q_{k+1}^{\mathsf{T}}(\beta e_1) - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y \right) \right\|_2 \\ &= \left\| [Q_{k+1,k},\ q_{k+1}]^{\mathsf{T}}\,(\beta e_1) - \begin{bmatrix} R_k y \\ 0 \end{bmatrix} \right\|_2, \end{aligned} \tag{4.36}$$

其中 $Q_{k+1,k}$ 是由 Q_{k+1} 的前 k 列组成的子矩阵, q_{k+1} 是 Q_{k+1} 的最后一列. 如果 R_k 非奇异, 则最小二乘问题 (4.35) 的解为

$$y_k = \beta R_k^{-1} Q_{k+1,k}^{\mathsf{T}} e_1.$$

因此,

$$x^{(k)} = x^{(0)} + \beta V_k R_k^{-1} Q_{k+1,k}^{\mathsf{T}} e_1. \tag{4.37}$$

$T_{k+1,k}$ 的 QR 分解的具体实现

首先, 由于 $T_{k+1,k}$ 是上 Hessenberg 矩阵, 因此可以采用 Givens 变换来做.

其次, 与 GMRES 类似, 我们可以借助递推方法来减少运算量, 即在 $T_{k,k-1}$ 的 QR 分解的基础上, 通过一次 Givens 变换得到 $T_{k+1,k}$ 的 QR 分解.

假定 $T_{k,k-1}$ 的 QR 分解为

$$T_{k,k-1} = (G_{k-1}G_{k-2}\cdots G_1)^\intercal R_{k,k-1} = Q_k R_{k,k-1},$$

其中 G_i 表示 Givens 变换, $Q_k = (G_{k-1}G_{k-2}\cdots G_1)^\intercal$, $R_{k,k-1}$ 是上三角

阵:

$$R_{k,k-1}=\begin{bmatrix}\tau_1^{(1)} & \tau_1^{(2)} & \tau_1^{(3)} & & & \\ & \tau_2^{(1)} & \tau_2^{(2)} & \tau_2^{(3)} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \tau_{k-3}^{(3)} \\ & & & & \ddots & \tau_{k-2}^{(2)} \\ & & & & & \tau_{k-1}^{(1)} \\ 0 & & \cdots & & \cdots & 0 \end{bmatrix}.$$

为了保证矩阵乘积的相容性, 我们假定 G_i 的维数是自动增长的, 即

$$G_i=\begin{bmatrix} I_{i-1} & & & \\ & c_i & s_i & \\ & -s_i & c_i & \\ & & & I_{k-i-1} \end{bmatrix}\in\mathbb{R}^{k\times k},\quad i=1,2,\dots k-1.$$

对于给定矩阵 B, 变换 $B \to G_i B$ 仅仅修改 B 的第 i 和第 $(i+1)$ 行. 因此,

$$T_{k+1,k} = \left[\begin{array}{c|c} & \begin{matrix}0\\ \vdots\\ 0\\ \beta_{k-1}\\ \alpha_k\end{matrix} \\ \hline T_{k,k-1} & \\ \hline \begin{matrix}0\end{matrix} & \begin{matrix}\beta_k\end{matrix} \end{array}\right] = \begin{bmatrix} Q_k & 0 \\ 0 & 1 \end{bmatrix} \left[\begin{array}{c|c} & Q_k^{-1}\begin{bmatrix}0\\ \vdots\\ 0\\ \beta_{k-1}\\ \alpha_k\end{bmatrix} \\ \hline R_{k,k-1} & \\ \hline 0 & \beta_k \end{array}\right] \triangleq \begin{bmatrix} Q_k & 0 \\ 0 & 1 \end{bmatrix} \tilde{T}_{k+1,k},$$

且

$$Q_k^{-1}\begin{bmatrix}0\\ \vdots\\ 0\\ 0\\ \beta_{k-1}\\ \alpha_k\end{bmatrix} = G_{k-1}G_{k-2}\cdots G_1\begin{bmatrix}0\\ \vdots\\ 0\\ \beta_{k-1}\\ \alpha_k\end{bmatrix} = G_{k-1}G_{k-2}\begin{bmatrix}0\\ \vdots\\ 0\\ 0\\ \beta_{k-1}\\ \alpha_k\end{bmatrix} = \begin{bmatrix}0\\ \vdots\\ 0\\ \tau_{k-2}^{(3)}\\ \tau_{k-1}^{(2)}\\ \tilde{\alpha}_k\end{bmatrix}.$$

构造 Givens 变换

$$G_k = \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & -s_k & c_k \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k+1)},$$

消去 $\tilde{T}_{k+1,k}$ 最后一列的最后一个元素 β_k, 即选取

$$c_k = \frac{\tilde{\alpha}_k}{\tau_k^{(1)}}, \quad s_k = \frac{\beta_k}{\tau_k^{(1)}} \quad \text{其中} \quad \tau_k^{(1)} = \sqrt{\tilde{\alpha}_k^2 + \beta_k^2}.$$

由于 $R_{k,k-1}$ 的最后一行全为 0, 且左乘 G_k 只会影响 $\tilde{T}_{k+1,k}$ 的最后两行, 因此

$$G_k\tilde{T}_{k+1,k} = \left[\begin{array}{c|c} & 0 \\ & \vdots \\ R_{k,k-1} & 0 \\ & \tau_{k-2}^{(3)} \\ & \tau_{k-1}^{(2)} \\ & \tau_k^{(1)} \\ \hline 0 & 0 \end{array}\right] = \begin{bmatrix} \tau_1^{(1)} & \tau_1^{(2)} & \tau_1^{(3)} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \tau_{k-3}^{(3)} & \\ & & & \ddots & \tau_{k-2}^{(2)} & \tau_{k-2}^{(3)} \\ & & & & \tau_{k-1}^{(1)} & \tau_{k-1}^{(2)} \\ & & & & & \tau_k^{(1)} \\ 0 & & \cdots & & \cdots & 0 \end{bmatrix} \triangleq R_{k+1,k}.$$

于是我们就得到 $T_{k+1,k}$ 的 QR 分解

$$T_{k+1,k} = Q_{k+1}R_{k+1,k},$$

其中 (这里我们假定 Givens 变换 G_i 的维数是自动增长的)

$$Q_{k+1} = \begin{bmatrix} Q_k & 0 \\ 0 & 1 \end{bmatrix} G_k^\intercal = (G_k G_{k-1} \dots G_1)^\intercal.$$

需要指出的是, $R_{k,k-1}$ 是 $R_{k+1,k}$ 的前 k 行和前 k 列. 因此, R_{k-1} 是 R_k

的 $(k-1)$ 阶顺序主子矩阵.

下面我们给出从 $x^{(k)}$ 到 $x^{(k+1)}$ 的递推公式. 设

$$P_k = [p_1, p_2, \ldots, p_k] \triangleq V_k R_k^{-1}.$$

通过直接计算, 由 $P_k R_k = V_k$ 可知

$$\begin{aligned} p_1 &= v_1/\tau_1^{(1)}, \\ p_2 &= \left(v_2 - \tau_1^{(2)} p_1\right)/\tau_2^{(1)}, \\ p_i &= \left(v_i - \tau_{i-1}^{(2)} p_{i-1} - \tau_{i-2}^{(3)} p_{i-2}\right)/\tau_i^{(1)}, \quad i = 3, 4, \ldots. \end{aligned} \tag{4.38}$$

又 $V_k = [V_{k-1}, v_k]$, 且 R_{k-1} 是 R_k 的 $(k-1)$ 阶顺序主子矩阵. 所以

$$P_k = [P_{k-1}, p_k].$$

记 $\beta Q_{k+1}^{\mathsf{T}} e_1$ 的前 k 个元素组成的向量为 $\xi^{(k)}$, 即

$$\beta Q_{k+1}^{\mathsf{T}} e_1 = \begin{bmatrix} \xi^{(k)} \\ \tilde{\xi}_{k+1} \end{bmatrix},$$

其中 $\tilde{\xi}_{k+1}$ 表示 $\beta Q_{k+1}^{\mathsf{T}} e_1$ 的最后一个元素. 易知 $Q_{k+1,k}^{\mathsf{T}} e_1$ 和 $Q_{k+1}^{\mathsf{T}} e_1$ 的前 k 个分量是一样的, 即 $\xi^{(k)} = \beta Q_{k+1,k}^{\mathsf{T}} e_1$. 因此

$$y^{(k)} = \beta R_k^{-1} Q_{k+1,k}^{\mathsf{T}} e_1 = R_k^{-1} \xi^{(k)}. \tag{4.39}$$

又

$$Q_{k+1} = \begin{bmatrix} Q_k & 0 \\ 0 & 1 \end{bmatrix} G_k^{\mathsf{T}},$$

所以, $Q_{k+1}(1, 1 : k-1) = Q_k(1, 1 : k-1)$, 即 $Q_{k+1}^{\mathsf{T}} e_1$ 和 $Q_k^{\mathsf{T}} e_1$ 的前 $k-1$ 个分量是一样的. 于是我们可以得到 $\xi^{(k)}$ 和 $\xi^{(k-1)}$ 之间的关系

式:

$$\xi^{(k)} = \begin{bmatrix} \xi^{(k-1)} \\ \xi_k \end{bmatrix}, \quad k = 2, 3, \ldots,$$

其中 ξ_k 表示 $\xi^{(k)}$ 的最后一个分量. 因此, 对所有正整数 k, $\xi^{(k)}$ 可以表示为

$$\xi^{(k)} = [\xi_1, \xi_2, \ldots, \xi_k]^\intercal, \quad k = 1, 2, \ldots.$$

根据 (4.37), 我们有

$$\begin{aligned} x^{(k)} &= x^{(0)} + V_k R_k^{-1} Q_{k+1,k}^{\intercal}(\beta e_1) \\ &= x^{(0)} + P_k \xi^{(k)} \\ &= x^{(0)} + [P_{k-1}, p_k] \begin{bmatrix} \xi^{(k-1)} \\ \xi_k \end{bmatrix} \\ &= x^{(0)} + P_{k-1}\xi^{(k-1)} + \xi_k p_k \\ &= \boxed{x^{(k-1)} + \xi_k p_k}, \end{aligned} \tag{4.40}$$

其中 p_k 可通过递推公式 (4.38) 来计算, 而 ξ_k 是 $\beta Q_{k+1}^{\intercal} e_1$ 的第 k 个分量. 又

$$\beta Q_{k+1}^{\intercal} e_1 = Q_{k+1}^{\intercal}(\beta e_1) = G_k \tilde{G}_{k-1} \cdots \tilde{G}_1(\beta e_1),$$

即向量 $\beta Q_{k+1}^{\intercal} e_1$ 可以通过将 Givens 变换依次作用在 βe_1 上得到.

下面考虑残量的计算. 由 (4.39) 可知

$$
\begin{aligned}
\|r_k\|_2 = \|b - Ax^{(k)}\|_2 &= \|r_0 - AV_k y_k\|_2 \\
&= \|\beta v_1 - V_{k+1} T_{k+1,k}\, y_k\|_2 \\
&= \|V_{k+1}(\beta e_1 - Q_{k+1} R_{k+1,k}\, y_k)\|_2 \\
&= \left\| Q_{k+1} \left(Q_{k+1}^{\intercal}(\beta e_1) - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y_k \right) \right\|_2 \\
&= \left\| \begin{bmatrix} \xi^{(k)} \\ \tilde{\xi}_{k+1} \end{bmatrix} - \begin{bmatrix} R_k y_k \\ 0 \end{bmatrix} \right\|_2 \\
&= |\tilde{\xi}_{k+1}|,
\end{aligned}
$$

其中 $\tilde{\xi}_{k+1}$ 是 $Q_{k+1}^{\intercal}(\beta e_1) = G_k G_{k-1} \cdots G_1(\beta e_1)$ 的最后一个分量.

综上所述, 根据 p_k 的递推公式 (4.38) 和 $x^{(k)}$ 的递推公式 (4.40), 我们就可以构造下面的 MINRES 方法.

算法 4.3 Minimal Residual (MINRES) Method

1: **给定初值** $x^{(0)}$, **(相对) 精度要求** $\varepsilon > 0$ **和最大迭代步数** IterMax
2: **计算** $r_0 = b - Ax^{(0)}$ **和** $\beta = \|r_0\|_2$
3: **令** $\beta_0 = 0, v_0 = 0, p_{-1} = 0, p_0 = 0$
4: $v_1 = r_0/\beta, \quad \xi = \beta e_1$
5: **for** $k = 1, 2, \ldots, n$ **do**
6: $\quad w_k = Av_k - \beta_{k-1}v_{k-1}$
7: $\quad \alpha_k = (w_k, v_k)$
8: $\quad w_k = w_k - \alpha_k v_k$
9: $\quad \beta_k = \|w_k\|_2$
10: $\quad$ **if** $k > 2$ **then** % apply $G_{k-1}G_{k-2}$ to the last column of $T_{k+1,k}$
11: $\quad\quad \begin{bmatrix} \tau_{k-2}^{(3)} \\ \tilde{\beta}_{k-1} \end{bmatrix} = \begin{bmatrix} c_{k-2} & s_{k-2} \\ -s_{k-2} & c_{k-2} \end{bmatrix} \begin{bmatrix} 0 \\ \beta_{k-1} \end{bmatrix}$
12: $\quad$ **end if**

if $k > 1$ **then**

$$\begin{bmatrix} \tau_{k-1}^{(2)} \\ \tilde{\alpha}_k \end{bmatrix} = \begin{bmatrix} c_{k-1} & s_{k-1} \\ -s_{k-1} & c_{k-1} \end{bmatrix} \begin{bmatrix} \tilde{\beta}_{k-1} \\ \alpha_k \end{bmatrix}$$

end if

if $|\tilde{\alpha}_k| > |\beta_k|$ **then** % form the Givens rotation G_k

set $c_k = \frac{1}{\sqrt{1+\gamma^2}}$, $s_k = c_k\gamma$ where $\gamma = \beta_k/\tilde{\alpha}_k$

else

set $s_k = \frac{1}{\sqrt{1+\gamma^2}}$, $c_k = s_k\gamma$ where $\gamma = \alpha_k/\tilde{\beta}_k$

end if

$\tau_k^{(1)} = c_k\tilde{\alpha}_k + s_j\beta_k$ % apply G_k to last column of $\tilde{T}_{k+1,k}$

$$\begin{bmatrix} \xi_k \\ \xi_{k+1} \end{bmatrix} = \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \xi_k \\ 0 \end{bmatrix}$$ % apply G_k to ξ

$p_k = \left(v_k - \tau_{k-1}^{(2)} p_{k-1} - \tau_{k-2}^{(3)} p_{k-2}\right) / \tau_k^{(1)}$ where $p_0 = p_{-1} = 0$

$x^{(k)} = x^{(k-1)} + \xi_k p_k$

```
    relres = |ξ_{k+1}|/β
    if relres< ε then    % check convergence
        break
    end if
    v_{k+1} = w_k/β_k
end for
if relres < ε then
    输出近似解 x^{(k)} 及相关信息
else
    输出算法失败信息
end if
```

与 GMRES 相比, MINRES 方法的优点是

- MINRES 充分利用了系数矩阵的对称性, 这使得每步迭代的运算量不会随着迭代步数的增加而增加;
- 在迭代过程中, 我们不需要存储所有的 v_i 和 p_i, 因此存储量也与迭代步数无关.

注记

CG 方法也可以用来求解对称不定线性方程组, 但可能会产生中断.

在 MINRES 方法中, 我们用了三项递推公式, 好处是可以有效节省运算量和存储量, 但缺点是, 随着迭代步数的增加, 舍入误差的影响会越来越大.

Sleijpen, van der Vorst 和 Modersitzki 曾经指出, 在 MINRES 中, 舍入误差是按 $\kappa(A)^2$ 的速度增长的. 而在 GMRES 中, 舍入误差只按 $\kappa(A)$ 的

速度增长. 因此, 对于坏条件问题, 在使用 MINRES 需要加倍小心, 特别是迭代步数比较大的情况下. 此时, 我们也可以选择 SYMMLQ 方法.

4.4 SYMMLQ 方法

在 SYMMLQ 方法中, 我们取 $\mathcal{L}_k = \mathcal{K}_k$, 即在仿射空间 $x^{(0)} + \mathcal{K}_k$ 中寻找近似解 $x^{(k)}$, 满足

$$x^{(k)} \in x^{(0)} + \mathcal{K}_k \quad \text{且} \quad b - Ax^{(k)} \perp \mathcal{K}_k.$$

SYMMLQ (Symmetric LQ) 方法的收敛速度可能不如 MINRES, 但对于坏条件问题, SYMMLQ 通常比 MINRES 更稳定.

注记

当 A 对称正定时, SYMMLQ 方法与 CG 方法是等价的.

设 $x^{(k)}$ 是 SYMMLQ 在 $x^{(0)} + \mathcal{K}_k$ 中找到的近似解, 则 $x^{(k)}$ 可表示为

$$x^{(k)} = x^{(0)} + V_k y_k \tag{4.41}$$

其中 y_k 是下面方程的解:

$$T_k y = \beta e_1 \tag{4.42}$$

如果 A 是不定的, 则 T_k 也可能是不定的, 因此就不能用 LDL$^\intercal$ 分解来解方程组 (4.42). 此时, 我们采用 LQ 分解 (类似于 QR 分解), 即

$$T_k = \tilde{L}_k Q_k \tag{4.43}$$

其中 $\tilde{L}_k \in \mathbb{R}^{k\times k}$ 是下三角矩阵, $Q_k \in \mathbb{R}^{k\times k}$ 是正交矩阵. 于是

$$x^{(k)} = x^{(0)} + V_k T_k^{-1}(\beta e_1) = x^{(0)} + V_k Q_k^\intercal \tilde{L}_k^{-1}(\beta e_1) = x^{(0)} + \tilde{P}_k \tilde{z}^{(k)},$$

其中

$$\tilde{P}_k \triangleq V_k Q_k^{\intercal} \in \mathbb{R}^{n\times k}, \quad \tilde{z}^{(k)} \triangleq \tilde{L}_k^{-1}(\beta e_1) \in \mathbb{R}^k.$$

由于 T_k 是三对角矩阵, 因此 T_k 的 LQ 分解可以通过一系列的 Givens 变换来说实现. 同样地, T_{k+1} 的 LQ 分解可以在 T_k 的 LQ 分解的基础上, 通过一次 Givens 变换得到. 下面给出具体推导过程.

$k = 1$ 时

此时 $T_k = \alpha_1$, 无需做任何分解,

$k = 2$ 时

当 $k = 2$ 时, $T_k = \begin{bmatrix} \alpha_1 & \beta_1 \\ \beta_1 & \alpha_2 \end{bmatrix}$. 只需一次 Givens 变化即可得到 T_k 的 LQ 分解: $T_k = \tilde{L}_2 G_1^{\intercal}$.

$T_k \to T_{k+1}$

设 T_k 的 LQ 分解为

$$T_k = \tilde{L}_k (G_1 G_2 \cdots G_{k-1})^\intercal \triangleq \tilde{L}_k Q_k,$$

其中 $Q_k = (G_1 G_2 \cdots G_{k-1})^\intercal$. 这里 G_i 是 Givens 变换:

$$G_i = \begin{bmatrix} I_{i-1} & & & \\ & c_i & s_i & \\ & -s_i & c_i & \\ & & & I_{k-i-1} \end{bmatrix} \in \mathbb{R}^{k\times k}, \quad i = 1, 2, \ldots, k-1.$$

由于 T_k 是三对角, 因此 $\tilde{L}_k = T_k Q_k^{\mathsf{T}}$ 只有三条对角线非零, 即

$$\tilde{L}_k = \begin{bmatrix} l_1^{(1)} & & & & & & \\ l_2^{(2)} & l_2^{(1)} & & & & & \\ l_3^{(3)} & l_3^{(2)} & l_3^{(1)} & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & l_{k-1}^{(3)} & l_{k-1}^{(2)} & l_{k-1}^{(1)} & & \\ & & & l_k^{(3)} & l_k^{(2)} & \tilde{l}_k^{(1)} \end{bmatrix}.$$

注记

这里最右下角的元素上面有个波浪号, 是用于区分 $\tilde{L}_k$ 和 L_k, 其中 L_k 表示 $\tilde{L}_{k+1}$ 的 k 阶顺序主子矩阵. 即 $\tilde{L}_k$ 并不是 $\tilde{L}_{k+1}$ 的子矩阵.

令

$$\tilde{Q}_{k+1} \triangleq \begin{bmatrix} Q_k & 0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k+1)},$$

则

$$T_{k+1}\tilde{Q}_{k+1}^{\intercal} = \left[\begin{array}{c|c} T_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \beta_k \end{matrix} \\ \hline 0 \cdots 0\ \beta_k & \alpha_{k+1} \end{array}\right] \begin{bmatrix} Q_k^{\intercal} & 0 \\ 0 & 1 \end{bmatrix} = \left[\begin{array}{c|c} \tilde{L}_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \beta_k \end{matrix} \\ \hline 0 \cdots 0\ l_{k+1}^{(3)}\ \tilde{\beta}_k & \alpha_{k+1} \end{array}\right],$$

其中

$$\begin{bmatrix} 0 \cdots 0\ l_{k+1}^{(3)}\ \tilde{\beta}_k \end{bmatrix} = \begin{bmatrix} 0 \cdots 0\ \beta_k \end{bmatrix} \cdot Q_k^{\intercal} = \begin{bmatrix} 0 \cdots 0\ \beta_k \end{bmatrix} \cdot G_{k-1}.$$

即 $l_{k+1}^{(3)} = -s_{k-1}\beta_k, \tilde{\beta}_k = c_{k-1}\beta_k$.

构造 Givens 变换 G_k, 消去 $T_{k+1}\tilde{Q}_{k+1}^{\intercal}$ 倒数第二行的最后一个分量 β_k,

即

$$G_k^{\mathsf{T}} = \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & -s_k & c_k \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k+1)},$$

其中

$$c_k = \frac{\tilde{l}_k^{(1)}}{l_k^{(1)}}, \quad s_k = \frac{\beta_k}{l_k^{(1)}}, \quad l_k^{(1)} = \sqrt{\left(\tilde{l}_k^{(1)}\right)^2 + \beta_k^2}.$$

令 $Q_{k+1} = G_k^{\mathsf{T}} \tilde{Q}_{k+1}$. 由于右乘 G_k 时只会影响到最后两列的元素, 所以

$$T_{k+1}Q_{k+1}^{\mathsf{T}} = \left[\begin{array}{ccc|c} & \tilde{L}_k & & \begin{matrix}0\\ \vdots \\ 0 \\ \beta_k\end{matrix} \\ \hline 0 & \cdots & 0\ l_{k+1}^{(3)}\ \tilde{\beta}_k & \alpha_{k+1} \end{array}\right] G_k = \left[\begin{array}{ccccccc|c} l_1^{(1)} & & & & & & & 0 \\ l_2^{(2)} & l_2^{(1)} & & & & & & \vdots \\ l_3^{(3)} & l_3^{(2)} & l_3^{(1)} & & & & & \vdots \\ & \ddots & \ddots & \ddots & & & & 0 \\ & & l_k^{(3)} & l_k^{(2)} & l_k^{(1)} & & & 0 \\ \hline 0 & \cdots & 0 & l_{k+1}^{(3)} & l_{k+1}^{(2)} & & & \tilde{l}_{k+1}^{(1)} \end{array}\right] \triangleq \tilde{L}_{k+1}.$$

于是, 我们就得到 T_{k+1} 的 LQ 分解

$$T_{k+1} = \tilde{L}_{k+1} Q_{k+1}.$$

记 $\tilde{L}_k$ 的 $k-1$ 阶顺序主子矩阵为 L_{k-1}, $\tilde{L}_{k+1}$ 的 k 阶顺序主子矩阵为 L_k. 由于 L_k 和 $\tilde{L}_k$ 只有第 (k,k) 位置上的元素不同, 因此 L_{k-1} 也是 L_k 的 $k-1$ 阶顺序主子矩阵. 于是我们就得到下面的递推关系:

L_{j-1} 是 L_j 的 $j-1$ 阶顺序主子矩阵, $j=1,2,....$

令 $z^{(k)}=L_k^{-1}(\beta e_1)\in\mathbb{R}^k$. 由于 $\tilde{z}^{(k)}=\tilde{L}_k^{-1}(\beta e_1)$, 所以

$$z^{(k)}=\begin{bmatrix} z^{(k-1)} \\ z_k \end{bmatrix}, \quad \tilde{z}^{(k)}=\begin{bmatrix} z^{(k-1)} \\ \tilde{z}_k \end{bmatrix},$$

其中 $z^{(k-1)}=L_{k-1}^{-1}(\beta e_1)$, z_k 和 $\tilde{z}_k$ 分别表示 $z^{(k)}$ 和 $\tilde{z}^{(k)}$ 的最后一个元素. 因此, 我们可以将 $z^{(k)}$ 和 $\tilde{z}^{(k)}$ 写为

$$z^{(k)}=[z_1,\ldots,z_{k-1},z_k]^{\mathsf{T}}, \quad \tilde{z}^{(k)}=[z_1,\ldots,z_{k-1},\tilde{z}_k]^{\mathsf{T}}, \quad k=1,2,\ldots,$$

其中

$$\tilde{z}_k = \frac{l_k^{(1)} z_k}{\tilde{l}_k^{(1)}}.$$

由 $L_k z^{(k)} = \beta e_1$ 可知

$$\begin{cases} z_1 = \beta / l_1^{(1)}, \\ z_2 = -l_2^{(2)} z_1 / l_2^{(1)}, \\ z_i = -\left(l_i^{(3)} z_{i-2} + l_i^{(2)} z_{i-1} \right) / l_i^{(1)}, \quad i = 3, 4, \ldots, k. \end{cases} \tag{4.44}$$

分别记 $\tilde{P}_k = V_k Q_k^{\intercal}$ 和 $\tilde{P}_{k+1} = V_{k+1} Q_{k+1}^{\intercal}$ 的前 $k-1$ 列和前 k 列组成的矩阵为 P_{k-1} 和 P_k, 则

$$
\begin{aligned}
\tilde{P}_{k+1} = V_{k+1} Q_{k+1}^{\intercal} &= [V_k, v_{k+1}] \begin{bmatrix} Q_k^{\intercal} & 0 \\ 0 & 1 \end{bmatrix} G_k \\
&= \left[\tilde{P}_k, v_{k+1}\right] G_k \\
&= [P_{k-1}, \tilde{p}_k, v_{k+1}] \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & -s_k & c_k \end{bmatrix} \qquad (4.45) \\
&= [P_{k-1}, p_k, \tilde{p}_{k+1}] \\
&= [P_k, \tilde{p}_{k+1}] \,,
\end{aligned}
$$

其中 $\tilde{p}_k$ 和 $\tilde{p}_{k+1}$ 分别表示 $\tilde{P}_k$ 和 $\tilde{P}_{k+1}$ 的最后一列. 因此, P_k 和 P_{k-1} 有下面的递推关系

$$
P_k = [P_{k-1}, p_k],
$$

其中 p_k 表示 P_k 的最后一列. 所以, 我们可以将 P_k 统一写为

$$P_k = [p_1, p_2, \ldots, p_k], \quad k = 1, 2, \ldots.$$

易知 $\tilde{p}_1 = \tilde{P}_1 = v_1$. 由 (4.45) 可知

$$\begin{cases} \tilde{p}_1 = v_1, \\ p_k = c_k \tilde{p}_k - s_k v_{k+1}, \\ \tilde{p}_{k+1} = s_k \tilde{p}_k + c_k v_{k+1}, \quad k = 1, 2, \ldots. \end{cases} \tag{4.46}$$

令 $\tilde{x}^{(k)} = x^{(0)} + P_k z^{(k)}$, $k = 1, 2, \ldots$, 则

$$\tilde{x}^{(k)} = x^{(0)} + P_k z^{(k)} = x^{(0)} + [P_{k-1}, p_k] \begin{bmatrix} z^{(k-1)} \\ z_k \end{bmatrix} = \tilde{x}^{(k-1)} + z_k p_k, \quad k = 1, 2, \ldots. \tag{4.47}$$

于是有

$$\begin{aligned} x^{(k+1)} = x^{(0)} + \tilde{P}_{k+1}\tilde{z}^{(k+1)} &= x^{(0)} + [P_k, \tilde{p}_{k+1}] \begin{bmatrix} z^{(k)} \\ \tilde{z}_{k+1} \end{bmatrix} \\ &= x^{(0)} + P_k z^{(k)} + \tilde{z}_{k+1}\tilde{p}_{k+1} \\ &= \tilde{x}^{(k)} + \tilde{z}_{k+1}\tilde{p}_{k+1}. \end{aligned} \tag{4.48}$$

有了这个关系式后, 我们在实际计算中只需计算 $\tilde{x}^{(k)}$.

由定理 5 可知

$$r_k = b - Ax^{(k)} = -\beta_k(e_k^{\mathsf{T}} y_k)v_{k+1}. \tag{4.49}$$

因此, 为了计算残量的值, 我们需要计算 y_k, 即解方程 (4.42). 但事实上, 我们只需知道 y_k 最后一个分量即可, 无需把整个 y_k 都计算出来. 我们注意到 T_k 是对称的, 因此 $T_k^{\mathsf{T}} y_k = T_k y_k = \beta e_1$, 所以

$$\tilde{L}_k^{\mathsf{T}} y_k = Q_k(\beta e_1).$$

观察上述等式两边的最后一个元素可知

$$\begin{aligned}\tilde{l}_k^{(1)} e_k^\intercal y_k = e_k^\intercal \tilde{L}_k^\intercal y_k &= e_k^\intercal Q_k(\beta e_1) \\ &= \beta e_k^\intercal (G_1 G_2 \cdots G_{k-1})^\intercal e_1 \\ &= \beta (G_1 G_2 \cdots G_{k-1} e_k)^\intercal e_1 \\ &= \beta s_1 s_2 \cdots s_{k-1}.\end{aligned}$$

由 Givens 变换 G_k 的具体构造公式可知, $s_k \tilde{l}_k^{(1)} + c_k \beta_k = 0$. 所以

$$\begin{aligned}r_k = -\beta_k (e_k^\intercal y_k) v_{k+1} &= -\left(\beta s_1 s_2 \cdots s_{k-1} \beta_k / \tilde{l}_k^{(1)}\right) v_{k+1} \\ &= (\beta s_1 s_2 \cdots s_{k-1} s_k / c_k) v_{k+1},\end{aligned}$$

即

$$\|r_k\|_2 = |\beta s_1 s_2 \cdots s_{k-1} s_k / c_k| = |c_{k-1} s_k / c_k| \cdot \|r_{k-1}\|_2.$$

算法 4.4 SYMMLQ 方法

1: **给定初值** $x^{(0)}$ **和 (相对) 精度要求** $\varepsilon > 0$
2: **计算** $r_0 = b - Ax^{(0)}$ **和** $\beta = \|r_0\|_2$
3: $v_1 = r_0/\beta, p_1 = v_1, \xi_0 = \beta, \tilde{x}^{(0)} = x^{(0)}$
4: **for** $k = 1, 2, \ldots,$ **do**
5: $w_k = Av_k - \beta_{k-1}v_{k-1}$ where $\beta_0 = 0$ and $v_0 = 0$
6: $\alpha_k = (w_k, v_k)$
7: $\tilde{w}_k = w_k - \alpha_k v_k$
8: $\beta_k = \|\tilde{w}_k\|_2$
9: **if** $k = 1$ **then**
10: $\tilde{l}_k^{(1)} = \alpha_k$
11: **end if**
12: **if** $k = 2$ **then**
13: $\tilde{\beta}_{k-1} = \beta_{k-1}$
14: **end if**

15: **if** $k > 2$ **then** % apply G_{k-2} to the last row of T_k

16: $\begin{bmatrix} l_k^{(3)} & \tilde{\beta}_{k-1} \end{bmatrix} = \begin{bmatrix} 0 & \beta_{k-1} \end{bmatrix} \begin{bmatrix} c_{k-2} & s_{k-2} \\ -s_{k-2} & c_{k-2} \end{bmatrix}$

17: **end if**

18: **if** $k > 1$ **then** % form the Givens rotation G_{k-1}

19: **if** $|\tilde{l}_{k-1}^{(1)}| > |\beta_{k-1}|$ **then**

20: set $c_{k-1} = \frac{1}{\sqrt{1+\gamma^2}}$, $s_{k-1} = -c_{k-1}\gamma$ where $\gamma = \beta_{k-1}/\tilde{l}_{k-1}^{(1)}$

21: **else**

22: set $s_{k-1} = \frac{1}{\sqrt{1+\gamma^2}}$, $c_{k-1} = -s_{k-1}\gamma$ where $\gamma = \tilde{l}_{k-1}^{(1)}/\beta_{k-1}$

23: **end if**

24: **end if**

25: **if** $k > 1$ **then** % apply G_{k-1} to the last two columns of $T_k\tilde{Q}_{k-1}$

26: $l_{k-1}^{(1)} = \sqrt{\left(\tilde{l}_{k-1}^{(1)}\right)^2 + \beta_{k-1}^2}$

$$\begin{bmatrix} l_k^{(2)} & \tilde{l}_k^{(1)} \end{bmatrix} = \begin{bmatrix} \tilde{\beta}_{k-1} & \alpha_k \end{bmatrix} \begin{bmatrix} c_{k-1} & s_{k-1} \\ -s_{k-1} & c_{k-1} \end{bmatrix}$$

end if

% compute z_k

if $k = 2$ **then**

$z_1 = \beta / l_1^{(1)}$

end if

if $k = 3$ **then**

$z_2 = -l_2^{(2)} z_1 / l_2^{(1)}$

end if

if $k > 3$ **then**

$$z_{k-1} = -\left(l_{k-1}^{(3)} z_{k-3} + l_{k-1}^{(2)} z_{k-2} \right) / l_{k-1}^{(1)}$$

end if

% compute p_k

if $k = 1$ **then**

41: $\tilde{p}_1 = v_1$

42: **end if**

43: **if** $k > 1$ **then**

44: $p_{k-1} = c_{k-1}\tilde{p}_{k-1} - s_{k-1}v_k$

45: $\tilde{p}_k = s_{k-1}\tilde{p}_{k-1} + c_{k-1}v_k$

46: **end if**

47: % update $\tilde{x}^{(k)}$

48: **if** $k > 1$ **then**

49: $\tilde{x}^{(k-1)} = \tilde{x}^{(k-2)} + z_{k-1}p_{k-1}$

50: **end if**

51: % check convergence

52: **if** $k > 1$ **then**

53: $\xi_{k-1} = (c_{k-2}s_{k-1}/c_{k-1})\,\xi_{k-2}$

54: **if** $|\xi_{k-1}| < \varepsilon$ **then**

55: $x^{(k)} = \tilde{x}_{k-1} + \left(l_k^{(1)} z_k / \tilde{l}_k^{(1)}\right) \tilde{p}_k$

```
            stop
        end if
    end if
    v_{k+1} = w̃_k / β_k
end for
```

注记

由 (4.49) 可知, 在 SYMMLQ 方法中, 残量是相互正交的.

注记

如果 A 是对称正定的, 则 SYMMLQ 与 CG 等价, 此时 SYMMLQ 极小化 $\|x^{(k)} - x_*\|_A$. 如果 A 是不定的, 则没有这个最优性质. 但可以证明, SYMMLQ 在仿射空间 $x^{(0)} + A\mathcal{K}_k(A, r_0)$ 中极小化 $\|x^{(k)} - x_*\|_2$, 参见 [? ?].